Innovation, Sciences et Développement économique Canada
Office de la Propriété Intellectuelle du Canada

Innovation, Science and Economic Development Canada
Canadian Intellectual Property Office

CA 3092647 C 2022/12/06

(11)(21) 3 092 647

(12) BREVET CANADIEN
CANADIAN PATENT

(13) C

(86) Date de dépôt PCT/PCT Filing Date: 2019/03/04
(87) Date publication PCT/PCT Publication Date: 2019/09/06
(45) Date de délivrance/Issue Date: 2022/12/06
(85) Entrée phase nationale/National Entry: 2020/08/28
(86) N° demande PCT/PCT Application No.: US 2019/020536
(87) N° publication PCT/PCT Publication No.: 2019/169384
(30) Priorité/Priority: 2018/03/02 (US62/637,934)

(51) Cl.Int./Int.Cl. *G06F 17/14* (2006.01), *G06F 17/16* (2006.01), *G06N 3/02* (2006.01), *G06N 7/08* (2006.01), *G16C 10/00* (2019.01), *G16C 20/70* (2019.01)

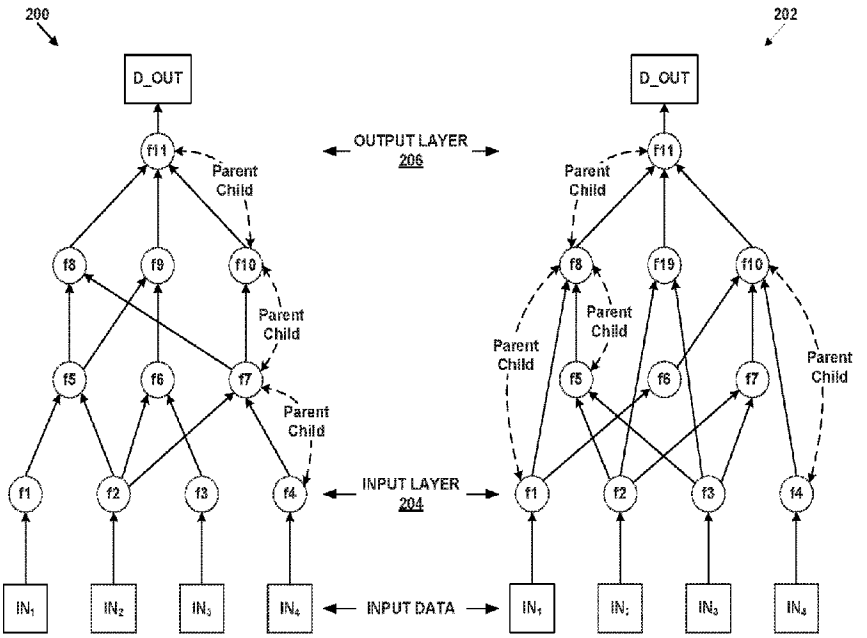
(72) Inventeur/Inventor:
KONDOR, IMRE, US

(73) Propriétaire/Owner:
THE UNIVERSITY OF CHICAGO, US

(74) Agent: FASKEN MARTINEAU DUMOULIN LLP

(54) Titre : ARCHITECTURE DE RESEAU NEURONAL COVARIANTE POUR DETERMINER DES POTENTIELS ATOMIQUES

(54) Title: COVARIANT NEURAL NETWORK ARCHITECTURE FOR DETERMINING ATOMIC POTENTIALS




(57) Abrégé/Abstract:
Methods and systems for computationally simulating an N-body physical system are disclosed. A compound object X having N elementary parts E may be decomposed into J subsystems, each including one or more of the elementary parts and having a position vector $r_j$ and state vector $\psi_j$. A neural network having J nodes each corresponding to one of the subsystems may be constructed, the nodes including leaf nodes, a non-leaf root node, and intermediate non-leaf nodes, each being configured to compute an activation corresponding to the state of a respective subsystem. Upon receiving input data for the parts E, each node may compute $\psi_j$ from $r_j$ and $\psi_j$ of its child nodes using a covariant aggregation rule representing $\psi_j$ as a tensor that is covariant to rotations of the rotation group SO(3). A Clebsch-Gordan transform may be applied to reduce tensor products to irreducible covariant vectors, and $\psi_j$ of the root node may be computed as output of the ANN.

(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
06 September 2019 (06.09.2019)

WIPO | PCT

(10) International Publication Number
WO 2019/169384 A1

(51) International Patent Classification:
*G06F 17/14* (2006.01) *G06N 7/08* (2006.01)
*G06F 17/16* (2006.01) *G16C 10/00* (2019.01)
*G06K 9/62* (2006.01) *G16C 20/70* (2019.01)
*G06N 3/02* (2006.01)

(21) International Application Number:
PCT/US2019/020536

(22) International Filing Date:
04 March 2019 (04.03.2019)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
62/637,934 02 March 2018 (02.03.2018) US

(71) Applicant: **THE UNIVERSITY OF CHICAGO** [US/US]; 5801 S. Ellis Street, Chicago, IL 60637 (US).

(72) Inventor: **KONDOR, Imre**; Department of Computer Science - Ryerson Laboratory, 1100 East 58th Street, Chicago, IL 60637 (US).

(74) Agent: **GRABELSKY, David, A.**; McDonnell Boehnen Hulbert & Berghoff LLP, 300 South Wacker Drive, Chicago, IL 60606 (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, DJ, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IR, IS, JO, JP, KE, KG, KH, KN, KP, KR, KW, 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, LT, 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))*

WO 2019/169384 A1

(54) Title: COVARIANT NEURAL NETWORK ARCHITECTURE FOR DETERMINING ATOMIC POTENTIALS

(57) Abstract: Methods and systems for computationally simulating an N-body physical system are disclosed. A compound object X having N elementary parts E may be decomposed into J subsystems, each including one or more of the elementary parts and having a position vector $r_j$ and state vector $\psi_j$. A neural network having J nodes each corresponding to one of the subsystems may be constructed, the nodes including leaf nodes, a non-leaf root node, and intermediate non-leaf nodes, each being configured to compute an activation corresponding to the state of a respective subsystem. Upon receiving input data for the parts E, each node may compute $\psi_j$ from $r_j$ and $\psi_j$ of its child nodes using a covariant aggregation rule representing $\psi_j$ as a tensor that is covariant to rotations of the rotation group SO(3). A Clebsch-Gordan transform may be applied to reduce tensor products to irreducible covariant vectors, and $\psi_j$ of the root node may be computed as output of the ANN.

# Covariant Neural Network Architecture for Determining Atomic Potentials


## Statement Regarding Federally Sponsored Research or Development

**[0001]** This invention was made with government support under grant number D16AP00112 awarded by the Defense Advanced Research Projects Agency. The government has certain rights in the invention.


## [0002] Background

**[0003]** In principle, quantum mechanics provides a perfect description of the forces governing the behavior of atomic systems such as crystals and biological molecules. However, for systems larger than a few dozen atoms, solving the Schrodinger equation explicitly, on present day computers, is generally not a feasible proposition. Density Functional Theory (DFT), a widely used approximation in quantum chemistry, has trouble scaling to more than about a hundred atoms.

**[0004]** In view of such limitations, a majority of practical work in molecular dynamics typically foregoes modeling electrons explicitly, and falls back on the fundamentally classical (i.e., non-quantum) Born–Oppenheimer approximation, which treats atoms as solid balls that exert forces on nearby balls prescribed by so called (effective) atomic potentials. This



Date reçue / Date received 2021-11-22

approximation assumes that the potential attached to atom $i$ is $\phi_i(\hat{r}_1, \ldots, \hat{r}_k)$, with $\hat{r}_j = r_{p_j} - r_i$, where $r_i$ is the position vector of atom $i$ and $r_{pj}$ is the position vector of its $j$'th neighbor. The total force experienced by atom $i$ is then simply the negative gradient $F_i = -\nabla_{r_i}\phi_i(\hat{r}_1, \ldots, \hat{r}_k)$. Classically, in molecular dynamics $\phi_i$ is usually given in terms of a closed form formula with a few tunable parameters. Know techniques in this area are usually characterized according to empirical potentials or empirical force fields.

**[0005]** While empirical potentials may be fast to evaluate, they are crude models of the quantum interactions between atoms, limiting the accuracy of molecular simulation. More recently, machine learning has been applied to molecular simulations, showing some promise to bridge the gap between the quantum and classical worlds by *learning* the aggregate force on each atom as a function of the positons of its neighbors from a relatively small number of DFT calculations. Since its introduction, the amount of research and development in so-called machine learned atomic potentials (MLAP) has expanded significantly, and molecular dynamics simulations based on this approach may be showing evidence of results that outperform other methods.

## Summary

[0006] Much of the work in machine learning algorithms in area of, and related to, molecular simulations has been applied to the MLAP problem, from genetic algorithms, through kernel methods, to neural networks. However, the inventor has recognized that rather than the statistical details of the specific learning algorithm, a more appropriate focus for problems of this type may be the representation of the atomic environment, i.e., the choice of learning features that the algorithm is based on. This situation may arise in other areas applied machine learning as well. For example, such representational issues also play a role in computer vision and speech recognition. What makes the situation in Physics applications somewhat special is the presence of constraints and invariances that the representation must satisfy not just in an approximate, but in the *exact* sense. Rotation invariance provides instructive, contrasting examples. Specifically, if rotation invariance is not fully respected by an image recognition system, some objects might be less likely to be accurately detected in certain orientations than in others. In a molecular dynamics setting, however, using a potential that is not fully rotationally invariant would not just degrade accuracy, but would likely lead to entirely unphysical molecular trajectories.

[0007] Recent efforts in MLAP work have been shifting from fixed input features towards representations learned from the data itself, exemplified in particular by application of "deep" neural networks to represent atomic environments. It has been recognized that certain concepts from the mainstream neural networks research, such as convolution and equivariance, can be repurposed to this domain. This may reflect an underlying analogy between MLAP and computer vision. More particularly, in both domains two competing objectives need to be met for success:

1. The ability to capture structure in the input data at multiple different length (or size) scales, i.e., to construct a *multiscale* representation of the input image or the atomic environment.
2. The above-mentioned invariance property with respect to spatial transformations, including translations, rotations, and possibly scaling.

**[0008]** The inventor has further recognized that many of the concepts involved in learnable multiscale representations may be extended to create a neural network architecture where the individual "neurons" correspond to physical subsystems endowed with their own internal state. In the present disclosure, such neural networks are referred to as "N–body networks." The structure and behavior of the resulting model follows a tradition of coarse graining and representation theoretic ideas in Physics, and provides a learnable and multiscale representation of the atomic environment that is fully covariant to the action of the appropriate symmetries. What is more, the scope of the underlying ideas a broader, meaning that N–body networks have potential application in modeling other types of many-body Physical systems, as well.

