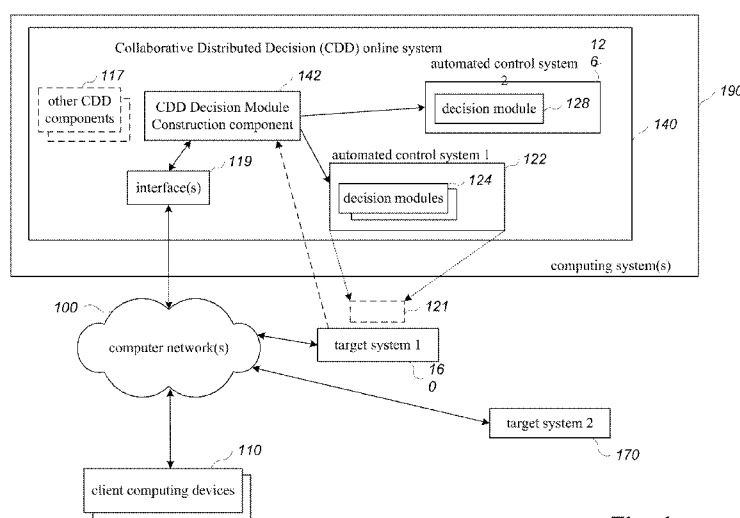




- (51) **International Patent Classification:**
G05B 19/418 (2006.01)
- (21) **International Application Number:**
PCT/US2015/037022
- (22) **International Filing Date:**
22 June 2015 (22.06.2015)
- (25) **Filing Language:** English
- (26) **Publication Language:** English
- (30) **Priority Data:**
62/015,018 20 June 2014 (20.06.2014) US
- (71) **Applicant:** ATIGEO CORP. [US/US]; 800 Bellevue Way NE, Suite 600, Bellevue, Washington 98004 (US).
- (72) **Inventors:** KOHN, Wolf; 800 Bellevue Way NE, Suite 600, Bellevue, Washington 98004 (US). SANDOVAL, Michael, Luis; 800 Bellevue Way NE, Suite 600, Bellevue, Washington 98004 (US). VETTRIVEL, Vishnu; 800 Bellevue Way NE, Suite 600, Bellevue, Washington 98004 (US). CROSS, Jonathan; 800 Bellevue Way NE, Suite 600, Bellevue, Washington 98004 (US). KNOX, Jason; 800 Bellevue Way NE, Suite 600, Bellevue, Washington 98004 (US). TALBY, David; 800 Bellevue Way NE, Suite 600, Bellevue, Washington 98004 (US). LAZARUS, Mike; 800 Bellevue Way NE, Suite 600, Bellevue, Washington 98004 (US).
- (74) **Agents:** WHITE, James, A. D. et al.; Seed Intellectual Property Law Group PLLC, Suite 5400, 701 Fifth Avenue, Seattle, Washington 98104-7064 (US).
- (81) **Designated States** (*unless otherwise indicated, for every kind of national protection available*): AE, AG, AL, AM, AO, AT, AU, AZ, BA, BB, BG, BH, BN, BR, BW, BY, BZ, CA, CH, CL, CN, CO, CR, CU, CZ, DE, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IR, IS, JP, KE, KG, KN, KP, KR, KZ, LA, LC, LK, LR, LS, LU, LY, MA, MD, ME, MG, MK, MN, MW, MX, MY, MZ, NA, NG, NI, NO, NZ, OM, PA, PE, PG, PH, PL, PT, QA, RO, RS, RU, RW, SA, SC, SD, SE, SG, SK, SL, SM, ST, SV, SY, TH, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, ZA, ZM, ZW.
- (84) **Designated States** (*unless otherwise indicated, for every kind of regional protection available*): ARIPO (BW, GH, GM, KE, LR, LS, MW, MZ, NA, RW, SD, SL, ST, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, RU, TJ, TM), European (AL, AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HR, HU, IE, IS, IT, LI, LU, LV, MC, MK, MT, NL, NO, PL, PT, RO, RS, SE, SI, SK, SM, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, KM, ML, MR, NE, SN, TD, TG).
- Published:**
— with international search report (Art. 21(3))

(54) **Title:** COOPERATIVE DISTRIBUTED CONTROL OF TARGET SYSTEMS**Fig. 1**

(57) **Abstract:** Techniques are described for implementing automated control systems that manipulate operations of specified target systems, such as by modifying or otherwise manipulating inputs or other control elements of the target system that affect its operation (e.g., affect one or more outputs of the target system). An automated control system for such a target system may in some situations have a distributed architecture that provides cooperative distributed control of the target system, such as with multiple decision modules that each control a portion of the target system and operate in a partially decoupled manner with respect to each other, with the various decision modules' operations being at least partially synchronized and each having a consensus with one or more other decision modules, even if a fully synchronized convergence of all decision modules at all times is not guaranteed.

COOPERATIVE DISTRIBUTED CONTROL OF TARGET SYSTEMS

BACKGROUND

[0001] Various attempts have been made to implement automated control systems for various types of physical systems that have inputs or other control elements that the control system can manipulate to attempt to provide desired output or other behavior of the physical systems being controlled. Such automated control systems have used various types of architectures and underlying computing technologies to attempt to implement such functionality, including to attempt to deal with issues related to uncertainty in the state of the physical system being controlled, the need to make control decisions in very short amounts of time to provide real-time or near-real-time control and with only partial information, etc.

[0002] However, various difficulties exist with existing automated control systems and their underlying architectures and computing technologies, including with respect to managing large numbers of constraints (sometimes conflicting), operating in a coordinated manner with other systems, etc.

BRIEF DESCRIPTION OF THE DRAWINGS

[0003] Figure 1 is a network diagram illustrating an example environment in which a system for performing cooperative distributed control of target systems may be configured and initiated.

[0004] Figure 2 is a network diagram illustrating an example environment in which a system for performing cooperative distributed control of target systems may be implemented.

[0005] Figure 3 is a block diagram illustrating example computing systems suitable for executing an embodiment of a system for performing cooperative distributed control of target systems in configured manners.

[0006] Figure 4 illustrates a flow diagram of an example embodiment of a Collaborative Distributed Decision (CDD) System routine.

[0007] Figures 5A-5B illustrate a flow diagram of an example embodiment of a CDD Decision Module Construction routine.

[0008] Figures 6A-6B illustrate a flow diagram of an example embodiment of a decision module routine.

[0009] Figures 7A-7B illustrate a flow diagram of an example embodiment of a CDD Control Action Determination routine.

[0010] Figures 8A-8B illustrate a flow diagram of an example embodiment of a CDD Coordinated Control Management routine.

[0011] Figure 9 illustrates a flow diagram of an example embodiment of a routine for a target system being controlled.

[0012] Figure 10 illustrates a network diagram of a portion of a distributed architecture of an example CDI system.

[0013] Figure 11 illustrates a network diagram of a portion of an example Rules Builder component.

[0014] Figure 12 illustrates a network diagram of example sub-components of an example Rules Builder component.

[0015] Figure 13 illustrates a network diagram of interactions of portions of a Rules Builder component with an executing agent.

[0016] Figure 14 illustrates a network diagram of interactions of portions of a Rules Builder component with chattering components.

[0017] Figure 15 illustrates a diagram of a sliding window for use with chattering.

[0018] Figure 16 illustrates a diagram of using Lebesgue integration to approximate a control trajectory for chattering.

[0019] Figure 17 illustrates a diagram of various interactions of different portions of a CDI system.

[0020] Figure 18 illustrates a diagram of example different types of rules.

[0021] Figure 19 illustrates an example user interface related to medical / clinical auto-coding.

[0022] Figure 20 illustrates a diagram of some components of a CDI system.

[0023] Figure 21 illustrates a diagram of performing Pareto processing for use with mean field techniques.

[0024] Figure 22 illustrates a network diagram of an example decision module agent.

[0025] Figure 23 illustrates a network diagram of an example of offline workflows for knowledge capture.

[0026] Figure 24 illustrates a network diagram of an example of workflows for mean field computation and Pareto Optimal.

[0027] Figure 25 illustrates a network diagram of an example of an automated control system for a home solar micro-grid electrical generating system.

[0028] Figure 26 illustrates a diagram of workflow and components of a portion of a CDI system.

[0029] Figure 27 illustrates a diagram of a workflow for an inference process portion of a CDI system.

[0030] Figure 28 illustrates a diagram of an overview workflow for a portion of a CDI system.

[0031] Figures 29A-29K illustrate examples of using a CDI system to iteratively determine near-optimal solutions over time for controlling a target system.

[0032] Figure 30 is a diagram illustrating additional example of portions of one or more embodiments.

[0033] Figure 31 is a diagram illustrating additional example of portions of one or more embodiments.

[0034] Figure 32 is a diagram illustrating additional example of portions of one or more embodiments.

[0035] Figure 33 is a diagram illustrating additional example of portions of one or more embodiments.

[0036] Figure 34 is a diagram illustrating additional example of portions of one or more embodiments.

[0037] Figure 35 is a diagram illustrating additional example of portions of one or more embodiments.

[0038] Figure 36 is a diagram illustrating additional example of portions of one or more embodiments.

DETAILED DESCRIPTION

[0039] Techniques are described for implementing automated control systems to control or otherwise manipulate at least some operations of specified physical systems or other target systems. A target system to be controlled or otherwise manipulated may have numerous elements that are inter-connected in various manners, with a subset of those elements being inputs or other control elements that a corresponding automated control system may modify or otherwise manipulate in order to affect the operation of the target system. In at least some embodiments and situations, a target system may further have one or more outputs that the manipulations of the control elements affect, such as if the target system is producing or modifying physical goods or otherwise producing physical effects.

[0040] As part of implementing such an automated control system for a particular target system, an embodiment of a Collaborative Distributed Decision (CDD) system may use the described techniques to perform various automated activities involved in constructing and implementing the automated control system – a brief introduction to some aspects of the activities of the CDD system is provided here, with additional details included below. In particular, the CDD system may in some embodiments implement a Decision Module Construction component that interacts with one or more users to obtain a description of a target system, including restrictions related to the various elements of the target system, and one or more goals to be achieved during control of the target system – the Decision Module Construction component then performs various automated actions to generate, test and deploy one or more executable decision modules (also referred to at times as “decision elements” and/or “agents”) to use in performing the control of

the target system. When the one or more executable decision modules are deployed and executed, the CDD system may further provide various components within or external to the decision modules being executed to manage their control of the target system, such as a Control Action Determination component of each decision module to optimize or otherwise enhance the control actions that the decision module generates, and/or one or more Coordinated Control Management components to coordinate the control actions of multiple decision modules that are collectively performing the control of the target system. Additional details related to such components of the CDD system and their automated operations are included below.

[0041] As noted above, the described techniques may be used to provide automated control systems for various types of physical systems or other target systems. In one or more embodiments, an automated control system is generated and provided and used to control a micro-grid electricity facility, such as at a residential location that includes one or more electricity sources (*e.g.*, one or more solar panel grids, one or more wind turbines, etc.) and one or more electricity storage and source mechanisms (*e.g.*, one or more batteries). The automated control system may, for example, operate at the micro-grid electricity facility (*e.g.*, as part of a home automation system), such as to receive requests from the operator of a local electrical grid to provide particular amounts of electricity at particular times, and to control operation of the micro-grid electricity facility by determining whether to accept each such request. If a request is accepted, the control actions may further include selecting which electricity source (*e.g.*, solar panel, battery, etc.) to use to provide the requested electricity, and otherwise the control actions may further include determine to provide electricity being generated to at least one energy storage mechanism (*e.g.*, to charge a battery). Outputs of such a physical system include the electricity being provided to the local electrical grid, and a goal that the automated control system implements may be, for example, is to maximize profits for the micro-grid electricity facility from providing of the electricity. It will be appreciated that such a physical system being

controlled and a corresponding automated control system may include a variety of elements and use various types of information and perform various types of activities, with additional details regarding such an automated control system being included below.

[0042] In one or more embodiments, an automated control system is generated and provided and used to control a vehicle with a motor and in some cases an engine, such as an electrical bicycle in which power may come from a user who is pedaling and/or from a motor powered by a battery and/or an engine. The automated control system may, for example, operate on the vehicle or on the user, such as to control operation of the vehicle by determining whether at a current time to remove energy from the battery to power the motor (and if so to further determine how much energy to remove from the battery) or to instead add excess energy to the battery (e.g., as generated by the engine, and if so to further determine how much energy to generate from the engine; and/or as captured from braking or downhill coasting). Outputs of such a physical system include the effects of the motor to move the vehicle, and a goal that the automated control system implements may be, for example, to move the vehicle at one or more specified speeds with a minimum of energy produced from the battery, and/or to minimize use of fuel by the engine. It will be appreciated that such a physical system being controlled and a corresponding automated control system may include a variety of elements and use various types of information and perform various types of activities, with additional details regarding such an automated control system being included below.

[0043] In one or more embodiments, an automated control system is generated and provided and used to manage product inventory for one or more products at one or more locations, such as a retail location that receives products from one or more product sources (e.g., when ordered or requested by the retail location) and that provides products to one or more product recipients (e.g., when ordered or requested by the recipients). The automated control system may, for example, operate at the retail location and/or at a remote network-accessible location, such

as to receive requests from product recipients for products, and to control operation of the product inventory at the one or more locations by selecting at a current time one or more first amounts of one or more products to request from the one or more product sources, and by selecting at the current time one or more second amounts of at least one product to provide to the one or more product recipients. Outputs of such a physical system include products being provided from the one or more locations to the one or more product recipients, and a goal that the automated control system implements may be, for example, to maximize profit of an entity operating the one or more locations while maintaining the inventory at one or more specified levels. It will be appreciated that such a physical system being controlled and a corresponding automated control system may include a variety of elements and use various types of information and perform various types of activities, with additional details regarding such an automated control system being included below.

[0044] In one or more embodiments, an automated control system is generated and provided and used to manage cyber-security for physical computing resources being protected from unauthorized operations and/or to determine a risk level from information provided by or available from one or more information sources. The automated control system may, for example, operate at the location of the computing resources or information sources and/or at a remote network-accessible location, such as to receive information about attempts (whether current or past) to perform operations on computing resources being protected or about information being provided by or available from the one or more information sources, and to control operation of the cyber-security system by determine whether a change in authorization to a specified type of operation is needed and to select one or more actions to take to implement the change in authorization if so determined, and/or to determine whether a risk level exceeds a specified threshold and to select one or more actions to take to mitigate the risk level. A goal that the automated control system implements may be, for example, to minimize unauthorized operations that are performed and/or to minimize the risk level. It will

be appreciated that such a target system being controlled and a corresponding automated control system may include a variety of elements and use various types of information and perform various types of activities, with additional details regarding such an automated control system being included below.

[0045] In one or more embodiments, an automated control system is generated and provided and used to manage transactions being performed in one or more financial markets, such as to buy and/or sell physical items or other financial items. The automated control system may, for example, operate at the one or more financial markets or at a network-accessible location that is remote from the one or more financial markets, such as to control operation of the transactions performed by determining whether to purchase or sell particular items at particular times and to select one or more actions to initiate transactions to purchase or sell the particular items at the particular times. A goal that the automated control system implements may be, for example, to maximize profit while maintaining risk below a specified threshold. It will be appreciated that such a target system being controlled and a corresponding automated control system may include a variety of elements and use various types of information and perform various types of activities, with additional details regarding such an automated control system being included below.

[0046] In one or more embodiments, an automated control system is generated and provided and used to perform coding for medical procedures, such as to allow billing to occur for medical procedures performed on humans. The automated control system may, for example, operate at a location at which the medical procedures are performed or at a network-accessible location that is remote from such a medical location, such as to control operation of the coding that is performed by selecting particular medical codes to associate with particular medical procedures in specified circumstances. A goal that the automated control system implements may be, for example, to minimize errors in selected medical codes that cause revenue leakage. It will be appreciated that such a target system being controlled and a corresponding automated control system may

include a variety of elements and use various types of information and perform various types of activities, with additional details regarding such an automated control system being included below.

[0047] It will also be appreciated that the described techniques may be used with a wide variety of other types of target systems, some of which are further discussed below, and that the invention is not limited to the techniques discussed for particular target systems and corresponding automated control systems.

[0048] As noted above, a Collaborative Distributed Decision (CDD) system may in some embodiments use at least some of the described techniques to perform various automated activities involved in constructing and implementing a automated control system for a specified target system, such as to modify or otherwise manipulate inputs or other control elements of the target system that affect its operation (e.g., affect one or more outputs of the target system). An automated control system for such a target system may in some situations have a distributed architecture that provides cooperative distributed control of the target system, such as with multiple decision modules that each control a portion of the target system and that operate in a partially decoupled manner with respect to each other. If so, the various decision modules' operations for the automated control system may be at least partially synchronized, such as by each reaching a consensus with one or more other decision modules at one or more times, even if a fully synchronized convergence of all decision modules at all times is not guaranteed or achieved.

[0049] The CDD system may in some embodiments implement a Decision Module Construction component that interacts with one or more users to obtain a description of a target system, including restrictions related to the various elements of the target system, and one or more goals to be achieved during control of the target system – the Decision Module Construction component then performs various automated actions to generate, test and deploy one or more executable decision modules to use in performing the control of the target system. The Decision Module Construction component may thus operate as part of a

configuration or setup phase that occurs before a later run-time phase in which the generated decision modules are executed to perform control of the target system, although in some embodiments and situations the Decision Module Construction component may be further used after an initial deployment to improve or extend or otherwise modify an automated control system that has one or more decision modules (e.g., while the automated control system continues to be used to control the target system), such as to add, remove or modify decision modules for the automated control system.

[0050] In some embodiments, some or all automated control systems that are generated and deployed may further provide various components within them for execution during the runtime operation of the automated control system, such as by including such components within decision modules in some embodiments and situations. Such components may include, for example, a Control Action Determination component of each decision module (or of some decision modules) to optimize or otherwise determine and improve the control actions that the decision module generates. For example, such a Control Action Determination component in a decision module may in some embodiments attempt to automatically determine the decision module's control actions for a particular time to reflect a near-optimal solution with respect to or one more goals and in light of a model of the decision module for the target system that has multiple inter-related constraints – if so, such a near-optimal solution may be based at least in part on a partially optimized solution that is within a threshold amount of a fully optimized solution. Such determination of one or more control actions to perform may occur for a particular time and for each of one or more decision modules, as well as be repeated over multiple times for ongoing control by at least some decision modules in some situations. In some embodiments, the model for a decision module is implemented as a Hamiltonian function that reflects a set of coupled differential equations based in part on constraints representing at least part of the target system, such as to allow the model and its Hamiltonian function

implementation to be updated over multiple time periods by adding additional expressions within the evolving Hamiltonian function.

[0051] In some embodiments, the components included within a generated and deployed automated control system for execution during the automated control system's runtime operation may further include one or more Coordinated Control Management components to coordinate the control actions of multiple decision modules that are collectively performing the control of a target system for the automated control system. For example, some or all decision modules may each include such a Control Action Determination component in some embodiments to attempt to synchronize that decision module's local solutions and proposed control actions with those of one or more other decision modules in the automated control system, such as by determining a consensus shared model with those other decision modules that simultaneously provides solutions from the decision module's local model and the models of the one or more other decision modules. Such inter-module synchronizations may occur repeatedly to determine one or more control actions for each decision module at a particular time, as well as to be repeated over multiple times for ongoing control. In addition, each decision module's model is implemented in some embodiments as a Hamiltonian function that reflects a set of coupled differential equations based in part on constraints representing at least part of the target system, such as to allow each decision module's model and its Hamiltonian function implementation to be combined with the models of one or more other decision modules by adding additional expressions for those other decision modules' models within the initial Hamiltonian function for the local model of the decision module.

[0052] Use of the described techniques may also provide various types of benefits in particular embodiments, including non-exclusive examples of beneficial attributes or operations as follows:

- Infer interests/desired content in a cold start environment where textual (or other unstructured) data is available and with minimal user history;

- Improve inference in a continuous way that can incorporate increasingly rich user histories;
- Improve inference performance with the addition of feedback, explicit/implicit, positive/negative and preferably in a real-time or near-real-time manner;
- Derive information from domain experts that provide business value, and embed them in inference framework;
- Dynamically add new unstructured data that may represent new states, and update existing model in a calibrated way;
- Renormalize inference system to accommodate conflicts;
- Immediately do inferencing in a new environment based on a natural language model;
- Add new information as a statistical model, and integrate with a natural language model to significantly improve inference/prediction;
- Integrate new data and disintegrate old data in a way that only improves performance;
- Perform inferencing in a data secure way;
- Integrate distinct inferencing elements in a distributed network and improve overall performance;
- Easily program rules and information into the system from a lay-user perspective;
 - Inexpensively perform computer inferences in a way that is suitable for bandwidth of mobile devices; and
 - Incorporate constraint information.

It will be appreciated that some embodiments may not include all some illustrative benefits, and that some embodiments may include some benefits that are not listed.

[0053] For illustrative purposes, some embodiments are described below in which specific types of operations are performed, including with respect to using the described techniques with particular types of target systems and to perform particular types of control activities that are determined in particular manners. These examples are provided for illustrative purposes and are simplified for the

sake of brevity, and the inventive techniques may be used in a wide variety of other situations, including in other environments and with other types of automated control action determination techniques, some of which are discussed below.

[0054] Figure 1 is a network diagram illustrating an example environment in which a system for performing cooperative distributed control of one or more target systems may be configured and initiated. In particular, an embodiment of a CDD system 140 is executing on one or more computing systems 190, including in the illustrated embodiment to operate in an online manner and provide a graphical user interface (GUI) (not shown) and/or other interfaces 119 to enable one or more remote users of client computing systems 110 to interact over one or more intervening computer networks 100 with the CDD system 140 to configure and create one or more decision modules to include as part of an automated control system to use with each of one or more target systems to be controlled.

[0055] In particular, target system 1 160 and target system 2 170 are example target systems illustrated in this example, although it will be appreciated that only one target system or numerous target systems may be available in particular embodiments and situations, and that each such target system may include a variety of mechanical, electronic, chemical, biological, and/or other types of components to implement operations of the target system in a manner specific to the target system. In this example, the one or more users (not shown) may interact with the CDD system 140 to generate an example automated control system 122 for target system 1, with the automated control system including multiple decision modules 124 in this example that will cooperatively interact to control portions of the target system 1 160 when later deployed and implemented. The process of the users interacting with the CDD system 140 to create the automated control system 122 may involve a variety of interactions over time, including in some cases independent actions of different groups of users, as discussed in greater detail elsewhere. In addition, as part of the process of creating and/or training or testing automated control system 122, it may perform one or more interactions with the target system 1 as illustrated, such as to obtain

partial initial state information, although some or all training activities may in at least some embodiments include simulating effects of control actions in the target system 1 without actually implementing those control actions at that time.

[0056] After the automated control system 122 is created, the automated control system may be deployed and implemented to begin performing operations involving controlling the target system 1 160, such as by optionally executing the automated control system 122 on the one or more computing systems 190 of the CDD system 140, so as to interact over the computer networks 100 with the target system 1. In other embodiments and situations, the automated control system 122 may instead be deployed by executing local copies of some or all of the automated control system 122 (e.g., one or more of the multiple decision modules 124) in a manner local to the target system 1, as illustrated with respect to a deployed copy 121 of some or all of automated control system 1, such as on one or more computing systems (not shown) that are part of the target system 1.

[0057] In a similar manner to that discussed with respect to automated control system 122, one or more users (whether the same users, overlapping users, or completely unrelated users to those that were involved in creating the automated control system 122) may similarly interact over the computer network 100 with the CDD system 140 to create a separate automated control system 126 for use in controlling some or all of the target system 2 170. In this example, the automated control system 126 for target system 2 includes only a single decision module 128 that will perform all of the control actions for the automated control system 126. The automated control system 126 may similarly be deployed and implemented for target system 2 in a manner similar to that discussed with respect to automated control system 122, such as to execute locally on the one or more computing systems 190 and/or on one or more computing systems (not shown) that are part of the target system 2, although a deployed copy of automated control system 2 is not illustrated in this example. It will be further appreciated that the automated control systems 122 and/or 126 may further include other components and/or functionality that are separate from the particular decision modules 124 and 128,

respectively, although such other components and/or functionality are not illustrated in Figure 1.

[0058] The network 100 may, for example, be a publicly accessible network of linked networks, possibly operated by various distinct parties, such as the Internet, with the CDD system 140 available to any users or only certain users over the network 100. In other embodiments, the network 100 may be a private network, such as, for example, a corporate or university network that is wholly or partially inaccessible to non-privileged users. In still other embodiments, the network 100 may include one or more private networks with access to and/or from the Internet. Thus, while the CDD system 140 in the illustrated embodiment is implemented in an online manner to support various users over the one or more computer networks 100, in other embodiments a copy of the CDD system 140 may instead be implemented in other manners, such as to support a single user or a group of related users (e.g., a company or other organization), such as if the one or more computer networks 100 are instead an internal computer network of the company or other organization, and with such a copy of the CDD system optionally not being available to other users external to the company or other organizations. The online version of the CDD system 140 and/or local copy version of the CDD system 140 may in some embodiments and situations operate in a fee-based manner, such that the one or more users provide various fees to use various operations of the CDD system, such as to perform interactions to generate decision modules and corresponding automated control systems, and/or to deploy or implement such decision modules and corresponding automated control systems in various manners. In addition, the CDD system 140, each of its components (including component 142 and optional other components 117, such as one or more CDD Control Action Determination components and/or one or more CDD Coordinated Control Management components), each of the decision modules, and/or each of the automated control systems may include software instructions that execute on one or more computing systems (not shown) by one or more processors (not shown), such as to configure those processors and

computing systems to operate as specialized machines with respect to performing their programmed functionality.

[0059] Figure 2 is a network diagram illustrating an example environment in which a system for performing cooperative distributed control of target systems may be implemented, and in particular continues the examples discussed with respect to Figure 1. In the example environment of Figure 2, target system 1 160 is again illustrated, with the automated control system 122 now being deployed and implemented to use in actively controlling the target system 1 160. In the example of Figure 2, the decision modules 124 are represented as individual decision modules 124a, 124b, etc., to 124n, and may be executing locally to the target system 1 160 and/or in a remote manner over one or more intervening computer networks (not shown). In the illustrated example, each of the decision modules 124 includes a local copy of a CDD Control Action Determination component 144, such as with component 144a supporting its local decision module 124a, component 144b supporting its local decision module 124b, and component 144n supporting its local decision module 124n. Similarly, the actions of the various decision modules 124 are coordinated and synchronized in a peer-to-peer manner in the illustrated embodiment, with each of the decision modules 124 including a copy of a CDD Coordinated Control Management component 146 to perform such synchronization, with component 146a supporting its local decision module 124a, component 146b supporting its local decision module 124b, and component 146n supporting its local decision module 124n.

[0060] As the decision modules 124 and automated control system 122 execute, various interactions 175 between the decision modules 124 are performed, such as to share information about current models and other state of the decision modules to enable cooperation and coordination between various decision modules, such as for a particular decision module to operate in a partially synchronized consensus manner with respect to one or more other decision modules (and in some situations in a fully synchronized manner in which the consensus actions of all of the decision modules 124 converge). During operation

of the decision modules 124 and automated control system 122, various state information 143 may be obtained by the automated control system 122 from the target system 160, such as initial state information and changing state information over time, and including outputs or other results in the target system 1 from control actions performed by the decision modules 124.

[0061] The target system 1 in this example includes various control elements 161 that the automated control system 122 may manipulate, and in this example each decision module 124 may have a separate group of one or more control elements 161 that it manipulates (such that decision module A 124a performs interactions 169a to perform control actions A 147a on control elements A 161a, decision module B 124b performs interactions 169b to perform control actions B 147b on control elements B 161b, and decision module N 124n performs interactions 169n to perform control actions N 147n on control elements N 161n). Such control actions affect the internal state 163 of other elements of the target system 1, including optionally to cause or influence one or more outputs 162. As operation of the target system 1 is ongoing, at least some of the internal state information 163 is provided to some or all of the decision modules to influence their ongoing control actions, with each of the decision modules 124a-124n possibly having a distinct set of state information 143a-143n, respectively, in this example.

[0062] As discussed in greater detail elsewhere, each decision module 124 may use such state information 143 and a local model 145 of the decision module for the target system to determine particular control actions 147 to next perform, such as for each of multiple time periods, although in other embodiments and situations, a particular automated control system may perform interactions with a particular target system for only one time period or only for some time periods. For example, the local CDD Control Action Determination component 144 for a decision module 124 may determine a near-optimal location solution for that decision module's local model 145, and with the local CDD Coordinated Control Management component 146 determining a synchronized consensus solution to reflect other of the decision modules 124, including to update the decision module's local model

145 based on such local and/or synchronized solutions that are determined. Thus, during execution of the automated control system 122, the automated control system performs various interactions with the target system 160, including to request state information, and to provide instructions to modify values of or otherwise manipulate control elements 161 of the target system 160. For example, for each of multiple time periods, decision module 124a may perform one or more interactions 169a with one or more control elements 161a of the target system, while decision module 124b may similarly perform one or more interactions 169b with one or more separate control elements B 161b, and decision module 124n may perform one or more interactions 169n with one or more control elements N 161n of the target system 160. In other embodiments and situations, at least some control elements may not perform control actions during each time period.

[0063] While example target system 2 170 is not illustrated in Figure 2, further details are illustrated for decision module 128 of automated control system 126 for reference purposes, although such a decision module 128 would not typically be implemented together with the decision modules 124 controlling target system 1. In particular, the deployed copy of automated control system 126 includes only the single executing decision module 128 in this example, although in other embodiments the automated control system 126 may include other components and functionality. In addition, since only a single decision module 128 is implemented for the automated control system 126, the decision module 128 includes a local CDD Control Action Determination component 244, but does not in the illustrated embodiment include any local CDD Coordinated Control Management component, since there are not other decision modules with which to synchronize and interact.

[0064] While not illustrated in Figures 1 and 2, the distributed nature of operations of automated control systems such as those of 122 allow partially decoupled operations of the various decision modules, include to allow modifications to the group of decision modules 124 to be modified over time while the automated

control system 122 is in use, such as to add new decision modules 124 and/or to remove existing decision modules 124. In a similar manner, changes may be made to particular decision modules 124 and/or 128, such as to change rules or other restrictions specific to a particular decision module and/or to change goals specific to a particular decision module over time, with a new corresponding model being generated and deployed within such a decision module, including in some embodiments and situations while the corresponding automated control system continues control operations of a corresponding target system. In addition, while each automated control system is described as controlling a single target system in the examples of Figures 1 and 2, in other embodiments and situations, other configurations may be used, such as for a single automated control system to control multiple target systems (e.g., multiple inter-related target systems, multiple target systems of the same type, etc.), and/or multiple automated control systems may operate to control a single target system, such as by each operating independently to control different portions of that target control system. It will be appreciated that other configurations may similarly be used in other embodiments and situations.

[0065] Figure 3 is a block diagram illustrating example computing systems suitable for performing techniques for implementing automated control systems to control or otherwise manipulate at least some operations of specified physical systems or other target systems in configured manners. In particular, Figure 3 illustrates a server computing system 300 suitable for providing at least some functionality of a CDD system, although in other embodiments multiple computing systems may be used for the execution (e.g., to have distinct computing systems executing the CDD Decision Module Construction component for initial configuration and setup before run-time control occurs, and one or more copies of the CDD Control Action Determination component 344 and/or the CDD Coordinated Control Managements component 346 for the actual run-time control). Figure 3 also illustrates various client computer systems 350 that may be used by customers or other users of the CDD system 340, as well as one or more target systems (in this example, target

system 1 360 and target system 2 370, which are accessible to the CDD system 340 over one or more computer networks 390).

[0066] The server computing system 300 has components in the illustrated embodiment that include one or more hardware CPU (“central processing unit”) computer processors 305, various I/O (“input/output”) hardware components 310, storage 320, and memory 330. The illustrated I/O components include a display 311, a network connection 312, a computer-readable media drive 313, and other I/O devices 315 (e.g., a keyboard, a mouse, speakers, etc.). In addition, the illustrated client computer systems 350 may each have components similar to those of server computing system 300, including one or more CPUs 351, I/O components 352, storage 354, and memory 357, although some details are not illustrated for the computing systems 350 for the sake of brevity. The target systems 360 and 370 may also each include one or more computing systems (not shown) having components that are similar to some or all of the components illustrated with respect to server computing system 300, but such computing systems and components are not illustrated in this example for the sake of brevity.

[0067] The CDD system 340 is executing in memory 330 and includes components 342-346, and in some embodiments the system and/or components each includes various software instructions that when executed program one or more of the CPU processors 305 to provide an embodiment of a CDD system as described elsewhere herein. The CDD system 340 may interact with computing systems 350 over the network 390 (e.g., via the Internet and/or the World Wide Web, via a private cellular network, etc.), as well as the target systems 360 and 370 in this example. In this example embodiment, the CDD system includes functionality related to generating and deploying decision modules in configured manners for customers or other users, as discussed in greater detail elsewhere herein. The other computing systems 350 may also be executing various software as part of interactions with the CDD system 340 and/or its components. For example, client computing systems 350 may be executing software in memory 357 to interact with CDD system 340 (e.g., as part of a Web browser, a specialized

client-side application program, etc.), such as to interact with one or more interfaces (not shown) of the CDD system 340 to configure and deploy automated control systems (e.g., stored automated control systems 325 that were previously created by the CDD system 340) or other decision modules 329, as well as to perform various other types of actions, as discussed in greater detail elsewhere. Various information related to the functionality of the CDD system 340 may be stored in storage 320, such as information 321 related to users of the CDD system (e.g., account information), and information 323 related to one or more target systems.

[0068] It will be appreciated that computing systems 300 and 350 and target systems 360 and 370 are merely illustrative and are not intended to limit the scope of the present invention. The computing systems may instead each include multiple interacting computing systems or devices, and the computing systems/nodes may be connected to other devices that are not illustrated, including through one or more networks such as the Internet, via the Web, or via private networks (e.g., mobile communication networks, etc.). More generally, a computing node or other computing system or device may comprise any combination of hardware that may interact and perform the described types of functionality, including without limitation desktop or other computers, database servers, network storage devices and other network devices, PDAs, cell phones, wireless phones, pagers, electronic organizers, Internet appliances, television-based systems (e.g., using set-top boxes and/or personal/digital video recorders), and various other consumer products that include appropriate communication capabilities. In addition, the functionality provided by the illustrated CDD system 340 and its components may in some embodiments be distributed in additional components. Similarly, in some embodiments some of the functionality of the CDD system 340 and/or CDD components 342-346 may not be provided and/or other additional functionality may be available.

[0069] It will also be appreciated that, while various items are illustrated as being stored in memory or on storage while being used, these items or portions of them

may be transferred between memory and other storage devices for purposes of memory management and data integrity. Alternatively, in other embodiments some or all of the software modules and/or systems may execute in memory on another device and communicate with the illustrated computing systems via inter-computer communication. Thus, in some embodiments, some or all of the described techniques may be performed by hardware means that include one or more processors and/or memory and/or storage when configured by one or more software programs (e.g., by the CDD system 340 and/or the CDD components 342-346) and/or data structures, such as by execution of software instructions of the one or more software programs and/or by storage of such software instructions and/or data structures. Furthermore, in some embodiments, some or all of the systems and/or components may be implemented or provided in other manners, such as by using means that are implemented at least partially or completely in firmware and/or hardware, including, but not limited to, one or more application-specific integrated circuits (ASICs), standard integrated circuits, controllers (e.g., by executing appropriate instructions, and including microcontrollers and/or embedded controllers), field-programmable gate arrays (FPGAs), complex programmable logic devices (CPLDs), etc. Some or all of the components, systems and data structures may also be stored (e.g., as software instructions or structured data) on a non-transitory computer-readable storage medium, such as a hard disk or flash drive or other non-volatile storage device, volatile or non-volatile memory (e.g., RAM), a network storage device, or a portable media article to be read by an appropriate drive (e.g., a DVD disk, a CD disk, an optical disk, etc.) or via an appropriate connection. The systems, components and data structures may also in some embodiments be transmitted as generated data signals (e.g., as part of a carrier wave or other analog or digital propagated signal) on a variety of computer-readable transmission mediums, including wireless-based and wired/cable-based mediums, and may take a variety of forms (e.g., as part of a single or multiplexed analog signal, or as multiple discrete digital packets or frames). Such computer program products may also take other forms in other

embodiments. Accordingly, the present invention may be practiced with other computer system configurations.

[0070] Figure 4 is a flow diagram of an example embodiment of a Collaborative Distributed Decision (CDD) system routine 400. The routine may, for example, be provided by execution of the CDD system 340 of Figure 3 and/or the CDD system 140 of Figure 1, such as to provide functionality to construct and implement automated control systems for specified target systems.

[0071] The illustrated embodiment of the routine begins at block 410, where information or instructions are received. If it is determined in block 420 that the information or instructions of block 410 include an indication to create or revise one or more decision modules for use as part of an automated control system for a particular target system, the routine continues to block 425 to initiate execution of a Decision Module Construction component, and in block 430 obtains and stores one or more resulting decision modules for the target system that are created in block 425. One example of a routine for such a Decision Module Construction component is discussed in greater detail with respect to Figures 5A-5B.

[0072] After block 430, or if it is instead determined in block 420 that the information or instructions received in block 410 are not to create or revise one or more decision modules, the routine continues to block 440 to determine whether the information or instructions received in block 410 indicate to deploy one or more created decision modules to control a specified target system, such as for one or more decision modules that are part of an automated control system for that target system. The one or more decision modules to deploy may have been created immediately prior with respect to block 425, such that the deployment occurs in a manner that is substantially simultaneous with the creation, or in other situations may include one or more decision modules that were created at a previous time and stored for later use. If it is determined to deploy one or more such decision modules for such a target system, the routine continues to block 450 to initiate the execution of those one or more decision modules for that target system, such as on one or more computing systems local to an environment of the target system,

or instead on one or more remote computing systems that communicate with the target system over one or more intermediary computer networks (e.g., one or more computing systems under control of a provider of the CDD system).

[0073] After block 450, the routine continues to block 460 to determine whether to perform distributed management of multiple decision modules being deployed in a manner external to those decision modules, such as via one or more centralized Coordinated Control Management components. If so, the routine continues to block 465 to initiate execution of one or more such centralized CDD Coordinated Control Management components for use with those decision modules. After block 465, or if it is instead determined in block 460 to not perform such distributed management in an external manner (e.g., if only one decision module is executed, if multiple decision modules are executed but coordinate their operations in a distributed peer-to-peer manner, etc.), the routine continues to block 470 to optionally obtain and store information about the operations of the one or more decision modules and/or resulting activities that occur in the target system, such as for later analysis and/or reporting.

[0074] If it is instead determined in block 440 that the information or instructions received in block 410 are not to deploy one or more decision modules, the routine continues instead to block 485 to perform one or more other indicated operations if appropriate. For example, such other authorized operations may include obtaining results information about the operation of a target system in other manners (e.g., by monitoring outputs or other state information for the target system), analyzing results of operations of decision modules and/or activities of corresponding target systems, generating reports or otherwise providing information to users regarding such operations and/or activities, etc. In addition, in some embodiments the analysis of activities of a particular target system over time may allow patterns to be identified in operation of the target system, such as to allow a model of that target system to be modified accordingly (whether manually or in an automated learning manner) to reflect those patterns and to respond based on them. In addition, as discussed in greater detail elsewhere, distributed operation of multiple

decision modules for an automated control system in a partially decoupled manner allows various changes to be made while the automated control system is in operation, such as to add one or more new decision modules, to remove one or more existing decision modules, to modify the operation of a particular decision module (e.g., by changing rules or other information describing the target system that is part of a model for the decision module), etc. In addition, the partially decoupled nature of multiple such decision modules in an automated control system allows one or more such decision modules to operate individually at times, such as if network communication issues or other problems prevent communication between multiple decision modules that would otherwise allow their individualized control actions to be coordinated – in such situations, some or all such decision modules may continue to operate in an individualized manner, such as to provide useful ongoing control operations for a target system even if optimal or near-optimal solutions cannot be identified from coordination and synchronization between a group of multiple decision modules that collectively provide the automated control system for the target system.

[0075] After blocks 470 or 485, the routine continues to block 495 to determine whether to continue, such as until an explicit indication to terminate is received. If it is determined to continue, the routine returns to block 410, and otherwise continues to block 499 and ends.

[0076] Figures 5A-5B illustrate a flow diagram of an example embodiment of a CDD Decision Module Construction routine 500. The routine may, for example, be provided by execution of the component 342 of Figure 3 and/or the component 142 of Figure 1, such as to provide functionality to allow users to provide information describing a target system of interest, and to perform corresponding automated operations to construct one or more decision modules to use to control the target system in specified manners. While the illustrated embodiment of the routine interacts with users in particular manners, such as via a displayed GUI (graphical user interface), it will be appreciated that other embodiments of the routine may interact with users in other manners, such as via a defined API

(application programming interface) that an executing program invokes on behalf of a user. In some embodiments, the routine may further be implemented as part of an integrated development environment or other software tool that is available for one or more users to use, such as by implementing an online interface that is available to a variety of remote users over a public network such as the Internet, while in other embodiments a copy of the CDD system and/or particular CDD components may be used to support a single organization or other group of one or more users, such as by being executed on computing systems under the control of the organization or group. In addition, the CDD Decision Module Construction component may in some embodiments and situations be separated into multiple sub-components, such as a rules editor component that users interact with to specify rules and other description information for a target system, and a rules compiler engine that processes the user-specified rules and other information to create one or more corresponding decision modules.

[0077] The illustrated embodiment of the routine 500 begins at block 510, where the routine provides or updates a displayed user interface to one or more users, such as via a request received at an online version of component that is implementing the routine, or instead based on the routine being executed by one or more such users on computing systems that they control. While various operations are shown in the illustrated embodiment of the routine as occurring in a serial manner for the purpose of illustration, it will be appreciated that user interactions with such a user interface may occur in an iterative manner and/or over multiple periods of time and/or user sessions, including to update a user interface previously displayed to a user in various manners (e.g., to reflect a user action, to reflect user feedback generated by operation of the routine or from another component, etc.), as discussed further below.

[0078] After block 510, the routine continues to block 520 to receive information from one or more such users describing a target system to be controlled, including information about a plurality of elements of the target system that include one or more manipulatable control elements and optionally one or more outputs that the

control elements affect, information about rules that specify restrictions involving the elements, information about state information that will be available during controlling of the system (e.g., values of particular elements or other state variables), and one or more goals to achieve during the controlling of the target system. It will be appreciated that such information may be obtained over a period of time from one or more users, including in some embodiments for a first group of one or more users to supply some information related to a target system and for one or more other second groups of users to independently provide other information about the target system, such as to reflect different areas of expertise of the different users and/or different parts of the target system.

[0079] After block 520, the routine continues to block 525 to identify any errors that have been received in the user input, and to prompt the user(s) to correct those errors, such as by updating the display in a corresponding manner as discussed with respect to block 510. While the identification of such errors is illustrated as occurring after the receiving of the information in block 520, it will be appreciated that some or all such errors may instead be identified as the users are inputting information into the user interface, such as to identify syntax errors in rules or other information that the users specify. After block 525, the illustrated embodiment of the routine continues to block 530 to optionally decompose the information about the target system into multiple subsets that each correspond to a portion of the target system, such as with each subset having one or more different control elements that are manipulatable by the automated control system being created by the routine, and optionally have overlapping or completely distinct goals and/or sets of rules and other information describing the respective portions of the target system. As discussed in greater detail elsewhere, such decomposition, if performed, may in some situations be performed manually by the users indicating different subgroups of information that they enter, and/or in an automated manner by the routine based on an analysis of the information that has been specified (e.g., based on the size of rules and other descriptive information supplied for a target system, based on inter-relationships between different rules

or goals or other information, etc.). In other embodiments, no such decomposition may be performed.

[0080] After block 530, the routine continues to block 535 to, for each subset of target system description information (or for all the received information if no such subsets are identified), convert that subset (or all the information) into a set of constraints that encapsulate the restrictions, goals, and other specified information for that subset (or for all the information). In block 540, the routine then identifies any errors that occur from the converting process, and if any are identified, may prompt the user to correct those errors, such as in a manner similar to that described with respect to blocks 525 and 510. While not illustrated in this example, the routine may in some situations in blocks 525 and/or 540 return to block 510 when such errors are identified, to display corresponding feedback to the user(s) and to allow the user(s) to make corrections and re-perform following operations such as those of blocks 520-540. The errors identified in the converting process in block 540 may include, for example, errors related to inconsistent restrictions, such as if the restrictions as a group are impossible to satisfy.

[0081] After block 540, the routine continues to block 545 to, for each set of constraints (or a single constraint set if no subsets were identified in block 530), apply one or more validation rules to the set of constraints to test overall effectiveness of the corresponding information that the constraints represent, and to prompt the one or more users to correct any errors that are identified in a manner similar to that with respect to blocks 525, 540 and 510. Such validation rules may test one or more of controllability, observability, stability, and goal completeness, as well as any user-added validation rules, as discussed in greater detail elsewhere. In block 550, the routine then converts each validated set of constraints to a set of coupled differential equations that model at least a portion of the target system to which the underlying information corresponds.

[0082] After block 550, the routine continues to block 553 to perform activities related to training a model for each set of coupled differential equations, including

to determine one or more of a size of a training time window to use, size of multiple training time slices within the time window, and/or a type of training time slice within the time window. In some embodiments and situations, the determination of one or more such sizes or types of information is performed by using default or pre-specified information, while in other embodiments and situations the users may specify such information, or an automated determination of such information may be performed in one or more manners (e.g., by testing different sizes and evaluating results to find sizes with the best performance). Different types of time slices may include, for example, successions of time slices that overlap or do not overlap, such that the training for a second time slice may be dependent only on results of a first time slice (if they do not overlap) or instead may be based at least in part on updating information already determined for at least some of the first time slice (if they do overlap in part or in whole). After block 553, the routine continues to block 555 to, for each set of coupled differential equations representing a model, train the model for that set of coupled differential equations using partial initial state information for the target system, including to estimate values of variable that are not known and/or directly observable for the target system by simulating effects of performing control actions over the time window, such as for successive time slices throughout the time window, and to test the simulated performance of the trained model. Additional details related to training and testing are included elsewhere herein.

[0083] After block 555, the routine continues to block 560 to determine whether the training and testing was successful, and if not returns to block 510 to display corresponding feedback information to the users to allow them to correct errors that caused the lack of success. If it is instead determined in block 560 that the testing and training were successful, however, the routine continues instead to block 570 to generate an executable decision module for each trained and tested model that includes that model, as well as a local CCD Control Action Determination component that the decision module will use when executed to determine optimal or near-optimal control actions to perform for the target system

based on the information included in the model, and in light of the one or more goals for that decision module. The generated executable decision module may in some embodiments and situations further include a local CCD Coordinated Control Management component to coordinate control actions of multiple decision modules that collectively will provide an automated control system for the target system, such as by synchronizing respective models of the various decision modules over time. After block 570, the routine continues to block 580 to provide the generated executable decision modules for use, including to optionally store them for later execution and/or deployment.

[0084] After block 580, the routine continues to block 595 to determine whether to continue, such as until an explicit indication to terminate is received. If it is determined to continue, the routine returns to block 510, and otherwise continues to block 599 and ends.

[0085] Figures 6A-6B illustrate a flow diagram of an example embodiment of a routine 600 corresponding to a generic representation of a decision module that is being executed. The routine may, for example, be provided by execution of a decision module 329 or as part of an automated control system 325 of Figure 3 and/or a decision module 124 or 128 of Figures 1 or 2, such as to provide functionality for controlling at least a portion of a target system in a manner specific to information and a model encoded for the decision module, including to reflect one or more goals to be achieved by the decision module during its controlling activities. As discussed in greater detail elsewhere, in some embodiments and situations, multiple decision modules may collectively and cooperatively act to control a particular target system, such as with each decision module controlling one or more distinct control elements for the target system or otherwise representing or interacting with a portion of the target system, while in other embodiments and situations a single decision module may act alone to control a target system. The routine 600 further reflects actions performed by a particular example decision module when it is deployed in controlling a portion of a target system, although execution of at least portions of a decision module may

occur at other times, such as initially to train a model for the decision module before the decision module is deployed, as discussed in greater detail with respect to the CDD Decision Module Construction routine 500 of Figures 5A-5B.

[0086] The illustrated embodiment of the routine 600 begins at block 610, where an initial model for the decision module is determined that describes at least a portion of a target system to be controlled, one or more goals for the decision module to attempt to achieve related to control of the target system, and optionally initial state information for the target system. The routine continues to block 615 to perform one or more actions to train the initial model if needed, as discussed in greater detail with respect to blocks 553 and 555 of Figures 5A-5B - in some embodiments and situations, such training for block 615 is performed only if initial training is not done by the routine 500 of Figures 5A-5B, while in other embodiments and situations the training of block 615 is performed to capture information about a current state of the target system at a time that the decision module begins to execute (e.g., if not immediately deployed after initial creation and training) and/or to re-train the model at times as discussed in greater detail with respect to routine 700 of Figures 7A-7B as initiated by block 630.

[0087] After block 615, the routine continues to block 617 to determine a time period to use for performing each control action decision for the decision module, such as to reflect a rate at which control element modifications in the target system are needed and/or to reflect a rate at which new incoming state information is received that may alter future manipulations of the control elements. The routine then continues to block 620 to start the next time period, beginning with a first time period moving forward from the startup of the execution of the decision module. Blocks 620-680 are then performed in a loop for each such time period going forward until execution of the decision module is suspended or terminated, although in other embodiments a particular decision module may execute for only a single time period each time that it is executed.

[0088] In block 625, the routine optionally obtains state information for the time period, such as current state information that has been received for the target

system or one or more related external sources since the last time period began, and/or by actively retrieving current values of one or more elements of the target system or corresponding variables as needed. In block 630, the routine then initiates execution of a local CCD Control Action Determination component of the decision module, with one example of such a routine discussed in greater detail with respect to routine 700 of Figures 7A-7B. In block 635, the results of the execution of the component in block 630 are received, including to either obtain an updated model for the decision module with a local solution for the current time period and decision module that includes one or more proposed control action determinations that the decision module may perform for the current time period, or to receive an indication that no local solution was found for the decision module in the allowed time for the execution of the component in block 630. It is then determined in block 640 whether a solution was found, and if so continues to block 642 to store the updated model for the decision module, and otherwise continues to block 643 to use the prior model for the decision module to determine one or more control action determinations that are proposed for the current time period based on a previous model (e.g., that does not reflect recent changes in state information and/or recent changes in activities of other decision modules, if any), as discussed in greater detail with respect to routine 700 of Figures 7A-7B.

[0089] After blocks 642 or 643, the routine continues to block 644 to determine if other decision modules are collectively controlling portions of the current target system, such as part of the same automated control system as the local decision module, and if so continues to block 645. Otherwise, the routine selects the local proposed control actions of the decision module as a final determined control action to perform, and continues to block 675 to implement those control actions for the current time period.

[0090] If there are other operating decision modules, the routine in block 645 determines if the local decision module includes a local copy of a CDD Coordinated Control Management (CCM) component for use in synchronizing the proposed control action determinations for the decision module's local solutions

with activities of other decision modules that are collectively controlling the same target system. If so, the routine continues to block 647 to provide the one or more proposed control action determinations of the decision module and the corresponding current local model for the decision module to the local CDD CCM component, and otherwise continues to block 649 to provide the one or more proposed control action determinations for the decision module and the corresponding local model of the decision module to one or more centralized CDD CCM components.

[0091] After blocks 647 or 649, the routine continues to block 655 to obtain results of the actions of the CDD CCM component(s) in blocks 647 or 649, including to either obtain a further updated model resulting from synchronization of the local model for the current decision module with information from one or more other decision modules, such that the further updated model indicates one or more final control action determinations to perform for the time period for the current decision module, or an indication that no such synchronization was completed in the allowed time. The routine continues to block 660 to determine whether the synchronization was completed, and if so continues to block 665 to store the further updated model from the synchronization, and otherwise continues to block 670 to use the prior proposed control action determinations locally to the decision module as the final control action determinations for the time period.

[0092] After blocks 665 or 670, the routine continues to block 675 to implement the one or more final determined control actions for the decision module in the target system, such as by interacting with one or more effectuators in the target system that modify values or otherwise manipulate one or more control elements of the target system, or by otherwise providing input to the target system to cause such modifications or other manipulations to occur. In block 680, the routine optionally obtains information about the results in the target system of the control actions performed, and stores and/or provides information to the CDD system about such obtained results and/or about the activities of the decision module for the current time period.

[0093] After block 680, the routine continues to block 695 to determine whether to continue, such as until an indication to terminate or suspend is received (e.g., to reflect an end to current operation of the target system or an end of use of the decision module to control at least a portion of the target system). If it is determined to continue, the routine returns to block 620 to start the next time period, and otherwise continues to block 699 and ends.

[0094] Figures 7A-7B are a flow diagram of a example embodiment of a CDD Control Action Determination routine 700. The routine may, for example, be provided by execution of the component 344 of Figure 3 and/or components 144a-n or 244 of Figure 2, such as to determine control actions for a decision module to propose and/or implement for a target system during a particular time period, including in some embodiments to perform an optimization to determine near-optimal actions (e.g., within a threshold of an optimal solution) to perform with respect to one or more goals if possible. While the illustrated embodiment of the routine is performed in a manner local to a particular decision module, such that some or all decision modules may each implement a local version of such a routine, in other embodiments the routine may be implemented in a centralized manner by one or more components with which one or more decision modules interact over one or more networks, such as with a particular decision module indicated to be used at a particular time rather than acting on behalf of the local decision module.

[0095] The illustrated embodiment of the routine 700 begins at block 703, where information or a request is received. The routine continues to block 705 to determine a type of the information or request, and to proceed accordingly. In particular, if a request is received in block 703 to attempt to determine a solution for a current time period given a current model of the local decision module, the routine continues to block 710 to begin to perform such activities, as discussed in greater detail with respect to block 710-790. If it is instead determined in block 705 that a request to relax one or more rules or other restrictions for the current model of the local decision module is received, such as discussed in greater detail

with respect to blocks 760 and 765, the routine continues to block 765. If it is determined in block 705 that a request is received to repair one or more rules or other restrictions for the current model of the local decision module, such as discussed in greater detail with respect to blocks 775 and 780, the routine continues to block 780 to obtain user input to use during the rule repair process (e.g., to interact with a CDD Decision Module Construction component, or to instead interact with one or more users in another manner), such as to allow the current model for the local decision module to later be updated and replaced based on further resulting user actions, or if operation of the target system can be suspended, to optionally wait to further perform the routine 700 until such an updated model is received. If it is instead determined in block 705 that the information or request is of another type, the routine continues instead to block 708 to perform one or more other indicated operations as appropriate, and to then proceed to block 799. Such other indicated operations may include, for example, receiving information about current models and/or control actions proposed or performed by one or more other decision modules that are collectively controlling a target system with the local decision module (such as for use in synchronizing the model of the local decision module with such other decision modules by generating a consensus or converged shared model, as discussed in greater detail with respect to routine 800 of Figures 8A-8B), to receive updates to a model or underlying information for the model for use in ongoing operation of the routine 700 (e.g., from a CDD Decision Module Construction component, such as results from interactions performed in block 780), to receive current state information for the target system, such as for use as discussed in routine 600 of Figures 6A-6B, etc.

[0096] If it determined in block 705 that a request for a solution was received in block 703 for a current time period and based on a current model of the local decision module, the routine continues to block 710 to receive a current set of coupled differential equations that represent the current model for the local decision module of at least a portion of the target system, optionally along with

additional state information for the target system for the current time. The routine then continues to block 715 to determine whether to train or re-train the model, such as if the routine is called for a first time upon initial execution of a corresponding decision module or if error measurements from ongoing operations indicate a need for re-training, as discussed in greater detail with respect to blocks 755, 770 and 730. If it is determined to train or re-train the model, the routine continues to block 720 to determine one or more of the size of a training time window, size of training time slices within the time window, and/or type of training time slices within the training time window, such as in a manner similar to that previously discussed with respect to block 553 of routine 500 of Figures 5A-5B. After block 720, the routine continues to block 725 to use partial initial state information for the target system to train the model, including to estimate values of state variables for the target system that are not known and/or directly observable, by simulating effects of performing control actions over the time window for each of the time slices, as discussed in greater detail with respect to block 555 of routine 500 of Figures 5A-5B.

[0097] After block 725, or if it is instead determined in block 715 not to train or re-train the model, the routine continues to block 730 to perform a piecewise linear analysis to attempt to determine a solution for the current model and any additional state information that was obtained in block 710, with the solution (if determined) including one or more proposed control action determinations for the local decision module to take for a current time period, as well as in some embodiments to use one or more model error gauges to make one or more error measurements with respect to the current model, as discussed in greater detail elsewhere. The routine then continues to block 735 to determine if the operations in block 730 determined a solution within a amount of time allowed for the operation of block 730 (e.g., a defined subset or fraction of the current time period), and if so continues to block 740 to update the current set of coupled differential equations and the resulting current model for the local decision module

to reflect the solution, with the resulting updated information provided as an output of the routine 700.

[0098] If it is instead determined in block 735 that the operations in block 730 did not determine a solution, the routine continues to block 745 to determine if additional time is available within the current time period for further attempts to determine a solution, and if not continues to block 790 to provide output of the routine 700 indicating that no solution was determined for the current time period.

[0099] If additional time is available within the current time period, however, the routine continues to perform blocks 755-780 to perform one or more further attempts to identify the solution - it will be appreciated that one or more of the operations of blocks 755-780 may be repeatedly performed multiple times for a given time period if sufficient time is available to continue further solution determination attempts. In particular, the routine continues to block 755 if additional time is determined to be available in block 745, where it determines whether the measurements from one or more gauges indicate model error measurements that are over one or more thresholds indicating modifications to the model are needed, such as based on the model error measurements from the gauges discussed with respect to block 730. If not, the routine continues to block 760 to determine whether there are one or more rules or other restrictions in the current model that are available to be relaxed for the current time period (that have not previously attempted to be relaxed during the time period, if this is not the first pass through this portion of the routing for the current time period), and if so continues to block 765 to relax one or more such rules or other restrictions and to return to block 730 to re-attempt the piecewise linear analysis with the revised model based on those relaxed rules or other restrictions.

[00100] If it is instead determined in block 755 that the model error measurements from one or more of the gauges are sufficient to satisfy one or more corresponding thresholds, the routine continues instead to block 770 to determine whether to re-train the model based on one or more of the gauges indicating sufficient errors to do so, such as based on accumulated errors over one or more time periods of

updates to the model. If so, the routine returns to block 720 to perform such re-training in blocks 720 and 725, and then continues to block 730 to re-attempt the piecewise linear analysis with the resulting re-trained model.

[00101] If it is instead determined in block 770 not to re-train the model (or if the model was re-trained already for the current time period and the resulting re-attempt in block 730 again failed to find a solution), the routine continues to block 775 to determine whether the model error measurements from one or more of the gauges indicate a subset of one or more rules or other restrictions in the model that potentially have errors that need to be repaired. If so, the routine continues to block 780 to provide information to one or more users via the CDD Decision Module Construction component, to allow the users to revise the rules or other restrictions as appropriate, although in other embodiments some or all such rule repair activities may instead be attempted or performed in an automated manner. After block 780, or if it is instead determined in block 775 not to repair any rules, the routine continues to block 790 to provide an indication that no solution was determined for the current time period. After blocks 740, 708, or 790, the routine continues to block 799 and ends. It will be appreciated that if the routine 700 was instead implemented as a centralized routine that supports one or more decision modules remote from the executing component for the routine, the routine 700 may instead return to block 703 to await further information or requests.

[00102] Figures 8A-8B are a flow diagram of an example embodiment of a CDD Coordinated Control Management routine 800. The routine may, for example, be provided by execution of the component 346 of Figure 3 and/or the components 146a-n of Figure 2, such as to attempt to synchronize current models and their proposed control actions between multiple decision modules that are collectively controlling a target system. In the illustrated embodiment of the routine, the synchronization is performed in a pairwise manner between a particular local decision module's local current model and an intermediate shared model for that decision module that is based on information about the current state of one or more other decision modules, by using a Pareto game technique to determine a

Pareto equilibrium if possible that is represented in a consensus shared model, although in other embodiments other types of synchronization methods may be used. In addition, in the illustrated embodiment, the routine 800 is performed in a local manner for a particular local decision module, such as by being included within that local decision module, although in other embodiments the routine 800 may be implemented in a centralized manner to support one or more decision modules that are remote from a computing system implementing the component for the routine and that communicate with those decision modules over one or more computer networks, such as with a particular decision module indicated to be used at a particular time rather than acting on behalf of the local decision module.

[00103] The illustrated embodiment of the routine 800 begins at block 805, where it waits to receive information or another indication. The routine continues to block 810 to determine if a consensus model or other updated information for another decision module has been received, such as from a copy of the routine 800 executing for that other decision module, and if so continues to block 815 to use the received information to update local intermediate shared model information for use with the local decision module on whose behalf the current copy of the routine 800 is executing, as discussed in greater detail with respect to block 830. If it is instead determined in block 810 that the information or request received in block 805 is not information related to one or more other decision modules, or after block 815, the routine continues to block 820 to determine whether to currently perform a synchronization for the current local model of the local decision module by using information about an intermediate shared model of the local decision module that includes information for one or more other decision modules, such as to do such synchronization each time that an update to the local decision module's model is received (e.g., based on operation of the routine 700 for a copy of the CDD Control Action Determination component local to that decision module) in block 805 and/or each time that information to update the local decision module's intermediate shared model is received in block 805 and used in block 815, or instead as explicitly indicated in block 805 - if the synchronization is to currently be

performed, the routine continues to block 825 and begins to perform blocks 820-880 related to such synchronization activities. Otherwise, the routine continues to block 885 to perform one or more other indicated operations as appropriate, such as to receive requests from the CDD system or other requestor for current information about operation of the routine 800 and/or to provide corresponding information to one or more entities (e.g., to reflect prior requests), etc.

[00104] If it is determined in block 820 that synchronization is to be currently performed, such as based on updated model-related information that is received in block 805, the routine continues to block 825 to obtain a current local model for the local decision module to use in the synchronizing, with the model including one or more proposed control actions to perform for a current time period based on a local solution for the local decision module. The routine then continues to block 830 to retrieve information for an intermediate shared model of the local decision module that represents information for one or more other decision modules (e.g., all other decision modules) that are collectively participating in controlling the target system, with that intermediate shared model similarly representing one or more other proposed control actions resulting from local solutions of those one or more other decision modules, optionally after partial or complete synchronization has been performed for those one or more other decision modules between themselves.

[00105] The routine then continues to block 835 to attempt to determine a consensus shared model that synchronizes the current model of the local decision module and the intermediate shared model by simultaneously providing solutions to both the local decision module's current model and the intermediate shared model. In some embodiments, the operations of block 835 are performed in a manner similar to that discussed with respect to blocks 710-730 of routine 700 of Figure 7A-7B, such as if the local model and the intermediate shared model are combined to create a combination model for whom one or more solutions are to be identified. As discussed in greater detail elsewhere, in some embodiments, the local current model and intermediate shared model may each be represented by a

Hamiltonian function to enable a straightforward creation of such a combined model in an additive manner for the respective Hamiltonian functions, with the operations of routines 600 and/or 700 of Figures 6A-6B and 7A-7B, respectively, similarly representing the models that they update and otherwise manipulate using such Hamiltonian functions.

[00106] After block 835, the routine continues to block 840 to determine whether the operations of block 835 succeeded in an allowed amount of time, such as a fraction or other portion of the current time period for which the synchronization is attempted to be performed, and if so the routine continues to block 845 to update both the local model and the intermediate shared model of the local decision module to reflect the consensus shared model. As earlier noted, if sufficient time is allowed for each decision module to repeatedly determine a consensus shared model with changing intermediate shared models representing one or more other decision modules of a collective group, the decision modules of the collective group may eventually converge on a single converged shared model, although in other embodiments and situations there may not be sufficient time for such convergence to occur, or other issues may prevent such convergence. After block 845, the routine continues to block 850 to optionally notify other decision modules of the consensus shared model determined for the local decision module (and/or of a converged shared model, if the operations of 835 were a last step in creating such a converged shared model), such as if each of the notified decision modules is implementing its own local version of the routine 800 and the provided information will be used as part of an intermediate shared model of those other decision modules that includes information from the current local decision module's newly constructed consensus shared model.

[00107] If it is instead determined in block 840 that a synchronization did not occur in the allowed time, the routine continues to perform blocks 860-875 to re-attempt the synchronization with one or more modifications, sometimes repeatedly if sufficient time is available, and in a manner similar to that discussed with respect to blocks 745-780 of routine 700 of Figures 7A-7B. In the illustrated example, the

routine determines in block 860 if additional time is available for one or more such re-attempts at synchronization, and if not the routine continues to block 880 to provide an indication that no synchronization was performed within the allowed time. Otherwise, the routine continues to block 870 to take one or more actions to perform one or more of relaxing rules or other restrictions, repairing rules or other restrictions, and/or re-training a model, with respect to one or both of the current model of the local decision module and/or one or more other decision modules whose information is represented in the intermediate shared model of the local decision module. If it is determined in block 870 to proceed in this manner, the routine continues to block 875 to perform corresponding actions, sometimes one at a time in a manner similar to that discussed with respect to routine 700, including to cause resulting updates to the current model of the local decision module and/or to the local intermediate shared model of the local decision module, after which the routine returns to block 835 to re-attempt to synchronize the local model and the intermediate shared model of the local decision module.

[00108] If it is instead determined in block 870 that no further actions are to be performed with respect to relaxation, repair and/or re-training, the routine continues instead to block 880. After blocks 850, 880 or 885, the routine continues to block 895 to determine whether to continue, such as until an explicit indication to terminate or suspend operation of the routine 800 is received, such as to reflect an end to operation of the target system and/or an end to use of the local decision module and/or a collective group of multiple decision modules to control the target system. If it is determined to continue, the routine returns to block 805, and otherwise continues to block 899 and ends.

[00109] Figure 9 illustrates a flow diagram of an example embodiment of a routine 900 performed for a representative generic target system, with respect to interactions between the target system and one or more decision modules that are controlling at least a portion of the target system. The routine may, for example, be provided by execution of a target system 360 and/or 370 of Figure 3, and/or a target system 160 and/or 170 of Figures 1 and 2, such as to implement operations

specific to the target system. It will be appreciated that the illustrated embodiment of the routine focuses on interactions of the target system with the one or more decision modules, and that many or all such target systems will perform many other operations in a manner specific to those target systems that are not illustrated here for the purpose of brevity.

[00110] The routine begins at block 910, where it optionally provides initial state information for the target system to a CDD system for use in an automated control system of the CDD system for the target system, such as in response to a request from the CDD system or its automated control system for the target system, or instead based on configuration specific to the target system (e.g., to be performed upon startup of the target system). After block 910, the routine continues to block 920 to receive one or more inputs from a collective group of one or more decision modules that implement the automated control system for the target system, including one or more modified values for or other manipulations of one or more control elements of a plurality of elements of the target system that are performed by one or more such decision modules of the automated control system. As discussed in greater detail elsewhere, the blocks 920, 930, 940 may be repeatedly performed for each of multiple time periods, which may vary greatly in time depending on the target system (e.g., a microsecond, a millisecond, a hundredth of a second, a tenth of a second, a second, 2 seconds, 5 seconds, 10 seconds, 15 seconds, 30 seconds, a minute, 5 minutes, 10 minutes, 15 minutes, 30 minutes, an hour, etc.).

[00111] After block 920, the routine continues to block 930 to perform one or more actions in the target system based on the inputs received, including to optionally produce one or more resulting outputs or other results within the target system based on the manipulations of the control elements. In block 940, the routine then optionally provides information about the outputs or other results within the target system and/or provides other current state information for the target system to the automated control system of the CDD system and/or to particular decision modules of the automated control system. The routine then continues to block

995 to determine whether to continue, such as until an explicit indication to terminate or suspend operation of the target system is received. If it is determined to continue, the routine returns to block 920 to begin a next set of control actions for a next time period, and otherwise continues to block 999 and ends. As discussed in greater detail elsewhere, state information that is provided to a particular decision module may include requests from external systems to the target system, which the automated control system and its decision modules may determine how to respond to in one or more manners.

[00112] The following sections describe a variety of specific, non-exclusive embodiments in which some or all of the described techniques may be implemented. It will be appreciated that particular details of particular embodiments may not be included in or true for all embodiments, and that the described embodiments may be implemented individually or in combination with any and all other combinations of other embodiments.

[00113] The following discusses several non-exclusive example embodiment, in which one or more specified embodiments of the Collaborative Distributed Decision system are each referred to as a Collaborative Distributed Inferencing (CDI) system or a Cooperative Distributed Inferencing Platform (CDIP), in which one or more specified embodiments of the Decision Module Construction component are each referred to as the “Rules Builder” or including or being performed by a “Rule Conversion Engine” (RCE) or a “CDL Compiler” or a “Rule(s) Entry Interface” (REI) or a “Rules and Goals UI”, in which one or more specified embodiments of the Control Action Determination component are each referred to as having “chattering” subcomponents or performing “chattering” functionality or including or being performed by a “Query Response Engine” (QRE) or an “Optimization Element” or via “Hamiltonian-Chattering Control”, in which one or more specified embodiments of the Coordinated Control Management component are each referred to as having or performing “mean field” information or functionality or including or being performed by a “Tellegen” agent or a “Pareto Multi-Criteria Optimization Engine” (PMOE) or a “Pareto element”, in which

decision modules are referred to as “agents” or “peer agents” or “agent nodes” or “decision elements”, in which an automated control system may include a “cluster” of agents and in some cases is referred to as a “distributed architecture”, in which a target system is referred to as an “application domain”, in which a decision module’s stored information about a target system includes an “Internal Heterogeneous Database” (IHDB) and/or a “Domain Rules Knowledge Base”, in which a decision module’s consensus shared model is referred to as an “internal optimum” generated using mean field information, in which changes propagated from one decision module to others is referred to as a “delta”, etc.

[00114] CDI is built using a Peer-to-Peer (P2P) based distributed architecture that allows for partitioning the overall optimization problem into smaller sub-tasks or work-loads between peer agents. CDI Peer Agents are equally privileged participants in the application domain. They are configured to form a peer-to-peer network of nodes. This allows each agent in the network to solve the problem independently, using its own internal knowledge base. Each agent also internally engages in Pareto game playing to reach internal optimum before sharing changes with other agents. The agents then communicate the mean field changes with other agents in the network using a gossip protocol to synchronize their internal mean field approximation in an eventually consistent fashion. Figure 10 illustrates a network diagram 1000 of a portion of a distributed architecture of an example CDI system.

CDI Cluster Setup

[00115] When system is initially configured certain agents are tagged as seed-nodes in the network. The seed agent nodes can be started in any order and it is not necessary to have all the seed agent nodes running. When initializing the CDI cluster at least one seed agent node is preferably running, otherwise the other CDI seed nodes may not become initialized and other CDI agent nodes might not join the cluster.

CDI Agent Cluster Registry

[00116] Each agent has a built-in registry of active CDI agent nodes in the cluster. This agent registry is started on all CDI agent nodes and is updated with every change in cluster membership including new agents joining the cluster, agents leaving the cluster, agent timeouts etc. Agents inform each other of active membership through a heartbeat mechanism. The registry is eventually consistent, i.e. changes may not immediately be visible at other nodes, but typically they will be fully replicated to all other nodes after a few seconds. The changes are propagated using the change deltas and are disseminated in a scalable way to other nodes with a gossip protocol.

CDI Cluster Agent Failure Detection

[00117] The CDI agent nodes in the cluster monitor each other by sending heartbeats to detect if a CDI agent node is unreachable from the rest of the CDI cluster. The heartbeat arrival times are interpreted by an implementation of the Phi Accrual Failure Detector (Hayashibara, Défago, Yared, & Katayama, 2004).

[00118] The suspicion level of failure is given by a value called phi. The basic idea of the phi failure detector is to express the value of phi on a scale that is dynamically adjusted to reflect current network conditions.

The value of phi is calculated as:

$$\text{Phi} = -\log_{10} (1 - F(\text{timeSinceLastHeartbeat}))$$

F is the cumulative distribution function of a normal distribution with mean and standard deviation estimated from historical heartbeat inter-arrival times.

CDI Cluster Agent Network Partition

[00119] Once a CDI agent node becomes network partitioned it will stop receiving mean field updates from the other agents in the Cluster. This however, should not prevent it from continuing to perform local optimizations based on its internal knowledge base, local mean field and sensor inputs that it will continue to receive.

[00120] The Rules Builder contains the following components:

- Rules Entry Interface
- Validator

- CDL Compiler
- Step Processor
- Workflow Remote Control
- Compiled Rules Engine Artifact

And interacts directly with the following components:

- Master Agent.

[00121] The Master Agent is responsible for interaction with all Chattering components, including the following:

- System Trainer & Bootstrapper
- Runner

The System also utilizes a Persistent Store to access state and necessary models/data. Figure 11 illustrates a network diagram 1100 of a portion of an example Rules Builder component.

Rules Entry Interface:

[00122] The Rules Entry Interface is responsible for receiving rules in a syntax familiar to a domain expert. The entry interface is a text entry tool where a domain expert would insert the rules, goal and system information (rules script(s)) to provide continuous control suggestions for a given problem. The entered rules script(s) would be composed in the CDI Domain-Specific Language (DSL). The CDI DSL facilitates functional translation of the entered rules script(s) into a problem definition which houses the control definitions (measurable variable with ranges allowed and any associated constraints on the control variable).

For example:

[00123] In the entry interface using the CDI DSL one would specify an upper and lower bound using the following syntax first defining a rule, with the parameters and the rule logic that evaluates against the system state.

```
rule id: "max production", params: ["production"], { it <= 15.0 }
rule id: "min production", params: ["production"], { it >= -50.0 }
control var: "production", min: "min production", max: "max production"
```

[00124] In the entry interface using the CDI DSL one would specify the dynamics (variables that change with relation to other measurable variables and the definition of this change)

delta var: "inventory", params: ["production", "demand"], {p, d -> p - d}

[00125] In the entry interface using the CDI DSL one would specify the goal (an objective i.e., to minimize or maximize a relationship between the control and controllable dynamics).

goal objective: "minimize", params: ["inventory", "production"], {inventory, production -> ((inventory - 10)*(inventory - 10)) + (production * production) }

[00126] Finally, the entry interface is also responsible for facilitating a domain expert to provide certain settings that are used by the system as well as providing user workflow steps. The settings provided represent key: value terms used by the system and are available throughout by the Settings Provider encapsulated within the Rules Engine. An domain expert might write initial values of the following form:

settings << [initialState: "20", initialCoState: "0", terminalCoState: "0", numChatteringLevels: "9", horizon: "3", delta: "0.1", iterations: "15"]

CDL Compiler:

[00127] The CDL Compiler is responsible for the conversion of said rules into evaluable constraints, system information and an optimization goal. The CDI Compiler translates the entered script(s) into a problem definition. A problem definition is composed of labeled rules. A rule is a tuple comprised of a unique name and a Term. A Term is an evaluable function where the logic was authored as previously described above, the input to which contains a representation of the system state (StateMap).

Validator:

[00128] The Validator is responsible for the validation of converted constraints with regards to restrictions the optimization problem needs to satisfy as a prerequisite to its solution and/or reaching a solution quality threshold.

[00129] The converted constraints defined in the problem definition, along with the settings provider, is validated for controllability, observability, stability and goal completeness.

controllability -- this check ensures that every control variable has a defined rule relating it to one or more dynamic state variables. The above example satisfies this check since our single control variable 'production' is used in the dynamic definition of 'inventory';

observability -- checks the statemap against the defined rules in the script(s). An observability check does not impede the solution of a problem, however if failing, represents that our problem may not be well defined;

stability -- this check ensures that our system will converge over time on a solution, which can be tested in the testing stage; and

goal completeness -- ensures that every control variable appears in the goal.

Compiled Rules Engine:

[00130] The conversion of rule scripts into mathematical expressions that can be yielded through an explicit contract to the chattering agent for use in solving the optimization goal, manifests in the compiled artifact that we label the Compiled Rules Engine

[00131] The Rules Engine defines and implements several interfaces for acquiring the values of these mathematical expressions, as later described with respect to “chattering” components. The Rules Engine is compiled as an artifact available to the Chattering component by the Compiler and made available for use in continuous processing. Figure 12 illustrates a network diagram 1200 of example sub-components of an example Rules Builder component, in which compilation is dependent upon successful Validation of the observability, completeness, controllability, and workflow step validation. Any errors/warnings are returned to the domain expert for correction. Once corrections/improvements are complete, a new Compiled Rules Engine artifact is produced along with new commands delivered to the Step Processor.

Steps Processor and Remote Control:

[00132] Furthermore, the rules builder is also responsible for facilitating the workflow steps an individual domain expert would need to undertake before deploying such a system. These steps include:

- training an optimization system using the converted rules, with bootstrapped sensor data;
- testing the resulting control actuator inputs (the end results of the chattering optimization); and
- persisting the trained model for use in the running of the overall system.

[00133] The workflow component allows for training and testing a system. This is integrated into the DSL in the form of a workflow step builder.

[00134] For example,

```

workflow {
    train.onComplete {
        persist: model, '/tmp/model'
        run.withSensorData('/tmp/simulatedData').after(10 minutes)
        {
            persist: results, '/tmp/results'
            shutdown
        }
    }
}

```

[00135] Here a domain expert instructs the default bootstrapper to be used in loading predefined sensor data, train the system on this data, persist the trained model output to the file '/tmp/model', then once trained to begin running the system (as a single agent). The inputs are tested by analyzing the results from the run. The artifacts and model information are persisted to a persistent store. The Steps Processor facilitates the dispatch of interpreted workflow steps to the Remote Control for delegation to the Master Agent of a running system for processing.

[00136] Figure 13 illustrates a network diagram 1300 of interactions of portions of a Rules Builder component with an executing agent, including the Steps processor,

the Remote Control and interactions with a running agent (the Step Processor handles interpretation of workflow steps, and the steps are delivered for dispatch by the Remote Control, which passes commands to the Master Agent; the Master Agent handles communication between a running system's controlled elements and the Remote Control, and examples of the Controlled Elements are the Trainer and Runner).

[00137] Figure 14 illustrates a network diagram 1400 of interactions of portions of a Rules Builder component with chattering components, including a Running Agent that delivers messages and interacts with the chattering components to facilitate training and running the system (the Trainer persists its model once training has completed; The Trainer invokes a Bootstrapper for acquisition of training data; The Trainer utilizes the Chattering Library; The Trainer utilizes the Chattering Library; Chattering utilizes the compiled rules; Chattering persists its control suggestions; Bootstrapper persists training data).

[00138] With respect to the discussion below, Figure 15 illustrates a diagram 1500 of a sliding window for use with chattering, and is referred to in the text for the example embodiment as "Figure 1", while Figure 16 illustrates a diagram 1600 of using Lebesgue integration to approximate a control trajectory for chattering, and is referred to in the text for the example embodiment as "Figure 2".

1. Parametric chattering control over a rolling horizon

The real-time problem we would like to solve is really over an infinite time horizon in real-time, but our approach is to consider a window of fixed length, and then apply invariant imbedding as we increase the window length to achieve an optimal relaxed control. The chattering control is applied to the window of fixed length, and results in a relatively easily solved problem. Then the window slides to a point where, perhaps, a new measurement is available, or a new control action is applicable. And the algorithm iterates in this manner.

To develop our chattering control method, we first start with a classical, deterministic control problem of the form,

$$\begin{aligned}
 & \min_{u(t) \in U} \int_{t_0}^T \mathcal{L}(x(t), u(t)) dt + \psi(x(T)) & (1) \\
 & \text{s.t. } x(t) = x(t_0) + \int_{t_0}^t f(x(\tau), u(\tau)) d\tau \\
 & \text{or alternatively} \\
 & \text{s.t. } \dot{x}(t) = f(x(t), u(t)) \\
 & \text{for } t_0 \leq t \leq T
 \end{aligned}$$

where $x(t) \in \mathbb{R}^n$, $u(t) \in U \subset \mathbb{R}^m$, $\mathcal{L}$ is twice continuously differentiable with respect to x and continuous with respect to u , $\psi(x(T)) : \mathbb{R}^n \rightarrow \mathbb{R}$, and is continuous and twice continuously differentiable with respect to x . Also, $f(x(t), u(t))$ is twice continuously differentiable with respect to x and u . The time horizon T is assumed to be finite and known. We also assume initial conditions $x(t_0)$ are known.

The objective is to get the behavior at T based on information from the recent past. We develop discrete time windows with dynamics of recent

past, update the state with sensory information, and construct an open loop feedback strategy.

We reformulate Problem (1) to a standard form. First we introduce a new variable $x_{n+1}(t)$, with $x_{n+1}(t_0) = 0$, and defined by

$$\dot{x}_{n+1}(t) = \mathcal{L}(x(t), u(t)). \quad (2)$$

Note that, for $t_0 \leq t \leq T$,

$$x_{n+1}(t) = \int_{t_0}^t \mathcal{L}(x(\tau), u(\tau)) d\tau.$$

Further, convert the terminal term in the criterion by defining a state variable $y(t) \in \mathbb{R}^1$, with $y(t_0) = 0$, and

$$\dot{y}(t) = \frac{\partial \psi(x(t))}{\partial x(t)} f(x(t), u(t)) \quad (3)$$

for $t_0 \leq t \leq T$. Note that $y(T) = \psi(x(T))$ for any trajectory since

$$\dot{y}(t) = \frac{\partial \psi(x(t))}{\partial x(t)} \dot{x}(t).$$

The original control problem (1) is now of the form

$$\min_{u(t) \in U} x_{n+1}(T) + y(T)$$

$$\begin{aligned} \text{s.t. } x_{n+1}(t) &= \int_{t_0}^t \mathcal{L}(x(\tau), u(\tau)) d\tau \\ y(t) &= \int_{t_0}^t \frac{\partial \psi}{\partial x} f(x(\tau), u(\tau)) d\tau \\ x(t) &= x(t_0) + \int_{t_0}^t f(x(\tau), u(\tau)) d\tau \end{aligned}$$

For ease of notation, we create an $n+2$ dimensional state vector,

$$z(t) = \begin{bmatrix} x_{n+1}(t) \\ x(t) \\ y(t) \end{bmatrix}$$

and denote the dynamics in integral form as

$$z(t) = z(t_0) + \int_{t_0}^t F(z(\tau), u(\tau)) d\tau$$

or, if the dynamics are differentiable, as

$$\dot{z}(t) = F(z(t), u(t)) = \begin{bmatrix} \dot{x}_{n+1}(t) \\ \dot{x}(t) \\ \dot{y}(t) \end{bmatrix} = \begin{bmatrix} \mathcal{L}(x(t), u(t)) \\ f(x(t), u(t)) \\ \frac{\partial \phi}{\partial x} f(x(t), u(t)) \end{bmatrix}.$$

Note that the state vector $z(t)$ captures the original dynamics, criterion and terminal cost.

Without loss of generality, we now consider the following optimal control problem

$$\begin{aligned} \min_{u(t) \in U} \quad & Qz(T) \\ \text{s.t. } z(t) \quad &= z(t_0) + \int_{t_0}^t F(z(\tau), u(\tau)) d\tau \end{aligned} \quad (4)$$

where Q is a row vector of dimension $n+2$ with $Q = [1 \ 0_{1 \times n} \ 1]$, so that $Qz(T) = x_{n+1}(T) + y(T)$. We assume $z(t_0)$ is known (since $x(t_0)$ is assumed known, $x_{n+1}(t_0) = 0$, and $y(t_0) = 0$).

For the rolling horizon approach, we consider the initial problem on a window $[t_0, \gamma]$, partitioned into N sufficiently small intervals of length Δ . We let Δ be the same for all intervals for ease of notation. Note that $t_0 + N\Delta = \gamma$. We get a sequence of windows as the window moves, and the k th window is $[t_0 + k\Delta, \gamma + k\Delta]$. See Figure 1 for an illustration of the sliding window.

We reformulate the control problem specified in (4) for the k th window, as follows,

$$\begin{aligned} \min_{u(t) \in U} \quad & Qz(\gamma + k\Delta) \\ \text{s.t. } z(t) \quad &= z(t_0 + k\Delta) + \int_{t_0 + k\Delta}^t F(z(\tau), u(\tau)) d\tau \\ &\text{for } t_0 + k\Delta \leq t \leq \gamma + k\Delta. \end{aligned} \quad (5)$$

We let $k = 0, 1, \dots$, and consider the end of each window as corresponding to real-time.

Our approach is to *chatter* on the control, in a similar manner as in Kohn et al. (2010). In this paper, we use a probability distribution to derive a chattering control in a probability space, and find an approximation to the relaxed control problem.

The relaxed form of the control problem for the k th window specified in (5) is

$$\begin{aligned} \min_{0 \leq \alpha(t) \leq 1} \quad & Qz(\gamma + k\Delta) \\ \text{s.t. } \quad & z(t) = z(t_0 + k\Delta) + \int_{t_0+k\Delta}^t \int_{\Omega} F(z(\tau), c) d\alpha d\tau \\ & \int_{\Omega} d\alpha = 1 \end{aligned} \quad (6)$$

where c is the control distribution defined by constructing the control probability distribution α . The resulting α^* provides the optimal distribution that solves the problem in (5).

Problem (6) can be viewed as a *relaxed optimal control problem* with respect to the Young measure.

Note that Δ and γ must be sufficiently small, so that F remains continuous and differentiable in z , and measurable in the control.

The idea underlying the chattering control is to approximate the control trajectory using Lebesgue integration instead of the more traditional Riemann integration. See Figure 2.

The chattering approximation to the relaxed problem (6) for the k th window is

$$\begin{aligned} \min_{0 \leq \alpha_i(t) \leq 1, \forall i \text{ and } t} \quad & Qz(\gamma + k\Delta) \\ \text{s.t. } \quad & z(t + \Delta) = z(t) + \Delta \sum_{i=1}^I F(z(t), \alpha_i) \alpha_i(t) \\ & \sum_{i=1}^I \alpha_i(t) = 1 \quad \text{for } t = t_0 + k\Delta, t_0 + k\Delta + \Delta, \dots, \gamma + k\Delta - \Delta \end{aligned} \quad (7)$$

where α_i for $i = 1, \dots, I$ is the i^{th} quantization level of $u(t)$ at time t , providing specific control levels for the Lebesgue integration, and I is the number of levels in the interval. For ease of notation, we let the number of levels I be the same over the whole time interval. The sum of α_i over i must equal one,

for all t , since α_i is representing a probability. Also, upper and lower bounds on α_i must be satisfied, i.e., $0 \leq \alpha_i \leq 1$ for all i and all t .

Notice that $c_i(t)$ has the same dimension as $u(t)$. And the number of levels I must be large enough to cover the range of $u(t)$. For example, if $u(t)$ is in two dimensions, and the number of levels on dimension 1 is three, and the number of levels on dimension 2 is two, then the number of levels in $c_i(t)$ is six.