**[0009]** Further still, the inventor has recognized that the machinery of group representation theory, specifically the concept of Clebsch–Gordan decompositions, can be used to design neural networks that are covariant to the action of a compact group yet are computationally efficient. This aspect is related to the other recent areas of interest involving generalizing the notion of convolutions to graphs, manifolds, and other domains, as well as the question of generalizing the concept of equivariance (covariance) in general. Analytical techniques in these recent areas have employed generalized Fourier representations of one type or another, but to ensure equivariance the nonlinearity was always applied in the time domain.

However, projecting back and forth between the time domain and the frequency domain can be a major bottleneck in terms of computation time and efficiency. In contrast, the inventor has recognized that application of the Clebsch–Gordan transform allows computation of one type of nonlinearity, namely tensor products, entirely in the Fourier domain. Accordingly, example methods and system disclosed herein provide for a significant improvement over other existing and previous analysis techniques, and provides the groundwork for efficient N–body networks for simulation and modeling of a wide variety of types of many-body Physical systems.

**[0010]** Thus, in one respect, example embodiments may involve a method for computationally simulating an N–body physical system, wherein the N–body physical system is represented mathematically as a compound object X having N elementary parts $E = \{e_i\}$, $i = 1,\ldots,N$, each $e_i$ representing one of the N bodies of the N–body physical system, wherein X is hierarchically decomposed into J subsystems, $P_j$, $j = 1,..,J$, each $P_j$ comprising one or more of the elementary parts of E, and wherein each $P_j$ is described by a position vector $\mathbf{r}_j$ and an internal state vector $\psi_j$, the method being implemented on a computing device and comprising: constructing a hierarchical artificial neural network (ANN) having J nodes each corresponding to one of the J subsystems, the J nodes including $m \geq 2$ leaf nodes as ANN inputs, $m = 1$ non-leaf root node as ANN output, and $m \geq 1$ intermediate non-leaf nodes, wherein each node is a neuron of the ANN and is configured to compute an activation corresponding to a different one of the internal state vectors $\psi_j$, and wherein: for each leaf node, $\psi_j$ describes the internal state of a respective one of the $P_j$ subsystems having just a single elementary part $e_i$, for each given intermediate non-leaf node, $\psi_j$ describes the internal state of a respective one of the $P_j$ subsystems having $2 \leq k < N$ parts $e_i$ that are each comprised in a child node of the given intermediate non-leaf node, and for the root node, $\psi_j$ describes the internal state of a subsystem

$P_j$ having $k = N$ elementary parts $e_i$ that are each comprised in a child node of the root node; at the computing device, receiving input data to the leaf nodes specifying respective position vectors and respective internal state vectors of the N elementary parts E; for each given non-leaf node, computing $\psi_j$ from the position vectors and internal states of all the child nodes of the given non-leaf node according to a covariant aggregation rule that represents $\psi_j$ as a tensor object that is covariant to rotations of the rotation group SO(3); applying a Clebsch-Gordan transform to reduce tensor products of the state vectors of the nodes to irreducible covariant vectors; and computing $\psi_j$ of the root node as output of the ANN, to determine a simulation of the internal state of the N–body physical system.

**[0011]** In another respect, example embodiments may involve a computing device configured for computationally simulating an N–body physical system, wherein the N–body physical system is represented mathematically as a compound object X having N elementary parts $E = \{e_i\}$, $i = 1,\ldots,N$, each $e_i$ representing one of the N bodies of the N–body physical system, wherein X is hierarchically decomposed into J subsystems, $P_j$, $j = 1,..,J$, each $P_j$ comprising one or more of the elementary parts of E, and wherein each $P_j$ is described by a position vector $\mathbf{r}_j$ and an internal state vector $\psi_j$, the computing device comprising: one or more processors; and memory configured to store computer-executable instructions that, when executed by the one or more processors, cause the computing device to carry out computational operations including: constructing a hierarchical artificial neural network (ANN) having J nodes each corresponding to one of the J subsystems, the J nodes including $m \geq 2$ leaf nodes as ANN inputs, $m = 1$ non-leaf root node as ANN output, and $m \geq 1$ intermediate non-leaf nodes, wherein each node is a neuron of the ANN and is configured to compute an activation corresponding to a different one of the internal state vectors $\psi_j$, and wherein: for each leaf node,

$\psi_j$ describes the internal state of a respective one of the $P_j$ subsystems having just a single elementary part $e_i$, for each given intermediate non-leaf node, $\psi_j$ describes the internal state of a respective one of the $P_j$ subsystems having $2 \leq k < N$ parts $e_i$ that are each comprised in a child node of the given intermediate non-leaf node, and for the root node, $\psi_j$ describes the internal state of a subsystem $P_j$ having $k = N$ elementary parts $e_i$ that are each comprised in a child node of the root node; receiving input data to the leaf nodes specifying respective position vectors and respective internal state vectors of the N elementary parts E; for each given non-leaf node, computing $\psi_j$ from the position vectors and internal states of all the child nodes of the given non-leaf node according to a covariant aggregation rule that represents $\psi_j$ as a tensor object that is covariant to rotations of the rotation group SO(3); applying a Clebsch-Gordan transform to reduce a tensor product of the state vectors of the nodes to irreducible covariant vectors; and computing $\psi_j$ of the root node as output of the ANN to determine the internal state of the N–body physical system.

**[0012]** In still another respect, example embodiments may involve an article of manufacture comprising a non-transitory computer readable media having computer-readable instructions stored thereon for computationally simulating an N–body physical system, wherein the N–body physical system is represented mathematically as a compound object X having N elementary parts $E = \{e_i\}$, $i = 1,\ldots,N$, each $e_i$ representing one of the N bodies of the N–body physical system, wherein X is hierarchically decomposed into J subsystems, $P_j$, $j = 1,..,J$, each $P_j$ comprising one or more of the elementary parts of E, and wherein each $P_j$ is described by a position vector $\mathbf{r}_j$ and an internal state vector $\psi_j$, and wherein the instructions, when executed by one or more processors of a computing device, cause the computing device to carry out operations including: constructing a hierarchical artificial neural network (ANN) having J nodes

each corresponding to one of the J subsystems, the J nodes including $m \geq 2$ leaf nodes as ANN inputs, $m = 1$ non-leaf root node as ANN output, and $m \geq 1$ intermediate non-leaf nodes, wherein each node is a neuron of the ANN and is configured to compute an activation corresponding to a different one of the internal state vectors $\psi_j$, and wherein: for each leaf node, $\psi_j$ describes the internal state of a respective one of the $P_j$ subsystems having just a single elementary part $e_i$, for each given intermediate non-leaf node, $\psi_j$ describes the internal state of a respective one of the $P_j$ subsystems having $2 \leq k < N$ parts $e_i$ that are each comprised in a child node of the given intermediate non-leaf node, and for the root node, $\psi_j$ describes the internal state of a subsystem $P_j$ having $k = N$ elementary parts $e_i$ that are each comprised in a child node of the root node; receiving input data to the leaf nodes specifying respective position vectors and respective internal state vectors of the N elementary parts E; for each given non-leaf node, computing $\psi_j$ from the position vectors and internal states of all the child nodes of the given non-leaf node according to a covariant aggregation rule that represents $\psi_j$ as a tensor object that is covariant to rotations of the rotation group SO(3); applying a Clebsch-Gordan transform to reduce a tensor product of the state vectors of the nodes to irreducible covariant vectors; and computing $\psi_j$ of the root node as output of the ANN to determine the internal state of the N–body physical system.

**[0013]** These as well as other embodiments, aspects, advantages, and alternatives will become apparent to those of ordinary skill in the art by reading the following detailed description, with reference where appropriate to the accompanying drawings. Further, this summary and other descriptions and figures provided herein are intended to illustrate embodiments by way of example only and, as such, that numerous variations are possible. For instance, structural elements and process steps can be rearranged, combined, distributed,

eliminated, or otherwise changed, while remaining within the scope of the embodiments as claimed.

## Brief Description of Drawings

[0014] Figure 1 depicts a simplified block diagram of an example computing device, in accordance with example embodiments.

[0015] Figure 2 is a conceptual illustration of two types of tree-like artificial neural network, one strict tree-like and the other non-strict tree-like, in accordance with example embodiments.

[0016] Figure 3A is a conceptual illustration of an N–body system, in accordance with example embodiments.

[0017] Figure 3B is a conceptual illustration of an N–body system showing a second level of substructure, in accordance with example embodiments.

[0018] Figure 3C is a conceptual illustration of an N–body system showing a third level of substructure, in accordance with example embodiments.

[0019] Figure 3D is a conceptual illustration of an N–body system showing a fourth level of substructure, in accordance with example embodiments.

[0020] Figure 3E is a conceptual illustration of an N–body system showing a fifth level of substructure, in accordance with example embodiments.

[0021] Figure 3F is a conceptual illustration of a decomposition of an N–body system in terms of subsystems and internal states, in accordance with example embodiments.

[0022] Figure 4A is a conceptual illustration of compositional scheme for a compound object representing an N–body system, in accordance with example embodiments.

[0023] Figure 4B is a conceptual illustration of compositional neural network for simulating an N–body system, in accordance with example embodiments.

[0024] Figure 5 is a flow chart of an example method, in accordance with example embodiments.

## Detailed Description

[0025] Example methods, devices, and systems are described herein. It should be understood that the words "example" and "exemplary" are used herein to mean "serving as an example, instance, or illustration." Any embodiment or feature described herein as being an "example" or "exemplary" is not necessarily to be construed as preferred or advantageous over other embodiments or features unless stated as such. Thus, other embodiments can be utilized and other changes can be made without departing from the scope of the subject matter presented herein.

[0026] Accordingly, the example embodiments described herein are not meant to be limiting. It will be readily understood that the aspects of the present disclosure, as generally described herein, and illustrated in the figures, can be arranged, substituted, combined, separated, and designed in a wide variety of different configurations. For example, the separation of features into "client" and "server" components may occur in a number of ways.

[0027] Further, unless context suggests otherwise, the features illustrated in each of the figures may be used in combination with one another. Thus, the figures should be generally viewed as component aspects of one or more overall embodiments, with the understanding that not all illustrated features are necessary for each embodiment.

[0028] Additionally, any enumeration of elements, blocks, or steps in this specification or the claims is for purposes of clarity. Thus, such enumeration should not be interpreted to require or imply that these elements, blocks, or steps adhere to a particular arrangement or are carried out in a particular order.

### I. Introduction

[0029] Example embodiments of a covariant hierarchical neural network architecture,

referred to herein as "N–body comp-nets," are described herein in terms of molecular structure, and in particular, atomic potentials of molecular systems. The example of such molecular systems provides a convenient basis for connecting analytic concepts of N–body comp-nets to physical systems that may be illustratively conceptualized. For example, a physical hierarchy of structures and substructures of molecular constituents (e.g., atoms) may lend itself to a descriptive visualization. Similarly, the concept of rotational and/or translational invariance (or, more generally, invariance to spatial transformations) may be easily grasped at a conceptual level in terms of the ability of a neural network to learn to recognized complex systems regardless of their spatial orientations when presented to the neural network. And consideration of learning atomic and/or molecular potentials of such systems can help tie the structure of the constituents to their physics in an intuitive manner. However, the example of molecular/atomic systems and potentials is not, and should not, be viewed as limiting with respect to either the analytical framework or the applicability of N–body comp-nets.

**[0030]** More specifically, the challenges described above – namely the ability to recognize multiscale structure while maintaining invariance with respect to spatial transformation – may be met by the inventor's novel application of concepts of group representation theory to neural networks. The inventor's introduction of Clebsch–Gordan decompositions into hierarchically structured neural networks is one aspect of example embodiments described herein that makes N–body comp-nets broadly applicable to problems beyond the example of molecular/atomic systems and potentials. In particular, it supplies an analytical prescription for how neural networks may be constructed and/or adapted to simulate a wide range of physical systems, as well as address problems in areas such as computer vision, and computer graphics (and, more generally, point-cloud representations), among others.

[0031] In relation to physical systems described by way of example herein, neurons of an example N–body comp-net may be described as representing internal states of subsystems of a physical system being modeled. This too, however, is a convenient illustration that may be conceptually connected to the physics of molecular and/or atomic systems. Thus, in accordance with example embodiments, internal state may be a convenient computational representation of the activations of neurons of a comp-net. In other applications, the activations may be associated with other physical properties or analytical characteristics of the problem at hand. In either case (and in others), a common aspect of activations of a comp-net is the transformational properties provided by tensor representation and the Clebsch–Gordan decompositions it admits. These are aspects that enable neural networks to meet challenges that have previously vexed their operation. Practical applications of simulations of N–body comp-nets are extensive.

[0032] In relation to molecular structure and dynamics, N–body comp-nets may be used to learn, compute, and/or predict (in addition to potential energies) forces, metastable states, and transition probabilities. Applied or integrated in a context of larger structure, N–body comp-nets may be extended to areas of material design, such as tensile strength, design of new drug compounds, simulation of protein folding, design of new battery technologies and new types of photovoltaics. Other areas of applicability of N–body comp-nets may include prediction of protein-ligand interactions, protein-protein interactions, and properties of small molecules, including solubility and lipophilicity. Additional applications may also include protein structure prediction and structure refinement, protein design, DNA interactions, drug interactions, protein interactions, nucleic acid interactions, protein-lipid-nucleic acid interactions, molecule/ligand interactions, drug permeability measurements, and predicting protein folding and unfolding. As this list of examples suggests, N–body comp-nets may provide a basis for wide applicability,

both in terms of the classes and/or types of specific problems tackled, and the conceptual variety of problems they can address.

## II. Example Computing Devices

**[0033]** Figure 1 is a simplified block diagram of a computing device 100, in accordance with example embodiments. As shown, the computing device 100 may include processor(s) 102, memory 104, network interface(s) 106, and an input/output unit 108. By way of example, the components are communicatively connected by a bus 110. The bus could also provide power from a power supply (not shown). In particular, computing device 100 may be configured to perform at least one function of and/or related to implementing all or portions of artificial neural networks 200, 202, and/or 400-B, machine learning system 700, and/or method 500, all of which are described below.

**[0034]** Memory 104 may include firmware, a kernel, and applications, among other forms and functions of memory. As described, the memory 104 may store machine-language instructions, such as programming code or non-transitory computer-readable storage media, that may be executed by the processor 102 in order to carry out operations that implement the methods, scenarios, and techniques as described herein and in accompanying documents and/or at least part of the functionality of the example devices, networks, and systems described herein. In some examples, memory 104 may be implemented using a single physical device (e.g., one magnetic or disc storage unit), while in other examples, memory 104 may be implemented using two or more physical devices. In some examples, memory 104 may include storage for one or more machine learning systems and/or one or more machine learning models as described herein.

**[0035]** Processors 102 may include one or more general purpose processors and/or one or more special purpose processors (e.g., digital signal processors (DSPs) or graphics processing

units (GPUs). Processors 102 may be configured to execute computer-readable instructions that are contained in memory 104 and/or other instructions as described herein.

**[0036]** Network interface(s) 106 may provide network connectivity to the computing system 100, such as to the internet or other public and/or private networks. Networks may be used to connect the computing system 100 with one or more other computing devices, such as servers or other computing systems. In an example embodiment, multiple computing systems could be communicatively connected, and example methods could be implemented in a distributed fashion.

**[0037]** Client device 112 may be a user client or terminal that includes an interactive display, such as a GUI. Client device 112 may be used for user access to programs, applications, and data of the computing device 100. For example, a GUI could be used for graphical interaction with programs and applications described herein. In some configurations, the client device 112 may itself be a computing device; in other configurations, the computing device 100 may incorporate, or be configured to operate as, a client device.

**[0038]** Database 114 may include input data, such as images, configurations of N–body systems, or other data used in the techniques described herein. Data could be acquired for processing and/or recognition by a neural network, including artificial neural networks 200, 202, and/or 400-B. The data could additionally or alternatively be training data, which may be input to a neural network, for training, such as determination of weighting factors applied at various layers of the neural network. Database 114 could be used for other purposes as well.

## III. Example Artificial Neural Networks for Representing Structured Objects

**[0039]** Example embodiments of N–body neural networks for simulation and modeling may be described in terms of some of the structures and features of "classical" feed-forward

neural networks. Accordingly, a brief review of classical feed-forward networks is presented below in order to provide a context for describing an example general purpose neural architecture for representing structured objects referred to herein as "compositional networks."

**[0040]** A prototypical feed-forward neural network consists of some number of neurons $\{n_i^\ell\}$ arranged in $L$+1 distinct layers. Layer $\ell$=0 is referred to as the "input layer," and is where training and testing data enter the network, while the inputs of the neurons in layers $\ell = 1, 2, \ldots, L$ are the outputs $\{f_j^{\ell-1}\}$ of the neurons in the previous layer. Each neuron computes its output, also called its "activation," using a simple rule such as

$$f_i^\ell = \sigma\left(\sum_j w_j^\ell f_j^{\ell-1} + b_\ell\right), \quad (1)$$

where the $\{w_j^\ell\}$ weights and $\{b_\ell\}$ biases are learnable parameters, while $\sigma$ is a fixed nonlinearity, such a sigmoid function or a ReLU operator. The output of the network appears in layer $L$, also referred to as the "output layer." As computational entities or constructs implemented as software or other machine language code executable on a computing device, such as computing device 100, neural networks are also commonly referred to as "artificial neural networks" or ANNs. The term ANN may also refer to a broader class of neural network architectures than feed-forward networks, and is used without loss of generality to refer to example embodiments of neural networks described herein.

**[0041]** During training of a feed-forward neural network, training data are input, and the output layer results are compared with the desired output by means of a loss function. The gradient of the loss may be back-propagated through the network to update the parameters, typically by some variant of stochastic gradient descent. During real-time or "live" operation, testing data, representing some object (e.g., a digital image) or system (e.g., a molecule) having an unknown *a priori* output result, are fed into the network. The result may represent a

prediction by the network of the correct output result to within some prescribed statistical uncertainty, for example. The accuracy of the prediction may depend on the appropriateness of the network configuration for solving the problem, as well as the amount and/or quality of the training.

**[0042]** The neurons and layers of feed-forward neural networks may be arranged in tree-like structures. Figure 2 is a conceptual illustration of two types of tree-like artificial neural network. In particular, ANN 200 depicts a feed-forward neural network having strict tree-like structure, while ANN 200 depicts a feed-forward neural network having non-strict tree-like. Both ANNs have an input layer 204 having four neurons f1, f2, f3, and f4, and an output layer 206 having a single neuron f11. Neurons are also referred to as "nodes" in describing their configuration and connections in a neural network. The four input neurons in the example are referred to as "leaf-nodes," and the single output neuron is referred to as a "root node."

**[0043]** In each ANN, neurons f5, f6, and f7 reside in a first "hidden layer" after the input layer 204, and neurons f8, f9, and f10 reside in a second hidden layer, which is also just before the output layer 206. The neurons in the hidden layers are also referred to as "hidden node" and/or "non-leaf nodes." Note that the root node is also a non-leaf node. In addition, there could be ANNs having more than two hidden layers, or even just one hidden layer.

**[0044]** Input data $IN_1$, $IN_2$, $IN_3$, and $IN_4$ are input to the input neurons of each layer, and a single output D_OUT is output from the output neuron of each ANN. Connections between neurons (directed arrows in Figure 2) correspond to activations fed forward from one neuron to the next. In particular, one or more nodes that provide input to a given node are referred to as "child nodes" of the given node, and the given node is referred to as the "parent node" of the child nodes. For the purposes of the discussion herein, strict tree-like ANNs, such as ANN 200,

are distinguished from non-strict tree-like ANNs, such as ANN 202, by the types of parent-child connections they each have.

**[0045]** More specifically, in a strict tree-like ANN, the each child node of a parent node resides in a layer immediately prior to the layer in which the parent node resides. Three examples are indicated in ANN 200. Namely, f4 which is a child of f7 resides in the layer immediately prior to the f7's layer. Similarly, f7 which is a child of f10 resides in the layer immediately prior to the f10's layer, and f10 which is a child of f11 resides in the layer immediately prior to the f11's layer. It may be seen by inspection that the same relationship holds for all the connected nodes of ANN 200.

**[0046]** In a non-strict tree-like ANN, the each child node of a parent node resides in a layer prior to the layer in which the parent node resides, but it need not be the immediately prior layer. Three examples are indicated in ANN 202. Namely, f1 which is a child of f8 resides two layers ahead of f8's layer. Similarly, f4 which is a child of f10 resides two layers ahead of f10's layer. However, and f5 which is a child of f8 resides in the layer immediately prior to the f8's layer. Thus, a non-strict tree-like ANN may include a mix of inter-layer relationships.

**[0047]** Feed-forward neural networks (especially "deep," i.e., ones with many layers) have been demonstrated to be quite successful in their predicative capabilities due in part to their ability to implicitly decompose complex objects into their constituent parts. This may be particularly the case for "convolutional" neural networks (CNNs), commonly used in computer vision. In CNNs, the weights in each layer are tied together, which tends to force the neurons to learn increasingly complex visual features, from simple edge detectors all the way to complex shapes such as human eyes, mouths, faces, and so on.