If we know $z(t_0 + k\Delta)$, we could solve the chattering problem (7) for α_i at discrete times $t_0 + k\Delta, t_0 + k\Delta + \Delta, \dots, \gamma + k\Delta - \Delta$ using an optimization engine, such as FMINCON. This involves evaluating F at all discrete times and c_i levels, so we only want to solve this chattering problem occasionally. Instead of solving it frequently, we develop continualized incremental dynamics and an incremental optimization that has less computation. Then we only solve (7) when we have a large change in values or the accumulated error grows to be too large.

Also, if we know $z(t_0 + k\Delta)$, we can easily solve the one-time period chattering problem for $\alpha_i(t_0 + k\Delta)$. For a single time period, the chattering problem (7) reduces to

$$\begin{aligned} \min_{\alpha_i(t_0 + k\Delta)/\alpha} \quad & Q \left(z(t_0 + k\Delta) + \Delta \sum_{i=1}^I F(z(t_0 + k\Delta), \alpha) \alpha_i(t_0 + k\Delta) \right) \quad (8) \\ & \sum_{i=1}^I \alpha_i(t_0 + k\Delta) = 1 \\ & 0 \leq \alpha_i(t_0 + k\Delta) \leq 1. \end{aligned}$$

Notice that one-time period chattering problem (8) is very easy to solve, since it is a type of relaxed knapsack problem, so the optimal α is determined by rank ordering the coefficients in the objective (since the coefficients in the constraint are all equal to one). Also note that Q only has two non-zero values, so only the first and last elements of F need to be evaluated (at $z(t_0 + k\Delta)$ and all c_j).

Hence we know analytically that the optimal α_i is given by,

$$\alpha_i(t_0 + k\Delta)^* = \begin{cases} 1 & \text{if } i \text{ provides the smallest value} \\ & \text{of all } QF(z(t_0 + k\Delta), c_j), j = 1, \dots, I, \\ 0 & \text{otherwise.} \end{cases} \quad (9)$$

2. Incremental Optimization and Continualized Incremental Dynamics

We start by adding an argument to the variables in (7) to indicate they are associated with the k th window problem. For instance, $z(t, k)$ indicates the state at time t in the k th window. We also linearize with respect to F , yielding

$$z(t, k) = z(t, k-1) + \delta z(t, k) + O(\Delta^2) \quad (10)$$

for $t_0 + k\Delta \leq t \leq \gamma + (k-1)\Delta$. We use a continualization approach [7] to yield

$$\delta z(t, k) = \sum_{i=1}^I \frac{\partial F(z(t_0 + k\Delta, k), c_i)}{\partial z} \alpha_i(t, k) \delta z(t, k)$$

where $\delta z(t, k)$ is the change in $z(t, k)$.

For most of the systems we are considering, the value of the Jacobian matrix, $\frac{\partial F(z(t_0 + k\Delta, k), c_i)}{\partial z}$, changes slowly with the state, so we can approximate it by its value at the state at the beginning of the time window $z(t_0 + k\Delta, k)$. Thus, the matrix is only computed at the beginning of the time window. It also could be computed using finite differencing.

With this approximation, the incremental optimization problem with continualized incremental dynamics is given by,

$$\begin{aligned} \min_{\{\alpha_i(t, k)\}} \quad & Q\delta z(\gamma + k\Delta, k) \\ \text{s.t. } \delta z(t, k) = \quad & \sum_{i=1}^I \frac{\partial F(z(t_0 + k\Delta, k), c_i)}{\partial z} \alpha_i(t, k) \delta z(t, k) \\ & \sum_{i=1}^I \alpha_i(t, k) = 1 \\ & 0 \leq \alpha_i(t, k) \leq 1. \end{aligned} \quad (11)$$

From Pontryagin's minimum principle for the incremental problem (11), the three necessary conditions for optimality are,

1.

$$\delta \dot{z}^*(t, k) = \left(\sum_{i=1}^I \frac{\partial F(z(t_0 + k\Delta, k), c_i)}{\partial z} \alpha_i(t, k) \right) \delta z^*(t, k) \quad (12)$$

with initial condition $\delta z(t_0 + k\Delta)$

2.

$$\dot{p}(t, k) = - \left(\sum_{i=1}^I \frac{\partial F(z(t_0 + k\Delta, k), c_i)}{\partial z} \alpha_i(t, k) \right) p(t, k) \quad (13)$$

with terminal condition $p(\gamma + k\Delta) = Q^T$. Notice that the p equation is not incremental (p , not δp), but it includes an approximation because the Jacobian matrix is evaluated at the beginning of the window.

3.

$$\begin{aligned} p^T(t, k) \sum_{i=1}^I \frac{\partial F(z(t_0 + k\Delta, k), c_i)}{\partial z} \alpha_i^*(t, k) \delta z^*(t, k) \\ \leq p^T(t, k) \sum_{i=1}^I \frac{\partial F(z(t_0 + k\Delta, k), c_i)}{\partial z} \alpha_i(t, k) \delta z(t, k) \end{aligned}$$

for α satisfying $\sum_{i=1}^I \alpha_i(t, k) = 1$ and $0 \leq \alpha_i(t, k) \leq 1$.

Solving (12) and (13) will allow us to propagate δz over the length of the window, and then update the Jacobian matrix when the error reaches a threshold value.

[00139] The following discusses an example implementation that may be used for such chattering.

Setup

The optimization problem we're solving is the following:

$$\begin{aligned} \min_u \quad & \int_{t_0}^T \mathcal{L}(x(t), u(t)) dt + \psi(x(T)) \\ \text{s.t.} \quad & \dot{x}(t) = f(x(t), u(t)) \end{aligned}$$

The function $\mathcal{L}$ is the objective function; this comes from adding the original objective function specified in the DSL with penalty terms for the rules.

The function f specifies the dynamics of the system; these are specified in the DSL.

The variable x represents the state of the system at time t ; the variable u is the control (to be found) at time t .

Finally, ψ represents the cost at being in the terminal state $x(T)$. This term is typically set to 0, but does not have to be.

The rules-builder engine creates code that allows us to compute functions used in this system, specifically

$$\begin{aligned} & \mathcal{L}(x(t), u(t)) \\ & f(x(t), u(t)) \\ & \psi(x(t)) \end{aligned}$$

For these algorithms, we introduce two new (scalar) variables

$$\begin{aligned} x_{n+1}(t) &= \int_{t_0}^t \mathcal{L}(x(\tau), u(\tau)) d\tau \\ y(t) &= \int_{t_0}^t \frac{\partial \psi}{\partial x} f(x(\tau), u(\tau)) d\tau \end{aligned}$$

Let z be the vector

$$z(t) = \begin{bmatrix} x_{n+1}(t) \\ x(t) \\ y(t) \end{bmatrix}$$

Define the function $F(x, u, t)$ as

$$F(x(t), u(t)) = \begin{bmatrix} \mathcal{L}(x(t), u(t)) \\ f(x(t), u(t)) \\ \frac{\partial \mathcal{L}(x(t))}{\partial x(t)} f(x(t), u(t)) \end{bmatrix}$$

Also define the matrix $Q = [10_{1 \times n} 1]$.

Observe that all entries of F may be computed at any time based on the functions provided by the rule builder. (Use finite differencing to compute $\frac{\partial \mathcal{L}(x(t))}{\partial x(t)}$.)

Chattering – Offline Training

The notation here is the same as in the paper *Parametric Chattering Control of Dynamic Systems*.

Inputs:

- the functions $\mathcal{L}$, f , and ϕ .
- a time horizon T
- the time step Δ
- a time window size γ
- the initial state $x(0)$
- a guess $\dot{x}_0$ for the initial rate of change in the state x
- the terminal costate $p(T)$, typically assumed to be 0.
- $u_{\min}$ and $u_{\max}$, the min and max possible values for each control u
- c , the number of control levels for each control u

Outputs:

Let $N = T/\Delta$ be the number of intervals.

- the optimal controls $u(t)$ for each $t = 0, \Delta, 2\Delta, \dots, N\Delta$.
- the costate (assuming these controls) $p(t)$ for each $t = 0, \Delta, 2\Delta, \dots, N\Delta$.
- the state (corresponding to picking optimal controls) $x(t)$ for each $t = 0, \Delta, 2\Delta, \dots, N\Delta$.
- status information such as the number of iterations, whether there was a failure, etc.

Steps:

Algorithm 1 Finding optimal control within window k

- 1: for each component $j = 1, \dots, k$, construct r_j control levels by linearly interpolating from $u_{\min,j}$ to $u_{\max,j}$
- 2: create a list all possible control combinations of each level for each control $j = 1, \dots, k$. There are $I = \prod_{j=1}^k r_j$ total such combinations. Let c_i be the i th such control combination, for $i = 1, \dots, I$.
- 3: Solve the following problem

$$\begin{aligned}
 \min_{\alpha} \quad & Qz(\gamma + k\Delta) \\
 \text{s.t.} \quad & z(t + \Delta) = z(t) + \Delta \sum_{i=1}^I F(z(t), c_i) \alpha_i(t) \\
 & \sum_{i=1}^I \alpha_i(t) = 1 \quad \text{for } t = t_0 + k\Delta, \dots, \gamma + k\Delta - \Delta \\
 & 0 \leq \alpha_i(t) \leq 1 \quad \text{for all } i, t
 \end{aligned}$$

Note this is a nonlinear optimization problem and is solved using a nonlinear solver such as FMINCON.

- 4: the optimal control at each time is given by $\sum_{i=1}^I \alpha_i c_i$
- 5: find $p(t)$ at every time from $t = t_0$ to γ via numerically integrating the following ordinary differential equation backwards in time (using any standard solver):

$$\begin{aligned}
 p(\gamma, 0) &= Q \\
 p(t, k) &= - \left(\sum_{i=1}^I \frac{\partial F(z(t_0 + k\Delta, k), c_i)}{\partial z} \alpha_i(t, k) p(t, k) \right)
 \end{aligned}$$

- 6: find $\delta z(t)$ at every time from $t = t_0$ to γ by numerically integrating the following ordinary differential equation forwards in time:

$$\begin{aligned}
 \delta z(0, 0) &= \delta z_0 \\
 \delta z(t, k) &= \sum_{i=1}^I \frac{\partial F(z(t_0 + k\Delta, k), c_i)}{\partial z} \alpha_i(t, k) \delta z(t, k)
 \end{aligned}$$

Note that the matrix $\frac{\partial F(z(t_0 + k\Delta, k), c_i)}{\partial z}$ is the same between this step and the previous, so it doesn't have to be recomputed.

- 7: update the state z at every time using

$$z(t, k) = z(t, k-1) + \delta z(t, k)$$

- 8: return the state z , the change in state δz , the chattering levels α , the controls u and the costate p at every time in the window
-

Algorithm 2 Chattering offline training

- 1: Solve the chattering problem on the first window with $k = 0$ using Algorithm 1.
- 2: **for** $k = 2, 3, \dots$ (where k is the window number) **do**
- 3: Solve the incremental optimization problem to get the optimal controls at the end of the window

$$\begin{aligned}
& \min_{\alpha} \quad Q\delta z(\gamma + k\Delta, k) \\
& \text{s.t.} \quad \delta \dot{z} = \sum_{i=1}^I \frac{\partial F(z(t_0 + k\Delta, k), c_i)}{\partial z} \alpha_i(t, k) \delta z(t, k) \\
& \quad \sum_{i=1}^I \alpha_i(t) = 1 \\
& \quad 0 \leq \alpha_i(t) \leq 1 \quad \text{for all } i
\end{aligned}$$

(This is a much easier optimization problem than finding the controls in Algorithm 1 as it is only for a single time step.)

- 4: update our estimates of $p(t)$ at every time in the window by numerically integrating the following ODE backward:

$$\begin{aligned}
& p(\gamma, 0) = Q \\
& \dot{p}(t, k) = - \left(\sum_{i=1}^I \frac{\partial F(z(t_0 + k\Delta, k), c_i)}{\partial z} \alpha_i(t, k) p(t, k) \right)
\end{aligned}$$

- 5: update $\delta z(t)$ at every time in the window by numerically integrating the following ODE forward:

$$\begin{aligned}
& \delta z(0, 0) = \delta z_0 \\
& \delta \dot{z}(t, k) = \sum_{i=1}^I \frac{\partial F(z(t_0 + k\Delta, k), c_i)}{\partial z} \alpha_i(t, k) \delta z(t, k)
\end{aligned}$$

- 6: update the state z at every time using

$$z(t, k) = z(t, k-1) + \delta z(t, k)$$

- 7: **end for**
 - 8: **return** the state z , the change in state δz , the chattering levels α , the controls u and the costate p at every time from 0 to T .
-

Chattering – Online Prediction

Inputs:

- the hamiltonian function H
- the time step Δ
- the current state $x(t)$
- the current costate $p(t)$
- $u_{\min}$ and $u_{\max}$, the min and max possible values for each control u
- c , the number of control levels for each control u

Outputs:

- the optimal control $u(t)$
- the new state $x(t + \Delta)$
- the new costate $p(t + \Delta)$

Steps:

Algorithm 3 Chattering online updating

- 1: given $x(t)$ and $p(t)$ at the current time t .
- 2: compute optimal control $u(t)$ by solving:

$$\begin{aligned} \min_{\alpha} \quad & Q \left(x(t_0 + k\Delta) + \Delta \sum_{i=1}^I F(x(t_0 + k\Delta), \alpha_i(t_0 + k\Delta)) \right) \\ \text{s.t.} \quad & \sum_{i=1}^I \alpha_i(t_0 + k\Delta) = 1 \\ & 0 \leq \alpha_i(t_0 + k\Delta) \leq 1 \end{aligned}$$

This problem has a closed-form solution, as mentioned in the paper.

- 3: $u(t) = \sum_{i=1}^I \alpha_i v_i$
- 4: update $\delta x(t)$ over the last Δ using

$$\begin{aligned} \delta x(0, 0) &= \delta x_0 \\ \delta x(t, k) &= \sum_{i=1}^I \frac{\partial F(x(t_0 + k\Delta, k), v_i)}{\partial x} \alpha_i(t, k) \delta x(t, k) \end{aligned}$$

- 5: update the state x at every time using

$$x(t, k) = x(t, k-1) + \delta x(t, k)$$

- 6: return $x(t_{s+1}), u(t)$
-

Chattering – Computing optimal values for T and Δ

Inputs

- The function $F(x, u, t)$ (from the rule builder).
- A required error tolerance (provided in the DSL by the customer).
- an initial guess for T and Δ

Outputs

- Δ , the time step to use
- T , the time horizon

Steps

Algorithm 4 Finding Δ and T

- 1: Solve the offline chattering problem from $t = 0$ to $t = T$ to get controls and state for every time from $t = 0$ to T .
 - 2: Compute the Jacobian matrices $A(t) = \frac{\partial F(x(t))}{\partial x(t)}$ at each time $t = 0$ to $t = T$ using finite differencing.
 - 3: for $i = 1, 2, \dots, N$ do
 - 4: let $t = i\Delta$
 - 5: Compute the error matrix $E(t) = \int_0^t (A(t) - A(0))e^{A(0)(t-\tau)} d\tau$ via numerical integration.
 - 6: Compute the total error as the sum of the absolute values of the entries of $E(t)$
 - 7: if $E > \text{error tolerance}$ then
 - 8: let $T = (i - 1)\Delta$
 - 9: break
 - 10: end if
 - 11: end for
 - 12: Compute the Jacobian matrix $J = \frac{\partial F(x)}{\partial x}$ at the current time using finite differencing.
 - 13: Find the eigenvalues $\lambda_1, \dots, \lambda_n$ of J
 - 14: Take the eigenvalue λ_i with largest magnitude $|\lambda_i|$.
 - 15: Let $\Delta = 1/|\lambda_i|$
-

[00140] The following discusses further details regarding possible use with such chattering.

[00141] In the chattering algorithm, there are two timing parameters, the length of the window T , and the time slice Δ (where $N\Delta = T$). For accuracy purposes, these two parameters should be sufficiently small, but for computational efficiency, these two parameters should be large. Therefore, we want to keep the parameters as large as possible, as long as the error is tolerable.

[00142] The incremental state equation (for δz) relies on the Jacobian matrix evaluated at the beginning of the time window. To characterize the error, we compare two linear systems, one that has a constant matrix, denoted $A(t_0)$, and the other has a time varying matrix, denoted $A(t)$.

Consider both linear systems:

$$\dot{x} = A(t)x(t) \quad (1)$$

$$\dot{y} = A(t_0)y(t) \quad (2)$$

with the same initial conditions, $x(t_0) = y(t_0)$.

The solution to (1) is of the form

$$x(t) = \phi(t, t_0)x(t_0)$$

where $\phi(t, t_0)$ satisfies several properties,

$$\dot{\phi}(t, t_0) = A(t)\phi(t, t_0)$$

$$\phi(t_0, t_0) = I$$

$$\phi^{-1}(t, t_0) = \phi(t_0, t).$$

The solution to (2) is of the form

$$y(t) = e^{A(t-t_0)}x(t_0).$$

The error due to approximating $A(t)$ with $A(t_0)$ is:

$$E(t) = \phi(t, t_0) - e^{A(t-t_0)}.$$

[00143] Taking the derivative yields

$$\dot{E}(t) = \dot{\phi}(t, t_0) - A(t_0)e^{A(t-t_0)}$$

and replacing $\dot{\phi}(t, t_0)$ with $A(t)\phi(t, t_0)$ yields

$$\dot{E}(t) = A(t)\phi(t, t_0) - A(t_0)e^{A(t-t_0)}$$

and substituting for $\phi(t, t_0)$ yields

$$\begin{aligned}\dot{E}(t) &= A(t)(E(t) + e^{A(t-t_0)}) - A(t_0)e^{A(t-t_0)} \\ &= A(t)E(t) + (A(t) - A(t_0))e^{A(t-t_0)}.\end{aligned}$$

This provides the error as a function of $(A(t) - A(t_0))$, and if the eigenvalues of $A(t)$ for all t have real values less than zero, the error is linearly proportional to $(A(t) - A(t_0))$.

[00144] The procedure to choose T is to ask the customer what size of error is tolerable. The customer may say an error of 20% is tolerable, in which case the window length T can increase until reaching the associated threshold. The measure of the error is related to the difference,

$$\sum_{i=1}^I \frac{\partial F}{\partial z} \alpha_i \Big|_t - \sum_{i=1}^I \frac{\partial F}{\partial z} \alpha_i \Big|_{t_0}$$

The size of the time slice Δ is related to the magnitude of the largest eigenvalue of $\frac{\partial F}{\partial z}$. Let $|\lambda| = \sqrt{\lambda_{real}^2 + \lambda_{imag}^2}$ and suppose $|\lambda|_i$ has the largest value. Then

$$\Delta = \frac{1}{|\lambda|_i}.$$

[00145] The following discusses further details regarding an example of synchronizing a decision module's model and current information with that of information from one or more other decision modules.

[00146] In the multi-agent self-organizing architecture of CDI, all agents synchronize in some way. This is analogous in the worst case to the many-body problem, which Isaac Newton first formulated, and is unsolvable for three or more bodies. However, good approximations for systems of more than two bodies are computationally tractable. The approach in CDI is to solve a series of two-body

problems that emulate a mean field aggregation, and update sequentially through a Pareto game.

[00147] Consider N agents, and each agent i , $i = 1, \dots, N$, has its own optimization problem with criterion, state and control:

$$\begin{aligned} & \text{Min}_{u_i} J_i(x_i, u_i) \\ & \text{s. t. } \dot{x}_i = f(x_i, u_i) \end{aligned}$$

The N problem in a single optimization problem is not algorithmically solvable, but it is possible to solve algorithmically a Pareto-optimization problem with two players (the Pareto game), and approximate the solution to the N player problem.

The two-agent (agent 1 and agent 2) Pareto-optimization problem is

$$\text{Min}_{u_1, u_2, \alpha_1, \alpha_2} \alpha_1 J_1(x_1, u_1) + \alpha_2 J_2(x_2, u_2)$$

subject to

$$\begin{aligned} \dot{x}_1 &= f(x_1, u_1) \\ \dot{x}_2 &= f(x_2, u_2) \\ \alpha_1(t) + \alpha_2(t) &= 1 \end{aligned}$$

[00148] The CDI approach is that each agent i plays a two-agent game with the CDI Mean Field agent composed of the criteria from all other agents except agent i . Instead of solving the two-agent optimization problem directly, we convert the formulation to a Hamiltonian problem (using Pontryagin's minimum principle).

[00149] The Hamiltonians are additive for Pareto optimization. Therefore, we can add local Hamiltonians for individual agents to create an aggregate mean field Hamiltonian.

[00150] Let H_i be the local Hamiltonian for agent i and let H_i^{MF} be the mean field Hamiltonian composed of the Hamiltonians for all other agents, excluding i . That is, the mean field Hamiltonian for agent i is a functional form of the Hamiltonians of the other agents.

Then the two-agent optimization problem is

$$\text{Min}_{u_i, u_i^{MF}, \alpha_1, \alpha_2} \alpha_1 H_i(x_i, u_i) + \alpha_2 H_i^{MF}(x_i^{MF}, u_i^{MF})$$

subject to the state and costate equations of the combined Hamiltonian, and

$$\alpha_1(t) + \alpha_2(t) = 1.$$

The state consists of $[x_i, x_i^{MF}]^T$, the costate is $[p_i, p_i^{MF}]^T$, and the control is $[u_i, u_i^{MF}]^T$. The initial conditions for x_i^{MF} and p_i^{MF} come from the previous solution, and u_i is the local solution from the previous pass.

[00151] The solution to the two-agent Pareto game provides $\alpha_1(t)$ and $\alpha_2(t)$, and the total Hamiltonian for agent i is updated:

$$H_i^T = \alpha_1 H_i + \alpha_2 H_i^{MF}$$

The modified Hamiltonian for agent i is constructed by projecting the total Hamiltonian into the local state space. The modified mean field Hamiltonian is constructed by projecting the total Hamiltonian into the mean field state space.

[00152] Now, the another agent plays the game, solving the two-agent Pareto game, and updating locally, and each agent updates its mean field agent when it is ready to play its game.

[00153] The following discusses further details regarding example embodiments.

Overview

[00154] The Cooperative Distributed Inference (CDI) platform is a unique advanced technology offering to enable near-optimal and near-real-time decision making on vast amount of heterogeneous and distributed information, in complex knowledge-based decision support systems, combining absolute, hard and soft rules to handle various requirements from natural or governing laws, policies, and best practices. Figure 17 illustrates a diagram 1700 of various interactions of different portions of a CDI system.

[00155] The CDI platform features a Distributed Architecture (DA) for resolving queries by accessing information from both an Internal Heterogeneous Database (IHDB) and external data sources referred to as Sensors. CDI utilizes a network of computing devices as its nodes – called Decision Elements (DE) – that cooperate to resolve queries given to them.

[00156] The Decision Elements (DE) in a given DA can work together to reach best outcome for the whole group, i.e., reaching Pareto Efficiency (also referred to as Pareto equilibrium) – a stable state where no change by any individual can be made to make the sum of whole group better.

[00157] Each DE can solve the problem independently if provided with complete knowledge and data. DE solves the query with a very unique approach, using Optimal Control Theory, starting with a technique called analytic continuation (transforming query and rules into differential equations whose dependent variables represent internal variables and parameters of the rules), then using its own internal knowledge to solve the query, providing an outcome.

[00158] In distributed environments, group of Decision Elements synchronize in an iterative process utilizing updates from other DEs and provide a final result, via a Pareto multi criteria optimization strategy.

[00159] The CDI platform needs rules, not necessarily exact criteria, to reach the objective state, producing the near-optimal solution. This is a very attractive feature when exact quantitative criteria cannot be provided in advance due to uncertainty or other reasons.

Platform Features

[00160] CDI is perfect for big data analytics, supporting many data types:

- Structured: databases, ontologies
- Semi-structured: spreadsheets, CSV/TSV, forms, emails
- Unstructured: web pages, blogs, text documents
- Symbolic: business rules, math formulas

CDI's distributed computing architecture also make it very scalable to handle large amount (peta-size) of heterogeneous data ingestion while performing real-time analytic results even in microseconds (depending in part on resources available and the complexity of queries).

Rules Support

[00161] CDI can integrate different types of business rules: absolute, hard and soft rules, from natural or government laws, operational or policy requirements,

and practice guidelines. Absolute rules and hard rules always take logic value 0 (false) or 1 (true) when instantiated. Soft rules, however, may take any value in the interval $[0,1]$, or more generally more than two values. Absolute rules reflects a must-satisfy situation, such as FDA/USDA requirements; hard rules are operational requirements such as “no serious persistent side effects”, but which may be temporarily relaxed in specific situations; and soft rules can come from guidelines or experiences, such as “better not give drug XYZ to diabetes patients with heart problems”. Figure 18 illustrates a diagram 1800 of examples of different types of rules. Please note that all chaining of the rules may happen automatically in a dynamic manner during the optimization process. CDI handles this complexity with said techniques above by solving a distributed continuous-space optimization problem.

Self-Adapting And Learning

[00162] CDI platform features a self-learning design. As CDI converts the original query solving into an optimal control problem, it can use feedbacks from the environment (external sensor or internal updates from peer decision elements) to refine its internal model: a Hamilton-Jacobi-Bellman equation will be updated to reflect new information and automatically form soft-rule like constraints internally. The process to update internal mathematical model is similar to reconstructing 3D images from CT or MRI imaging by tomographic reconstruction, but used for dynamic systems.

Scalability and Performance

[00163] CDI platform enables functionality to:

- Integrate data that may include 1,000,000+ variables and 100,000+ of constraints
- Provide data integration over evolving distributed network
- Specify queries over a broad range of languages
- Specify queries of a broad range of complexity
- Provide best known response to queries at the local level
- Operate in a variety of environments including cloud-based or local deployments
- Real-time processing of queries

Comparisons

[00164] CDI stands out in being able to do the following things:

- Approximate the optimal solution of NP-hard problems (such as planning and scheduling) by mapping criteria and constraints onto a continuous space and solve it as an Optimal Control problem with polynomial-time algorithms. The solutions offered by CDI may be near optimal rather than exactly optimal but the running time will be greatly shortened and some large-scale intractable problems can become solvable by CDI.
- Blend adaptive learning together with rules composition. Traditional rule engines might support hard and soft rules (constraints), but the scoring mechanism and weights need be manually adjusted to meet the goals, while in CDI, via feedback mechanism, these weights can be optimized as internal configurations.

The unique abilities make CDI an ideal choice for intelligent decision support systems.

Sample Applications

[00165] CDI platform works great in areas where there are a lot of heterogeneous data, government compliances and business requirements, such as healthcare and energy.

Clinical Auto-Coding Application

[00166] A Clinical Auto-Coding (C.A.C.) application can be used to detect medical under-coding, over-coding, and miscoding, highlighting potential opportunities where higher billing is justified. It automatically generates clinical codes, including ICD-10 directly from clinical encounter notes such as physician notes, lab results, and discharge records, while supporting workflows accommodating the roles played by administrators, coders and doctors in coding. Figure 19 illustrates an example user interface 1900 related to medical / clinical auto-coding. C.A.C. incorporates user feedback to learn coding best practices as it processes records. With its sophisticated adaptive technology, C.A.C. improves over time, optimizing coding for each organization to improve the efficiency, accuracy and revenue capture of the medical coding activity

Clinical Intelligence Web Services

[00167] CDI powers the following intelligent healthcare services that can be easily integrated:

- Clinical Record Intelligence - For analysis of EMRs (electronic medical records) and encounter notes, identification of actionable clinical terms and concepts in those documents. The services can extract concepts, understand issues, and analyze documents.

- Clinical Coding Intelligence - For inferring clinical codes from a set of clinical concepts, grouping codes, correlating codes, analyze documents, audit and justification, and compliance.

- Patient-Centric Intelligence - Secure delivery of end-user applications to predict, recommend, and explain personalized actions to improve patient outcomes. It provides a patient centric view, can provide medical risk prediction and proactive monitoring for patients, automated data abstraction, actionable recommendations, and more.

All these web services run in secure cloud with full compliance measurement in place.

Fraud, Waste and Abuse Detection

[00168] In healthcare, CDI helps assess and monitor risk of medical fraud, waste and abuse (FWA) by uncovering providers, and to a lesser extent pharmacies, who are suspected of having committed fraud via a variety of schemes. Similar application domains include financial frauds.

[00169] By ingesting a wide variety of data that is difficult to link (including live public data), a FWA service analyzes data to find evidence and patterns of fraud, and provides a dashboard application for agents/detectives to prioritize and act on the discovered suspected cases of medical fraud. The service features strict and fuzzy rules-based detection as well as automatic pattern discovery, and runs in real-time and continuous mode to support proactive monitoring and action.

Energy Intelligence

[00170] Smart Grid can involve uncertain bi-directional exchange in distributed grids, so intelligent control can be used to provide active synchronization between the network of element controllers and the outside grid management system allows high quality of service to be maintained in a cost-effective manner, hence the dynamics can be learned from sensory observations. CDI enables distributed micro-grid control, supports inductive modeling with “soft” rules that are learned and continuously updated, to optimize the grid.

Distributed Architecture (DA)

[00171] The DA is a network of interacting components called decision elements (DEs). The DEs collaborate in the resolution of a query posed by one of them. The DA's block diagram is shown in Figure 18, where Sensors refer to external input data in general. There is an additional translation layer to process external data to be consumed by DEs.

Decision Element (DE)

[00172] A decision element is a higher-level functional component solving queries locally. It can have subcomponents such as a programmable search engine, internal heterogeneous database, Inference engine, Inference rule base, API / user interface, and network interface. A decision element is capable of providing a quick and near-optimal solution to a complex query with complete input of data.

Internal Heterogeneous Database (IHDB) and External Knowledge Base (EKB)

[00173] IHDB is data preprocessed and stored by a specific decision element (DE). Figure 20 illustrates a diagram 2000 of some components of a CDI system, including Knowledge Bases containing data sources ranging from domain knowledge to general facts. IHDB encodes knowledge into knowledge components (KC's). Each KC is used and updated by a DE in the DA, and multiple KCs may share rules.

[00174] External Knowledge Base (EKB) refers to data, including rules, as input into specific decision elements, such as patient's body temperature, blood pressure, or instantiated rule to determine if patient's cholesterol level is high. EKB can also

contain communicated information from other DEs. Domains for variables include: real, complex, integer, binary numbers and symbolic token on finite domains.

Interfacing Components

[00175] The following components act between users and the data via API and/or GUI.

- Rule Entry Interface - It provides the entry of rules into the IHDB, validates the specification of rules before insertion, and route the rules to the appropriate DE for insertion to their respective knowledge components.
- Sensor Ingestion Interface - A sensor is a machine or service where external data exist. Sensor Ingestion Interface (SII) enables the system to add or remove a sensor, poll a sensor and submit data to the network.
- Query Language Interface - Query Language Interface accepts the query, submits it to the system (Distributed Architecture) and provides response. It is an API and flexible UI can be built on top of it.

Minimization Function Generator (MFG)

[00176] The minimization function generator converts a query to a minimization function (i.e. analytic continuation). This is useful because the problem is converted from search problem in a large discrete space (like graph search problems, which are usually NP-complete) into an optimization problem in continuous space (polynomial algorithms exist). An analogy is NP-hard integer programming, while continuous linear programming has a very efficient solution.

Query Response Engine (QRE)

[00177] The Query Response Engine is the core of the whole system responsible solving the query (locally). A mathematical model is constructed based on these equations obtained from previous steps, containing the current state and object state (goal state). The continuous-space optimization automatically handles forward and backward rule chaining by moving along the trajectory toward target state. It also manages uncertainty by keeping a large set of possible states and reducing the solution space only when more information becomes available.

Standard optimization techniques (e.g. Newton-Raphson Method) can be employed to solve the problem.

[00178] Feedbacks and updates (data from EKB or other DE in the architecture) will be used to refine the mathematical model over time; therefore, the core engine is self-adapting.

Pareto Multi-Criteria Optimization Engine

[00179] The Pareto Multi-Criteria Optimization Engine (PMOE) is the aggregation step where all DE in the network settle to obtain a good stable solution – a state where no improvements can be made to any individual DE without reducing the whole team's performance – a state belongs Pareto Optimal Set in Game Theory. It is like each DE is playing a game and they communicate and interact and work together to make the best outcome for the criteria (query). To efficiently synchronize all decision elements, Mean Field Theory is applied for dimension reduction using knowledge obtained. Figure 21 illustrates a diagram 2100 of performing Pareto processing for use with mean field techniques, including to reach Pareto efficiency.

Data Exchange Specifications

[00180] The system can take many different data types via ingestion API and has adapters from different public data sources. The supported data types include common ones from XML, CSV, TSV, SQL, Spreadsheet, JSON, and more. OData support is also available. Output types can be API (XML or JSON), as well as exporting to CSV, SQL or directly to services such as SOLR and Cassandra.

Running Environment

[00181] CDI can run in the cloud as Software as a Service platform. We can provide whole end-to-end support by setting up the infrastructure in a Virtual Private Cloud or deploy and configure it with the cluster clients provide. The system features SaaS architecture and provides APIs to be used by third parties. For advanced integration requests, Java/Scala and Python libraries are available.

[00182] To sum up, CDI platform utilizes many advanced techniques from Mean Field Theory, Lagrangian and Hamiltonian functions, Pareto Optimal Set, Gauge Theory, and so on derived from modern mathematics, quantum physics, optimal control and game theory to achieve high performance. In the detailed computation process, lots of transformation, approximation, optimization, caching and other advanced computing techniques are used to improve accuracy, speed and scalability. Due to this unique approach, CDI is able to solve complex decision making problem in a smoothly, efficient manner and achieve near optimal results.

[00183] The Cooperative Distributed Inference (CDI) platform is a unique advanced technology offering to enable near-optimal and near real-time decision making on vast amount of heterogeneous and distributed information, in complex knowledge-based decision support systems, combining absolute, hard and soft rules to handle various requirements from natural or governing laws, policies, and best practices.

[00184] Each agent in the Distributed Architecture is a Decision Element that can solve the problem independently, provided with knowledge base and data. In a very unique manner, the agent uses Optimal Control Theory to obtain a near optimal solution, starting with a technique called analytic continuation (transforming query and rules into differential equations whose dependent variables represent internal variables and parameters of the rules), then using its own internal knowledge to solve the problem as an optimization problem in continuous space, to allow for efficient solving.

[00185] The outcome will be fed back to the Rules Editor and optimization process (the Chattering algorithm) to enable automatic adjustment of weights of soft rules (constraints) and achieve optimal score of the objective function.

[00186] A CDI agent is an independent decision element that can take in sensor data (input from the environment, streaming or in batches), as well as the knowledge-bases (rules composed by domain experts, including absolute, hard and soft rules) to generate a set of partial differential functions (Lagrangian

constraints), through continualization. Similarly, the objective is also converted into a minimization function, as part of the Lagrangians which will be solved via the Hamilton–Jacobi–Bellman equation. Each agent will talk to other agents via a “mean field” abstraction layer to greatly reduce the communication and computation overhead, and incorporate additional information to reach the global optimal state (a stable, near optimal set of states across all agents). Finally, the agent exercises control over the system. Figure 22 illustrates a network diagram 2200 of an example decision module agent.

[00187] Various domain knowledge is captured from experts in the form of Domain Specific Language. Some are constraints; logic forms need be converted to Boolean equations; and variables in soft rules take values [0..1]. This creates a set of equations to be solved by the optimization algorithm. The optimization process takes data from a time range and finds the best state configuration to reach optimality. It starts with a “learning” process to find a good initial configuration by taking in a short history of data, before processing real-time streams. Figure 23 illustrates a network diagram 2300 of an example of offline workflows for knowledge capture.

[00188] Each CDI agent computes a “mean field” view of the system via its neighbors, and responds to queries with the latest updates. The approximate mean field view of a group greatly reduces computational dimensions. The agents synchronize with others and understand the global state via the mean-field approximation. They engage in games to reach Pareto Optimal (also referred to as Pareto equilibrium) – the best output. Figure 24 illustrates a network diagram 2400 of an example of workflows for mean field computation and Pareto Optimal.

[00189] An example home solar micro-grid system illustrates one example embodiment of a CDI application or automated control system, which takes the sensor data from the solar panel (in the house), substation, and power network, and decides whether or not to fulfill the utility requests in real time using a set of complex rules. Figure 25 illustrates a network diagram 2500 of an example of an

automated control system for a home solar micro-grid electrical generating system.

[00190] Figures 26-28 provide further details regarding operations of portions of example CDI systems and their sub-components, and Figures 29A-29K illustrate examples of using a CDI system to iteratively determine near-optimal solutions over time for controlling a target system in diagrams 2900A-2900K. In particular, Figure 26 illustrates a further diagram 2600 of workflow and components of a portion of a CDI system, Figure 28 illustrates a diagram 2800 of an overview workflow for a portion of a CDI system, and Figure 27 illustrates a diagram 2700 of workflow for an inference process portion of a CDI system.

[00191] Further details related to an example CDI system are shown below and included in provisional U.S. Patent Application No. 62/015,018, filed June 20, 2014 and entitled "Methods And Systems For Cooperative Distributed Inferencing." In addition, further details related to example details of using gauges to perform model error measurements are included in provisional U.S. Patent Application No. 62/182,796, filed June 22, 2015 by applicant Atigeo Corp. and entitled "Gauge Systems," which is hereby incorporated by reference in its entirety.

Current notation	Comments
t	Notation for algorithmic time.
$q(t)$	Notation of canonical coordinate vector for entire system.
q	Notation of canonical coordinate vector dropping time dimension.
$q^{(f)}$	Notation of canonical coordinate vector for specific function f .
$\dot{q}$	Notation of first time derivative of canonical coordinate vector.
$\ddot{q}$	Notation of second time derivative of canonical coordinate vector.
h	Notation of HEAD of Horn clause.
$\varphi(q)$	Notation of generic proposition.
$\sigma(q)$	Notation of generic proposition

	alternate to φ .
T_i	Notation of the TV of a soft rule.
$\check{r}(q; \varphi, \sigma)$	Generic equational form relating two propositions.
$\check{\varphi}(q)$	Notation of the equational form of $\varphi(q)$.
$\varphi_Q(q)$	Notation for proposition defined by the query.
$\check{\varphi}_Q(q)$	Notation for equation defined by the query.
$J(q)$	Notation for minimization function for the query.
$\mathcal{L}$	Notation for static Lagrangian
$\mathcal{L}_k^{(o,T)}$	Notation for total static Lagrangian for DE_k .
q	
$\{p_a\}$	
$u^{(k)}$	
$H_k^{(o)}$	Primary Hamiltonian for the absolute rules for DE_k .
$H_k^{(A)}$	Hamiltonian for the Tellegen agent of the total Hamiltonian's rules.
$H_k^{(T)}$	Total Hamiltonian for DE_k .

Overview

[00192] This document introduces and specifies the architecture for the Cooperative Distributed Inferencing Platform (CDIP). The primary instance of this is the Distributed Architecture (DA) for resolving queries by accessing both an Internal Heterogeneous Database (IHDB) populated by a special class of Horn Clause rules and external data sources referred to as sensors.

[00193] The architecture implements a network of active devices at its nodes. These devices are called Decision Elements (DE's). The DE's cooperate in the resolution of a query posed to one or several of them. The DE's in a given DA are referred to as the team.

[00194] Every DE in a team is programmed to transform rules in its domain, determined by a posed query, into an ordinary differential equation (ODE), whose dependent variables represent internal variables and parameters. The dependent

variables include unknowns of the query posed to the DE. The DE's in the architecture are synchronized via a Pareto multi criteria optimization strategy.

[00195] The components of the CDIP include:

- Application requirements that the system is designed to accommodate.
- Functional requirements that satisfy the application requirements and pertain directly to the construction and operation of the system components.
- Subcomponents which are necessary to implement the functional requirements
- Limitations which highlight noteworthy constraints which are inherent the specified implementation of the architecture.
- Architectural flow describes key aspects of the architecture that indicate how the system is to be constructed given the specified essence and key behavior of the subcomponents.
- Software realization of the architecture describes the key pieces of software necessary for system implementation.
- Data describes the kinds of data the system is expected to accept as input and produce as output.
- Data exchange protocols reference key data types and structures that need to be exchanged across the system and the protocols for exchange.
- Environment describes the particulars of the environments that the system will be able to operate in and therefore should be tested in.
- Testing describes how the system should be tested given the data and operating environments.

Subcomponents

[00196] Subcomponents are fundamental parts of the architecture that perform particular roles. This section contains descriptions for each of the subcomponents of the architecture. The subcomponents are:

- The Distributed Architecture (DA).
- The Internal Heterogeneous Database (IHDB).
- The Rule Entry Interface (REI).
- The Rule Editor (RE).

- The External Knowledge Base (EKB).
- The Sensor Ingestion Interface (SII).
- The Rule Conversion Engine (RCE).
- The Decision Element (DE).
- The Query Language Interface (QLI).
- The Minimization Function Generator (MFG).
- The Query Response Engine (QRE).
- The Pareto Multi-Criteria Optimization Engine (PMOE).

Distributed Architecture (DA)

[00197] The DA is a platform of interacting components called DE's. The DE's collaborate in the resolution of a query posed by one of them. The DE's implement a distributed, dynamic optimization process, herein referred as the optimization process (OP). OP implements an optimization process that computes an answer to the active queries as a function of data stored in two categories of repositories: IHDB and EKB's. These repositories of the data needed to implement OP given a query.

[00198] The EKB's are a collection of public or private repositories of knowledge relevant to the DE posing a query. A DE has a list of external repositories (LER). Each entry in an LER includes 1) a protocol, 2) a heading sub-list, and 3) a translation grammar. Each protocol entry prescribes the access procedure to the corresponding knowledge repository. Each heading sub-list entry contains a summary of the knowledge contents of the corresponding repository. Finally, each translation grammar entry provides a procedure for converting knowledge elements of the corresponding repository in to the rule representation in the IHDB of the DE. This representation is discussed below.

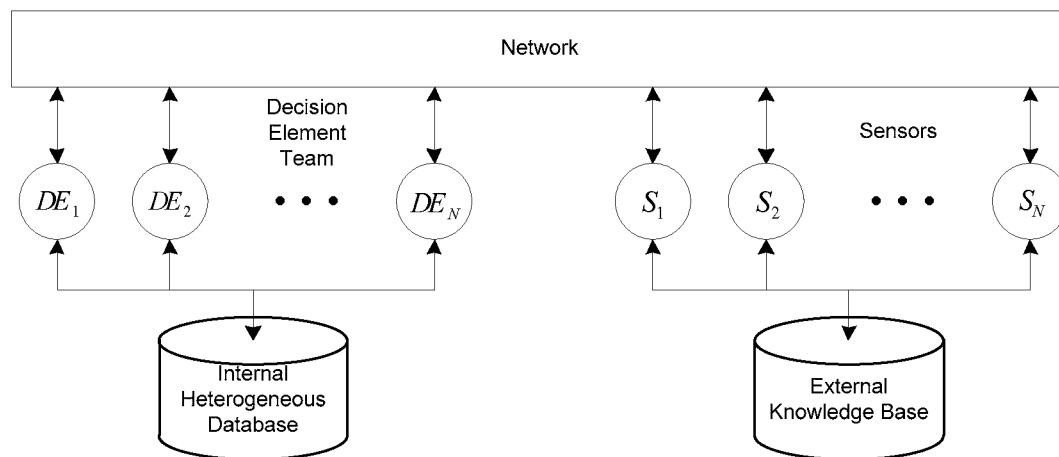


Illustration AA

[00199] The DA's block diagram is shown in Illustration AA. Functional characteristics of this architecture and in particular, the DE's, IHDB, and the sensors are described, including the following concepts:

- The DA
- A process for resolving queries by accessing the IHDB and External Knowledge Bases (EKB's) through sensors
- The constitution of DE's
- A query and corresponding rules transformation into an ODE
- The orchestration of a team of DE's through a Pareto multi criteria optimization strategy

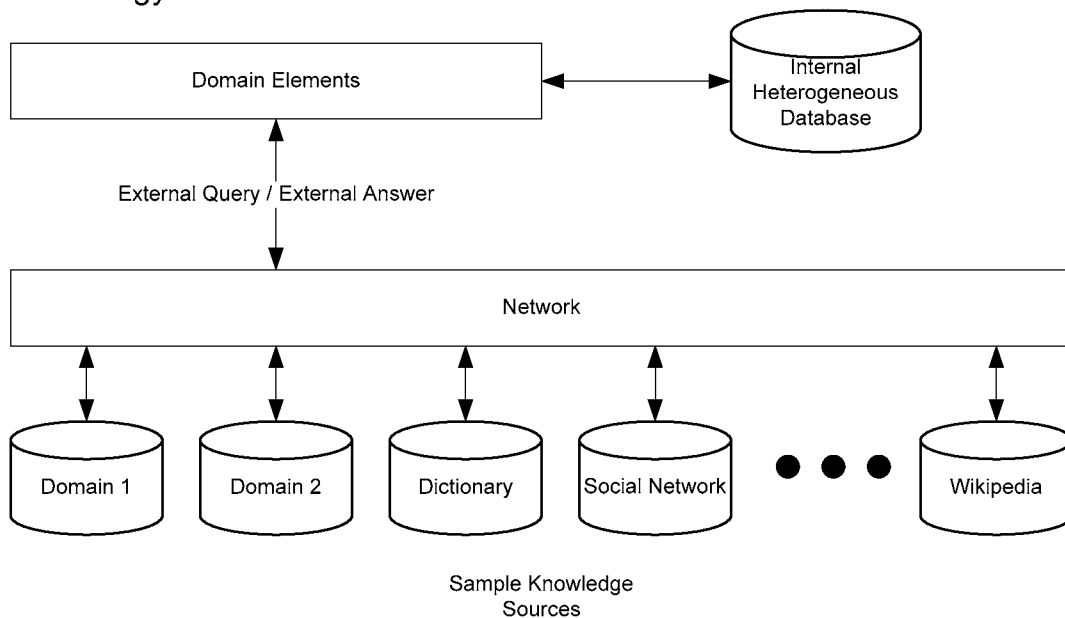


Illustration BB

The Internal Heterogeneous Database (IHDB)Composition of IHDB as a set of Knowledge Components

[00200] The IHDB encodes knowledge about the implemented application. The IHDB is divided into knowledge components (KC's). Each KC is consulted and updated by a DE in the DA. Any pair of KC's may have an overlapping set of rules by which they operate, but there is no a priori constraint on intersections or inclusion. The collection of KC's constitutes the existing knowledge of the system.

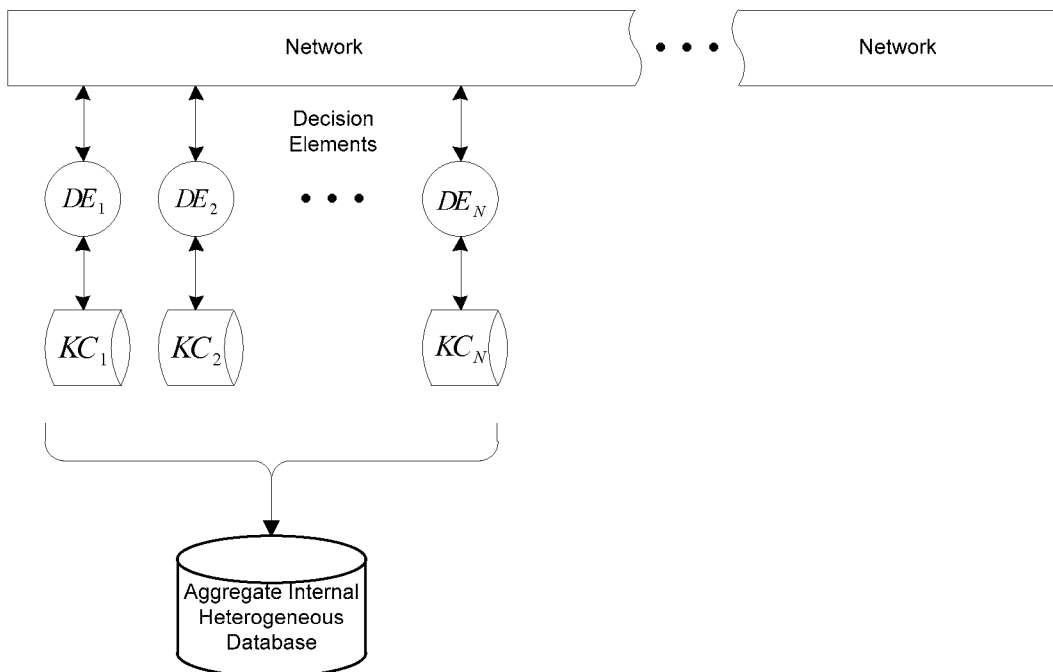


Illustration CC

Algorithmic formulation of a rule

[00201] A KC is a collection of rules, written in a restrictive Horn clause format. The rules are logic entities. When a rule is instantiated, it has a logic value. The logic values a rule can have are taken from the interval $[0,1]$. The entire system of rules is evaluated using variables and parameters which are collectively referred to as the generalized coordinates of the system and are indexed as follows:

$$q(t) = \{q^{(1)}(t), \dots, q^{(N)}(t)\}.$$

(3.2-1)

[00202] The time argument t refers to the algorithmic time of the system which means that it has no other meaning than as a continuous index with respect to the evolution of the system. There is therefore no requirement that it correspond to a physical aspect of the system although this may naturally occur. Physical time may be represented specifically by a canonical coordinate of choice $q^{(i)}(t)$. Alternatively, we may refer to the q 's without expressly stating the independent time argument and write

$$q(t) = \{q^{(1)}, \dots, q^{(N)}\}. \quad (3.2-2)$$

Then we should also note that the time derivatives are referred to notationally as

$$\dot{q} = \frac{dq(t)}{dt}, \quad \ddot{q} = \frac{d^2q(t)}{dt^2} \quad (3.2-3)$$

[00203] These coordinates are referred to variously as q depending on the context and the expected arguments of the function to which they are applied. When it is necessary to distinguish between more than one q in equational form we generally write q_f where f denotes the reference function or appropriate domain. Typically, we assume without loss of generality the entire set of canonical coordinates q is an argument to any function, term or proposition. In practice, we may further assume it is possible to apply the particular required coordinates as need to mathematical construct in question.

[00204] The rules in each knowledge component are of three types: absolute rules, hard rules, and soft rules. Absolute rules and hard rules take logic value 0 (false) or 1 (true) when instantiated. Soft rules take any value in the interval $[0,1]$.

[00205] The format of the restrictive Horn Clauses in the IHDB is illustrated in Fig. 3.2-2. A Horn Clause is an object composed of two objects a HEAD and a BODY connected by backward Implication ($\Leftarrow$). The logic implication transfers the logic

value of the BODY to the HEAD. If the rule is an absolute rule or a hard rule, the logic value is 1 (if the BODY is logically true) or 0 (if the BODY is logically false). If the rule is a soft rule, the logic value transferred by the body is any number in $[0, 1]$.

[00206] The HEAD is a data structure composed of two objects: A name, h , and a list of arguments described by the argument vector $q = (q^{(1)}, \dots, q^{(n)})$. The list of arguments includes variables and parameters. The variables take values in the domain of the rule and the parameters are constants passed to the rule and unchanged by the instantiation of the rule. The domain of the rule is a set of values that each of its variables can take. In general, variables can take values over numerical or symbolic domains. As an example, a symbolic domain can be a list of diseases. A numeric domain can be a set of pairs of numbers representing blood pressure.

[00207] For the applications of CDI, the domains for variables are: real numbers, complex numbers (floating point and floating point complex numbers), integer numbers, binary numbers and symbolic token on finite domains.

[00208] The BODY of a clause is a data structure composed of one or more terms, denoted $\varphi_i(q)$. The composition operation is extended-and, denoted by: $\wedge$. The extended-and works as a regular and in absolute rules and hard rules and as a functional product² on soft rules.

[00209] A rule with a head but not a body is called a fact. A fact's truth value is determined on the basis of the instantiation of its variables.

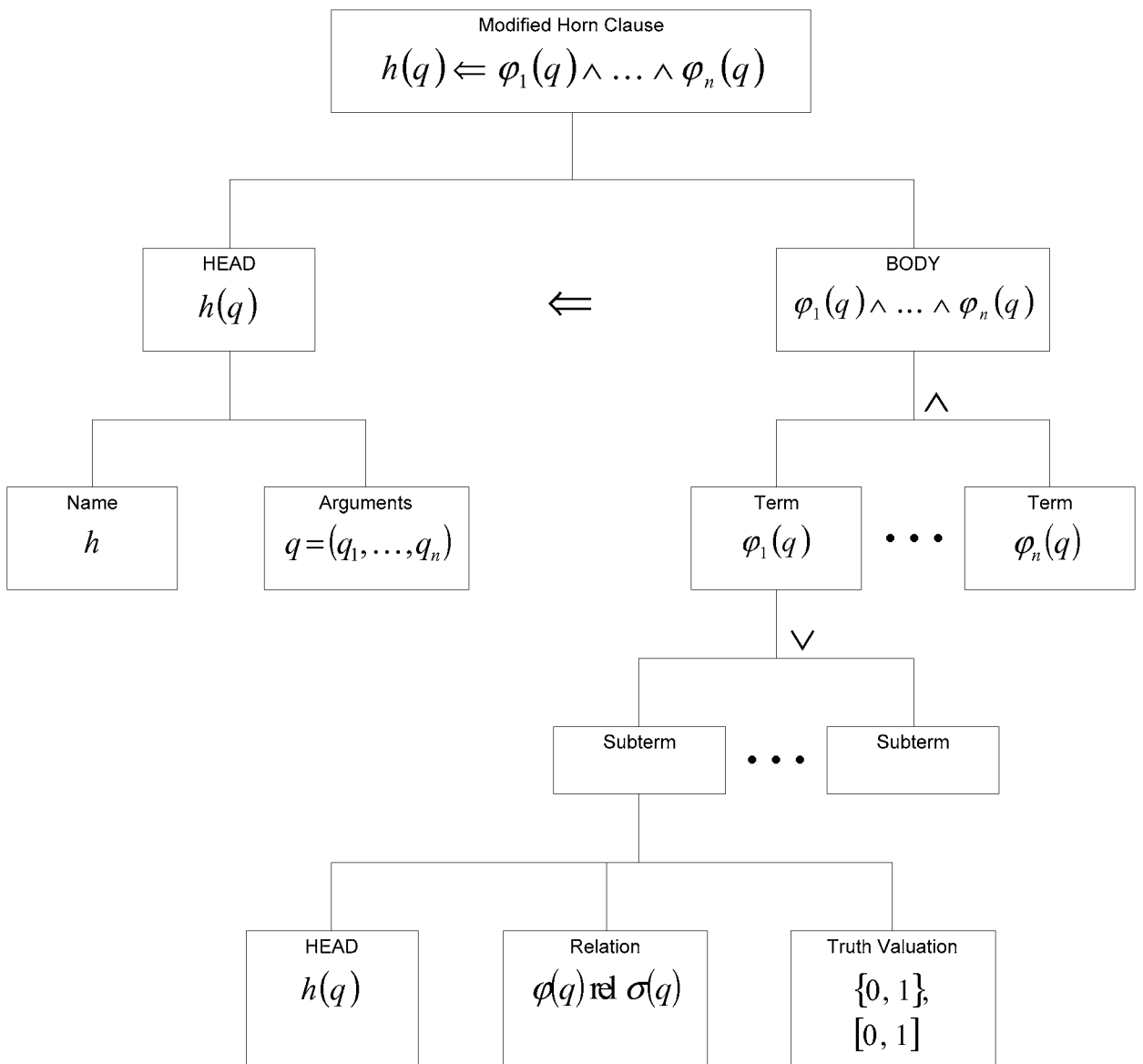


Illustration DD

[00210] Each term in the body of a rule is an extended disjunction (or denoted by $\vee$) of sub-terms. The $\vee$ operator behaves like the standard-or for absolute and hard rules and behaves in a functional form, described later, when connecting sub-terms encoding heads of soft rules.

[00211] A sub-term is either the HEAD of a rule, a relation or a truth valuation (TV). When it is a HEAD it may have the same name as the one in the HEAD of the rule but with different arguments. This provides a recursive mechanism for rule evaluation.

[00212] When a rule has a sub-term that is the head of another rule it is said that the two rules are chained together by the corresponding sub-term. Note that a rule can be chained to several rules via corresponding sub-terms.

Constraint domains

[00213] Constraint domains augment the BODY clause of Horn clauses to facilitate dynamic programming. Constraints are specified as a relationship between terms. Define the relationship between two terms

$$\varphi(q) \text{ rel } \sigma(q). \quad (3.2-4)$$

as a member of the following set

$$\text{rel} \in \{=, \neq, \leq, \geq, \text{statistical propagation}\}. \quad (3.2-5)$$

[00214] A relation can be of two types numeric or symbolic. Numeric relations establish equational forms between two functional forms. (For the initial phase only polynomial and affine linear functional forms will be considered.)

[00215] In general, an equational form is a set of one or more relations. For numeric relations, $\varphi(q) \text{ rel } \sigma(q)$, $\text{rel} \in \{=, \neq, \leq, \geq, <, >, \text{statistical propagation}\}$. The table below gives the relations considered and their symbols.

Numeric Relation	Symbol	Code Form
Equality	=	$\varphi = \sigma$
Disequation	$\neq$	$\varphi \neq \sigma$
Less-inequality	<	$\varphi < \sigma$
Less-Equal	$\leq$	$\varphi \leq \sigma$
Great-equality	>	$\varphi > \sigma$
Great- inequality	$\geq$	$\varphi \geq \sigma$

Table EE

[00216] The adopted code forms are the ones used in constraint logic programming.

[00217] A symbolic relation can be of two types: inclusion and constraint. Inclusion relations are of the form:

$$q \in \text{Set}$$

(3.2-6)

Where x is a variable or a parameter, $\in$ is the inclusion symbol and Set is a set of symbolic forms or a set of numbers or a composite set of the form shown in the table below.

Composite Set	Symbol	Code Form
Intersection	$\cap$	Set1/\Set2
Union	$\cup$	Set1\Set2
Complement	$\backslash$	$\backslash$ Set

Table FF

Constraint forms of the symbolic relational type may be one or a set of the forms presented in the table below For numeric relations, $\varphi(q) \text{ rel } \sigma(q)$, $\text{rel} \in \{=, \neq, <, >, \leq, \geq\}$.

Symbolic Relation	Symbol	Code Form
Equal	$=$	$\varphi\# = \sigma$
Not Equal	$\neq$	$\varphi\# \neq \sigma$
Is Contained	$\subset$	$\varphi\# < \sigma$
Contains	$\supset$	$\varphi\# > \sigma$
Is Contained or Equal	$\subseteq$	$\varphi\# \leq \sigma$
Contains or Equal	$\supseteq$	$\varphi\# \geq \sigma$

Table GG

[00218] A TV is either a variable or a constant with values in the interval $[0, 1]$. The TV of a Horn Clause that is an absolute rule or a hard rule can only take two values: 1 or 0. The TV when instantiated is 0 or 1. If the TV for an absolute or hard rule is 1, the rule is said to be in inactive state; if the TV is 0, the rule is said to be in active state.

[00219] The TV, T_i , of a soft rule satisfies

$$0 \leq T_i \leq 1.$$

(3.2-7)

If T_i above satisfies,

$$T_i \geq T_{\text{threshold}} \quad (3.2-8)$$

the soft clause is said to be in inactive state. If

$$T_i < T_{\text{threshold}}, \quad (3.2-9)$$

the soft clause is said to be in active state, where $T_{\text{threshold}}$ is a constant in $[0,1]$ defined for each soft clause. The default value is 0.5.

[00220] This concludes the description of the knowledge representation. The instantiation process of the goal in a DE, as function of its knowledge base, is carried out by the inference engine of the DE (see Fig. 3.7-1). This process is the central component of CDI and will be described later on the document.

Summary of terminology

[00221] The following table summarizes the terminology we have just reviewed.

Reference term	Definition
proposition	Defined as a construct as in the propositional calculus where the proposition takes on the value of true or false.
term	Recursively according to its assigned sub-term.
sub-term	A sub-term may be a Horn clause, a relation between two other sub-terms or an extended truth valuation depending on the context of absolute, hard or soft rules. In the case of absolute and hard rules it may be evaluated as a proposition. In the case of soft rules it takes a value on the interval $[0,1]$ and is considered to be active or true in the case that it exceeds its specific threshold.
Horn clause	A disjunction of terms with at most one positive term.
definite clause	A Horn clause with exactly one positive term.
goal clause	A Horn clause with no positive terms.
fact	A definite clause with no negative terms.

head	The positive term of a definite clause.
inactive state	The case when a rule will not apply for constrained optimization.
active state	The case when a rule will apply for constrained optimization.
truth value, TV	The value that is used to determine whether a rule is active or inactive.

Table HH

Horn clause example

[00222] The following example illustrates a Horn clause:

$$\begin{aligned}
 &\text{has\_fever}(\text{name}, \text{temperature}, \text{white\_count}, \text{heartrate}, \text{blood\_pressure}) \\
 &\quad \Leftarrow (\text{temperature} > 37) \\
 &\quad \wedge ((\text{heartrate} \geq 70) \vee \text{bp}(\text{name}, \text{temperature}, \text{blood\_pressure}) \\
 &\quad \vee \text{wc}(\text{name}, \text{white\_count}))
 \end{aligned}
 \tag{3.2-10}$$

The clause establishes under which conditions the patient of name name, has a fever. The variables in clause are:

name, temperature, white_count, heartrate, blood_pressure.

When instantiated they represent, respectively, the name of the patient, his current body temperature, his white blood cell count, his heart rate range, and his blood pressure.

[00223] The clause body includes other clauses: bp (blood pressure) and wc (white count).

[00224] This completes the specification of the rule-based framework. The next step is to specify a complete process for converting all rules of this form to a set of equations.

Rule Entry Interface (REI)

[00225] The Rule Entry Interface provides a mechanism for:

- Providing an API for the entry of rules into the IHDB.
- Validating the specification of rules to be inserted into the IHDB.
- Routing the rules to the appropriate DE's for insertion to their respective KC's.

Rule Editor (RE)

[00226] The Rule Editor allows users to specify rules associated with the systems to be interrogated.

External Knowledge Base (EKB)

- It may be distributed or not
- It may be persisted or not
- It may be persisted locally or remotely to an agent
- It may or may not be architecturally co-located with the IHDB
- A sensor may include any source of data used by the agent
- It may use various types of buses for data communication
- Sensors may or may not be co-located with agents/DEs

Rule Conversion Engine (RCE)

[00227] The rule conversion engine converts rules of the IHDB into equations.

Method for specification of a simple term as an equation

[00228] Consider the term $\varphi(q)$ with the following truth assignment.

$$\varphi(q) = \begin{cases} T & q \in \mathcal{D}_\varphi \\ F & q \notin \mathcal{D}_\varphi \subset \mathcal{D} \end{cases} \quad (3.6-1)$$

Then we can define the set of arguments which yield positive truth assignment.

$$\{q \in \mathcal{D}_\varphi \mid \varphi(q) \leftarrow T\}. \quad (3.6-2)$$

Define the corresponding equation $\check{\varphi}(q)$ of the term $\varphi(q)$ as

$$\check{\varphi}(q) = \begin{cases} 1 & \varphi(q) \leftarrow T \\ 0 & \varphi(q) \leftarrow F \end{cases} \quad (3.6-3)$$

Then extend the range of $\check{\varphi}(a)$ to the closed unit interval

$$\check{\varphi}(q) \rightarrow [0,1]. \quad (3.6-4)$$

Revisiting the taxonomy of absolute, hard and soft rules, we recognize that hard and soft rules (terms in this example) can take values along the interval

$$0 \leq \check{\varphi}(q) \leq 1. \quad (3.6-5)$$

And for absolute rules we add the constraint $\check{\varphi}(a) \rightarrow \{0,1\}$

$$\check{\varphi}(q)(1 - \check{\varphi}(q)) = 0. \quad (3.6-6)$$

Conversion of fundamental clauses of propositional calculus to equations

[00229] Define the following notation for the propositional calculus.

Symbol	Function
$\wedge$	And
$\vee$	Or
$\Rightarrow$	Implication
$\sim$	Not
$\exists$	Exists
$\forall$	All
Fuzzy rule	

Table II

[00230] **Theorem 3.6.1.** Given the method for the specification of equations from propositions, we prove the following transformations.

Proposition	Equation
$\sim \varphi(q)$	$1 - \check{\varphi}(q)$
$\varphi(q) \wedge \sigma(q)$	$\check{\varphi}(q) \cdot \check{\sigma}(q)$
$\varphi(q) \vee \sigma(q)$	$\check{\varphi}(q) + \check{\sigma}(q) - \check{\varphi}(q) \cdot \check{\sigma}(q)$
$\varphi(q) \Rightarrow \sigma(q)$	$1 - \check{\varphi}(q) + \check{\varphi}(q) \cdot \check{\sigma}(q)$
$\varphi_1(q) \wedge \varphi_2(q) \wedge \cdots \wedge \varphi_{k-1}(q) \wedge \varphi(q) \Rightarrow \varphi(q)$ (tail recursive)	$\check{\varphi}(n, q) = \frac{\check{h}(n-1, q)}{\check{\sigma}(n, q)\check{\varphi}(n-1, q) - 1}$

Table JJ

Proof by enumeration for equational representation of conjunction

- [00231] Define the function $\check{r}(q; \varphi, \sigma)$ which represents the equation corresponding to conjunction ($\wedge$). Verify by enumeration the correspondence of the mathematical equation values corresponding to the mapping $T \rightarrow 1$ and $F \rightarrow 0$.

$\varphi(q)$	$\wedge$	$\sigma(q)$
T	T	T
T	F	F
F	F	T
F	F	F

Table KK

$\check{r}(q; \varphi, \sigma) = \check{\varphi}(q) \cdot \check{\sigma}(q)$		
	$1 = 1 \cdot 1$	
	$0 = 1 \cdot 0$	
	$0 = 0 \cdot 1$	
	$0 = 0 \cdot 0$	

Proof by enumeration for equational representation of disjunction

- [00232] Define the function $\check{r}(q; \varphi, \sigma)$ which represents the equation corresponding to disjunction ($\vee$). Verify by enumeration the correspondence of the mathematical equation values corresponding to the mapping $T \rightarrow 1$ and $F \rightarrow 0$.

$\varphi(q)$	$\vee$	$\sigma(q)$
T	T	T
T	T	F
F	T	T
F	F	F

Table LL

$\check{r}(q; \varphi, \sigma) = \check{\varphi}(q) + \check{\sigma}(q) - \check{\varphi}(q) \cdot \check{\sigma}(q)$		
	$1 = 1 + 1 - 1 \cdot 1$	
	$1 = 1 + 0 - 1 \cdot 0$	
	$1 = 0 + 1 - 0 \cdot 1$	
	$0 = 0 + 0 - 0 \cdot 0$	

Proof by enumeration for equational representation of negation

- [00233] Define the function $\check{r}(q; \varphi, \sigma)$ which represents the equation corresponding to negation ($\neg$). Verify by enumeration the correspondence of the mathematical equation values corresponding to the mapping $T \rightarrow 1$ and $F \rightarrow 0$.

$\varphi(q)$	$\sim \varphi(q)$
T	F
F	T

Table MM

$\check{r}(q; \varphi, \sigma) = 1 - \check{\varphi}(q)$	
	$0 = 1 - 1$
	$1 = 1 - 0$

Proof by enumeration for equational representation of implication

[00234] Define the function $\check{r}(q; \varphi, \sigma)$ which represents the equation corresponding to disjunction ($\Rightarrow$). First note the equivalence of

$$\varphi(q) \Rightarrow \sigma(q) \text{ and } \sim\varphi(q) \vee \sigma(q). \quad (3.6-7)$$

Verify by enumeration the correspondence of the mathematical equation values corresponding to the mapping $T \rightarrow 1$ and $F \rightarrow 0$.

$\sim\varphi(q)$	$\vee$	$\sigma(q)$
T	T	T
T	F	F
F	T	T
F	F	F

Table NN

$\check{r}(q; \varphi, \sigma) = 1 - \check{\varphi}(q) + \check{\varphi}(q) \cdot \check{\sigma}(q)$
$1 = 1 - 1 + 1 \cdot 1$
$0 = 1 - 1 + 1 \cdot 0$
$1 = 1 - 0 + 0 \cdot 1$
$1 = 1 - 0 + 0 \cdot 0$

Proof for equational representation of tail recursion

[00235] Tail recursion is propositionally defined as

$$\varphi(q) \Leftarrow \varphi_1(q) \wedge \varphi_2(q) \wedge \cdots \wedge \varphi_{k-1}(q) \wedge \varphi(q) \quad (3.6-8)$$

where s represent the current state. To develop an equational representation of the recursive formulation, first define the general function $\check{\varphi}(n, q)$ where n represents the n^{th} iteration of the tail recursion and $\check{\varphi}(n, q)$ is the logical consequent. Then rewrite the above formulation using the recursive step.

$$\check{\varphi}_1(n, q) \wedge \check{\varphi}_2(n, q) \wedge \cdots \wedge \check{\varphi}_{k-1}(n, q) \wedge \check{\varphi}(n-1, q) \Rightarrow \check{\varphi}(n, q) \quad (3.6-9)$$

Define

$$\begin{aligned} \check{\sigma}(n, q) &\cong \check{\varphi}_1(n, q) \wedge \check{\varphi}_2(n, q) \wedge \cdots \wedge \check{\varphi}_{k-1}(n, q) \\ \check{r}(n-1, q) &\cong \check{\sigma}(n, q) \wedge \check{\varphi}(n-1, q) \end{aligned}$$

(3.6-10)

Then the tail recursion is rewritable as

$$\begin{aligned} \tilde{\sigma}(n, q) \wedge \tilde{\varphi}(n-1, q) &\Rightarrow \tilde{\varphi}(n, q) \\ \tilde{r}(n-1, q) &\Rightarrow \tilde{\varphi}(n, q). \end{aligned} \quad (3.6-11)$$

According to the equational representation of implication, let

$$\check{h}(n-1, q) = 1 - \tilde{\sigma}(n, q) \cdot \tilde{\varphi}(n-1, q) + \tilde{\sigma}(n, q) \cdot \tilde{\varphi}(n-1, q) \cdot \tilde{\varphi}(n, q). \quad (3.6-12)$$

Since by definition $\hat{r}(n-1, q) = \tilde{\sigma}(n, q) \cdot \tilde{\varphi}(n-1, q)$. Then

$$\check{\varphi}(n, q) = \frac{\hat{h}(n-1, q) + \tilde{\sigma}(n, q) \cdot \tilde{\varphi}(n-1, q) - 1}{\tilde{\sigma}(n, q) \cdot \tilde{\varphi}(n-1, q)} \quad (3.6-13)$$

with boundary condition $n = 0$.

Converting rules based system of inference to the problem of constrained minimization

[00236]

Consider the following example.

Converting rules to constraints

The preceding discussion has established an algorithm for convert rules of the form

$$h(q) \Leftarrow \varphi_1(q) \wedge \varphi_2(q) \wedge \cdots \wedge \varphi_m(q) \quad (3.6-14)$$

To constraints of the form

$$\check{h}(q) = \check{\varphi}_1(q) \cdot \check{\varphi}_2(q) \cdot \cdots \cdot \check{\varphi}_m(q). \quad (3.6-15)$$

Decision Element (DE)

[00237] A diagram of the Decision Element Architecture is shown in illustration OO.

It is composed six elements:

- Programmable search engine (PSE)
- Internal heterogeneous database (IHDB)
- Inference engine (IE)
- Inference rule base (IRB)
- API / user interface
- Network interface (NI)

A functional description of these elements follows.

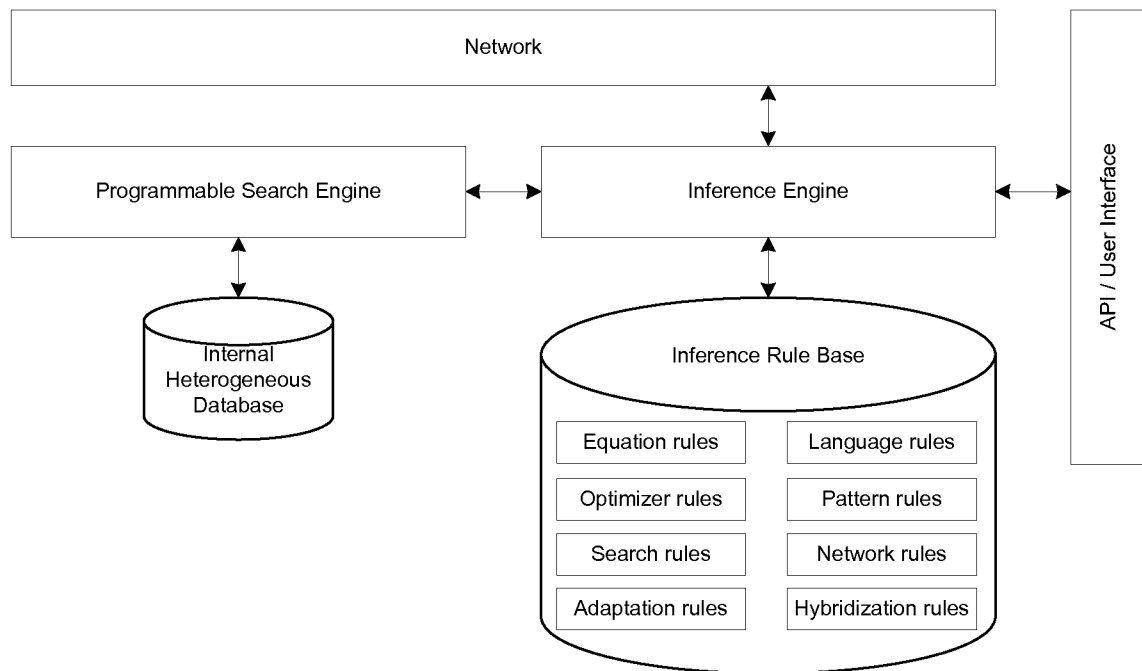


Illustration OO

List of External Repositories (LER)

[00238] A DE has a List of External Repositories (LER). Each entry in an LER includes 1) a protocol, 2) a heading sub-list, and 3) a translation grammar. Each protocol entry prescribes the access procedure to the corresponding external knowledge repository. Each heading sub-list entry contains a summary of the knowledge contents of the corresponding repository. Finally, each translation

grammar entry provides a procedure for converting knowledge elements of the corresponding repository in to the rule representation in the IHDB of the DE.

Programmable search engine

[00239] The programmable search engine implements a standard hashing algorithm for detecting active rules as a function of the current instantiation of the variables in a variable buffer (VB) of the IE, and the contents of the active rule buffer (ARB). The VB contains the variables that form part of the query and all additional variables incorporated to this buffer during the inference process (IP). The VB includes all relevant data from the EKB beneficial to perform the query. The IP will be described below. The ARB contains all the rules that are currently active in the IP.

[00240] The search hashing algorithm is characterized by the search rules in the Inference Rule Base (see Figure 4).

Internal heterogeneous database

[00241] The IHDB is the repository of the application clauses associated with the DE. These encode the domain of knowledge characterizing the expertise of the DE. For example in a medical application, a decision element may deal with expertise on heart illnesses, and the corresponding clauses might encode diagnoses and treatments for these diseases.

Inference engine

[00242] The IE encodes an algorithm, the IP, for assigning values to the variables appearing in the query. The IP is summarized in the block diagram of Illustration PP.

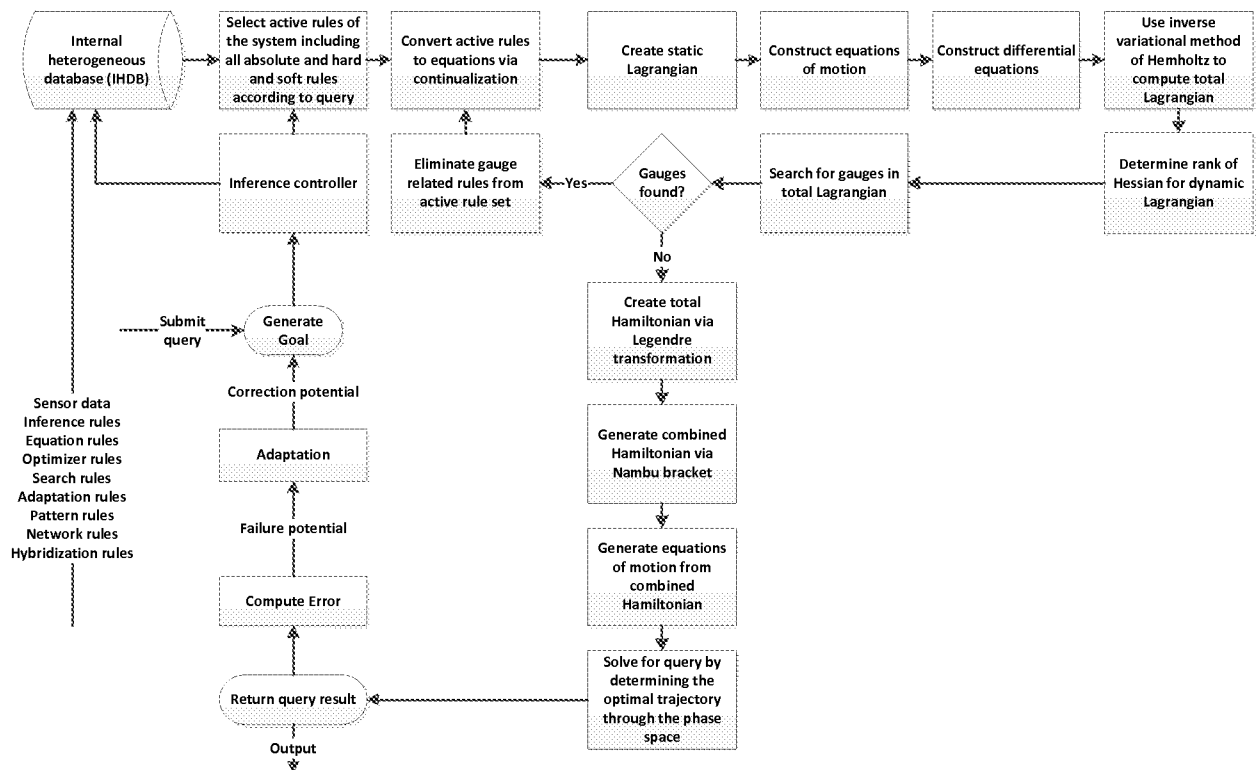


Illustration PP

Inference rule types

[00243] The DE incorporates inference rules (IR) that are a collection of rules for transforming and inferring instantiations of the goal. These rules provide the Inference Engine with directives for processing database rules to give a satisfactory instantiation to a given query or to request additional information so that a satisfactory instantiation can be generated. They are organized according to their functionality as follows. (See Illustration PP)

Equation rules

[00244] These rules include the formal rules for inference. This includes all rules for natural language modeling from first principles.

Optimizer rules

[00245] These rules include rules for finding the interior point in optimization.

Search rules

[00246] These rules include rules for identifying the nature of insufficient potential. The goal is to apply these rules to acquire additional information required to satisfy the optimization goal.

Adaptation rules

[00247] Adaptation rules are used to update the soft rules to relax them further to reduce the complexity and constraints of the optimization problem. The adaptation also serves to update the search rules to improve information acquisition.

Language statistics and pattern rules

[00248] These rules embody the machine learning models.

Network rules

[00249] These rules define how information is distributed over the network and what information is available from which resources.

Hybridization rules

[00250] The rules define how other rules may be combined.

User interface

[00251] The UI provides the utilities for entering queries, pragma rules, displaying query answers, status and for general interaction with the IE.

Network interface

[00252] The NI provides a generic mechanism for interacting with other DE's via a procedure termed companionship. The companionship procedure implements the active coupling for the cooperation of the DE's in query resolution. This procedure is not hierarchical and implements a Pareto Agreement set strategy as the mechanism for CDI.

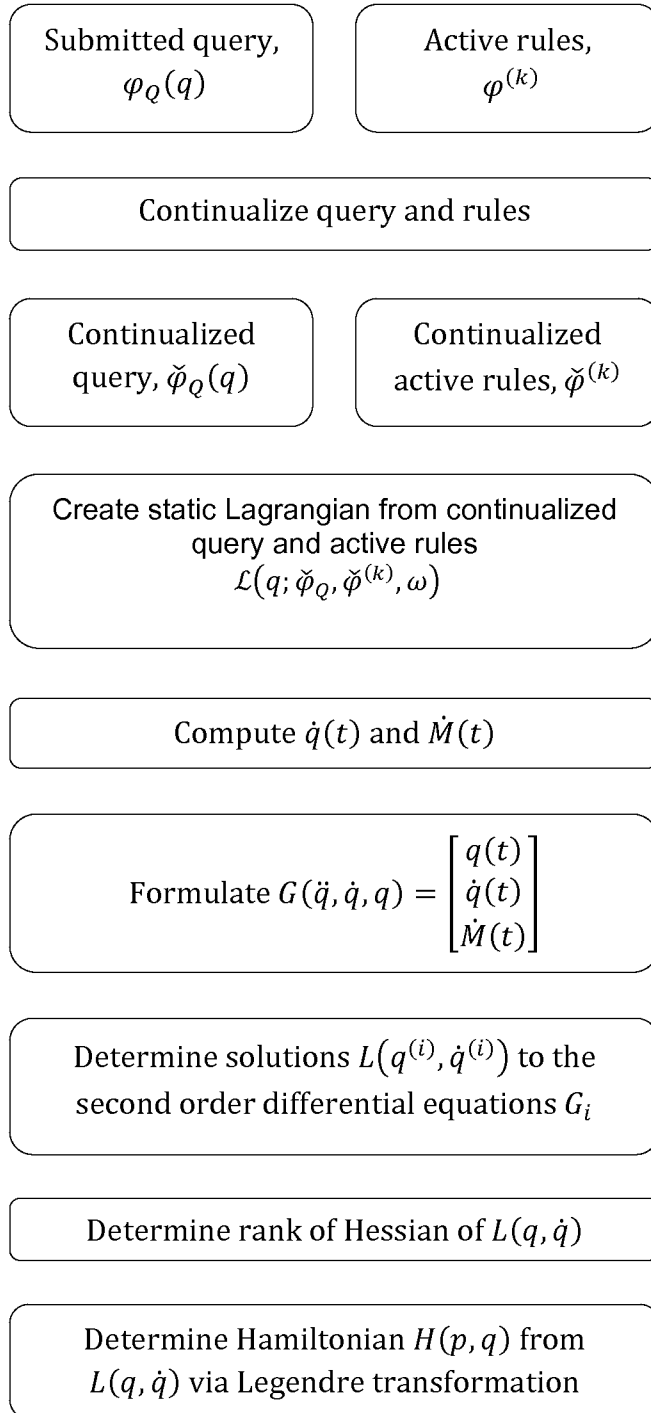
Query Language Interface (QLI)

Illustration QQ

Process for determining active constraints

[00253] The process for determining active constraints is available elsewhere herein.

Minimization Function Generator (MFG) and process for determining active constraints

[00254] The minimization function generator converts a query to a minimization function. Again, we assume without loss of generality the entire set of canonical coordinates q is an argument to any proposition φ_i . In practice, we may further assume it is possible to apply the particular required coordinates as need to the proposition or function in question. Then let $\varphi^{(k)}$ be the set of propositions associated with DE_k in the context of query Q . These propositions are composed of the proposition associated with the query $\varphi_Q(q)$, and other propositions $\varphi_i(q)$, comprising the constraints of the system. The proposition $\varphi_Q(q)$ associated with a given query Q can be converted to an equation $\check{\varphi}_Q(q)$. Queries that are satisfiable specify a set.

$$\{q | \varphi_Q(q) \leftarrow T\} \quad (3.10-1)$$

Similarly, a satisfied query represented as an equation is also a set

$$\{q | \check{\varphi}_Q(q) = 1\}. \quad (3.10-2)$$

Relaxing the values that $\check{\varphi}_Q(\cdot)$ can take to include the unit interval so that soft rules are incorporated yields the following constrained optimization expression. Let $J(q) = (\check{\varphi}_Q(q) - 1)^2$. Then specify the optimization

$$\min_q J(q) \quad (3.10-3)$$

Subject to:

1. $\check{\varphi}_Q(q) \leq 1$
2. $\check{\varphi}_Q(q) \geq 0$
3. A knowledge base on the set $\{\check{\varphi}_1(q), \dots, \check{\varphi}_n(q), \dots, \check{\varphi}_{n+s}(q)\} \subseteq \check{\varphi}^{(k)}$ which represents a further set of active constraints specific to the problem:
 - a. $\check{\varphi}_i(q) \geq 0$ for $1 \leq i \leq n$,
 - b. $\check{\varphi}_i(q) \leq 1$ or, equivalently $-(\check{\varphi}_i(q) - 1) \geq 0$ for $1 \leq i \leq n$,
 - c. and in the case of absolute rules $\check{\varphi}_l(q)(1 - \check{\varphi}_l(q)) = 0$ for $n < l \leq n + s$.

Introduce the indicator functions

$$V_{\check{\varphi}_i}^- = \begin{cases} 0 & \check{\varphi}_i(q) \geq 0 \\ \infty & \check{\varphi}_i(q) < 0 \end{cases} \quad (3.10-4)$$

and

$$V_{\check{\varphi}_i}^+ = \begin{cases} 0 & 1 - \check{\varphi}_i(q) \geq 0 \\ \infty & 1 - \check{\varphi}_i(q) < 0 \end{cases} \quad (3.10-5)$$

which yields the two logarithmic barrier functions

$$\tilde{V}_{\check{\varphi}_i}^- = -\log(\check{\varphi}_i(q)) \quad (3.10-6)$$

and

$$\tilde{V}_{\check{\varphi}_i}^+ = -\log(1 - \check{\varphi}_i(q)). \quad (3.10-7)$$

According to the method of Lagrange multipliers, combine this with the equality constraints to form the static Lagrangian function

$$\begin{aligned}
\mathcal{L} \left(q; \check{\varphi}_Q, \check{\varphi}^{(k)}, \omega_1^{(+)}, \dots, \omega_n^{(+)}, \omega_{n+1}^{(-)}, \dots, \omega_{2n}^{(-)}, \omega_{2n+1}^{(\lambda)}, \dots, \omega_{2n+s}^{(\lambda)}, \omega_{2n+s+1}^{(Q)}, \omega_{2n+s+2}^{(Q)} \right) \\
= \check{\varphi}_Q(q) + \sum_{i=1}^n \left[\omega_i^{(+)} \check{V}_{\check{\varphi}_i}^+ + \omega_{n+i}^{(-)} \check{V}_{\check{\varphi}_i}^- \right] + \sum_{l=1}^s \omega_{2n+l}^{(\lambda)} \check{\varphi}_l(q) (1 - \check{\varphi}_l(q)) \\
- \omega_{2n+s+1}^{(Q)} \log(\check{\varphi}_Q(q)) - \omega_{2n+s+2}^{(Q)} \log(1 - \check{\varphi}_Q(q)),
\end{aligned}
\tag{3.10-8}$$

the roots of which can be found using a formulation of Newton-Raphson. Since $\mathcal{L}$ here includes absolute, hard and soft rules we may call it the total static Lagrangian for DE_k and refer to it as $\mathcal{L}_k^{(T)}$.

Construct equations of motion

[00255] Information for equations of motion is available elsewhere herein.

Query Response Engine (QRE) which includes process for constructing differential equations

Application of Newton-Raphson

[00256] Consider a continuous analog of the independent variables of $\mathcal{L}(\cdot)$

$$q = q(t) = \begin{bmatrix} q^{(1)}(t) \\ \vdots \\ q^{(v)}(t) \end{bmatrix}
\tag{3.12-1}$$

where each of the v total independent variables of $\mathcal{L}(\cdot)$ is mapped to its corresponding position in $q(t)$, the column vector that is represented with a lower-case q . To reiterate, the independent variable t refers algorithmic time as opposed to physical time which may also be represented in the system. The corresponding unconstrained optimization goal can be written as

$$\min_{q_1, \dots, q_v} \mathcal{L} \left(q^{(1)}(t), \dots, q^{(v)}(t) \right)
\tag{3.12-2}$$

so that $\nabla \mathcal{L}(q)$

$$\nabla \mathcal{L}(q(t)) = \begin{bmatrix} \frac{\partial \mathcal{L}}{\partial q^{(1)}} \\ \vdots \\ \frac{\partial \mathcal{L}}{\partial q^{(v)}} \end{bmatrix} = \begin{bmatrix} \nabla \mathcal{L}_1 \\ \vdots \\ \nabla \mathcal{L}_v \end{bmatrix} = 0, \quad (3.12-3)$$

with positive definite Hessian matrix

$$\nabla^2 \mathcal{L}(q(t)) = \begin{bmatrix} \frac{\partial^2 \mathcal{L}}{\partial q^{(1)} \partial q^{(1)}} & \cdots & \frac{\partial^2 \mathcal{L}}{\partial q^{(1)} \partial q^{(v)}} \\ \vdots & \ddots & \vdots \\ \frac{\partial^2 \mathcal{L}}{\partial q^{(v)} \partial q^{(1)}} & \cdots & \frac{\partial^2 \mathcal{L}}{\partial q^{(v)} \partial q^{(v)}} \end{bmatrix} = \begin{bmatrix} \nabla \mathcal{L}_{11} & \cdots & \nabla \mathcal{L}_{1v} \\ \vdots & \ddots & \vdots \\ \nabla \mathcal{L}_{v1} & \cdots & \nabla \mathcal{L}_{vv} \end{bmatrix} \Rightarrow 0. \quad (3.12-4)$$

Write the recursion for Newton's method

$$q_{(k+1)}(t) = q_{(k)}(t) - \left(\nabla^2 \mathcal{L}(q_{(k)}(t)) \right)^{-1} \nabla \mathcal{L}(q_{(k)}(t)). \quad (3.12-5)$$

This is equivalently rewritten

$$\frac{q_{(k+1)}(t) - q_{(k)}(t)}{\delta} = -\frac{1}{\delta} \left(\nabla^2 \mathcal{L}(q_{(k)}(t)) \right)^{-1} \nabla \mathcal{L}(q_{(k)}(t)). \quad (3.12-6)$$

Via continualization we approximate the derivative

$$\dot{q}(t) = \frac{dq(t)}{dt} = - \left(\nabla^2 \mathcal{L}(q(t)) \right)^{-1} \nabla \mathcal{L}(q(t)). \quad (3.12-7)$$

Translation of inverted matrix

[00257] Consider M, an invertible and positive definite matrix. Then we make the following provable assertions.

1. $A^T A$ is symmetric.
2. $-A^T A$ has negative eigenvalues.

Define

$$\frac{dM(t)}{dt} = -A^T A M(t) + A^T \quad (3.12-8)$$

Then as $t \rightarrow \infty$, $M(t) \rightarrow A^{-1} = \nabla^2 \mathcal{L}(q_{(k)}(t))^{-1}$. Using (3.12-3) and (3.12-4) approximate $\dot{q}(t)$ by rewriting the derivative in the context of $M(t)$. This yields the following two equations.

$$\dot{q}(t) = -M(t) \nabla L(q(t)) = \begin{bmatrix} m_{11} & \cdots & m_{1v} \\ \vdots & \ddots & \vdots \\ m_{v1} & \cdots & m_{vv} \end{bmatrix} \begin{bmatrix} \nabla \mathcal{L}_1 \\ \vdots \\ \nabla \mathcal{L}_v \end{bmatrix} = \begin{bmatrix} m_{11} \nabla \mathcal{L}_1 + \cdots + m_{1v} \nabla \mathcal{L}_v \\ \vdots \\ m_{v1} \nabla \mathcal{L}_1 + \cdots + m_{vv} \nabla \mathcal{L}_v \end{bmatrix} \quad (3.12-9)$$

$$\begin{aligned} \frac{dM(t)}{dt} &= -\left(\nabla^2 \mathcal{L}(q(t))\right)^T \left(\nabla^2 \mathcal{L}(q(t))\right) M(t) + \left(\nabla^2 \mathcal{L}(q(t))\right)^T \\ &= -\begin{bmatrix} \nabla \mathcal{L}_{11} & \cdots & \nabla \mathcal{L}_{v1} \\ \vdots & \ddots & \vdots \\ \nabla \mathcal{L}_{1v} & \cdots & \nabla \mathcal{L}_{vv} \end{bmatrix} \begin{bmatrix} \nabla \mathcal{L}_{11} & \cdots & \nabla \mathcal{L}_{1v} \\ \vdots & \ddots & \vdots \\ \nabla \mathcal{L}_{v1} & \cdots & \nabla \mathcal{L}_{vv} \end{bmatrix} \begin{bmatrix} m_{11} & \cdots & m_{1v} \\ \vdots & \ddots & \vdots \\ m_{v1} & \cdots & m_{vv} \end{bmatrix} + \begin{bmatrix} \nabla \mathcal{L}_{11} & \cdots & \nabla \mathcal{L}_{v1} \\ \vdots & \ddots & \vdots \\ \nabla \mathcal{L}_{1v} & \cdots & \nabla \mathcal{L}_{vv} \end{bmatrix} \\ &= -\begin{bmatrix} \nabla \mathcal{L}_{11}^2 + \cdots + \nabla \mathcal{L}_{v1}^2 & \cdots & \nabla \mathcal{L}_{11} \nabla \mathcal{L}_{1v} + \cdots + \nabla \mathcal{L}_{v1} \nabla \mathcal{L}_{vv} \\ \vdots & \ddots & \vdots \\ \nabla \mathcal{L}_{11} \nabla \mathcal{L}_{1v} + \cdots + \nabla \mathcal{L}_{v1} \nabla \mathcal{L}_{vv} & \cdots & \nabla \mathcal{L}_{1v}^2 + \cdots + \nabla \mathcal{L}_{vv}^2 \end{bmatrix} \begin{bmatrix} m_{11} & \cdots & m_{1v} \\ \vdots & \ddots & \vdots \\ m_{v1} & \cdots & m_{vv} \end{bmatrix} \\ &\quad + \begin{bmatrix} \nabla \mathcal{L}_{11} & \cdots & \nabla \mathcal{L}_{v1} \\ \vdots & \ddots & \vdots \\ \nabla \mathcal{L}_{1v} & \cdots & \nabla \mathcal{L}_{vv} \end{bmatrix} \\ &= -\begin{bmatrix} (\nabla \mathcal{L}_{11}^2 + \cdots + \nabla \mathcal{L}_{v1}^2)m_{11} + \cdots & \cdots & (\nabla \mathcal{L}_{11}^2 + \cdots + \nabla \mathcal{L}_{v1}^2)m_{1v} + \cdots \\ +(\nabla \mathcal{L}_{11} \nabla \mathcal{L}_{1v} + \cdots + \nabla \mathcal{L}_{v1} \nabla \mathcal{L}_{vv})m_{v1} & \cdots & +(\nabla \mathcal{L}_{11} \nabla \mathcal{L}_{1v} + \cdots + \nabla \mathcal{L}_{v1} \nabla \mathcal{L}_{vv})m_{vv} \\ +\nabla \mathcal{L}_{11} & \cdots & +\nabla \mathcal{L}_{v1} \\ \vdots & \ddots & \vdots \\ (\nabla \mathcal{L}_{11} \nabla \mathcal{L}_{1v} + \cdots + \nabla \mathcal{L}_{v1} \nabla \mathcal{L}_{vv})m_{11} + \cdots & \cdots & (\nabla \mathcal{L}_{11} \nabla \mathcal{L}_{1v} + \cdots + \nabla \mathcal{L}_{v1} \nabla \mathcal{L}_{vv})m_{1v} + \cdots \\ +(\nabla \mathcal{L}_{1v}^2 + \cdots + \nabla \mathcal{L}_{vv}^2)m_{v1} & \cdots & +(\nabla \mathcal{L}_{1v}^2 + \cdots + \nabla \mathcal{L}_{vv}^2)m_{vv} \\ +\nabla \mathcal{L}_{1v} & \cdots & +\nabla \mathcal{L}_{vv} \end{bmatrix} \\ &= \begin{bmatrix} \nabla m_{11} & \cdots & \nabla m_{1v} \\ \vdots & \ddots & \vdots \\ \nabla m_{v1} & \cdots & \nabla m_{vv} \end{bmatrix} \end{aligned} \quad (3.12-10)$$

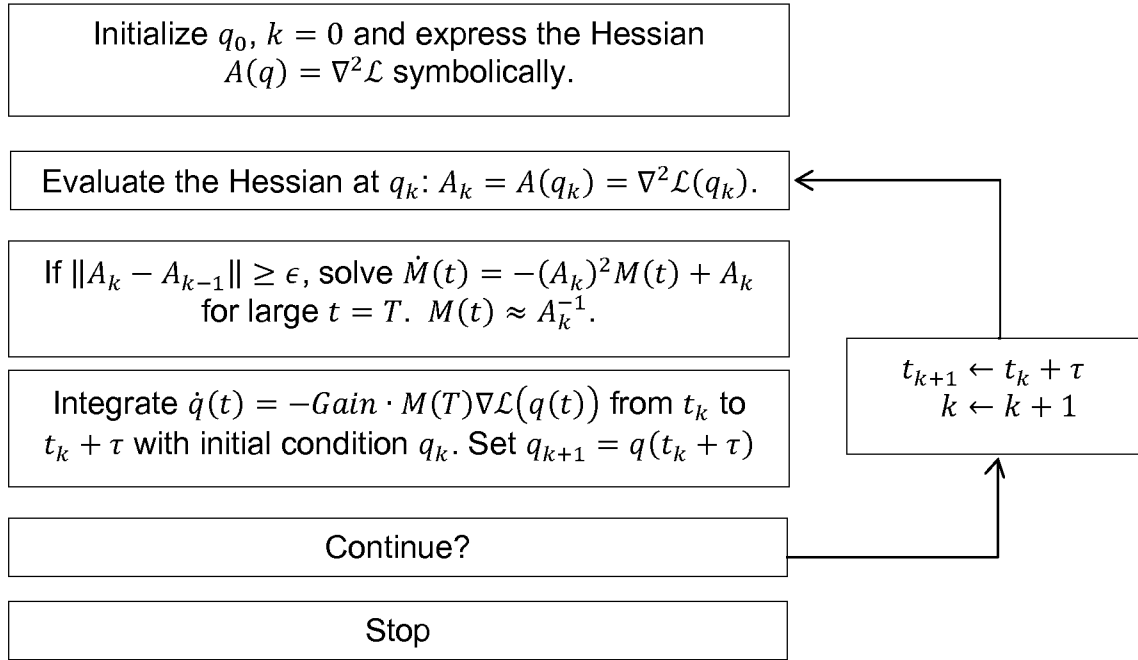
The approximation proceeds as follows:

1. Fix $M(0) = \nabla^2 \mathcal{L}(q(0))$ and $\nabla^2 \mathcal{L}(q(t))$.
2. Use the variation of constants formula to solve

$$M(T) = e^{-[\nabla^2 \mathcal{L}(q(T))]^2 T} M(0) + \left[\int_0^T e^{-[\nabla^2 \mathcal{L}(q(\tau))]^2 (T-\tau)} d\tau \right] \nabla^2 \mathcal{L}(q(T))$$

applying the Magnus expansion to compute the integral.

[00258] The following figure documents the flow of computation, flowing down unless otherwise indicated.



Process for determining dynamic Lagrangian via Hemholtz equations

Given

$$G_i(\ddot{q}, \dot{q}, q) = \sum_{j=1}^n W_{i,j}(\dot{q}, q) (\dot{q}, q) \ddot{q}^{(j)} + K_i(\dot{q}, q) = 0 \quad j = 1, \dots, n \quad (3.12-11)$$

If the three conditions

$$\frac{\partial G_i}{\partial \ddot{q}^{(j)}} = \frac{\partial G_j}{\partial \ddot{q}^{(i)}},$$

$$\frac{\partial G_i}{\partial \dot{q}^{(j)}} + \frac{\partial G_j}{\partial \dot{q}^{(i)}} = \frac{d}{dt} \left(\frac{\partial G_i}{\partial \ddot{q}^{(j)}} + \frac{\partial G_j}{\partial \ddot{q}^{(i)}} \right),$$

$$\frac{\partial G_i}{\partial q^{(j)}} - \frac{\partial G_j}{\partial q^{(i)}} = \frac{1}{2} \frac{d}{dt} \left(\frac{\partial G_i}{\partial \dot{q}^{(j)}} - \frac{\partial G_j}{\partial \dot{q}^{(i)}} \right),$$

(3.12-12)

with $i, j = 1, \dots, n$ hold, then

$$\sum_{j=1}^n \frac{\partial^2 L}{\partial \dot{q}^{(i)} \partial \dot{q}^{(j)}} \ddot{q}^{(j)} + \frac{\partial^2 L}{\partial q^{(j)} \partial \dot{q}^{(i)}} - \frac{\partial L}{\partial q^{(i)}} = G_i, \quad i = 1, \dots, n$$

(3.12-13)

This is a second order, linear hyperbolic differential equation on the Lagrangian L . It can be solved efficiently by the method of characteristics.

Let

$$G(\ddot{q}, \dot{q}, q) = \begin{bmatrix} q(t) \\ \dot{q}(t) \\ \dot{M}(t) \\ q^{(1)} \\ \vdots \\ q^{(v)} \\ m_{11} \nabla \mathcal{L}_1 + \dots + m_{1v} \nabla \mathcal{L}_v \\ \vdots \\ m_{v1} \nabla \mathcal{L}_1 + \dots + m_{vv} \nabla \mathcal{L}_v \\ \nabla m_{11} \\ \vdots \\ \nabla m_{v1} \\ \vdots \\ \nabla m_{1v} \\ \vdots \\ \nabla m_{vv} \end{bmatrix}$$

(3.12-14)

Process for determining Hessian rank of dynamic Lagrangian

[00259] Information for determining Hessian rank of dynamic Lagrangian is available elsewhere herein.

Converting the Lagrangian to the Hamiltonian via the Legendre transformation.

[00260] In our formulation the Lagrangian, $L_k^{(T)}(q, \dot{q}; \omega)$, may be converted to the Hamiltonian using the Legendre transformation, so that

$$\begin{aligned} H_k^{(T)}(q, p; \omega) &= \frac{\partial L_k^{(T)}}{\partial \dot{q}} \dot{q} - L_k^{(T)}(q, \dot{q}; \omega) \\ &= p^T \dot{q} - L_k^{(T)}(q, \dot{q}; \omega) \end{aligned} \quad (3.12-15)$$

Pareto Multi-Criteria Optimization Engine (PMOE)

[00261] Consider the problem of determining the relaxed Pareto optimal solution to a given system query at a given time step. There are N decision elements, $k = 1, \dots, N$. A given decision element, DE_k , has the following associated parameters which are constituent to the ARB:

- A generalized set of coordinates relevant to DE_k , q .
- A generalized set of linearly independent momenta $\{p_a\}$ where the index a refers the linearly independent momenta selected from the canonical set p .
- A set of control parameters ω for hard a soft rules of the system, where $0 \leq \omega_i \leq 1$.

The ARB has the following components which determine the constraints of DE_k :

- The Hamiltonian which identifies the fundamental dynamics of the system of the system for the k 'th decision element denoted

$$H_k^{(o)}(q, \{p_a\}). \quad (3.13-1)$$

- The summation of the first class constraints of the system, which is

$$\sum_i \omega_i f_i(q^{(i)}, \omega_i)$$

(3.13-2)

- The summation of the second class constraints of the system which is

$$\sum_i g_i(q^{(i)}, \omega_i)$$

(3.13-3)

- The Tellegen agent which is a function of the Hamiltonians of the absolute rules of the other $N - 1$ decision elements in the system

$$H_k^{(A)} = F_k^{(A)}(H_1^{(T)}, \dots, H_{k-1}^{(T)}, H_{k+1}^{(T)}, \dots, H_K^{(T)})$$

(3.13-4)

- The total Hamiltonian of the system is denoted $H^{(T)}$.
- Approximations to the various Hamiltonian's are denoted $\hat{H}_k^{(A)}$, $\hat{H}^{(T)}$ and $\hat{H}_k^{(o)}$ for the Tellegen, total, and DE-level Hamiltonians respectively.

System initialization

[00262] Determining the relaxed Pareto optimal point of the system is a process which includes:

- Initialization of N decision elements.
- Synchronization through companionship of each of the N decision elements with its respective Tellegen agent.

[00263] Illustration RR shows the information components of the DE that are constituent to updating and being updated by the network at initialization.

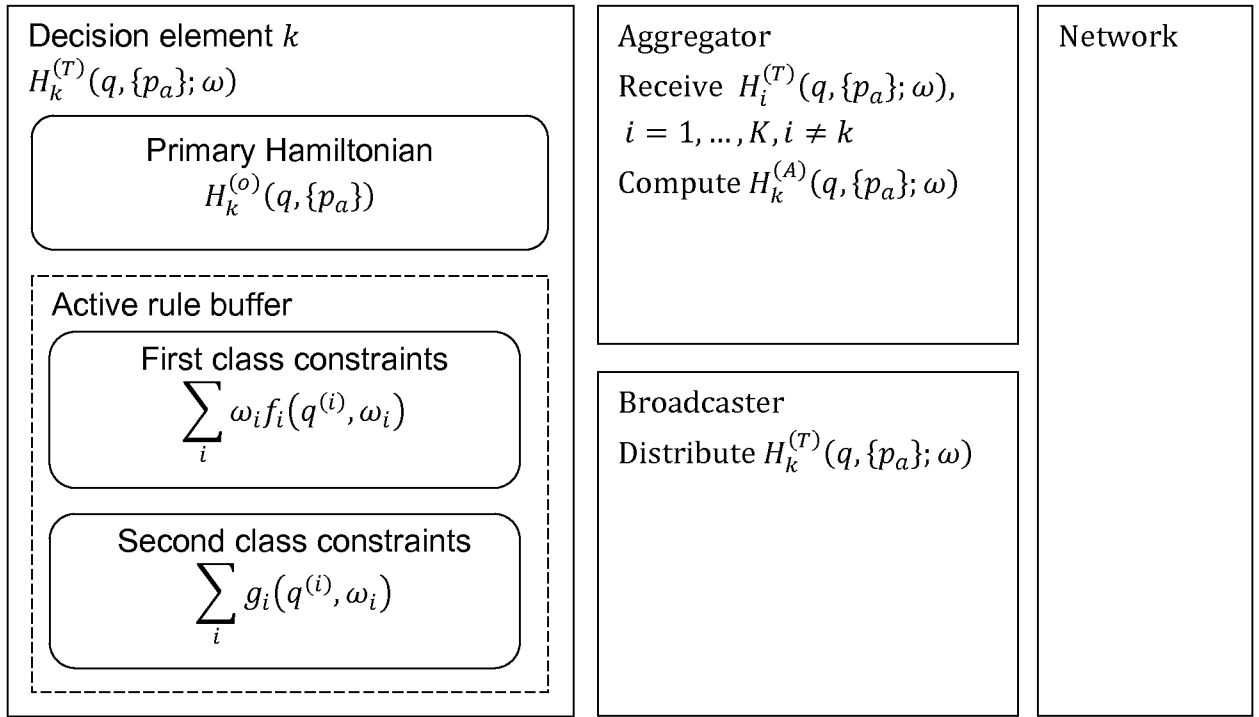


Illustration RR

System operation

[00264] Illustration SS shows how decision elements interact with the network, receive queries, and return results. In this example, the distributed system effectively implements an abstract classifier that has no real implementation. The DE's receive sensor data from the network which includes new available information which may benefit classification. The user submits a query that is received by a DE which then returns a result.

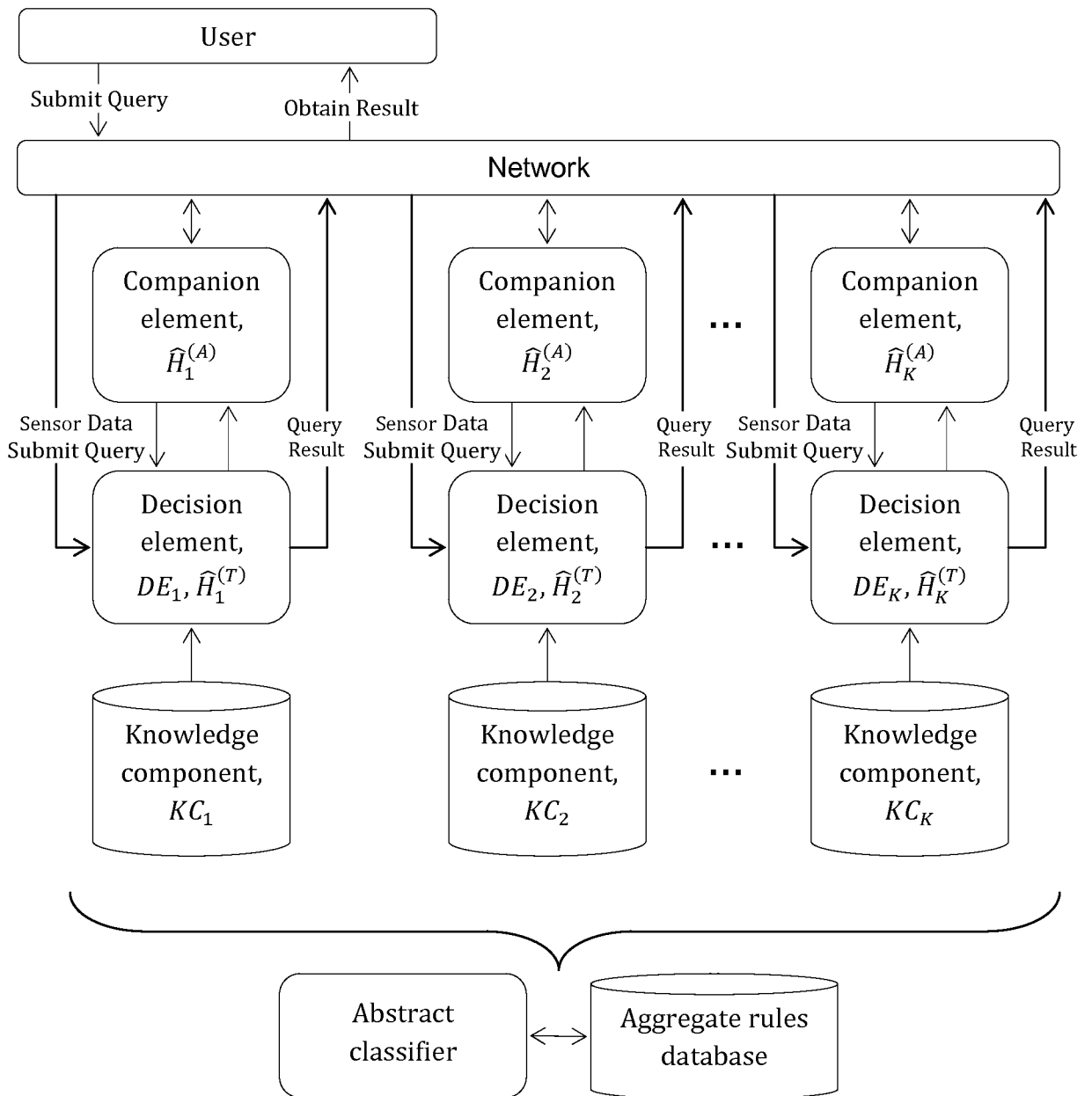


Illustration SS

[00265]

Illustration TT represents the iterative process of updating the Hamiltonian associated with DE_k .

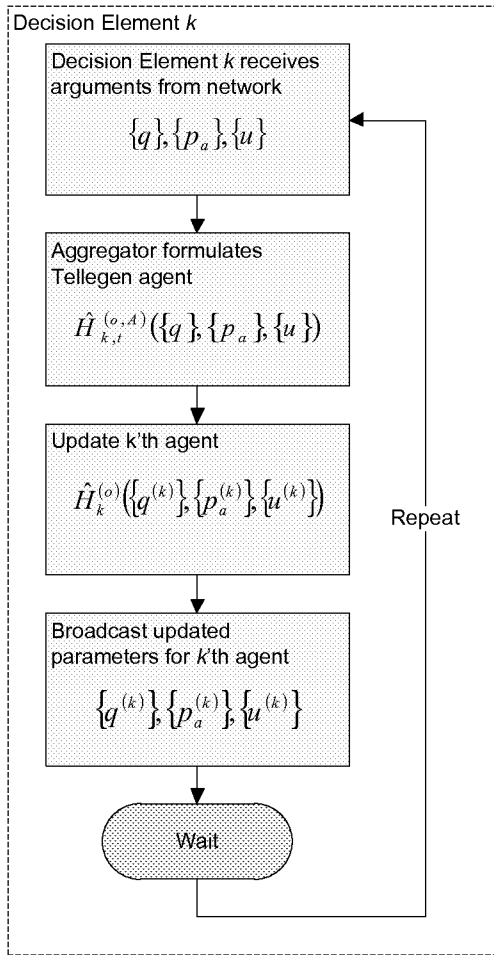


Illustration TT

Gauge Systems in a Hamiltonian Domain

[00266]

The time integral of the Lagrangian $L(q, \dot{q})$ is the action S_L defined as

$$S_L = \int_{t_1}^{t_2} L(q, \dot{q}) dt$$

where $\dot{q} = \frac{dq(t)}{dt}$. The Lagrangian conditions for stationarity are first that

$$\frac{d}{dt} L_{\dot{q}^{(n)}} - L_{q^{(n)}} = 0$$

(3.14-1)

where $n = 1, \dots, N$, $L_{\dot{q}^{(n)}} = \frac{\partial L}{\partial \dot{q}^{(n)}}$, and $L_{q^{(n)}} = \frac{\partial L}{\partial q^{(n)}}$. And, secondarily

$$\left[\sum_{n'=1}^N \ddot{q}^{(n')} \right] L_{\dot{q}^{(n)} \dot{q}^{(n)}} = L_{q^{(n)}} - \dot{q}^{(n)} L_{\dot{q}^{(n)} q^{(n)}} \quad (3.14-2)$$

where $\ddot{q}^{(n')} = \frac{d^2 q^{(n')}}{dt^2}$ and $L_{\dot{q}^{(n)} \dot{q}^{(n)}} = \frac{\partial^2 L}{\partial (\dot{q}^{(n)})^2}$. The generalized accelerations $\ddot{q}^{(n)}$ are immediately determined if $L_{\dot{q}^{(n)} \dot{q}^{(n)}}$ is invertible, or equivalently

$$\det(L_{\dot{q}^{(n)} \dot{q}^{(n)}}) \neq 0 \quad (3.14-3)$$

for $i = 1, \dots, N$. If for some n , $\det(L_{\dot{q}^{(n)} \dot{q}^{(n)}}) = 0$, the acceleration vector $\ddot{q}^{(n)}$ will not be uniquely determined.

[00267] The departing point for the Hamiltonian approach is the definition of conjugate momentum

$$p_n = L_{\dot{q}^{(n)}} \quad (3.14-4)$$

where $n = 1, \dots, N$. We will see that (3.14-3) is the condition of non-invertibility of

$$L_{\dot{q} \dot{q}} = \begin{bmatrix} L_{\dot{q}^{(1)} \dot{q}^{(1)}} & \cdots & L_{\dot{q}^{(1)} \dot{q}^{(N)}} \\ \vdots & \ddots & \vdots \\ L_{\dot{q}^{(N)} \dot{q}^{(1)}} & \cdots & L_{\dot{q}^{(N)} \dot{q}^{(N)}} \end{bmatrix}$$

of the velocities of the functions of the coordinates q and momenta p . In other words, in this case, the momenta defined in (3.14-4) are not all independent. Define the relations that follow from (3.14-4) as

$$\Phi_m(q, p) \quad (3.14-5)$$

where $m = 1, \dots, M$. Write (3.14-4) in vector notation as

$$p = L_{\dot{q}}(q, \dot{q}).$$

Then compatibility demands

$$\Phi_m(q, L_{\dot{q}}(q, \dot{q})) = 0$$

is an identity with $m = 1, \dots, M$.

[00268] Relations specified in (3.14-5) are called primary constraints. For simplicity let's assume that $\text{rank}(L_{\dot{q}\dot{q}})$ is constant throughout the phase space, $(q, \dot{q})$, so that (3.14-5) defines a submanifold smoothly embedded in the phase space. This manifold is known as the primary constraint surface. Let

$$\text{rank}(L_{\dot{q}\dot{q}}) = N - M' \quad (3.14-6)$$

Then there are M' independent constraints among (3.14-5) and the primary constraint surface is a phase space submanifold of dimension $2N - M'$.

[00269] We do not assume that all the constraints are linearly independent so that

$$M' \leq M. \quad (3.14-7)$$

It follows from (3.14-5) that the inverse transformation from the p 's to the q 's is multivalued. That is, given q, p that satisfies (3.14-5), the inverse image $(q, \dot{q})$ that satisfies

$$p = \left(\frac{\partial L}{\partial \dot{q}} \right)^T \quad (3.14-8)$$

is not unique, since (3.14-8) defines a map from a $2N$ -dimensional manifold $(q, \dot{q})$ to the smaller $(2N - M')$ -dimensional manifold. Thus the inverse image of the points of (3.14-5) form a manifold of dimension M' .

Conditions on the Constraint Function

[00270] There exist many equivalent ways to represent a given surface by means of equations of the form of (3.14-5). For example the surface $p_1 = 0$ can be represented equivalently by $p_1^2 = 0$, $\sqrt{|p_1|} = 0$, or redundantly by $p_1 = 0$ and $p_1^2 = 0$. To use the Hamiltonian formalism, it is necessary to impose some restrictions which the regularity conditions for the constraints.

Regularity Conditions

[00271] The $(2N - M')$ -dimensional constraint surface $\phi_m(q, p)$ should be covered of open region: in each region the constraints can be split into independent constraints

$$\{\phi_{m'} | m' = 1, \dots, M'\}.$$

Their Jacobian matrix

$$\left\{ \frac{\partial \phi_{m'}}{\partial p_n, q^{(n)}} \right\} = \begin{bmatrix} \frac{\partial \phi_1}{\partial p_1, q^{(1)}} & \dots & \frac{\partial \phi_1}{\partial p_n, q^{(n)}} \\ \vdots & \ddots & \vdots \\ \frac{\partial \phi_{m'}}{\partial p_1, q^{(1)}} & \dots & \frac{\partial \phi_{m'}}{\partial p_n, q^{(n)}} \end{bmatrix}$$

with $m' = 1, \dots, M'$ and $n = 1, \dots, N$, is of rank M' .

[00272] The dependent constraints ϕ_m , $m = M' + 1, \dots, M$ of the other $\phi_{m'} = 0 \Rightarrow \phi_m = 0$. Alternatively the condition on the Jacobian.

1. The function $\phi_{m'}$ can be taken locally as the first M' coordinates of a new regular system in the vicinity of the constraint surface or the differentials $d\phi_1, \dots, d\phi_{M'}$ are locally linearly independent:

$$d\phi_1 \wedge \dots \wedge d\phi_{M'} \neq 0 \quad (3.14-9)$$

2. The variations $\delta\phi_{m'}$ are of order ϵ for arbitrary variations $\delta q^{(i)}$, δp_i of order ϵ (Dirac's approach).

Theorem 3.14.1. If a smooth, phase space function G vanishes on $\{\phi_m = 0\}$ then

$$G = \sum_{m=1}^M g^{(m)} \phi_m \quad (3.14-10)$$

Proof: (local proof). Set $\phi_{m'}$, $m' = 1, \dots, M'$ as coordinates $(y_{m'}, x_\alpha)$ with $y_{m'} = \phi_{m'}$. In these coordinates $G(0, x) = 0$ and

$$\begin{aligned} G(y, x) &= \int_0^1 \frac{d}{dt} G(ty, x) dt \\ &= \sum_{m'=1}^{M'} y_{m'} \int_0^1 \frac{\partial}{\partial y_{m'}} G(ty, x) dt \\ &= \sum_{m'=1}^{M'} g^{(m')}(y, x) \phi_{m'}(y, x) \end{aligned}$$

with

$$g^{(m')}(y, x) = \int_0^1 \frac{\partial}{\partial y_{m'}} G(ty, x) dt. \quad (3.14-11)$$

Theorem 3.14.2. If the sum $\sum (\lambda^{(n)} \delta q^{(n)} + \mu_n \delta p_n) = 0$ for arbitrary variations $\delta q^{(i)}$, δp_i tangent to the constraint surface $\{\phi_m(q, p) = 0 | m = 1, \dots, M\}$, then

$$\lambda^{(n)} = \sum_{m=1}^M u^{(m)} \frac{\partial \phi_m}{\partial q^{(n)}} \quad (3.14-12)$$

$$\mu_n = \sum_{m=1}^M u^{(m)} \frac{\partial \phi_m}{\partial p_n} \quad (3.14-13)$$

Proof. The dimension of $\{\phi_m\}$ is $2N - M'$. Thus the variations at a point (p, q) forms a $2N - M'$ dimensional space

$$\sum_{n=1}^N (\lambda^{(n)} \delta q^{(n)} + \mu_n \delta p_n) = 0 \quad (3.14-14)$$

By the singularity assumption, there exists exactly M' solutions to (3.14-14). Clearly, the gradients $\left\{ \frac{\partial \phi_{m'}}{\partial q^{(n)}} \right\}$ and $\left\{ \frac{\partial \phi_{m'}}{\partial p_n} \right\}$ are linearly independent. They are the basis for solutions to (3.14-14).

[00273] Note that in the presence of redundant constraints, the functions $u^{(m)}$ exist but are not unique.

Canonical Hamiltonian

[00274] The Hamiltonian in canonical coordinates is

$$H(q, p) = \sum_{n=1}^N \dot{q}^{(n)} p_n - L(q, \dot{q}) \quad (3.14-15)$$

The rate $\dot{q}$ enters through the combination through conjugate momenta defined for each coordinate

$$p_n(q, \dot{q}) = L_{\dot{q}^{(n)}}(q, \dot{q}) \quad (3.14-16)$$

This remarkable property is essential for the Hamiltonian approach. It is verified by evaluating the change δH involved by arbitrary independent variations of position and velocities.

$$\begin{aligned} \delta H &= \sum_{n=1}^N (\dot{q}^{(n)} \delta p_n + \delta \dot{q}^{(n)} p_n) - \delta L \\ &= \sum_{n=1}^N (\dot{q}^{(n)} \delta p_n + \delta \dot{q}^{(n)} p_n) - \sum_{n=1}^N (L_{q^{(n)}} \delta q^{(n)} + L_{\dot{q}^{(n)}} \delta \dot{q}^{(n)}) \end{aligned} \quad (3.14-17)$$

Utilizing (3.14-16) in (3.14-17) yields

$$\delta H = \sum_{n=1}^N \left(\dot{q}^{(n)} \delta p_n - L_{q^{(n)}} \delta q^{(n)} \right) \quad (3.14-18)$$

The Hamiltonian defined by (3.14-15) is not unique as a function of p, q . This can be inferred from (3.14-18) by noticing that $\{\delta p_n | n = 1, \dots, N\}$ are not all independent. They are restricted to preserve the primary constraints $\phi_m \approx 0$ which are identities when the p 's are expressed as functions of q 's via (3.14-16).

[00275] Using the definition of the differential in several variables applied to $\delta H = \delta H(\{q^{(n)}\}, \{p_n\})$, (3.14-18) can be rewritten

$$\sum_{n=1}^N \left(\frac{\partial H}{\partial q^{(n)}} \delta q^{(n)} + \frac{\partial H}{\partial p_n} \delta p_n \right) = \sum_{n=1}^N \left(\dot{q}^{(n)} \delta p_n - \delta q^{(n)} \frac{\partial L}{\partial q^{(n)}} \right)$$

or

$$\sum_{n=1}^N \left(\frac{\partial H}{\partial q^{(n)}} + \frac{\partial L}{\partial q^{(n)}} \right) \delta q^{(n)} + \sum_{n=1}^N \left(\frac{\partial H}{\partial p_n} - \dot{q}^{(n)} \right) \delta p_n = 0 \quad (3.14-19)$$

From theorem 2 we then conclude for each n that.

$$\frac{\partial H}{\partial q^{(n)}} + \frac{\partial L}{\partial q^{(n)}} = \sum_{m=1}^M u^{(m)} \frac{\partial \phi_m}{\partial q^{(n)}}$$

and

$$\frac{\partial H}{\partial p_n} - \dot{q}^{(n)} = \sum_{m=1}^M u^{(m)} \frac{\partial \phi_m}{\partial p_n}. \quad (3.14-20)$$

So for each n :

$$\dot{q}^{(n)} = \frac{\partial H}{\partial p_n} + \sum_{m=1}^M u^{(m)} \frac{\partial \phi_m}{\partial p_n}, \quad n = 1, \dots, N \quad (3.14-21)$$

and

$$-\frac{\partial L}{\partial q^{(n)}} = \frac{\partial H}{\partial q^{(n)}} + \sum_{m=1}^M u^{(m)} \frac{\partial \phi_m}{\partial q^{(n)}}, \quad n = 1, \dots, N. \quad (3.14-22)$$

Note that if the constraints are independent, the vectors $\sum_{n=1}^N \frac{\partial \phi_m}{\partial p_n}$, $m = 1, \dots, M$ are also independent because of the regularity conditions (this is proved later). Hence no two sets of $\{u^{(m)} | m = 1, \dots, M\}$ can yield the same velocities via (3.14-21).

[00276] Thus, using

$$\dot{q}^{(n)} = \frac{\partial H}{\partial p_n} + \sum_{m=1}^M u^{(m)}(q, \dot{q}) \frac{\partial \phi_m}{\partial p_n}(q, p(q, \dot{q}))$$

we can find $u^{(m)}(p, \dot{q})$. If we define the transformation from $(q, \dot{q})$ to the manifold $\{\phi_m(q, p) = 0 | m = 1, \dots, M\}$, from $q, \dot{q}, u \rightarrow q, p, u$ by

$$\begin{aligned} q &= q, & n &= 1, \dots, N \\ p_n &= L_{q^{(n)}}(q, \dot{q}), & n &= 1, \dots, N - M' \\ u^{(m)} &= u^{(m)}(q, \dot{q}), & m &= 1, \dots, M' \end{aligned}$$

We see that this transformation is invertible since one has from $q, p, u \rightarrow q, \dot{q}, u$

$$\begin{aligned} q &= q \\ \dot{q}^{(n)} &= \frac{\partial H}{\partial p_n} + \sum_{m=1}^M u^{(m)} \frac{\partial \phi_m}{\partial p_n} \\ \phi_m(q, p) &= 0 \end{aligned}$$

Thus invertibility of the Legendre transformation when

$$\det(L_{\dot{q}\dot{q}}) \neq 0$$

can be regained at the prices of adding extra variables.

Action Principle of the Hamiltonian Form

[00277] With (3.14-21) and (3.14-22) we can rewrite (3.14-1) in the equivalent Hamiltonian form

$$\begin{aligned}\dot{q}^{(n)} &= \frac{\partial H}{\partial p_n} + \sum_{m=1}^M u^{(m)} \frac{\partial \phi_m}{\partial p_n} \\ \dot{p}_n &= -\frac{\partial H}{\partial q^{(n)}} - \sum_{m=1}^M u^{(m)} \frac{\partial \phi_m}{\partial q^{(n)}} \\ \phi_m(q, p) &= 0, \quad m = 1, \dots, M'\end{aligned}\quad (3.14-23)$$

The Hamiltonian Equations (3.14-23) can be derived from the following variational principle:

$$\delta \int_{t_1}^{t_2} \left[\sum_{n=1}^N \dot{q}^{(n)} p_n - H - \sum_{m=1}^M u^{(m)} \phi_m \right] = 0 \quad (3.14-24)$$

for arbitrary variations of $\delta q^{(n)}$, δp_n , and $\delta u^{(m)}$ subject to

$$\delta q(t_1) = \delta q(t_2) = 0$$

where the $u^{(m)}$ appear now as Lagrange multipliers enforcing the primary constraints

$$\phi_m(q, p) = 0, \quad m = 1, \dots, M.$$

Let $F(p, q)$ be an arbitrary function of the canonical variables, then

$$\begin{aligned}\frac{dF}{dt} &= \sum_{n=1}^N \frac{\partial F}{\partial q^{(n)}} \dot{q}^{(n)} + \sum_{n=1}^N \frac{\partial F}{\partial p_n} \dot{p}_n \\ &= \sum_{n=1}^N \frac{\partial F}{\partial q^{(n)}} \left[\frac{\partial H}{\partial p_n} + \sum_{m=1}^M u^{(m)} \frac{\partial \phi_m}{\partial p_n} \right] + \sum_{n=1}^N \frac{\partial F}{\partial p_n} \left[-\frac{\partial H}{\partial q^{(n)}} - \sum_{m=1}^M u^{(m)} \frac{\partial \phi_m}{\partial q^{(n)}} \right] \\ &= [F, H] + \sum_{m=1}^M u^{(m)} [F, \phi_m]\end{aligned}\quad (3.14-25)$$

The equation (3.14-25) introduces the new binary operator $[\cdot, \cdot]$ which is the Poisson bracket and has the form

$$\begin{aligned} [F, G] &= \sum_{n=1}^N \left[\frac{\partial F}{\partial q^{(n)}} \frac{\partial G}{\partial p_n} + \frac{\partial F}{\partial p_n} \frac{\partial G}{\partial q^{(n)}} \right] \\ &= \sum_{n=1}^N \left[F_{q^{(n)}} G_{p_n} + F_{p_n} G_{q^{(n)}} \right] \end{aligned} \quad (3.14-26)$$

Secondary Constraints

[00278] The basic consistency condition is that the primary constraints be preserved

in time. So for

$$F(p, q) = \phi_m(q, p)$$

we should have that $\dot{\phi}_m = 0$. $\{\phi_m(q, p) = 0\}$. So this means

$$[\phi_m, H] + \sum_{m'=1}^{M'} u^{(m')} [\phi_m, \phi_{m'}] = 0 \quad (3.14-27)$$

This equation can either reduce to a relation independent of the $u^{(m')}$, or, it may impose a restriction on the u 's.

$$u = -\{[\phi_m, \phi_{m'}]\}[\phi_m, H](q, p) \quad (3.14-28)$$

In the case (3.14-27) is independent of the u 's (3.14-27) is called a secondary constraint. The fundamental difference of secondary constraints with respect to primary constraints is that primary constraints are the consequence of the definition (3.14-8) while secondary constraints depend on the dynamics.

[00279] If $X(q, p) = 0$ is an external constraint, we must impose a compatibility condition

$$[X, H] + \sum_{m=1}^{M'} u^{(m)} [X, \phi_m] = 0 \quad (3.14-29)$$

Next we need to test whether this constraint:

$$\Phi(p, q) = [X, H] + \sum_{m=1}^{M'} u^{(m)} [X, \phi_m] = 0 \quad (3.14-30)$$

$$(3.14-31)$$

Implies new secondary constraints or whether it only restricts the u 's. After the process is finished we are left with a number of secondary constraints which will be denoted by

$$\phi_k = 0, \quad k = M + 1, \dots, M + K$$

where K is the total number of secondary constraints. In general, it will be useful to denote all the constraints (primary and secondary) in a uniform way as

$$\phi_j(q, p) = 0, \quad j = 1, \dots, M + K = J \quad (3.14-32)$$

We make the same regularity assumptions on the full set of constraints.

Weak and Strong Equations

[00280] Equation (3.14-32) can be written as

$$\phi_j(\cdot) \approx 0 \quad (3.14-33)$$

[00281] To emphasize, the quantity ϕ_j is numerically restricted to be zero but does not vanish throughout the space. What this means is that ϕ_j has non-zero Poisson brackets with the canonical variables.

[00282] Let F, G be functions that coincide on the manifold $\{\phi_j \approx 0 | j = 1, \dots, J\}$ are said to be weakly equal and denoted by $F \approx G$. On the other hand, an equation that holds throughout the entire phase space and not just on the submanifold $\{\phi_j \approx 0\}$ is called strong. Hence, by theorem 1

$$F \approx G \Leftrightarrow F - G = \sum_{j=1}^J c^{(j)}(p, q) \phi_j \quad (3.14-34)$$

Restrictions on the Lagrange Multipliers

[00283] Assume that we have found a complete set of constraints

$$\{\phi_j \approx 0 | j = 1, \dots, J\} \quad (3.14-35)$$

$$[\phi_j, H] + \sum_{m=1}^M u^{(m)} [\phi_j, \phi_m] \approx 0 \quad (3.14-36)$$

We consider (3.14-36) as a set of non-homogeneous linear equations with $M \leq J$ unknowns with coefficients that are functions of the q 's and p 's.

[00284] The general solution of (3.14-36) for each j is of the form

$$u^{(m)} = U^{(m)} + V^{(m)}, \quad m = 1, \dots, M \quad (3.14-37)$$

with $V^{(m)}$ the solution of the homogeneous equation

$$\sum_{m=1}^M V^{(m)} [\phi_j, \phi_m] \approx 0 \quad (3.14-38)$$

The most general solution of (3.14-38) is a linear combination of linearly independent solutions of $V_\alpha^{(m)}$ where $\alpha = 1, \dots, A$ with $A \leq M$. Under the assumption that the matrix

$$\begin{bmatrix} [\phi_1, \phi_1] & \cdots & [\phi_1, \phi_M] \\ \vdots & \ddots & \vdots \\ [\phi_J, \phi_1] & \cdots & [\phi_J, \phi_M] \end{bmatrix} \quad (3.14-39)$$

is of constant rank, the number of independent solutions A is the same for all p, q . Thus the general solution to (3.14-36) can be written as

$$u^{(m)} \approx U^{(m)} + \sum_{\alpha=1}^A v^{(\alpha)} V_\alpha^{(m)}, \quad m = 1, \dots, M \quad (3.14-40)$$

Irreducible and Reducible Cases

[00285] If the equations $\{\phi_j = 0 | j = 1, \dots, J\}$ are not independent, one says that the constraints are reducible. The system is irreducible when the constraints are independent. However the separation of constraints into dependent and independent ones might be difficult to perform. It also may disturb invariance properties under some important symmetry. In some cases it may be impossible to separate irreducible from irreducible contexts. Reducible cases arise for example when the dynamical coordinates include p-form gauge fields.

[00286] Any irreducible set of constraints can always be replaced by a reducible set by introducing constraints of the ones already at hand. The formalism should be invariant under such replacements.

Total Hamiltonian

[00287] We now discuss details of the dynamic equation (3.14-25)

$$\dot{F} \approx \left[F, H' + \sum_{\alpha=1}^A v^{(\alpha)} \phi_\alpha \right] \quad (3.14-41)$$

where from (3.14-40)

$$H' = H + \sum_{m=1}^M U^{(m)} \phi_m$$

and

$$\phi_\alpha = \sum_{m=1}^M V_\alpha^{(m)} \phi_m, \quad \alpha = 1, \dots, A \quad (3.14-42)$$

This is the result of theorem 3 (see below).

[00288] Theorem 3.

$$\left[F, \sum_{m=1}^M U^{(m)} \phi_m \right] \simeq \sum_{m=1}^M U^{(m)} [F, \phi_m] \quad (3.14-43)$$

$$\left[F, \sum_{\alpha=1}^A V_\alpha^{(m)} \phi_m \right] \simeq \sum_{\alpha=1}^A V_\alpha^{(m)} [F, \phi_m] \quad (3.14-44)$$

Proof.

$$\begin{aligned} \left[F, \sum_{m=1}^M U^{(m)} \phi_m \right] &= \sum_{i=1}^N \left\{ \frac{\partial F}{\partial q^{(i)}} \frac{\partial}{\partial p_i} \sum_{m=1}^M U^{(m)} \phi_m - \frac{\partial F}{\partial p_i} \frac{\partial}{\partial q^{(i)}} \sum_{m=1}^M U^{(m)} \phi_m \right\} \\ &= \sum_{i=1}^N \left\{ \frac{\partial F}{\partial q^{(i)}} \left[\sum_{m=1}^M \frac{\partial U^{(m)}}{\partial p_i} \phi_m + \sum_{m=1}^M U^{(m)} \frac{\partial \phi_m}{\partial p_i} \right] \right. \\ &\quad \left. - \sum_{i=1}^N \left\{ \frac{\partial F}{\partial p_i} \left[\sum_{m=1}^M \frac{\partial U^{(m)}}{\partial q^{(i)}} \phi_m + \sum_{m=1}^M U^{(m)} \frac{\partial \phi_m}{\partial q^{(i)}} \right] \right\} \right\} \\ &= \sum_{m=1}^M \{ [F, U^{(m)}] \phi_m + U^{(m)} [F, \phi_m] \} \end{aligned}$$

So

$$\left[F, \sum_{m=1}^M U^{(m)} \phi_m \right] - \sum_{m=1}^M U^{(m)} [F, \phi_m] = \sum_{m=1}^M [F, U^{(m)}] \phi_m \quad (3.14-45)$$

and from (3.14-34) in (3.14-45), (3.14-43) follows. By a similar process we show (3.14-44). We now prove the validity of (3.14-41).

[00289] **Theorem 4.** Let $F(q,p)$ be a regular function, then $F(p,q)$ propagates in time according to the approximate equation (3.14-41).

Proof. From (3.14-25),

$$\frac{dF}{dt} = [F, H] + \sum_{m=1}^M u^{(m)} [F, \phi_m]. \quad (3.14-46)$$

From (3.14-40) into (3.14-46) we obtain,

$$\frac{dF}{dt} \approx [F, H] + \sum_{m=1}^M \left\{ U^{(m)} + \sum_{\alpha=1}^A v^{(\alpha)} V_{\alpha}^{(m)} \right\} [F, \phi_m]$$

or

$$\frac{dF}{dt} \approx [F, H] + \sum_{m=1}^M U^{(m)} [F, \phi_m] + \sum_{m=1}^M \sum_{\alpha=1}^A v^{(\alpha)} V_{\alpha}^{(m)} [F, \phi_m] \quad (3.14-47)$$

Thus from (3.14-43) and (3.14-44) of theorem 3, we get

$$\begin{aligned} \frac{dF}{dt} &\approx [F, H] + \sum_{m=1}^M [F, U^{(m)} \phi_m] + \sum_{\alpha=1}^A v^{(\alpha)} \left[F, \sum_{m=1}^M V_{\alpha}^{(m)} \phi_m \right] \\ &\approx \left[F, H + \sum_{m=1}^M U^{(m)} \phi_m + \sum_{\alpha=1}^A v^{(\alpha)} \sum_{m=1}^M V_{\alpha}^{(m)} \phi_m \right] \\ &\approx \left[F, H' + \sum_{m=1}^M U^{(m)} \phi_m + \sum_{\alpha=1}^A v^{(\alpha)} \phi_{\alpha} \right] \end{aligned} \quad (3.14-48)$$

with

$$H' = H + \sum_{m=1}^M U^{(m)} \phi_m \quad (3.14-49)$$

$$\phi_{\alpha} = \sum_{m=1}^M V_{\alpha}^{(m)} \phi_m \quad (3.14-50)$$

Now define

$$H_T = H' + \sum_{\alpha=1}^A v^{(\alpha)} \phi_{\alpha}. \quad (3.14-51)$$

So we obtain

$$\frac{dF}{dt} \approx [F, H_T] \quad (3.14-52)$$

First and Second Class Functions

[00290] The distinction between primary and secondary constraints is of little importance. We now consider a **fundamental classification**. It depends on the concept of first class and second class functions.

[00291] **Definition 1.** A function $F(q, p)$ is said to be first class if its Poisson bracket with every constraint vanishes weakly, $[F, \phi_j] \approx 0, j = 1, \dots, J$. A function of the canonical variables that is not first class is called second class. Thus F is second class if $[F, \phi_k] \not\approx 0$ for at least one $k, k = 1, \dots, M$.

[00292] **Theorem 5.** If F and G are first class functions, then their Poisson bracket is also a first class function.

[00293] **Proof:** By Hypothesis,

$$[F, \phi_j] = \sum_{k=1}^M f_j^{(k)} \phi_k \quad (3.14-53)$$

$$[G, \phi_j] = \sum_{l=1}^M g_j^{(l)} \phi_l \quad (3.14-54)$$

Applying the Jacobi identity, we get

$$[[F, G], \phi_j] = [F, [G, \phi_j]] - [G, [F, \phi_j]]$$

$$\begin{aligned}
&= \left[F, \sum_{l=1}^M g_j^{(l)} \phi_l \right] - \left[G, \sum_{k=1}^M f_j^{(k)} \phi_k \right] \\
&= \sum_i \left\{ \frac{\partial F}{\partial q^{(i)}} \frac{\partial}{\partial p_i} \sum_{l=1}^M g_j^{(l)} \phi_l - \frac{\partial F}{\partial p_i} \frac{\partial}{\partial q^{(i)}} \sum_{l=1}^M g_j^{(l)} \phi_l \right\} \\
&\quad - \sum_n \left\{ \frac{\partial G}{\partial q^{(n)}} \frac{\partial}{\partial p_n} \sum_{k=1}^M f_j^{(k)} \phi_k - \frac{\partial G}{\partial p_n} \frac{\partial}{\partial q^{(n)}} \sum_{k=1}^M f_j^{(k)} \phi_k \right\} \\
&= \sum_i \left\{ \frac{\partial F}{\partial q^{(i)}} \sum_{l=1}^M \left\{ \frac{\partial g_j^{(l)}}{\partial p_i} \phi_l + g_j^{(l)} \frac{\partial \phi_l}{\partial p_i} \right\} - \frac{\partial F}{\partial p_i} \sum_{l=1}^M \left\{ \frac{\partial g_j^{(l)}}{\partial q^{(i)}} \phi_l + g_j^{(l)} \frac{\partial \phi_l}{\partial q^{(i)}} \right\} \right\} \\
&\quad - \sum_n \left\{ \frac{\partial G}{\partial q^{(n)}} \sum_{k=1}^M \left\{ \frac{\partial f_j^{(k)}}{\partial p_n} \phi_k + f_j^{(k)} \frac{\partial \phi_k}{\partial p_n} \right\} - \frac{\partial G}{\partial p_n} \sum_{k=1}^M \left\{ \frac{\partial f_j^{(k)}}{\partial q^{(n)}} \phi_k + f_j^{(k)} \frac{\partial \phi_k}{\partial q^{(n)}} \right\} \right\} \\
&= \sum_{l=1}^M \left\{ \phi_l \sum_i \left\{ \frac{\partial F}{\partial q^{(i)}} \frac{\partial g_j^{(l)}}{\partial p_i} - \frac{\partial F}{\partial p_i} \frac{\partial g_j^{(l)}}{\partial q^{(i)}} \right\} + g_j^{(l)} \sum_i \left\{ \frac{\partial F}{\partial q^{(i)}} \frac{\partial \phi_l}{\partial q^{(i)}} - \frac{\partial F}{\partial p_i} \frac{\partial \phi_l}{\partial p_i} \right\} \right\} \\
&\quad - \sum_{k=1}^M \left\{ \phi_k \sum_n \left\{ \frac{\partial G}{\partial q^{(n)}} \frac{\partial f_j^{(k)}}{\partial p_n} - \frac{\partial G}{\partial p_n} \frac{\partial f_j^{(k)}}{\partial q^{(n)}} \right\} \right. \\
&\quad \left. + f_j^{(k)} \sum_n \left\{ \frac{\partial G}{\partial q^{(n)}} \frac{\partial \phi_k}{\partial p_n} - \frac{\partial G}{\partial p_n} \frac{\partial \phi_k}{\partial q^{(n)}} \right\} \right\} \\
&= \sum_{l=1}^M \left\{ \phi_l [F, g_j^{(l)}] + g_j^{(l)} [F, \phi_l] \right\} - \sum_{k=1}^M \left\{ \phi_k [G, f_j^{(k)}] + f_j^{(k)} [G, \phi_k] \right\} \\
&= \sum_{l=1}^M [F, g_j^{(l)}] \phi_l - \sum_{k=1}^M [G, f_j^{(k)}] \phi_k + \sum_{l'=1}^M \left\{ \sum_{l=1}^M g_j^{(l)} f_l^{(l')} \right\} \phi_{l'} - \sum_{k'=1}^M \left\{ \sum_{k=1}^M f_j^{(k)} g_k^{(k')} \right\} \phi_{k'} \\
&\approx 0
\end{aligned}$$

We now use theorem 5 to show the following.

[00294] **Theorem 6.** H' defined by (3.14-49) and ϕ_α defined by (3.14-50) are first class functions.

[00295] **Proof:** This follows directly from (3.14-36) and (3.14-38).

[00296] We learn from theorem 6 that the total Hamiltonian defined by (3.14-51) is the sum of the first class Hamiltonian H' and the first class primary constraints ϕ_α multiplied by arbitrary coefficients.

First Class Constraints as Generators of Gauge Transformations

[00297] Gauge transformations are transformations that do not change the physical state.

[00298] The presence of arbitrary functions of time $v^{(\alpha)}$, $\alpha = 1, \dots, A$ in the total Hamiltonian, H_T (see (3.14-51)) imply that not all the q 's and p 's are observable given a set of q 's and p 's where the state of the physical system is uniquely determined. However the converse is not true: there is more than one set of values of the canonical variables that defines a state. To illustrate this, we see that if we give an initial set of values of physical state at time t , we expect the equations of motion to fully determine the state at other times. Thus any ambiguity in the value of the canonical variables at $t_2 \neq t_1$ should be irrelevant from the physical point of view.

A Derivation Example

[00299] We propose here an alternate formulation of Dirac's formalism.

Primary Constraints

[00300] Recall that the momenta, canonically conjugate to the generalized "coordinates" $q^{(j)}$, $j = 1, \dots, N$ is given by

$$p_j = \frac{\partial L(q, \dot{q})}{\partial \dot{q}^{(j)}}, \quad j = 1, \dots, N. \quad (E - 1)$$

For non-singular systems the equations allows us to express $\dot{q}^{(j)}$, $j = 1, \dots, N$ in terms of the canonical variables,

$$\dot{q}^{(i)} = f_i(q, p), \quad i = 1, \dots, N \quad (E - 2)$$

By performing a Legendre transformation

$$H_c(p, q) = \sum_{i=1}^N p_i f_i(q, p) + L(q, f(p, q))$$

We obtain the Hamiltonian of the system H_c . And from this function we obtain the standard equations of motion of the system.

$$\begin{aligned}\dot{q} &= \frac{\partial H_c}{\partial p} \\ \dot{p} &= -\frac{\partial H_c}{\partial q}\end{aligned}\quad (E -- 3)$$

For (E -- 2) to be well-defined we need to have the Hessian W of satisfy

$$\det W \neq 0 \quad (E -- 4)$$

In this case the accelerations $\ddot{q}^{(i)}$ are uniquely determined by the $q^{(i)}$ and $\dot{q}^{(i)}$.

[00301] When $\det W \neq 0$, the Hamiltonian equations of motion do not take the standard form, and we speak of a singular Lagrangian. For illustration purposes, consider a Lagrangian of the form

$$L(q, \dot{q}) = \frac{1}{2} \sum_{i=1}^N \sum_{j=1}^N W_{ij}(q) \dot{q}^{(i)} \dot{q}^{(j)} + \sum_{i=1}^N \eta_i(q) \dot{q}^{(i)} - V(q) \quad (E -- 5)$$

with W a symmetric matrix. From (E -- 1), the canonical momentum for (E -- 5) is given by

$$p_i = \frac{1}{2} \sum_{j=1}^N W_{ij}(q) \dot{q}^{(j)} + \eta_i(q), \quad i = 1, \dots, n. \quad (E -- 6)$$

If W is singular of rank R_W , then it possesses $N - R_W$ eigenvectors with corresponding zero eigenvalues. Then for eigenvectors $v_j^{(\alpha)}$

$$\sum_{j=1}^N W_{ij}(q) v_j^{(\alpha)}(q) = 0, \quad \alpha = 1, \dots, N - R_W$$

So pre-multiplying (E -- 6) by $v_j^{(\alpha)}$ and summing over i we get

$$\begin{aligned} \sum_{i=1}^N v_i^{(\alpha)}(q) p_i &= \sum_{i=1}^N \left[\sum_{j=1}^N \left(v_i^{(\alpha)}(q) W_{ij}(q) \dot{q}^{(j)} \right) + v_i^{(\alpha)}(q) \eta_i(q) \right] \\ &= \sum_{i=1}^N v_i^{(\alpha)}(q) \eta_i(q), \quad \alpha = 1, \dots, N - R_W \end{aligned}$$

So

$$\sum_{i=1}^N v_i^{(\alpha)}(q) (p_i - \eta_i(q)) = 0, \quad \alpha = 1, \dots, N - R_W. \quad (E - 7)$$

Let $\{p_\alpha\}$, $\alpha = 1, \dots, N - R_W$, denote the linearly dependent elements of p . Let $\{p_\alpha\}$, $a = 1, \dots, R_a$ be the momenta satisfying (E -- 1). Then the constraint equations are of the form

$$\sum_{\beta=1}^{N-R_W} M_{\alpha\beta}(q) p_\beta - F_\alpha(q, \{p_a\}) = 0, \quad \alpha = 1, \dots, N - R_W \quad (E - 8)$$

$$M_{\alpha\beta}(q) = v_\beta^{(\alpha)}$$

and

$$F_\alpha(q, \{p_\beta\}) = \sum_{i=1}^N v_i^{(\alpha)}(q) \eta_i(q) + \sum_{b=1}^{R_W} v_b^{(\alpha)}(q) p_b \quad (E - 9)$$

The matrix $\{M_{\alpha\beta}\}$ is necessarily invertible because otherwise M would possess eigenvectors with zero eigenvalues, implying existence of additional constraints.

Note that (E -- 8) can be written as

$$p_\alpha - g_\alpha(q, \{p_a\}) = 0, \quad \alpha = 1, \dots, N - R_W$$

with

$$g_\alpha(q, \{p_a\}) = \sum_{\beta=1}^{N-R_W} M_{\alpha\beta}^{-1} F_\beta(q, \{p_a\}) \quad (E - 10)$$

with $\dim\{p_a\} = R_W$. So we can define,

$$\phi_\alpha(q, p) = p_\alpha - g_\alpha(q, \{p_a\}) = 0, \quad \alpha = 1, \dots, N - R_W$$

(E -- 11)

In Dirac's terminology, constraints of the form of (E -- 11) are referred to as primary constraints. Although the derivation above is based on a Lagrangian, quadratic in the velocity terms, it is generally valid for Lagrangians which depend on q and $\dot{q}$ but not on higher derivatives.

[00302] **Note:** Primary constraints follow exclusively from the definition of canonical momenta.

[00303] The derivation above is valid for general Lagrangians and their Hessian. Let's assume $\{W_{ij}(q, \dot{q})\}$ is the Hessian of a given Lagrangian L . Let $\{W_{ab}|a, b = 1, \dots, R_W\}$ be the largest sub-matrix of $\{W_{ij}\}$ with suitable rearrangement if necessary. We then solve (E -- 1) for R_W velocities $\dot{q}^{(a)}$ in terms of $\{q^{(i)}|i = 1, \dots, n\}$, $\{p_a|a = 1, \dots, R_W\}$ and $\{q^{(\alpha)}|\alpha = 1, \dots, N - R_W\}$. That is

$$\dot{q}^{(a)} = f_a(q, \{p_b\}, \{\dot{q}^{(\beta)}\}) \quad (E -- 12)$$

with $a, b = 1, \dots, R_W$ and $\beta = R_W + 1, \dots, N$.

[00304] Inserting these relations into (E -- 1), we get relations of the form

$$p_j = h_j(q, \{p_a\}, \{\dot{q}^{(\alpha)}\}) \quad (E -- 13)$$

with $a, j = 1, \dots, R_W$ and $\alpha = R_W + 1, \dots, N$. This relation reduces to an identity by construction. The remaining equations are of the form

$$p_\alpha = h_\alpha(q, \{p_a\}, \{\dot{q}^{(\beta)}\}) \quad (E -- 14)$$

with $\alpha = 1, \dots, N - R_W$. However, the right hand side cannot depend on $\{\dot{q}^{(\beta)}\}$ since otherwise we could express more velocities in terms of the momenta of the coordinates of the momenta and the remaining velocities.

Hamiltonian Equations of Motion for Constrained Systems

[00305] **Theorem 3.16.1.** In the space Γ_p define by $\Gamma_p = \{\phi_\alpha(p, q) | \alpha = 1, \dots, N - R_W\}$ where ϕ_α is defined as (E -- 11). The Hamiltonian is only a function of $\{q^{(i)} | i = 1, \dots, N\}$ and momenta $\{p_a | a = 1, \dots, R_W\}$ and does not depend on $\{\dot{q}^{(\alpha)} | \alpha = 1, \dots, N - R_W\}$

[00306] **Proof.** On Γ_p the Hamiltonian is given by

$$H_o = H_c|_{\Gamma_p} = \sum_{a=1}^{R_W} p_a f_a - \sum_{\alpha=1}^{N-R_W} g_\alpha \dot{q}^{(\alpha)} - L(q, \{f_b\}, \{\dot{q}^{(\beta)}\}) \quad (E -- 15)$$

where f_a , $a = 1, \dots, N - R_W$ is given by (E -- 12) and g_α , $\alpha = 1, \dots, R_W$ is given by (E -- 10). We want to show that H_o does not depend on $\dot{q}^{(\beta)}$, $\beta = 1, \dots, N - R_W$. We compute

$$\begin{aligned} \frac{\partial H_o}{\partial \dot{q}^{(\beta)}} &= \sum_{a=1}^{R_W} p_a \frac{\partial f_a}{\partial \dot{q}^{(\beta)}} - g_\beta - \sum_{a=1}^{R_W} \frac{\partial L}{\partial \dot{q}^{(a)}} \Big|_{\dot{q}^{(a)}=f_a} \frac{\partial f_a}{\partial \dot{q}^{(\beta)}} - \frac{\partial L}{\partial \dot{q}^{(\beta)}} \Big|_{\dot{q}^{(a)}=f_a} \\ &= \sum_{a=1}^{R_W} \left(p_a - \frac{\partial L}{\partial \dot{q}^{(a)}} \Big|_{\dot{q}^{(a)}=f_a} \right) \frac{\partial f_a}{\partial \dot{q}^{(\beta)}} - g_\beta - \frac{\partial L}{\partial \dot{q}^{(\beta)}} \Big|_{\dot{q}^{(a)}=f_a} \end{aligned} \quad (E -- 16)$$

Since by definition

$$p_a = \frac{\partial L}{\partial \dot{q}^{(a)}}, \quad a = 1, \dots, R_W$$

And from (E -- 11)

$$g_\beta = p_\beta = \frac{\partial L}{\partial \dot{q}^{(\beta)}} \Big|_{\dot{q}^{(a)}=f_a}.$$

So

$$\frac{\partial H_o}{\partial \dot{q}^{(\beta)}} = 0, \quad \beta = 1, \dots, N - R_W.$$

(E -- 17)

and therefore

$$H_o(q, \{p_a\}, \{\dot{q}^{(a)}\}) = H_o(q, \{p_a\}).$$

[00307] **Theorem 3.16.2.** In the presence of primary constraints (E -- 11), the Hamilton equations of motion are given by

$$\begin{aligned} \dot{q}^{(i)} &= \frac{\partial H_o}{\partial p_i} + \sum_{\beta=1}^N \dot{q}^{(\beta)} \frac{\partial \phi_\beta}{\partial p_i}, \quad i = 1, \dots, N \\ \dot{p}_i &= -\frac{\partial H_o}{\partial q^{(i)}} + \sum_{\beta=1}^n \dot{q}^{(\beta)} \frac{\partial \phi_\beta}{\partial q^{(i)}}, \quad i = 1, \dots, N \\ \phi_\alpha(p, q) &= 0, \quad \alpha = 1, \dots, N - R_W \end{aligned} \quad (E -- 18)$$

where $\dot{q}^{(\beta)}$ are a priori underdetermined velocities.

[00308] **Proof:** From (E -- 15) we obtain and the application of **Theorem 3.16.1**

$$\begin{aligned} \frac{\partial H_o}{\partial p_a} &= f_a + \sum_{b=1}^{R_W} p_b \frac{\partial f_b}{\partial p_a} + \sum_{\beta=1}^{N-R_W} \frac{\partial g_\beta}{\partial p_a} \dot{q}^{(\beta)} - \sum_{b=1}^{R_W} \frac{\partial L}{\partial \dot{q}^{(b)}} \frac{\partial f_b}{\partial p_a} \\ &= \dot{q}^{(a)} + \sum_{b=1}^{R_W} \left(p_b - \frac{\partial L}{\partial \dot{q}^{(b)}} \right) \frac{\partial f_b}{\partial p_a} + \sum_{\beta=1}^{N-R_W} \frac{\partial g_\beta}{\partial p_a} \dot{q}^{(\beta)} \\ &= \dot{q}^{(a)} + \sum_{\beta=1}^{N-R_W} \frac{\partial g_\beta}{\partial p_a} \dot{q}^{(\beta)} \end{aligned} \quad (E -- 19)$$

with $a = 1, \dots, n - R_W$. Further

$$\begin{aligned} \frac{\partial H_o}{\partial q^{(i)}} &= \sum_{b=1}^{R_W} p_b \frac{\partial f_b}{\partial q^{(i)}} + \sum_{\beta=1}^{N-R_W} \dot{q}^{(\beta)} \frac{\partial g_\beta}{\partial q^{(i)}} - \frac{\partial L}{\partial q^{(i)}} \Big|_{\dot{q}^{(a)}=f_a} - \sum_{b=1}^{R_W} \frac{\partial L}{\partial \dot{q}_b} \Big|_{\dot{q}_b=f_b} \frac{\partial f_b}{\partial q^{(i)}} \\ &= \sum_{b=1}^{R_W} \left(p_b - \frac{\partial L}{\partial \dot{q}^{(b)}} \Big|_{\dot{q}^{(b)}=f_b} \right) \frac{\partial f_b}{\partial q^{(i)}} + \sum_{\beta=1}^{N-R_W} \dot{q}^{(\beta)} \frac{\partial g_\beta}{\partial q^{(i)}} - \frac{\partial L}{\partial q^{(i)}} \Big|_{\dot{q}^{(a)}=f_a} \end{aligned}$$

$$\begin{aligned}
&= \sum_{\beta=1}^{N-R_W} \dot{q}^{(\beta)} \frac{\partial g_{\beta}}{\partial q^{(i)}} - \frac{\partial L}{\partial q^{(i)}} \Big|_{\dot{q}^{(a)}=f_a} \\
&= \sum_{\beta=1}^{N-R_W} \dot{q}^{(\beta)} \frac{\partial g_{\beta}}{\partial q^{(i)}} - \frac{d}{dt} \left(\frac{\partial L}{\partial \dot{q}^{(i)}} \right) \Big|_{\dot{q}^{(a)}=f_a}
\end{aligned} \tag{E -- 20}$$

from (add reference).

$$\frac{\partial H_o}{\partial q^{(i)}} = -\dot{p}_i + \sum_{\beta=1}^{N-R_W} \dot{q}^{(\beta)} \frac{\partial g_{\beta}}{\partial q^{(i)}} \tag{E -- 21}$$

From (E -- 19) and (E -- 20) we get:

$$\begin{aligned}
\dot{q}^{(a)} &= \frac{\partial H_o}{\partial p_a} - \sum_{\beta=1}^{N-R_W} \frac{\partial g_{\beta}}{\partial p_a} \dot{q}^{(\beta)}, \quad a = 1, \dots, R_W \\
\dot{p}_i &= -\frac{\partial H_o}{\partial q^{(i)}} + \sum_{\beta=1}^{N-R_W} \dot{q}^{(\beta)} \frac{\partial g_{\beta}}{\partial q^{(i)}}, \quad i = 1, \dots, N
\end{aligned} \tag{E -- 22}$$

Since $\frac{\partial H_o}{\partial p_{\alpha}} = 0$ and $\frac{\partial \phi_{\beta}}{\partial p_{\alpha}} = \delta_{\beta\alpha}$ we can supplement these equations with

$$\dot{q}^{(\alpha)} = \frac{\partial H_o}{\partial p_{\alpha}} - \sum_{\beta=1}^{N-R_W} \frac{\partial g_{\beta}}{\partial p_{\alpha}} \dot{q}^{(\beta)}, \quad \alpha = 1, \dots, N - R_W \tag{E -- 23}$$

So we can write

$$\begin{aligned}
\dot{q}^{(i)} &= \frac{\partial H_o}{\partial p_i} + \sum_{\beta=1}^{N-R_W} \frac{\partial g_{\beta}}{\partial p_i} \dot{q}^{(\beta)}, \quad i = 1, \dots, N \\
\dot{p}_i &= -\frac{\partial H_o}{\partial q^{(i)}} - \sum_{\beta=1}^{N-R_W} \dot{q}^{(\beta)} \frac{\partial g_{\beta}}{\partial q^{(i)}}, \quad i = 1, \dots, N
\end{aligned} \tag{E -- 24}$$

For consistency with (E -- 11) we should write

$$\dot{q}^{(\alpha)} = \frac{d}{dt} - g_{\alpha}(q, \{p_a\}), \quad \alpha = 1, \dots, N - R_W \quad (E -- 25)$$

where $\dot{p}_{\alpha}$ is given by the right hand side of (E -- 22).

Streamlining the Hamiltonian equation of motion (EOM)

[00309] **Definition 3.16-1.** A function f is weakly equal to g denoted by $f \approx g$, if f and g are equal on the subspace defined by the primary constraints,

$$\phi_{\beta} = 0 \text{ when } f|_{\Gamma_p} = g|_{\Gamma_p}$$

and

$$f(q, p) \approx g(q, p) \Leftrightarrow f(q, p) = g(q, p) \text{ when } \{\phi_{\alpha}(q, p) = 0\}$$

[00310] **Theorem 3.16.3.** Assume f, g are defined over the entire space spanned by $\{q^{(i)}\}, \{p_i\}$. Then if

$$f(q, p)|_{\Gamma_p} = g(q, p)|_{\Gamma_p} \quad (E -- 26)$$

Then

$$\frac{\partial}{\partial q^{(i)}} \left(f - \sum_{\beta} \phi_{\beta} \frac{\partial f}{\partial p_{\beta}} \right) \simeq \frac{\partial}{\partial q^{(i)}} \left(h - \sum_{\beta} \phi_{\beta} \frac{\partial h}{\partial p_{\beta}} \right)$$

and

$$\frac{\partial}{\partial p_i} \left(f - \sum_{\beta} \phi_{\beta} \frac{\partial f}{\partial p_{\beta}} \right) \simeq \frac{\partial}{\partial p_i} \left(h - \sum_{\beta} \phi_{\beta} \frac{\partial h}{\partial p_{\beta}} \right) \quad (E -- 27)$$

for $i = 1, \dots, N$.

[00311] **Proof:** Consider the two functions $f(q, \{p_a\}, \{p_{\beta}\})$ and $h(q, \{p_a\}, \{p_{\beta}\})$. Using (E -- 11) and from the hypothesis of the theorem,

$$f(q, \{p_a\}, \{g_{\alpha}\}) = h(q, \{p_a\}, \{g_{\alpha}\})$$

(E -- 28)

Thus it follows

$$\left(\frac{\partial f}{\partial q^{(i)}} + \sum_a \frac{\partial f}{\partial p_a} \frac{\partial p_a}{\partial q^{(i)}} + \sum_\beta \frac{\partial f}{\partial p_\beta} \frac{\partial g_\beta}{\partial q^{(i)}} \right)_{\Gamma_p} = \left(\frac{\partial h}{\partial q^{(i)}} + \sum_a \frac{\partial h}{\partial p_a} \frac{\partial p_a}{\partial q^{(i)}} + \sum_\beta \frac{\partial h}{\partial p_\beta} \frac{\partial g_\beta}{\partial q^{(i)}} \right)_{\Gamma_p} \quad (E -- 29)$$

and

$$\left(\frac{\partial f}{\partial p_i} + \sum_{a \neq i} \frac{\partial f}{\partial p_a} \frac{\partial p_a}{\partial p_i} + \sum_\beta \frac{\partial f}{\partial p_\beta} \frac{\partial g_\beta}{\partial p_i} \right)_{\Gamma_\beta} = \left(\frac{\partial h}{\partial p_i} + \sum_{a \neq i} \frac{\partial h}{\partial p_a} \frac{\partial p_a}{\partial p_i} + \sum_\beta \frac{\partial h}{\partial p_\beta} \frac{\partial g_\beta}{\partial p_i} \right)_{\Gamma_\beta} \quad (E -- 30)$$

Note since $\phi_\alpha(q, p) = p_\alpha - g_\alpha(q, \{p_a\})$, we have

$$\frac{\partial g_\beta}{\partial q^{(i)}} = - \frac{\partial \phi_\beta(q, p)}{\partial q^{(i)}}$$

and

$$\frac{\partial g_\beta}{\partial p_i} = - \frac{\partial \phi_\beta(q, p)}{\partial p_i}$$

and

$$\partial \phi_\alpha(q, p) = 0$$

for $\alpha = 1, \dots, N - R_W$. We have

$$\left(\frac{\partial f}{\partial q^{(i)}} - \sum_\beta \frac{\partial f}{\partial p_\beta} \frac{\partial \phi_\beta}{\partial q^{(i)}} \right)_{\Gamma_p} = \left(\frac{\partial h}{\partial q^{(i)}} - \sum_\beta \frac{\partial h}{\partial p_\beta} \frac{\partial \phi_\beta}{\partial q^{(i)}} \right)_{\Gamma_\beta}$$

which can be written as

$$\frac{\partial}{\partial q^{(i)}} \left(f - \sum_{\beta} \phi_{\beta} \frac{\partial f}{\partial p_{\beta}} \right) \simeq \frac{\partial}{\partial q^{(i)}} \left(h - \sum_{\beta} \phi_{\beta} \frac{\partial h}{\partial p_{\beta}} \right)$$

since $\phi_{\beta} \frac{\partial^2 f}{\partial p_{\beta}^2} = 0$ because $\phi_{\beta} = 0$. Similarly,

$$\frac{\partial}{\partial p_i} \left(f - \sum_{\beta} \phi_{\beta} \frac{\partial f}{\partial p_{\beta}} \right) \simeq \frac{\partial}{\partial p_i} \left(h - \sum_{\beta} \phi_{\beta} \frac{\partial h}{\partial p_{\beta}} \right)$$

[00312] Corrolary 3.16-1.

$$\begin{aligned} \dot{q}^{(i)} &= \frac{\partial H}{\partial p_i} + \sum_{\beta} v^{(\beta)} \frac{\partial \phi_{\beta}}{\partial p_i} \\ \dot{p}_i &= -\frac{\partial H}{\partial q^{(i)}} - \sum_{\beta} v^{(\beta)} \frac{\partial \phi_{\beta}}{\partial q^{(i)}} \end{aligned}$$

for $i = 1, \dots, N$.

[00313] **Proof.** We consider two Hamiltonians $H(\{q^{(i)}\}, \{p_i\})$ and $H_o(\{q^{(i)}\}, \{p_a\})$. Define $H(\{q^{(i)}\}, \{p_i\})$ as follows

$$H(\{q^{(i)}\}, \{p_i\}) \approx H_o(\{q^{(i)}\}, \{p_a\}).$$

Then using the result of Theorem 3.16.1, from (E -- 29) with $f = H$ and $h = H_o$

$$\frac{\partial H_o}{\partial q^{(i)}} \approx \frac{\partial}{\partial q^{(i)}} \left(H - \sum_{\beta=1}^{N-R_w} \phi_{\beta} \frac{\partial H}{\partial p_{\beta}} \right) \quad (\text{E -- 31})$$

$$\frac{\partial H_o}{\partial p_i} \approx \frac{\partial}{\partial p_i} \left(H - \sum_{\beta=1}^{N-R_w} \phi_{\beta} \frac{\partial H}{\partial p_{\beta}} \right) \quad (\text{E -- 32})$$

Using (E -- 31) and (E -- 32) in (E -- 24), we get

$$\dot{q}^{(i)} \approx \frac{\partial}{\partial p_i} \left(H - \sum_{\beta} \phi_{\beta} \frac{\partial H}{\partial p_{\beta}} \right) + \sum_{\beta} \dot{q}^{(\beta)} \frac{\partial \phi_{\beta}}{\partial p_i}$$

and

$$\dot{p}_i \approx -\frac{\partial}{\partial q^{(i)}} \left(H - \sum_{\beta} \phi_{\beta} \frac{\partial H}{\partial p_{\beta}} \right) - \sum_{\beta} \dot{q}^{(\beta)} \frac{\partial \phi_{\beta}}{\partial q^{(i)}}$$

or

$$\dot{q}^{(i)} \approx \frac{\partial}{\partial p_i} \left(H - \sum_{\beta} \phi_{\beta} \left(\frac{\partial H}{\partial p_{\beta}} - \dot{q}^{(\beta)} \right) \right)$$

and

$$\dot{p}_i \approx -\frac{\partial}{\partial q^{(i)}} \left(H - \sum_{\beta} \phi_{\beta} \left(\frac{\partial H}{\partial p_{\beta}} - \dot{q}^{(\beta)} \right) \right) \quad (E -- 33)$$

Define

$$v_{\beta} \equiv \dot{q}^{(\beta)} - \frac{\partial H}{\partial p_{\beta}}$$

$$H_T \equiv H + \sum_{\beta} v^{(\beta)} \phi_{\beta}$$

So (E -- 33) becomes

$$\dot{q}^{(i)} \approx \frac{\partial H_T}{\partial p_i}$$

$$\dot{p}_i \approx -\frac{\partial H_T}{\partial q^{(i)}} \quad (E -- 34)$$

Constrained Hamiltonian Systems

[00314] Local symmetries on a Lagrangian based model. Consider

$$q^{(i)} \rightarrow q^{(i)}(t) + \delta q^{(i)}(t)$$

$$\dot{q}^{(i)} \rightarrow \dot{q}^{(i)}(t) + \delta \dot{q}^{(i)}(t)$$

with $i = 1, \dots, N$. The action of the system is given by

$$S(q, \dot{q}) = \int L(q, \dot{q}) dt$$

where q and $\dot{q}$ are n -dimensional column vectors. The action differential

$$\begin{aligned} \delta S &= \int L(q + \delta q, \dot{q} + \delta \dot{q}) dt - \int L(q, \dot{q}) dt \\ &= \int L(q + \delta q, \dot{q} + \delta \dot{q}) dt - \int L(q, \dot{q}) dt \\ &= \int \left[\sum_i \frac{\partial L}{\partial q^{(i)}} \delta q^{(i)} + \sum_i \frac{\partial L}{\partial \dot{q}^{(i)}} \delta \dot{q}^{(i)} \right] dt \\ &= - \int \sum_i \left[\frac{d}{dt} \frac{\partial L}{\partial \dot{q}^{(i)}} - \frac{\partial L}{\partial q^{(i)}} \right] \delta q^{(i)} dt \\ &= - \sum_i dt \sum_i E_i^{(o)}(q, \dot{q}, \ddot{q}) \delta q^{(i)} \end{aligned}$$

where we define the Euler-Lagrange differential operator

$$E_i^{(o)}(q, \dot{q}, \ddot{q}) = \frac{d}{dt} \frac{\partial L}{\partial \dot{q}^{(i)}} - \frac{\partial L}{\partial q^{(i)}}.$$

Note that

$$\int \sum_{i=1}^N E_i^{(o)}(q, \dot{q}, \ddot{q}) \delta q^{(i)} dt \equiv 0 \quad (3.17-1)$$

on shell. Expanding $E_i^{(o)}$

$$\begin{aligned} E_i^{(o)}(q, \dot{q}, \ddot{q}) &= \sum_j \left[\frac{\partial^2 L(q, \dot{q})}{\partial \dot{q}^{(i)} \partial \ddot{q}^{(j)}} \ddot{q}^{(j)} + \frac{\partial^2 L(q, \dot{q})}{\partial \dot{q}^{(i)} \partial q^{(j)}} \dot{q}^{(j)} \right] - \frac{\partial L(q, \dot{q})}{\partial q^{(i)}} \\ &= \sum_j W_{ij}(q, \dot{q}) \ddot{q}^{(j)} + \sum_j \frac{\partial^2 L(q, \dot{q})}{\partial \dot{q}^{(i)} \partial q^{(j)}} \dot{q}^{(j)} - \frac{\partial L(q, \dot{q})}{\partial q^{(i)}} \\ &= \sum_j W_{ij}(q, \dot{q}) \ddot{q}^{(j)} + k_i(q, \dot{q}) \end{aligned}$$

If L is singular, $W_{(N \times N)}$ is not invertible so (3.17-1) cannot be solved for $\ddot{q}_i$, $i = 1, \dots, N$. If $\text{Rank}(W(q, \dot{q})) = R_W$ on shell, then there exist $N - R_W$ in the theory.

There exist $N - R_W$ independent left (or right) zero mode eigenvectors $w_i^{(o,k)}$, $i = 1, \dots, N - R_W$ such that

$$\sum_i w_i^{(o,k)}(q, \dot{q}) W_{ij}(q, \dot{q}) = 0, \quad k = 1, \dots, N - R_W \quad (3.17-2)$$

Thus

$$\phi^{(o,k)} = \sum_{i=1}^N w_i^{(o,k)}(q, \dot{q}) E_i^{(o)}(q, \dot{q}, \ddot{q})$$

depend on q and $\dot{q}$ only. The $\phi^{(o,k)}$ also vanish on shell:

$$\phi^{(o,k)}(q, \dot{q}) = 0, \quad k = 1, \dots, N - R_W$$

[00315] The set $\{\phi^{(o,k)} | k = 1, \dots, N - R_W\}$ are the zero generation constraints. It is possible that not all the $\{\phi^{(o,k)}\}$ are linearly independent. So we may find linear combinations of the zero mode eigenvectors

$$v_i^{(o,n_o)} = \sum_k c_k^{(n_o)} w_i^{(o,k)}$$

such that we have

$$G^{(o,n_o)} = v^{(o,n_o)} E^{(o)} \equiv 0, \quad n_o, \dots, N_o \quad (3.17-3)$$

These are called gauge identities.

[00316] Any variation δq_i , $i = 1, \dots, N$, of the form

$$\delta q_i = \sum_{n_o} \varepsilon_{n_o} v_i^{(o,n_o)}$$

Is action invariant by (3.17-1).