**[0048]** There has been recent interest in extending neural networks to learning from structured objects, such as graphs. A range of architectures have been proposed for this purpose, many of them based on various generalizations of the notion of convolution to these domains.

**[0049]** One particular architecture, which makes the part-based aspect of neural modeling very explicit, is that of "compositional networks" ("comp-nets"), introduced previously by the inventor. In accordance with example embodiments, comp-nets may represent a structured object $X$ in terms of a decomposition of $X$ into a hierarchy of parts, subparts, subsubparts, and so on, down to some number of elementary parts $\{e_l\}$. Referring to the parts, subparts, subsubparts, and so on, simply as "parts" or "subsystems" $P_i$, the decomposition may be considered as forming a so-called "composition scheme" of a collection of $P_i$ that make up the hierarchy.

**[0050]** Figures 3A-3F illustrate conceptually the decomposition of an N–body physical system, such as a molecule, into an example hierarchy of subsystems. Figure 3A first shows the example N–body physical system made up of constituent particles, such as atoms of a molecule. In the figure, and arrow point from a central particle and labeled **F** may represent the aggregate or total vector force on the central particle due to the physical interaction of the other particles. These might be electrostatic or other inter-atomic forces, for example.

**[0051]** By way of example, Figure 3B shows a first level of the subsystem hierarchy. In the illustration, particular groupings of the particles represent a first level of subsystem or subparts. As shown, there appears to be six grouping, corresponding to six subparts. Figure 3C show a next (second) level of the subsystem hierarchy, again by way of example. In this case, there are four groupings, corresponding to four subparts. Similarly, Figure 3D shows the third level of groupings, this one having three subsystems, and Figure 3E shows the top level of the hierarchy, having a single grouping that includes all of the lower level subsystems. In practice,

the decomposition may be determined according to known or expected properties of the physical system under consideration. It will be appreciated, however, that the conceptual illustrations of Figures 3A-3E do not necessarily convey any such physical considerations. Rather, they merely depict the decomposition concept for the purposes of the discussion herein.

**[0052]** Figure 3F illustrates how a decomposition scheme may be translated to a comp-net architecture. In accordance with example embodiments, each subsystem of a hierarchy may correspond to a node (or neuron) of an ANN in which successive layers represent successive layers of the compositional hierarchy. In accordance with at least some example embodiments, and as described in detail below, each subsystem may be described by a spatial position vector and an internal state vector. This is indicated by the labels $\mathbf{r}$ and $|\psi\rangle$ in each of the subsystems shown in Figure 3F. In the comp-net representation, the internal state vector of each subsystem may be computed as the activation of the corresponding neuron. The inputs to each non-leaf node may be the activation of one or more child nodes of one or more prior layers, each child node representing a subsystem of a lower level of the hierarchy.

**[0053]** Returning to consideration of the decomposition and the composition scheme, since each part $P_i$ can be a sub-part of more than one higher level part, the composition scheme is not necessarily a strict tree, but is rather a DAG (directed acyclic graph). An exact definition, in accordance with example embodiments, is as follows.

**[0054] Definition 1**. Let $X$ be a compound object with n elementary parts $\varepsilon = \{e_1, \ldots, e_n\}$. A "composition scheme" $D$ for $X$ is a directed acyclic graph ($DAG$) in which each node $n_i$ is associated with some subset $P_i$ of $\varepsilon$ (these subsets are called the parts of $X$) in such a way that

1. If $n_i$ is a leaf node, then $P_i$ contains a single elementary part $e_{\xi(i)}$.

2. $D$ has a unique root node $n_i$, which corresponds to the entire set $\{e_1, \ldots, e_n\}$.

3. For any two nodes $n_i$ and $n_j$, if $n_i$ is a descendant of $n_j$, then $P_i \subset P_j$.

**[0055]** In accordance with example embodiments, a comp-net is a composition scheme that may be reinterpreted as a feed-forward neural network. In particular, in a comp-net each neuron $n_i$ also has an activation $f_i$. For leaf nodes, $f_i$ may be some simple pre-defined vector representation of the corresponding elementary part $e_{\xi(i)}$. For internal nodes, $f_i$ may be computed from the activations $f_{\mathrm{ch}_1}, \ldots, f_{\mathrm{ch}_k}$ of the children of $n_i$ by the use of some aggregation function $\Phi\left(f_{\mathrm{ch}_1}, \ldots, f_{\mathrm{ch}_k}\right)$ similar to equation (1). Finally, the output of the comp-net is the output of the root node $n_r$.

**[0056]** Figures 4A and 4B further illustrate by way of example a translation from a hierarchical composition scheme of a compound object to a corresponding compositional neural network (comp-net). Specifically, Figure 4A depicts a composition scheme 400-A in which leaf-nodes $n_1$, $n_2$, $n_3$, and $n_4$ of the first (lowest) level of the hierarchy correspond to single-element subsystems $\{e_1\}$, $\{e_2\}$, $\{e_3\}$, and $\{e_4\}$, respectively.

**[0057]** At the next (second) level up in the hierarchy, non-leaf nodes $n_5$, $n_6$, and $n_7$ each contain two-element subsystems, each subsystem being "built" from a respective combination of two first-level subsystems. For example, as shown, $n_5$ contains $\{e_3, e_4\}$ from nodes $n_3$ and $n_4$, respectively. The arrows pointing from $n_3$ and $n_4$ to $n_5$ indicate this relationship.

**[0058]** At the third level up, non-leaf nodes $n_8$, $n_9$, and $n_{10}$ each contain three-element subsystems, each subsystem being built from a respective combination of subsystems from the previous levels. For example, as shown, $n_{10}$ contains $\{e_1, e_4\}$ from the two-element subsystem at $n_6$, and $\{e_2\}$ from the single-element subsystem at $n_2$. The arrows pointing from $n_6$ and $n_2$ to $n_{10}$ indicate this relationship.

**[0059]** Finally, at the top level, the (non-leaf) root node $n_r$ contains all four elementary parts in subsystem $\{e_1, e_2, e_3, e_4\}$ from the previous level. Note that subsystems at a given level above the lowest ("leaf") level may overlap in terms of common (shared) elementary parts and/or common (shared) lower-level subsystems. It may also be seen by inspection that the example composition scheme 400-A corresponds to a non-strict tree-like structure.

**[0060]** Figure 4B illustrates an example comp-net 400-B that corresponds to the composition scheme 400-A of Figure 4A. In this illustration, the neurons $\{n_1\}$, $\{n_2\}$, ..., $\{n_r\}$ of comp-net 400-B correspond, respectively, to the nodes $n_1$, $n_2$, ..., $n_r$ of the composition scheme 400-A, and the arrow connecting the neurons correspond to the relationships between the nodes in the composition scheme. The neuron are also associated with respective activations $f_1$, $f_2$, ..., $f_r$, as shown. Thus, as illustrated in this example, the activations $f_3$ and $f_4$ are inputs to neuron (node) $\{n_5\}$, which uses them in computing its activation $f_5$.

**[0061]** The inventor has previously detailed the behavior of comp-nets under transformations of $X$, in particular, how to ensure that the output of the network is invariant with respect to spurious permutations of the elementary parts, whilst retaining as much information about the combinatorial structure of $X$ as possible. This is significant in graph learning, where $X$ is a graph, $e_1, \ldots, e_n$ are its vertices, and $\{P_i\}$ are subgraphs of different radii. The proposed solution, "covariant compositional networks" (CCNs), involves turning the $\{f_i\}$ activations into tensors that transform in prescribed ways with respect to permutations of the elementary parts making up each $P_i$.

**[0062]** Referring again to Figure 3F, the activations of the nodes of a comp-net may describe states of the subsystems corresponding to the nodes. In a representation of a physical N–body system, the computation of the state of a given node may characterize physical

interactions between the constituent subsystems of the given node. In accordance with example embodiments, and as described in detail below, the activations are constructed to ensure that the states are tensorial objects with spatial character, and in particular that they are covariant with rotations in the sense that they transform under rotations according to specific irreducible representations of the rotation group.

## IV. Analytical Description of, and Theoretical Bases for, Covariant Comp-Nets

### [0063] A. Compositional Models for Atomic Environments

**[0064]** Decomposing complex systems into a hierarchy of interacting subsystems at different scales is a recurring theme in physics, from coarse graining approaches to renormalization group theory. The same approach applied to the atomic neighborhood lends itself naturally to learning force fields. For example, to calculate the aggregate force on the central atom, in a first approximation one might just sum up independent contributions from each of its neighbors. In a second approximation, one would also consider the modifying effect of the local neighborhoods of the neighbors. A third order approximation would involve considering the neighborhoods of the atoms in these neighborhoods, and so on.

**[0065]** The inventor has recognized that the compositional networks formalism is thus a natural framework for force field learning. In particular, comp-nets may be considered in which the elementary parts correspond to actual physical atoms, the internal nodes correspond to subsystems $P_i$ made up of multiple atoms. In accordance with example embodiments, the corresponding activation, now denoted $\psi_i$, and referred to herein as the *state* of $P_i$, may effectively be considered a learned coarse grained representation of $P_i$. What makes physical problems different from, such as learning graphs, however, is their spatial character. In particular:

1. Each subsystem $P_i$ may now also be associated with a vector $\boldsymbol{r}_i \in \mathbb{R}^3$ specifying its spatial position.
2. The interaction between two subsystems $P_i$ and $P_j$ depends not only on their relative positions, but also on their relative orientations. Therefore, $\psi_i$ and $\psi_j$ must also have spatial character, somewhat similarly to the terms of the monopole, dipole, quadrupole, etc. expansions, for example.

If the entire the atomic environment is rotated around the central atom by some rotation $R \in \mathrm{SO}(3)^3$, the position vectors transform as $r_i \longmapsto Rr_i$. Mathematically, the second point above says that the $\psi_i$ activations (states) must also transform under rotations in a predictable way, which is expressed by saying that they must be rotationally *covariant*.

**[0066] Group Representations And *N*–Body Networks**

**[0067]** Just as covariance to permutations is a critical constraint on the graph CCNs, covariance to rotations is the guiding principle behind CCNs for learning atomic force fields. To describe this concept in its general form, a starting assumption may be taken to be that any given activation $\psi$ is representable as a $d$ dimensional (complex valued) vector, and that the transformation that $\psi$ undergoes under a rotation $R$ is linear, i.e., $\psi \longmapsto \rho(R)\psi$ for some matrix $\rho(R)$.