Given this definition of δq_i and (3.17-3), we conclude

$$\begin{aligned}
\delta S &= \int dt \sum_{i=1}^N E_i^{(o)}(q, \dot{q}, \ddot{q}) \sum_{n_o} \varepsilon_{n_o}(t) v_i^{(o, n_o)} \\
&= \int dt \sum_{i=1}^N \varepsilon_{n_o} \sum_{n_o} E_i^{(o)}(q, \dot{q}, \ddot{q}) v_i^{(o, n_o)}(q, \dot{q}) \\
&= \int dt \sum_{i=1}^N \varepsilon_{n_o} G^{(o, n_o)} \\
&\equiv 0
\end{aligned}$$

everywhere. The remaining zero generating modes which we denote by $u^{(o, n_o)}$ lead to genuine constraints. They are of the form $\phi^{(o, n_o)}(q, \dot{q}) = 0$ on shell, where

$$\phi^{(o, n_o)} = u^{(o, n_o)} E^{(o)}. \quad (3.17-4)$$

[00317] The algorithm now proceeds as follows. We separate the gauge identities (3.17-3) from the nontrivial constraints (3.17-4) and will list them separately. They will be used for determining local symmetry transformations.

[00318] Next we want to search for additional constraints. We do this by searching for further functions of the coordinates and velocities which vanish in the space of physical trajectories. To this effect consider the following $N + N_o$ vector constructed from $E^{(o)}$ and the time derivative of the constraints (3.17-4)

$$[E^{(1)}] = \begin{bmatrix} E^{(o)} \\ \frac{d}{dt}(u^{(o, 1)} E^{(o)}) \\ \vdots \\ \frac{d}{dt}(u^{(o, n_o)} E^{(o)}) \end{bmatrix} = \begin{bmatrix} E^{(o)} \\ \frac{d}{dt} \phi^{(o)} \end{bmatrix} \quad (3.17-5)$$

by construction. The constraint $\phi^{(o)}$ is valid for all time and therefore $\frac{d}{dt} \phi^{(o)} = 0$ on shell, but

$$\frac{d\phi^{(o,i)}}{dt} = \nabla \dot{q}(u^{(o,i)}E^{(o)})\ddot{q} + \nabla q(u^{(o,i)}E^{(o)})\dot{q} \quad (3.17-6)$$

So

$$[E_{i_1}^{(1)}] = \sum_{j=1}^n W_{i_1 j}^{(1)}(q, \dot{q}) \dot{q}^{(j)} + k_{i_1}^{(1)}(q, \dot{q})$$

where $i_1 = 1, \dots, N + N_o$, and

$$[W_{i_1 i}^{(1)}] = \begin{bmatrix} W^{(o)} \\ \nabla \dot{q}(u^{(o,i)}E^{(o)}) \\ \vdots \\ \nabla \dot{q}(u^{(o,N_o)}E^{(o)}) \end{bmatrix}$$

$$[k_{i_1}^{(1)}] = \begin{bmatrix} k^{(o)} \\ \sum_j \frac{\partial}{\partial q^{(j)}} (u^{(o,i)}E^{(o)}) \dot{q}^{(j)} \\ \vdots \\ \sum_j \frac{\partial}{\partial q^{(j)}} (u^{(o,N_o)}E^{(o)}) \dot{q}^{(j)} \end{bmatrix} \quad (3.17-7)$$

We next look for the zero modes of $W^{(1)}$. By construction, these zero modes include the o modes of the previous level. The gauge identities at level 1 are.

$$G^{(1,n_1)} = v^{(1,n_1)} E^1 - \sum_{n_o=1}^{N_o} M_{n_1 n_o}^{(1,0)} (u^{(o,n_o)} E^{(o)}) \equiv 0 \quad (3.17-8)$$

where $n_1 = 1, \dots, N_1$ and the genuine constraints are of the form

$$\phi^{(1,n_1)} = \phi^{(1,n_1)} E^1 = 0 \quad (3.17-9)$$

with $n_1 = 1, \dots, N_1$ on shell.

[00319] We next adjoin the new identities (3.17-8) to the ones determined earlier (3.17-3) with the remaining constraints (3.17-9) we proceed as before, adjoining their time derivatives to (3.17-5) and construct $W_{i_1 i}^{(1)}$ and $k_{i_1}^{(1)}$.

[00320] The iterative process will terminate at some level M if either i) there is not further zero modes, or ii) the new constraints can be expressed as linear combinations of previous constraints.

The maximal set of linearly independent gauge identities generated by the algorithm

[00321] Note that the algorithm steps are of the form

$$G^{(0,n_0)} = u^{(0,n_0)} E^{(0)} \equiv 0 \quad (3.17-10)$$

$$G^{(l,n_l)} = u^{(l,n_l)} E^{(l)} - \sum_{l'=0}^{l-1} \sum_{n_{l'}=0}^{N_{l'}} M_{n_l n_{l'}}^{(l,l')} \phi^{(l',n_{l'})} \quad (3.17-11)$$

with $L = 1, \dots, N_L$. The $M_{n_l n_{l'}}^{(l,l')}$ are only functions of q and $\dot{q}$. And

$$\phi^{(l,n_l)} = u^{(l,n_l)} E^{(l)}, \quad n_l = 1, \dots, N_l, \quad (3.17-12)$$

$$E^{(l)} = \begin{bmatrix} E^{(0)} \\ \frac{d\phi^{(0)}}{dt} \\ \vdots \\ \frac{d\phi^{(l-1)}}{dt} \end{bmatrix} \quad (3.17-13)$$

where $\phi^{(l)}$ is a column vector with N_l components $\phi^{(l,n_l)}$. Thus we conclude from (3.17-13) and (3.17-11) that the general form of the gauge identity given by (3.17-11) is of the form

$$G^{(l,n_l)} = \sum_{i=1}^{N_l} \sum_{l=1}^M \sum_{m=1}^l \zeta_{mi}^{(l,m_l)} \frac{d^m}{dt^m} E_i^{(o)} \equiv 0 \quad (3.17-14)$$

where $\zeta_{mi}^{(l,m_l)}(q, \dot{q})$ and $N_l < M$. From (3.17-14) it also follows that

$$\sum_{l=1}^M \sum_{n_l=1}^l \varepsilon^{(l,n_l)} G^{(l,n_l)} \equiv 0 \quad (3.17-15)$$

This identity can also be written as

$$\sum \delta q^{(i)} E_i^{(o)} - \frac{d}{dt} F$$

where

$$\delta q^{(i)} = \sum_{l=1}^M \sum_{n_l=1}^{N_l} \sum_{m=q}^l (-1)^m \frac{d^m}{dt^m} \zeta_m^{(l,n_l)} \varepsilon^{(l,m_l)}(t) \quad (3.17-16)$$

and F is a complicated function of q and $\dot{q}$. By collecting indices l, n_l together

$$\delta q_i = \sum_{l=1}^M \sum_{n_l=1}^{N_l} \sum_{m=q}^l (-1)^m \zeta_{mi}^{(a)} \varepsilon^{(a)}(t)$$

Example of constrained Hamiltonian system in Lagrangian form

[00322]

Let

$$L(q, \dot{q}) = \frac{1}{2} \dot{q}^{2(1)} + \dot{q}^{(1)} q^{(2)} + \frac{1}{2} (q^{(1)} - q^{(2)})^2 \quad (3.17-17)$$

$$E^{(o)} = \begin{bmatrix} \frac{d}{dt} \frac{\partial}{\partial \dot{q}^{(1)}} - \frac{\partial L}{\partial q^{(1)}} \\ \frac{d}{dt} \frac{\partial}{\partial \dot{q}^{(2)}} - \frac{\partial L}{\partial q^{(2)}} \end{bmatrix} = \begin{bmatrix} \ddot{q}^{(1)} + 2q^{(2)} - q^{(1)} \\ q^{(1)} - q^{(2)} \end{bmatrix} \quad (3.17-18)$$

$$W = \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix} \quad (3.17-19)$$

$$k = \begin{bmatrix} \dot{q}^{(2)} - q^{(1)} + q^{(2)} \\ -\dot{q}^{(1)} - q^{(2)} + q^{(1)} \end{bmatrix} \quad (3.17-20)$$

The only o mode is

$$u^{(o)} = [0,1]$$

Then

$$\begin{aligned} E^{(o)} &= W^{(o)}\ddot{q} + k^{(o)} \\ &= \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} \ddot{q}^{(1)} \\ \ddot{q}^{(2)} \end{bmatrix} + \begin{bmatrix} \dot{q}^{(2)} - q^{(1)} + q^{(2)} \\ -\dot{q}^{(1)} - q^{(2)} + q^{(1)} \end{bmatrix} \end{aligned}$$

Then

$$\begin{aligned} u^{(o)}E^{(o)} &= [0 \quad 1] \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} \ddot{q}^{(1)} \\ \ddot{q}^{(2)} \end{bmatrix} + \begin{bmatrix} \dot{q}^{(2)} - q^{(1)} + q^{(2)} \\ -\dot{q}^{(1)} - q^{(2)} + q^{(1)} \end{bmatrix} \\ &= -\dot{q}^{(1)} - q^{(2)} + q^{(1)} \\ &= 0 \end{aligned} \quad (3.17-21)$$

on shell. Then there are no gauge identities for $E^{(o)}$. Now construct $E^{(1)}$.

$$E^{(1)} = \begin{bmatrix} E^{(o)} \\ \frac{d}{dt} u^{(o)} E^{(o)} \end{bmatrix} = \begin{bmatrix} \dot{q}^{(2)} - q^{(1)} + q^{(2)} \\ -\dot{q}^{(1)} - q^{(2)} + q^{(1)} \\ -\ddot{q}^{(1)} - \dot{q}^{(2)} + \dot{q}^{(1)} \end{bmatrix}$$

which can be written

$$\begin{aligned} E^{(1)} &= W^{(1)}\ddot{q} + k^{(1)} \\ &= \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ -1 & 0 \end{bmatrix} \begin{bmatrix} \ddot{q}^{(1)} \\ \ddot{q}^{(2)} \end{bmatrix} + \begin{bmatrix} \dot{q}^{(2)} - q^{(1)} + q^{(2)} \\ -\dot{q}^{(1)} - q^{(2)} + q^{(1)} \\ -\dot{q}^{(2)} + \dot{q}^{(1)} \end{bmatrix} \end{aligned}$$

There zero modes of $W^{(1)}$ are

$$W^{(1)} \begin{Bmatrix} 0 & 1 & 0 \\ 1 & 0 & 1 \end{Bmatrix}$$

[00323] The first zero mode is the previous one augmented by one dimension and reproduces the previous constraint. The second mode reproduces the negative of the constraint (3.17-21). That is,

$$v^{(1)}E^{(1)} = -u^{(o)}E^{(o)}$$

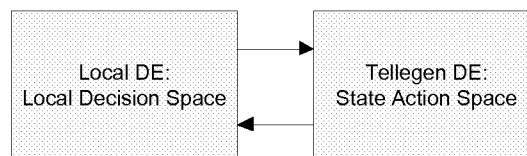
with $v^{(1)} = [1 \ 0 \ 1]$. This leads to the gauge identity

$$G^{(1)} = v^{(1)}E^{(1)} + u^{(o)}E^{(o)} \equiv 0$$

Companionship: Reconciling agents in the network.

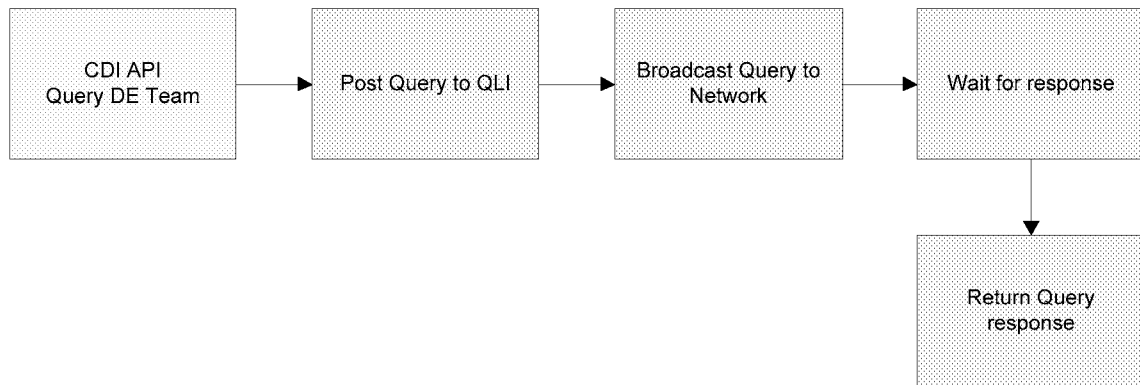
[00324] The outline of the companionship process is as follows for a system of N agents.

- Determine the state action space of the system for $N - 1$ agents to create a Tellegen decision element.
- Update the remaining agent with the Tellegen DE.
- Repeat process so that all N agents are updated with respect to their Tellegen DEs.

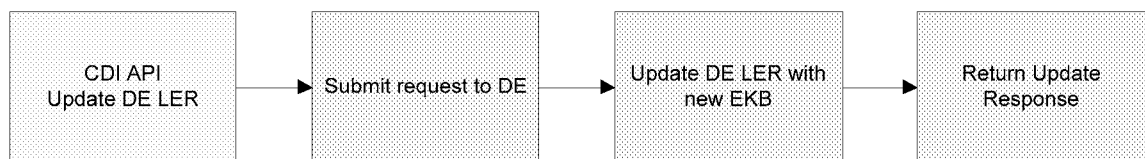


- User submits query.
- System used KB to establish equations of motion for system in Lagrangian or Hamiltonian form.
- System determines optimal trajectory via optimization algorithm of the equations of motion that conform to the principle of least action.

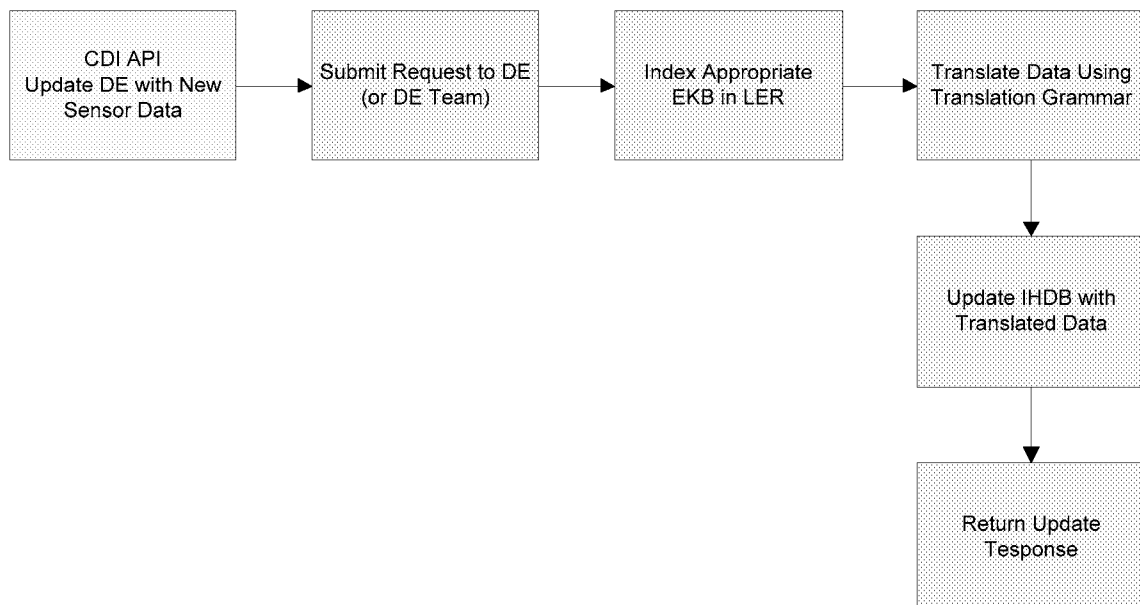
- System returns solution which is a point in the phase space and also serves as an answer to the query.



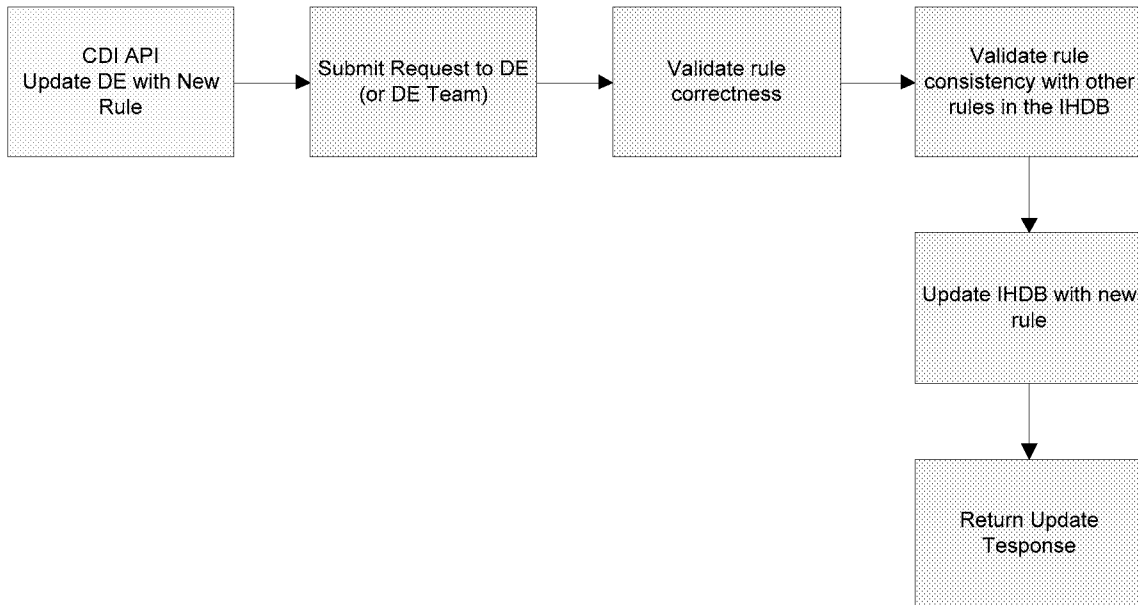
Flow for updating DE's with new external repositories.



Flow for updating DE's with new sensor data.



Flow for updating DE's with new rules.



[00325] The following describes an example use of the CDD system with respect to a target system that includes a vehicle being controlled.

1. Overview

[00326] Hamiltonian dynamic control technology is applied to a motorized bicycle to extend the range of the bicycle by recharging the battery through dynamic braking, harmonic harvesting and pedaling, using Hamiltonian-Chattering control. The proposed dynamic controller has the ability to synchronize different power phases from multiple frequencies to reduce energy dissipation.

[00327] The electric bicycle controller of this embodiment generates all electric power runs with a maximum of 23 miles between charges on a flat trajectory. The use of the described technology can sustain the charge of the battery through pedaling and braking, so there is no need to plug-in to recharge. The design goal is to achieve a range of 50-75 miles between charges without pedaling. The rider of the bicycle will have several options: there will be an accelerator in the handle bars to set the desired speed; amount of braking; and percentage charge settings

that will determine how much of the energy from pedaling goes to charge the battery. The bicycle with the controller can be used on a route profile that includes segments of hills and plains, so that the battery will be used to go up hill, and can be recharged on the downhill and during pedaling on the flat areas, and then used again on the next uphill.

2. Bicycle Control System Design with Hamiltonian-Chattering Control

[00328] The motorized bicycle with dynamic control is designed to satisfy the commands of the rider (accelerator, braking, and percentage charge settings when pedaling), to satisfy regulation rules and rules for the rider, and to be energy efficient through harmonic harvesting and dynamic braking. The controller sends control signals to the battery and motor to determine the velocity rate, charge, temperature, field and armature.

[00329] The proposed control design and implementation processes are based on 1) constructing online a function called the Pontryagin Hamiltonian that encodes the desired behavior, the current state of system rules, including sensory data, and the dynamic constraint rules (requirements) and 2) the online construction of an implementation of the control law using Variational Chattering.

[00330] The proposed approach, which is based on Pontryagin's principle of optimality *seeks* to determine the Pontryagin Hamiltonian directly, instead of relying on an accurate description of the dynamics of the system as in conventional optimal control. For systems such as the motorized bicycle, the dynamics of the driver and the dynamics describing the interactions between all the components are largely unknown and very difficult to obtain. Whereas, knowing a good estimate of the system Hamiltonian of the system provides enough information to satisfy desired behavior of the controlled system.

[00331] The controller implementation constructs a Hamiltonian. The desired behavior rules that define the Hamiltonian-Chattering controls system for the motorized bicycle are to significantly increase the efficiency of the motorized bicycle through harmonic harvesting and dynamic braking rules and drivability

rules. These rules ensure that the changes in the motor from drive to charge are not perceived as being jerky, but rather, the driver experiences a smooth ride. Other rules may be added to the criterion using CDI, such as, a rule describing the durability of the battery.

[00332] Figure SS illustrates the design of the Hamiltonian-Chattering control system, providing an overview of the continuous time design, from which is extracted a discrete design system including delay, latency, and memory for a detailed implementation.

[00333] The Hamiltonian is estimated from sensor data based on the Radon reconstruction approach. In Figure SS, the sensors provide information to the tomograph, which estimates the total Hamiltonian of the system. Given knowledge of the Hamiltonian from physical principles (as with the bicycle and motor), it can be used to correct the estimation.

[00334] The control generator uses our chattering control methodology to create the control signal. The control signal is used by the "plant", and is also used in the tomograph with an update of the estimated Hamiltonian.

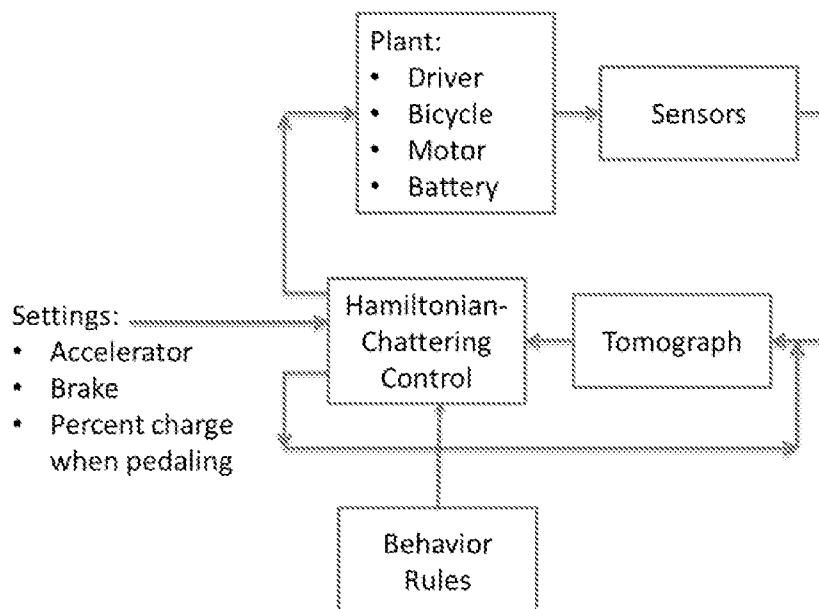


Figure SS

[00335] Figure TT illustrates the flow of power, control, and sensor signals in the motorized bicycle with dynamic control. The driver has several options of how to ride the bicycle: setting the accelerator; setting the brake; and selecting the percentage of pedaling power to be used in charging the battery (e.g., 5%, 10%, 15%). The control system has several modalities: accelerator with harmonic harvesting, braking, and pedaling with percent of charge for the battery.

[00336] As an illustration, when the driver sets the accelerator, a control signal from the driver is sent to the controller, which then sends a control signal to the actuator. The actuator converts the control signal into a power signal that drives the motor. When the driver brakes, the control signal from the controller alerts the actuator to stop driving the motor. In addition, the friction from braking produces a reverse torque in the motor, which makes the motor act as a generator in charging the battery. When the driver is pedaling and selects a percentage of power to be used in charging the battery, the control signal tells the actuator to utilize the appropriate amount of power produced to charge the battery. In Figure SS, the box containing the actuator, motor and battery also contains a charging module.

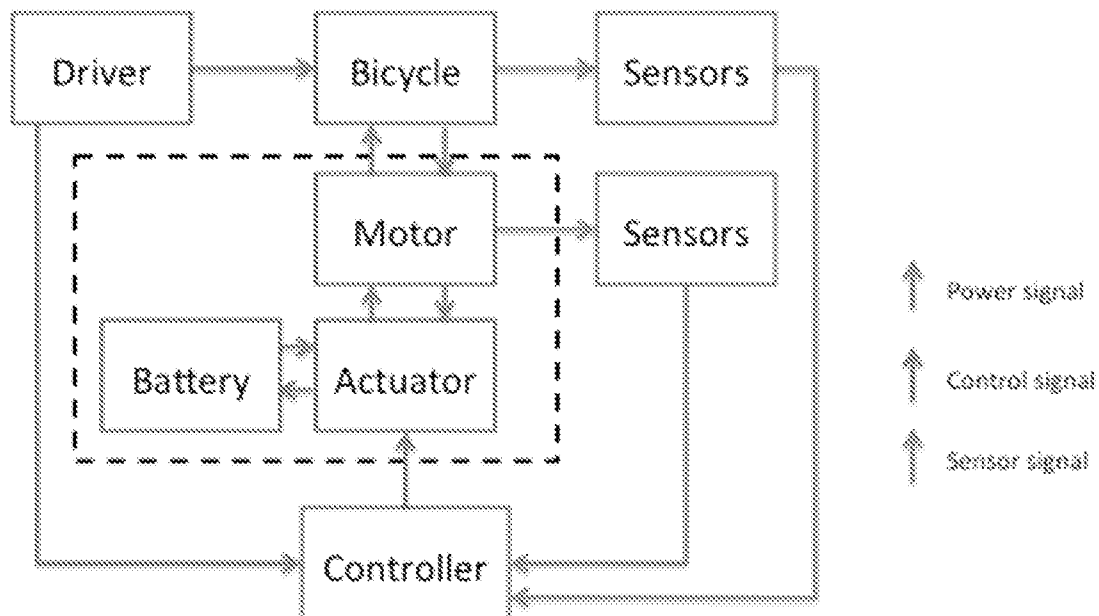


Figure TT

3. Open Loop Feedback Strategy

[00337] The Hamiltonian-Chattering control system for the motorized bicycle is implemented with an Open-Loop-Feedback-strategy. This strategy consists of two processes: a Hamiltonian Upgrade Process, which realized the feedback mechanism of the strategy; and a Control Propagation Process that realized its open loop mechanism.

[00338] The Hamiltonian Upgrade Process is a discrete time operation that involves the incorporation of active rules and sensory data to encode a new Hamiltonian. It involves a tomograph for function reconstruction that will be detailed in a separate document. Some characteristics of this process are illustrated in Figures UU, VV, and WW. The Control Propagation Process is implemented by a chattering approach described below.

[00339] One innovation is to use open loop feedback based on the Hamiltonian, instead of a dynamic programming approach using the Hamiltonian-Jacobi-Bellman theory and extensions. This approach strives to get a deterministic model asymptotically by using feedback on the Hamiltonian directly as compared to using feedback on states. If the system is deterministic, the Pontryagin Hamiltonian approach gives the same answer as dynamic programming (i.e., feedback on states). This ensures better representation of the system dynamics because the open-loop feedback mechanism on the Hamiltonian can implicitly accommodate for the structural uncertainties in the Hamiltonian while this is not always the case for a feedback mechanism on the states. Additionally, this approach is computationally practical and defeats the curse of dimensionality associated with dynamic programming because the elements (including tomography, parallel transport, and chattering control) are quickly computable in polynomial time.

[00340] The Bellman's formulation for the dynamic programming assumes that the Hamiltonian function is known and it doesn't change for the entire horizon of the dynamic program. While, this is not true in this case because the Hamiltonian function is estimated iteratively which changes the manifold on which the system trajectories lie. Hence, the open loop feedback strategy gives us another

advantage which allows us to obtain and control the system dynamics using Pontryagin's equations for the window of time where the system operates in an open loop mechanism, while using the feedback mechanism at the edge of this window to parallel transport the states, co-states and constraints ensuring conformity.

[00341] In a conventional optimal control approach, the criterion and dynamics are assumed to be known or partially known. In contrast, our approach does not assume explicit knowledge of the dynamics or the criterion. We assume, as given, rules that characterize the desired behavior (hard and soft rules), rules that characterize the conservation principles of the dynamics (absolute rules), and rules that represent the sensory data (soft rules). We construct a functional form, called Pontryagin Hamiltonian, which encodes the knowledge contained in the rules.

[00342] The total Pontryagin Hamiltonian (subscripted with $\mathcal{C}$) has the following form,

$$H_{\mathcal{C}}(x(t), p(t), u(t)) = H_0(x(t), p(t), u(t)) + \sum_i \lambda_i \phi_i + \sum_i \mu_i \psi_i$$

where the first term describes the fundamental dynamics, the second term represents the rules that generate holonomic constraints, and third term represents the rules that generate non-holonomic constraints. The functional form of the Pontryagin Hamiltonian is generated algorithmically.

[00343] In a Pontryagin Hamiltonian constructed by CDI, the criteria are defined by the rules implicitly, and represented in the Pontryagin Hamiltonian.

[00344] The following theorem establishes the existence of an optimal control signal $u^*(t)$. The theorem expresses this existence through three conditions.

[00345] Theorem: Given a Pontryagin Hamiltonian $H_{\mathcal{C}}(x(t), p(t), u(t))$, there exists an optimal control $u^*(t)$ satisfying the following three conditions:

$$(1) \quad H_{\mathcal{C}}(x^*(t), p(t), u^*(t)) \leq H_{\mathcal{C}}(x^*(t), p(t), u(t)) \quad \forall u \in U$$

$$(2) \quad \dot{x}^*(t) = \frac{\partial H_C(x^*(t), p(t), u^*(t))}{\partial p(t)}$$

$$(3) \quad \dot{p}(t) = -\frac{\partial H_C(x^*(t), p(t), u^*(t))}{\partial x(t)}$$

that minimizes

$$\min_u \int_0^T (H_C(x, p, u) - p^T \dot{x}) dt + g(x(T))$$

where $x(0) = x_0$ and $p(T) = \partial g(x(T)) / \partial x$.

[00346] The Pontryagin Hamiltonian is estimated recursively over time. For each Hamiltonian $H_C(x^*(t), p(t), u^*(t))$ in the convergent sequence of Hamiltonians, there exists a control $u^*(t)$ that satisfies the optimality condition in (1).

[00347] We estimate a Hamiltonian form using the desired behavior and conservation rules in a Monte Carlo simulation to obtain an initial estimate of a Pontryagin Hamiltonian $\hat{H}_C|_{t=0}$. This functional form is a valid Pontryagin Hamiltonian over the entire space $X \times P \times U$, the symplectic space of the initial Pontryagin Hamiltonian.

[00348] The open loop feedback strategy consists of constructing a sequence of Hamiltonians starting with the initial Hamiltonian estimate along the system trajectory $x(t), p(t), u(t)$, corrected by sensory observations.

[00349] If $H_C(x(t), p(t), u(t))$ is differentiable with respect to $u(t)$, then condition (1) implies the following necessary condition for optimality,

$$(4) \quad \frac{\partial H_C(x^*(t), p(t), u^*(t))}{\partial u(t)} = 0.$$

Condition (4) is approximated by solutions of the following together with $\frac{\partial^2 H_C(x^*(t), p(t), u^*(t))}{\partial u(t)^2} > 0$,

$$(5) \quad \dot{u}^*(t) = \lim_{\delta \rightarrow 0} -\frac{1}{\delta} \left(\frac{\partial^2 H_C(x^*(t), p(t), u^*(t))}{\partial u(t)^2} \right)^{-1} \frac{\partial H_C(x^*(t), p(t), u^*(t))}{\partial u(t)}.$$

The inverse form in (5) can be approximated by a solution of a matrix differential equation, thus avoiding computing matrix inverses.

[00350]

Our open loop feedback approach with Hamiltonian-chattering control is illustrated in Figures UU and VV. It consists of the following steps:

Step 0: Given $\hat{H}_C|_t$ at time t , and $x(t) = x_t, p(t) = p_t, u(t) = u_t$ from sensor measurements. Also given terminal condition, $p(T) = p_T$.

Step 1: Propagate trajectories forward from time t to the end of the time horizon T , using

$$\begin{aligned}\dot{x}(t) &= \frac{\partial H_C(x(t), p^f(t), u(t))}{\partial p} \\ \dot{p}^f(t) &= -\frac{\partial H_C(x(t), p^f(t), u(t))}{\partial x} \\ \dot{u}(t) &= -\delta \left(\frac{\partial^2 H_C(x(t), p^f(t), u(t))}{\partial u(t)^2} \right)^{-1} \frac{\partial H_C(x(t), p^f(t), u(t))}{\partial u(t)}\end{aligned}$$

and starting from initial conditions $x(t) = x_t, p^f(t) = p_t, u(t) = u_t$. Also, $\delta \ll \Delta$.

Step 2: Propagate the trajectory $\dot{p}^b(t)$ backwards, from time T to $t + \Delta$, using

$$\dot{p}^b(t) = -\frac{\partial H_C(x(t), p^b(t), u(t))}{\partial x}$$

and terminal conditions $x(T), p^b(T) = p_T, u(T)$.

Step 3: Propagate the estimate of the total Pontryagin Hamiltonian $\hat{H}_C|_t$ to get $\tilde{H}_{t+\Delta^-|t}$, using $x(t), p^b(t)$, and $u(t)$ (see [2]).

Step 4: Perform Hamiltonian-Chattering control on $[t, t + \Delta]$.

Step 5: Implement the chattering control.

Step 6: Collect measurements of $x(t), p(t), u(t)$ over $[t, t + \Delta]$.

Step 7: Estimate $\hat{H}_C|_{t+\Delta}$ using the measurements and tomography.

Step 8: Calculate the total Hamiltonian,

$$H_C(x(t + \Delta), p(t + \Delta), u(t + \Delta)) = \hat{H}_C|_{t+\Delta} + \tilde{H}_{t+\Delta^-|t}.$$

The summation is valid because $\hat{H}$ and $\tilde{H}$ are orthogonal.

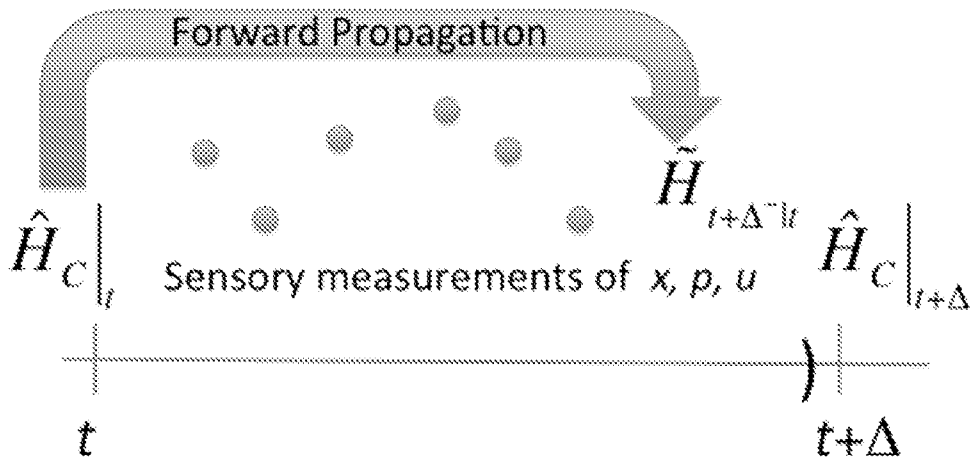


Figure UU. Illustration of open loop feedback strategy.

[00351]

In this open loop feedback approach, the initial condition x_0 is known, and $x(t)$ at time t can be calculated or measured from sensors. The initial condition for u at time 0 is not usually known, and is chosen arbitrarily, however, additional information may be incorporated if available. The terminal condition for $p(T) = p_T$ is known, and given by $p(T) = \partial g(x(T))/\partial x$. For the bicycle, the terminal cost $g(x(T))$ is zero, so $p_T = 0$.

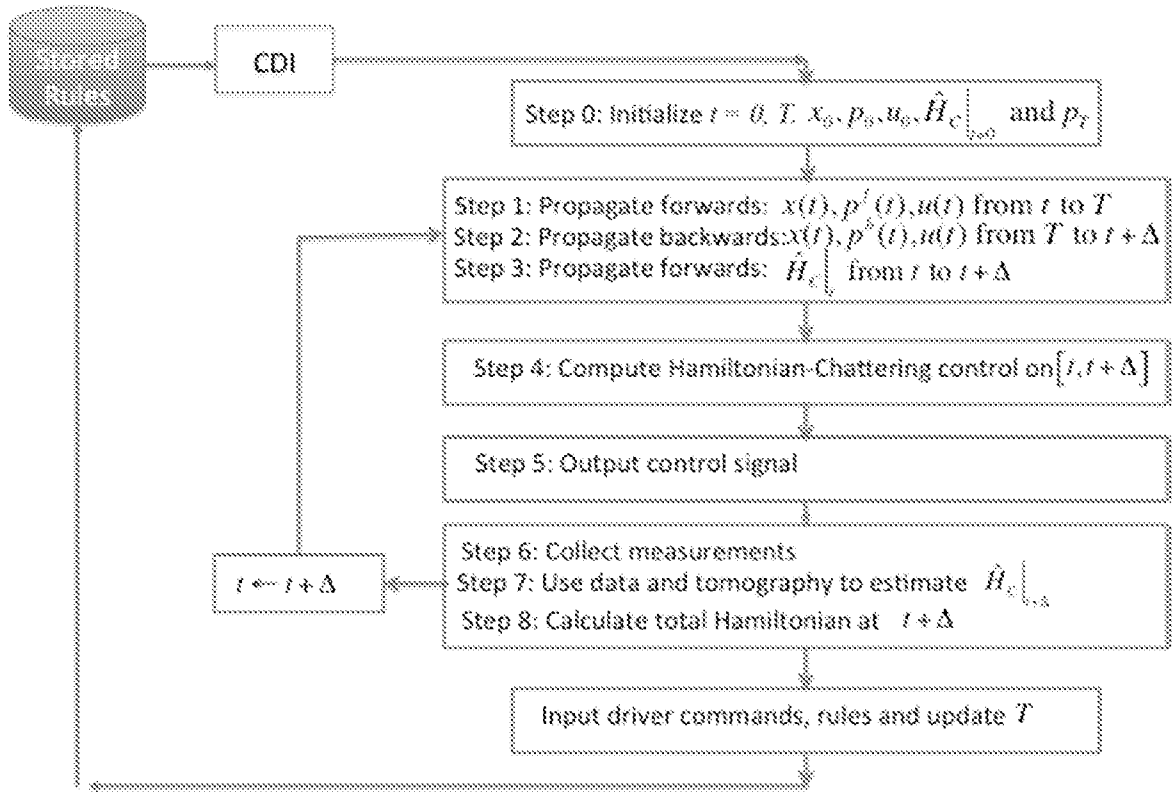


Figure VV. Flowchart of open loop feedback strategy with Hamiltonian-chattering.

[00352]

Figure WW illustrates the overall dynamic control methodology. In Figure WW, the Hamiltonian tomograph estimates the total Hamiltonian from sensors, using a Radon reconstruction approach (tomography). The Hamiltonian model generator constructs Hamiltonians using physical models when available (e.g., the bicycle and motor), and incorporates absolute and hard rules. The tomograph estimation does not separate components from the rules, the driver, and physical components. It estimates the total Hamiltonian, including the dynamic soft rules. This is expressed as:

$$H_C(x(t), p(t), u(t)) = H_0(x(t), p(t), u(t)) + \sum_i \lambda_i \phi_i + \sum_i \mu_i \psi_i.$$

The Hamiltonian corrector can reconcile the Hamiltonian coming from the Hamiltonian Model Generator with the total Hamiltonian coming from the Hamiltonian Tomograph.

[00353] Given the Hamiltonian, the control generator uses chattering to provide the control signals. The chattering principles have been established on the Lagrangian and state model, and are extendible to a Hamiltonian model. The control signals are obtained, and passed to the actuator to control the motor.

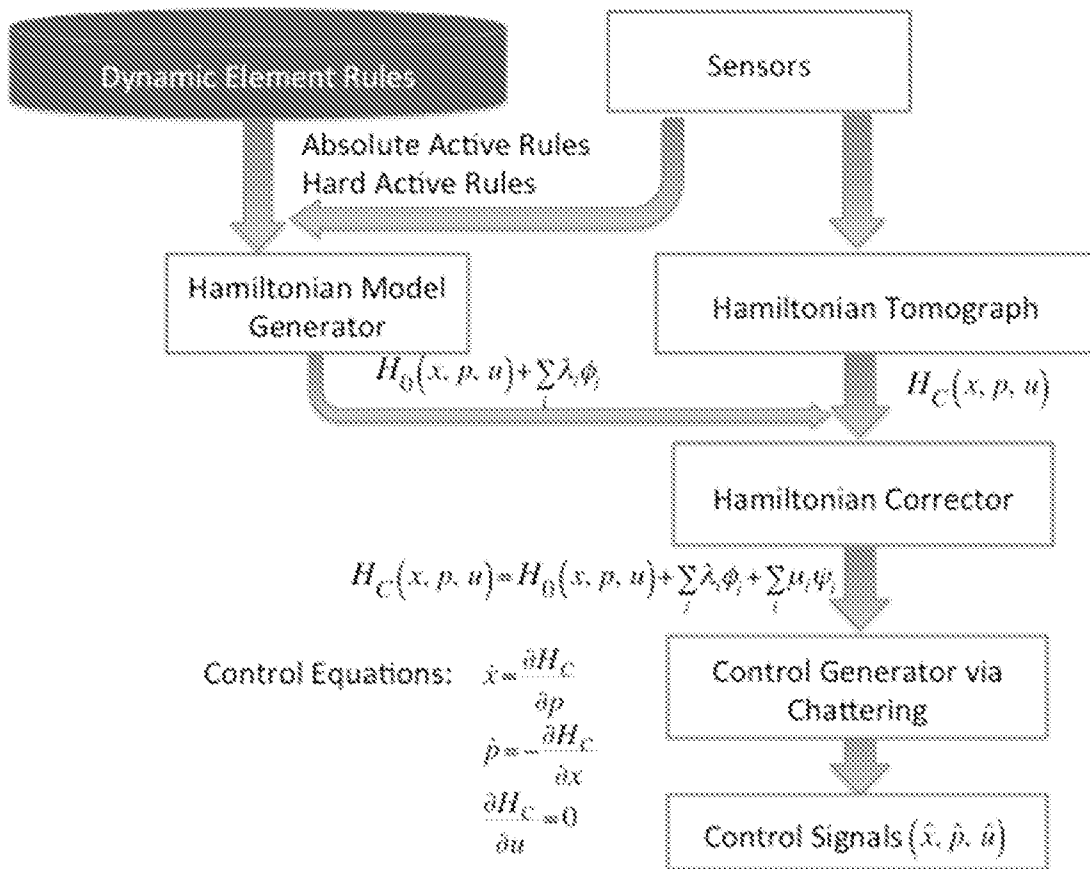


Figure WW. Overview of Dynamic Control Methodology.

4. Hamiltonian Construction

[00354] The dynamic control methodology includes a Hamiltonian model generator. Two methods of developing the Hamiltonian are used; one is using the physical description of the system, and the second is using tomography when physical laws are largely unknown. For example, the Hamiltonians for the bicycle, motor, and battery are constructed using the known physical laws, but the Hamiltonian for the

driver is unknown. The total Hamiltonian is estimated using tomography given empirical data.

[00355] The Hamiltonian is a function of the state x , momentum p , and the control u .

4.1 Hamiltonian Model Generator via Lagrangian

One approach is to develop the Hamiltonian $H(x, p, u)$ from the Lagrangian description of the mechanical system. In general, the Hamiltonian can be expressed as

$$H(x, p, u) = pu - L(x, u)$$

where x is the state, $\dot{x} = dx/dt$ is the velocity, $p = \partial L / \partial \dot{x}$ is the momentum, and the Lagrangian $L(x, \dot{x}, u)$ can be computed by solving the inverse Lagrangian problem from the equations of motion:

$$\frac{d}{dt} \frac{\partial L(x, u)}{\partial \dot{x}} - \frac{\partial L(x, u)}{\partial x} + \frac{\partial P}{\partial \dot{x}} = F_{cr} + F_{np}$$

where F_{cr} are the constraint holonomic forces, F_{np} are the non-holonomic constraint forces and P is the dissipative energy (if any). This approach may be used for the bicycle, where we have an expression for the Lagrangian from physical principles, but it is not a valid approach when we do not have a model of the physical system.

4.2 Hamiltonian Construction via Tomography

[00356] A Radon transform is used to approximate the Hamiltonian $\hat{H}_T|_t$ by probing densities of energy level curves. The set of states $X \subset R^n$ is often referred to as the configuration space. Figure XX illustrates the phase space $X \times P \times U$, with Hamiltonian $H(x, p, u)$ level curves, where p is called the momentum, $p \in P \subset R^n$, and $u \in U$. The line in the figure represents a probe, with points along the probe that are observed, and the density along the probe is estimated. Using a generalized tomography approach, the observation Hamiltonian can be "reconstructed" from observations. In the control model, u is the control action,

$u \in R$, and may represent torque on the back-wheel, voltage across the motor, the actuator, or other physical controls.

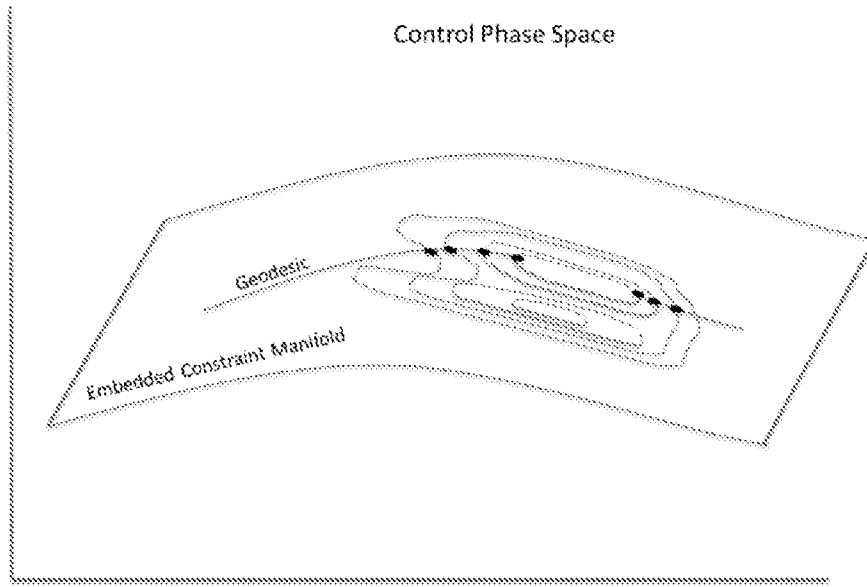


Figure XX. Hamiltonian $H(x, p, u)$ level curves in the phase space $X \times P \times U$ with points along a probe.

5. Chattering Control

[00357] The chattering control operates on the evaluated total Hamiltonian for each profile setting selected by the user. It tracks the driver state, and determines the optimal control u^* for the selected profile setting. This total Hamiltonian consists of the system Hamiltonian (H_S) and the Hamiltonian rules (denoted ϕ_i).

[00358] As described above, as long as the Hamiltonian of the system is known, the Pontryagin optimization problem can be solely represented in terms of the Hamiltonian and its partial derivatives. The optimization problem is,

$$\min_u \int_0^T (H_c(x, p, u) - p^T \dot{x}) dt + g(x(T))$$

where the terminal cost for the bicycle is zero, i.e., $g(x(T)) = 0$.

[00359] The equation of state from the Hamiltonian including rules, as derived in CDI, is given as

$$\begin{aligned}\frac{dx}{dt} &= \frac{\partial H_C}{\partial p} = \frac{\partial H_S}{\partial p} + \sum_i \lambda_i \frac{\partial \phi_i}{\partial p} + \sum_i \mu_i \frac{\partial \psi_i}{\partial p} \\ \frac{dp}{dt} &= -\frac{\partial H_C}{\partial x} = -\frac{\partial H_S}{\partial x} - \sum_i \lambda_i \frac{\partial \phi_i}{\partial x} - \sum_i \mu_i \frac{\partial \psi_i}{\partial x} \\ \frac{du}{dt} &= \lim_{\delta \rightarrow 0} -\frac{1}{\delta} \left(\frac{\partial^2 H_C(x(t), p(t), u(t))}{\partial u(t)^2} \right)^{-1} \frac{\partial H_C(x(t), p(t), u(t))}{\partial u(t)}\end{aligned}$$

where the control is rate of voltage, v , supplied to each phase of the motor,

$$u = \dot{v}.$$

It is preferable to control the rate of the voltage rather than the voltage itself because we want to be able to chatter the system to charge and discharge almost instantaneously and also prevent the motor to experience unwanted spikes, which we cannot ensure when controlling the voltage directly. Hence we introduce (β, ϑ) as the voltage state and costate and modify the total Hamiltonian as $H_C \leftarrow H_C + \vartheta^T u$.

We solve the above three equations of state by first discretizing the time interval $[0, T]$ into short intervals of time of length Δ . Note, the intervals do not have to be of equal length, but here we keep them of equal length Δ for ease of notation. Second, we use a Lebesgue-like approximation of the integral, as illustrated in Figure YY.

[00360] Figure YY illustrates the chattering levels of control, where k is the index for the chattering levels, and c_k^t denotes the value of the k th control level over time interval $[t, t + \Delta]$. In Figure YY, the shaded regions correspond to the amount of time the control spends at each level.

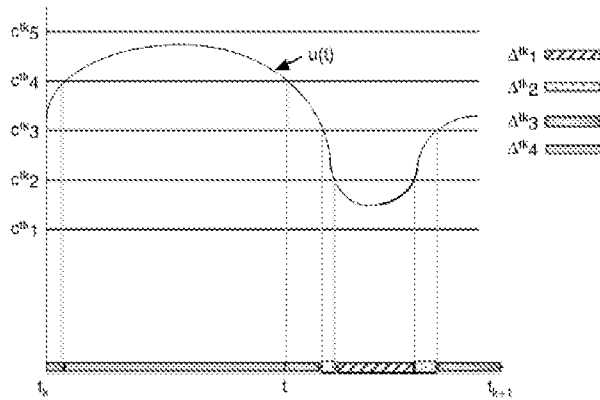


Figure YY. Chattering levels for the control on an interval (from Kohn, et al., 2010).

[00361] The idea behind chattering control is to solve for the amount of time α_k^t the optimal control $u^*(t)$ spends at level c_k^t during the time interval $[t, t + \Delta]$, as illustrated in Figure ZZ. A flowchart of the open loop strategy with the chattering control is in Figure VV. Any terminal condition, including the fixed terminal co-state and others on the terminal state, is propagated backwards by parallel transporting these constraints along a geodesic of the embedded manifold defined by the constraints. If the dynamics of the co-state given by equations (3) are parallel to the geodesics of the embedded manifold, those equations are used to propagate terminal conditions backwards, thereby arriving at constraints (7). The state and control estimate required to propagate the terminal co-state is computed using the 4th order Runge-Kutta method for the equations (2) and (5). Now we have an initial value problem, with the initial values computed at the end of the previous interval, which are used in the approximated equations of state to solve the chattering optimization problem for the current interval.

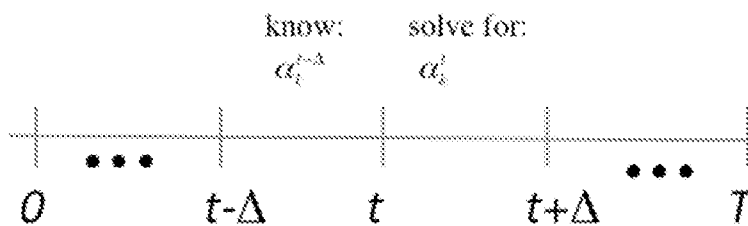


Figure ZZ. Chattering control on each interval $[t, t + \Delta]$.

[00362] This leads to the following linear program,

$$\min_{\alpha_k^t, \forall k} \sum_k H_C|_t(x_t, p_t, c_k^t) \alpha_k^t \Delta \quad (6)$$

$$\text{subject to } p(t) - p(t + \Delta) = \sum_k \frac{\partial H_C|_t}{\partial x}(x_t, p_t, c_k^t) \alpha_k^t \Delta \quad (7)$$

$$\sum_k \alpha_k^t = 1 \quad (8)$$

$$\alpha_k^t \geq 0 \quad \text{for all } k.$$

The chattering control seeks to minimize the total Pontryagin Hamiltonian that satisfies condition (1), repeated here as,

$$H_C(x^*(t), p(t), u^*(t)) \leq H_C(x^*(t), p(t), u(t)) \quad \forall u \in U.$$

The fundamental lemma of calculus of variations (see [6]), says that, if $H_C > 0$, and $H_C(x^*(t), p(t), u^*(t)) \leq H_C(x^*(t), p(t), u(t)) \quad \forall u \in U$, then $\int H_C(x^*(t), p(t), u^*(t)) dt \leq \int H_C(x^*(t), p(t), u(t)) dt$, weakly a.e., so we use this condition as an objective function in a weak sense, and minimize

$$\min_u \int_t^{t+\Delta} H_C(x, p, u) d\tau.$$

[00363] The objective function in (6) is used as a chattering approximation to this integral.

[00364] The constraints in (7) are approximations to condition (3). Constraint (8) ensures that α_k^t cover the interval, and the final constraints ensure non-negativity. The linear program can be solved numerically with any standard numerical LP solver.

[00365] The optimal solution can be interpreted as the probability the optimal control spends at the different control levels. The levels c_i^t are decided based on the driver's "comfortability", so that the changes in control are smooth enough for a comfortable ride.

[00366] This approach uses an open-loop feedback control strategy where the evaluated Hamiltonian is assumed constant for the short intervals of time Δ and the above optimization problem is solved for every such time interval. The

algorithm for this approach to solve the optimal control problem is shown in Figure YY.

[00367] The linear program in the chattering algorithm uses the Hamiltonian evaluated at every time interval to compute the Pontryagin criteria and the necessary constraints by computing the partial derivatives of the Hamiltonian with respect to the control variable at every Δ time step. This gives us a great advantage over conventional methods as the algorithm does not have to accommodate for state space explosion, i.e curse of dimensionality. This is true because the control space is usually very small for all practical purposes. The algorithm described above only needs to calculate the following quantities: $\frac{\partial H_C}{\partial u}$, and $\left(\frac{\partial^2 H_C}{\partial u^2}\right)^{-1}$. The quantities $\frac{\partial H_C}{\partial p}$ and $\frac{\partial H_C}{\partial x}$ can be measured as they are just precisely equal to $\dot{x}$ and $-\dot{p}$, respectively.

[00368] Tests of the Hamiltonian Chattering control on a set of test Hamiltonians for which the state and the co-state are plotted in Figures AB-AG.

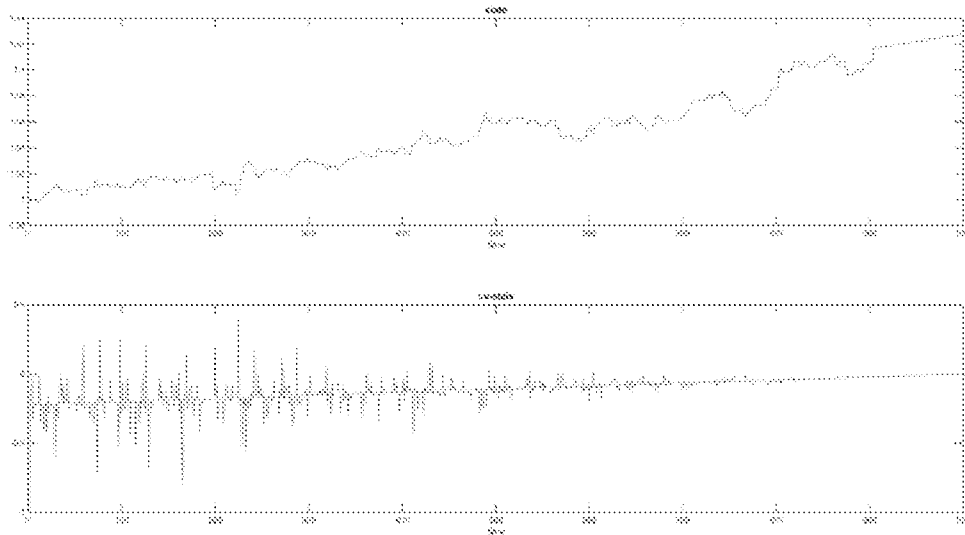


Figure AB $H = p(x + u) - (x^2 + u^2)$

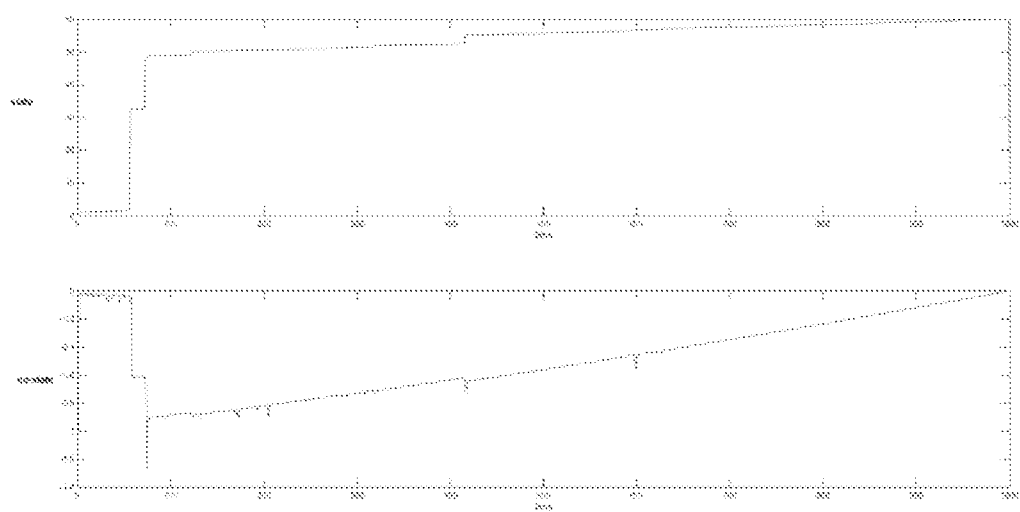


Figure AC. $H = p(x + u^2 + u) - (x^2 + u^2)$

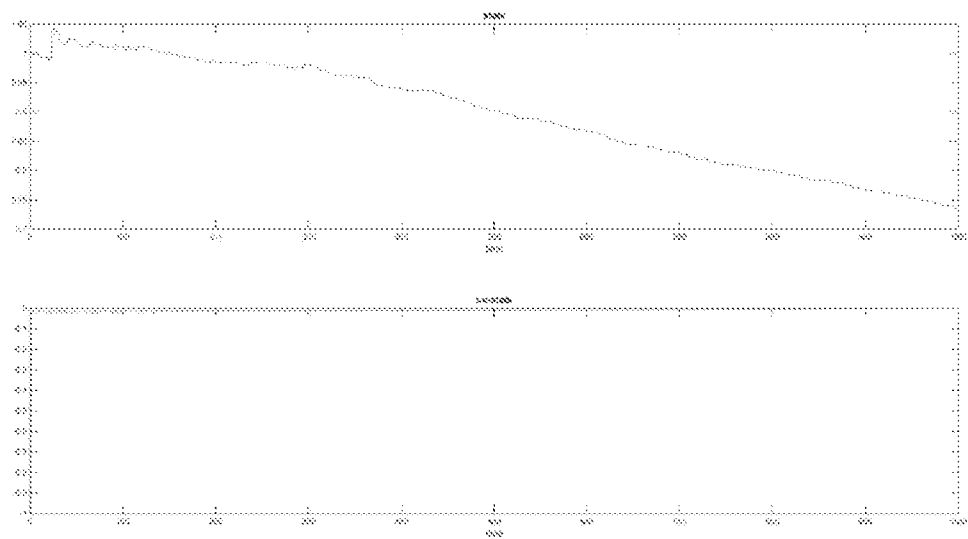
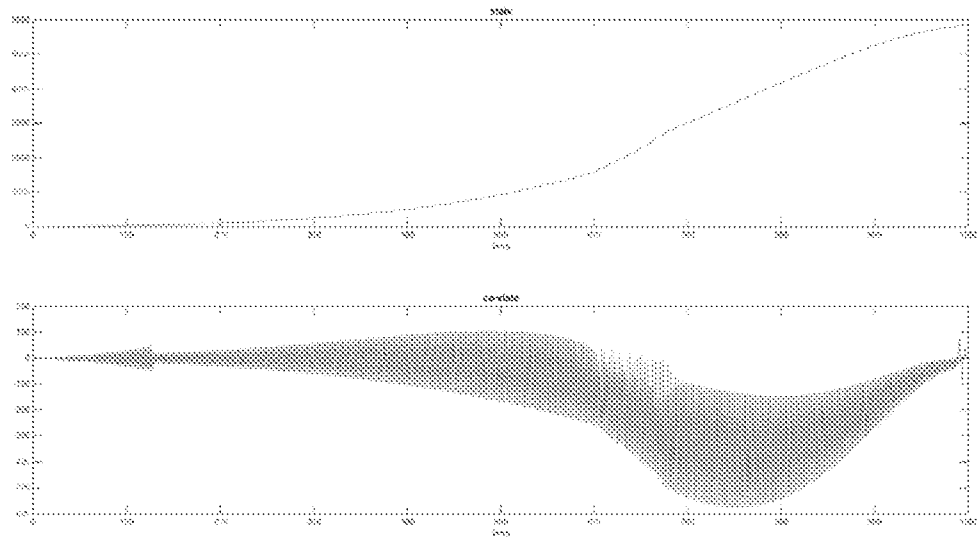
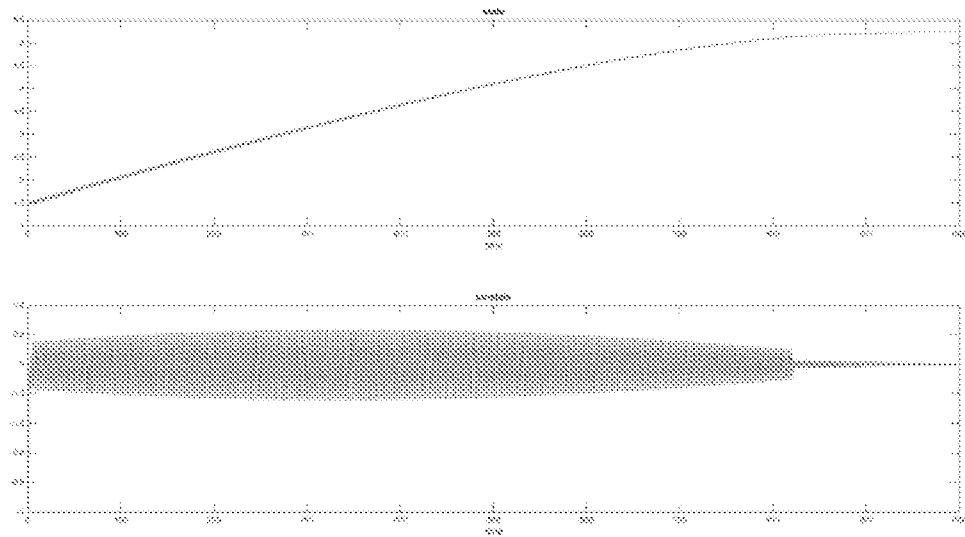


Figure AC. $H = p(x + u) - x^2$

Figure AD. $H = p(x + u) - ux^2$ Figure AE. $H = p(x + u) - (ux^2 + u^2)$

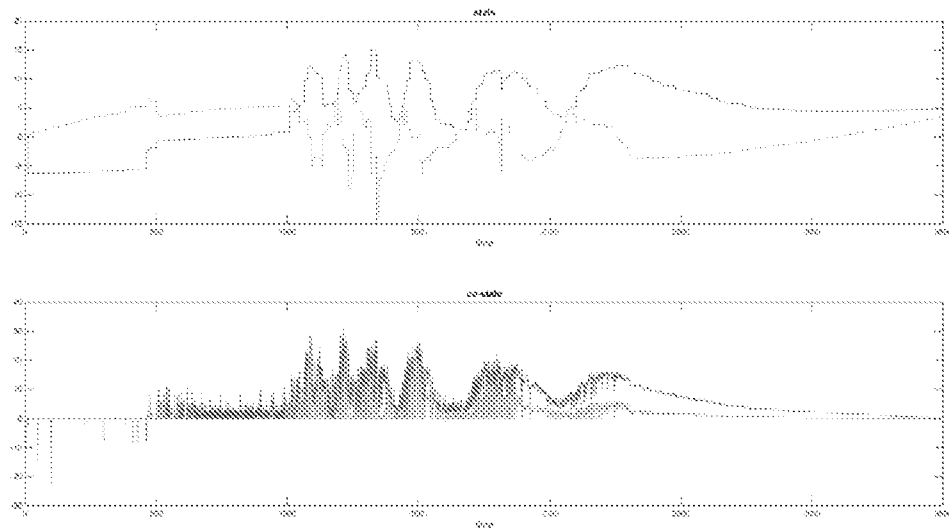


Figure AF. $H = u^2 + x_1^2 + p_1x_2 + p_2(-x_1 + x_2(1.4 - 0.14x_2^2)) + 4u$

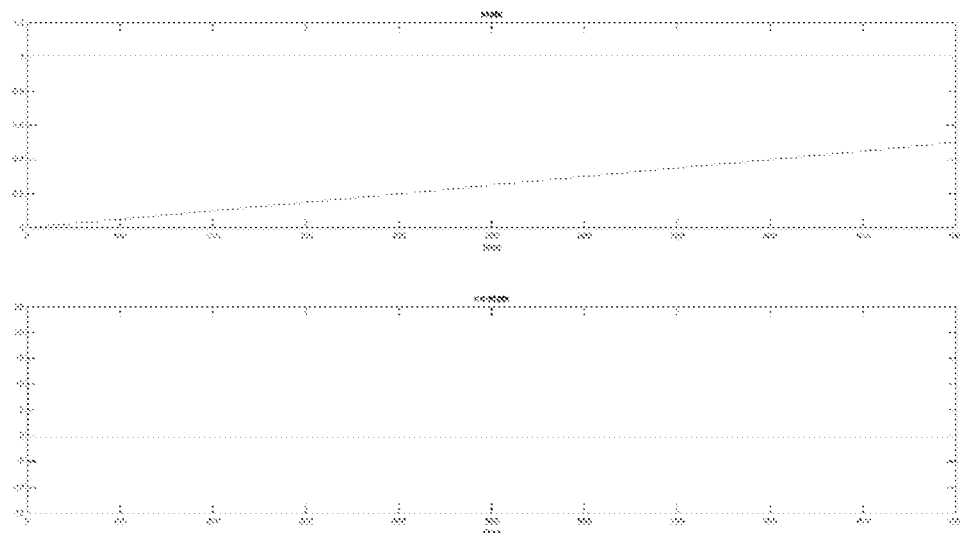


Figure AG. $H = u^2 + p_1x_2 + p_2x_1$

6. Results Describing Characteristics of the Motorized Bicycle

[00369] This section provides results describing characteristics of the motorized bicycle, including the range of power contained in multiple harmonics and the bandwidth response of the bicycle.

[00370] Power harvested from three harmonics (H1, H2, and H3) are used as feedback sources to drive the motor. Typically, harmonics are modeled as a

function of the load, but we model the harmonics as EMF dependent sources in the motor model.

[00371] Figure AH provides a circuit diagram of the motor with harmonics as two power sources.

[00372] The harmonics have constant frequencies, but the phases are not constant and need to be synchronized. The phases, denoted $\psi(t, EMF)$, are a function of time and EMF, and EMF is a function of current, $EMF = k\phi I_a$, where ϕ is the magnetic field, k is a constant, and I_a is current. Current is easily measured, whereas EMF is not, so we work with current. The parameter ϕ is measured and k is estimated from the Kalman filter.

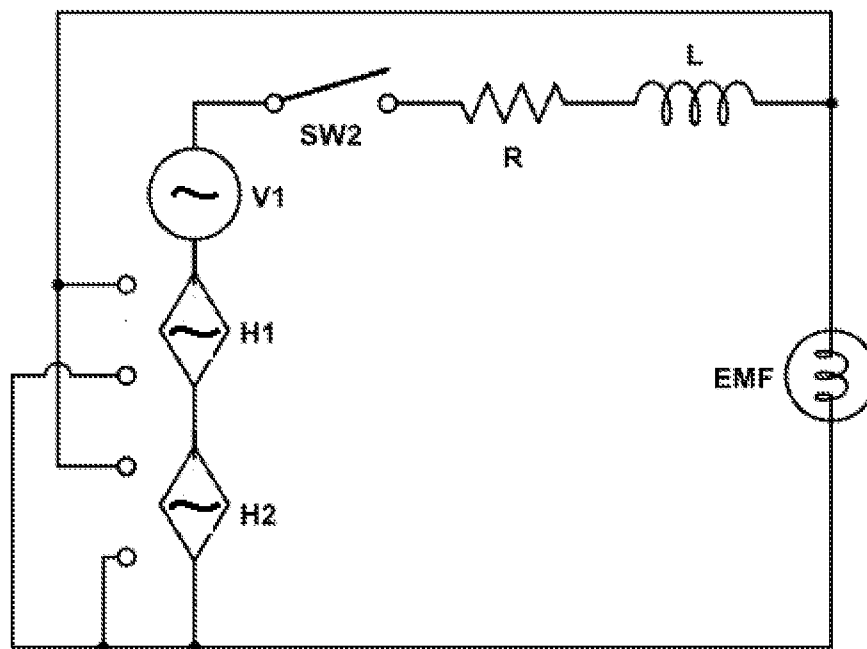


Figure AH. A circuit diagram of the motor with harmonics as power sources.

[00373] Figure AI illustrates the power spectra on the power scale (log equivalent) for frequencies of the motor at two different loads. As can be seen in Figure AI, the higher harmonics contain a lot of energy that is currently wasted because it is

outside the range of the operating frequency. We use an approach to harvest the energy in the higher frequencies and feedback the energy to the motor as a voltage power source.

[00374] Figure AJ illustrates the bandwidth of the input signal, obtained by a Fourier analysis on the linearized bicycle equations of motion about the stable state at various velocities along the trajectory. When the velocity of the bicycle is high, the bandwidth is low, implying that the system can only operate within a small interval (bandwidth) of the input signal and remain stable. At low velocities, there is a bifurcation of bandwidth due to gyroscopic effects and environmental effects (road profile, wind, etc.). At high velocities, the gyroscopic effects dominate the environmental effects, so there is only one bandwidth.

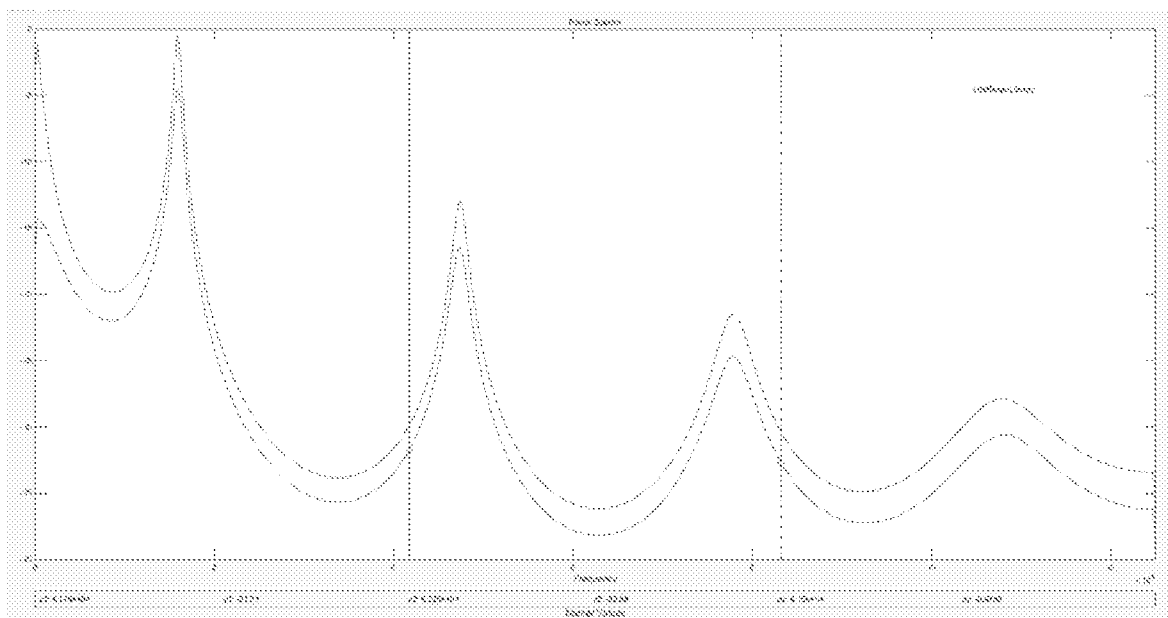


Figure AI. Power spectrum from the motor at two different loads.

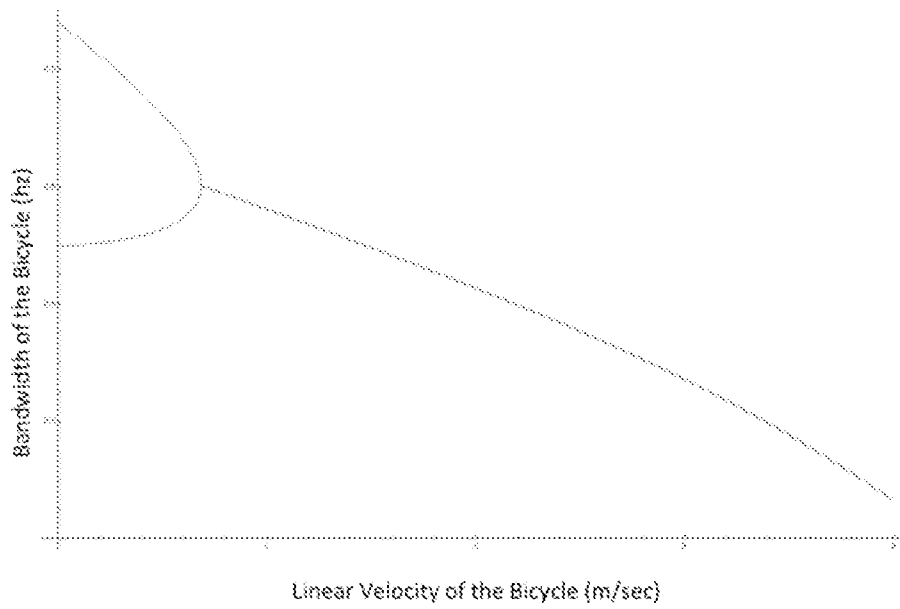


Figure AJ. Velocity and bandwidth of the bicycle.

[00375] The following describes an example use of the CDD system with respect to controlling inventory management.

[00376] Choose the control $u(t)$ to minimize the cost function

$$J(u) = \Psi(x(T)) + \int_0^T L(x, u, t) dt.$$

The state $x(t)$ evolves according to the state equations

$$\dot{x}(t) = f(x, u, t)$$

with constraints

$$a \leq x(t) \leq b$$

$$c \leq u(t) \leq d$$

with $t \in [0, T]$. Define the costate $\lambda(t)$. The Hamiltonian is

$$H(x, \lambda, u, t) = \lambda(t)\dot{x}(t) + L(x, u, t).$$

And the dynamics are governed by

$$\begin{aligned}\dot{\lambda}(t) &= -\frac{\partial H}{\partial x} \\ \dot{x}(t) &= \frac{\partial H}{\partial \lambda}\end{aligned}$$

[00377]

The specification of the inventory control problem is:

$T = 365$	is the terminal time
$u(t) = P(t)$	is the control
$x(t) = I(t)$	is the state
$\dot{x}(t) = f(x, u, t) = P(t) - D(t)$	is the dynamics
$L(x, u, t) = (I(t) - 10)^2$	is the objective function
$\Psi(x(T)) = 0$	is the terminal condition

With constraints:

$$0 \leq I(t) \leq 100$$

$$0 \leq P(t) \leq 40$$

The Hamiltonian is then

$$\begin{aligned}H(x(t), u(t), \lambda(t), t) &= H(I(t), P(t), \lambda(t), t; D(t)) \\ &= \lambda(t)(P(t) - D(t)) + (I(t) - 10)^2\end{aligned}$$

Then

$$\dot{\lambda}(t) = -\frac{\partial H}{\partial x} = -\frac{\partial H}{\partial I} = -2(I(t) - 10)$$

$$\dot{x}(t) = \frac{\partial H}{\partial \lambda} = P(t) - D(t)$$

[00378] The *HamiltonianProvider* signature is currently specified as

```
public interface HamiltonianProvider {

    Double getHamiltonian(StateMap overallState, Double[] dynamicState, Double[]
    costate, Integer t, Double[] control);

    Double[] getStateJacobian(StateMap overallState, Double[] dynamicState,
    Double[] costate, Integer t, Double[] control);

    Double[] getCostateJacobian(StateMap overallState, Double[] dynamicState,
    Double[] costate, Integer t, Double[] control);
}
```

The functions correspond as follows:

$H(x(t), u(t), \lambda(t), t)$	getHamiltonian
$\dot{x}(t)$	getStateJacobian
$\dot{\lambda}(t)$	getCostateJacobian

[00379] The following describes an example use of the CDD system with respect to controlling a micro-grid electrical generation facility.

[00380] Micro-grids require active control to maintain quality of service and to interface with the power grid in a bidirectional manner. Further, micro-grids should be justified by environmental, governmental, and economic viability. We present a programmable architecture for active, optimal distributed control of elements of the

grid to achieve desired behavior. This architecture includes a distributed inductive engine for learning the local dynamics of generators and loads in the micro-grid. It generates feedback laws that are adapted to the current status of the micro-grid, and responds to anomalous events in a resilient manner. One novelty is that control laws are extracted online for bidirectional discontinuous non-linear loads by mean field methods from physics outside the standard design methodologies for piecewise linear quadratic controls.

I. INTRODUCTION

[00381] Consider a micro-grid that serves the energy needs for a building or a cluster of buildings, such as a campus, a hospital complex, an industrial complex, a residential neighborhood, or a military installation. The classic approach to maintain and install such a micro-grid involves a careful design based on detailed stochastic models of the dynamics of the loads and generators. However, the dynamics of a micro-grid with renewable energy sources and smart loads cannot be modeled by standard modeling of dynamical systems in meaningful ways. Our approach uses logic rules to encode customizable dynamic principles of the loads and generation devices, including uncertainties, by learning and adaptation without using an explicit model.

[00382] The high cost of energy and the need for high quality of service and reliability of the micro-grid, suggest the implementation of automated adaptive control strategies that are the result of multi-objective optimal control laws. Microgrids with generation resources are connected to the general power grid to be able to satisfy the variable loads in a cost-effective manner. For micro-grids having generation resources, the bidirectional exchange of power provides additional opportunities to optimize performance.

[00383] Instead, a new architecture for micro-grid distributed intelligent control and management is implemented. The approach we follow is based on principles of Hybrid Systems and Optimization. We build a distributed network of intelligent element controllers that can provide synchronization of a micro-grid to the power

grid, and provide feedback control across the micro-grid to achieve optimal performance.

II. INTELLIGENT CONTROL AND MANAGEMENT ARCHITECTURE OF A MICRO-GRID

[00384] A micro-grid is a network of electric devices connected by conductor lines. A device is attached to a bus and connected carrying loads, generating units, or a connection to another grid. Our micro-grid architecture includes both controlled and uncontrolled devices. A micro-grid diagram is shown in Figure 1 included inline below. Controlled devices, as shown in Figure 1, are directly connected in a feedback configuration to an intelligent control element, called an Element Controller. The purpose of these control elements is to drive the grid to a desired dynamic behavior. This desired behavior is effected by control actions generated by these intelligent control elements as a function of sensory data measuring the status of the devices in the micro-grid, expressed desired behavior, a partial dynamic model of the grid and a corpus of behavior generated by active, inductive learning from the sensory data.

[00385] To a significant extent, this paper is concerned with establishing an implementable technology for micro-grid intelligent control. Why does a micro-grid need intelligent control? We explore four reasons. 1) Conventional optimal control techniques require an accurate model of the dynamic behavior of the loads, such as an air conditioner, or a hospital MRI, in the micro-grid. The main problem that we face today is that modeling the loads at sufficient accuracy is impractical. Further, many current loads such as battery chargers, high temperature crystal growing,

MRI machines and others have nonlinear and discontinuous dynamics. We propose a methodology based on machine learning, load forecasting and rule-based optimization to identify the dynamic behavior of the loads using sensory data in real time. The methodology we propose includes a component that extracts structural characteristics of the dynamics from historic and current data, and improves the accuracy of the model as time goes on. Thus the response of our intelligent control system also improves over time. Further more, as the system evolves due to changes in the load profile or generation profile, our intelligent control system adapts to those changes through learning.

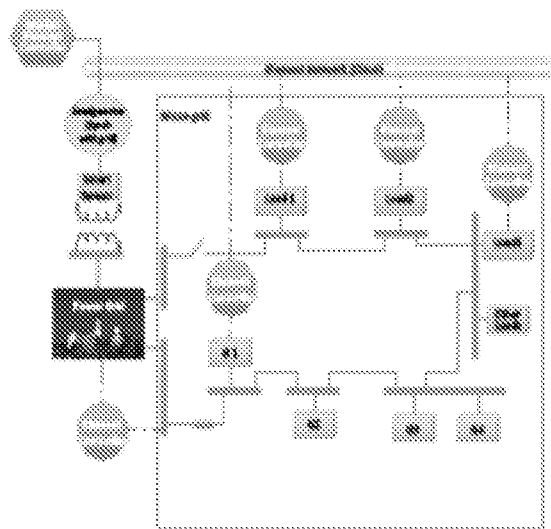


Fig. 1. A micro-grid system.

2) Micro-grids are distributed dynamical systems. In the past, micro-grids were designed with sufficient excess capacity to satisfy peak demand with high probability. For economic reasons, we can no longer afford to waste this excess capacity. Therefore, the distributed control must accurately match the variability of the demand with the multiple sources of generation so that peak demand and base demand are satisfied. For micro-grids, a centralized control design cannot provide quality feedback control to the multiplicity of active devices of the micro-grid. In our distributed control, local intelligent feedback controllers are connected to form a collaborative network that manages the control action for each active device. Furthermore, this network includes a micro-grid management server that provides optimization criteria, and operational target settings for each element controller.

3) Micro-grids exchange energy with the environment and are open systems. The nature of this bi-directional exchange is stochastic, and the dynamics of this stochasticity has to be learned from observations. This exchange requires the intelligent control to provide active synchronization control. This active synchronization control between the network of element controllers and the outside grid management system is essential to maintain high quality of service in a cost-effective manner.

4) Micro-grids are distributed heterogeneous systems. We apply Tellegen's theorem [20] that states that in solving distributed control by a network of agents, we can regard an agent controlling a subsystem as interacting with an aggregated agent, representing an average control for the micro-grid. The interaction between an agent and the average agent is formulated as an optimal control system subject to constraints. We model this average agent opponent using an method from particle physics called mean field. Several subsystems, each controlled by a separate intelligent control element, may have interrelated structures, such as physical proximity or similar structures, and are best analyzed as a lumped single system. A single local controller may best control a town, factory, wind-farm, or solar complex, so far as interaction with a larger global grid is concerned. What is extraordinarily useful in our model is that we can employ a "divide and conquer" strategy. We can interactively compute homogeneous subsystems and apply "divide and conquer" strategies to simplify control and speed up response.

A. Economic Justification

Renewable sources and non-traditional loads have the potential to improve the economics of electric power delivery. However, their characteristics differ substantially from standard power components, and new methodologies are necessary to integrate them in an economic and secure fashion. Because of their non-traditional dynamics, we have to control the micro-grid to achieve performance with regard to cost, resiliency, and quality of service. Catastrophic cascading power failures of interconnected systems are also a driver for implementing power delivery as a network of micro-grids.

Current software was designed to identify on the fly possible cascading failure modes anywhere in the system and to use this information to initiate isolation of failing systems and creation of a decomposition of working systems into independently operating subsystems. This was a perspicacious and laudable goal. But now a constantly increasing proportion of generated power comes from widely distributed non-traditional sources of very different operating characteristics. Power sources range from gas, coal, oil, nuclear and hydroelectric generating plants to wind, solar, and geothermal. The new disparate sources of large irregular variations are very highly distributed, and were not factors in the design of current system controllers for distribution of power. We see no way to use traditional centralized control, due to random combinations of these disparate power sources. A huge obstacle is figuring out the effect of a disturbance in one subsystem on a distant subsystem, and then communicating to the distant systems fast enough to disconnect, break into self-standing subsystems, and stabilize them. Another obstacle is how to choose subsystem decomposition for use when failures occur anywhere in the system.

This paper proposes a global distributed control system with no central arbiter. We formulate this system using measurements of deviations from "local invariants." These invariants (gauges) characterize what should be happening if

the subsystems are interacting properly. Large deviations from invariant behavior are what constitute the warning signals for initiating safe disconnection into subsystems. So disconnection is to be based on the size of deviations from expected measured invariant behavior of interactions of subsystems. That is, observed deviations from invariance can be used to drive the controllers that actuate separation of the global system into self-standing subsystems when subsystems collapse.

The formulas for these invariants and establishing what to monitor to compute them uses our hybrid systems methodology based on the constrained calculus of variations. We construct input-output equations for suggested self-standing subsystems. These are coupled by the connections between them, called a Tellegen decomposition [20]. These coupled equations are used as constraints. To make this feasible, the equations for each putative subsystem are obtained by an averaging method from particle physics. We design a cost function which evaluates the risks of catastrophic failure based on a given control scheme as a function of measurable states of subsystems. The controls for the system are chosen to minimize the cost subject to the constraints. With real time computation that we cannot describe here, failure to compute a controller of acceptable cost cause the separation of the system into independent subsystems. Simulation is to be used to test decompositions into subsystems for performance characteristics. There are many ways to break up the graph of interconnections of sources and sinks of power use into self-standing systems. Which are best remains to be analyzed.

B. Architecture of the Intelligent Feedback Control

Figure 2 illustrates our proposed architecture for the network of element controllers connected to the micro-grid management server. The convention we use in the figure is that the PSC boxes represent programmable signal conditioners that transform digital signals from a computer to device signals and vice versa. PSC's are crucial components for our intelligent control architecture and must be designed with care to effectively interface the intelligent control with the physical devices. In Figure 2, shaded wide arrows represent outside signals, and the thin arrows represent computer signals. The fundamental functions of the micro-grid management server are to convey the optimization criteria and goal settings for each of the element controllers, and to receive real-time data from the network. In Figure 2, the micro-grid management server abstracts data from the element network to use in learning, and makes it available back to the element network for restarting operations. The micro-grid management server also schedules the interaction with the element network, composed of multiple active element controllers.

The element controllers are the active participants in implementing the intelligence of the micro-grid. They implement conventional feedback control, inductive active learning and intelligent feedback control. The intelligent feedback control involves the implementation of rule based control laws that are partially learned from the sensory data.

They control one or more devices whose dynamic load depends on variations in the environment, or in the operation of the device. An example of the first kind is an air conditioning/heating unit; an example of the second kind is equipment such as a crystal-growing furnace that draws loads from 10 kilowatt hours to 100 kilowatt hour during its operation. Their inputs and outputs are illustrated in the bottom left of Figure 2. The sensory signals and actuator command signals are not routed through the element network, to increase reliability of the element. If the element controller is disconnected from the network due to unwarranted interruption, it will continue functioning as a conventional feedback loop that maintains the active device functionality and issues an alarm to the micro-grid management server. For this reason, our design separates the sensor and actuator signals from the communication bus.

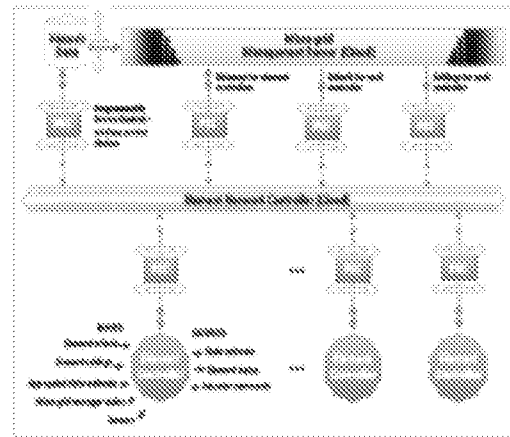


Fig. 2. Overview of our architecture for intelligent control and management.

A detailed block diagram describing the functional blocks of the element controller is illustrated in Figure 3. The architecture of the element controller, for the most part, can be implemented using digital signal processing [21]. The control functionality of the element controller is realized through five feedback loops:

- 1) Conventional feedback loop
- 2) Inductive active learning loop
- 3) Synchronization loop
- 4) Intelligent controller loop
- 5) Amalgamator controller loop

These loops are coupled and they synchronize to achieve the overall desired functionality.

1) *Conventional Feedback Loop*: This loop (connecting the Controller, Actuator Model, Dynamics Model and State Estimator with inputs from Sensors, through a PSC, and Element Settings and Criteria, and outputs control action through the Actuators) implements the classical actuator-sensor control. In most operations, it is implemented through one or more PID's. The loop is designed to fulfill requirements such as rise-time, overshoot limitation, phase and gain margins, and bandwidth signal-to-noise ratio specifications. Because we expect the sensory signals to be affected by

environmental noise, the state estimator is designed to generate estimates that respond to this uncertainty.

2) *Learning Loop*: The inductive active learning loops add an Inductive Learning Controller to adapt the Controller, Dynamics Model and State Estimator with the Dynamic Element Rules. Most installations that are already in operation will have an existing conventional loop, so the other loops in the element controller will be compatible with the existing loop. For new installations, the element controller is designed in a modular fashion, so it is possible to modify loop components as necessary. An example of the need for a learning loop is a crystal furnace in which the thermal control is well-understood and implemented through a conventional feedback loop, but the magnetic confinement dynamics of the plasma is exceedingly difficult to model with conventional control techniques. However, the combination of sensory data with inductive learning and Maxwell equations expressed with rules may provide a dynamic model that is locally valid and can be used to design on-line modifications to the controller so that the resulting feedback loop produces the desired behavior.

This example points out the need for inductive learning as a means to implement effective feedback control when the dynamics can only be partially identified from historic sensor-actuator data. We construct a local model in real-time using our inductive active learning technique. Our inductive active learning technique constructs operational rules that characterize the dynamics model of the device under active control (see [17]). The learning loop illustrates the functionality of the learning loop. It modifies the state estimator, the dynamics model, and the controller to minimise error criteria between the actual sensor-actuator data and the modeled sensor-actuator estimates. One of the fundamental hard rules in the inductive active learning loop is to ensure that no changes to the model will lead to unstable behavior on the micro-grid. This requirement implies that the inductive active learning controller must include constraining rules of adaptability authority of parameters and structures of the controller, dynamics model and state estimator that ensure input-output stability for the micro-grid.

Awareness of the effect of the device on the micro-grid via "awareness rules" allow the learning loop to participate in a synchronization strategy with the amalgamator to achieve the desired micro-grid behavior.

Knowledge uncertainty is unavoidable, even in simple grid systems. The structural modifications of any one of the controller, dynamics model, and state estimator, have one of the fundamental purposes to improve the quality of the models. This requires increasing the input-output matching of the impedance of the model with the actual impedance of the device, and thus reducing structural uncertainty. In general, we will represent impedance with an operator, because, in the micro-grid domain when non-traditional elements are present, single frequency (AC) representation of impedance is not enough to characterize the dynamics of the device.

3) **Synchronization Loop.** When multiple active devices form part of a micro-grid, the classical approach is to introduce inter-device constraints to the specification of each

of the active controllers. This requires extensive modeling and is specific to each micro-grid. Our approach is substantially different. We use a concept developed in particle physics in which the interaction of a single particle with the rest of the particles is mediated by an “average mode” called the mean-field approximation. In our case, each element will be interacting with the rest of the elements in the micro-grid by participating in a dynamic synchronization with the mean field element: the controller of the element will participate in a synchronization event with the amalgamator controller representing the rest of the micro-grid.

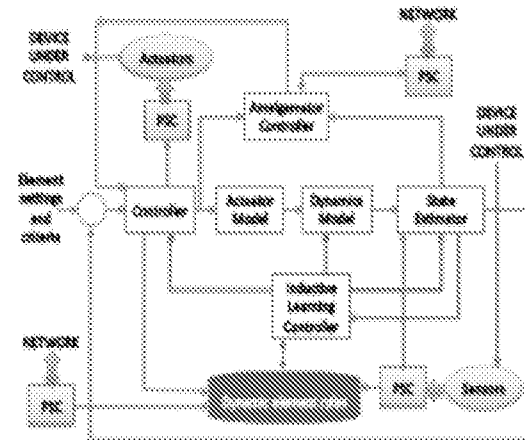


Fig. 3. Element controller arbitrary time

The essence of this synchronization involves the construction of a variational model of the device under control representing the aggregated micro-grid. The synchronization controller reaches and maintains a steady interaction when an invariance (gauge) condition between the controller dynamics and the amalgamator dynamics is attained and maintained. When external events occur and this invariance condition fails, the interaction between the controller and amalgamator controller changes the control signal to reacquire invariance. The information needed to build and maintain the variational model of the amalgamator requires data from the network. The network data is interpreted by the amalgamator as sensory data. The amalgamator constructs an aggregated state estimate projection to the space of the device under control. The aggregated state estimate projection is computed from the micro-grid state and the element state.

The synchronization loop is affected by uncertainty in the controller, the state estimator, and the amalgamator controller. We impose a quality synchronization level, high enough to guarantee satisfactory operation, measured in terms of the spread around the mean field equilibrium.

4) *Intelligent Controller Loop*: The intelligent controller loop implements the learning loop procedure, the compatibility procedures with the synchronization loop, and with the conventional loop. It involves the flow of signals for knowledge acquisition, and the generation of incremental updates to the device model, controller, and state estimator to respond to dynamic events. The mechanical aspect of the intelligent loop procedure is unique to us. Classical learning

loops are query-answering procedures implemented through inference mechanisms such as *modus ponens* and rule-chaining strategies. Our approach involves the transformation of the active rules in the dynamic element rule repository into constraints and the transformation of the current query to optimization-tracking criteria.

In a network of classical devices, loads and generators, the Kirchhoff and Ohm's laws accurately describe the energy flow and state of the devices in the network. Computing dynamic currents and voltages in the micro-grid can be achieved by extracting the Lagrangian or equivalently the network Hamiltonian that are used in the element controllers. However, with non-classical devices or sources that are pervasive in a micro-grid, network Hamiltonian and the Hamiltonian devices are almost impossible to determine by classical network procedures. Therefore, we use rules to characterize the dynamic behavior of the system. The rules that characterize the dynamic behavior of non-classical components are of three different types; they may come from physical principles, desired operational behavior for the multi-objective optimization, and/or from sensory observations.

The dynamic element rule repository is composed of three types of rules: absolute rules, hard rules, and soft rules. Absolute rules represent operational principles that must be satisfied for the operation of the device under control. Hard rules are rules that must be satisfied for adequate performance, and soft rules are acquired by inductive active learning during the operation of the micro-grid or from phenomenological or empirical observations. In our approach, soft rules are transformed to inequality constraints, while absolute and hard rules are transformed into equality constraints. The truth-value of the rules is translated into numerical binary values: 0 for false, and 1 for truth. Soft rules have values that can be scaled into a closed $[0,1]$ interval.

Soft rules transform the problem from a pure logical inference to a hybrid one involving both inference and probabilistic extraction of control laws for the control elements that are in synchrony with the other control elements. The synchrony is established continuously with respect to the amalgamator control law. In a recent paper [15] we showed that this provides active robust synchronization of the element with the other control elements and the grid management system (see Figure 2).

During operation, only the rules that are active are transformed into functional constraints. This is essential for real-time computational purposes. An absolute or hard rule is active when it is violated (value of 0). A soft rule is active when its truth-value is below a pre-agreed threshold. The threshold may vary among soft values. Soft rules that are learned through inductive learning include the functional structure and the logic value. The chaining of the active rules is established by the interaction in the functional representation. The inductive learning loop is realized dynamically by constructing a Lagrangian that adjoins the functional relations of the active rules to the criterion. The off-diagonal elements of the Hessian of the Lagrangian associated with the optimization problem represent the chaining of the

corresponding active rules.

The intelligent controller implements a discovery engine (an automated search engine) for extracting the active rules as a function of sensor and actuator signals. It also generates new rules using a repair mechanism that forms part of the inductive learning loop procedure. This repair mechanism is unique to our approach [7]. It operates by adding corrective increments to the control signal as a function of detected operational error. This is one of the fundamental aspects of the learning capabilities of our intelligent controller.

3) *Amalgamator control loop*: The amalgamator controller, appearing as a box in Figure 3, implements the projection of the micro-grid dynamic model into the domain of the element controller. A more detailed architecture of the amalgamator controller is illustrated in Figure 4. The purpose of the amalgamator is to establish explicitly the constraint of the device under control with respect to the micro-grid. The unique aspect of our approach is to provide effective synchronization and reacquisition of synchronization using an optimal synchronization strategy between the amalgamator and the controller.

It is important to address the issue of how to characterize uncertainty. In most implementations of smart grid control and management, uncertainty is described by assuming a probability distribution on the state. We consider this unsatisfactory because the probability distribution is unknown and does not satisfy the Hadamard condition, meaning that the distribution should be continuously dependent on the observed data. Our approach is to represent uncertainty on the state, the structures, and the signals with means and spreads that are continuously computable from the data without assuming any probability distribution. This approach requires the propagation of sets representing the states and the structures. This method is referred to in the literature as differential set inclusion.

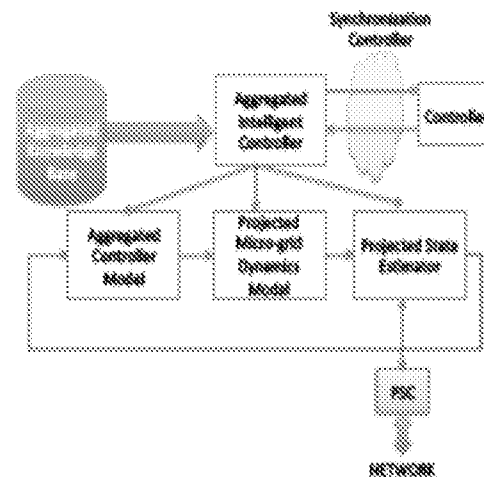


Fig. 4. Amalgamator controller architecture.

C. The Tomograph for Estimating the Lagrangian via Inference

Given a micro-grid, a tomograph is used to construct the

Lagrangian for non-standard elements in the network of the micro-grid. We characterize the micro-grid with a Lagrangian that is the difference between the energy and potential field,

$$L(x, i) = E(x, i) - V(x)$$

where the sum for the k th element is the current flowing the element in the network,

$$x_k(t) = \int_0^t i_k(\tau) d\tau.$$

The energy and potential functions are obtained from all the elements in the micro-grid,

$$E(x, i) = \sum_k E_k(x, i)$$

$$V(x) = \sum_k V_k(x)$$

due to conservation of energy. The Lagrangian model of circuits is well-known. For standard elements, such as capacitors, inductors, transformers, classic generators and classic loads, the relationships between current and voltage across the ports are known. However, the Lagrangian for a rule-based system must be learned. The relationship between current and voltage across the ports for non-standard elements, such as wind turbines, solar panels and smart loads, are only partially known and must be estimated from sensory data.

We use a dynamic tomograph [2, 18] and an identification process to improve our estimate of $E_k(x, i)$ and $V_k(x)$ for each k corresponding to a non-standard element. The tomograph provides functions of the form,

$$E_k(x_k, i_k, \text{other states})$$

$$V_k(x_k, \text{other states})$$

so that we are building the Lagrangian "on the fly". This is what we refer to as *structural adaptation*. Once we have an estimate of the Lagrangian, we construct the estimated Hamiltonian,

$$\hat{H}(x(t), p(t)) = p(t)^T x(t) - L(x(t), i(t))$$

where $p(t)$ is the momentum in the micro-grid.

D. Hamiltonian Dynamics

Once we have the estimated Hamiltonian, we used it to construct the control Hamiltonian,

$$H_c(x(t), p(t), u(t)) = \hat{H}(x(t), p(t)) + V_c(x(t), u(t))$$

where $u(t)$ is the control signal, and $V_c(x(t), u(t))$ is a known potential function. The rules that reside in the "dynamic element rules," and the Hamiltonians of individual devices are maintained in the "inductive learning controller" (see Figure 3). The Hamiltonian model,

$$H_c(x(t), p(t), u(t))$$

is constructed from the rules, physical principles, historical data, and adapted to current sensory data. The procedure in the inductive learning controller obtains the optimal control law by constructing the state equation (1), co-state equation (2),

and meta-control equation (3), shown as follows:

$$\frac{dx(t)}{dt} = \frac{\partial H_c}{\partial p} + \sum_i \lambda_i \frac{\partial \phi_i(x, p, u)}{\partial p} \quad (1)$$

$$\frac{dp(t)}{dt} = -\frac{\partial H_c}{\partial x} - \sum_i \lambda_i \frac{\partial \phi_i(x, p, u)}{\partial x} \quad (2)$$

$$\frac{d\lambda(t)}{dt} = -\left(\frac{\partial^2 H_c}{\partial u^2} + \sum_i \lambda_i \frac{\partial^2 \phi_i(x, p, u)}{\partial u^2} \right)^{-1} \left(\frac{\partial H_c}{\partial u} + \sum_i \lambda_i \frac{\partial \phi_i(x, p, u)}{\partial u} \right) \quad (3)$$

The algorithm represented by equations (1), (2), and (3), is implemented with efficient numerical integration routines. In most cases, we use Runge-Kutta of the 3rd order, specifically modified for these equations [12, 15, 22].

If the costate equation

$$p(t) = \frac{\partial L(x(t), i(t))}{\partial i} \quad (4)$$

is regular (Dirac), then we have a unique solution

$$i(t) = f_i(x(t), p(t)) \quad (5)$$

but if the Hessian of the Lagrangian is not invertible (not regular Lagrangian), then from the Inverse Function Theorem, we cannot solve i in (4). This case is likely for our rule-based system. Then we separate the state vector into two parts; the physical state components and the constraint state components. For every device in the network, energy must be conserved, so the flow (current) across the device is a function of voltage and is expressed in a Hamiltonian system.

The Inductive Learning Controller takes the rules from the dynamic element rules repository, and transforms the logic rules, denoted $\phi_{\text{logic}}(x)$ and $\phi^{[i]}$, into continuous functions,

denoted $\tilde{\phi}_{\text{logic}}(x)$ and $\tilde{\phi}^{[i]}$. The x vector contains the state estimate and the variables associated with the rules. The Lagrangian $L(x, \tilde{\phi}_{\text{logic}}(x), \tilde{\phi}^{[i]}, u)$ is estimated using the

continualized rules and the method of Lagrange multipliers. The u vector represents the level of unsatisfaction of the soft rules. In order to minimize the Lagrangian, the rates of $\tilde{x}(t)$ and $\tilde{M}(t)$ are computed, where $\tilde{M}(t)$ is a positive semi-definite matrix used to estimate the inverse of the Hessian of the Lagrangian. For details, refer to [13] and [18]. Now the dynamics of the device is formulated as $G(x, \tilde{x}, x)$ and the Hamiltonian $\hat{H}(x, p)$ can be computed. Thus we have estimated the Hamiltonian for a non-classical device from rules! Given the Hamiltonian, we update according to changes in the system. Then we use our meta-control approach, see [12], to optimize the Hamiltonian and determine optimal control actions. This provides the feedback control loop for the Inductive Learning Controller.

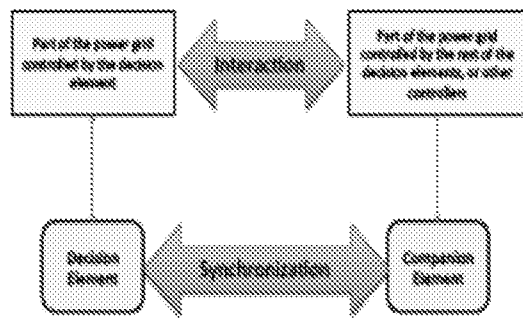


Fig. 5. Element synchronization via Tellegen.

III. MICRO-GRID CONTROL EXAMPLE

The micro-grid intelligent control and management system is applied to a micro-grid with three generators, two solar and one gas turbine, shown in Figure 6. The load is an air conditioning unit for an office building with varying requirements for air conditioning based on historical weather conditions measured by sensors. A simulation was run for 10,000 minutes, with historic data for all sensor data. The relative cost of energy at the load for the controlled micro-grid, with respect to the open loop uncontrolled system, is shown in Figure 7.

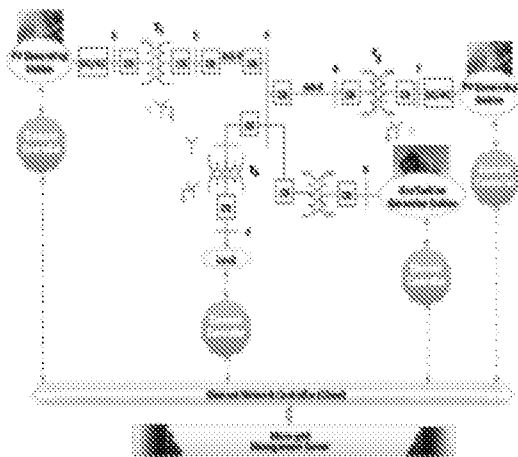


Fig. 6. Micro-grid example with three generators, two solar and one gas turbine, and an air conditioning unit as the load.

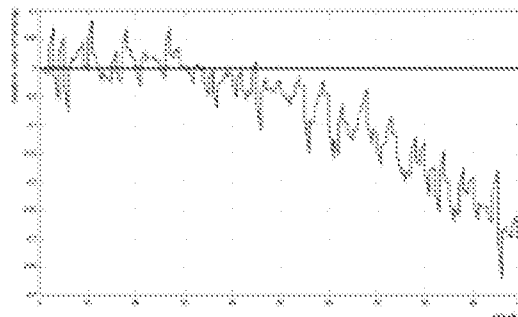


Fig. 7. Relative cost of energy at the load of the controlled micro-grid, compared to the uncontrolled system.

At time zero, in Figure 7, the relative cost of energy at the load compared to the open loop system is zero. In the early minutes, the element controllers are actively learning the variations due to weather impact on the load and solar panels. At 10 minutes, the element controllers begin to improve performance due to the constructed Hamiltonian models. The cost savings decrease to 100 minutes, as shown in Figure 7, and stabilize at approximately 30 percent saving for the rest of the time horizon.

The simulation demonstrates the value of synchronizing the generation facilities with the varying load, and adjusting the scheduling of their contribution to the load dynamically. A significant portion of the savings were realized by shutting down the gas turbine for periods where the demand for air conditioning was slight.

IV. CONCLUSION

The proposed architecture allows distributed control of devices with only local models built by active learning on sensor information. Practical development of models that are valid for the entire range of operation are non-existent and difficult to approximate. Our approach of using local models that are dynamically updated is practical. The distributed nature of our control (at the edge) increases the robustness, resiliency, and cost-effectiveness of the micro-grid.

REFERENCES

- [1] Dimas, A. L., and Hatziargyris, N. D., *Operation of a Multiagent System for Microgrid Control*, IEEE Transactions on Power Systems, Vol. 20, No. 3, 1447-1455, 2005.
- [2] Helgason, S., *Integral Geometry and Radon Transforms*, Springer Science+Business Media, LLC, 2010.
- [3] Jia, H., Chen, W., and Rong, X., *A review on distributed energy resources and MicroGrid*, Renewable & Sustainable Energy Reviews, 12, 2472-2483, 2008.
- [4] Jimeno, J., Anduega, J., Oyarzabal, J., and de Muro, A. G., *Architecture of a microgrid energy management system*, European Transactions on Electrical Power, 21, 1142-1158, 2011.
- [5] Keyhani, Ali, *Design of Smart Power Grid Renewable Energy Systems*, Chapters 5 and 6, John Wiley and Sons, Inc., Hoboken, NJ, 2011.
- [6] Kolm, W., Brayman, V., Cholewinski, P., and Nerode, A., "Control in hybrid systems," *International Journal of Hybrid Systems*, Vol. 3(3), pages 109-150, 2003.
- [7] Kolm, W., Brayman V., Littleton, J., "Repair-control of enterprise systems using RFID sensory data," *IEEE Transactions*, Vol. 37, pages 281-291, 2005.

[00386] The following describes another example use of the CDD system with respect to controlling a micro-grid electrical generation facility.

[00387] The frequency tracking hybrid optimal control system of this example constructs the dynamics from historic data including the load signal from the utility, and signals of the sensors in the devices. The model is used to generate a feedback control law that generates adjustments of the control of the battery, the inverter, and the solar panels. The control system includes a demand forecasting system, which estimates the state of the system from historic and current data from the tracking signal, as well as empirical facts input by the users. The forecasting system will attempt to optimize the demand forecasting extraction by maximizing the residual entropy. The demand forecaster will also estimate the uncertainty level of the forecast.

System Controller

[00388] The controller of the system will generate a strategy for optimizing revenue based on the physical characteristics of the distributed power system, the rules for quality, longevity of the batteries, and contract satisfaction with the utility. The controller will determine the optimal State of Charge strategy using a revenue utility criterion functional.

[00389] In order to achieve the general behavior described above, we program our controller to maximize revenue by minimizing the Total Frequency Regulation Revenue over a variable time horizon. This optimization will give different results depending on the time horizon selected.

[00390] The constraints will include:

1. Energy Consumption Balance Constraint over a time horizon (based on battery characteristics);
2. Cost of Energy Consumption Balance Constraint;
3. Demand Tracking per Site Threshold Constraint;
4. Cycle Limitation Constraint to maintain quality of the battery;

5. Interconnection Budget Constraint between Battery and Solar PV;
6. State of Charge Limitation Constraint;
7. Performance Score Threshold Constraint;

In addition to the constraints, we will provide a synchronization constraint for delivering distributed frequency targeted demand.

Forecaster

[00391] We program our generic forecasting engine to predict:

RegD Signal

Frequency Regulation Market Price

Battery Aging (in “real time”)

[00392] Our forecaster includes a Parameter Adaptation Engine for tuning the parameters, and a Learning Algorithm based on repair for improving monotonically the local models generated in the forecast and in the controller over time.

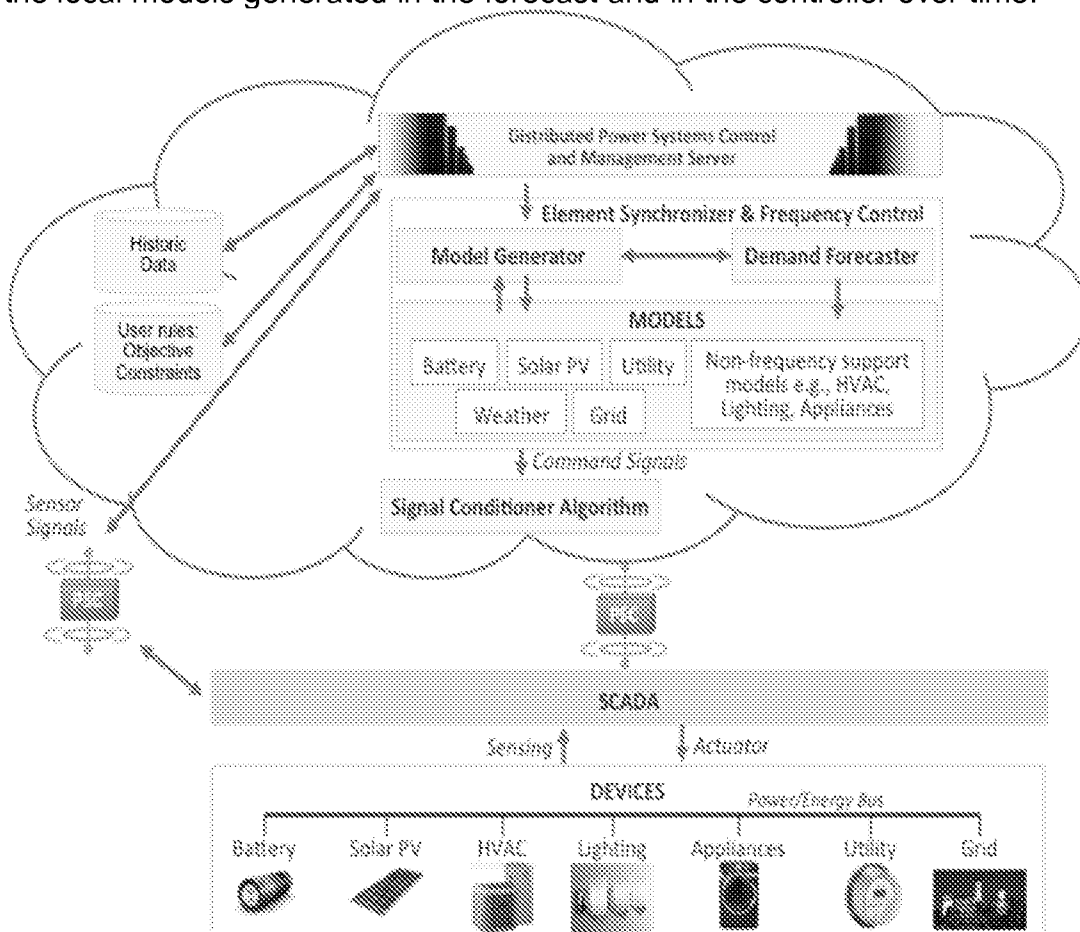


Figure AK

Technology Overview

[00393] Figure AK implements the programmable Hamiltonian controller. It generates a local Hamiltonian model based on state estimates (forecast signal) from the Hamiltonian programmable forecaster (see Figure AL). It also uses sensory data to generate the Hamiltonian of the system via a tomographic algorithm. The Hamiltonian is used to determine the local dynamics of the state, costate, and control system. Notice that the controller responds to the utility signal continuously (with two second updates). The hard rules, which represent the battery longevity empirical principle and the interaction with the inverter and the solar panels, are encoded in the controller data base, and translated into functional forms. The absolute rules are always active and they represent the safety margins for the battery, the response to the utility contract, and the stability of the system. The soft rules are determined from the optimality condition and from the tomograph.

[00394] Figure AL illustrates the programmable Hamiltonian forecaster. The Hamiltonian estimate computed in the controller is input to the forecaster. The system also uses historic data to tune up parameters of the system. Its central objective is to generate the conditional probability density of the state, as a function of time using a Dirac bracket. The probability density of the state is used to compute mean values of the state, costate and control variables, which are fed to the programmable Hamiltonian controller (see Figure AK).

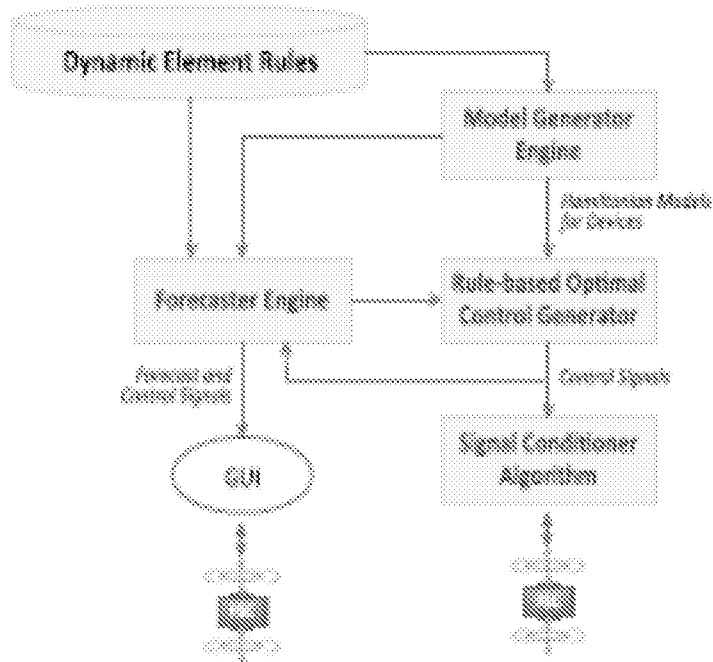


Figure AL

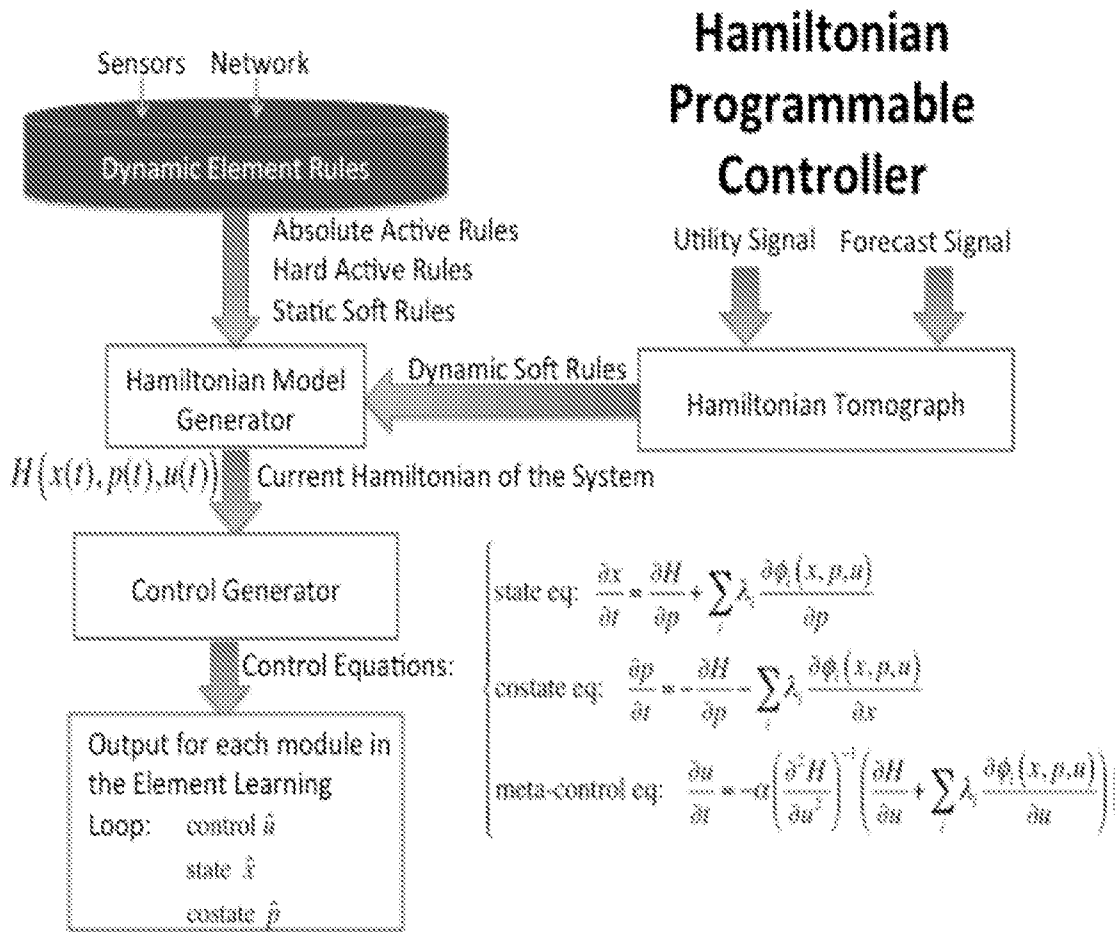


Figure AM

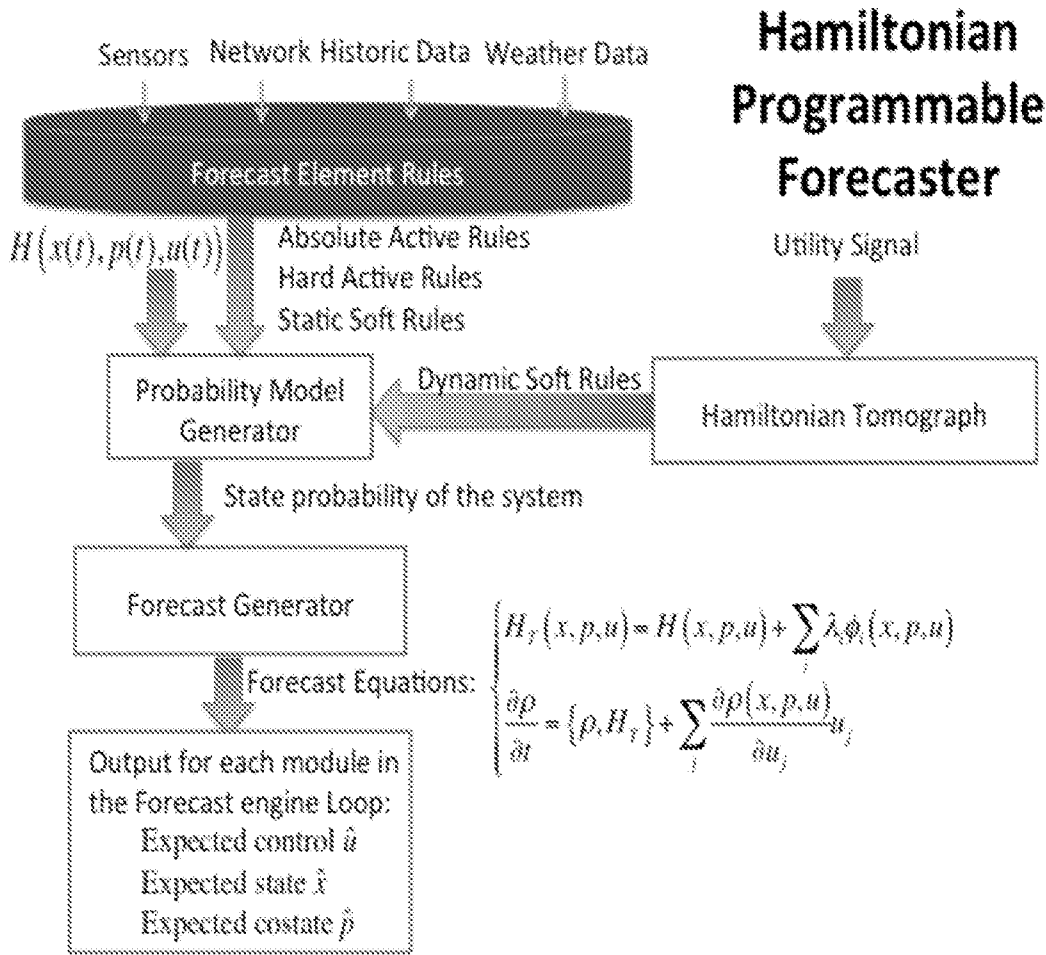


Figure AN

Solution Architecture

[00395] The proposed solution is composed of an integrated software platform. The software is centrally deployed either on a secure cloud or an on premise server cluster.

[00396] Figure AO details the central software components that comprise the overall solution. The only external interfaces of this centrally deployed package is a user interface for human operators, a software interface that streams data into the system, and another SCADA software interface through which the system effects the controlled elements. The existing SCADA system is responsible for directly communicating with the controlled elements, for both sensors and effectors.

[00397] The software components depict how internally different software components are used to implement the flows described in Figures AL and AM. The reusable analytics components include the core Multi-Criteria Optimization engine, Inferencing engine, and the ability to load and execute goals and rule sets locally. They also include the communications protocols that provide the integration with SCADA (external sensors and effectuators), as well as with the central management platform.

[00398] The software components on the left side on Figure AO enable human operators to configure and operate the system. In conjunction, they have three primary functions. First, they provide a user interface to define rules, goals and constraints across all controlled elements; second, it aggregates data from across the entire system, and provides view for real-time monitoring and troubleshooting; and third, it implements the “authorization to proceed” protocol, used when the rules dictate that a human decision is required.

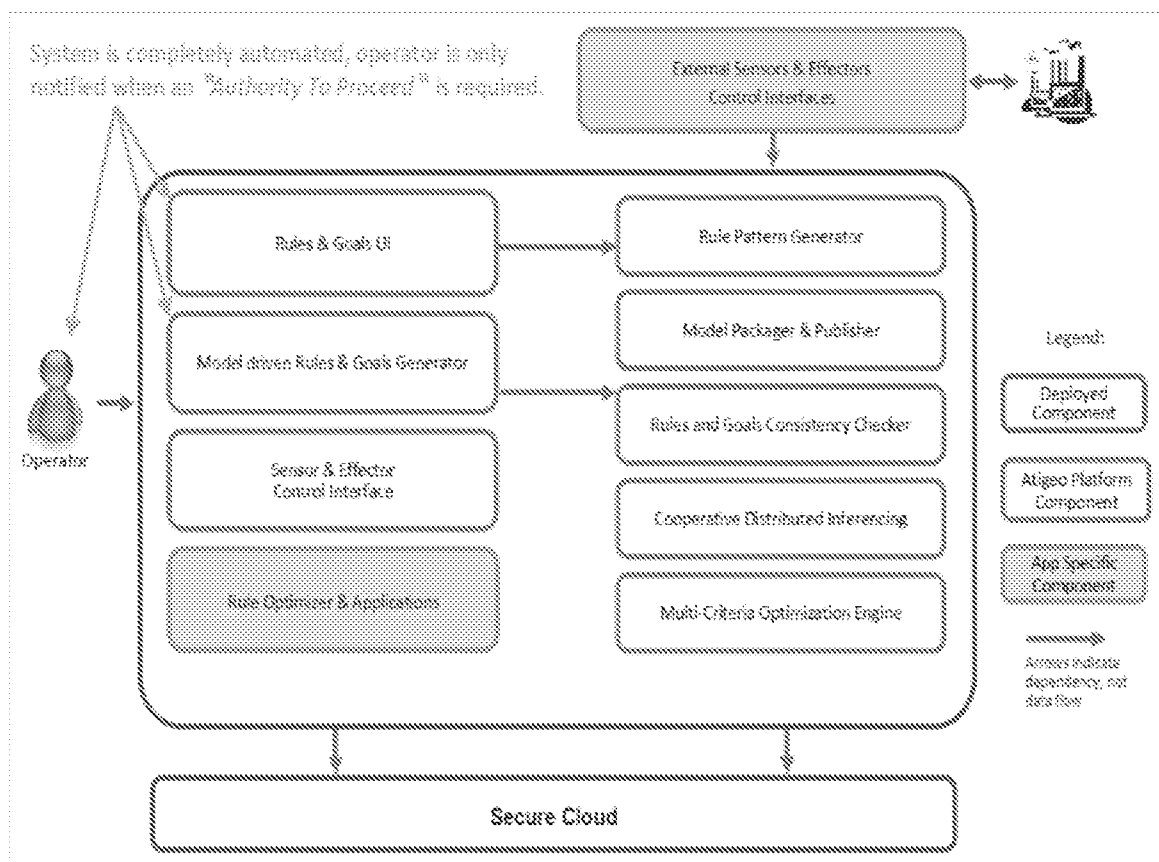
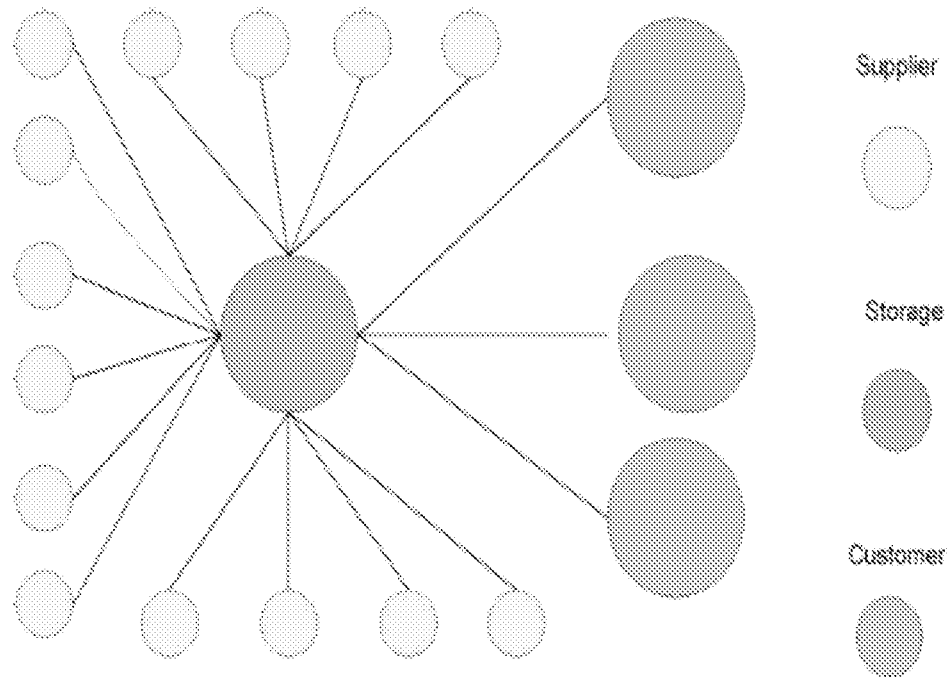


Figure AO

[00399] The following describes another example use of the CDD system with respect to controlling inventory management, such as for a retail location.

Actual Supply Chain Graph (with intermediate storage)



Sample Problem Statement

✦ Given:

- ✦ List of demands
- ✦ List of existing supplies
- ✦ List of suppliers for each item, with lead times, fixed & varied costs, min and max quantities per ordering
- ✦ Inventory carrying costs per item
- ✦ Penalty costs per unit item for not satisfying demand
- ✦ Days the item can be stored before perishing

✦ Decisions to be made:

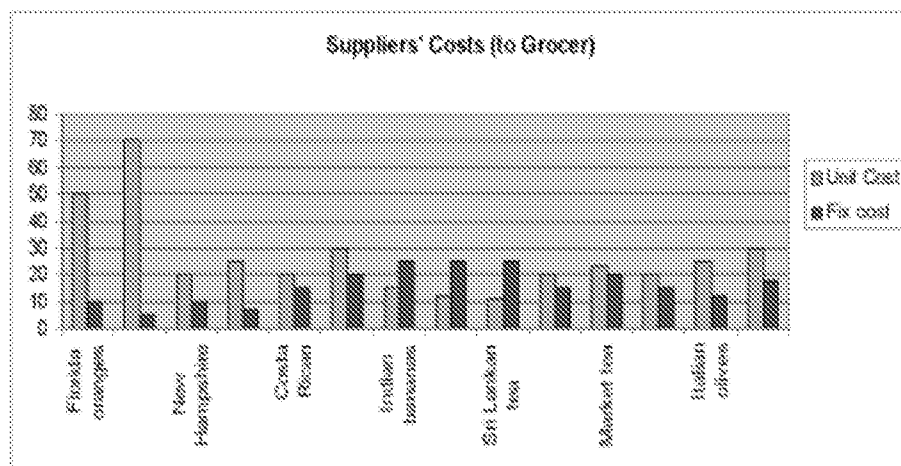
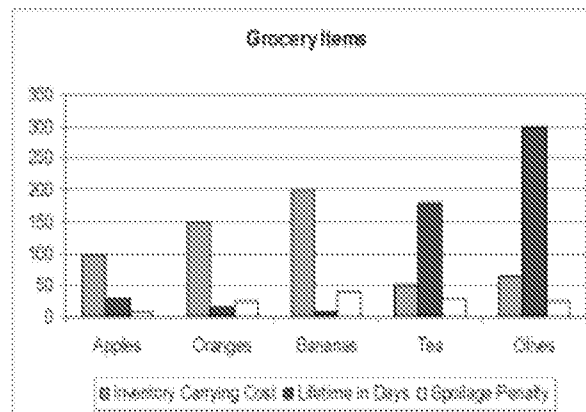
- ✦ Allocation of existing supplies to existing demands
- ✦ Ordering of new supplies from appropriate suppliers
- ✦ Objective: Minimize the total cost

Sample Application Problem Statement

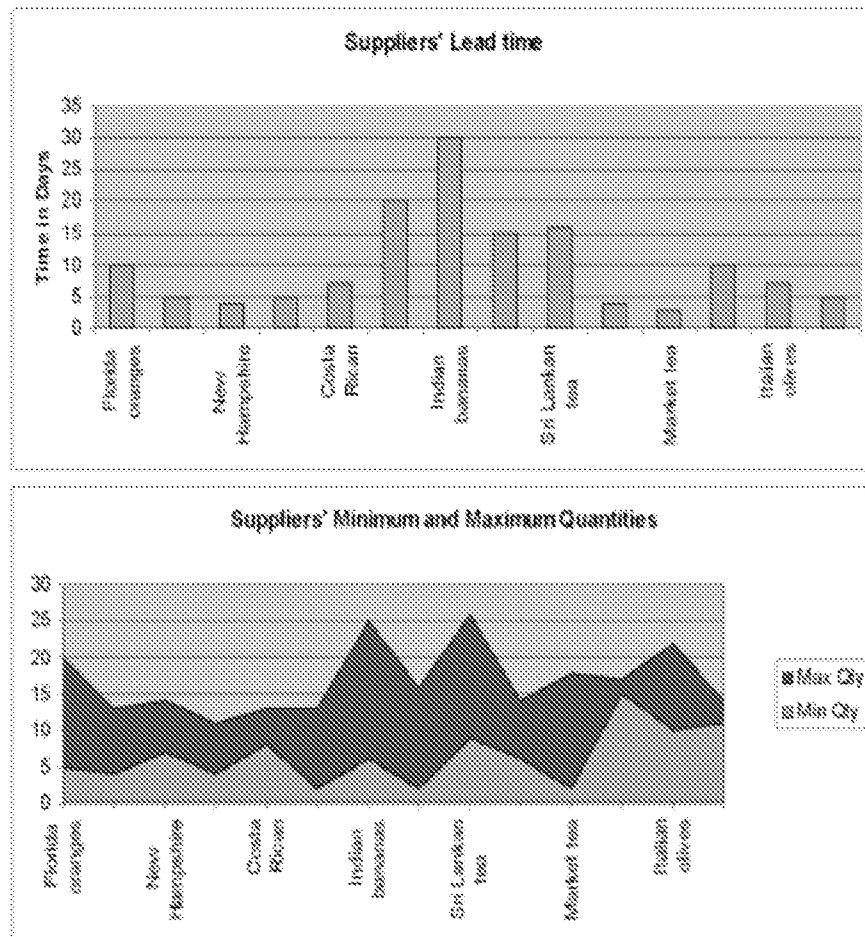
✦ Rules:

- ✦ The customer of type 1 always has to be satisfied
- ✦ The customer of type 2 has to be satisfied in at least 40% cases
- ✦ The customer of type 3 has to be satisfied in at least 25% cases
- ✦ Nothing can be ordered from a supplier within the lead time

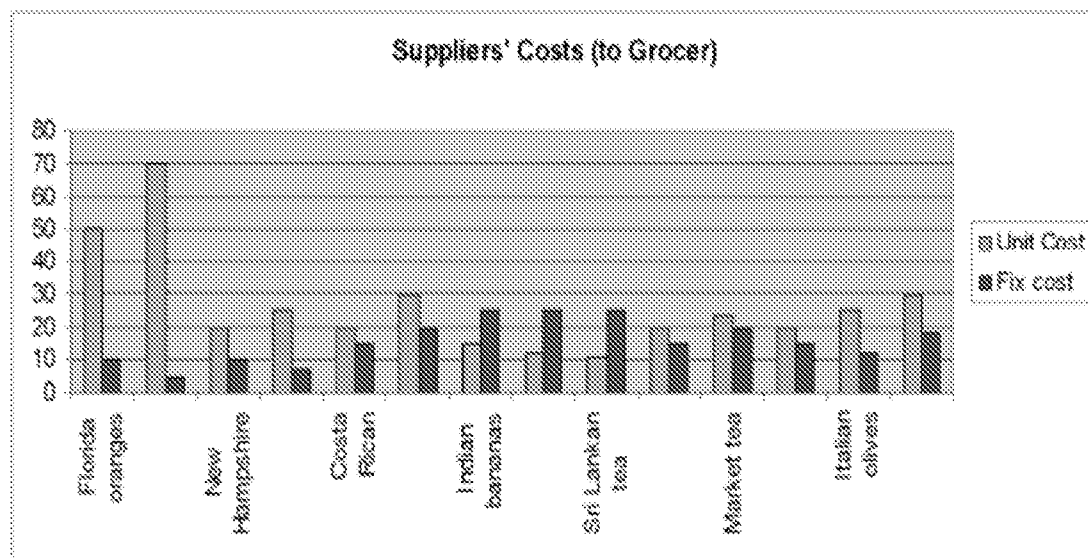
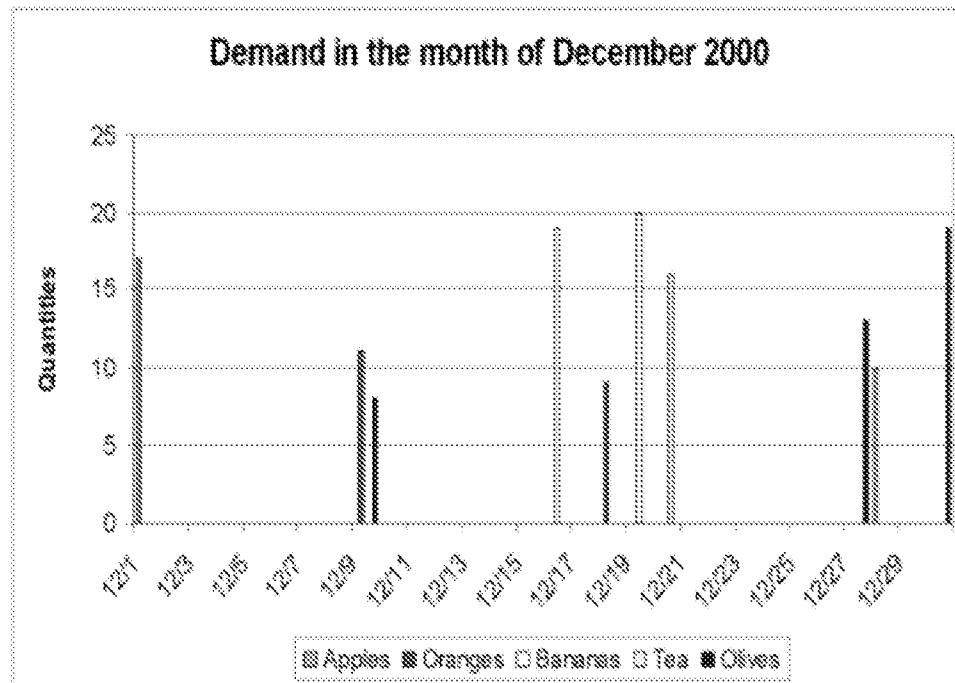
Sample Parameters

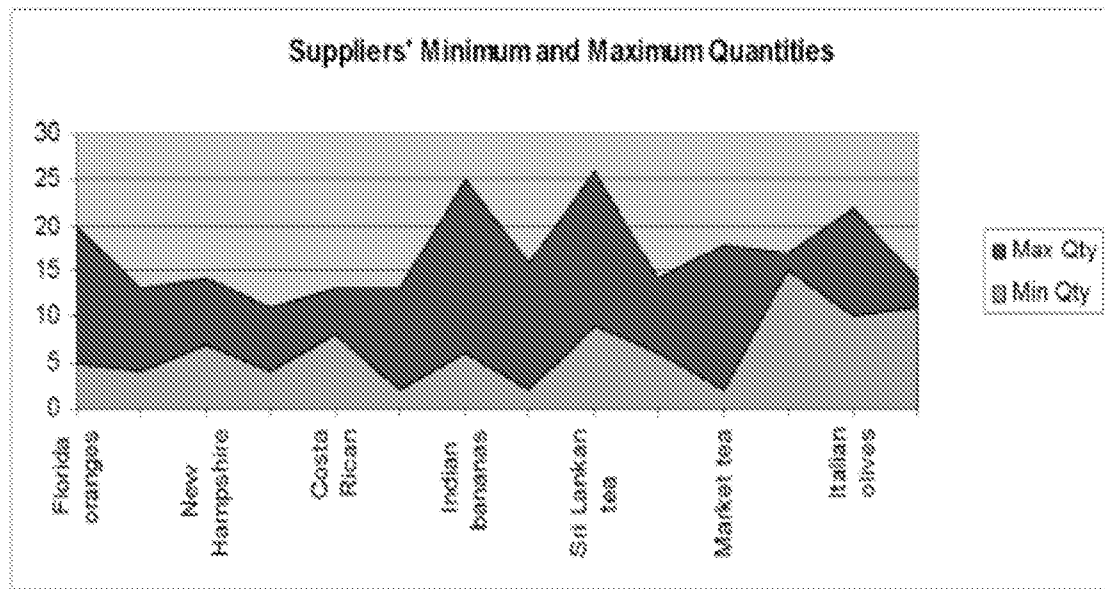


Sample Parameters

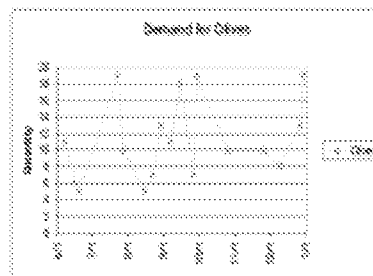
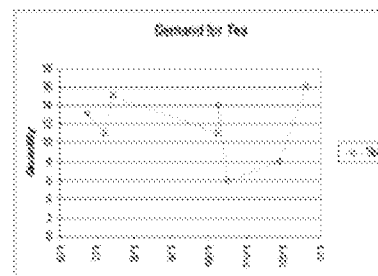
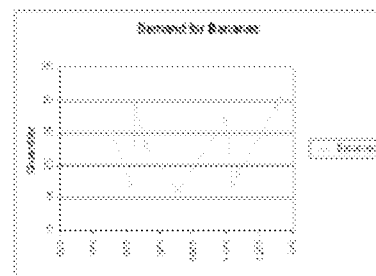
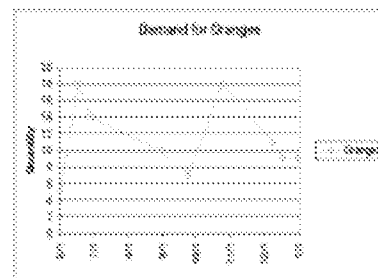
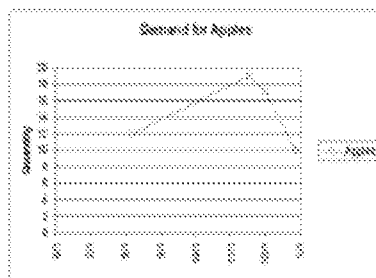


Demand by Month

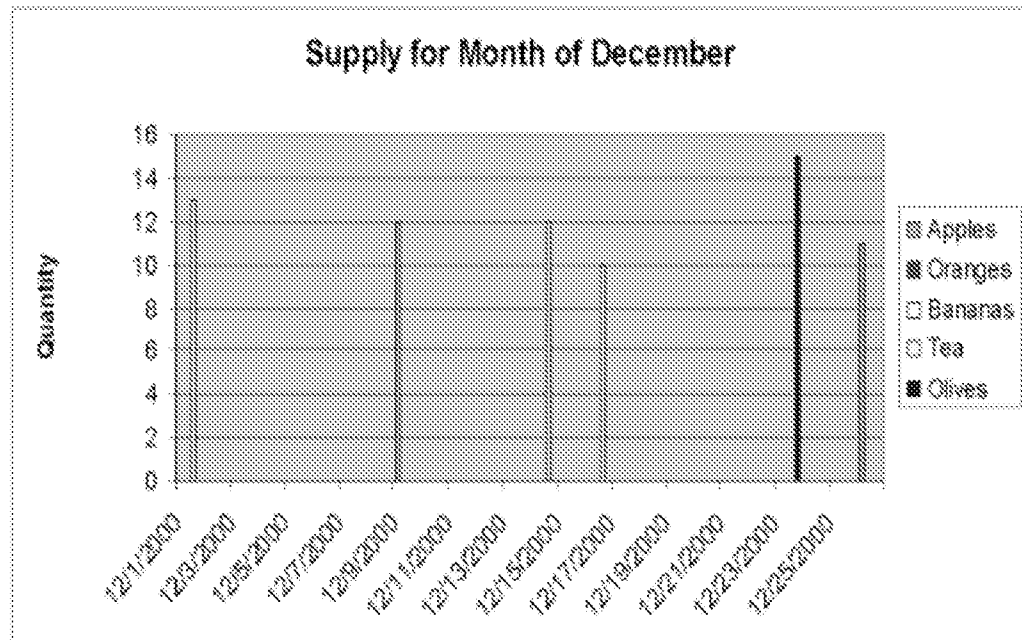




Sample Demand



Sample Supply



Sample Application Parameters

Part	Desc.	Inv. Carr. Cost	Life. Days	Penalty	
1	Apple	100	30	10	
2	Orange	150	15	25	
3	Banana	200	10	39	
4	Tea	50	180	30	
5	Olive	65	300	25	
Suppliers					
Item	Supplier	Lead time	Unit Cost	Fix cost	Min Qty
2	Florida	10	50	10	5
2	California	5	70	5	4
1	NewHampshire	4	20	10	7
1	Colorado	5	25	7	4
3	CostaRica	7	20	15	8
3	Italy	20	30	20	2
3	India	30	15	25	6
4	India	15	12	25	2
4	SriLanka	16	11	25	9
4	England	4	20	15	6
4	Market	3	23	20	2
5	Greece	10	20	15	15
5	Italy	7	25	12	10
5	Market	5	30	18	11
Note: No unit conversions necessary					
Note: All time units are in days					

Sample Application Run Time Data

Supply Item	Quantity	Date	Demand Custom	unit rev.	Item	Quantity	Date
2	11	6/1/2000	2	74.21	2	5	06/01/00
3	14	6/4/2000	2	95.58	5	11	06/07/00
4	10	6/5/2000	2	94.53	2	18	06/17/00
4	11	6/14/2000	2	97.72	5	6	06/17/00
4	15	6/16/2000	2	74.28	5	5	06/20/00
4	10	6/19/2000	2	118.23	4	13	06/23/00
1	14	6/22/2000	2	113.02	2	14	06/29/00
2	9	6/26/2000	1	105.69	4	11	07/07/00
3	9	6/28/2000	1	92.75	4	15	07/14/00
3	5	7/1/2000	2	82.19	3	15	07/18/00
5	9	7/10/2000	2	109.8	5	19	07/23/00
3	5	7/13/2000	2	101.3	5	10	07/28/00
3	7	7/18/2000	2	85.23	3	7	08/05/00
3	7	7/27/2000	3	119.74	1	12	08/06/00
2	13	8/3/2000	2	111.14	3	19	08/09/00
2	10	8/9/2000	2	100.87	5	5	08/15/00
4	6	8/12/2000	2	75.35	3	13	08/18/00
5	7	8/22/2000	2	87.77	5	7	08/22/00
5	7	8/26/2000	2	98.46	5	13	08/29/00
3	6	8/31/2000	3	93.05	2	10	08/30/00
5	9	9/5/2000	3	90.86	5	11	09/07/00
5	9	9/8/2000	2	118.33	5	18	09/15/00
2	6	9/9/2000	2	114.76	3	6	09/16/00
2	7	9/17/2000	1	95.22	2	7	09/24/00
5	13	9/25/2000	1	94.11	5	7	09/26/00
5	13	9/29/2000	1	79.96	5	19	09/29/00
4	8	10/7/2000	2	84.11	4	11	10/08/00
2	11	10/10/2000	1	99.88	4	14	10/09/00
5	11	10/13/2000	3	81.8	4	6	10/16/00
5	8	10/20/2000	2	78.01	2	18	10/24/00
5	13	10/20/2000	2	96.96	5	10	10/26/00
4	7	10/25/2000	2	76.47	3	17	10/31/00

Sample Problem Model

➤ Model Dynamics is composed of five elements:

- Market Dynamics Conservation
- Inventory Conservation
- Envelop Cost Conservation
- Polling Mechanism
- System Lagrangian

Supply Chain Dynamics: with Intermediate storage

➤ Market Conservation

$$\dot{Z}_i^j(t) = -\rho_i^j Z_i^j(t) + \theta_i^j(t) - v_i^j(t) \quad i=1,\dots,3 \quad j=1,\dots,5$$

where

$Z_i^j(t)$ is the unsatisfied demand of product j by customer i at time t

$\theta_i^j(t)$ is the continualized demand of product j by customer i at time t

$v_i^j(t)$ is the continualized delivery action of product j to customer i at time t

ρ_i^j algorithmic control constant

Supply Chain Dynamics: with Intermediate storage

Inventory Conservation

$$\dot{X}^j(t) = -\lambda^j X^j(t) + \sum_{k \in S_j} s_k^j(t) + \hat{u}^j(t) - p^j(t) \quad j=1, \dots, S$$

where

$X^j(t)$ is the inventory accumulation of product j at time t

$\sum_{k \in S_j} s_k^j(t)$ is the total rate of supply of product j from suppliers k at time t

$\hat{u}^j(t) = \sum_{k \in S_j} u_k^j(t - \Delta_k^j)$ is the cumulative ordering action of product j from suppliers $k \in S_j$ at time t

Δ_k^j is the lead time for product j from supplier k

$p^j(t) = \sum_{i=1}^3 v_i^j(t)$ is the cumulative rate of delivery of product j to customers i

Supply Chain Dynamics: with Intermediate storage

➤ Envelop Cost Conservation

$$\dot{C}(t) = -\mu C(t) + \sum_{j=1}^3 (\alpha^j \hat{u}^j(t) + \beta^j) + \sum_{j=1}^3 \gamma^j X^j + \sum_{j=1}^3 \sum_{l=1}^3 \bar{w}_l^j z_l^j(t)$$

where

$C(t)$ is the cumulative criterion at time t

γ^j is the inventory carrying cost per unit of product per unit time

$\alpha^j = \max_k \alpha_k^j$ is the envelop unit cost

$\beta^j = \max_k \beta_k^j$ is the envelop fix cost

$\bar{w}_l^j = \frac{w_l \delta^j}{\sum_{j=1}^3 \sum_{l=1}^3 w_l \delta^j}$ is the cost for not satisfying demand

δ^j penalty for not satisfying demand for product j

w_l is the relative weight for not satisfying the demand from customer l

Supply Chain Dynamics: with Intermediate storage

➤ Polling Mechanism: Quantizer

$$u_k^j(t - \Delta_k^j) = \begin{cases} \left\lfloor \frac{\alpha_k^j \hat{u}^{j*}(t)}{\sum_k \alpha_k^j} \right\rfloor & \text{if } u_{k \min}^j \leq \hat{u}^j(t) \leq u_{k \max}^j \\ 0 & \text{otherwise} \end{cases}$$

where

$u_{k \min}^j$ is the minimum quantity of product j that can be ordered from supplier k

$u_{k \max}^j$ is the maximum quantity of product j that can be ordered from supplier k

α_k^j is the unit cost of product j if ordered from supplier k

Agent Knowledge Model: with Intermediate storage

➤ System Lagrangian

State	Decision	External Drive
$\tilde{X}(t) = \begin{bmatrix} Z(t) \\ X(t) \\ C(t) \end{bmatrix}$	$U = \begin{bmatrix} \hat{u}(t) \\ v(t) \end{bmatrix}$	$S(t) = \begin{bmatrix} \theta(t) \\ \hat{s}(t) \\ \hat{\beta}(t) \end{bmatrix}$

Continualized Model $\dot{\tilde{X}}(t) = A\tilde{X}(t) + BU(t) + S(t)$

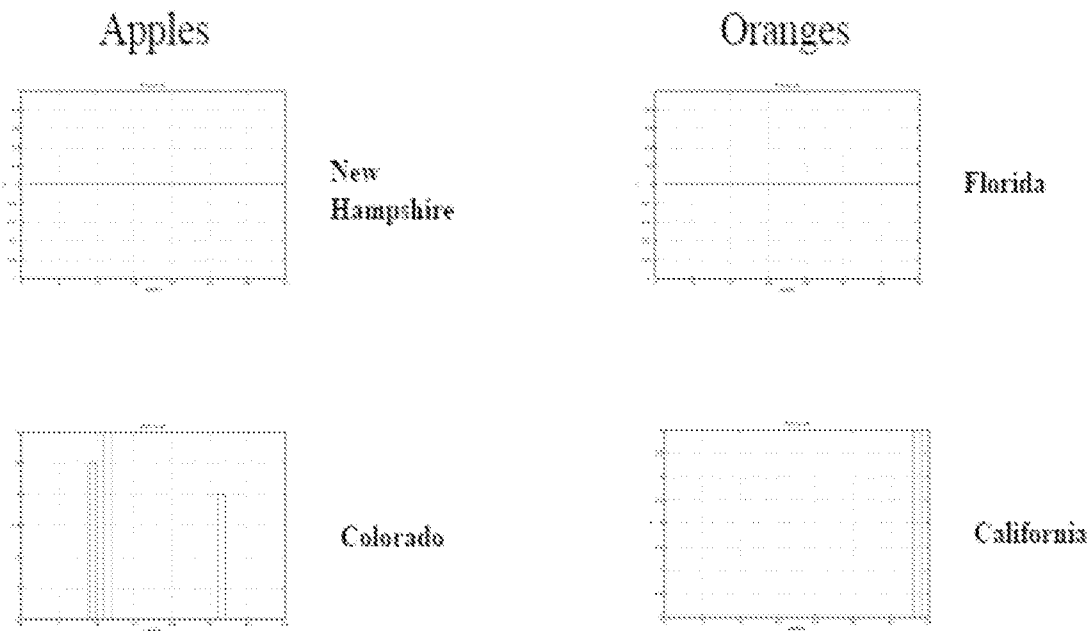
Continualized Lagrangian $L(\tilde{X}, U) = \tilde{X}^T Q \tilde{X} + U^T R U$

Continualized Criterion $\int_0^T L(\tilde{X}, U) dt + \tilde{X}^T(T) F \tilde{X}(T)$, where $T = 250$ days

Affine Decision Function $U(t) = K\tilde{X}(t) + g(t)$

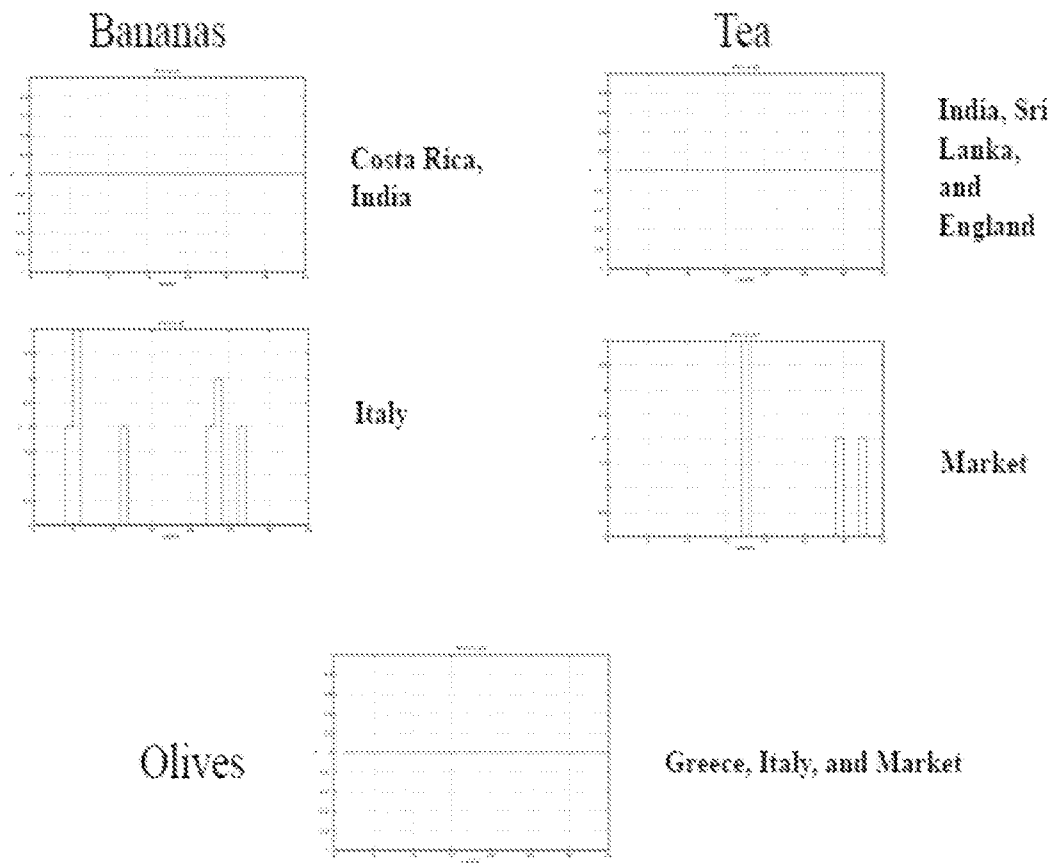
Computed Actions

What to buy from whom and when

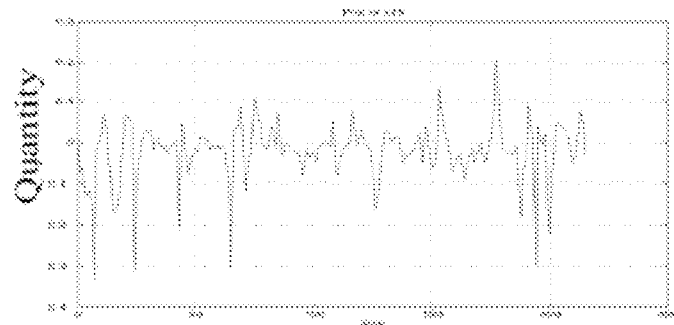


Computed Actions

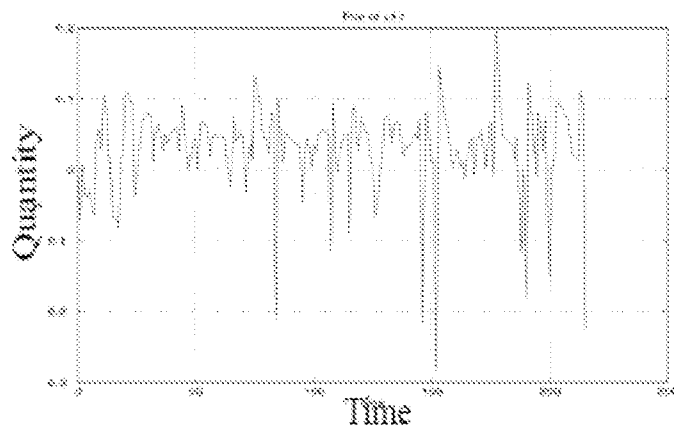
What to buy from whom and when



Continualized Computed Inventories

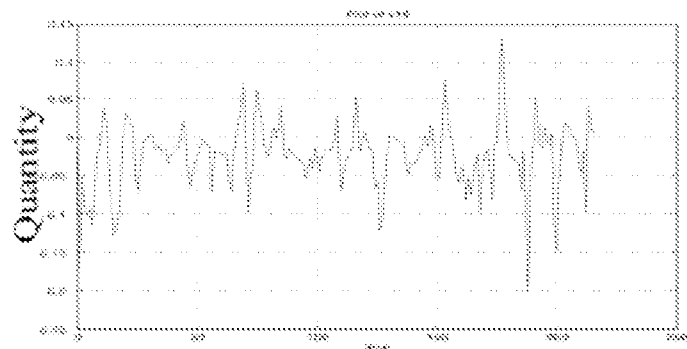


Apples

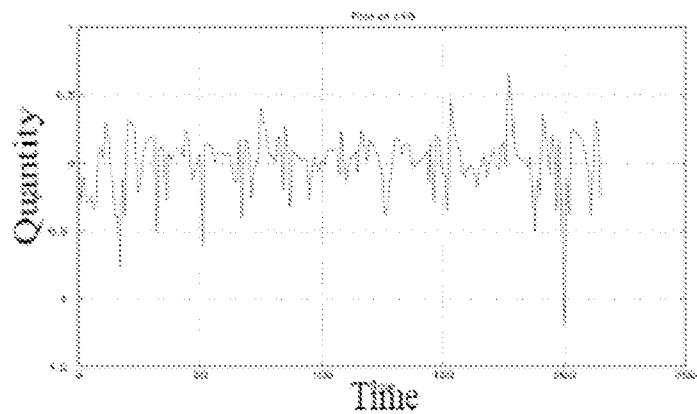


Oranges

Continualized Computed Inventories

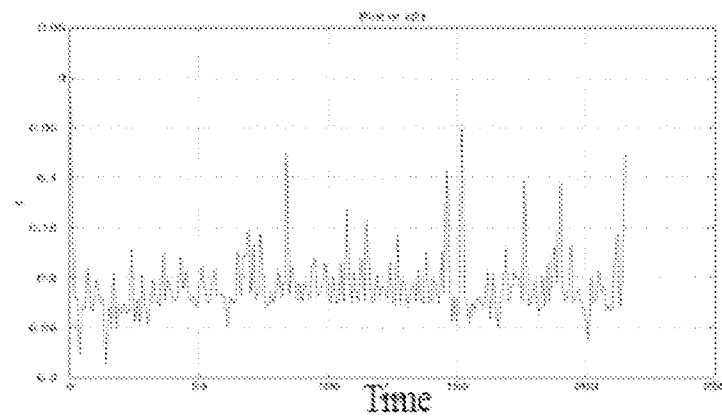
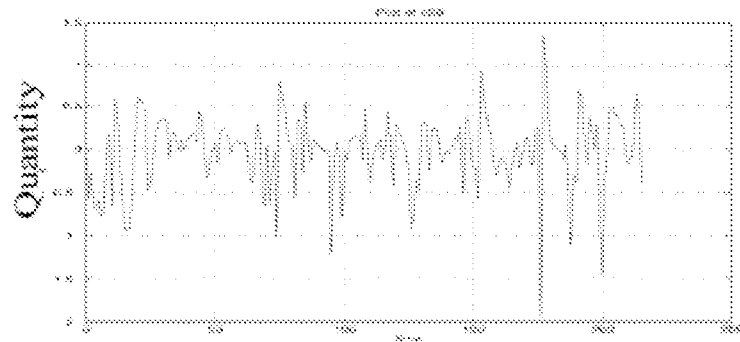


Bananas

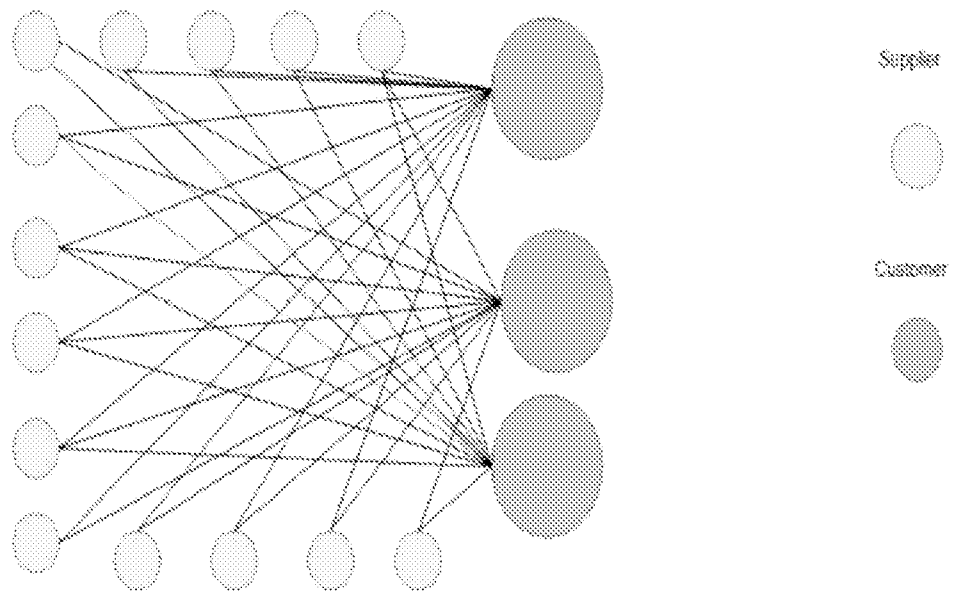


Tea

Continualized Computed Inventories

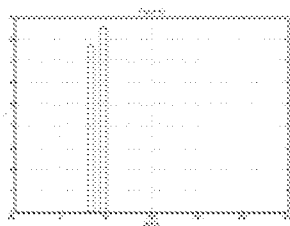


Direct Supply Chain Graph: with no intermediate storage



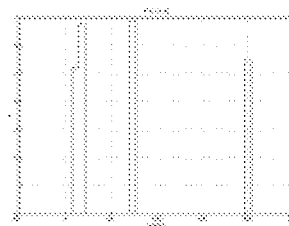
Computed Actions

Apples

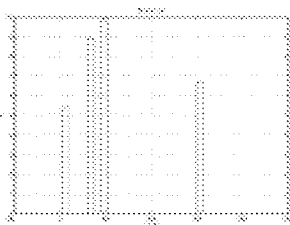


New
Hampshire

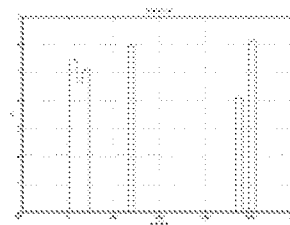
Oranges



Florida

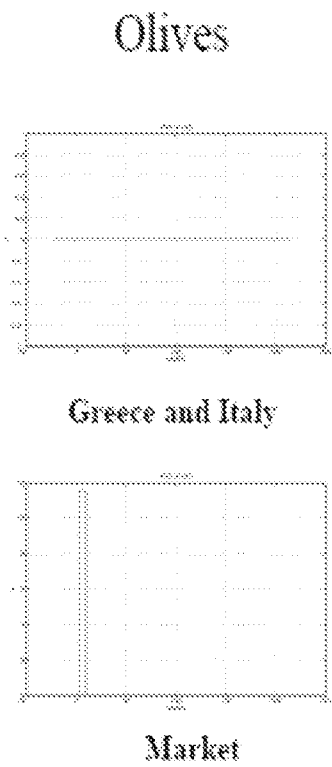
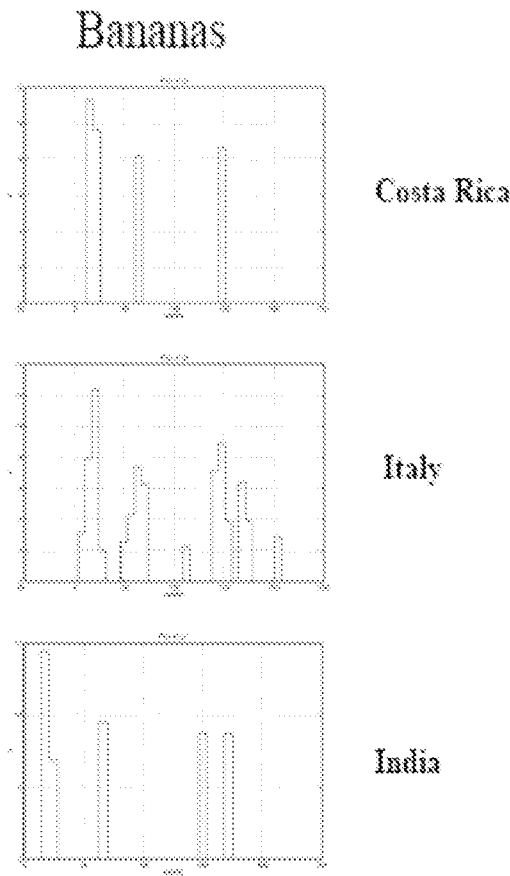


Colorado



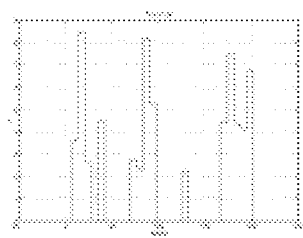
California

Computed Actions

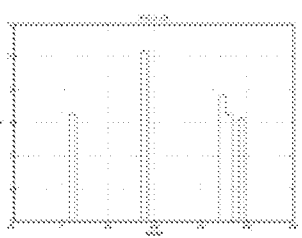


Computed Actions

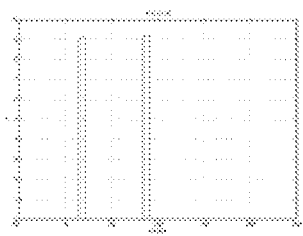
Tea



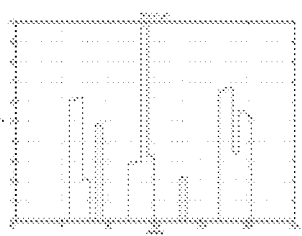
India



England

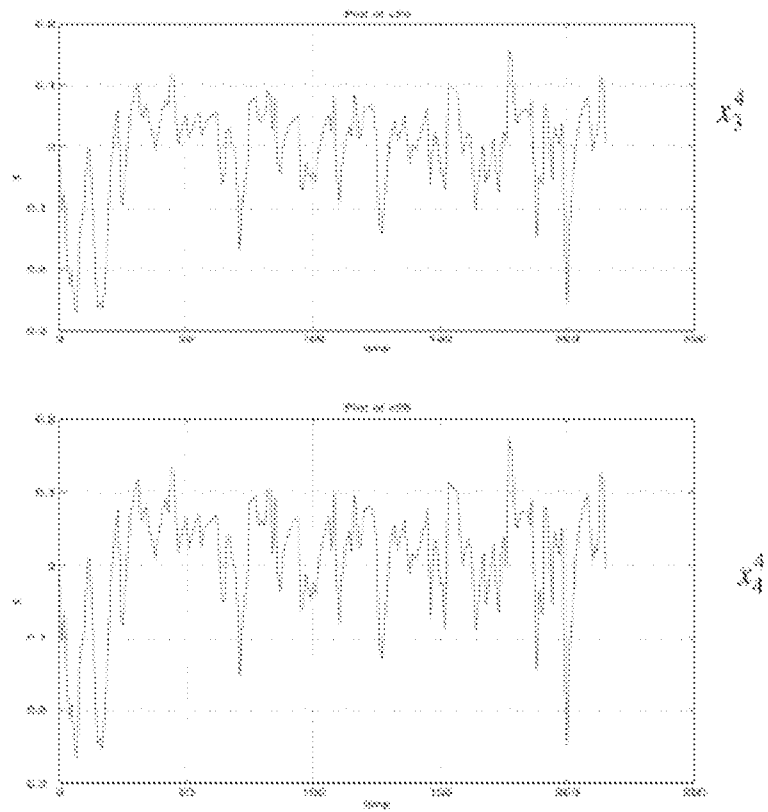


Sri Lanka

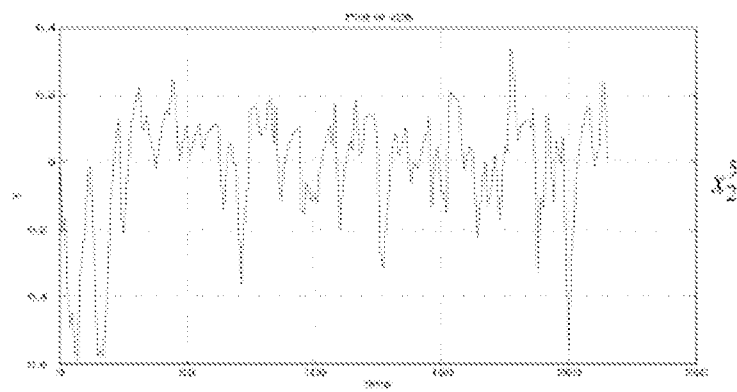
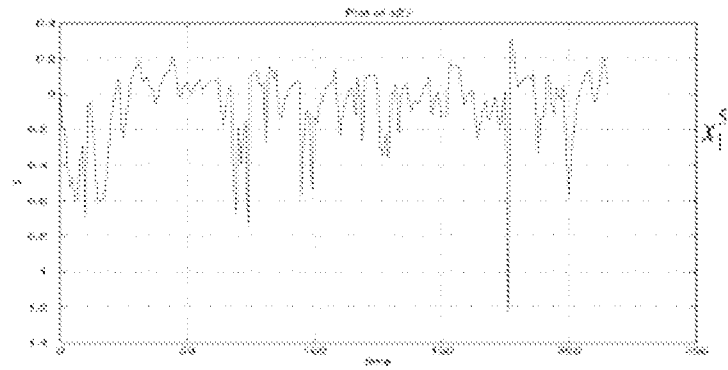


Market

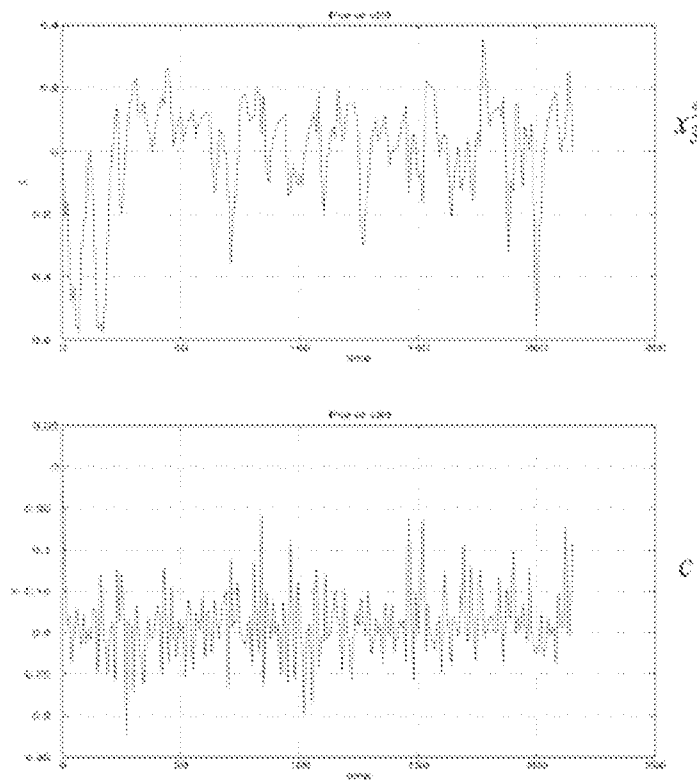
Continualized Computed Inventories



Continualized Computed Inventories



Continualized Computed Inventories



Summary of Agent characteristics

➤ **Proposed agent Cluster architecture**

- Fully asynchronous
- Cluster synchronization without umpire
- Arbitrary extensibility
- Real Time Distributed system
- Programmable

➤ **Proposed Agent dynamics**

- Optimal Hybrid control based
- Built-in adaptive capabilities
- Active learning
- Asymptotically stable

[00400] The following provides an example of use of the CDD system with a target system involving a day trader financial setting.