**[0068]** The linearity assumption is sufficient to guarantee that for $R, R' \in \mathrm{SO}(3), \rho(R)\rho(R') = \rho(RR')$. Complex matrix valued functions satisfying this criterion are called *representations* of the group SO(3). Standard theorems in representation theory indicate that any compact group $G$ (such as $SO(3)$) has a sequence of so-called inequivalent irreducible representations $\rho_0,\rho_1,\ldots$ ("irreps," for short), and that any other representation $\mu$ of $G$

can be reduced into a direct sum of irreps in the sense that there is some invertible matrix $C$ and sequence of integers $\tau_0,\tau_1,...$ such that

$$\mu(R) = C^{-1}\left[\oplus_\ell \oplus_{m=1}^{\tau_\ell} \rho_\ell(R)\right] C. \tag{2}$$

**[0069]** Here $\tau_\ell$ is called the *multiplicity* of $\rho_\ell$ in $\mu$, and $\boldsymbol{\tau} = (\tau_0,\tau_1,...)$ is called the *type* of $\mu$. Another feature of the representation theory of compact groups is that the irreps can always be chosen to be unitary, i.e., $\rho(R^{-1}) = \rho(R)^{-1} = \rho(R)^\dagger$, where $M^\dagger$ denotes the Hermitian conjugate (conjugate transpose) of the matrix $M$. In the following it may be assumed that irreps satisfy this condition. If $\mu$ is also unitary, then the transformation matrix $C$ will be unitary too, so $C^{-1}$ may be replaced with $C^\dagger$.

**[0070]** In the specific case of the rotation group SO(3), the irreps are sometimes called Wigner D–matrices. The $\ell = 0$ irrep consists of the one dimensional constant matrices $\rho_0(R) = (1)$, the $\ell = 0$ irrep (up to conjugation) is equivalent to the rotation matrices themselves, while for general $\ell$, assuming that $(\theta, \phi, \psi)$ are the Euler angles of $R$, $[\rho_\ell(R)]_{m,m'} = e^{i\psi m'} Y_m^\ell(\theta, \phi)$, where $\{Y_m^\ell\}$ are the well known spherical harmonic functions. In general, the dimensionality of $\rho_\ell$ is $2\ell + 1$, i.e., $\rho_\ell(R) \in \mathbb{C}^{(2\ell+1)\times(2\ell+1)}$.

**[0071] Definition 2.** $\psi \in \mathbb{C}^d$ is said to be an SO(3)–covariant vector of type $\boldsymbol{\tau} = (\tau_0,\tau_1,\tau_2,...)$ if under the action of rotations it transforms as

$$\psi \longmapsto \left[\oplus_\ell \oplus_{m=1}^{\tau_\ell} \rho_\ell(R)\right]\psi. \tag{3}$$

Setting

$$\psi = \oplus_\ell \oplus_{m=1}^{\tau_\ell} \psi_m^\ell, \tag{4}$$

$\psi_m^\ell \in \mathbb{C}^{2\ell+1}$ may be called the (l, m)-fragment of $\psi$, and

$$\psi^\ell = \oplus_{m=1}^{\tau_\ell} \psi_m^\ell$$

may be called the $\ell$'th part of $\psi$. A covariant vector of type $\boldsymbol{\tau} = (0,0,...,0,1)$, where the single 1 corresponds to $\tau_k$, may be called an irreducible vector of order k or an irreducible $\rho_k$–vector. Note that a first order irreducible vector is just a scalar.

**[0072]** A benefit of the above definition is that each fragment $\psi_m^\ell$ transforms in the very simple way $\psi_m^\ell \longmapsto \rho_\ell(R)\psi_m^\ell$. Note that the terms "fragment" and "part" are not necessarily standard in the literature, but are used here for being useful in describing covariant neural architectures. Also note that unlike equation (2), there is no matrix $C$ in equations (3) and (4). This is because if a given vector $\psi$ transforms according to a general representation $\mu$ whose decomposition does include a nontrivial $C$, this matrix may be easily be factored out by redefining $\psi$ as $C\psi$. Here $\psi^\ell$ is sometimes also called the projection of $\psi$ to the $\ell$'th *isotypic subspace* of the representation space that $\psi$ lives in, and $\psi = \psi^0 \oplus \psi^1 \oplus ...$ is called the *isotypic decomposition* of $\psi$. With these representation theoretic tools in hand, the concept of SO(3)–covariant $N$–body neural networks may be defined as follows.

**[0073] Definition 3**. Let S be a physical system made up of n particles $\xi_1,...,\xi_n$. An SO(3)–covariant N–body neural network $N$ for $S$ is a composition scheme D in which

1. Each node $n_j$, which may also be referred to as a "gate," is associated with
   (a) a physical subsystem $P_j$ of $S$;
   (b) a vector $\boldsymbol{r}_j \in \mathbb{R}^3$ describing the spatial position of $P_j$;
   (c) a vector $\psi_j$ that that describes the internal state of $Pj$ and is type $\boldsymbol{\tau}_j$ covariant to rotations.
2. If $n_j$ is a leaf node, then $\psi_j$ is determined by the corresponding particle $\xi_j$.
3. If $n_j$ is a non-leaf node and its children are $n_{ch_1}, \ldots, n_{ch_k}$, then $\psi_j$ is computed as

$$\psi_j = \Phi_j\left(\hat{r}_{ch_1}, \ldots, \hat{r}_{ch_k}, \hat{r}_{ch_1}, \ldots, \hat{r}_{ch_k}, \psi_{ch_1} \ldots, \psi_{ch_k}\right), \tag{5}$$

where $\hat{r}_{ch_i} = r_{ch_i} - r_j$ and $\hat{r}_i = |\hat{r}_i|$. In the discussion herein, $\Phi_j$ is referred to as the local "aggregation rule."

4. D has a unique root $n_r$, and the output of the network, i.e., the learned state of the entire system is $\psi_r$. In the case of learning scalar valued functions, such as the atomic potential, $\psi_r$ is just a scalar.

**[0074]** In accordance with example embodiments, Definition 3 may be considered as defining a general architecture for learning the state of $N$– body physical systems with much wider applicability than just learning atomic potentials. Also in accordance with example embodiments the $\Phi_j$ aggregation rules may be defined in such a way as to guarantee that each $\psi_j$ is SO(3)–covariant. This is what is addressed in the following section.

**[0075] B. Covariant Aggregation Rules**

**[0076]** To define the aggregation function $\Phi$ to be used in SO(3)–covariant comp-nets, it may only be assumed that $\Phi$ is a polynomial in the relative positions $\hat{\boldsymbol{r}}_{ch_1}, \ldots, \hat{\boldsymbol{r}}_{ch_k}$, the constituent state vectors $\psi_{\mathrm{ch}_1} \ldots, \psi_{\mathrm{ch}_k}$ and the inverse distances $1/\hat{r}_{\mathrm{ch}_1}, \ldots, 1/\hat{r}_{\mathrm{ch}_k}$. Specifically, it may be said that $\Phi$ is a $(P,Q,S)$–order aggregation function if each component of $\psi = \Phi_j\left(\hat{r}_{\mathrm{ch}_1}, \ldots, \hat{r}_{\mathrm{ch}_k}, \hat{\boldsymbol{r}}_{\mathrm{ch}_1}, \ldots, \hat{\boldsymbol{r}}_{\mathrm{ch}_k}, \psi_{\mathrm{ch}_1} \ldots, \psi_{\mathrm{ch}_k}\right)$ is a polynomial of order at most $p$ in each component of $\boldsymbol{r}_{\mathrm{ch}_i}$, a polynomial of at most $q$ in each component of $\psi_{\mathrm{ch}_i}$, and a polynomial of order at most $s$ in each $1/\left(\hat{r}_{\mathrm{ch}_i}\right)$. Any such $\Phi$ can be expressed as

$$\Phi(\ldots) = \mathcal{L}\left(\oplus_{\mathrm{p,q,s}} \boldsymbol{r}_{\mathrm{ch}_1}^{\otimes_{p_1}} \otimes \ldots \otimes \boldsymbol{r}_{\mathrm{ch}_k}^{\otimes_{p_k}} \otimes \psi_{\mathrm{ch}_1}^{\otimes_{q_1}} \otimes \ldots \otimes \psi_{\mathrm{ch}_k}^{\otimes_{q_k}} \cdot \hat{r}_{\mathrm{ch}_1}^{-s_1} \cdot \ldots \cdot \hat{r}_{\mathrm{ch}_k}^{-s_k}\right), \qquad (6)$$

where $\mathbf{p}$, $\mathbf{q}$ and $\mathbf{s}$ are multi–indices of positive integers with $p_i \leq P, q_i \leq Q$ *and* $s_i \leq S$, and $\mathcal{L}$ is a linear function. The tensor products appearing in equation (6) are formidably large object and in most cases may be impractical to compute explicitly. Accordingly, this equation is meant to emphasize that any learnable parameters of the network must be implicit in the linear operator $\mathcal{L}$.

[0077] The more stringent requirements on $\mathcal{L}$ arise from the covariance criterion. The inventor has recognized that understanding these may be aided by the observation that for any sequence $\rho_1, \ldots, \rho_p$ of (not necessarily irreducible) representations of a compact group $G$, their tensor product

$$\rho(R) = \rho_1(R) \otimes \rho_2(R) \otimes \ldots \otimes \rho_p(R)$$

is also a representation of $G$. Consequently, $\rho$ has a decomposition into irreps, similar to equation (2). As an immediate corollary, any product of SO(3) covariant vectors can be similarly decomposed. In particular, by applying the appropriate unitary matrix $C$, the sum of tensor products appearing in equation (6) can be decomposed into a sum of irreducible fragments in the form

$$\oplus_{\ell=0}^{L} \oplus_{m=1}^{\tau'_\ell} \phi_m^\ell = C\left(\oplus_{\mathrm{p,q,s}} \boldsymbol{r}_{\mathrm{ch}_1}^{\otimes p_1} \otimes \ldots \otimes \boldsymbol{r}_{\mathrm{ch}_k}^{\otimes p_k} \otimes \psi_{\mathrm{ch}_1}^{\otimes q_1} \otimes \ldots \otimes \psi_{\mathrm{ch}_k}^{\otimes q_k} \cdot \hat{r}_{\mathrm{ch}_1}^{-s_1} \cdot \ldots \cdot \hat{r}_{\mathrm{ch}_k}^{-s_k}\right).$$

More explicitly,

$$\phi_m^\ell = T_m^\ell\left(\oplus_{\mathrm{p,q,s}} \boldsymbol{r}_{\mathrm{ch}_1}^{\otimes p_1} \otimes \ldots \otimes \boldsymbol{r}_{\mathrm{ch}_k}^{\otimes p_k} \otimes \psi_{\mathrm{ch}_1}^{\otimes q_1} \otimes \ldots \otimes \psi_{\mathrm{ch}_k}^{\otimes q_k} \cdot \hat{r}_{\mathrm{ch}_1}^{-s_1} \cdot \ldots \cdot \hat{r}_{\mathrm{ch}_k}^{-s_k}\right), \qquad (7)$$

where $T_1^0, \ldots, T_{\tau_0}^0, T_1^1, \ldots, T_{\tau_2}^0, \ldots, T_{\tau_L}^L$ is an appropriate sequence of projection operators. In accordance with example embodiments, the following proposition may provide a foundational result.