The financial application provides a set of tested strategies (denoted S) for investment trading. The chattering algorithm generates a combined strategy (denoted C) that uses data of past performance, and rules developed for the application. Chattering is a Dynamic Feedback Optimization Variational technique for fusing a set S of multiple strategies to obtain a hybrid system that generates decisions that outperform the ones provide in S , according to rules that specify a criterion (such as maximize net return over time) and a set of constraints.

The chattering algorithm is implemented in a distributed manner, where there is one strategy per agent. A mean-field agent generates a combined trading strategy optimal with respect to given goal and satisfying a given set of rules.

The financial model is defined by twelve rules including absolute, hard and soft rules, as well as a goal or query that specifies the criterion for optimization.

The rules are expressed using the following notation:

- I is the set of tested investment strategies.
- A pure strategy, i , includes a portfolio of *unspecified* equities and an investment strategy, for each $i \in I$.
- N is the number of pure strategies in the set of strategies, I , and $N > 1$.
- T is the range of dependence of current value of a strategy. The time interval $[t_0, t_0 + T]$ is discretized, such that $t = t_0, t_0 + 1, t_0 + 2, \dots, t_0 + T - 1$. This example considers the time in days.
- $w_i(t)$ is the strategy response and is a dynamic response of the system.
- $\rho_i(t)$ is the fraction of funds allocated to strategy i at time t .
- $u_i(t)$ is the control which changes the level of funds allocated to strategy i at time t .
- $f_i(t)$ is amount of funds allocated at the beginning of the next time period and is defined as $f_i(t) = w_i(t) + u_i(t)$.

The financial model is defined by the following rules:

1. Strategy response (absolute rule): The strategy response, for each pure strategy i , i in S , is the daily return signal for day t , denoted $w_i(t)$, for $t = t_0, t_0 + 1, t_0 + 2, \dots, t_0 + T - 1$, is specified as

$$w_i(t) = \frac{\text{dividends\_ex}_i(t) + \text{close\_price}_i(t) - \text{close\_price}_i(t-1)}{\text{close\_price}_i(t-1)} f_i(t-1)$$

where $\text{dividends\_ex}_i(t)$ and $\text{close\_price}_i(t)$ are provided by the data associated with strategy i . As a part of this rule, the money market pure strategy ($i = 1$) is characterized by a constant daily return signal, specified as

$$w_1(t) = \left[\frac{\text{interest}}{252} + 1 \right] f_1(t-1)$$

and interest is provided by data.

2. Cumulative revenue (or loss) per strategy (absolute rule): The rule for cumulative return, in this financial example, is given in terms of the log cumulative return, at time t which is given by

$$S_i(t + \delta(S_i(t))) = S_i(t) + \log w_i(t)$$

where the increment $\delta(S_i(t))$ is either provided by data or estimated.

3. Continualization rule (hard): Accompanying the cumulative return rule is the differential approximation rule,

$$\frac{dS_i(t)}{dt} = \frac{1}{\delta(S_i(t))} \ln w_i(t)$$

4. Risk per strategy (soft): The risk associated with each strategy i is taken as the acceleration of the strategy, where a large acceleration is associated with high risk, and a low acceleration is associated with low risk. The acceleration predicts local strategy behavior, and is specified as

$$\frac{d^2 S_i(t)}{dt^2} = -\ln w_i(t) \frac{\partial(\delta(S_i(t)))}{\partial S_i} \frac{1}{(\delta(S_i(t)))^2} \frac{dS_i(t)}{dt} + \frac{1}{\delta(S_i(t))} \frac{1}{w_i(t)} \frac{dw_i(t)}{dt}$$

where the second term is usually negligible.

5. Capacity (absolute): There is an upper bound, U , on the total amount in dollars invested in all of the strategies, W .

$$\sum_{i=1}^N w_i(t) = W \leq U$$

6. Cash bounds per strategy (hard): There is a minimum and maximum level for each strategy in terms of amount invested, denoted $w_i^{\min}$ and $w_i^{\max}$, that provides upper and lower bounds on the dollar value committed to each strategy so that $w_i^{\min} \leq w_i(t) \leq w_i^{\max}$.
7. Trading bounds (soft): The rule on trading bounds gives upper and lower bounds on the percentage of the portfolio for each strategy. For example, one strategy may never drop below 4% of the portfolio, but should not exceed 10% of the portfolio. These are soft rules because the bounds may be violated slightly if needed to ensure the hard rules are satisfied. They are defined as

$$\rho_i^{\min} \leq \frac{w_i(t)}{W} = \rho_i(t) \leq \rho_i^{\max}$$

8. Control: The rule that captures the reallocation of funds per time period is determined by $u_i(t)$ such that

$$f_i^{\min} \leq f_i(t) = w_i(t) + u_i(t) \leq f_i^{\max}$$

$$\sum_{i=1}^N u_i(t) = 0$$

9. Transfer funds from one strategy to another (hard): The rule that captures the transfer of funds from one strategy to another includes the log-accumulated return minus the switching cost, and is specified as

$$C(t+1) = C(t) - g(S(t), \rho(t), w(t), u(t)) + h(S(t), VS(t), u(t)) \quad (*)$$

where $C(t_0) = 0$ and $C(t)$ is the log accumulated return minus switching cost of the composite strategy.

The function $g(S(t), \rho(t), w(t), u(t))$ incorporates the expected return across all the strategies while measuring the "cost" of switching ownership amount the strategies, where $S(t)$ is the cumulative revenue across all strategies, $\rho(t)$ is the current mixture vector of strategies, and $u(t)$ is the recommended mixture changes.

The volatility function, denoted $h(S(t), VS(t), u(t))$, also depends on $VS(t)$, the quadratic dispersion of $S(t)$ which is determined by the acceleration or risk across all strategies. The control is intended to maximize return while minimizing risk.

The expression in (*) can be approximated by,

$$C(t) = -\frac{1}{\theta(C(t))} g(S(t), \rho(t), w(t), u(t)) + h(S(t), VS(t), u(t))$$

10. Optimization criterion (goal/query): The optimization criterion is to maximize the log accumulated return minus the switching cost at the terminal time, or equivalently,

$$\min_{S(t), t_0 \leq t \leq t_0 + T} \{-C(t_0 + T)\}$$

11. Time horizon for each strategy (soft): The rule regarding time horizon for each strategy provides an historic interval $[t_c - T_i, t_c]$ for reconstructing the dynamics for strategy i , where t_c is the current time. The value for T_i is specified as

$$T_i = \arg \left(10 \log_{10} \left(\frac{1}{T_i} \int_0^{T_i} S_i(t_c) S_i(t_c - \tau) d\tau \right) \leq 3db \right)$$

where 3db is the minimum level at which the time autocorrelation of the current point depends on past points. Then, $T = \max_{i \in S} T_i$.

12. Mean field upgrade (soft): The rule regarding mean field upgrade provides a parameter that is a time constant for the time interval to update the mean field agent.

[00401] It will also be appreciated that in some embodiments the functionality provided by the routines discussed above may be provided in alternative ways, such as being split among more routines or consolidated into fewer routines. Similarly, in some embodiments illustrated routines may provide more or less functionality than is described, such as when other illustrated routines instead lack or include such functionality respectively, or when the amount of functionality that is provided is altered. In addition, while various operations may be illustrated as being performed in a particular manner (e.g., in serial or in parallel, synchronously or asynchronously, etc.) and/or in a particular order, those skilled in the art will

appreciate that in other embodiments the operations may be performed in other orders and in other manners. Those skilled in the art will also appreciate that the data structures discussed above may be structured in different manners, such as by having a single data structure split into multiple data structures or by having multiple data structures consolidated into a single data structure. Similarly, in some embodiments illustrated data structures may store more or less information than is described, such as when other illustrated data structures instead lack or include such information respectively, or when the amount or types of information that is stored is altered.

[00402] From the foregoing it will be appreciated that, although specific embodiments have been described herein for purposes of illustration, various modifications may be made without deviating from the spirit and scope of the invention. Accordingly, the invention is not limited except as by the appended claims and the elements recited therein. In addition, while certain aspects of the invention are presented below in certain claim forms, the inventors contemplate the various aspects of the invention in any available claim form. For example, while only some aspects of the invention may currently be recited as being embodied in a computer-readable medium, other aspects may likewise be so embodied.

[00403] Non-exclusive embodiments are reflected in each of the following clauses:

[00404] AAA. A computer-implemented method comprising:

receiving, by a collaborative distributed decision system implemented by one or more computing systems, system information from one or more users that describes a physical system having a plurality of inter-related elements and having one or more outputs whose values vary based at least in part on values of one or more manipulatable control elements of the plurality, wherein the system information includes multiple rules that each has one or more conditions to evaluate and that specify restrictions involving the plurality of elements, the

multiple rules including one or more soft rules whose conditions evaluate to one of three or more possible values under differing situations to represent varying degrees of uncertainty and further including additional rules whose conditions evaluate to either true or false under differing situations;

receiving, by the collaborative distributed decision system, objective information that identifies a goal to be achieved during controlling of the physical system;

obtaining, by the collaborative distributed decision system, sensor information that identifies current state information for at least one element of the plurality;

converting, by the collaborative distributed decision system, the system information and the objective information and the sensor information to coupled differential equations that represent a model describing a current state of the physical system;

performing, by the collaborative distributed decision system, a piecewise linear analysis of the coupled differential equations to identify one or more control actions that manipulate values of the one or more manipulatable control elements and that provide a solution for the goal, wherein the provided solution is within a threshold amount of an optimal solution for the goal; and

initiating performance of the one or more control actions in the physical system to manipulate values of the one or more manipulatable control elements and to cause resulting changes in the values of the one or more outputs.

[00405] BBB. The computer-implemented method of clause AAA alone or of clause AAA in combination with any or all other listed clauses, wherein the physical system is an electricity generating facility, wherein the plurality of inter-related elements include multiple alternative electricity sources within the electricity generating facility, wherein the manipulatable control elements include one or more controls to determine whether to accept a request to supply a specified amount of electricity at a current time and to select which alternative electricity source to provide the specified amount of electricity at the current time if accepted, wherein the outputs include the electricity being provided, and wherein the goal is

to maximize profits for the electricity generating facility from providing of the electricity.

[00406] CCC. The computer-implemented method of clause AAA alone or of clause AAA in combination with any or all other listed clauses, wherein the physical system is an energy generating facility, wherein the plurality of inter-related elements include at least one energy source within the energy generating facility and at least one energy storage mechanism within the energy generating facility, wherein the manipulatable control elements include one or more controls to determine whether to accept a request to supply a specified amount of energy at a current time and to determine to provide energy to the at least one energy storage mechanism at the current time if not accepted and to provide energy from the at least one energy source if accepted, wherein the outputs include the energy being provided, and wherein the goal is to maximize profits for the electricity generating facility from providing of the energy.

[00407] DDD. The computer-implemented method of clause AAA alone or of clause AAA in combination with any or all other listed clauses, wherein the physical system is a vehicle, wherein the plurality of inter-related elements include a motor and a battery of the vehicle, wherein the manipulatable control elements include one or more controls to select whether at a current time to remove energy from the battery to power the motor or to add excess energy to the battery and how much energy to remove from the battery, wherein the outputs include effects of the motor to move the vehicle, and wherein the goal is to move the vehicle at one or more specified speeds with a minimum of energy produced from the battery.

[00408] EEE. The computer-implemented method of clause DDD wherein the plurality of inter-related elements further includes an engine that is manipulatable to modify energy generated from the engine, wherein the manipulatable control elements further include one or more additional controls to determine how much energy to generate from the engine for use at least in part in adding the excess energy to the battery, and wherein the goal includes to minimize use of fuel by the engine.

[00409] FFF. The computer-implemented method of clause AAA alone or of clause AAA in combination with any or all other listed clauses, wherein the physical system includes product inventory at one or more locations, wherein the plurality of inter-related elements include one or more product sources that provide products and increase the inventory at the one or more locations and further include one or more product recipients that receive products and decrease the inventory at the one or more locations, wherein the manipulatable control elements include one or more first controls to select at a current time one or more first amounts of one or more products to request from the one or more product sources, and further include one or more second controls to select at the current time one or more second amounts of at least one product to provide to the one or more product recipients, wherein the outputs include products being provided from the one or more locations to the one or more product recipients, and wherein the goal is to maximize profit of an entity operating the one or more locations while maintaining the inventory at one or more specified levels.

[00410] GGG. A non-transitory computer-readable medium having stored contents that cause one or more computing systems of a collaborative distributed decision system to perform a method, the method comprising:

receiving, by the one or more computing systems, system information from one or more users that describes a target system having a plurality of elements that are inter-related and that include one or more manipulatable control elements with modifiable values, wherein the system information includes multiple rules that each has one or more conditions to evaluate and that specify restrictions involving the plurality of elements, the multiple rules including one or more soft rules whose conditions evaluate to one of three or more possible values under differing situations to represent varying degrees of uncertainty and further including additional rules whose conditions evaluate to either true or false under differing situations;

receiving, by the one or more computing systems, objective information that identifies a goal to be achieved while modifying the values of the one or more manipulatable control elements;

obtaining, by the one or more computing systems, sensor information that identifies information about a physical state of at least one element of the plurality at a specified time;

converting, by the one or more computing systems, the system information and the objective information and the sensor information to coupled differential equations that represent a model describing a state of the target system at the specified time;

performing, by the one or more computing systems, a piecewise linear analysis of the coupled differential equations to identify one or more values of the one or more manipulatable control elements that provide a solution for the goal for the specified time, wherein the provided solution is within a threshold amount of an optimal solution for the goal for the specified time; and

providing information about the identified one or more values, to enable modification of the one or more manipulatable control elements for the specified time to have the identified one or more values.

[00411] HHH. The non-transitory computer-readable medium of clause GGG alone or of clause GGG in combination with any or all other listed clauses, wherein the target system is a physical system having one or more outputs whose values vary based at least in part on the values of the manipulatable control elements, and wherein the stored contents include software instructions that, when executed, further cause the one or more computing systems to initiate performance of one or more control actions in the physical system to modify the one or more manipulatable control elements to have the identified one or more values and to cause resulting changes in the values of the one or more outputs.

[00412] III. The non-transitory computer-readable medium of clause GGG alone or of clause GGG in combination with any or all other listed clauses, wherein the target system includes one or more computing resources being protected from unauthorized operations, wherein the plurality of inter-related elements include one or more sources of attempts to perform operations, wherein the manipulatable control elements include one or more controls to determine whether a change in authorization to a specified type of operation is needed and to select one or more

actions to take to implement the change in authorization if so determined, and wherein the goal is to minimize unauthorized operations that are performed.

[00413] JJJ. The non-transitory computer-readable medium of clause GGG alone or of clause GGG in combination with any or all other listed clauses, wherein the target system includes one or more information sources to be analyzed to determine a risk level from information of the one or more information sources, wherein the manipulatable control elements include one or more controls to determine whether the risk level exceeds a specified threshold and to select one or more actions to take to mitigate the risk level, and wherein the goal is to minimize the risk level.

[00414] KKK. The non-transitory computer-readable medium of clause GGG alone or of clause GGG in combination with any or all other listed clauses, wherein the target system includes one or more financial markets, wherein the plurality of inter-related elements include items that can be purchased and/or sold in the one or more financial markets, wherein the manipulatable control elements include one or more controls to determine whether to purchase or sell particular items at particular times and to select one or more actions to initiate transactions to purchase or sell the particular items at the particular times, and wherein the goal is to maximize profit while maintaining risk below a specified threshold.

[00415] LLL. The non-transitory computer-readable medium of clause GGG alone or of clause GGG in combination with any or all other listed clauses, wherein the target system includes functionality to perform coding for medical procedures performed on humans, wherein the plurality of inter-related elements include a plurality of medical codes corresponding to a plurality of medical procedures, wherein the manipulatable control elements include one or more controls to select particular medical codes to associate with particular medical procedures in specified circumstances, and wherein the goal is to minimize errors in selected medical codes that cause revenue leakage.

[00416] MMM. A computer-implemented method comprising:
obtaining, by one or more computing systems of a collaborative distributed decision system, coupled differential equations that represent a current state of a

physical system and that are generated from system information and objective information and sensor information, wherein the physical system has a plurality of inter-related elements and has one or more outputs whose values vary based at least in part on values of one or more manipulatable control elements of the plurality, wherein the system information is supplied by one or more users to describe the physical system and includes multiple rules that each has one or more conditions to evaluate and that specify restrictions involving the plurality of elements, wherein the objective information identifies a goal to be achieved during controlling of the physical system, and wherein the sensor information identifies current state information for at least one element of the plurality and partial initial state information for the physical system at an earlier time;

performing, by the one or more computing systems, a piecewise linear analysis of the coupled differential equations to identify one or more control actions to take in the physical system that manipulate values of the one or more manipulatable control elements and that provide a solution for the goal within a threshold amount of an optimal solution for the goal, wherein the performing of the piecewise linear analysis includes:

dividing a time window from the earlier time to the specified time into a succession of a plurality of time slices that each, other than a first time slice of the succession, overlaps at least in part with a prior time slice of the succession;

evaluating, based on the partial initial state information, the coupled differential equations to identify an initial solution for the goal for the first time slice that includes simulating effects of manipulating the one or more manipulatable control elements to one or more initial values, and storing a model describing a state of the physical system for the first time slice that includes the simulated effects of the manipulating to the one or more initial values;

for each time slice of the succession after the first time slice, updating the stored model for the prior time slice to reflect an additional solution for the goal for the time slice that includes simulating effects of further manipulating the one or more manipulatable control elements to one or more additional values; and

after updating the stored model to reflect the additional solution for the goal for a last time slice of the succession, further updating the stored model for the last time slice to reflect a further solution for the goal for a next time period after the time window based at least in part on the identified current state information, wherein the further solution includes the identified one or more control actions to take in the physical system for a current time; and

providing information about the identified one or more control actions, to enable actions to be taken in the physical system for the current time to affect the outputs based on the identified one or more control actions.

[00417] NNN. The computer-implemented method of clause MMM alone or of clause MMM in combination with any or all other listed clauses, wherein the physical system is an electricity generating facility, wherein the plurality of inter-related elements include multiple alternative electricity sources within the electricity generating facility, wherein the manipulatable control elements include one or more controls to determine whether to accept a request to supply a specified amount of electricity at the current time and to select which alternative electricity source to provide the specified amount of electricity at the current time if accepted, wherein the outputs include the electricity being provided, and wherein the goal includes to maximize profits for the electricity generating facility from providing of the electricity.

[00418] OOO. The computer-implemented method of clause MMM alone or of clause MMM in combination with any or all other listed clauses, wherein the physical system is an energy generating facility, wherein the plurality of inter-related elements include at least one energy source within the energy generating facility and at least one energy storage mechanism within the energy generating facility, wherein the manipulatable control elements include one or more controls to determine whether to accept a request to supply a specified amount of energy at the current time and to determine to provide energy to the at least one energy storage mechanism at the current time if not accepted and to provide energy from the at least one energy source at the current time if accepted, wherein the outputs

include the energy being provided, and wherein the goal includes to maximize profits for the electricity generating facility from providing of the energy.

[00419] PPP. The computer-implemented method of clause MMM alone or of clause MMM in combination with any or all other listed clauses, wherein the physical system is a vehicle, wherein the plurality of inter-related elements include a motor and a battery of the vehicle, wherein the manipulatable control elements include one or more controls to select whether at the current time to remove energy from the battery to power the motor or to add excess energy to the battery and how much energy to remove from the battery, wherein the outputs include effects of the motor to move the vehicle at the current time, and wherein the goal includes to move the vehicle at one or more specified speeds with a minimum of energy produced from the battery.

[00420] QQQ. The computer-implemented method of clause PPP wherein the plurality of inter-related elements further includes an engine that is manipulatable to modify energy generated from the engine, wherein the manipulatable control elements further include one or more additional controls to determine how much energy to generate from the engine for use at least in part in adding the excess energy to the battery, and wherein the goal further includes to minimize use of fuel by the engine.

[00421] RRR. The computer-implemented method of clause MMM alone or of clause MMM in combination with any or all other listed clauses, wherein the physical system includes product inventory at one or more locations, wherein the plurality of inter-related elements include one or more product sources that provide products and increase the inventory at the one or more locations and further include one or more product recipients that receive products and decrease the inventory at the one or more locations, wherein the manipulatable control elements include one or more first controls to select at the current time one or more first amounts of one or more products to request from the one or more product sources, and further include one or more second controls to select at the current time one or more second amounts of at least one product to provide to the one or more product recipients, wherein the outputs include products being provided from

the one or more locations to the one or more product recipients, and wherein the goal includes to maximize profit of an entity operating the one or more locations while maintaining the inventory at one or more specified levels.

[00422] SSS. A non-transitory computer-readable medium having stored contents that cause one or more computing systems to perform a method, the method comprising:

obtaining, by the one or more computing systems, coupled differential equations that represent a state of a target system at a specified time and that are generated from system information and objective information and sensor information, wherein the target system has a plurality of elements that are inter-related and that include one or more control elements with modifiable values, wherein the system information is supplied by one or more users to describe the physical system and includes restrictions involving the plurality of elements, wherein the objective information identifies a goal to be achieved based at least in part on modifying the values of the control elements, and wherein the sensor information identifies state information for the specified time for at least one element of the plurality;

performing, by the one or more computing systems, a piecewise linear analysis of the coupled differential equations to identify a solution for the goal for the specified time within a threshold amount of an optimal solution for the goal, wherein the identified solution has one or more associated control actions that modify at least one value of at least one of the control elements in a specified manner, and wherein the performing of the piecewise linear analysis includes:

dividing a time window from an earlier time to the specified time into a succession of a plurality of time slices;

evaluating, based on initial state information for the earlier time, the coupled differential equations to identify an initial solution for the goal for a first time slice of the succession that includes simulating effects of modifying one or more values of the one or more manipulatable control elements in a specified initial manner, and storing a model describing a state of the target system for the

first time slice that includes the simulated effects of the modifying of the one or more values;

for each time slice of the succession after the first time slice, updating the stored model for a prior time slice to reflect an additional solution for the goal for the time slice that includes simulating effects of further modifying one or more values of the one or more manipulatable control elements; and

after updating the stored model to reflect the additional solution for the goal for a last time slice of the succession, further updating the stored model for the last time slice to reflect a further solution for the goal for a next time period after the time window based at least in part on the identified state information for the specified time, wherein the further solution includes the one or more associated control actions; and

providing information about the one or more associated control actions, to enable modification of the at least one value of the at least one control element for the specified time based on the one or more associated control actions.

[00423] TTT. The non-transitory computer-readable medium of clause SSS alone or of clause SSS in combination with any or all other listed clauses, wherein the target system is a physical system having one or more outputs whose values vary based at least in part on the values of the control elements, wherein the one or more computing systems are part of a collaborative distributed decision system, and wherein the stored contents include software instructions that, when executed, further cause the one or more computing systems to initiate performance of the one or more associated control actions in the physical system to modify the at least one value of the at least one control element and to cause resulting changes in the values of the one or more outputs for the specified time.

[00424] UUU. The non-transitory computer-readable medium of clause SSS alone or of clause SSS in combination with any or all other listed clauses, wherein the target system includes one or more computing resources being protected from unauthorized operations, wherein the plurality of inter-related elements include one or more sources of attempts to perform operations, wherein the control elements include one or more controls to determine whether a change in

authorization to a specified type of operation is needed and to select one or more actions to take to implement the change in authorization if so determined, and wherein the goal includes to minimize unauthorized operations that are performed.

[00425] VVV. The non-transitory computer-readable medium of clause SSS alone or of clause SSS in combination with any or all other listed clauses, wherein the target system includes one or more information sources to be analyzed to determine a risk level from information of the one or more information sources, wherein the control elements include one or more controls to determine whether the risk level exceeds a specified threshold and to select one or more actions to take to mitigate the risk level, and wherein the goal includes to minimize the risk level.

[00426] WWW. The non-transitory computer-readable medium of clause SSS alone or of clause SSS in combination with any or all other listed clauses, wherein the target system includes one or more financial markets, wherein the plurality of inter-related elements include items that can be purchased and/or sold in the one or more financial markets, wherein the control elements include one or more controls to determine whether to purchase or sell particular items at particular times and to select one or more actions to initiate transactions to purchase or sell the particular items at the particular times, and wherein the goal includes to maximize profit while maintaining risk below a specified threshold.

[00427] XXX. The non-transitory computer-readable medium of clause SSS alone or of clause SSS in combination with any or all other listed clauses, wherein the target system includes functionality to perform coding for medical procedures performed on humans, wherein the plurality of inter-related elements include a plurality of medical codes corresponding to a plurality of medical procedures, wherein the control elements include one or more controls to select particular medical codes to associate with particular medical procedures in specified circumstances, and wherein the goal includes to minimize errors in selected medical codes that cause revenue leakage.

[00428] AAAA. A computer-implemented method comprising:

obtaining, by one or more computing systems of a collaborative distributed decision system, and for each of multiple decision modules that collectively describe a state of a physical system at a current time, a distinct model associated with the decision module that is represented by a distinct set of coupled differential equations, wherein the physical system has a plurality of inter-related elements and has one or more outputs whose values vary based at least in part on one or more manipulatable control elements of the plurality, and wherein each of the multiple decision modules' associated distinct model describes state information for the current time of at least of a portion of the physical system and reflects a solution for a specified goal of the decision module that is to be achieved during controlling of the physical system;

generating, by the one or more computing systems, and for each of the multiple decision modules, a consensus shared model for the decision module for the current time by determining a Pareto equilibrium for the model associated with the decision module and for an intermediate shared model representing the models associated with some other of the multiple decision modules, wherein the consensus shared model for the decision module simultaneously provides solutions for the goals of the decision module and the some other decision modules such that the provided solutions have an associated error measurement within a defined threshold relative to a global optimal solution, and wherein the consensus shared model for the decision module has one or more associated control actions to perform in the physical system to manipulate the manipulatable control elements in a specified manner for the provided solutions; and

providing information about the one or more associated control actions of one or more of the consensus shared models, to enable actions to be taken in the physical system to affect the outputs based on the one or more associated control actions of the one or more consensus shared models.

[00429] BBBB. The computer-implemented method of clause AAAA alone or of clause AAAA in combination with any or all other listed clauses, wherein the multiple decision modules collectively include at least one overall objective to be achieved during controlling of the physical system, wherein the physical system is

an electricity generating facility, wherein the plurality of inter-related elements include multiple alternative electricity sources within the electricity generating facility, wherein the manipulatable control elements include one or more controls to determine whether to accept a request to supply a specified amount of electricity at the current time and to select which alternative electricity source to provide the specified amount of electricity at the current time if accepted, wherein the outputs include the electricity being provided, and wherein the at least one overall objective includes to maximize profits for the electricity generating facility from providing of the electricity.

[00430] CCCC. The computer-implemented method of clause AAAA alone or of clause AAAA in combination with any or all other listed clauses, wherein the multiple decision modules collectively include at least one overall objective to be achieved during controlling of the physical system, wherein the physical system is an energy generating facility, wherein the plurality of inter-related elements include at least one energy source within the energy generating facility and at least one energy storage mechanism within the energy generating facility, wherein the manipulatable control elements include one or more controls to determine whether to accept a request to supply a specified amount of energy at the current time and to determine to provide energy to the at least one energy storage mechanism at the current time if not accepted and to provide energy from the at least one energy source at the current time if accepted, wherein the outputs include the energy being provided, and wherein the at least one overall objective includes to maximize profits for the electricity generating facility from providing of the energy.

[00431] DDDD. The computer-implemented method of clause AAAA alone or of clause AAAA in combination with any or all other listed clauses, wherein the multiple decision modules collectively include at least one overall objective to be achieved during controlling of the physical system, wherein the physical system is a vehicle, wherein the plurality of inter-related elements include a motor and a battery of the vehicle, wherein the manipulatable control elements include one or more controls to select whether at the current time to remove energy from the battery to power the motor or to add excess energy to the battery and how much

energy to remove from the battery, wherein the outputs include effects of the motor to move the vehicle at the current time, and wherein the at least one overall objective includes to move the vehicle at one or more specified speeds with a minimum of energy produced from the battery.

[00432] EEEE. The computer-implemented method of clause DDDD wherein the plurality of inter-related elements further includes an engine that is manipulatable to modify energy generated from the engine, wherein the manipulatable control elements further include one or more additional controls to determine how much energy to generate from the engine for use at least in part in adding the excess energy to the battery, and wherein the at least one overall objective further includes to minimize use of fuel by the engine.

[00433] FFFF. The computer-implemented method of clause AAAA alone or of clause AAAA in combination with any or all other listed clauses, wherein the multiple decision modules collectively include at least one overall objective to be achieved during controlling of the physical system, wherein the physical system includes product inventory at one or more locations, wherein the plurality of inter-related elements include one or more product sources that provide products and increase the inventory at the one or more locations and further include one or more product recipients that receive products and decrease the inventory at the one or more locations, wherein the manipulatable control elements include one or more first controls to select at the current time one or more first amounts of one or more products to request from the one or more product sources, and further include one or more second controls to select at the current time one or more second amounts of at least one product to provide to the one or more product recipients, wherein the outputs include products being provided from the one or more locations to the one or more product recipients, and wherein the at least one overall objective includes to maximize profit of an entity operating the one or more locations while maintaining the inventory at one or more specified levels.

[00434] GGGG. A non-transitory computer-readable medium having stored contents that cause one or more computing systems of a collaborative distributed decision system to perform a method, the method comprising:

obtaining, by the one or more computing systems, and for each of multiple decision modules that collectively describe a state of a target system, a distinct model associated with the decision module, wherein the target system has a plurality of elements that are inter-related and that include one or more control elements with modifiable values, and wherein each of the multiple decision modules' associated distinct model describes information about a physical state of at least one element of the plurality and reflects a solution for a specified goal of the decision module that is to be achieved based at least in part on modifying of the values of the control elements;

generating, by the one or more computing systems, and for each of the multiple decision modules, a consensus shared model for the decision module by determining a Pareto equilibrium for the model associated with the decision module and for an intermediate shared model representing the models associated with some other of the multiple decision modules, wherein the consensus shared model for the decision module simultaneously provides solutions for the goals of the decision module and the some other decision modules such that the provided solutions have an associated error measurement within a defined threshold relative to a global optimal solution, and wherein the consensus shared model for the decision module has one or more associated control actions that modify the value of at least one of the control elements in a specified manner for the provided solutions; and

providing information about the one or more associated control actions of one or more of the consensus shared models, to enable modification of values of control elements based on the one or more associated control actions.

[00435]

HHHH. The non-transitory computer-readable medium of clause GGGG alone or of clause GGGG in combination with any or all other listed clauses, wherein the target system is a physical system having one or more outputs whose values vary based at least in part on the values of the control elements, and wherein the stored contents include software instructions that, when executed, further cause the one or more computing systems to initiate performance of the one or more associated control actions in the physical system

to modify the values of the control elements to have specified values and to cause resulting changes in the values of the one or more outputs for a specified time.

[00436] IIII. The non-transitory computer-readable medium of clause GGGG alone or of clause GGGG in combination with any or all other listed clauses, wherein the multiple decision modules collectively include at least one overall objective to be achieved based at least in part on modifying of the values of the control elements, wherein the target system includes one or more computing resources being protected from unauthorized operations, wherein the plurality of inter-related elements include one or more sources of attempts to perform operations, wherein the control elements include one or more controls to determine whether a change in authorization to a specified type of operation is needed and to select one or more actions to take to implement the change in authorization if so determined, and wherein the at least one overall objective includes to minimize unauthorized operations that are performed.

[00437] JJJJ. The non-transitory computer-readable medium of clause GGGG alone or of clause GGGG in combination with any or all other listed clauses, wherein the multiple decision modules collectively include at least one overall objective to be achieved based at least in part on modifying of the values of the control elements, wherein the target system includes one or more information sources to be analyzed to determine a risk level from information of the one or more information sources, wherein the control elements include one or more controls to determine whether the risk level exceeds a specified threshold and to select one or more actions to take to mitigate the risk level, and wherein the at least one overall objective includes to minimize the risk level.

[00438] KKKK. The non-transitory computer-readable medium of clause GGGG alone or of clause GGGG in combination with any or all other listed clauses, wherein the multiple decision modules collectively include at least one overall objective to be achieved based at least in part on modifying of the values of the control elements, wherein the target system includes one or more financial markets, wherein the plurality of inter-related elements include items that can be purchased and/or sold in the one or more financial markets, wherein the control

elements include one or more controls to determine whether to purchase or sell particular items at particular times and to select one or more actions to initiate transactions to purchase or sell the particular items at the particular times, and wherein the at least one overall objective includes to maximize profit while maintaining risk below a specified threshold.

[00439] LLLL. The non-transitory computer-readable medium of clause GGGG alone or of clause GGGG in combination with any or all other listed clauses, wherein the multiple decision modules collectively include at least one overall objective to be achieved based at least in part on modifying of the values of the control elements, wherein the target system includes functionality to perform coding for medical procedures performed on humans, wherein the plurality of inter-related elements include a plurality of medical codes corresponding to a plurality of medical procedures, wherein the control elements include one or more controls to select particular medical codes to associate with particular medical procedures in specified circumstances, and wherein the at least one overall objective includes to minimize errors in selected medical codes that cause revenue leakage.

[00440] MMMM. A computer-implemented method comprising:
generating, by one or more computing systems of a collaborative distributed decision system, an executable decision module for use with a physical system having a plurality of inter-related elements and having one or more outputs whose values vary based at least in part on one or more manipulatable control elements of the plurality, the generating including:

receiving, by the one or more computing systems and from one or more users, system information describing at least a portion of the physical system using multiple rules that each has one or more conditions to evaluate and that specify restrictions involving the plurality of elements;

converting, by the one or more computing systems and to multiple constraints, the system information and information about a goal to be achieved during controlling of the physical system and information about sensor variables whose values are available during the controlling of the physical system;

validating, by the one or more computing systems and based on multiple validation rules, the multiple constraints by testing at least one of controllability, observability, stability or goal completeness, wherein testing controllability ensures that each of the manipulatable control elements is specified in the multiple rules to affect one or more other elements of the plurality, wherein testing observability ensures that each of the sensor variables is specified in the multiple rules to relate to at least one element of the plurality, wherein testing stability ensures that solutions determined by the generated decision module during controlling of the at least portion of the physical system will converge with any other solutions determined by any other generated decision modules during controlling of other portions of the physical system, wherein testing goal completeness ensures that each of the manipulatable control elements is reflected in the goal;

converting, by the one or more computing systems, the validated constraints to coupled differential equations that represent a model for use in describing a state of the target system;

training, by the one or more computing systems, the model using state information for the plurality of elements for multiple times, to enable the model to determine values of one or more elements that are not directly observable; and

testing, by the one or more computing systems, performance of the trained model in controlling the at least portion of the physical system by simulating manipulations of the manipulatable control elements to affect the outputs in expected manners; and

if the testing is successful, providing the generated decision module with the trained model, to enable execution of the generated decision module to perform actual controlling of the at least portion of the physical system.

[00441] NNNN. The computer-implemented method of clause MMMM alone or of clause MMMM in combination with any or all other listed clauses, wherein the providing of the generated decision module with the trained model includes initiating the execution of the generated decision module in an environment that

provides the executing generated decision module with access to perform the actual controlling of the at least portion of the physical system.

[00442] OOOO. The computer-implemented method of clause MMMM alone or of clause MMMM in combination with any or all other listed clauses, wherein the generating of the executable decision module includes providing, by the one or more computing systems, an online interface to a rule builder component of the collaborative distributed decision system, and wherein the receiving of the system information from the one or more users includes receiving electronic communications at the online interface that are sent over one or more computer networks from one or more client devices of the one or more users.

[00443] PPPP. The computer-implemented method of clause MMMM alone or of clause MMMM in combination with any or all other listed clauses, wherein the physical system is an electricity generating facility, wherein the plurality of inter-related elements include multiple alternative electricity sources within the electricity generating facility, wherein the manipulatable control elements include one or more controls to determine whether to accept a request to supply a specified amount of electricity at a current time and to select which alternative electricity source to provide the specified amount of electricity at the current time if accepted, wherein the outputs include the electricity being provided, and wherein the goal includes to maximize profits for the electricity generating facility from providing of the electricity.

[00444] QQQQ. The computer-implemented method of clause MMMM alone or of clause MMMM in combination with any or all other listed clauses, wherein the physical system is an energy generating facility, wherein the plurality of inter-related elements include at least one energy source within the energy generating facility and at least one energy storage mechanism within the energy generating facility, wherein the manipulatable control elements include one or more controls to determine whether to accept a request to supply a specified amount of energy at a current time and to determine to provide energy to the at least one energy storage mechanism at the current time if not accepted and to provide energy from the at least one energy source at the current time if accepted, wherein the outputs

include the energy being provided, and wherein the goal includes to maximize profits for the electricity generating facility from providing of the energy.

[00445] RRRR. The computer-implemented method of clause MMMM alone or of clause MMMM in combination with any or all other listed clauses, wherein the physical system is a vehicle, wherein the plurality of inter-related elements include a motor and a battery of the vehicle, wherein the manipulatable control elements include one or more controls to select whether at a current time to remove energy from the battery to power the motor or to add excess energy to the battery and how much energy to remove from the battery, wherein the outputs include effects of the motor to move the vehicle at the current time, and wherein the goal includes to move the vehicle at one or more specified speeds with a minimum of energy produced from the battery.

[00446] SSSS. The computer-implemented method of clause RRRR wherein the plurality of inter-related elements further includes an engine that is manipulatable to modify energy generated from the engine, wherein the manipulatable control elements further include one or more additional controls to determine how much energy to generate from the engine for use at least in part in adding the excess energy to the battery, and wherein the goal further includes to minimize use of fuel by the engine.

[00447] TTTT. The computer-implemented method of clause MMMM alone or of clause MMMM in combination with any or all other listed clauses, wherein the physical system includes product inventory at one or more locations, wherein the plurality of inter-related elements include one or more product sources that provide products and increase the inventory at the one or more locations and further include one or more product recipients that receive products and decrease the inventory at the one or more locations, wherein the manipulatable control elements include one or more first controls to select at a current time one or more first amounts of one or more products to request from the one or more product sources, and further include one or more second controls to select at the current time one or more second amounts of at least one product to provide to the one or more product recipients, wherein the outputs include products being provided from

the one or more locations to the one or more product recipients, and wherein the goal includes to maximize profit of an entity operating the one or more locations while maintaining the inventory at one or more specified levels.

[00448] UUUU. A non-transitory computer-readable medium having stored contents that cause one or more computing systems to perform a method, the method comprising:

generating, by the one or more computing systems, an executable decision module for use with a target system having a plurality of elements that are inter-related and that include one or more control elements with modifiable values, the generating including:

receiving, by the one or more computing systems and from one or more users, system information describing at least a portion of the target system using multiple rules that each has one or more conditions to evaluate and that specify restrictions involving the plurality of elements;

converting, by the one or more computing systems and to multiple constraints, the system information and information about a goal to be achieved during modifying the values of the control elements and information about sensor variables whose values are available during operation of the target system;

validating, by the one or more computing systems and based on multiple validation rules, the multiple constraints by testing at least one of controllability, observability, stability or goal completeness; and

converting, by the one or more computing systems, the validated constraints to coupled differential equations that represent a model for use in describing a state of the target system; and

providing the generated decision module with the coupled differential equations representing the model, to enable execution of the generated decision module to perform modifying of the values of the control elements during the execution.

[00449] VVVV. The non-transitory computer-readable medium of clause UUUU alone or of clause UUUU in combination with any or all other listed clauses, wherein the target system is a physical system having one or more outputs whose

values vary based at least in part on the values of the control elements, wherein the one or more computing systems are part of a collaborative distributed decision system, and wherein the stored contents include software instructions that, when executed, further cause the one or more computing systems to:

train, by the one or more computing systems, the model using state information for the plurality of elements for multiple times, to identify values for the trained model of one or more elements of the plurality that are not directly observable; and

test, by the one or more computing systems, performance of the model in controlling the at least portion of the physical system by simulating modifications of the values of the control elements to affect the outputs in expected manners.

[00450] WWW. The non-transitory computer-readable medium of clause UUUU alone or of clause UUUU in combination with any or all other listed clauses, wherein the target system includes one or more computing resources being protected from unauthorized operations, wherein the plurality of inter-related elements include one or more sources of attempts to perform operations, wherein the control elements include one or more controls to determine whether a change in authorization to a specified type of operation is needed and to select one or more actions to take to implement the change in authorization if so determined, and wherein the goal includes to minimize unauthorized operations that are performed.

[00451] XXXX. The non-transitory computer-readable medium of clause UUUU alone or of clause UUUU in combination with any or all other listed clauses, wherein the target system includes one or more information sources to be analyzed to determine a risk level from information of the one or more information sources, wherein the control elements include one or more controls to determine whether the risk level exceeds a specified threshold and to select one or more actions to take to mitigate the risk level, and wherein the goal includes to minimize the risk level.

[00452] YYYY. The non-transitory computer-readable medium of clause UUUU alone or of clause UUUU in combination with any or all other listed clauses,

wherein the target system includes one or more financial markets, wherein the plurality of inter-related elements include items that can be purchased and/or sold in the one or more financial markets, wherein the control elements include one or more controls to determine whether to purchase or sell particular items at particular times and to select one or more actions to initiate transactions to purchase or sell the particular items at the particular times, and wherein the goal includes to maximize profit while maintaining risk below a specified threshold.

[00453] ZZZZ. The non-transitory computer-readable medium of clause UUUU alone or of clause UUUU in combination with any or all other listed clauses, wherein the target system includes functionality to perform coding for medical procedures performed on humans, wherein the plurality of inter-related elements include a plurality of medical codes corresponding to a plurality of medical procedures, wherein the control elements include one or more controls to select particular medical codes to associate with particular medical procedures in specified circumstances, and wherein the goal includes to minimize errors in selected medical codes that cause revenue leakage.

CLAIMS

What is claimed is:

- [c1] 1. A computer-implemented method comprising:
- receiving, by a collaborative distributed decision system implemented by one or more computing systems, system information from one or more users that describes a physical system having a plurality of inter-related elements and having one or more outputs whose values vary based at least in part on values of one or more manipulatable control elements of the plurality, wherein the system information includes multiple rules that each has one or more conditions to evaluate and that specify restrictions involving the plurality of elements, the multiple rules including one or more soft rules whose conditions evaluate to one of three or more possible values under differing situations to represent varying degrees of uncertainty and further including additional rules whose conditions evaluate to either true or false under differing situations;
- receiving, by the collaborative distributed decision system, objective information that identifies a goal to be achieved during controlling of the physical system;
- obtaining, by the collaborative distributed decision system, sensor information that identifies current state information for at least one element of the plurality;
- converting, by the collaborative distributed decision system, the system information and the objective information and the sensor information to coupled differential equations that represent a model describing a current state of the physical system;
- performing, by the collaborative distributed decision system, a piecewise linear analysis of the coupled differential equations to identify one or more control actions that manipulate values of the one or more manipulatable control elements

and that provide a solution for the goal, wherein the provided solution is within a threshold amount of an optimal solution for the goal; and

initiating performance of the one or more control actions in the physical system to manipulate values of the one or more manipulatable control elements and to cause resulting changes in the values of the one or more outputs.

[c2] 2. The computer-implemented method of claim 1 wherein the physical system is an electricity generating facility, wherein the plurality of inter-related elements include multiple alternative electricity sources within the electricity generating facility, wherein the manipulatable control elements include one or more controls to determine whether to accept a request to supply a specified amount of electricity at a current time and to select which alternative electricity source to provide the specified amount of electricity at the current time if accepted, wherein the outputs include the electricity being provided, and wherein the goal is to maximize profits for the electricity generating facility from providing of the electricity.

[c3] 3. The computer-implemented method of claim 1 wherein the physical system is an energy generating facility, wherein the plurality of inter-related elements include at least one energy source within the energy generating facility and at least one energy storage mechanism within the energy generating facility, wherein the manipulatable control elements include one or more controls to determine whether to accept a request to supply a specified amount of energy at a current time and to determine to provide energy to the at least one energy storage mechanism at the current time if not accepted and to provide energy from the at least one energy source if accepted, wherein the outputs include the energy being provided, and wherein the goal is to maximize profits for the electricity generating facility from providing of the energy.

[c4] 4. The computer-implemented method of claim 1 wherein the physical system is a vehicle, wherein the plurality of inter-related elements include a motor

and a battery of the vehicle, wherein the manipulatable control elements include one or more controls to select whether at a current time to remove energy from the battery to power the motor or to add excess energy to the battery and how much energy to remove from the battery, wherein the outputs include effects of the motor to move the vehicle, and wherein the goal is to move the vehicle at one or more specified speeds with a minimum of energy produced from the battery.

[c5] 5. The computer-implemented method of claim 4 wherein the plurality of inter-related elements further includes an engine that is manipulatable to modify energy generated from the engine, wherein the manipulatable control elements further include one or more additional controls to determine how much energy to generate from the engine for use at least in part in adding the excess energy to the battery, and wherein the goal includes to minimize use of fuel by the engine.

[c6] 6. The computer-implemented method of claim 1 wherein the physical system includes product inventory at one or more locations, wherein the plurality of inter-related elements include one or more product sources that provide products and increase the inventory at the one or more locations and further include one or more product recipients that receive products and decrease the inventory at the one or more locations, wherein the manipulatable control elements include one or more first controls to select at a current time one or more first amounts of one or more products to request from the one or more product sources, and further include one or more second controls to select at the current time one or more second amounts of at least one product to provide to the one or more product recipients, wherein the outputs include products being provided from the one or more locations to the one or more product recipients, and wherein the goal is to maximize profit of an entity operating the one or more locations while maintaining the inventory at one or more specified levels.

[c7] 7. The computer-implemented method of claim 1 wherein the additional rules include one or more absolute rules that specify non-modifiable restrictions

that are requirements regarding operation of the physical system, and further include one or more hard rules that specify restrictions regarding operation of the physical system that can be modified in specified situations.

[c8] 8. The computer-implemented method of claim 1 wherein the system information and the sensor information and the goal are associated with a first decision module of a plurality of decision modules of the collaborative distributed decision system, wherein the plurality of decision modules each has a distinct model describing the current state of the physical system that is based on a distinct set of system information and sensor information and one or more goals, and wherein the method further comprises determining an aggregated model that is based on the distinct models for the plurality of decision modules and that simultaneously provides solutions for the goals of each of the plurality of decision modules, wherein the aggregated model has one or more associated control actions to perform in the physical system to manipulate values of the one or more manipulatable control elements in a specified manner.

[c9] 9. The computer-implemented method of claim 8 further comprising determining the aggregated model by successively synchronizing, for each of the plurality of decision modules, the model for that decision module with a shared model describing the current state of the physical system maintained for an additional virtual decision module, and wherein the determined one or more associated control actions are based on results of the successive synchronizing.

[c10] 10. The computer-implemented method of claim 9 wherein the plurality of decision modules each has an associated Hamiltonian function that expresses the model for that decision module, wherein the shared model for the additional virtual decision module is expressed with an additional Hamiltonian function, and wherein the synchronizing, for each of the plurality of decision modules, of the model for that decision module with the shared model for the additional virtual decision module includes:

creating a combined Hamiltonian function that includes the Hamiltonian function associated with that decision module and the additional Hamiltonian function for the additional virtual decision module;

determining a Pareto equilibrium for the combined Hamiltonian function; and

before performing the synchronizing for a next of the plurality of decision modules, updating the shared model and the model specific to the decision module based on results of the determined Pareto equilibrium.

[c11] 11. The computer-implemented method of claim 8 wherein the model for each of the plurality of decision modules is expressed with a Hamiltonian function specific to that decision module, and wherein the method further comprises determining the aggregated model in a distributed manner by, for one of the plurality of decision modules:

successively synchronizing, for at least some of the other decision modules of the plurality, the model for the other decision module with the model for the one decision module by:

obtaining the Hamiltonian function specific to the other decision module;

creating a combined Hamiltonian function that includes the Hamiltonian function specific to the other decision module and the Hamiltonian function specific to the one decision module;

determining a Pareto equilibrium for the combined Hamiltonian function; and

before performing the synchronizing for a next of the at least some other decision modules of the plurality, updating the models for each of the one decision module and the other decision module based on results of the determined Pareto equilibrium; and

repeating the successive synchronizing until the plurality of decision models converge on a shared model describing the current state of the physical system that simultaneously provides solutions for the goals of each of the plurality of

decision modules and that is associated with the determined one or more additional control actions.

[c12] 12. The computer-implemented method of claim 8 further comprising, for each of the plurality of decision modules other than the first decision module:

receiving, by the collaborative distributed decision system, the system information for the decision module from one or more users to describe the physical system, wherein the system information for the decision module includes multiple rules for the decision module that are distinct from the multiple rules for the first decision module;

converting, by the collaborative distributed decision system, the system information for the decision module and the sensor information for the decision module and the one or more goals for the decision module to coupled differential equations for the decision module that are distinct from the coupled differential equations for the first decision module and that represent the model for the decision module;

performing, by the collaborative distributed decision system, a piecewise linear analysis of the coupled differential equations for the decision module to identify one or more control actions for the decision module that provide a solution for the decision module for the one or more goals of the decision module, wherein the provided solution for the decision module is within a threshold amount of an optimal solution for the one or more goals of the decision module; and

providing information about the one or more control actions for the decision module.

[c13] 13. The computer-implemented method of claim 8 further comprising, for each of the plurality of decision modules other than the first decision module:

receiving, by the collaborative distributed decision system, the one or more goals for the decision module that are to be achieved during the controlling of the physical system, wherein the one or more goals for the decision module are distinct from the goal for the first decision module;

converting, by the collaborative distributed decision system, the system information for the decision module and the sensor information for the decision module and the one or more goals for the decision module to coupled differential equations for the decision module that are distinct from the coupled differential equations for the first decision module and that represent the model for the decision module;

performing, by the collaborative distributed decision system, a piecewise linear analysis of the coupled differential equations for the decision module to identify one or more control actions for the decision module that provide a solution for the decision module for the one or more goals of the decision module, wherein the provided solution for the decision module is within a threshold amount of an optimal solution for the one or more goals of the decision module; and

providing information about the one or more control actions for the decision module.

[c14] 14. The computer-implemented method of claim 8 further comprising, for each of the plurality of decision modules other than the first decision module:

obtaining, by the collaborative distributed decision system, sensor information for the decision module that identifies current state information for at least one element of the plurality of inter-related elements for the physical system, wherein the sensor information for the decision module is distinct from the sensor information for the first decision module;

converting, by the collaborative distributed decision system, the system information for the decision module and the sensor information for the decision module and the one or more goals for the decision module to coupled differential equations for the decision module that are distinct from the coupled differential equations for the first decision module and that represent the model for the decision module;

performing, by the collaborative distributed decision system, a piecewise linear analysis of the coupled differential equations for the decision module to identify one or more control actions for the decision module that provide a solution

for the decision module for the one or more goals of the decision module, wherein the provided solution for the decision module is within a threshold amount of an optimal solution for the one or more goals of the decision module; and

providing information about the one or more control actions for the decision module.

[c15] 15. The computer-implemented method of claim 8 wherein the aggregated model that simultaneously provides solutions for the goals of each of the plurality of decision modules corresponds to a first time, and wherein the method further comprises, for each of multiple additional successive times after the first time, updating the aggregated model for the additional successive time by:

obtaining, by the collaborative distributed decision system, additional sensor information that identifies current state information at the additional successive time for one or more elements of the plurality;

creating, by the collaborative distributed decision system, additional coupled differential equations from the aggregated model and from the additional sensor information;

performing, by the collaborative distributed decision system, a piecewise linear analysis of the additional coupled differential equations to attempt to identify an additional solution at the additional successive time that simultaneously provides solutions for the goals of each of the plurality of decision modules; and

if the additional solution at the additional successive time is identified, updating the aggregated model to reflect the additional solution.

[c16] 16. The computer-implemented method of claim 15 further comprising, during one of the multiple additional successive times, modifying the plurality of decision modules to include one or more additional decision modules that each has a model describing the current state of the physical system that is different from the models of other of the plurality of decision modules and that includes a distinct set of system information and sensor information and one or more goals, and wherein the updating of the aggregated model after the one additional

successive time includes using the one or more additional decision modules as part of the plurality of decision modules.

[c17] 17. The computer-implemented method of claim 15 further comprising, during one of the multiple additional successive times, modifying the plurality of decision modules to remove one or more decision modules from the plurality of decision modules, and wherein the updating of the aggregated model after the one additional successive time includes using the plurality of decision modules without the one or more removed decision modules.

[c18] 18. The computer-implemented method of claim 15 further comprising:
during one of the multiple additional successive times, losing an ability to communicate with one or more decision modules of the plurality of decision modules; and

for each of one or more further successive times of the multiple additional successive times after the one additional successive time and while the ability to communicate with the one or more decision modules is unavailable, and for each of the one or more decision modules, individually updating the aggregated model for the further successive time using additional sensor information for the further successive time and using the distinct set of system information and one or more goals for that decision module.

[c19] 19. The computer-implemented method of claim 18 wherein the losing of the ability to communicate with the one or more decision modules is based on one or more unreliable network connections to the one or more decision modules, wherein at least some decision modules of the plurality retain the ability to communicate after the one additional successive time, and wherein the method further comprises, for each of the one or more further successive times, performing the updating for the further successive time of the aggregated model by using at least some decision modules without the one or more decision modules.

- [c20] 20. The computer-implemented method of claim 1 further comprising:
- storing, by the collaborative distributed decision system and for a first time corresponding to the performing of the piecewise linear analysis, a model describing a current state of the physical system at the first time that includes the goal information and the system information and information about the resulting changes in the values of the one or more outputs from the performance of the one or more control actions in the physical system; and
- at one or more later second times after the first time, updating the stored model by:
- obtaining, by the one or more computing systems, additional sensor information that identifies current state information at the second time for one or more elements of the plurality;
- creating, by the collaborative distributed decision system, additional coupled differential equations from the stored model and from the additional sensor information;
- performing, by the collaborative distributed decision system, a piecewise linear analysis of the additional coupled differential equations to attempt to identify an additional solution for the goal at the second time; and
- if the additional solution at the second time is identified, updating the stored model to reflect the additional solution for the second time.

- [c21] 21. The computer-implemented method of claim 20 wherein the one or more later second times include multiple additional successive times after the first time, and wherein the method further comprises identifying patterns in changes over the multiple additional successive times of the stored model, and using the identified patterns to control the physical system after the multiple additional successive times.

- [c22] 22. The computer-implemented method of claim 20 wherein the one or more later second times include multiple additional successive times after the first time, wherein the method further comprises modifying the multiple rules, during

one of the multiple additional successive times, and wherein the updating of the stored model after the one additional successive time includes using the modified rules.

[c23] 23. The computer-implemented method of claim 20 wherein the one or more later second times include multiple additional successive times after the first time, wherein the updating of the stored model for each of the multiple additional successive times is performed in a real-time manner after the obtaining of the additional sensor information for that additional successive time, and wherein the method further comprises providing real-time control of the physical system by performing one or more further control actions at each of the multiple additional successive times based on the stored model for that additional successive time.

[c24] 24. The computer-implemented method of claim 20 wherein the attempt to identify the additional solution for the goal at one of the second times does not succeed, and wherein the method further comprises:

determining, by the collaborative distributed decision system, at least one of the multiple rules to relax by modifying at least one of the specified restrictions corresponding to the determined at least one rules;

creating, by the collaborative distributed decision system, modified system information with the modified at least one specified restriction;

converting, by the collaborative distributed decision system, the modified system information and the objective information and the additional sensor information to further coupled differential equations;

performing, by the collaborative distributed decision system, a piecewise linear analysis of the further coupled differential equations to identify the one or more additional control actions that manipulate values of the one or more manipulatable control elements and that provide the additional solution for the goal at the one second time, wherein the provided additional solution is within a threshold amount of an optimal solution for the goal at the one second time; and

updating the stored model to reflect the additional solution for the one second time.

[c25] 25. The computer-implemented method of claim 24 wherein the modifying of the at least one specified restriction includes suspending one or more of the at least one specified restrictions for at least a period of time.

[c26] 26. The computer-implemented method of claim 24 wherein the attempt to identify the additional solution for the goal at the one second time does not succeed due to the restrictions involving the plurality of elements being over-constrained and not having any solution.

[c27] 27. A non-transitory computer-readable medium having stored contents that cause one or more computing systems of a collaborative distributed decision system to perform a method, the method comprising:

receiving, by the one or more computing systems, system information from one or more users that describes a target system having a plurality of elements that are inter-related and that include one or more manipulatable control elements with modifiable values, wherein the system information includes multiple rules that each has one or more conditions to evaluate and that specify restrictions involving the plurality of elements, the multiple rules including one or more soft rules whose conditions evaluate to one of three or more possible values under differing situations to represent varying degrees of uncertainty and further including additional rules whose conditions evaluate to either true or false under differing situations;

receiving, by the one or more computing systems, objective information that identifies a goal to be achieved while modifying the values of the one or more manipulatable control elements;

obtaining, by the one or more computing systems, sensor information that identifies information about a physical state of at least one element of the plurality at a specified time;

converting, by the one or more computing systems, the system information and the objective information and the sensor information to coupled differential equations that represent a model describing a state of the target system at the specified time;

performing, by the one or more computing systems, a piecewise linear analysis of the coupled differential equations to identify one or more values of the one or more manipulatable control elements that provide a solution for the goal for the specified time, wherein the provided solution is within a threshold amount of an optimal solution for the goal for the specified time; and

providing information about the identified one or more values, to enable modification of the one or more manipulatable control elements for the specified time to have the identified one or more values.

[c28] 28. The non-transitory computer-readable medium of claim 27 wherein the target system is a physical system having one or more outputs whose values vary based at least in part on the values of the manipulatable control elements, and wherein the stored contents include software instructions that, when executed, further cause the one or more computing systems to initiate performance of one or more control actions in the physical system to modify the one or more manipulatable control elements to have the identified one or more values and to cause resulting changes in the values of the one or more outputs.

[c29] 29. The non-transitory computer-readable medium of claim 27 wherein the target system includes one or more computing resources being protected from unauthorized operations, wherein the plurality of inter-related elements include one or more sources of attempts to perform operations, wherein the manipulatable control elements include one or more controls to determine whether a change in authorization to a specified type of operation is needed and to select one or more actions to take to implement the change in authorization if so determined, and wherein the goal is to minimize unauthorized operations that are performed.

[c30] 30. The non-transitory computer-readable medium of claim 27 wherein the target system includes one or more information sources to be analyzed to determine a risk level from information of the one or more information sources, wherein the manipulatable control elements include one or more controls to determine whether the risk level exceeds a specified threshold and to select one or more actions to take to mitigate the risk level, and wherein the goal is to minimize the risk level.

[c31] 31. The non-transitory computer-readable medium of claim 27 wherein the target system includes one or more financial markets, wherein the plurality of inter-related elements include items that can be purchased and/or sold in the one or more financial markets, wherein the manipulatable control elements include one or more controls to determine whether to purchase or sell particular items at particular times and to select one or more actions to initiate transactions to purchase or sell the particular items at the particular times, and wherein the goal is to maximize profit while maintaining risk below a specified threshold.

[c32] 32. The non-transitory computer-readable medium of claim 27 wherein the target system includes functionality to perform coding for medical procedures performed on humans, wherein the plurality of inter-related elements include a plurality of medical codes corresponding to a plurality of medical procedures, wherein the manipulatable control elements include one or more controls to select particular medical codes to associate with particular medical procedures in specified circumstances, and wherein the goal is to minimize errors in selected medical codes that cause revenue leakage.

[c33] 33. A system comprising:
one or more processors of one or more computing systems; and
a memory containing a plurality of modules of a collaborative distributed decision system that, when executed by at least one of the one or more

processors, cause the one or more processors to implement the collaborative distributed decision system, the plurality of modules including:

- a user interface module that generates a graphical user interface for use by one or more users and that receives, from the one or more users, system information that describes a physical target system having a plurality of inter-related elements and having one or more outputs whose values vary based at least in part on values of one or more manipulatable control elements of the plurality, wherein the system information includes multiple rules that each has one or more conditions to evaluate and that specify restrictions involving the plurality of elements;

- one or more sensor event modules that obtain sensor information identifying current state information for at least one element of the plurality;

- a target system representation builder module that converts the system information to a plurality of constraints, and that generates, from the plurality of constraints and the sensor information and a goal to be achieved during controlling of the physical target system, coupled differential equations that represent a model describing a current state of the physical target system;

- an optimization determination module that performs a piecewise linear analysis of the coupled differential equations to identify one or more control actions that manipulate values of the one or more manipulatable control elements and that provide a solution for the goal, wherein the provided solution is within a threshold amount of an optimal solution for the goal; and

- one or more output modules that provide information about the one or more control actions, to enable performance of the one or more control actions in the physical system to manipulate values of the one or more manipulatable control elements and to cause resulting changes in the values of the one or more outputs.

[c34]

34. The system of claim 33 further comprising:

a first decision module that includes the system information and the sensor information and the goal, and that stores the model describing the current state of the physical target system for the first decision module;

multiple other decision modules that each has a distinct model describing the current state of the physical target system that is based on a distinct set of system information and sensor information and one or more goals; and

a stored aggregated model that is based on the model for the first decision module and the distinct models for the multiple other decision modules and that simultaneously provides solutions for the goals of the first decision module and of each of the multiple other decision modules, wherein the aggregated model has one or more associated control actions to perform in the physical target system to manipulate values of the one or more manipulatable control elements in a specified manner

[c35] 35. The system of claim 33 further comprising:

the one or more manipulatable control elements; and

one or more effectuators to manipulate the values of the one or more manipulatable control elements and to cause the resulting changes in the values of the one or more outputs.

[c36] 36. A computer-implemented method comprising:

obtaining, by one or more computing systems of a collaborative distributed decision system, coupled differential equations that represent a current state of a physical system and that are generated from system information and objective information and sensor information, wherein the physical system has a plurality of inter-related elements and has one or more outputs whose values vary based at least in part on values of one or more manipulatable control elements of the plurality, wherein the system information is supplied by one or more users to describe the physical system and includes multiple rules that each has one or more conditions to evaluate and that specify restrictions involving the plurality of elements, wherein the objective information identifies a goal to be achieved during

controlling of the physical system, and wherein the sensor information identifies current state information for at least one element of the plurality and partial initial state information for the physical system at an earlier time;

performing, by the one or more computing systems, a piecewise linear analysis of the coupled differential equations to identify one or more control actions to take in the physical system that manipulate values of the one or more manipulatable control elements and that provide a solution for the goal within a threshold amount of an optimal solution for the goal, wherein the performing of the piecewise linear analysis includes:

dividing a time window from the earlier time to the specified time into a succession of a plurality of time slices that each, other than a first time slice of the succession, overlaps at least in part with a prior time slice of the succession;

evaluating, based on the partial initial state information, the coupled differential equations to identify an initial solution for the goal for the first time slice that includes simulating effects of manipulating the one or more manipulatable control elements to one or more initial values, and storing a model describing a state of the physical system for the first time slice that includes the simulated effects of the manipulating to the one or more initial values;

for each time slice of the succession after the first time slice, updating the stored model for the prior time slice to reflect an additional solution for the goal for the time slice that includes simulating effects of further manipulating the one or more manipulatable control elements to one or more additional values; and

after updating the stored model to reflect the additional solution for the goal for a last time slice of the succession, further updating the stored model for the last time slice to reflect a further solution for the goal for a next time period after the time window based at least in part on the identified current state information, wherein the further solution includes the identified one or more control actions to take in the physical system for a current time; and

providing information about the identified one or more control actions, to enable actions to be taken in the physical system for the current time to affect the outputs based on the identified one or more control actions.

[c37] 37. The computer-implemented method of claim 36 wherein the physical system is an electricity generating facility, wherein the plurality of inter-related elements include multiple alternative electricity sources within the electricity generating facility, wherein the manipulatable control elements include one or more controls to determine whether to accept a request to supply a specified amount of electricity at the current time and to select which alternative electricity source to provide the specified amount of electricity at the current time if accepted, wherein the outputs include the electricity being provided, and wherein the goal includes to maximize profits for the electricity generating facility from providing of the electricity.

[c38] 38. The computer-implemented method of claim 36 wherein the physical system is an energy generating facility, wherein the plurality of inter-related elements include at least one energy source within the energy generating facility and at least one energy storage mechanism within the energy generating facility, wherein the manipulatable control elements include one or more controls to determine whether to accept a request to supply a specified amount of energy at the current time and to determine to provide energy to the at least one energy storage mechanism at the current time if not accepted and to provide energy from the at least one energy source at the current time if accepted, wherein the outputs include the energy being provided, and wherein the goal includes to maximize profits for the electricity generating facility from providing of the energy.

[c39] 39. The computer-implemented method of claim 36 wherein the physical system is a vehicle, wherein the plurality of inter-related elements include a motor and a battery of the vehicle, wherein the manipulatable control elements include one or more controls to select whether at the current time to remove energy from

the battery to power the motor or to add excess energy to the battery and how much energy to remove from the battery, wherein the outputs include effects of the motor to move the vehicle at the current time, and wherein the goal includes to move the vehicle at one or more specified speeds with a minimum of energy produced from the battery.

[c40] 40. The computer-implemented method of claim 39 wherein the plurality of inter-related elements further includes an engine that is manipulatable to modify energy generated from the engine, wherein the manipulatable control elements further include one or more additional controls to determine how much energy to generate from the engine for use at least in part in adding the excess energy to the battery, and wherein the goal further includes to minimize use of fuel by the engine.

[c41] 41. The computer-implemented method of claim 36 wherein the physical system includes product inventory at one or more locations, wherein the plurality of inter-related elements include one or more product sources that provide products and increase the inventory at the one or more locations and further include one or more product recipients that receive products and decrease the inventory at the one or more locations, wherein the manipulatable control elements include one or more first controls to select at the current time one or more first amounts of one or more products to request from the one or more product sources, and further include one or more second controls to select at the current time one or more second amounts of at least one product to provide to the one or more product recipients, wherein the outputs include products being provided from the one or more locations to the one or more product recipients, and wherein the goal includes to maximize profit of an entity operating the one or more locations while maintaining the inventory at one or more specified levels.

[c42] 42. The computer-implemented method of claim 36 wherein the stored model for each of the time slices of the succession is expressed with a

Hamiltonian function specific to the time slice, and wherein each updating of the stored model for a prior time slice to reflect an additional solution for the goal includes modifying the Hamiltonian function expressed by the stored model for the prior time slice.

[c43] 43. The computer-implemented method of claim 36 wherein the updated stored model for the last time slice is expressed with a Hamiltonian function, and wherein the further updating of the stored model for the last time slice to reflect the further solution for the goal for the next time period includes modifying the Hamiltonian function based at least in part on the identified current state information.

[c44] 44. The computer-implemented method of claim 36 wherein the evaluation of the coupled differential equations to identify the initial solution for the goal for the first time slice and the updating for each time slice of the succession after the first time slice of the stored model is performed to train the updated stored model for the last time slice to reflect values of at least some of the plurality of inter-related elements for the current time that include one or more elements whose values are not directly observable, and wherein training of the updated stored model for the last time slice enables the further updating to reflect the further solution for the goal for the next time period.

[c45] 45. The computer-implemented method of claim 36 further comprising, for each of multiple additional times after the current time, adapting a current copy of the stored model to reflect the additional time by:

obtaining additional sensor information that identifies state information at the additional time for one or more elements of the plurality;

determining, by the one or more computing systems, if an updated copy of the stored model can be generated for the additional time by attempting to identify another solution for the goal for the additional time based at least in part on the additional sensor information, wherein the another solution, if identified, includes

one or more further control actions to take in the physical system for the additional time; and

if the updated copy of the stored model can be generated for the additional time, generating and storing the updated copy of the stored model, and providing information about one or more additional control actions to perform in the physical system for the additional time to further manipulate the manipulatable control elements in a specified manner.

[c46] 46. The computer-implemented method of claim 45 further comprising, for each of the multiple additional times and if the updated copy of the stored model for the additional time is not generated, providing information about one or more further associated control actions to perform in the physical system for that additional time to further manipulate the manipulatable control elements in a specified manner, wherein the one or more further associated control actions are based on the current copy of the stored model before updating for the additional time.

[c47] 47. The computer-implemented method of claim 46 wherein generating of the updated copy of the stored model for one of the multiple additional times includes a deadline for the generating to enable real-time control of the physical system to be performed based on performing control actions in the physical system for the one additional time, and wherein the generating of the updated copy of the stored model for the one additional time fails to complete before the deadline, such that the one or more further associated control actions for that one additional time are performed in the physical system for that one additional time to further manipulate the manipulatable control elements in a specified manner.

[c48] 48. The computer-implemented method of claim 45 wherein the updated copy of the stored model for one of the multiple additional times is not generated in a first attempt, and wherein the method further comprises generating the updated copy of the stored model for the one additional time during a second attempt by:

determining, by the one or more computing systems, at least one of the multiple rules to temporarily relax by modifying at least one of the specified restrictions corresponding to the determined at least one rule;

generating, by the one or more computing systems, additional coupled differential equations that represent a current state of the physical system for the one additional time based at least in part on the modified at least one specified restrictions for the determined at least one rule;

performing, by the one or more computing systems, a further piecewise linear analysis of the generated additional coupled differential equations to identify the another solution for the goal for the one additional time,

generating and storing the updated copy of the stored model for the one additional time; and

providing information about the one or more additional control actions of the updated copy of the stored model for the one additional time to perform in the physical system for the one additional time to further manipulate the manipulatable control elements in a specified manner.

[c49] 49. The computer-implemented method of claim 48 wherein the multiple rules include one or more absolute rules that specify non-modifiable restrictions that are requirements regarding operation of the physical system, and further include one or more hard rules that specify restrictions regarding operation of the physical system that can be modified in specified situations, and wherein each determined at least one rule is one of the hard rules.

[c50] 50. The computer-implemented method of claim 48 wherein the multiple rules include one or more soft rules whose conditions evaluate to one of three or more possible values under differing situations to represent varying degrees of uncertainty and further include additional rules whose conditions evaluate to either true or false under differing situations, and wherein one or more of the determined at least one rules are from the soft rules.

[c51] 51. The computer-implemented method of claim 45 wherein the determining if the updated copy of the stored model can be generated for one of the multiple additional times includes generating values for one or more model error measurements based at least in part on the current copy of the stored model for the one additional time, and includes determining that at least one of the generated values exceeds an error threshold, and

wherein the method further comprises generating the updated copy of the stored model for the one additional time by:

evaluating, by the one or more computing systems, the generated values for the one or more model error measurements to determine at least one of the multiple rules that is incorrect;

modifying, by the one or more computing systems, the determined at least one rule in a manner expected to reduce values for the one or more model error measurements;

generating, by the one or more computing systems, additional coupled differential equations that represent a current state of the physical system for the one additional time based at least in part on the modified determined at least one rule;

performing, by the one or more computing systems, a further piecewise linear analysis of the generated additional coupled differential equations to identify the another solution for the goal for the one additional time,

generating and storing the updated copy of the stored model for the one additional time; and

providing information about the one or more additional control actions of the updated copy of the stored model for the one additional time to perform in the physical system for the one additional time to further manipulate the manipulatable control elements in a specified manner.

[c52] 52. The computer-implemented method of claim 51 wherein the another solution identified for the goal for two or more of the additional times each has an associated error measurement within a defined threshold relative to an optimal

solution for the goal for that additional time, and wherein the one or more model error measurements are based on a rate of change of one or more of:

Hamiltonian functions expressed by two or more copies of the model for two or more times;

amounts of entropy included in two or more copies of the model for two or more times;

values of variables associated with the plurality of inter-related elements in state information for the physical system for two or more times; or

a reduction in the associated error measurements for the another solutions for the two or more additional times.

[c53] 53. The computer-implemented method of claim 45 wherein the determining if the updated copy of the stored model can be generated for one of the multiple additional times includes:

generating, by the one or more computing systems, the updated copy of the stored model for the one additional time;

generating, by the one or more computing systems, values for one or more model error measurements for the generated updated copy of the stored model for the one additional time;

determining, by the one or more computing systems, that at least one of the generated values exceeds an error threshold; and

replacing, by the one or more computing systems, the generated updated copy of the stored model for the one additional time with a new generated updated copy of the stored model for the one additional time, by causing a new copy of the model for the one additional time to be generated without using any past copies of the model, and storing the generated new copy of the model for the one additional time.

[c54] 54. The computer-implemented method of claim 36 further comprising modifying the multiple rules during one of the multiple additional times, and

wherein updating of copies of the stored model after the one additional time includes using the modified rules.

[c55] 55. The computer-implemented method of claim 36 further comprising, before the dividing of the time window into the succession of time slices, determining, by the one or more computing systems, at least one of a size of the time slices or a size of the time window.

[c56] 56. The computer-implemented method of claim 55 wherein the determining of the at least one of the size of the time slices or the size of the time window includes evaluating multiple test sizes for the at least one of the size of the time slices or the size of the time window, and selecting, based at least in part on the evaluating, one or more of the multiple sizes to use for the determined at least one size of the time slices or size of the time window.

[c57] 57. The computer-implemented method of claim 55 further comprising generating a Hamiltonian function to express a copy of the model of the state of the physical system before the dividing of the time window into the succession of time slices, and wherein the determining of the at least one of the size of the time slices or the size of the time window includes performing a symbolic computation analysis of the Hamiltonian function to identify one or more preferred sizes to use for the determined at least one size of the time slices or size of the time window.

[c58] 58. The computer-implemented method of claim 36 wherein the providing of the information about the identified one or more control actions includes performing, by the one or more computing systems, the actions in the physical system to affect the outputs by manipulating the manipulatable control elements in specified manners for the identified one or more control actions.

[c59] 59. A non-transitory computer-readable medium having stored contents that cause one or more computing systems to perform a method, the method comprising:

obtaining, by the one or more computing systems, coupled differential equations that represent a state of a target system at a specified time and that are generated from system information and objective information and sensor information, wherein the target system has a plurality of elements that are inter-related and that include one or more control elements with modifiable values, wherein the system information is supplied by one or more users to describe the physical system and includes restrictions involving the plurality of elements, wherein the objective information identifies a goal to be achieved based at least in part on modifying the values of the control elements, and wherein the sensor information identifies state information for the specified time for at least one element of the plurality;

performing, by the one or more computing systems, a piecewise linear analysis of the coupled differential equations to identify a solution for the goal for the specified time within a threshold amount of an optimal solution for the goal, wherein the identified solution has one or more associated control actions that modify at least one value of at least one of the control elements in a specified manner, and wherein the performing of the piecewise linear analysis includes:

dividing a time window from an earlier time to the specified time into a succession of a plurality of time slices;

evaluating, based on initial state information for the earlier time, the coupled differential equations to identify an initial solution for the goal for a first time slice of the succession that includes simulating effects of modifying one or more values of the one or more manipulatable control elements in a specified initial manner, and storing a model describing a state of the target system for the first time slice that includes the simulated effects of the modifying of the one or more values;

for each time slice of the succession after the first time slice, updating the stored model for a prior time slice to reflect an additional solution for

the goal for the time slice that includes simulating effects of further modifying one or more values of the one or more manipulatable control elements; and

after updating the stored model to reflect the additional solution for the goal for a last time slice of the succession, further updating the stored model for the last time slice to reflect a further solution for the goal for a next time period after the time window based at least in part on the identified state information for the specified time, wherein the further solution includes the one or more associated control actions; and

providing information about the one or more associated control actions, to enable modification of the at least one value of the at least one control element for the specified time based on the one or more associated control actions.

[c60] 60. The non-transitory computer-readable medium of claim 59 wherein the target system is a physical system having one or more outputs whose values vary based at least in part on the values of the control elements, wherein the one or more computing systems are part of a collaborative distributed decision system, and wherein the stored contents include software instructions that, when executed, further cause the one or more computing systems to initiate performance of the one or more associated control actions in the physical system to modify the at least one value of the at least one control element and to cause resulting changes in the values of the one or more outputs for the specified time.

[c61] 61. The non-transitory computer-readable medium of claim 59 wherein the target system includes one or more computing resources being protected from unauthorized operations, wherein the plurality of inter-related elements include one or more sources of attempts to perform operations, wherein the control elements include one or more controls to determine whether a change in authorization to a specified type of operation is needed and to select one or more actions to take to implement the change in authorization if so determined, and wherein the goal includes to minimize unauthorized operations that are performed.

[c62] 62. The non-transitory computer-readable medium of claim 59 wherein the target system includes one or more information sources to be analyzed to determine a risk level from information of the one or more information sources, wherein the control elements include one or more controls to determine whether the risk level exceeds a specified threshold and to select one or more actions to take to mitigate the risk level, and wherein the goal includes to minimize the risk level.

[c63] 63. The non-transitory computer-readable medium of claim 59 wherein the target system includes one or more financial markets, wherein the plurality of inter-related elements include items that can be purchased and/or sold in the one or more financial markets, wherein the control elements include one or more controls to determine whether to purchase or sell particular items at particular times and to select one or more actions to initiate transactions to purchase or sell the particular items at the particular times, and wherein the goal includes to maximize profit while maintaining risk below a specified threshold.

[c64] 64. The non-transitory computer-readable medium of claim 59 wherein the target system includes functionality to perform coding for medical procedures performed on humans, wherein the plurality of inter-related elements include a plurality of medical codes corresponding to a plurality of medical procedures, wherein the control elements include one or more controls to select particular medical codes to associate with particular medical procedures in specified circumstances, and wherein the goal includes to minimize errors in selected medical codes that cause revenue leakage.

[c65] 65. A system comprising:
one or more processors of one or more computing systems; and
one or more modules that, when executed by at least one of the one or more processors, cause the one or more processors to determine one or more

control actions to perform as part of controlling a physical system, the determining of the one or more control actions including:

obtaining coupled differential equations that represent a state of a physical system for a specified time and that are generated from system information and objective information and sensor information, wherein the physical system has a plurality of inter-related elements and has one or more outputs whose values vary based at least in part on values of one or more manipulatable control elements of the plurality, wherein the system information is supplied by one or more users to describe the physical system and includes restrictions involving the plurality of elements, wherein the objective information identifies a goal to be achieved during controlling of the physical system, and wherein the sensor information identifies state information for the specified time for at least one element of the plurality;

performing a first piecewise linear analysis of the coupled differential equations to train a model that describes a state of the physical system for the specified time and that includes values of at least some of the plurality of elements for the specified time, wherein the performing of the first piecewise linear analysis includes simulating effects of manipulating the one or more manipulatable control elements for each of one or more prior time periods before the specified time while satisfying the goal for the one or more prior time periods;

performing a second piecewise linear analysis of the coupled differential equations to identify one or more control actions to take in the physical system for the specified time that manipulate values of the one or more manipulatable control elements and that provide a solution for the goal for the specified time, wherein the performing of the piecewise linear analysis includes updating the model to reflect the solution for the goal for the specified time period based at least in part on the identified state information for the specified time; and

providing information about the identified one or more control actions, to enable actions to be taken in the physical system for the specified time to affect the outputs based on the identified one or more control actions.

[c66] 66. The system of claim 65 wherein the one or more modules are part of a collaborative distributed decision system and include software instructions for execution by the at least one processor, wherein the provided solution reflected in the updated model is within a threshold amount of an optimal solution for the goal for the specified time, and wherein the system further comprises one or more effectuators to perform the actions in the physical system for the specified time by manipulating the values of the one or more manipulatable control elements in specified manners for the identified one or more control actions to affect the outputs.

[c67] 67. The system of claim 66 wherein the performing of the first piecewise linear analysis of the coupled differential equations to train the model further includes:

for a time window from an earlier time to the specified time that includes the one or more prior time periods, dividing the time window into a succession of a plurality of time slices that each, other than a first time slice of the succession, overlaps at least in part with a prior time slice of the succession;

evaluating, based on initial state information for the earlier time, an initial version of the coupled differential equations to identify an initial solution for the goal for the first time slice that includes simulating effects of manipulating the one or more manipulatable control elements to one or more initial values, and storing an initial version of the model that describes the state of the physical system for the first time slice and includes the simulated effects of the manipulating to the one or more initial values; and

for each time slice of the succession after the first time slice, updating a version of the stored model from the prior time slice to reflect an additional solution for the goal for the time slice that includes simulating effects of further manipulating the one or more manipulatable control elements to one or more additional values, and

wherein the trained model is a version of the model after the updating to reflect the additional solution for the goal for a last time slice of the succession.

[c68] 68. The system of claim 65 wherein the one or more modules consist of one or more means for performing the determining of the one or more control actions to perform as part of controlling the physical system.

[c69] 69. A computer-implemented method comprising:
obtaining, by one or more computing systems of a collaborative distributed decision system, and for each of multiple decision modules that collectively describe a state of a physical system at a current time, a distinct model associated with the decision module that is represented by a distinct set of coupled differential equations, wherein the physical system has a plurality of inter-related elements and has one or more outputs whose values vary based at least in part on one or more manipulatable control elements of the plurality, and wherein each of the multiple decision modules' associated distinct model describes state information for the current time of at least of a portion of the physical system and reflects a solution for a specified goal of the decision module that is to be achieved during controlling of the physical system;

generating, by the one or more computing systems, and for each of the multiple decision modules, a consensus shared model for the decision module for the current time by determining a Pareto equilibrium for the model associated with the decision module and for an intermediate shared model representing the models associated with some other of the multiple decision modules, wherein the consensus shared model for the decision module simultaneously provides solutions for the goals of the decision module and the some other decision modules such that the provided solutions have an associated error measurement within a defined threshold relative to a global optimal solution, and wherein the consensus shared model for the decision module has one or more associated control actions to perform in the physical system to manipulate the manipulatable control elements in a specified manner for the provided solutions; and

providing information about the one or more associated control actions of one or more of the consensus shared models, to enable actions to be taken in the

physical system to affect the outputs based on the one or more associated control actions of the one or more consensus shared models.

[c70] 70. The computer-implemented method of claim 69 wherein the multiple decision modules collectively include at least one overall objective to be achieved during controlling of the physical system, wherein the physical system is an electricity generating facility, wherein the plurality of inter-related elements include multiple alternative electricity sources within the electricity generating facility, wherein the manipulatable control elements include one or more controls to determine whether to accept a request to supply a specified amount of electricity at the current time and to select which alternative electricity source to provide the specified amount of electricity at the current time if accepted, wherein the outputs include the electricity being provided, and wherein the at least one overall objective includes to maximize profits for the electricity generating facility from providing of the electricity.

[c71] 71. The computer-implemented method of claim 69 wherein the multiple decision modules collectively include at least one overall objective to be achieved during controlling of the physical system, wherein the physical system is an energy generating facility, wherein the plurality of inter-related elements include at least one energy source within the energy generating facility and at least one energy storage mechanism within the energy generating facility, wherein the manipulatable control elements include one or more controls to determine whether to accept a request to supply a specified amount of energy at the current time and to determine to provide energy to the at least one energy storage mechanism at the current time if not accepted and to provide energy from the at least one energy source at the current time if accepted, wherein the outputs include the energy being provided, and wherein the at least one overall objective includes to maximize profits for the electricity generating facility from providing of the energy.

[c72] 72. The computer-implemented method of claim 69 wherein the multiple decision modules collectively include at least one overall objective to be achieved during controlling of the physical system, wherein the physical system is a vehicle, wherein the plurality of inter-related elements include a motor and a battery of the vehicle, wherein the manipulatable control elements include one or more controls to select whether at the current time to remove energy from the battery to power the motor or to add excess energy to the battery and how much energy to remove from the battery, wherein the outputs include effects of the motor to move the vehicle at the current time, and wherein the at least one overall objective includes to move the vehicle at one or more specified speeds with a minimum of energy produced from the battery.

[c73] 73. The computer-implemented method of claim 72 wherein the plurality of inter-related elements further includes an engine that is manipulatable to modify energy generated from the engine, wherein the manipulatable control elements further include one or more additional controls to determine how much energy to generate from the engine for use at least in part in adding the excess energy to the battery, and wherein the at least one overall objective further includes to minimize use of fuel by the engine.

[c74] 74. The computer-implemented method of claim 69 wherein the multiple decision modules collectively include at least one overall objective to be achieved during controlling of the physical system, wherein the physical system includes product inventory at one or more locations, wherein the plurality of inter-related elements include one or more product sources that provide products and increase the inventory at the one or more locations and further include one or more product recipients that receive products and decrease the inventory at the one or more locations, wherein the manipulatable control elements include one or more first controls to select at the current time one or more first amounts of one or more products to request from the one or more product sources, and further include one or more second controls to select at the current time one or more second amounts

of at least one product to provide to the one or more product recipients, wherein the outputs include products being provided from the one or more locations to the one or more product recipients, and wherein the at least one overall objective includes to maximize profit of an entity operating the one or more locations while maintaining the inventory at one or more specified levels.

[c75] 75. The computer-implemented method of claim 69 wherein each of the multiple decision modules' associated distinct model is expressed with a Hamiltonian function specific to that decision module, wherein each of the multiple decision modules has a distinct intermediate shared model specific to the decision module that is used to generate the consensus shared model for the decision module, and wherein, for one of the multiple decision modules, the intermediate shared model for the one decision module is expressed with a composite Hamiltonian function that is based at least in part on the Hamiltonian functions for the models being represented for the some other decision modules.

[c76] 76. The computer-implemented method of claim 75 wherein the generating of the consensus shared model for the one decision module is performed to synchronize the model of the one decision module and the intermediate shared model specific to the one decision module and includes:

creating, by the one or more computing systems, a combined Hamiltonian function that includes the Hamiltonian function specific to the one decision module and the composite Hamiltonian function for the intermediate shared model for the one decision module;

determining, by the one or more computing systems, the Pareto equilibrium for the consensus shared model for the one decision module based on the combined Hamiltonian function;

updating, by the one or more computing systems, the model for the one decision module and the intermediate shared model for the one decision module based on results of the determined Pareto equilibrium; and

providing information to one or more other decision modules about the updating, to enable further updating of intermediate shared models specific to the one or more other decision modules based on the updating.

[c77] 77. The computer-implemented method of claim 69 wherein each of the multiple decision modules' associated distinct model is expressed with a Hamiltonian function specific to that decision module, wherein the intermediate shared model is associated with an additional virtual decision module and is expressed with an additional Hamiltonian function, and wherein the generating of the consensus shared models for the multiple decision modules includes successively synchronizing the intermediate shared model and the model of each of the multiple decision modules by:

creating, by the one or more computing systems, a combined Hamiltonian function that includes the Hamiltonian function specific to the decision module and the additional Hamiltonian function for the intermediate shared model;

determining, by the one or more computing systems, the Pareto equilibrium for the consensus shared model for the decision module based on the combined Hamiltonian function; and

before performing the synchronizing for a next decision module, updating, by the one or more computing systems, the model for the decision module and the intermediate shared model based on results of the determined Pareto equilibrium.

[c78] 78. The computer-implemented method of claim 69 further comprising, for each of multiple additional times after the current time and for one of the multiple decision modules, adapting the consensus shared model of the one decision module for the additional time by:

updating, by the one or more computing systems and for the additional time, the intermediate shared model used in the generating of the consensus shared model for the one decision module based at least in part on updates to the consensus shared models for one or more of the some other decision modules whose models are represented by that intermediate shared model;

obtaining additional sensor information that identifies state information at the additional time for one or more elements of the plurality;

updating, by the one or more computing systems and for the additional time, the model of the one decision module based at least in part on the additional sensor information;

determining, by the one or more computing systems and for the additional time, if an updated consensus shared model for the one decision module can be generated by attempting to determine an additional Pareto equilibrium for the updated model of the one decision module for the additional time and for the updated intermediate shared model for the additional time, wherein the updated consensus shared model, if generated, simultaneously provides further solutions for the additional time for the goals of the one decision module and of the some other decision modules whose models are represented by the intermediate shared model; and

if the updated consensus shared model for the additional time is generated, providing information about one or more additional associated control actions of that updated consensus shared model to perform in the physical system for the additional time to further manipulate the manipulatable control elements in a specified manner.

[c79] 79. The computer-implemented method of claim 78 further comprising, for each of the multiple additional times and if the updated consensus shared model for that additional time is not generated, providing information about one or more further associated control actions of the model of the one decision module for that additional time to be perform in the physical system for that additional time to further manipulate the manipulatable control elements in a specified manner.

[c80] 80. The computer-implemented method of claim 79 wherein generating of the updated consensus shared model for the one decision module for one of the multiple additional times includes a deadline for the generating to enable real-time control of the physical system to be performed based on performing control

actions in the physical system for the one additional time, and wherein the generating of the updated consensus shared model for the one additional time fails to complete before the deadline, such that the one or more further associated control actions of the model of the one decision module for that one additional time are performed in the physical system for that one additional time to further manipulate the manipulatable control elements in a specified manner.

[c81] 81. The computer-implemented method of claim 78 wherein the model for the one decision module is based in part on multiple rules that each has one or more conditions to evaluate and that specify restrictions involving the plurality of inter-related elements, wherein the updated consensus shared model for one of the multiple additional times is not generated in a first attempt due to a failure to determine the additional Pareto equilibrium for the one additional time, and wherein the method further comprises generating the updated consensus shared model for the one decision module for the one additional time during a second attempt by:

determining, by the one or more computing systems, at least one of the multiple rules to temporarily relax for the model for the one decision module by modifying at least one of the specified restrictions corresponding to the determined at least one rule;

further updating, by the one or more computing systems, the updated model of the one decision module for the one additional time based at least in part on the modified at least one specified restrictions for the determined at least one rule;

generating, by the one or more computing systems, the updated consensus shared model for the one decision module for the one additional time by determining a further Pareto equilibrium for the further updated model of the one decision module and for the updated intermediate shared model for the one additional time, wherein the generated updated consensus shared model simultaneously provides further solutions for the one additional time for the goals

of the one decision module and of the some other decision modules whose models are represented by the intermediate shared model; and

providing information about the one or more additional associated control actions of the generated updated consensus shared model to perform in the physical system for the one additional time to further manipulate the manipulatable control elements in a specified manner.

[c82] 82. The computer-implemented method of claim 81 wherein the multiple rules include one or more absolute rules that specify non-modifiable restrictions that are requirements regarding operation of the physical system, and further include one or more hard rules that specify restrictions regarding operation of the physical system that can be modified in specified situations, and wherein each determined at least one rule is one of the hard rules.

[c83] 83. The computer-implemented method of claim 81 wherein the multiple rules include one or more soft rules whose conditions evaluate to one of three or more possible values under differing situations to represent varying degrees of uncertainty and further include additional rules whose conditions evaluate to either true or false under differing situations, and wherein one or more of the determined at least one rules are from the soft rules.

[c84] 84. The computer-implemented method of claim 78 wherein the model for the one decision module is based in part on multiple rules that each has one or more conditions to evaluate and that specify restrictions involving the plurality of inter-related elements,

wherein the determining if the updated consensus shared model for the one decision module can be generated for one of the multiple additional times is performed by the one or more computing systems and includes generating values for one or more model error measurements for an initial version of the updated consensus shared model for the one decision module for the one additional time, and includes determining that at least one of the generated values exceeds an

error threshold, and includes rejecting the initial version of the updated consensus shared model for the one decision module based at least in part on the at least one generated values exceeding the error threshold, and

wherein the method further comprises generating the updated consensus shared model for the one decision module for the one additional time by:

evaluating, by the one or more computing systems, the generated values for the one or more model error measurements to determine at least one of the multiple rules that is incorrect;

modifying, by the one or more computing systems, the determined at least one rule in a manner expected to reduce values for the one or more model error measurements;

further updating, by the one or more computing systems, the updated model of the one decision module for the one additional time based at least in part on the modified determined at least one rules;

generating, by the one or more computing systems, the updated consensus shared model for the one decision module for the one additional time by determining a further Pareto equilibrium for the further updated model of the one decision module and for the updated intermediate shared model for the one additional time, wherein the generated updated consensus shared model simultaneously provides further solutions for the one additional time for the goals of the one decision module and of the some other decision modules whose models are represented by the intermediate shared model, and wherein additional generated values for the one or more model error measurements for the generated updated consensus shared model do not exceed the error threshold; and

providing information about the one or more additional associated control actions of the generated updated consensus shared model to perform in the physical system for the one additional time to further manipulate the manipulatable control elements in a specified manner.

[c85] 85. The computer-implemented method of claim 84 wherein the one or more model error measurements are based on a rate of change of one or more of:

Hamiltonian functions expressed by the updated consensus shared model for the one decision module for two or more times;

amounts of entropy included in the updated consensus shared model for the one decision module for two or more additional times;

values of variables associated with the plurality of inter-related elements in state information for the physical system for two or more additional times; or

a reduction in the associated error measurements for the provided solutions of the updated consensus shared model for the one decision module for two or more additional times.

[c86] 86. The computer-implemented method of claim 78 wherein the determining if the updated consensus shared model for the one decision module can be generated for one of the multiple additional times includes:

generating, by the one or more computing systems, the updated consensus shared model for the one decision module for the one additional time;

generating, by the one or more computing systems, values for one or more model error measurements for the generated updated consensus shared model for the one decision module for the one additional time;

determining, by the one or more computing systems, that at least one of the generated values exceeds an error threshold; and

replacing, by the one or more computing systems, the generated updated consensus shared model for the one decision module for the one additional time with a new generated consensus shared model for the one decision module for the one additional time, by:

causing a new model of the one decision module for the one additional time to be generated without using any past versions of models of the one decision module, and/or causing a new model of the intermediate shared model for the one decision module for the one additional time to be generated by generating new models of one or more of the some other decision modules whose models are represented by that intermediate shared model without using any past versions of models of the one or more some other decision modules; and

generating the new consensus shared model for the one decision module for the one additional time by determining a further Pareto equilibrium based on the new model of the one decision module for the one additional time and/or the new model of the intermediate shared model for the one decision module for the one additional time.

[c87] 87. The computer-implemented method of claim 69 further comprising generating a converged shared model that is based on the consensus shared models for the multiple decision modules and that simultaneously provides solutions for the goals of each of the multiple decision modules, wherein the generating of the converged shared model includes successively synchronizing pairs of models that each includes at least one consensus shared model to converge on the converged shared model, and wherein the converged shared model includes the one or more associated control actions of the one or more consensus shared models.

[c88] 88. The computer-implemented method of claim 69 wherein the obtaining of the distinct model for each of the multiple decision modules for the current time includes:

determining, by the one or more computing systems at an earlier time before the current time, and for each of the multiple decision modules, an earlier version of the model for the decision module, wherein each of the multiple decision modules' associated earlier version of the model describes additional state information for the earlier time of at least of a portion of the physical system;

determining, by the one or more computing systems, and for each of the multiple decision modules, an earlier version of the set of coupled differential equations for the decision module that is based in part on the additional state information for the earlier time of the decision module; and

updating, for each of the multiple decision modules, the earlier version of the model for the decision module to reflect an additional solution for the specified goal

of the decision module that is to be achieved during controlling of the physical system for the earlier time, by:

performing, by the one or more computing systems, a piecewise linear analysis of the earlier version of the set of coupled differential equations for the decision module to identify one or more additional control actions that manipulate the manipulatable control elements and that provide the additional solution for the specified goal of the decision module for the earlier time within a threshold amount of an optimal solution for the specified goal for the earlier time; and

modifying, by the one or more computing systems, the earlier version of the model for the decision module to include effects of using the identified one or more additional control actions for the decision module to manipulate the manipulatable control elements for the earlier time, and wherein the modified earlier version of the model is the model for the decision module for the current time.

[c89] 89. The computer-implemented method of claim 69 wherein the providing of the information about the one or more associated control actions of the one or more consensus shared models includes performing, by the one or more computing systems, the actions in the physical system to affect the outputs by manipulating the manipulatable control elements in specified manners for the one or more associated control actions.

[c90] 90. A non-transitory computer-readable medium having stored contents that cause one or more computing systems of a collaborative distributed decision system to perform a method, the method comprising:

obtaining, by the one or more computing systems, and for each of multiple decision modules that collectively describe a state of a target system, a distinct model associated with the decision module, wherein the target system has a plurality of elements that are inter-related and that include one or more control elements with modifiable values, and wherein each of the multiple decision modules' associated distinct model describes information about a physical state of at least one element of the plurality and reflects a solution for a specified goal of the

decision module that is to be achieved based at least in part on modifying of the values of the control elements;

generating, by the one or more computing systems, and for each of the multiple decision modules, a consensus shared model for the decision module by determining a Pareto equilibrium for the model associated with the decision module and for an intermediate shared model representing the models associated with some other of the multiple decision modules, wherein the consensus shared model for the decision module simultaneously provides solutions for the goals of the decision module and the some other decision modules such that the provided solutions have an associated error measurement within a defined threshold relative to a global optimal solution, and wherein the consensus shared model for the decision module has one or more associated control actions that modify the value of at least one of the control elements in a specified manner for the provided solutions; and

providing information about the one or more associated control actions of one or more of the consensus shared models, to enable modification of values of control elements based on the one or more associated control actions.

[c91] 91. The non-transitory computer-readable medium of claim 90 wherein the target system is a physical system having one or more outputs whose values vary based at least in part on the values of the control elements, and wherein the stored contents include software instructions that, when executed, further cause the one or more computing systems to initiate performance of the one or more associated control actions in the physical system to modify the values of the control elements to have specified values and to cause resulting changes in the values of the one or more outputs for a specified time.

[c92] 92. The non-transitory computer-readable medium of claim 90 wherein the multiple decision modules collectively include at least one overall objective to be achieved based at least in part on modifying of the values of the control elements, wherein the target system includes one or more computing resources

being protected from unauthorized operations, wherein the plurality of inter-related elements include one or more sources of attempts to perform operations, wherein the control elements include one or more controls to determine whether a change in authorization to a specified type of operation is needed and to select one or more actions to take to implement the change in authorization if so determined, and wherein the at least one overall objective includes to minimize unauthorized operations that are performed.

[c93] 93. The non-transitory computer-readable medium of claim 90 wherein the multiple decision modules collectively include at least one overall objective to be achieved based at least in part on modifying of the values of the control elements, wherein the target system includes one or more information sources to be analyzed to determine a risk level from information of the one or more information sources, wherein the control elements include one or more controls to determine whether the risk level exceeds a specified threshold and to select one or more actions to take to mitigate the risk level, and wherein the at least one overall objective includes to minimize the risk level.

[c94] 94. The non-transitory computer-readable medium of claim 90 wherein the multiple decision modules collectively include at least one overall objective to be achieved based at least in part on modifying of the values of the control elements, wherein the target system includes one or more financial markets, wherein the plurality of inter-related elements include items that can be purchased and/or sold in the one or more financial markets, wherein the control elements include one or more controls to determine whether to purchase or sell particular items at particular times and to select one or more actions to initiate transactions to purchase or sell the particular items at the particular times, and wherein the at least one overall objective includes to maximize profit while maintaining risk below a specified threshold.

[c95] 95. The non-transitory computer-readable medium of claim 90 wherein the multiple decision modules collectively include at least one overall objective to be achieved based at least in part on modifying of the values of the control elements, wherein the target system includes functionality to perform coding for medical procedures performed on humans, wherein the plurality of inter-related elements include a plurality of medical codes corresponding to a plurality of medical procedures, wherein the control elements include one or more controls to select particular medical codes to associate with particular medical procedures in specified circumstances, and wherein the at least one overall objective includes to minimize errors in selected medical codes that cause revenue leakage.

[c96] 96. A system comprising:
one or more processors of one or more computing systems; and
one or more modules that, when executed by at least one of the one or more processors, cause the one or more processors to determine one or more control actions to perform as part of controlling a physical system, the determining of the one or more control actions including:
obtaining, for each of multiple decision modules that collectively describe a state at a specified time of the physical system, a distinct model associated with the decision module, wherein the physical system has a plurality of inter-related elements and has one or more outputs whose values vary based at least in part on modifications to one or more control elements of the plurality, and wherein each of the multiple decision modules' associated distinct model describes state information for the specified time of at least of a portion of the physical system and reflects a solution for a specified goal of the decision module that is to be achieved during the controlling of the physical system;
generating a converged shared model that is based on the models for the multiple decision modules and that simultaneously provides solutions for the goals of each of the multiple decision modules, wherein the generating of the converged shared model includes successively synchronizing an intermediate version of the converged shared model with each of the models for the multiple

decision modules by determining a Pareto equilibrium for the intermediate version of the converged shared model and the model associated with the decision module, to converge on the converged shared model, and wherein the converged shared model identifies the one or more control actions to perform in the physical system to modify the control elements in a specified manner for the provided solutions; and

providing information about the one or more control actions, to enable actions to be taken in the physical system to affect the outputs based on the one or more control actions.

[c97] 97. The system of claim 96 wherein the one or more modules are part of a collaborative distributed decision system and include software instructions for execution by the at least one processor, wherein the provided solutions of the converged shared model have an associated error measurement within a defined threshold relative to a global optimal solution for the goals of each of the multiple decision modules, and wherein the system further comprises one or more effectuators to perform the actions in the physical system by manipulating values of the one or more manipulatable control elements in specified manners for the identified one or more control actions to affect the outputs.

[c98] 98. The system of claim 97 wherein the determining of the one or more control actions to perform further includes:

for each of the multiple decision modules, generating a consensus shared model for the decision module for the specified time by determining a Pareto equilibrium for the model associated with the decision module and for an intermediate shared model representing the models associated with some other of the multiple decision modules, wherein the consensus shared model for the decision module simultaneously provides solutions for the goals of the decision module and the some other decision modules such that the provided solutions have an associated error measurement within a defined threshold relative to a global optimal solution, and wherein the consensus shared model for the decision module has one or more associated control actions to perform in the physical system to

manipulate the manipulatable control elements in a specified manner for the provided solutions; and

wherein the generating of the converged shared model based on the models for the multiple decision modules includes using the generated consensus shared models for the multiple decision modules as part of the generating of the converged shared model.

[c99] 99. The system of claim 96 wherein the one or more modules consist of one or more means for performing the determining of the one or more control actions to perform as part of controlling the physical system.

[c100] 100. A computer-implemented method comprising:
generating, by one or more computing systems of a collaborative distributed decision system, an executable decision module for use with a physical system having a plurality of inter-related elements and having one or more outputs whose values vary based at least in part on one or more manipulatable control elements of the plurality, the generating including:

receiving, by the one or more computing systems and from one or more users, system information describing at least a portion of the physical system using multiple rules that each has one or more conditions to evaluate and that specify restrictions involving the plurality of elements;

converting, by the one or more computing systems and to multiple constraints, the system information and information about a goal to be achieved during controlling of the physical system and information about sensor variables whose values are available during the controlling of the physical system;

validating, by the one or more computing systems and based on multiple validation rules, the multiple constraints by testing at least one of controllability, observability, stability or goal completeness, wherein testing controllability ensures that each of the manipulatable control elements is specified in the multiple rules to affect one or more other elements of the plurality, wherein testing observability ensures that each of the sensor variables is specified in the

multiple rules to relate to at least one element of the plurality, wherein testing stability ensures that solutions determined by the generated decision module during controlling of the at least portion of the physical system will converge with any other solutions determined by any other generated decision modules during controlling of other portions of the physical system, wherein testing goal completeness ensures that each of the manipulatable control elements is reflected in the goal;

converting, by the one or more computing systems, the validated constraints to coupled differential equations that represent a model for use in describing a state of the target system;

training, by the one or more computing systems, the model using state information for the plurality of elements for multiple times, to enable the model to determine values of one or more elements that are not directly observable; and

testing, by the one or more computing systems, performance of the trained model in controlling the at least portion of the physical system by simulating manipulations of the manipulatable control elements to affect the outputs in expected manners; and

if the testing is successful, providing the generated decision module with the trained model, to enable execution of the generated decision module to perform actual controlling of the at least portion of the physical system.

[c101] 101. The computer-implemented method of claim 100 wherein the providing of the generated decision module with the trained model includes initiating the execution of the generated decision module in an environment that provides the executing generated decision module with access to perform the actual controlling of the at least portion of the physical system.

[c102] 102. The computer-implemented method of claim 100 wherein the generating of the executable decision module includes providing, by the one or more computing systems, an online interface to a rule builder component of the collaborative distributed decision system, and wherein the receiving of the system

information from the one or more users includes receiving electronic communications at the online interface that are sent over one or more computer networks from one or more client devices of the one or more users.

[c103] 103. The computer-implemented method of claim 100 wherein the physical system is an electricity generating facility, wherein the plurality of inter-related elements include multiple alternative electricity sources within the electricity generating facility, wherein the manipulatable control elements include one or more controls to determine whether to accept a request to supply a specified amount of electricity at a current time and to select which alternative electricity source to provide the specified amount of electricity at the current time if accepted, wherein the outputs include the electricity being provided, and wherein the goal includes to maximize profits for the electricity generating facility from providing of the electricity.

[c104] 104. The computer-implemented method of claim 100 wherein the physical system is an energy generating facility, wherein the plurality of inter-related elements include at least one energy source within the energy generating facility and at least one energy storage mechanism within the energy generating facility, wherein the manipulatable control elements include one or more controls to determine whether to accept a request to supply a specified amount of energy at a current time and to determine to provide energy to the at least one energy storage mechanism at the current time if not accepted and to provide energy from the at least one energy source at the current time if accepted, wherein the outputs include the energy being provided, and wherein the goal includes to maximize profits for the electricity generating facility from providing of the energy.

[c105] 105. The computer-implemented method of claim 100 wherein the physical system is a vehicle, wherein the plurality of inter-related elements include a motor and a battery of the vehicle, wherein the manipulatable control elements include one or more controls to select whether at a current time to remove energy

from the battery to power the motor or to add excess energy to the battery and how much energy to remove from the battery, wherein the outputs include effects of the motor to move the vehicle at the current time, and wherein the goal includes to move the vehicle at one or more specified speeds with a minimum of energy produced from the battery.

[c106] 106. The computer-implemented method of claim 105 wherein the plurality of inter-related elements further includes an engine that is manipulatable to modify energy generated from the engine, wherein the manipulatable control elements further include one or more additional controls to determine how much energy to generate from the engine for use at least in part in adding the excess energy to the battery, and wherein the goal further includes to minimize use of fuel by the engine.

[c107] 107. The computer-implemented method of claim 100 wherein the physical system includes product inventory at one or more locations, wherein the plurality of inter-related elements include one or more product sources that provide products and increase the inventory at the one or more locations and further include one or more product recipients that receive products and decrease the inventory at the one or more locations, wherein the manipulatable control elements include one or more first controls to select at a current time one or more first amounts of one or more products to request from the one or more product sources, and further include one or more second controls to select at the current time one or more second amounts of at least one product to provide to the one or more product recipients, wherein the outputs include products being provided from the one or more locations to the one or more product recipients, and wherein the goal includes to maximize profit of an entity operating the one or more locations while maintaining the inventory at one or more specified levels.

[c108] 108. The computer-implemented method of claim 100 wherein the training of the model includes generating values for one or more model error

measurements for one or more of the multiple times, and further includes determining that at least one of the generated values exceeds an error threshold, and wherein the method further comprises:

- evaluating, by the one or more computing systems, the generated values for the one or more model error measurements to determine at least one of the multiple rules that is incorrect;

- providing, by the one or more computing systems, feedback to the one or more users regarding the determined at least one rules that is incorrect;

- receiving, by the one or more computing systems and in response to the provided feedback, one or more revised versions of the determined at least one rules; and

- updating, by the one or more computing systems, the model to reflect the one or more revised versions of the determined at least one rules.

[c109] 109. The computer-implemented method of claim 108 wherein the training of the model for each of the multiple times includes identifying a solution for the goal for the time that has an associated error measurement within a defined threshold relative to an optimal solution for the goal for that time, and further includes updating a copy of the model for that time to reflect the identified solution, and wherein the one or more model error measurements are based on a rate of change of one or more of:

- Hamiltonian functions expressed by two or more copies of the model for two or more times;

- amounts of entropy included in two or more copies of the model for two or more times;

- values of variables associated with the plurality of inter-related elements in state information for the physical system for two or more times; or

- a reduction in the associated error measurements for the identified solutions for two or more times.

[c110] 110. The computer-implemented method of claim 108 wherein the multiple rules include one or more absolute rules that specify non-modifiable restrictions that are requirements regarding operation of the physical system, and further include one or more hard rules that specify restrictions regarding operation of the physical system that can be modified in specified situations, and wherein each determined at least one rule is one of the hard rules.

[c111] 111. The computer-implemented method of claim 108 wherein the multiple rules include one or more soft rules whose conditions evaluate to one of three or more possible values under differing situations to represent varying degrees of uncertainty and further include additional rules whose conditions evaluate to either true or false under differing situations, and wherein one or more of the determined at least one rules are from the soft rules.

[c112] 112. The computer-implemented method of claim 100 wherein the training of the model for one of the multiple times includes failing to identify a solution for the goal for the one time that satisfies one or more specified criteria, and wherein the method further comprises:

providing, by the one or more computing systems, feedback to the one or more users regarding the failing to identify the solution;

receiving, by the one or more computing systems and in response to the provided feedback, one or more revised versions of at least one of the goal or of one or more rules of the multiple rules; and

updating, by the one or more computing systems, the model to reflect the one or more revised versions of the at least one of the goal or of the one or more rules.

[c113] 113. The computer-implemented method of claim 100 wherein the testing of the performance of the trained model includes identifying one or more problems in the performance, and wherein the method further comprises:

providing, by the one or more computing systems, feedback to the one or more users regarding the one or more problems in the performance;

receiving, by the one or more computing systems and in response to the provided feedback, one or more revised versions of at least one of the goal or of one or more rules of the multiple rules; and

updating, by the one or more computing systems, the model to reflect the one or more revised versions of the at least one of the goal or of the one or more rules.

[c114] 114. The computer-implemented method of claim 113 wherein the identifying of the one or more problems in the performance of the trained model includes:

generating, by the one or more computing systems, values for one or more model error measurements for the trained model that accumulate over multiple times of using the trained model to identify a solution to the goal for that time; and

determining, by the one or more computing systems, that at least one of the generated values exceeds an error threshold.

[c115] 115. The computer-implemented method of claim 100 wherein the generated executable decision module is a first decision module for a first portion of the physical system that is described by the received system information, and wherein the method further comprises generating one or more additional second decision modules that each has distinct system information describing distinct second portions of the physical system.

[c116] 116. The computer-implemented method of claim 115 wherein the receiving of the system information from the one or more users includes:

receiving a plurality of rules from the one or more users that collectively describe all of the physical system and that include the multiple rules and that further include multiple additional rules;

receiving an identification from the one or more users of a first subset of the plurality of rules that includes the multiple rules for use with the first decision module; and

receiving additional identifications from the one or more users of one or more distinct second subsets of the plurality of rules for use with the one or more additional second decision modules, wherein each of the distinct second subsets includes at least one of the multiple additional rules,

and wherein the generating of the one or more additional second decision modules is based at least in part on the received additional identifications of the one or more distinct second subsets of the plurality of rules for use with the one or more additional second decision modules.

[c117] 117. The computer-implemented method of claim 115 wherein the receiving of the system information from the one or more users includes receiving a plurality of rules from the one or more users that collectively describe all of the physical system and that include the multiple rules and that further include multiple additional rules,

wherein the method further comprises analyzing, by the one or more computing systems, the plurality of rules to decompose the plurality of rules into multiple distinct subsets, wherein a first of the multiple distinct subsets includes the multiple rules, and wherein one or more distinct second subsets of the multiple subsets each includes at least one of the multiple additional rules,

wherein the generating of the executable decision module includes using the first subset for the generated executed decision module; and

wherein the generating of the one or more additional second decision modules is initiated by the one or more computing systems based at least in part on the analyzing to create a distinct decision module for each of the one or more distinct second subsets.

[c118] 118. The computer-implemented method of claim 100 wherein the receiving of the system information from the one or more users further includes:

validating syntax of the multiple rules as part of the receiving of the system information; and

if one or more errors are identified from the validating of the syntax, and before the converting of the system information and the information about the goal and the information about the sensor variables, providing feedback to the one or more users regarding the one or more errors to enable the one or more users to correct the one or more errors.

[c119] 119. The computer-implemented method of claim 100 wherein the converting of the system information and the information about the goal and the information about the sensor variables further includes:

validating syntax of the multiple rules; and

if one or more errors are identified from the validating of the syntax, and before the validating of the constraints, providing feedback to the one or more users regarding the one or more errors to enable the one or more users to correct the one or more errors.

[c120] 120. The computer-implemented method of claim 100 wherein the multiple validation rules are provided by the collaborative distributed decision system and are each associated with one of controllability, observability, stability or goal completeness, and wherein the validating of the multiple constraints includes testing all of controllability, observability, stability or goal completeness.

[c121] 121. The computer-implemented method of claim 100 wherein the multiple validation rules include one or more validation rules associated with controllability, and wherein the validating of the multiple constraints includes testing the controllability.

[c122] 122. The computer-implemented method of claim 100 wherein the multiple validation rules include one or more validation rules associated with

observability, and wherein the validating of the multiple constraints includes testing the observability.

[c123] 123. The computer-implemented method of claim 100 wherein the multiple validation rules include one or more validation rules associated with stability, and wherein the validating of the multiple constraints includes testing the stability.

[c124] 124. The computer-implemented method of claim 100 wherein the multiple validation rules include one or more validation rules associated with goal completeness, and wherein the validating of the multiple constraints includes testing the goal completeness.

[c125] 125. The computer-implemented method of claim 100 wherein the multiple validation rules further include one or more user-specified validation rules that are specified by the one or more users, and wherein the validating of the multiple constraints includes executing the one or more user-specified validation rules.

[c126] 126. The computer-implemented method of claim 100 wherein the validating of the multiple constraints includes identifying one or more problems from the testing, and wherein the method further comprises performing further interactions with the one or more users to correct the one or more problems before proceeding with the training and the testing.

[c127] 127. A non-transitory computer-readable medium having stored contents that cause one or more computing systems to perform a method, the method comprising:

generating, by the one or more computing systems, an executable decision module for use with a target system having a plurality of elements that are inter-

related and that include one or more control elements with modifiable values, the generating including:

receiving, by the one or more computing systems and from one or more users, system information describing at least a portion of the target system using multiple rules that each has one or more conditions to evaluate and that specify restrictions involving the plurality of elements;

converting, by the one or more computing systems and to multiple constraints, the system information and information about a goal to be achieved during modifying the values of the control elements and information about sensor variables whose values are available during operation of the target system;

validating, by the one or more computing systems and based on multiple validation rules, the multiple constraints by testing at least one of controllability, observability, stability or goal completeness; and

converting, by the one or more computing systems, the validated constraints to coupled differential equations that represent a model for use in describing a state of the target system; and

providing the generated decision module with the coupled differential equations representing the model, to enable execution of the generated decision module to perform modifying of the values of the control elements during the execution.

[c128] 128. The non-transitory computer-readable medium of claim 127 wherein the target system is a physical system having one or more outputs whose values vary based at least in part on the values of the control elements, wherein the one or more computing systems are part of a collaborative distributed decision system, and wherein the stored contents include software instructions that, when executed, further cause the one or more computing systems to:

train, by the one or more computing systems, the model using state information for the plurality of elements for multiple times, to identify values for the trained model of one or more elements of the plurality that are not directly observable; and

test, by the one or more computing systems, performance of the model in controlling the at least portion of the physical system by simulating modifications of the values of the control elements to affect the outputs in expected manners.

[c129] 129. The non-transitory computer-readable medium of claim 127 wherein the target system includes one or more computing resources being protected from unauthorized operations, wherein the plurality of inter-related elements include one or more sources of attempts to perform operations, wherein the control elements include one or more controls to determine whether a change in authorization to a specified type of operation is needed and to select one or more actions to take to implement the change in authorization if so determined, and wherein the goal includes to minimize unauthorized operations that are performed.

[c130] 130. The non-transitory computer-readable medium of claim 127 wherein the target system includes one or more information sources to be analyzed to determine a risk level from information of the one or more information sources, wherein the control elements include one or more controls to determine whether the risk level exceeds a specified threshold and to select one or more actions to take to mitigate the risk level, and wherein the goal includes to minimize the risk level.

[c131] 131. The non-transitory computer-readable medium of claim 127 wherein the target system includes one or more financial markets, wherein the plurality of inter-related elements include items that can be purchased and/or sold in the one or more financial markets, wherein the control elements include one or more controls to determine whether to purchase or sell particular items at particular times and to select one or more actions to initiate transactions to purchase or sell the particular items at the particular times, and wherein the goal includes to maximize profit while maintaining risk below a specified threshold.

[c132] 132. The non-transitory computer-readable medium of claim 127 wherein the target system includes functionality to perform coding for medical procedures performed on humans, wherein the plurality of inter-related elements include a plurality of medical codes corresponding to a plurality of medical procedures, wherein the control elements include one or more controls to select particular medical codes to associate with particular medical procedures in specified circumstances, and wherein the goal includes to minimize errors in selected medical codes that cause revenue leakage.

[c133] 133. A system comprising:
one or more processors of one or more computing systems; and
one or more modules that, when executed by at least one of the one or more processors, cause the one or more processors to generate an executable decision module to use as part of controlling a physical system, wherein the physical system has a plurality of inter-related elements and has one or more outputs whose values vary based at least in part on one or more manipulatable control elements of the plurality, and wherein the generating of the executable decision module includes:
receiving, from one or more users, system information describing at least a portion of the physical system using multiple rules that each has one or more conditions to evaluate and that specify restrictions involving the plurality of elements;
converting, to multiple constraints, the system information and information about a goal to be achieved during controlling of the physical system and information about sensor variables whose values are available during operation of the physical system;
converting, by the one or more computing systems, the constraints to coupled differential equations that represent a model for use in describing a state of the target system; and
training, by the one or more computing systems, the model using state information for the plurality of elements for multiple times, and testing performance

of the trained model in controlling the at least portion of the physical system by simulating manipulations of the manipulatable control elements to affect the outputs in expected manners; and

providing the generated decision module with the trained model, to enable execution of the generated decision module to perform further controlling of the at least portion of the physical system.

[c134] 134. The system of claim 133 wherein the one or more modules are part of a collaborative distributed decision system and include software instructions for execution by the at least one processor, wherein the training of the model includes identifying values of one or more of the plurality of elements that are not directly observable during operation of the physical system, and wherein the generating of the executable decision module further includes validating, based on multiple validation rules and before the providing of the generated decision module, the multiple constraints by testing at least one of controllability, observability, stability or goal completeness.

[c135] 135. The system of claim 133 wherein the one or more modules consist of one or more means for performing the generating of the executable decision module to use as part of controlling the physical system.

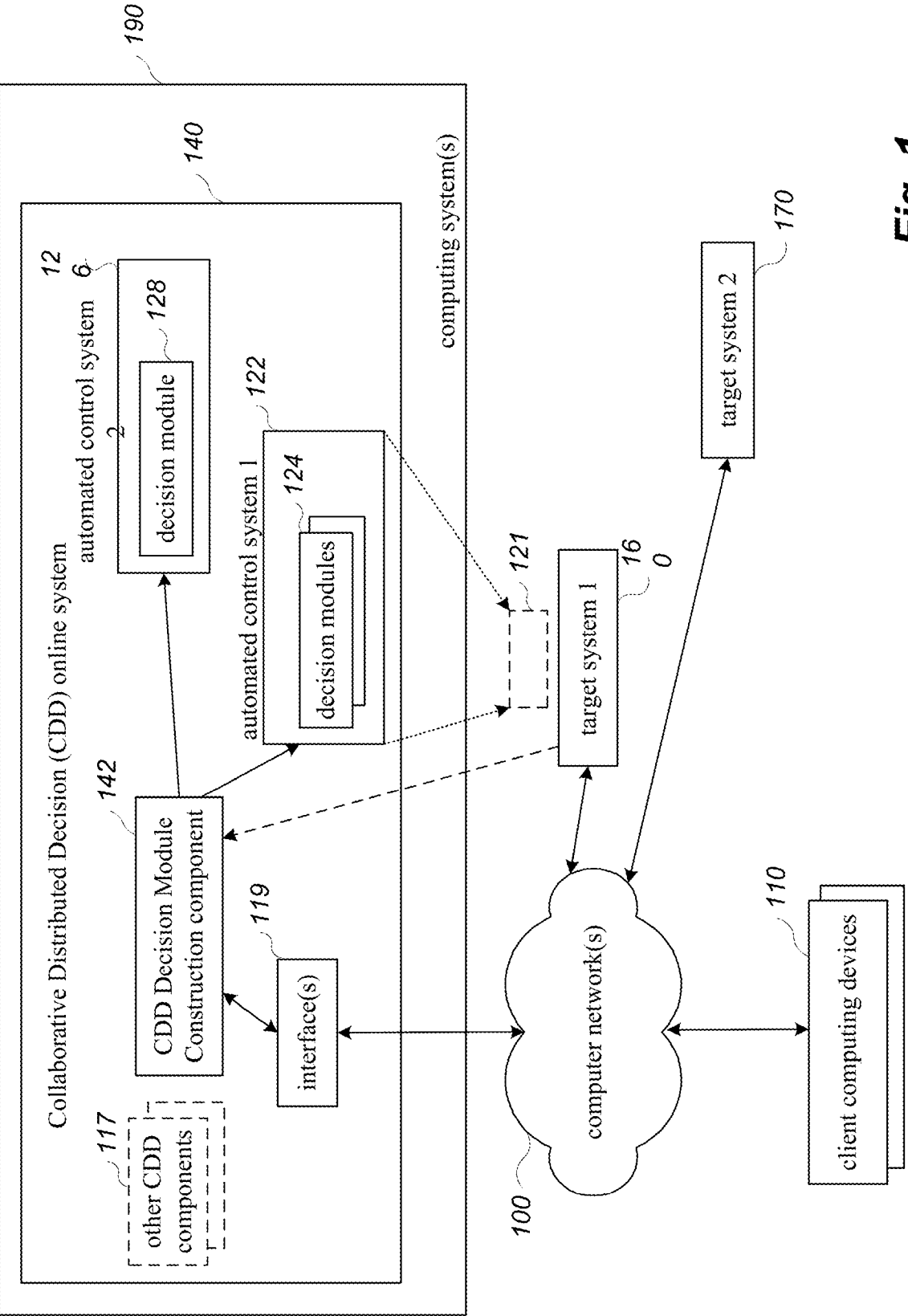
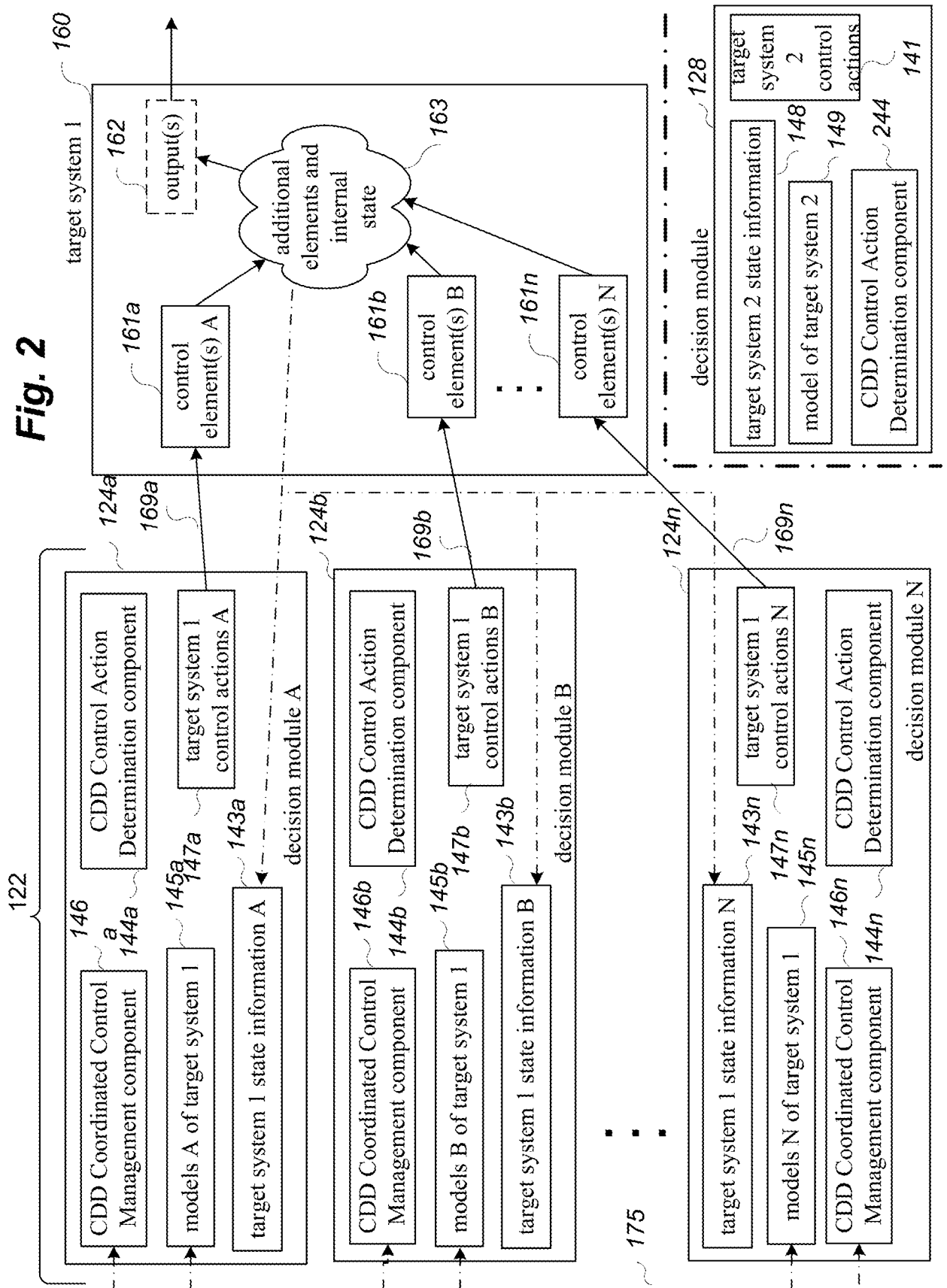


Fig. 1

Fig. 2



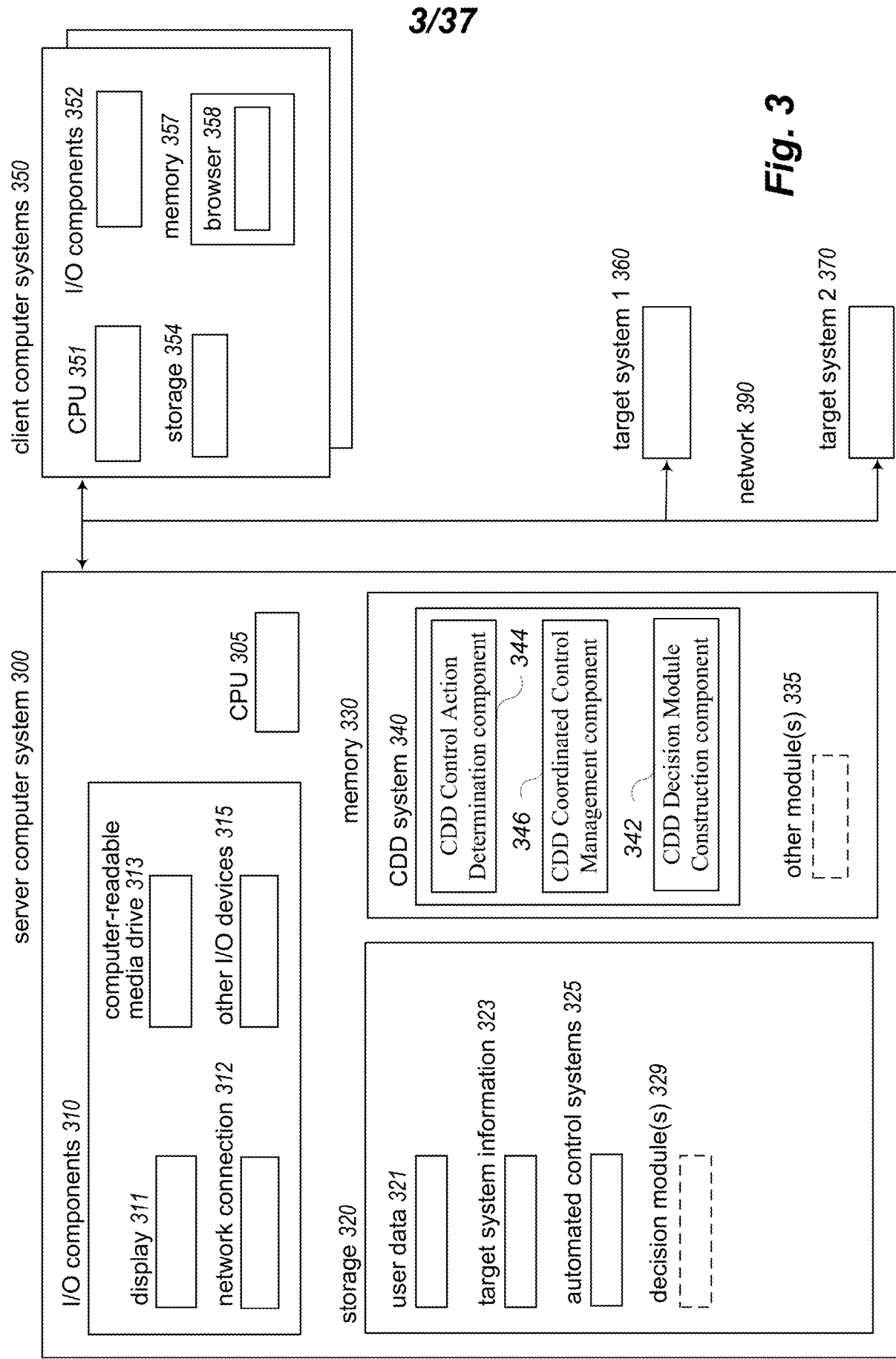
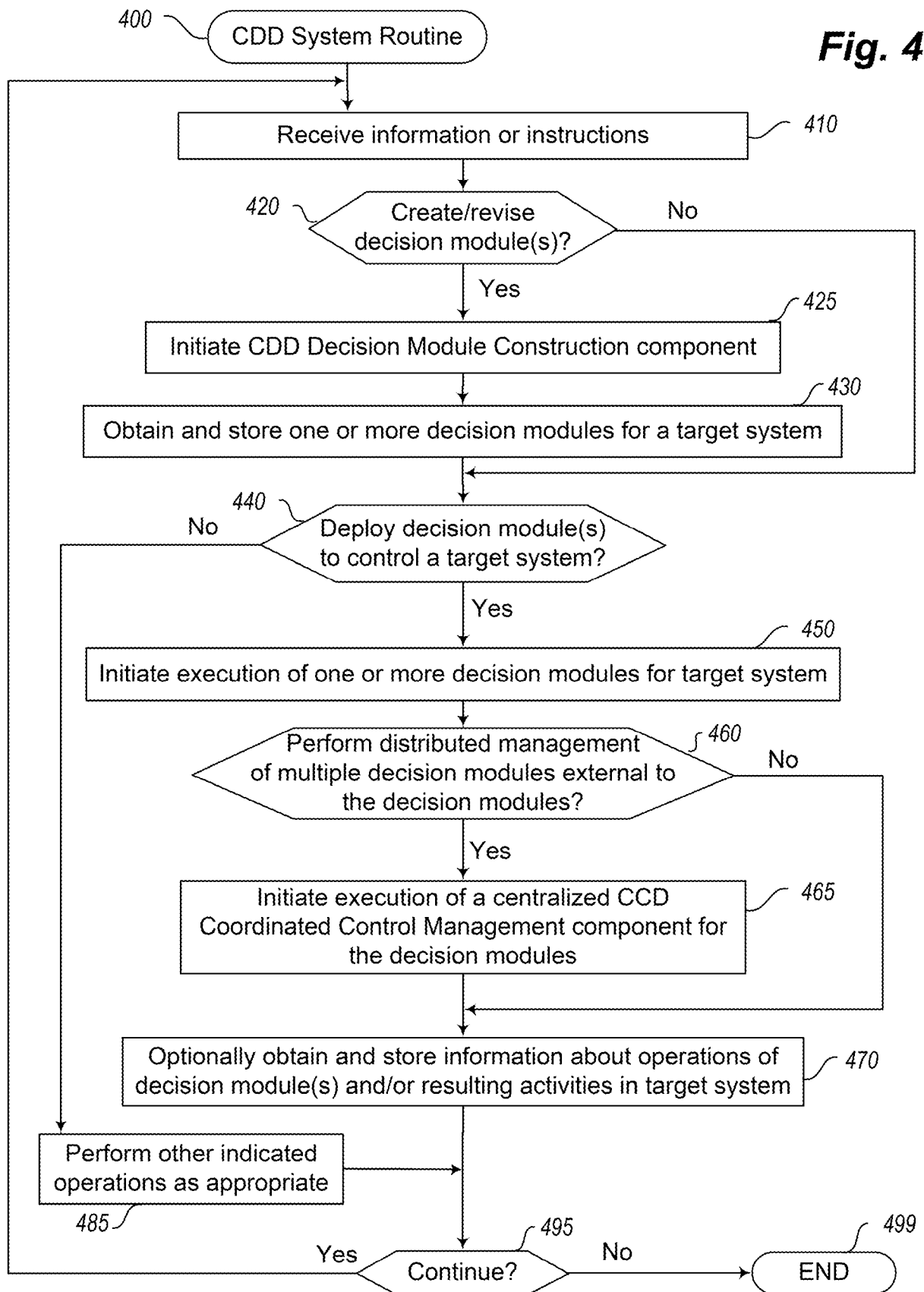


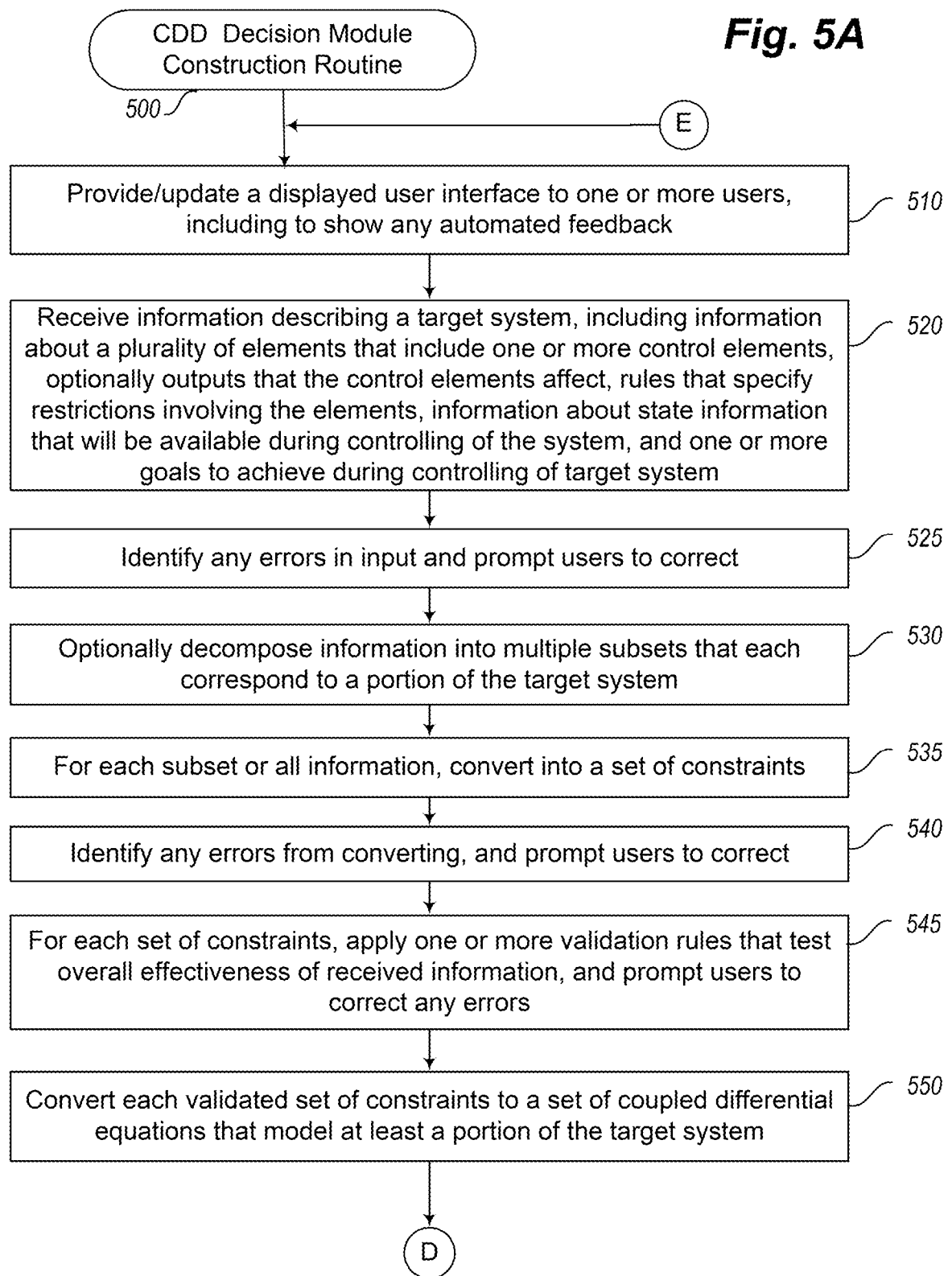
Fig. 3

4/37

Fig. 4

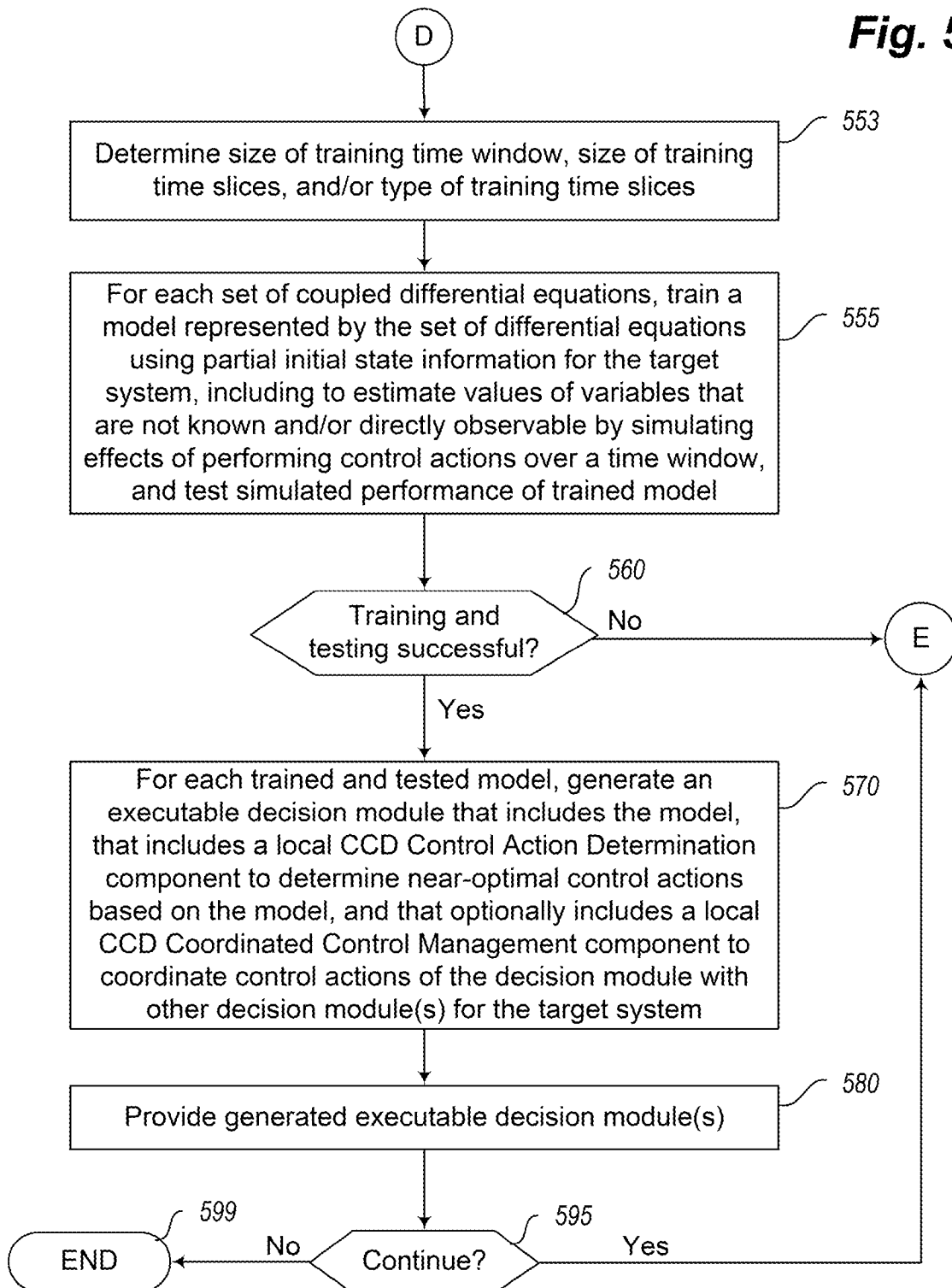


5/37

Fig. 5A

6/37

Fig. 5B



7/37

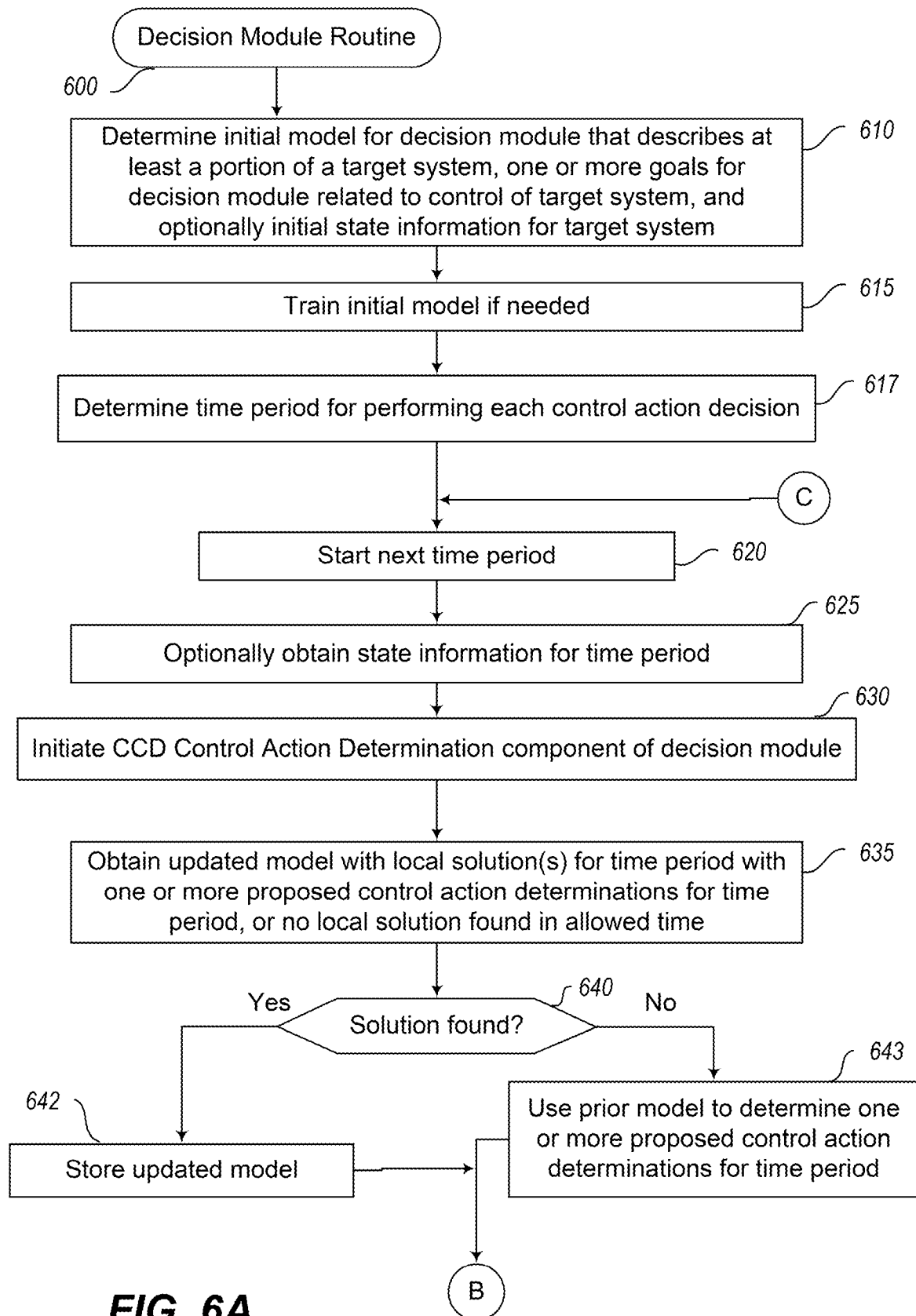
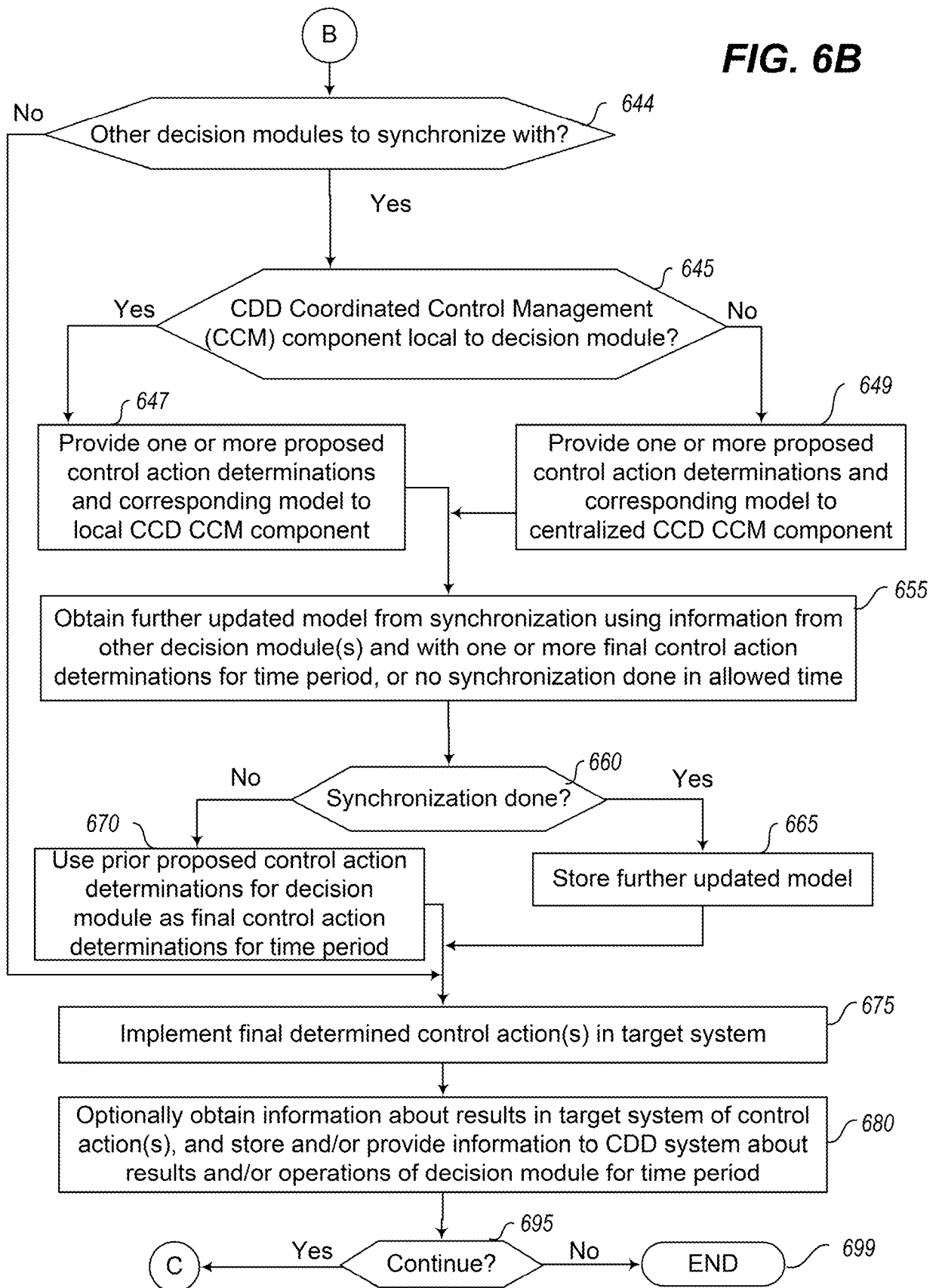


FIG. 6A

8/37

FIG. 6B



9/37

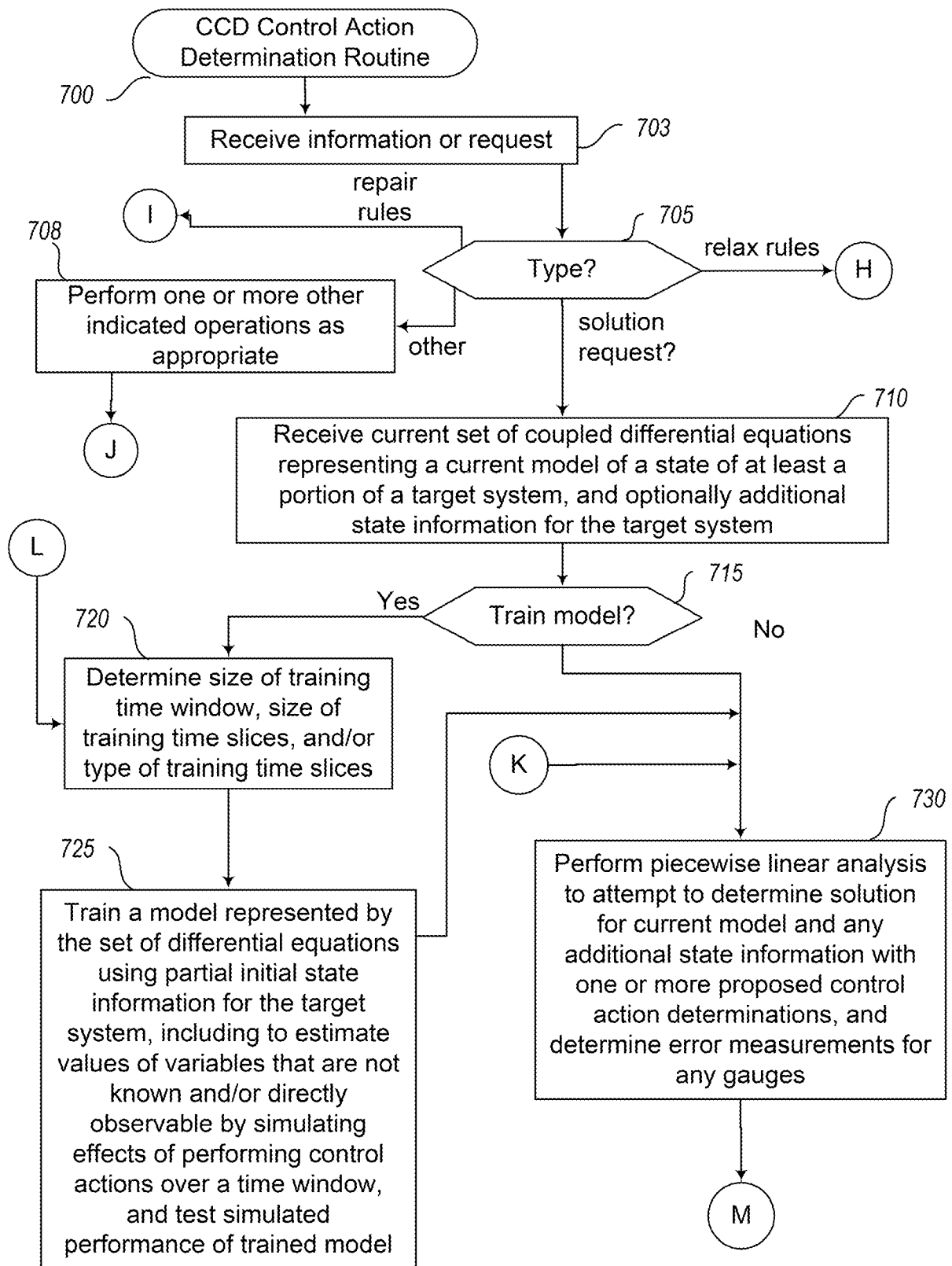


FIG. 7A

10/37

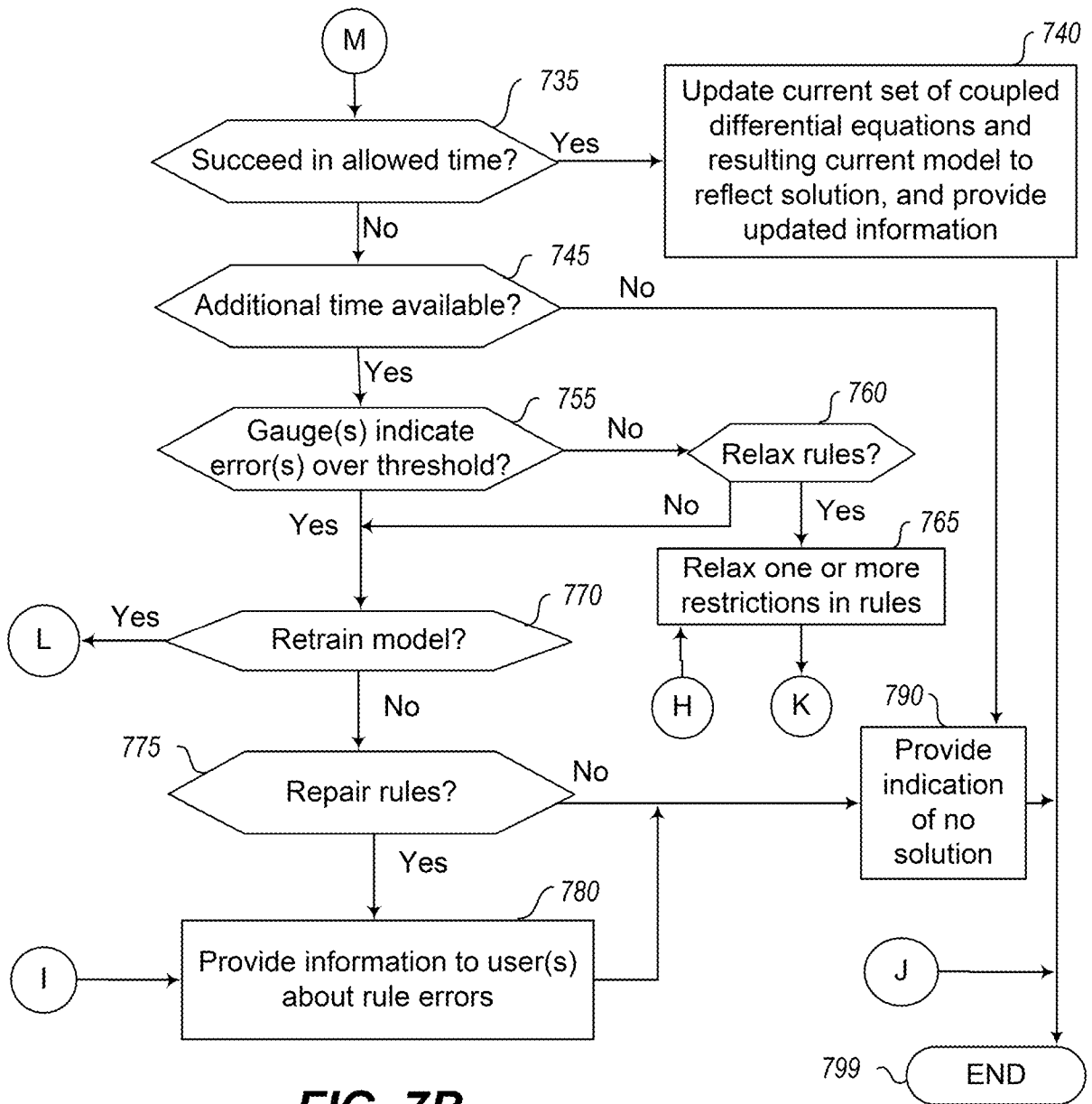
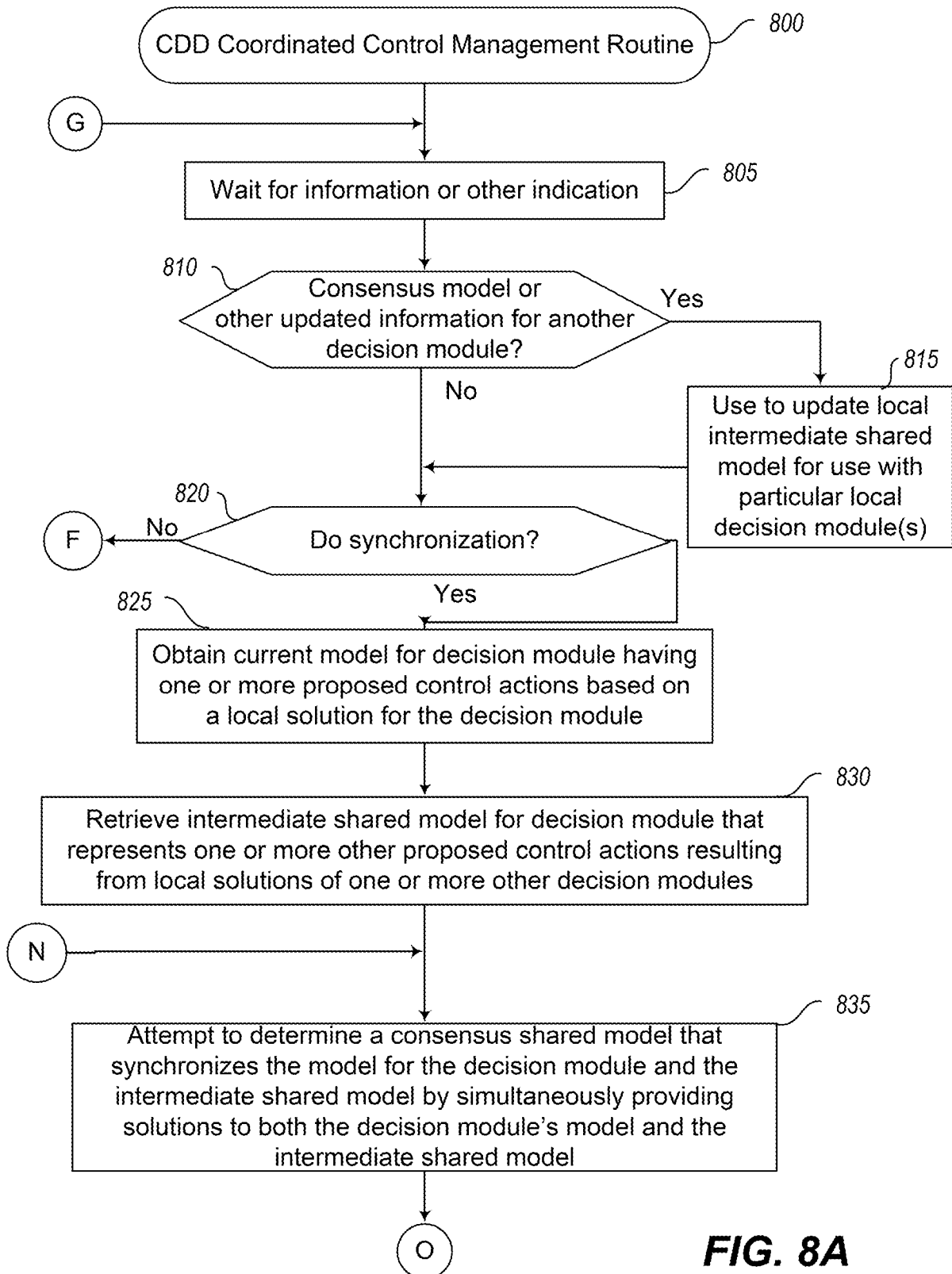


FIG. 7B

11/37



12/37

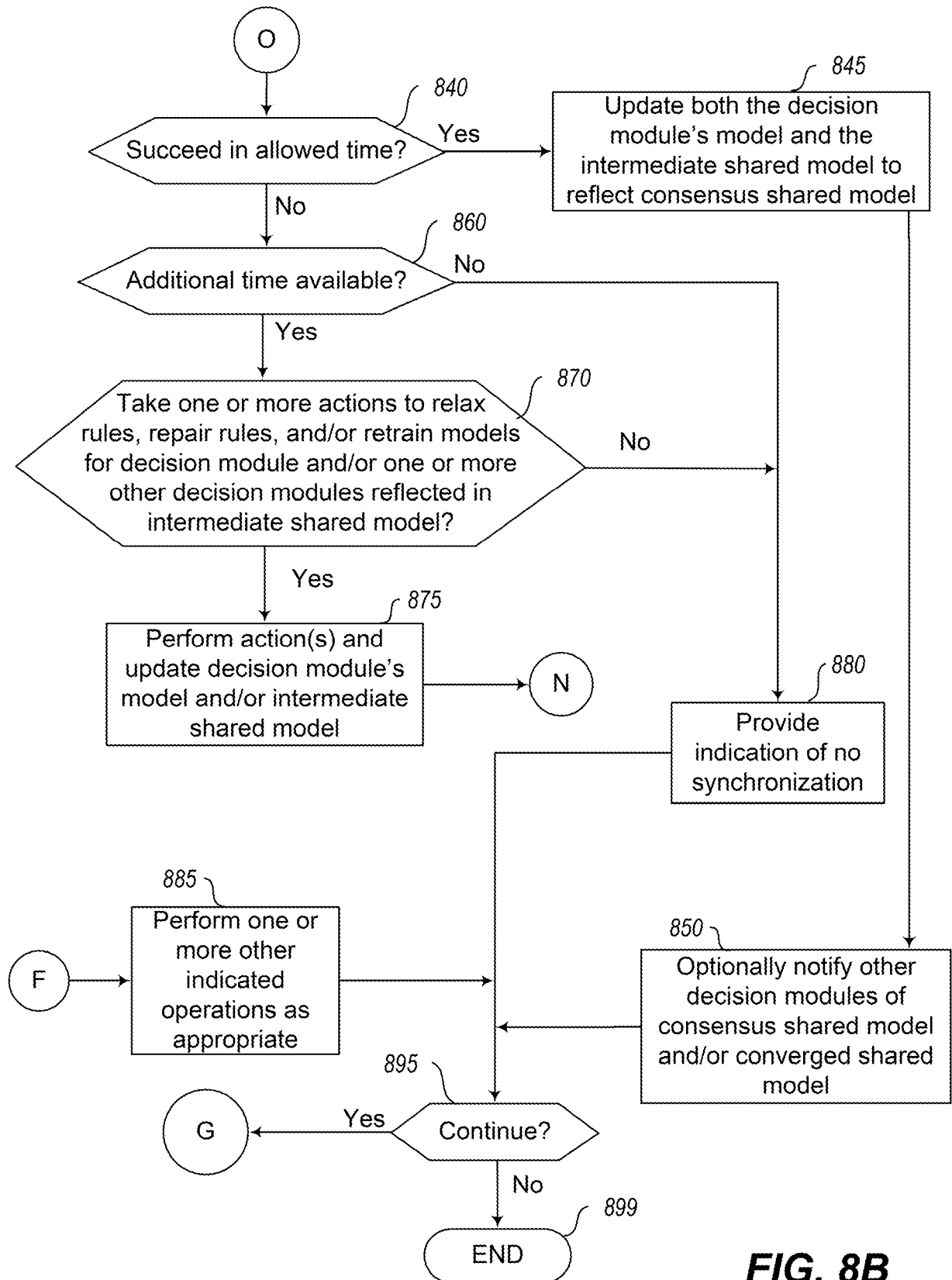
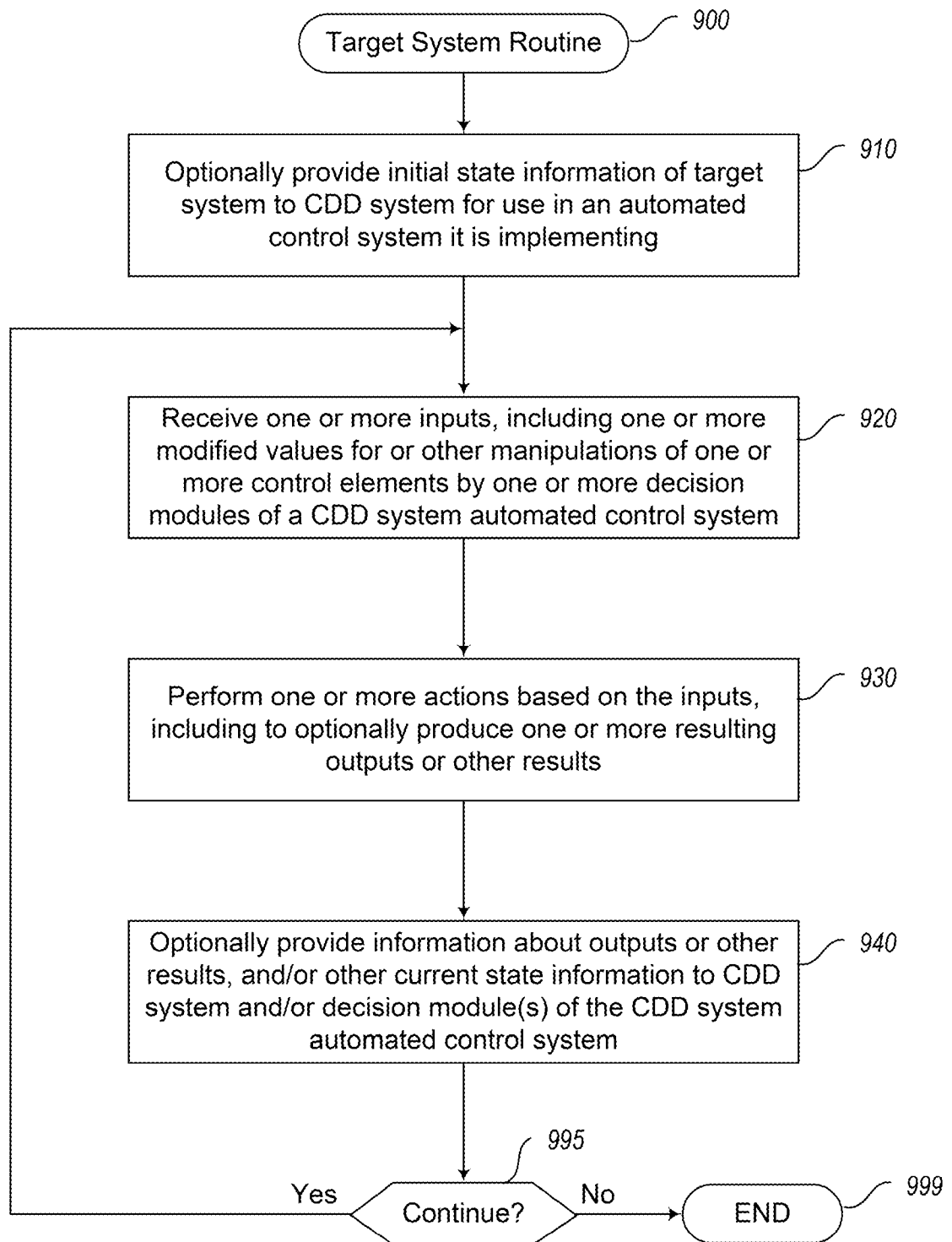


FIG. 8B

13/37

FIG. 9



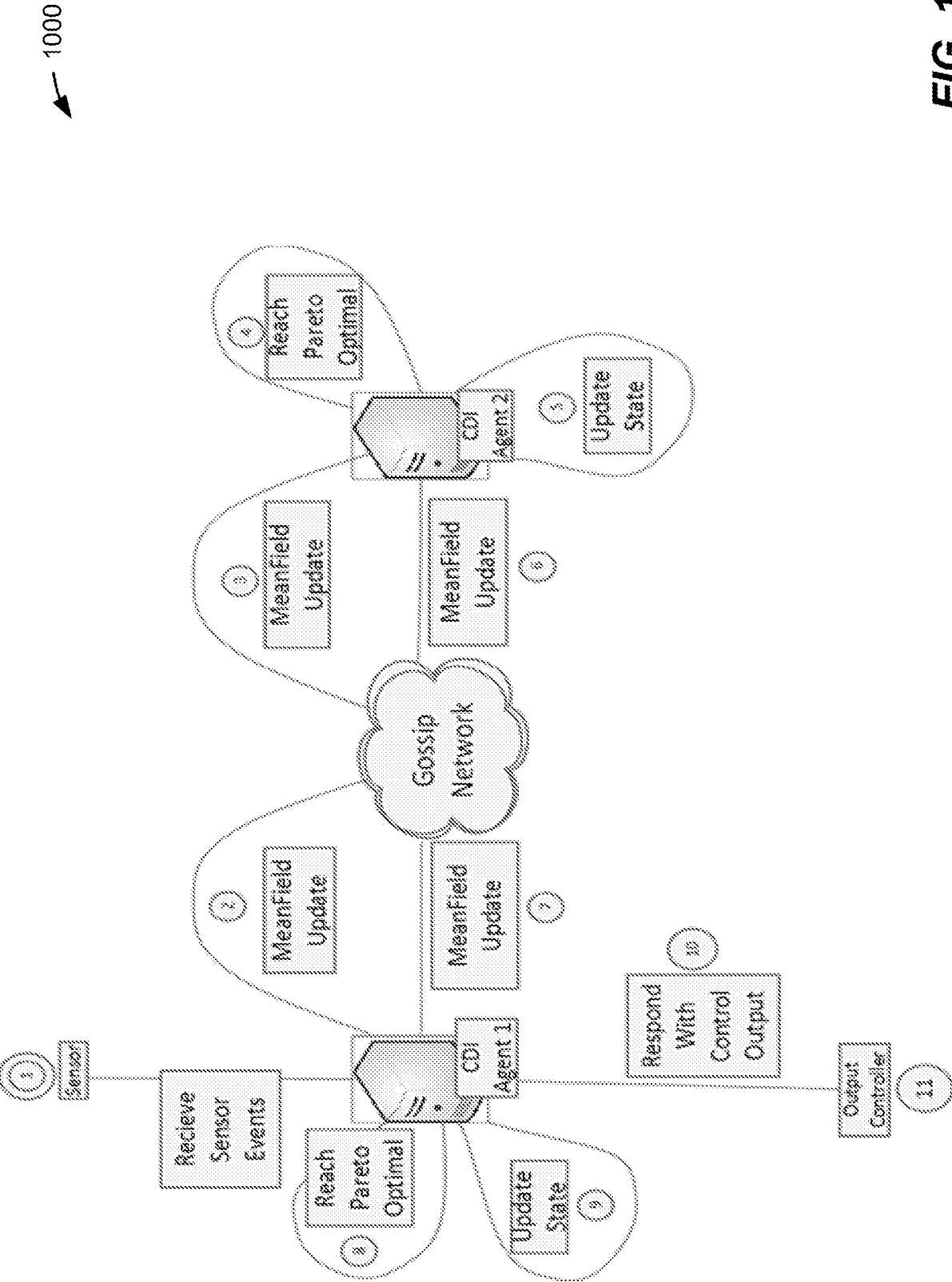
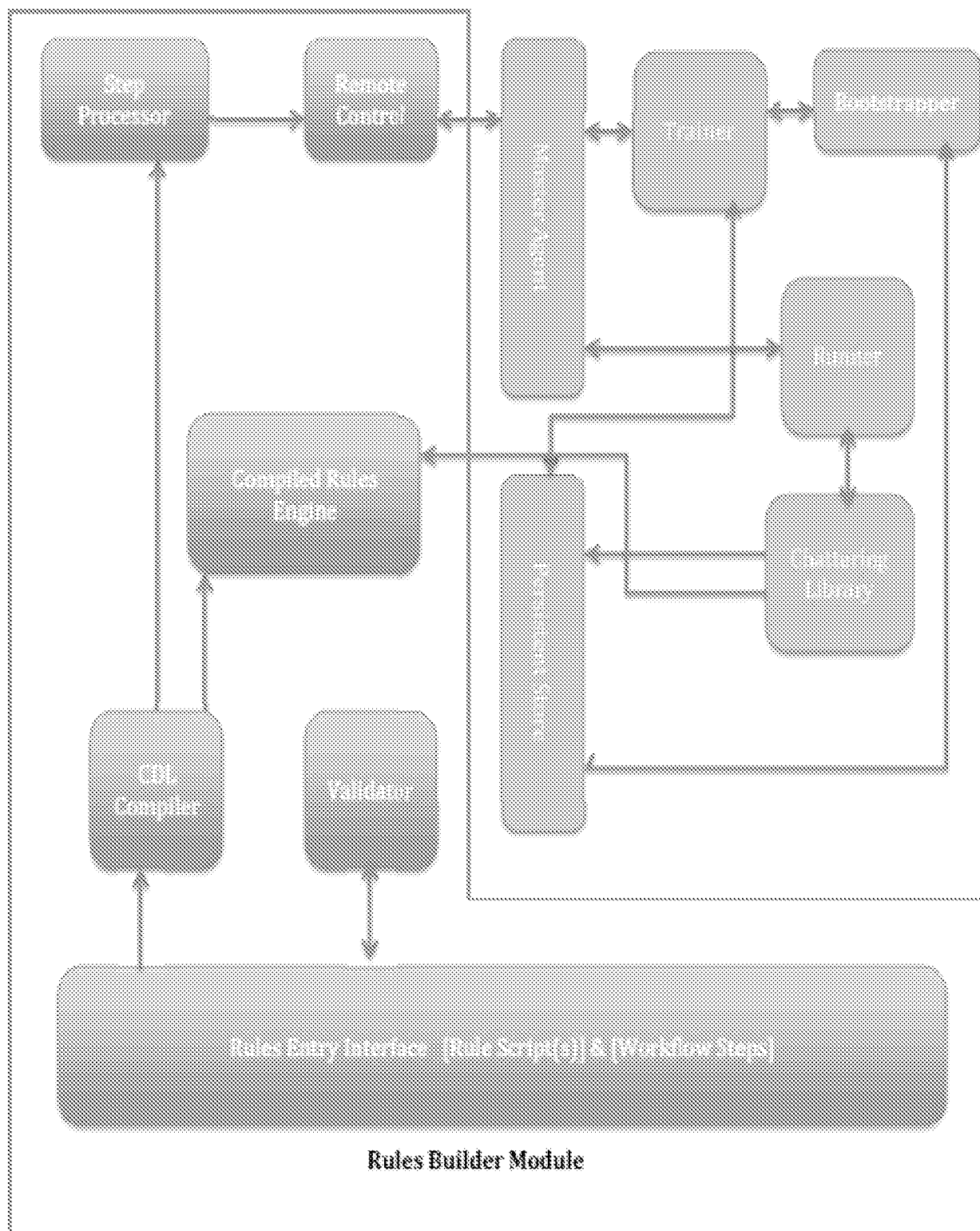