[0078] **Proposition 1**. The output of the aggregation function of equation (6) is a $\boldsymbol{\tau}$-covariant vector if and only if $\mathcal{L}$ is of the form

$$\mathcal{L}(\ldots) = \oplus_{\ell=0}^{L} \oplus_{m=1}^{\tau_\ell} \sum_{m'=1}^{\tau'_\ell} w_{m',m}^\ell \phi_{m'}^\ell. \qquad (8)$$

[0079] Equivalently, collecting all $\phi_{m'}^\ell$ fragments with the same $\ell$ into a matrix $\tilde{F}^\ell \in \mathbb{C}^{(2\ell+1)\times\tau'_\ell}$, all $\left(w_{m',m}^\ell\right)_{m',m}$ weights into a matrix $W^\ell \in \mathbb{C}^{\tau'_\ell \times \tau_\ell}$, and reinterpreting the output of $\mathcal{L}$ as a collection of matrices rather than a single long vector, equation (8) may be expressed as

$$\mathcal{L}(\ldots) = \left(\tilde{F}^0 W^0, \tilde{F}^1 W^1, \ldots, \tilde{F}^L W^L\right). \quad (9)$$

**[0080]** Proposition 1 indicates that $\mathcal{L}$ is only allowed to mix $\phi_m^\ell$ fragments with the same $\ell$, and that fragments can only be mixed in their entirety, rather than picking out their individual components. These are fundamental consequences of equivariance. However, there are no further restrictions on the $\left(W^\ell\right)_\ell$ mixing matrices.

**[0081]** In accordance with example embodiments, in an N–body neural network, the $W^\ell$ matrices are shared across (some subsets of) nodes, and it is these mixing (weight) matrices that the network learns from training data. The $\tilde{F}^\ell$ matrices can be regarded as generalized matrix valued activations. Since each $W^\ell$ interacts with the $F^\ell$ matrices linearly, the network can be trained the usual way by backpropagating gradients of whatever loss function is applied to the output node $n_r$, whose activation may typically be scalar valued.

**[0082]** It may be noted that N–body neural networks have no additional nonlinearity outside of $\Phi$, since that would break covariance. In contrast, in most existing neural network architectures, as explained above, each neuron first takes a linear combination of its inputs weighted by learned weights and then applies a fixed pointwise nonlinearity, $\sigma$. In accordance with architecture of N–body neural networks as described by way of example herein, the nonlinearity is hidden in the way that the $\phi_m^\ell$ fragments are computed, since a tensor product is a nonlinear function of its factors. On the other hand, mixing the resulting fragments with the $W^\ell$ weight matrices is a linear operation. Thus, in N–body neural networks as describe herein, the nonlinear part of the operation *precedes* the linear part.

**[0083]** The generic polynomial aggregation function of equation (6) may be too general to be used in a practical N–body network, and may be too costly computationally. Instead, in

accordance with example embodiments, a few specific types of low order gates may be used, such as those described below.

**[0084]** *Zeroth order interaction gates*

**[0085]** Zeroth order interaction gates aggregate the states of their children and combine them with their relative position vectors, but do not capture interactions between the children. A simple example of such a gate would be one where

$$\Phi(\ldots) = \mathcal{L}\left(\sum_{i=1}^{k}\left(\psi_{\mathrm{ch}_i} \otimes \hat{r}_{\mathrm{ch}_i}\right), \sum_{i=1}^{k} \hat{r}_{\mathrm{ch}_i}^{-1}\left(\psi_{\mathrm{ch}_i} \otimes \hat{r}_{\mathrm{ch}_i}\right), \sum_{i=1}^{k} \hat{r}_{\mathrm{ch}_i}^{-2}\left(\psi_{\mathrm{ch}_i} \otimes \hat{r}_{\mathrm{ch}_i}\right)\right). \quad (10)$$

**[0086]** Note that the summations in these formulae ensure that the output is invariant with respect to permuting the children and also reduce the generality of equation (6) because the direct sum is replaced by an explicit summation (this can also be interpreted as tying some of the mixing weights together in a particular way). Let $L$ be the largest $\ell$ for which $\tau_\ell \neq 0$ in the inputs. In the $L = 0$ case each $\psi_{\mathrm{ch}_i}$ state is a scalar quantity, such as electric charge. In the $L = 1$ case it is a vector, such as the dipole moment. In the $L = 2$ case it can encode the quadropole moment, and so on. A gate of the above form can learn how to combine such moments into a single (higher order) moment corresponding to the parent system.

**[0087]** It may be instructive to see how many parameters a gate of this type has. For this purpose, the simple case that each $\psi_{\mathrm{ch}_i}$ is of type $\boldsymbol{\tau} = (1,1,\ldots,1)$ (up to $\ell = L$) may be assumed. The type of $\hat{r}_{\mathrm{ch}_i}$ is (0,1). According to the Clebsch–Gordan rules, as described in more detail below, the product of two such vectors is a vector of type (1,3,2,...,2,1) (of length L+1). It may be further assume that desired output type is again $\boldsymbol{\tau} = (1,1,\ldots,1)$ of length L. This means that the $\ell = L + 1$ fragment does not even have to be computed, and the size of the weight matrices appearing in equation (9) are

$$W_0 \in \mathbb{C}^{1\times 3} \quad W_1 \in \mathbb{C}^{1\times 9} \quad W_2 \in \mathbb{C}^{1\times 6} \quad \ldots \quad W_L \in \mathbb{C}^{1\times 6}.$$

**[0088]** The size of these matrices changes dramatically as more "channels" are allowed. For example, if each of the input states are of type $\tau = (c, c, ..., c)$, the type of $\psi_{\mathrm{ch}_i} \otimes \hat{\boldsymbol{r}}_{\mathrm{ch}_i}$ becomes $(c, 3c, 2c, ..., 2c, 1c)$. Assuming again an output of type $\tau = (c, c, ..., c)$, the weight matrices become

$$W_0 \in \mathbb{C}^{c\times 3c} \quad W_1 \in \mathbb{C}^{c\times 9c} \quad W_2 \in \mathbb{C}^{c\times 6c} \quad ... \quad W_L \in \mathbb{C}^{c\times 6c}.$$

**[0089]** In many networks, however, the number of channels increases with height in the network. Allowing the output type to be as rich as possible, without inducing linear redundancies, the output type becomes $(3c, 9c, 6c, ..., 6c, 3c)$, and

$$W_0 \in \mathbb{C}^{3c\times 3c} \quad W_1 \in \mathbb{C}^{9c\times 9c} \quad W_2 \in \mathbb{C}^{9c\times 6c} \quad ... \quad W_L \in \mathbb{C}^{6c\times 6c}.$$

**[0090]** *First order interaction gates*

**[0091]** In first order interaction, gates each of the children interact with each other, and the parent aggregates these pairwise interactions. A simple example would be computing the total energy of a collection of charged bodies, which might be done with a gate of the form

$$\Phi(...) = \mathcal{L}\left(\sum_{i,j=1}^{k}\left(\psi_{ch_i}\otimes\psi_{ch_j}\otimes\hat{\boldsymbol{r}}_{ch_i}\otimes\hat{\boldsymbol{r}}_{ch_j}\right), \sum_{i,j=1}^{k}\hat{r}_{ch_i}^{-1}\hat{r}_{ch_j}^{-1}\left(\psi_{ch_i}\otimes\psi_{ch_j}\otimes\hat{\boldsymbol{r}}_{ch_i}\otimes\hat{\boldsymbol{r}}_{ch_j}\right),\right.$$
$$\left.\sum_{i,j=1}^{k}\hat{r}_{ch_i}^{-2}\hat{r}_{ch_j}^{-2}\left(\psi_{ch_i}\otimes\psi_{ch_j}\otimes\hat{\boldsymbol{r}}_{ch_i}\otimes\hat{\boldsymbol{r}}_{ch_j}\right), \sum_{i,j=1}^{k}\hat{r}_{ch_i}^{-3}\hat{r}_{ch_j}^{-3}\left(\psi_{ch_i}\otimes\psi_{ch_j}\otimes\hat{\boldsymbol{r}}_{ch_i}\otimes\hat{\boldsymbol{r}}_{ch_j}\right)\right). \quad (11)$$

**[0092]** Generalizing equation (6) slightly, if the interaction only depends on the relative positions of the child systems, another form that may be used is

$$\Phi(...) = \mathcal{L}\left(\sum_{i,j=1}^{k}\left(\psi_{ch_i}\otimes\psi_{ch_j}\otimes\hat{\boldsymbol{r}}_{ch_i,ch_j}\right), \sum_{i,j=1}^{k}\hat{r}_{ch_i,ch_j}^{-1}\left(\psi_{ch_i}\otimes\psi_{ch_j}\otimes\hat{\boldsymbol{r}}_{ch_i,ch_j}\right),\right.$$
$$\left.\sum_{i,j=1}^{k}\hat{r}_{ch_i,ch_j}^{-2}\left(\psi_{ch_i}\otimes\psi_{ch_j}\otimes\hat{\boldsymbol{r}}_{ch_i,ch_j}\right), \sum_{i,j=1}^{k}\hat{r}_{ch_i,ch_j}^{-3}\left(\psi_{ch_i}\otimes\psi_{ch_j}\otimes\hat{\boldsymbol{r}}_{ch_i,ch_j}\right)\right), \quad (12)$$

where $\hat{\boldsymbol{r}}_{\mathrm{ch}_i,\mathrm{ch}_j} = \hat{\boldsymbol{r}}_{\mathrm{ch}_i} - \hat{\boldsymbol{r}}_{\mathrm{ch}_j}$ and $\hat{r}_{\mathrm{ch}_i,\mathrm{ch}_j} = \left|\hat{\boldsymbol{r}}_{\mathrm{ch}_i,\mathrm{ch}_j}\right|$.

**[0093]** It will be appreciated that in the above, electrostatics was used only as an example. In practice, there would typically be no need to learn electrostatic interactions because

they are already described by classical physics. Rather, using the zeroth and first order interaction gates may be envisaged as constituents of a larger network for learning more complicated interactions with no simple closed form that nonetheless broadly follow similar scaling laws as classical interactions.

**[0094] C. Clebsch–Gordan Transforms**

**[0095]** It may now be explained how the $T^{\ell}_{m}$ projection maps appearing in equation (7) are computed. This is significant because the nonlinearities in N–body neural network as described herein are the tensor products, and, in accordance with example embodiments, the architecture needs to incorporate the ability to reduce vectors into a direct sum of irreducibles again straight after the tensor product operation.

**[0096]** To this end the inventor has recognized that representation theory provides a clear prescription for how this operation is to be performed. For any compact group $G$, given two irreducible representations $\rho_{\ell_1}$ and $\rho_{\ell_2}$, the decomposition of $\rho_{\ell_1} \otimes \rho_{\ell_2}$ into a direct sum of irreducibles

$$\rho_{\ell_1}(R) \otimes \rho_{\ell_2}(R) = C^{\dagger}_{\ell_1,\ell_2} \left[ \oplus_{\ell} \oplus_{m=1}^{\kappa_{\tau_1,\tau_2}(\ell)} \rho_{\ell}(R) \right] C_{\ell_1,\ell_2} \tag{13}$$

is called the Clebsch–Gordan transform. In the specific case of SO(3), the $\kappa$ multiplicities take on the very simple form

$$\kappa_{\ell_1,\ell_2}(\ell) = \begin{cases} 1 & \text{if } |\ell_1 - \ell_2| \leq \ell \leq \ell_1 + \ell_2 \\ 0 & \text{otherwise,} \end{cases}$$

and the elements of the $C_{\ell_1,\ell_2}$ matrices can also be computed relatively easily via closed form formulae.

**[0097]** It may be seen immediately that equation (13) prescribes how to reduce the product of covariant vectors into irreducible fragments. Assuming for example that $\psi_1$ is an

irreducible $\rho_{\ell_1}$ vector and $\psi_2$ is an irreducible $\rho_{\ell_2}$ vector, $\psi_1 \otimes \psi_2$ decomposes into irreducible fragments in the form

$$\psi_1 \otimes \psi_2 = \oplus_{\ell=|\ell_1-\ell_2|}^{\ell_1+\ell_2} \bar{\psi}^{\ell} \quad \text{where} \quad \bar{\psi}^{\ell} = C_{\ell_1,\ell_2,\ell}(\psi_1 \otimes \psi_2),$$

and $C_{\ell_1,\ell_2,\ell}$ is the part of $C_{\ell_1,\ell_2}$ matrix corresponding to the $\ell$'th "block." Thus, in this case the operator $T_1^{\ell}$ just corresponds to mutiplying the tensor product by $C_{\ell_1,\ell_2,\ell}$. By linearity, the above relationship also extends to non-irreducible vectors. If $\psi_1$ is of type $\tau_1$ and $\psi_2$ is of type $\tau_2$, then

$$\psi_1 \otimes \psi_2 = \oplus_{\ell} \oplus_{m=1}^{\kappa_{\tau_1,\tau_2}(\ell)} \bar{\psi}_m^{\ell}$$

where

$$\kappa_{\tau_1,\tau_2}(\ell) = \sum_{\ell_1} \sum_{\ell_2} [\tau_1]_{\ell_1} \cdot [\tau_2]_{\ell_2} \cdot \mathbb{I}[|\ell_1 - \ell_2| \leq \ell \leq \ell_1 + \ell_2],$$

and $\mathbb{I}[\cdot]$ is the indicator function. Once again, the actual $\bar{\psi}_m^{\ell}$ fragments are computed by applying the appropriate $C_{\ell_1,\ell_2,\ell}$ matrix to the appropriate combination of irreducible fragments of $\psi_1$ and $\psi_2$. It is also clear that by applying the Clebsch–Gordan decomposition recurisively, a tensor product of any order may be decomposed, for example,

$$\psi_1 \otimes \psi_2 \otimes \psi_3 \otimes \ldots \otimes \psi_k = \big((\psi_1 \otimes \psi_2) \otimes \psi_3\big) \otimes \ldots \otimes \psi_k.$$

**[0098]** In practical computations of such higher order products, optimizing the order of operations and reusing potential intermediate results may be used to minimize computational cost.

## V. Example Method

**[0099]** Example methods may be implemented as machine language instructions stored one or another form of the computer-readable storage, and accessible by the one or more processors of a computing device and/or system, and that, when executed by the one or more processors cause the computing device and/or system to carry out the various operations and functions of the methods described herein. By way of example, storage for instructions may

include a non-transitory computer readable medium. In example operation, the stored instructions may be made accessible to one or more processors of a computing device or system. Execution of the instructions by the one or more processors may then cause the computing device or system to carry various operations of the example method.

**[0100]** Figure 5 is a flow chart of an example method 500, according to example embodiments. Specifically, example method 500 is a computational method for simulating an N–body physical system, wherein the N–body physical system is represented mathematically as a compound object X having N elementary parts $E = \{e_i\}$, $i = 1,\ldots,N$, each $e_i$ representing one of the N bodies of the N–body physical system. In accordance with example embodiments, X may be hierarchically decomposed into J subsystems, $P_j$, $j = 1,..,J$, each $P_j$ may include one or more of the elementary parts of E. Further, each $P_j$ may be described by a position vector $\mathbf{r}_j$ and an internal state vector $\psi_j$. The steps of example method 500 may be carried out by a computing device, such as computing device 100.

**[0101]** At step 502, a hierarchical artificial neural network (ANN) having J nodes each corresponding to one of the J subsystems, may be constructed. In the context of a computer-implemented method, "constructing" an ANN may correspond to implementing the ANN in software or other machine language code. This may entail implementing data structures and operational and/or functional objects according to predefined classes as specified in various instructions, for example. The J nodes may $m \geq 2$ leaf nodes as ANN inputs, $m = 1$ non-leaf root node as ANN output, and $m \geq 1$ intermediate non-leaf nodes. Each node may be considered a neuron of the ANN and may be configured to compute an activation corresponding to a different one of the internal state vectors $\psi_j$ according to node type. In particular, for each leaf node, $\psi_j$ may describe the internal state of a respective one of the $P_j$ subsystems having just a single

elementary part $e_i$, for each given intermediate non-leaf node, $\psi_j$ may describe the internal state of a respective one of the $P_j$ subsystems having $2 \leq k < N$ parts $e_i$ that are each comprised in a child node of the given intermediate non-leaf node; and for the root node, $\psi_j$ may describe the internal state of a subsystem $P_j$ having $k = N$ elementary parts $e_i$ that are each part of a child node of the root node.

**[0102]** At step 502, the computing device may receive input data to the leaf nodes specifying respective position vectors and respective internal state vectors of the N elementary parts E.

**[0103]** At step 506, for each given non-leaf node, $\psi_j$ may be computed from the position vectors and internal states of all the child nodes of the given non-leaf node according to a covariant aggregation rule that represents $\psi_j$ as a tensor object that is covariant to rotations of the rotation group SO(3).

**[0104]** At step 508, a Clebsch-Gordan transform may be applied to reduce tensor products of the state vectors of the nodes to irreducible covariant vectors.

**[0105]** Finally, at step 510, $\psi_j$ of the root node may be computed as output of the ANN. As such, the result may take the form of, or correspond to, a simulation of the internal state of the N–body physical system.

**[0106]** In accordance with example embodiments, the tensor products of the state vectors and application of the Clebsch-Gordan transform entail mathematical operations that are nonlinear. Further applying the Clebsch-Gordan transform to reduce the tensor products of the state vectors of the nodes to irreducible covariant vectors may entail applying the nonlinear operations in Fourier space.

[0107] In accordance with example embodiments, the $m \geq 2$ leaf nodes may form an input layer of the hierarchical ANN, the $m = 1$ non-leaf root node may form an single-node output layer of the hierarchical ANN, and the $m \geq 1$ intermediate non-leaf nodes may be distributed among $m \geq 1$ intermediate layers of the hierarchical ANN. In addition, the hierarchical ANN is one of a strict tree-like structure, or a non-strict tree-like structure. As described above, in a strict tree-like structure, each successive layer after the input layer may include one or more parent nodes of one or more child nodes that reside only in an immediately preceding layer. As also described above, in a non-strict tree structure, each successive layer after the input layer may include one or more parent nodes of one or more child nodes that reside among more than preceding layer.

[0108] In further accordance with example embodiments, each given non-leaf node computing $\psi_j$ from the position vectors and internal states of all the child nodes of the given non-leaf node may entail the given non-leaf node receiving the activation of each of its child nodes. In an example embodiment, the activation of each given child node may correspond to the internal state of the given child node.

[0109] In accordance with example embodiments, the J subsystems may correspond to a hierarchy of substructures of the compound object X, from smallest to largest, the largest corresponding to the entirety of X. In this scheme, each of the $P_j$ subsystems that has just a single elementary part $e_i$ may correspond to a single one of the smallest substructures, the subsystem $P_j$ that has $k = N$ elementary parts $e_i$ may correspond to the largest substructure, and wherein the $P_j$ subsystems that have $2 \leq k < N$ parts $e_i$ may correspond to substructures between the smallest and largest.

[0110] In further accordance with example embodiments, the J subsystems may correspond to a hierarchy of substructures of the compound object X, such that each node of the hierarchical ANN corresponds to one of the substructures of the compound object X. As such, each respective non-leaf node may correspond to a respective substructure of the compound object X that includes the substructures of all of the child nodes of the respective non-leaf node, and each respective leaf node may correspond to a particular substructure of the compound object X comprising a single elementary part $e_i$. In an example embodiment, the internal state of each given subsystem may then correspond to a respective potential energy function due to physical interactions among the substructures of the child nodes of the node corresponding to the given subsystem.

[0111] In still further accordance with example embodiments, the hierarchical ANN may include adjustable weights shared among two or more of the nodes, such that the method further comprises training the ANN to learn the potential energy functions of all of the subsystems by adjusting the weights of the nodes corresponding to the subsystems.

[0112] In further accordance with example embodiments, training the ANN to learn the potential energy functions may entail providing training data to the input layer, where the training data includes for the N–body physical system one or more known training sets. Each training set may include (i) a given configuration of position vectors, and (ii) a known potential function for the given configuration. Training may thus entail, for each of the training sets, comparing a computed potential function output from the non-leaf root node with the known potential function for the given configuration, and based on the comparing, adjusting the weights to achieve agreement, to within a threshold level, between the computed potential functions and the known potential functions across the training sets.

**[0113]** Further, as the training sets may be associated with multiple different known configurations, an N–body comp-net may learn to recognize potentials from multiple examples. In this way, the N–body comp-net may later be applied to provide simulation results for new configurations that have not been previously analyzed. And as discussed above, learning molecular potentials represents a non-limiting example of physical properties or characteristics that an N–body comp-net may learn during training, and later predict from "live" testing data.

**[0114]** In further accordance with example embodiments, each of the training sets may include empirical measurements of the N–body physical system, ab initio computations of forces and energies of the N–body physical system, or a mixture of both.