FIG. 10

15/37

FIG. 11

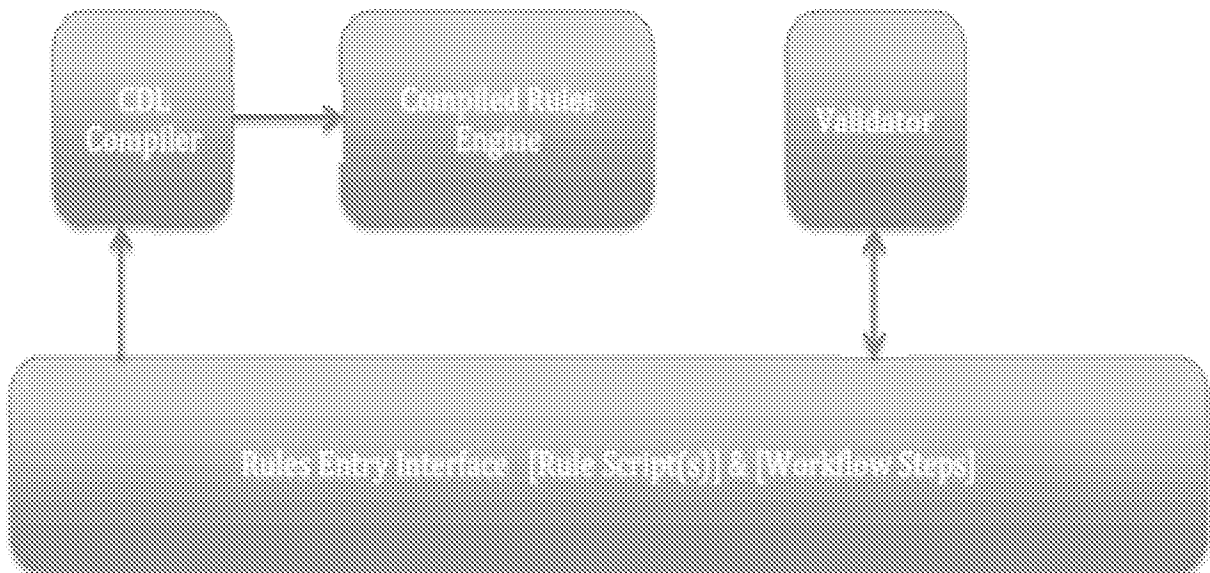
1100



16/37

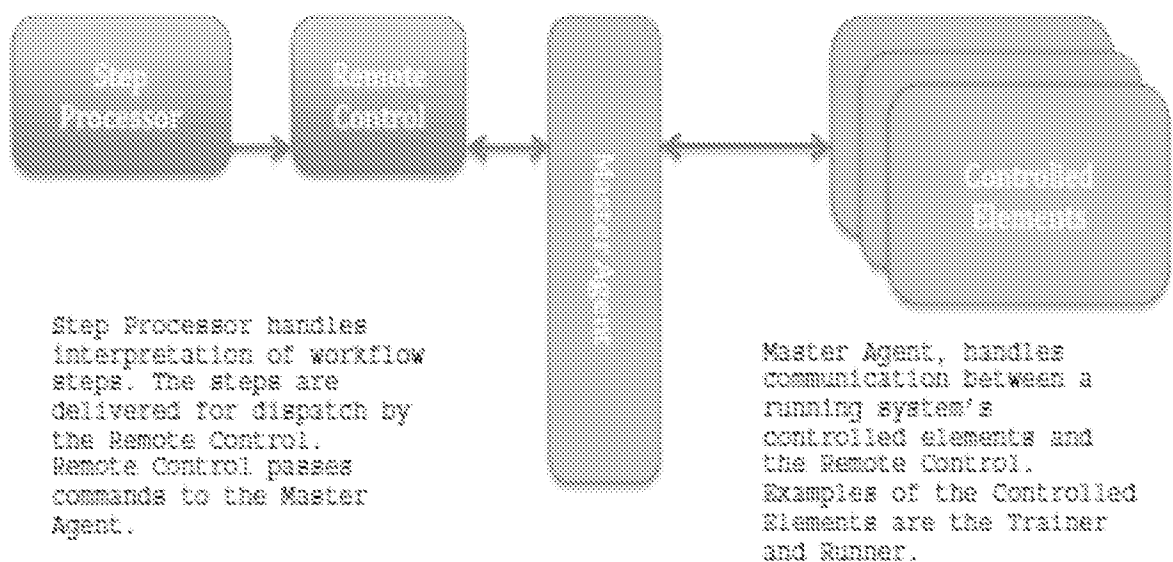
FIG. 12

1200

**FIG. 13**

1300

Diagram C.) Workflow Step Processing with Remote Control Messaging to Master Agent

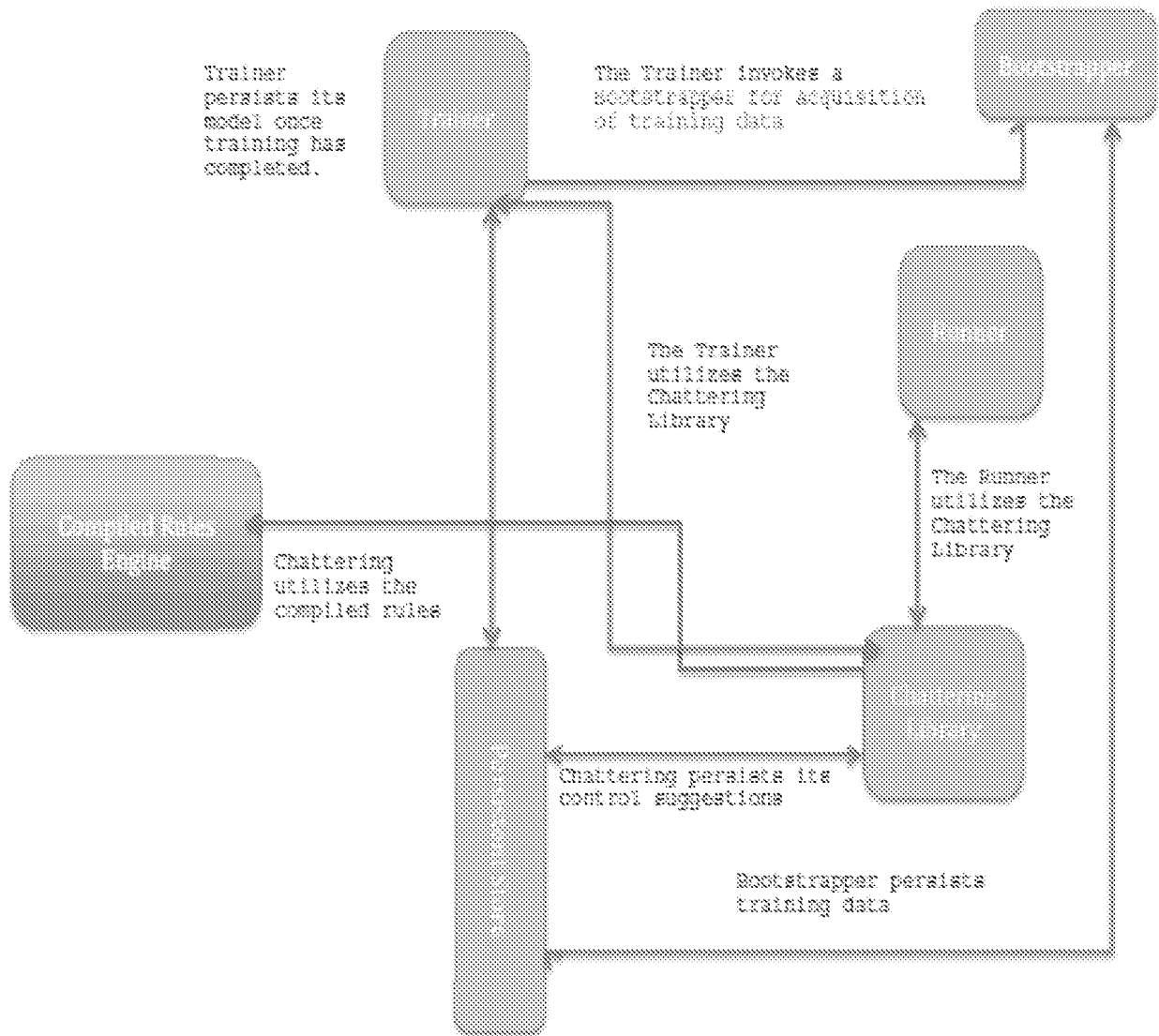


17/37

FIG. 14

1400

Diagram D.) Bootstrapping, Training, Running and Persisting state information



18/37

FIG. 15

1500

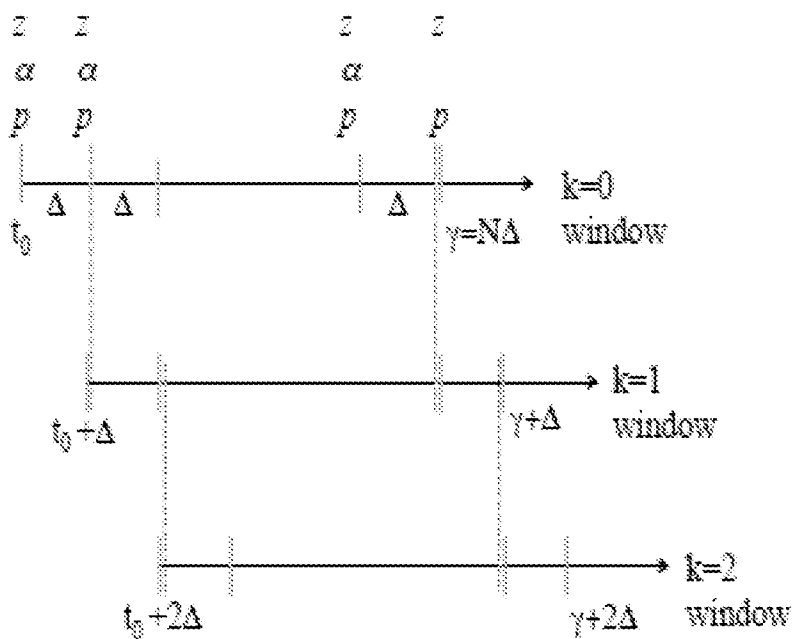
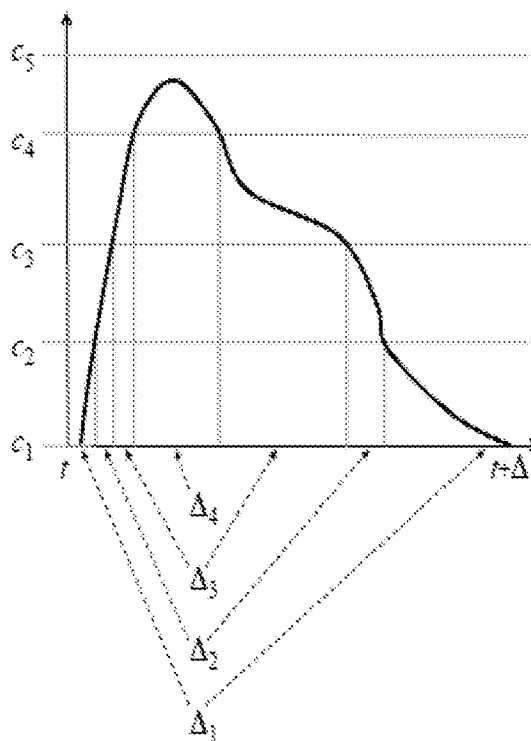


FIG. 16

1600



19/37

FIG. 17

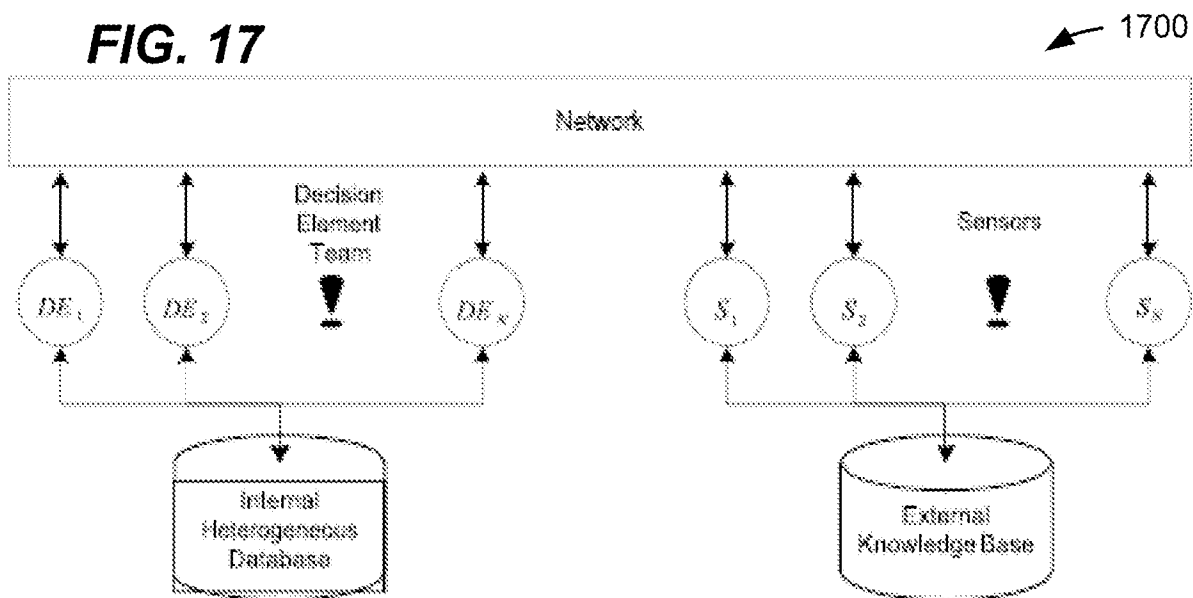


FIG. 18

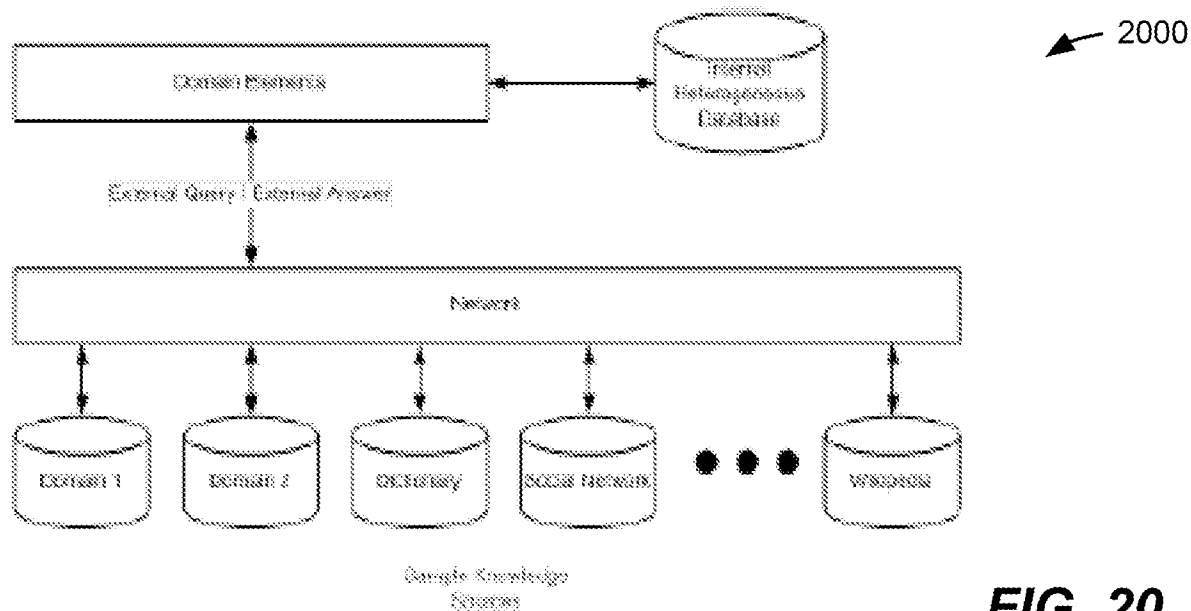
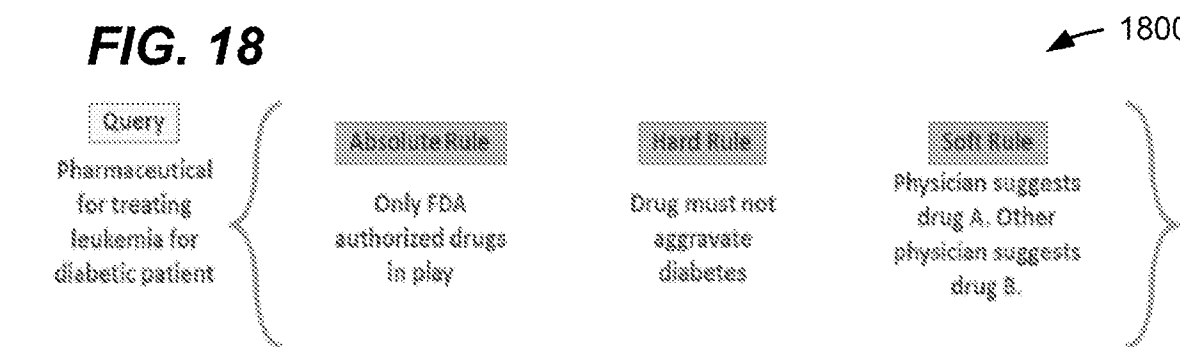


FIG. 20

FIG. 19

1900

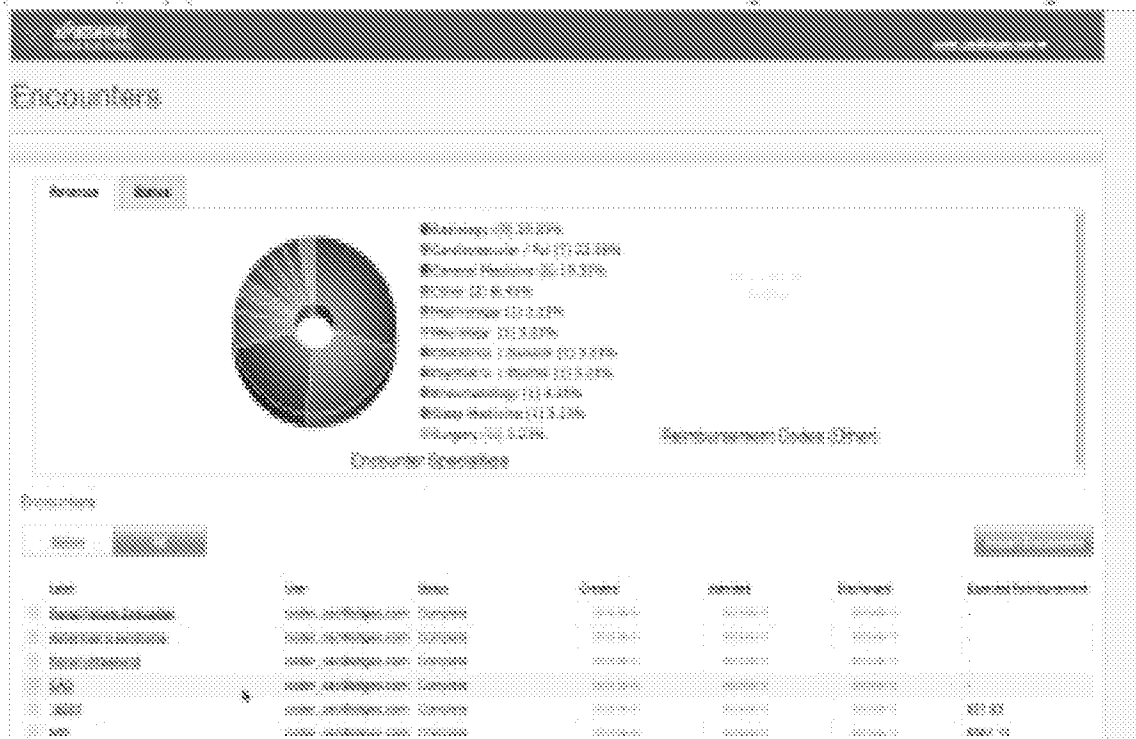


FIG. 21

2100

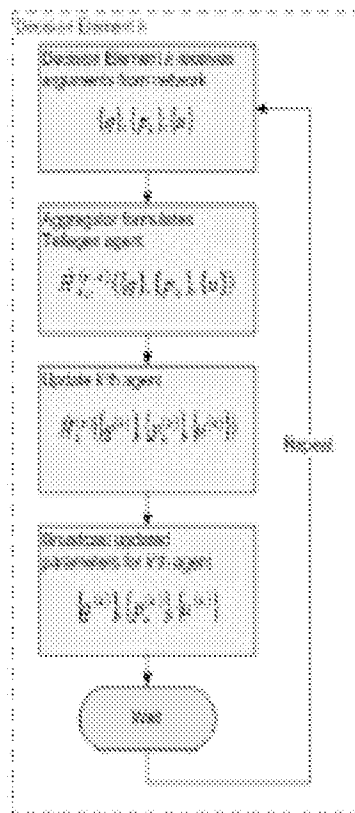
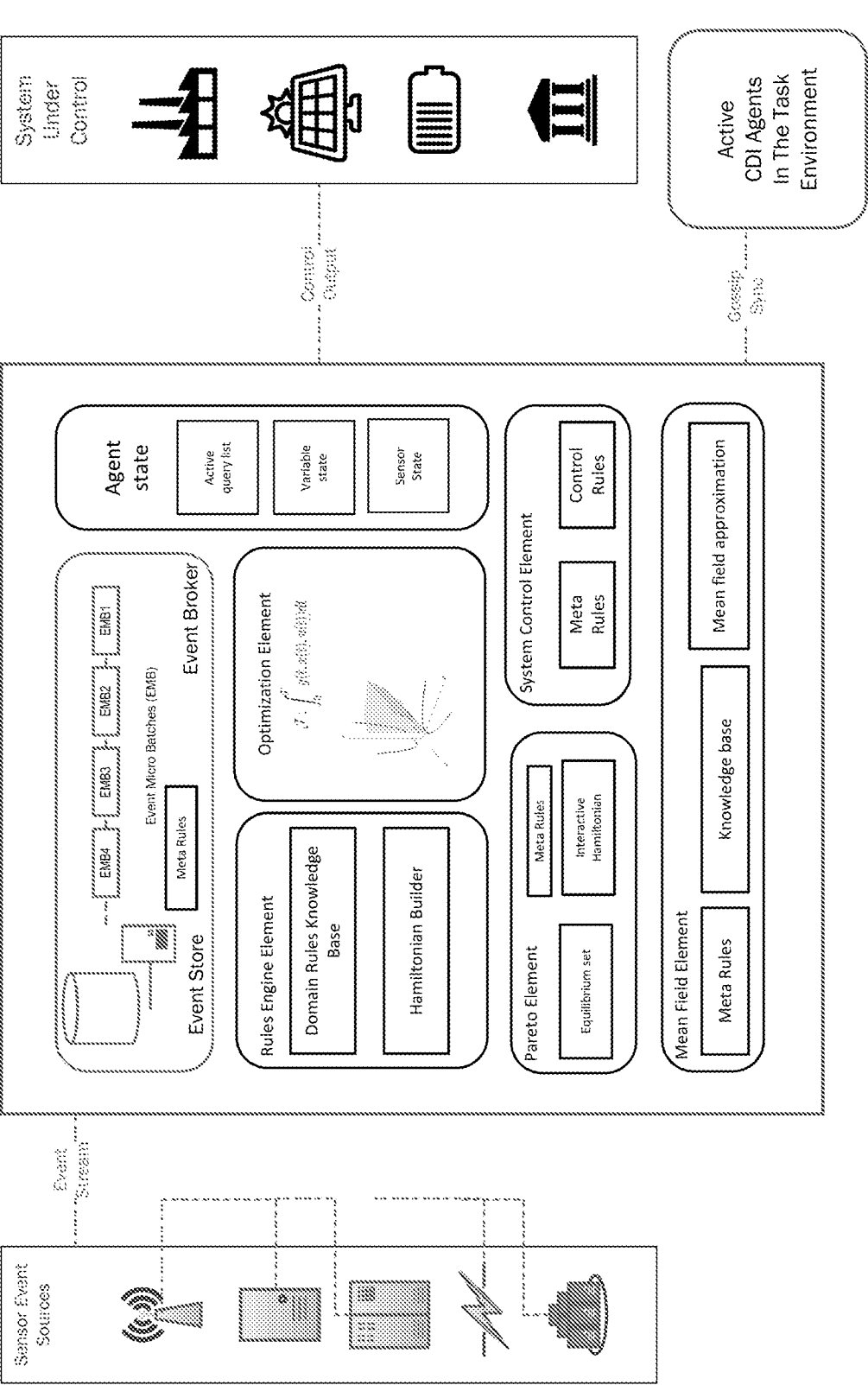


FIG. 22



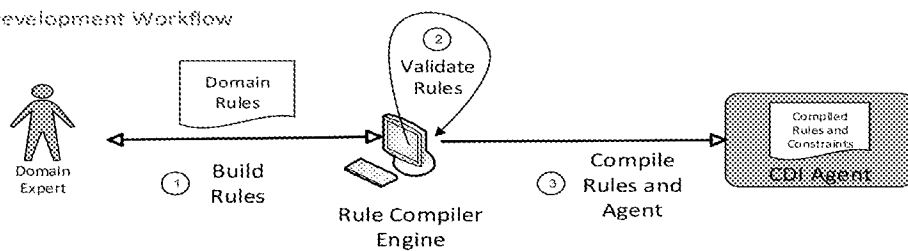
22/37

FIG. 23

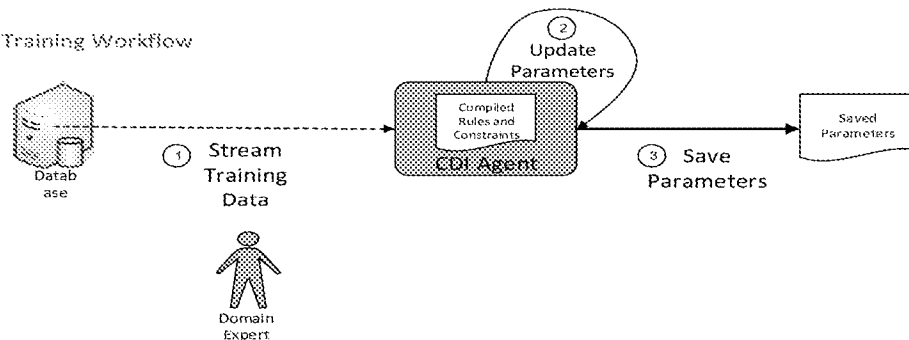
CDI Offline Workflows

2300

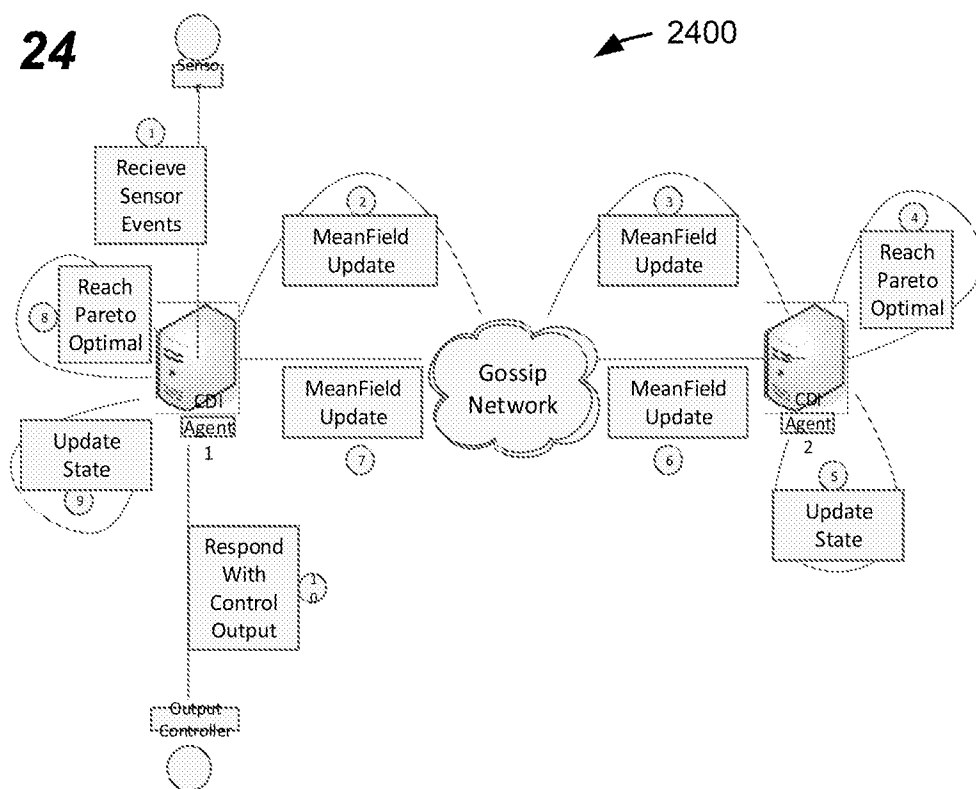
Development Workflow



Training Workflow

**FIG. 24**

2400



2500

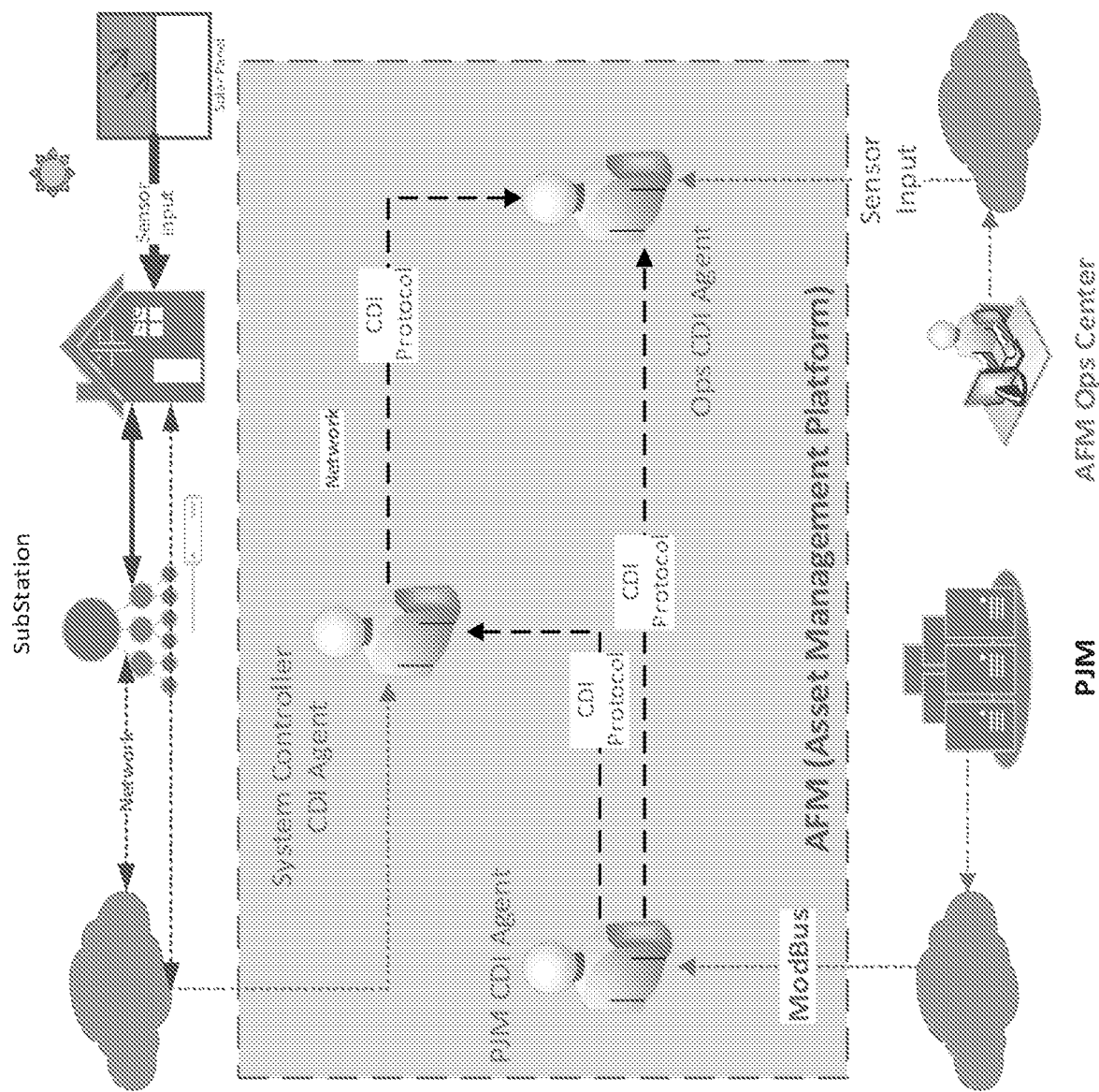
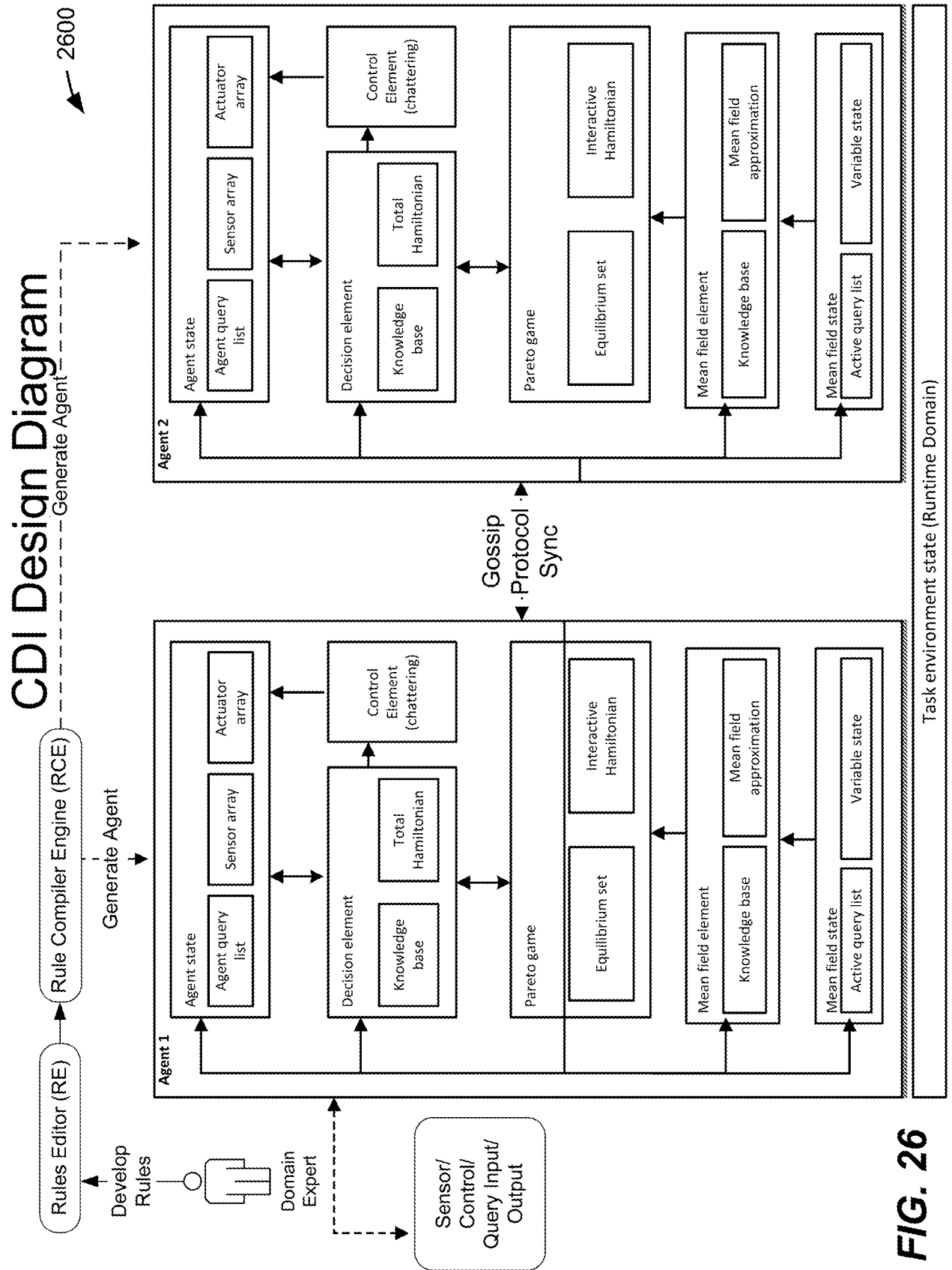


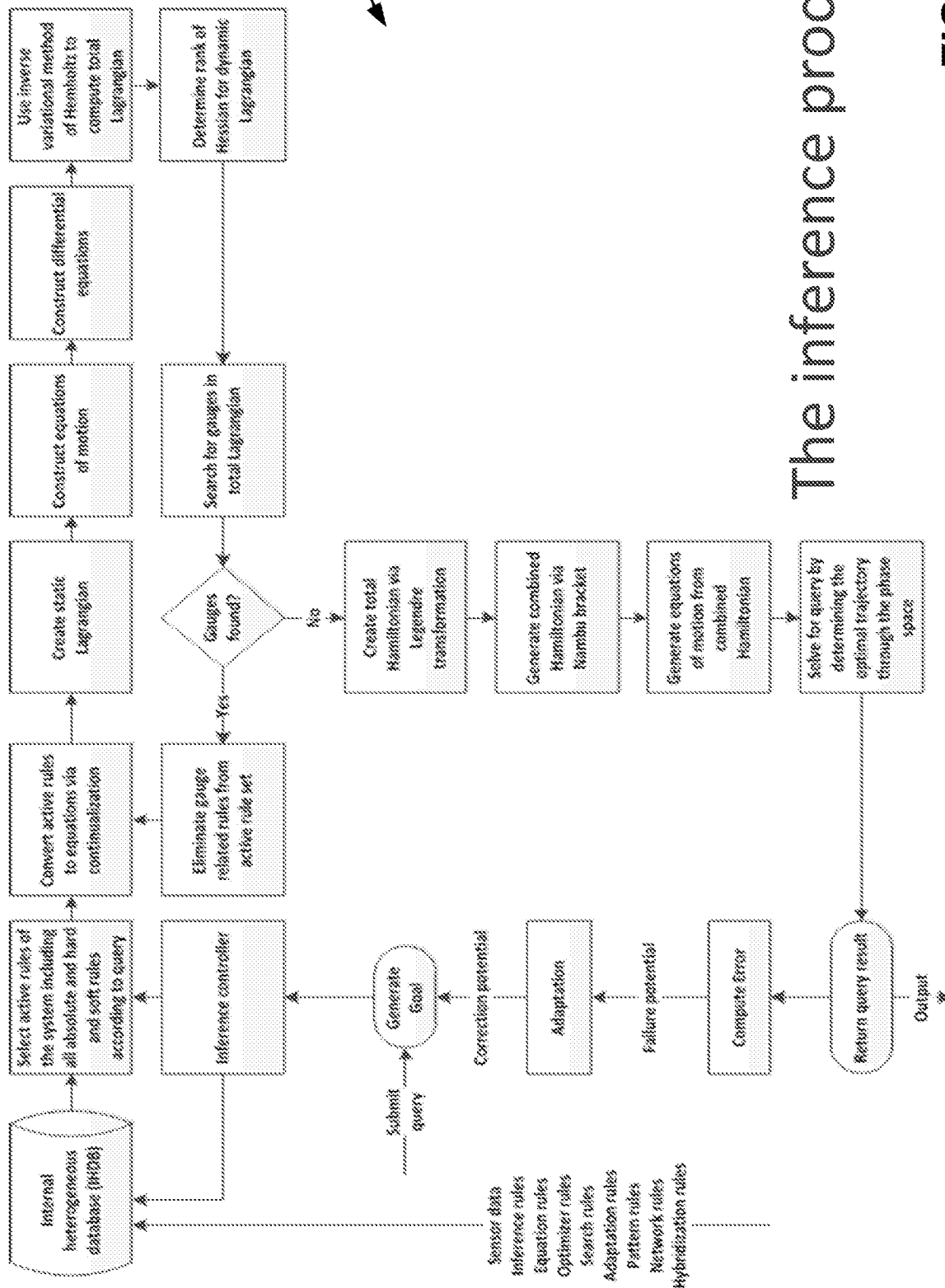
FIG. 25

24/37



25/37

2700



26/37

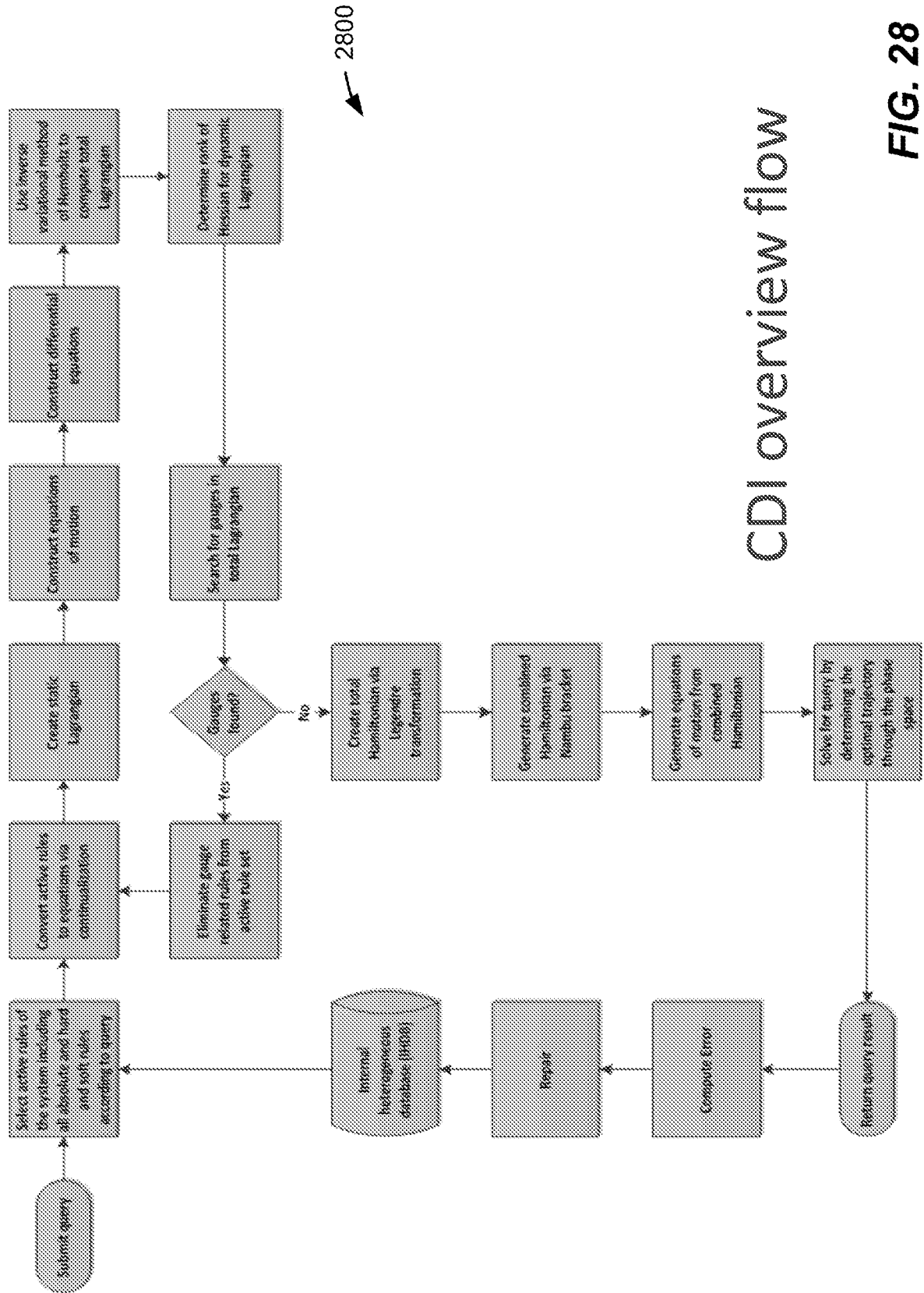


FIG. 28

2900A

HDP Step-1 An Injection map exists

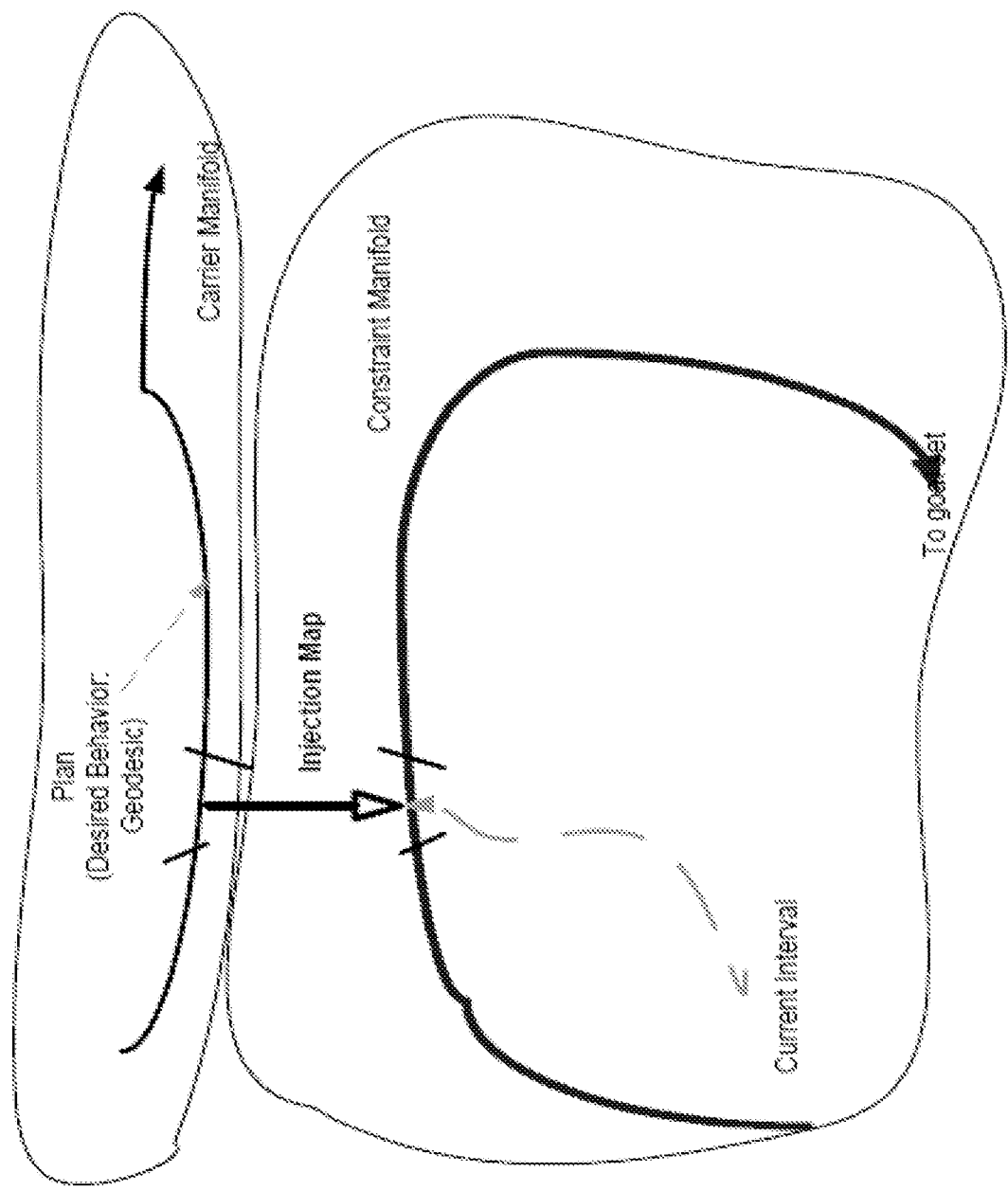


FIG. 29A

2900B

HDP Step-2a- If a Local Goal Exists

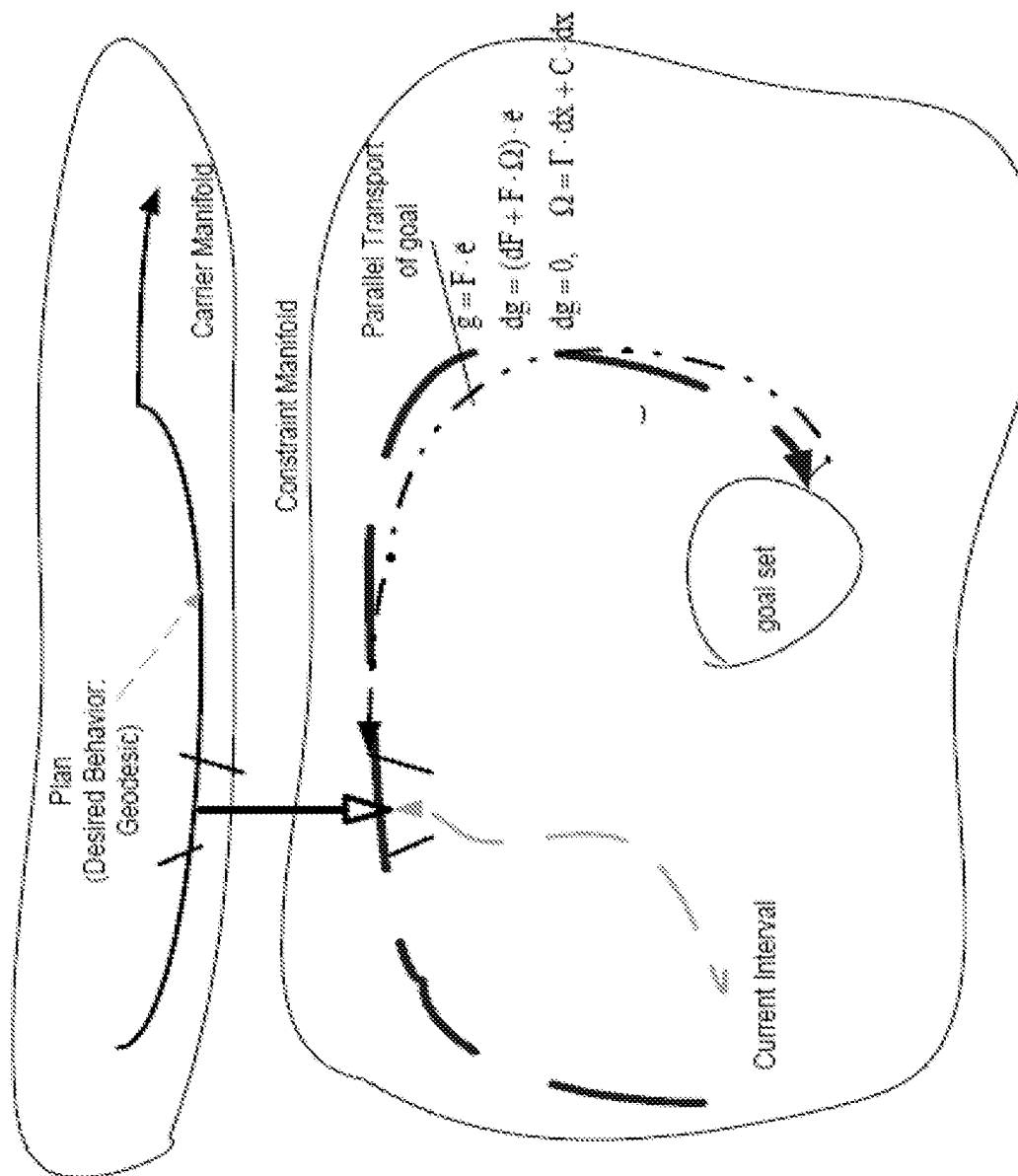


FIG. 29B

2900C

HDP Step-2b- If a Local Goal Does Not Exist

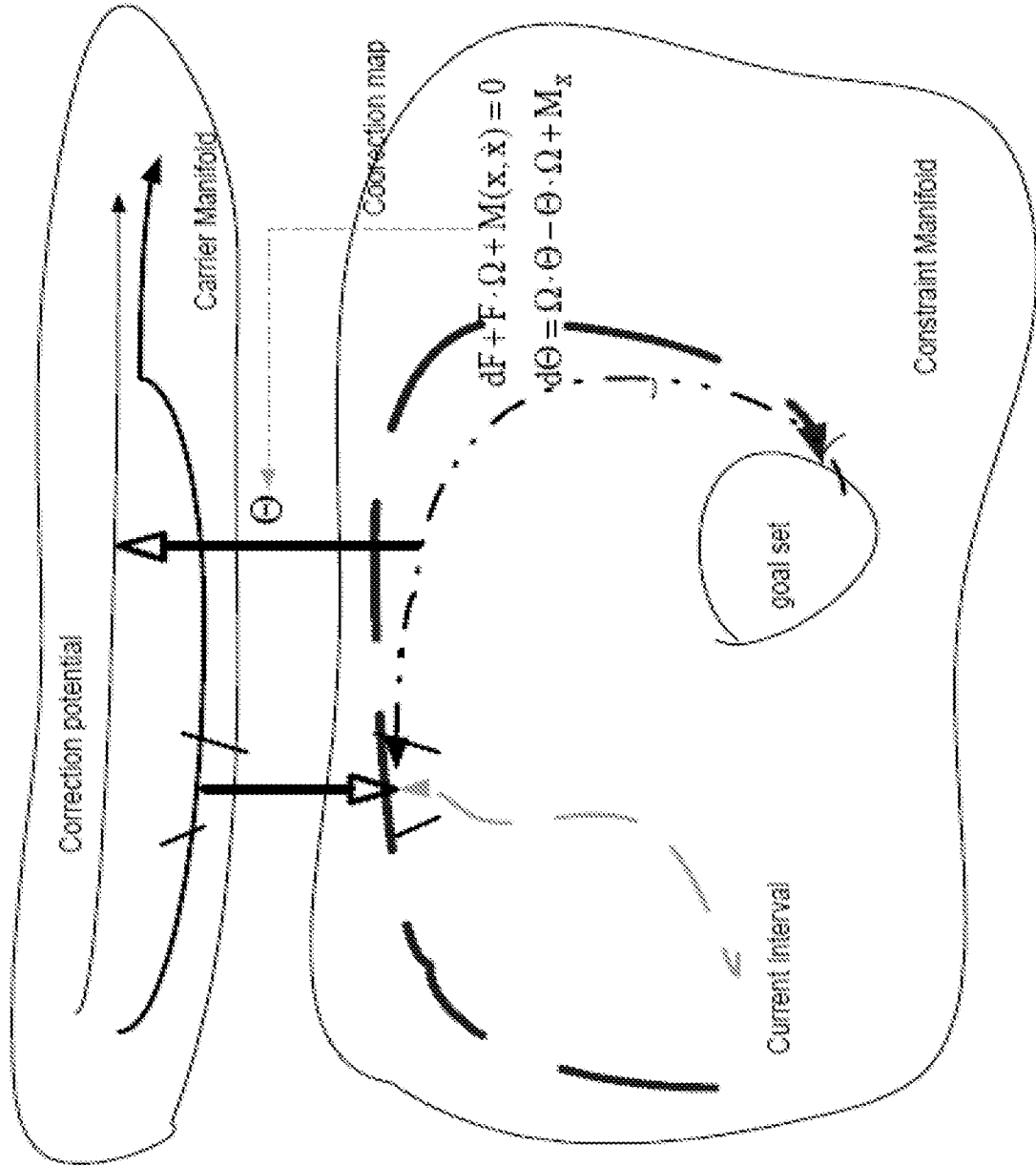


FIG. 29C

2900D

HDP STEP 3-a: -There is an optimal transition
-Constraints are satisfied
- No New rules active/deactive

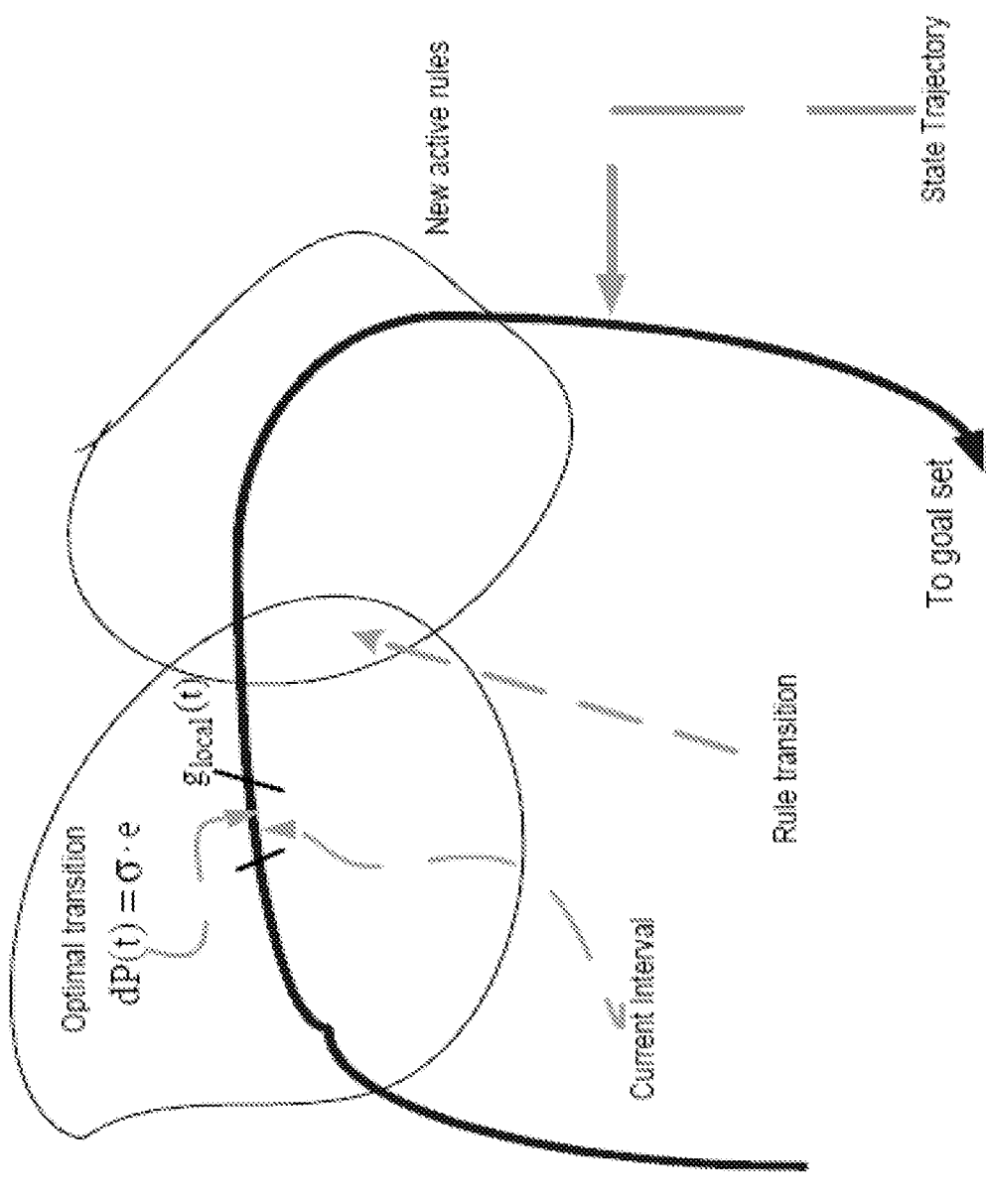


FIG. 29D

2900E

HDP STEP 3-b: -There is an optimal transition
-Constraints are satisfied
- New rules active/deactive

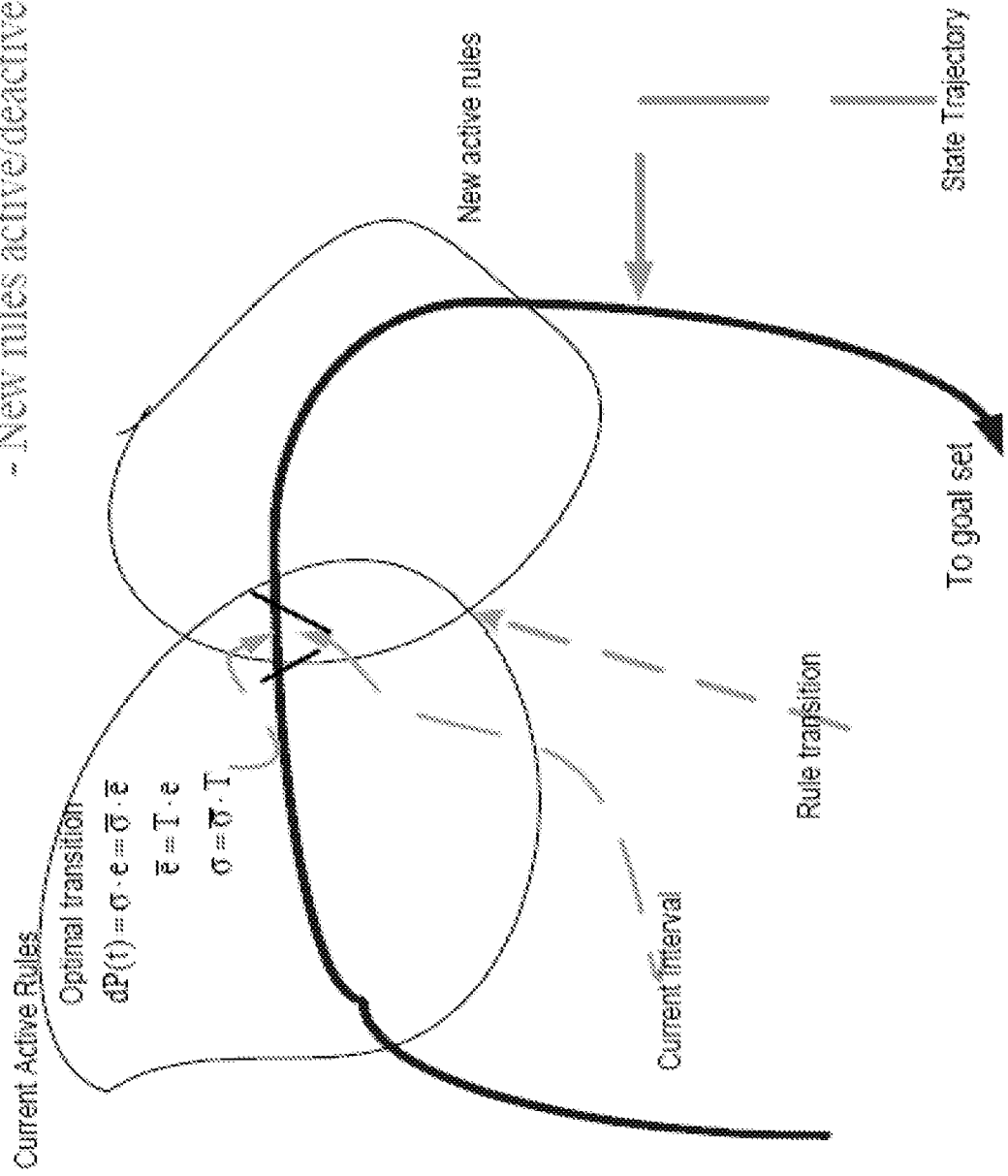


FIG. 29E

Agent Behavior is a Trajectory in Domain Manifold 2900F

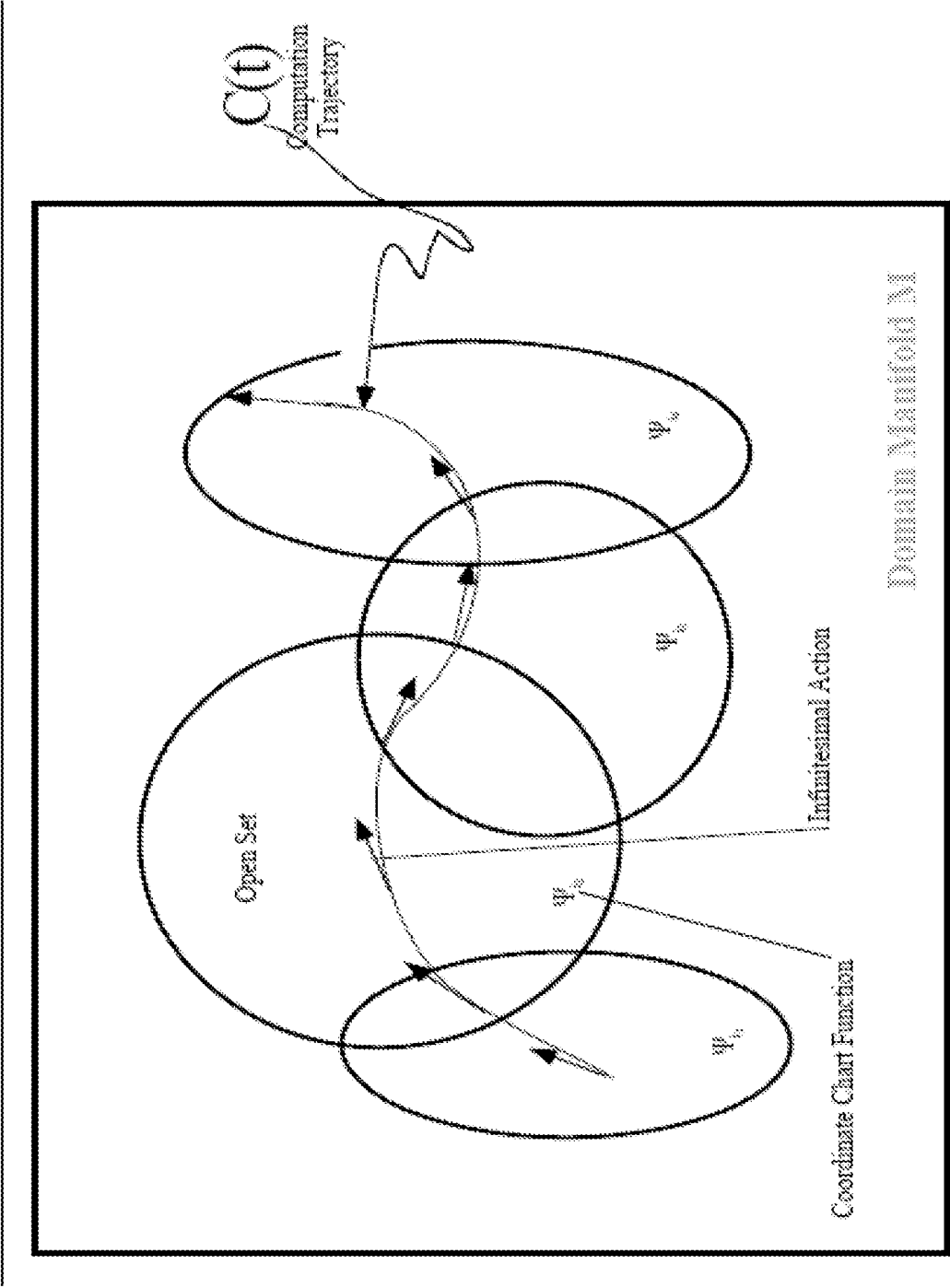
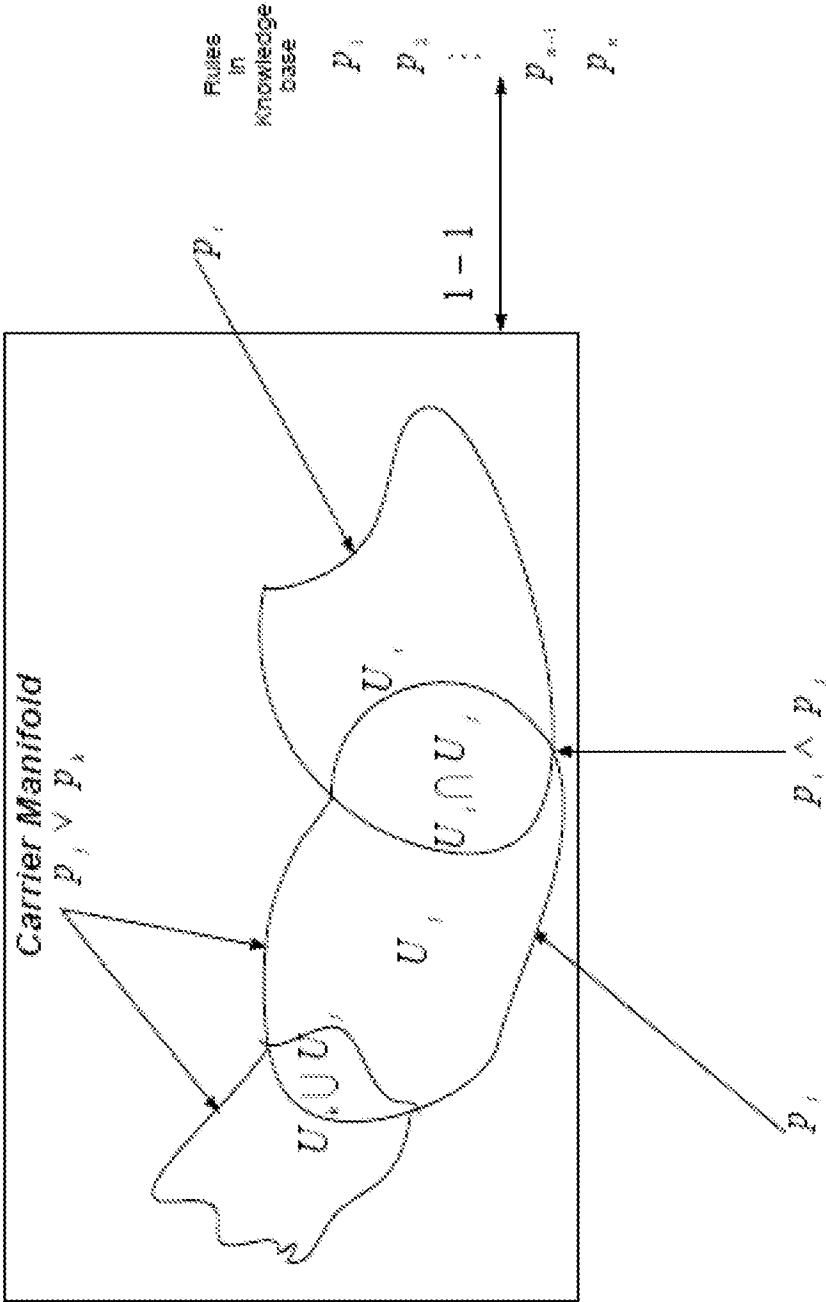


FIG. 29F

Relation Between Knowledge Base

2900G

And Carrier Manifold



As Binary Relations on the state of the process The rules in the knowledge base define a basis for the topology of the carrier manifold

FIG. 29G

Conceptual Illustration of an Agent's ϵ -Optimality Procedure

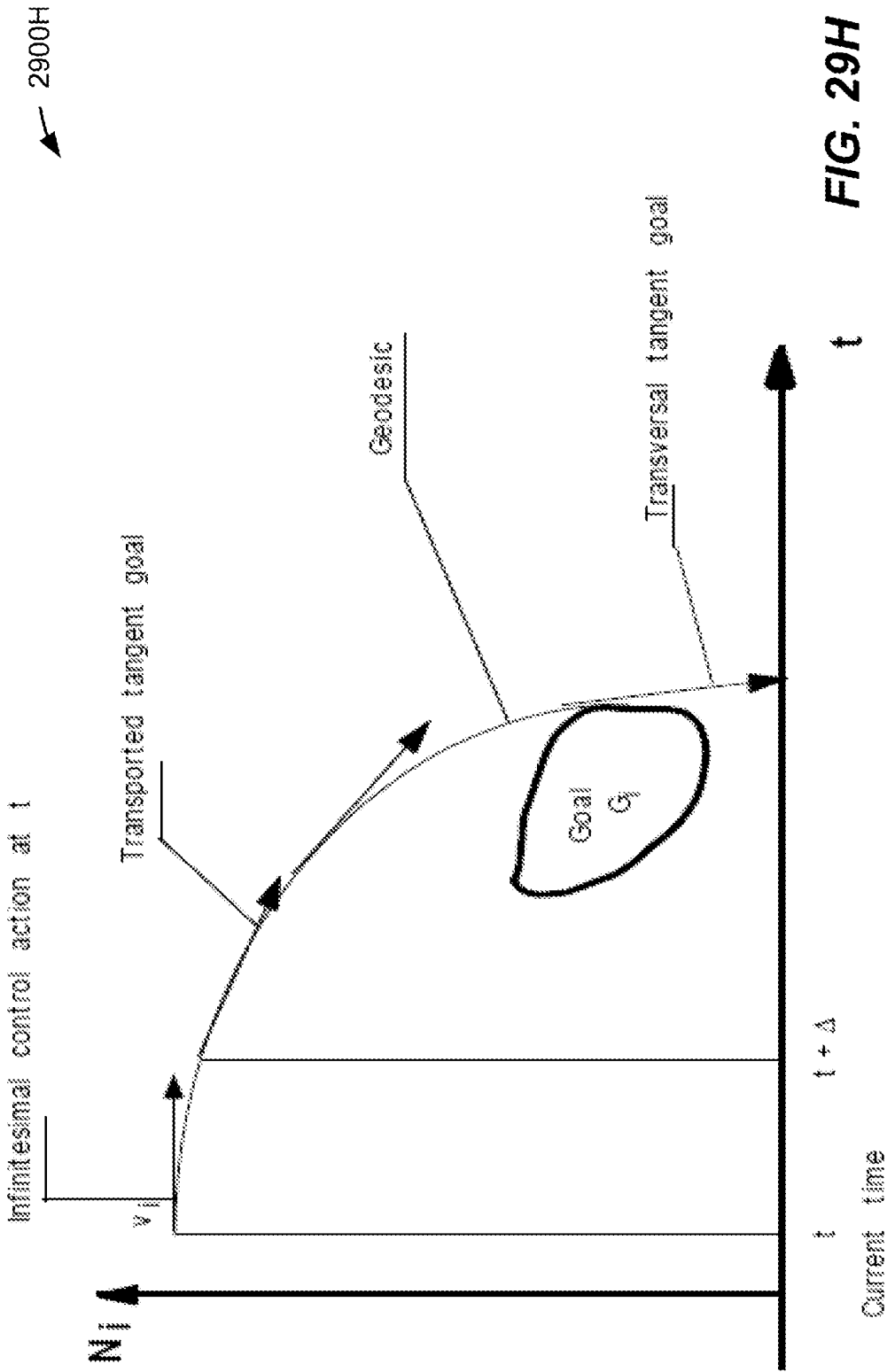


FIG. 29H

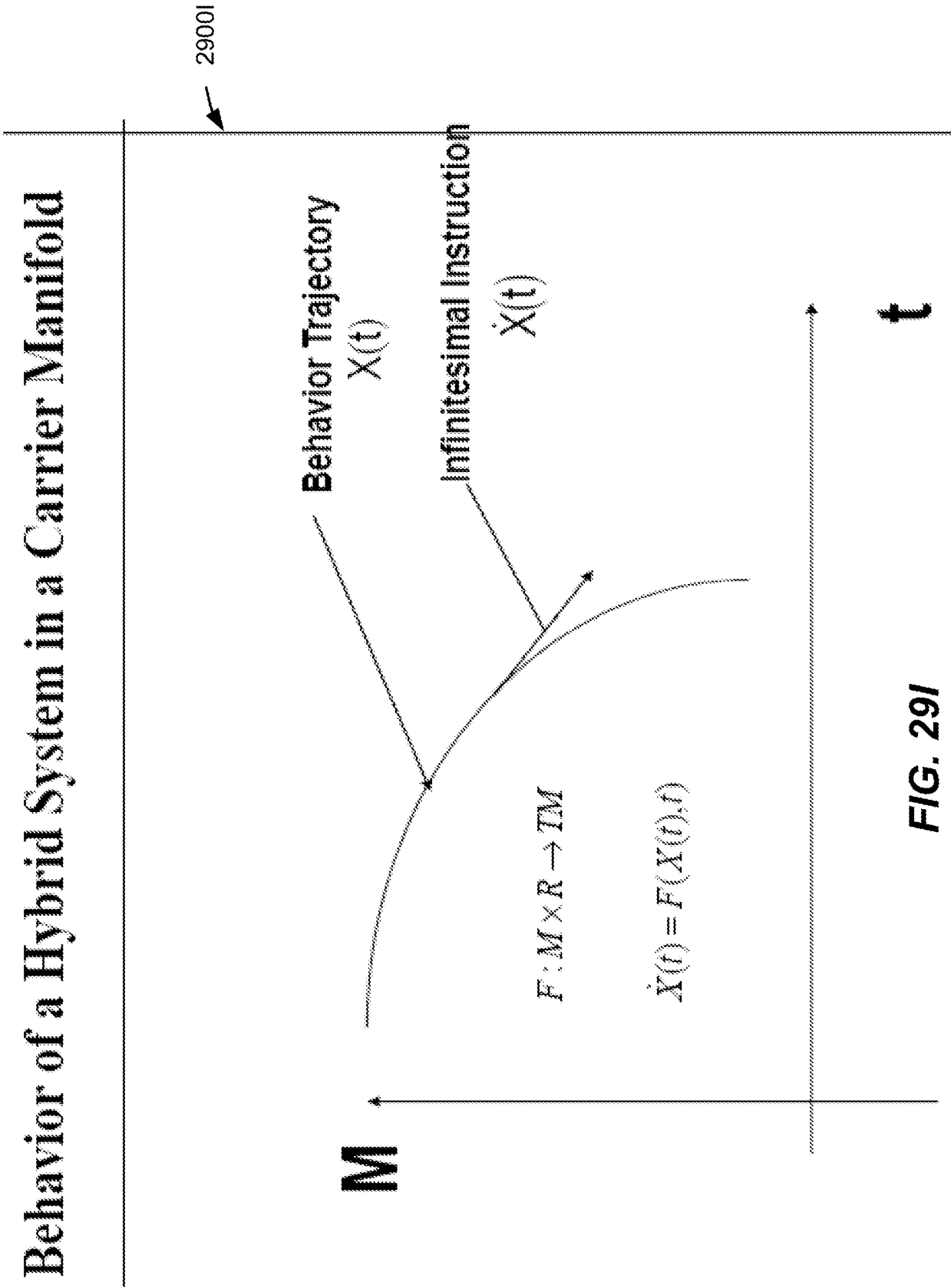


FIG. 29I

Agent Behavior is a Trajectory in Domain Manifold

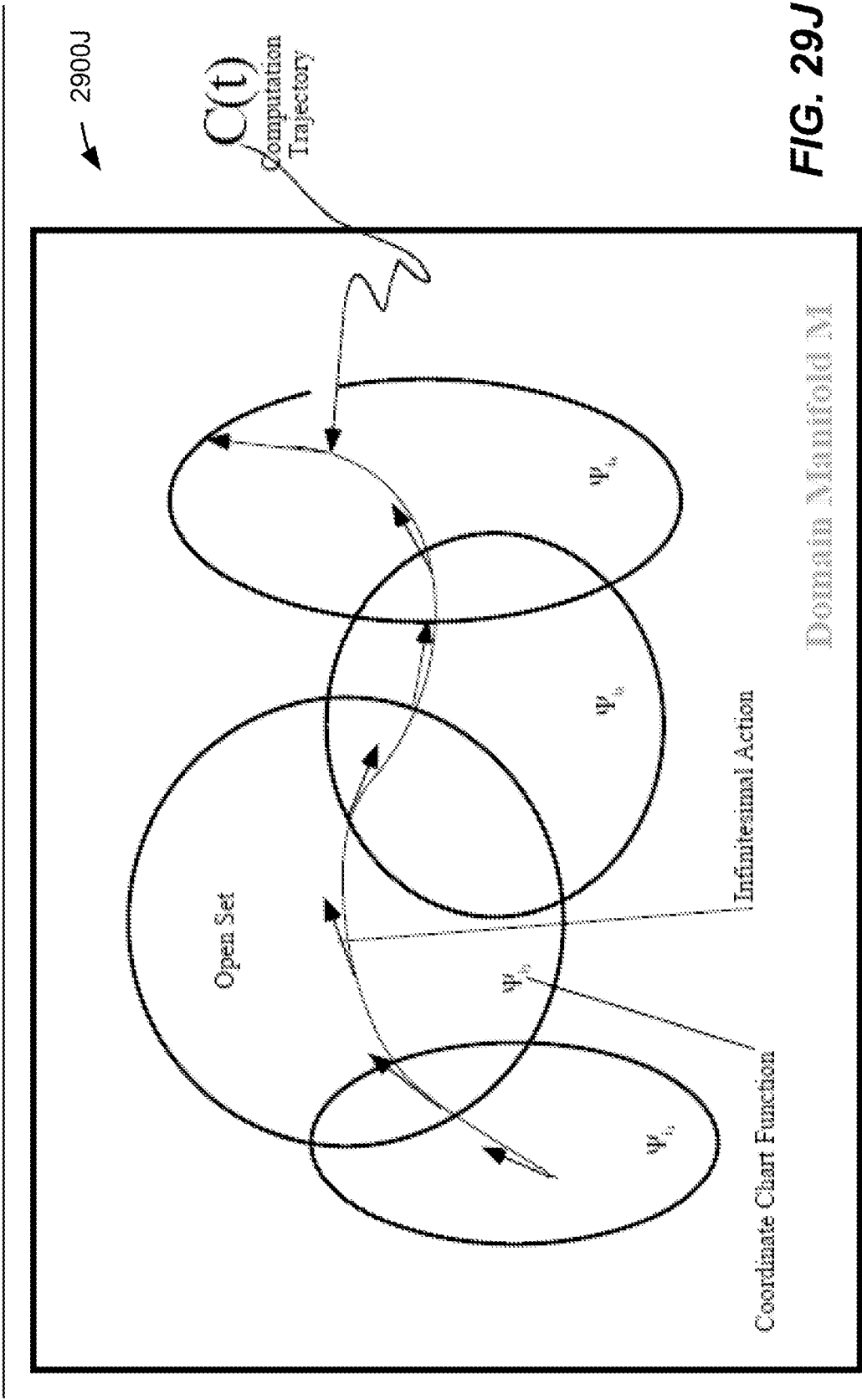


FIG. 29J

↖ 2900K

First Theorem of Chattering: Near optimal flow is integrand
of Action resulting from Chattering of primitive actions

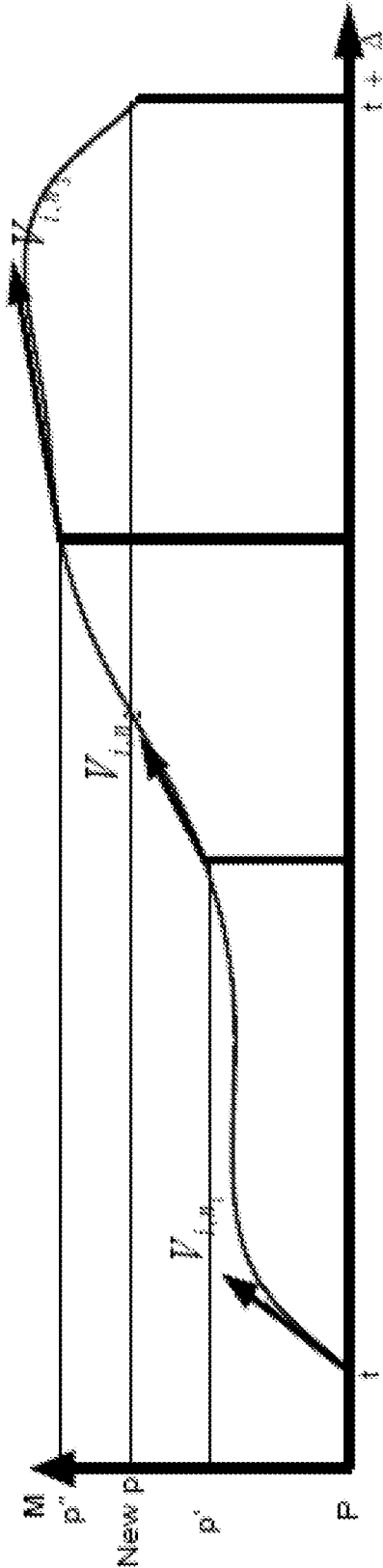


FIG. 29K

INTERNATIONAL SEARCH REPORT

International application No.

PCT/US15/37022

A. CLASSIFICATION OF SUBJECT MATTER

IPC(8) - G05B 19/418 (2015.01)

CPC - G05B 19/41865

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)

IPC(8): G05B 13/02, 15/02, 17/02, 19/418, 23/02; G06F 17/50, 19/00; G06Q 10/00 (2015.01)

CPC: G05B 13/02, 15/02, 17/02, 19/41865, 23/0205; G06F 17/50, 19/00; G06Q 10/06

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)

PatSeer (US, EP, WO, JP, DE, GB, CN, FR, KR, ES, AU, IN, CA, INPADOC Data); ProQuest; IEEE; Google/Google Scholar; collaborativ*, distribut*, decision*, user*, outputs*, values*, rule*, restriction*, true*, false*, sensor*, differential*, equation*, linear*, analy*, control*, actions*, goal*, threshold*, amount*

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
A	US 2011/0178622 A1 (TUSZYNSKI, S) 21 July 2011, Abstract, Paragraphs [0015], [0050], [0056], [0059]-[0062], [0067].	1-135
A	US 2003/0069868 A1 (VOS, J) 10 April 2003, [0012], [0234].	1-135
A	US 5727128 A (MORRISON, S) 10 March 1998, entire document.	1-135
A	US 2006/0229769 A1 (GRICHNIK, A et al.) 12 October 2006, entire document.	1-135
A	US 2013/0080530 A1 (FREES, L et al.) 28 March 2013, entire document.	1-135
A	US 2013/0253942 A1 (HONG KONG BAPTIST UNIVERSITY) 26 September 2013, entire document.	1-135

☐ Further documents are listed in the continuation of Box C.☐ See patent family annex.

* Special categories of cited documents:

"A" document defining the general state of the art which is not considered to be of particular relevance

"E" earlier application or patent but published on or after the international filing date

"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)

"O" document referring to an oral disclosure, use, exhibition or other means

"P" document published prior to the international filing date but later than the priority date claimed

"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention

"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone

"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art

"&" document member of the same patent family

Date of the actual completion of the international search

31 August 2015 (31.08.2015)

Date of mailing of the international search report

21 SEP 2015

Name and mailing address of the ISA/

Mail Stop PCT, Attn: ISA/US, Commissioner for Patents
P.O. Box 1450, Alexandria, Virginia 22313-1450

Facsimile No. 571-273-8300

Authorized officer

Shane Thomas

PCT Helpdesk: 571-272-4300

PCT OSP: 571-272-7774