**[0115]** In an example embodiment, method 500 may be applied to simulate molecules. As such, the compound object X may be or include molecules, and each elementary part $e_i$ may be an atom. In this application of method 500, $\psi_j$ for each node may represent atomic potentials and forces experienced by each corresponding subsystem $P_j$ due the presence and relative positions of each of the other $P_j$ subsystems.

## Conclusion

**[0116]** Using neural networks to learn to the behavior and properties of complex physical systems shows considerable promise. However, physical systems have nontrivial invariance properties (in particular, invariance to translations, rotations and the exchange of identical elementary parts) that must be strictly respected.

**[0117]** Methods and systems disclosed here employ a new type of generalized convolutional neural network architecture, N–body networks, which provides a flexible framework for modeling interacting systems of various types, while taking into account these invariances (symmetries). An example application for N–body networks learning atomic potentials (force fields) for molecular dynamics simulations. However, N–body networks may be used more broadly, for modeling a variety of systems.

**[0118]** N–body networks are distinguished from earlier neural network models for physical systems in that

1. The model is based on a hierarchical (but not necessarily strictly tree-like) decomposition of the system into subsystems at different levels, which is directly reflected in the structure of the neural network.
2. Each subsystem is identified with a "neuron" (or "gate") $n_i$ in the network, and the output (activation) $\psi_i$ of the neuron becomes a representation of the subsystem's internal state.
3. The $\psi_i$ states are tensorial objects with spatial character, in particular they are covariant with rotations in the sense that they transform under rotations according to specific irreducible representations of the rotation group. The gates are specially constructed to ensure that this covariance property is preserved through the network.

4. Unlike most other neural network architectures, the nonlinearities in N–body networks are not pointwise operations, but are applied in "Fourier space," i.e., directly to the irreducible parts of the state vector objects. This is only possible because (a) the nonlinearities arise as a consequence of taking tensor products of covariant objects (b) the tensor products are decomposed into irreducible parts by the Clebsch–Gordan transform.

**[0119]** Advantageously, the last of these ideas may be particularly promising, because it allows for constructing neural that operate entirely in Fourier space, and use tensor products combined with Clebsch–Gordan transforms to induce nonlinearities.

**[0120]** While example embodiments of N–body networks have been described in terms of molecular or atomic systems and potentials, applicability may be significantly broader. In particular, while $\psi_j$ of a given subsystem has be described as the "internal state" of a system (or subsystem), this should not be interpreted as limiting the scope with respect to other applications.

**[0121]** In addition, application of N–body networks to learning the energy function of the system is also just one possible non-limiting example. In particular, the architecture can also be used for learning a variety of other things, such as solubility, affinity for binding to some kind of target, and as well as other physical, chemical, or biological properties.

**[0122]** As a further example of broader applicability, DFT (e.g., ab initio) and other models that may provide training data and models for N–body networks can provide forces in addition to energies. The force information may be relatively easily integrated into the N–body network framework because the force is the gradient of the energy, and neural networks already propagate gradients. This opens the possibility of learning from derivatives as well.

**[0123]** More generally, neural networks may be flexibly extended and/or applied in joint operation. As such, the example application described herein may be considered a convenient

supervised learning setting for illustrative purposes. However, applying the Clebsch-Gordan approach to N–body comp-nets may also be used (possibly as part of a larger architecture) to optimize the structure of atomic systems or generate new molecules for a particular goal, such as drug design.

**[0124]** Example embodiments herein provide a novel and efficient approach to computationally simulating an N–body physical system with covariant, compositional neural networks.

**[0125]** While various aspects and embodiments have been disclosed herein, other aspects and embodiments will be apparent to those skilled in the art. The various aspects and embodiments disclosed herein are for purpose of illustration and are not intended to be limiting, with the true scope being indicated by the following claims.

## CLAIMS

What is claimed is:

1. A method for computationally simulating an N–body physical system, wherein the N–body physical system is represented mathematically as a compound object X having N elementary parts $E = \{e_i\}$, $i = 1,\ldots,N$, each $e_i$ representing one of the N bodies of the N–body physical system, wherein X is hierarchically decomposed into J subsystems, $P_j$, $j = 1,..,J$, each $P_j$ comprising one or more of the elementary parts of E, and wherein each $P_j$ is described by a position vector $\mathbf{r}_j$ and an internal state vector $\psi_j$, the method being implemented on a computing device and comprising:

constructing a hierarchical artificial neural network (ANN) having J nodes each corresponding to one of the J subsystems, the J nodes including $m \geq 2$ leaf nodes as ANN inputs, $m = 1$ non-leaf root node as ANN output, and $m \geq 1$ intermediate non-leaf nodes, wherein each node is a neuron of the ANN and is configured to compute an activation corresponding to a different one of the internal state vectors $\psi_j$, and wherein:

for each leaf node, $\psi_j$ describes the internal state of a respective one of the $P_j$ subsystems having just a single elementary part $e_i$,

for each given intermediate non-leaf node, $\psi_j$ describes the internal state of a respective one of the $P_j$ subsystems having $2 \leq k < N$ parts $e_i$ that are each comprised in a child node of the given intermediate non-leaf node,

and for the root node, $\psi_j$ describes the internal state of a subsystem $P_j$ having $k = N$ elementary parts $e_i$ that are each comprised in a child node of the root node;

at the computing device, receiving input data to the leaf nodes specifying respective position vectors and respective internal state vectors of the N elementary parts E;

for each given non-leaf node, computing $\psi_j$ from the position vectors and internal states of all the child nodes of the given non-leaf node according to a covariant aggregation rule that represents $\psi_j$ as a tensor object that is covariant to rotations of the rotation group SO(3);

applying a Clebsch-Gordan transform to reduce tensor products of the state vectors of the nodes to irreducible covariant vectors; and

computing $\psi_j$ of the root node as output of the ANN, to determine a simulation of the internal state of the N–body physical system.

2. The method of claim 1, wherein the tensor products of the state vectors and application of the Clebsch-Gordan transform comprise mathematical operations that are nonlinear,

and wherein applying the Clebsch-Gordan transform to reduce the tensor products of the state vectors of the nodes to irreducible covariant vectors comprises applying the nonlinear operations in Fourier space.

3. The method of claim 1, wherein the $m \geq 2$ leaf nodes form an input layer of the hierarchical ANN, the $m = 1$ non-leaf root node forms an single-node output layer of the hierarchical ANN, and the $m \geq 1$ intermediate non-leaf nodes are distributed among $m \geq 1$ intermediate layers of the hierarchical ANN,

and wherein the hierarchical ANN is one of:

a strict tree structure, each successive layer after the input layer comprising one or

more parent nodes of one or more child nodes that reside only in an immediately preceding layer; or

a non-strict tree structure, each successive layer after the input layer comprising one or more parent nodes of one or more child nodes that reside among more than preceding layer.

4. The method of claim 3, wherein each given non-leaf node computing $\psi_j$ from the position vectors and internal states of all the child nodes of the given non-leaf node comprises the given non-leaf node receiving the activation of each of its child nodes, the activation of each given child node comprising the internal state of the given child node.

5. The method of claim 1, wherein the J subsystems correspond to a hierarchy of substructures of the compound object X, from smallest to largest, the largest corresponding to the entirety of X,

wherein each of the $P_j$ subsystems that has just a single elementary part $e_i$ corresponds to a single one of the smallest substructures,

wherein the subsystem $P_j$ that has $k = N$ elementary parts $e_i$ corresponds to the largest substructure,

and wherein the $P_j$ subsystems that have $2 \leq k < N$ parts $e_i$ correspond to substructures between the smallest and largest.

6. The method of claim 1, wherein the J subsystems correspond to a hierarchy of substructures of the compound object X, and wherein each node of the hierarchical ANN

corresponds to one of the substructures of the compound object X,

wherein each respective non-leaf node corresponds to a respective substructure of the compound object X comprising the substructures of all of the child nodes of the respective non-leaf node,

wherein each respective leaf node corresponds to a particular substructure of the compound object X comprising a single elementary part $e_i$,

and wherein the internal state of each given subsystem corresponds to a respective potential energy function due to physical interactions among the substructures of the child nodes of the node corresponding to the given subsystem.

7. The method of claim 6, wherein the hierarchical ANN comprises adjustable weights shared among two or more of the nodes,

and wherein the method further comprises training the ANN to learn the potential energy functions of all of the subsystems by adjusting the weights of the nodes corresponding to the subsystems.

8. The method of claim 7, wherein training the ANN to learn the potential energy functions comprises:

providing training data to the input layer, the training data comprising for the N–body physical system one or more known training sets, each including: (i) a given configuration of position vectors, and (ii) a known potential function for the given configuration;

for each of the training sets, comparing a computed potential function output from the non-leaf root node with the known potential function for the given configuration; and

based on the comparing, adjusting the weights to achieve agreement, to within a threshold level, between the computed potential functions and the known potential functions across the training sets.

9. The method of claim 8, wherein each of the training sets is at least one of empirical measurements of the N–body physical system, or ab initio computations of forces and energies of the N–body physical system.

10. The method of claim 1, wherein the compound object X is comprised of molecules, wherein each elementary part $e_i$ is an atom,

and wherein $\psi_j$ for each node represents atomic potentials and forces experienced by each corresponding subsystem $P_j$ due the presence and relative positions of each of the other $P_j$ subsystems.

11. A computing device configured for computationally simulating an N–body physical system, wherein the N–body physical system is represented mathematically as a compound object X having N elementary parts $E = \{e_i\}$, $i = 1,\ldots,N$, each $e_i$ representing one of the N bodies of the N–body physical system, wherein X is hierarchically decomposed into J subsystems, $P_j$, $j = 1,..,J$, each $P_j$ comprising one or more of the elementary parts of E, and wherein each $P_j$ is described by a position vector $\mathbf{r}_j$ and an internal state vector $\psi_j$, the computing device comprising:

one or more processors; and

memory configured to store computer-executable instructions that, when executed by the

one or more processors, cause the computing device to carry out computational operations including:

constructing a hierarchical artificial neural network (ANN) having J nodes each corresponding to one of the J subsystems, the J nodes including $m \geq 2$ leaf nodes as ANN inputs, $m = 1$ non-leaf root node as ANN output, and $m \geq 1$ intermediate non-leaf nodes, wherein each node is a neuron of the ANN and is configured to compute an activation corresponding to a different one of the internal state vectors $\psi_j$, and wherein:

for each leaf node, $\psi_j$ describes the internal state of a respective one of the $P_j$ subsystems having just a single elementary part $e_i$,

for each given intermediate non-leaf node, $\psi_j$ describes the internal state of a respective one of the $P_j$ subsystems having $2 \leq k < N$ parts $e_i$ that are each comprised in a child node of the given intermediate non-leaf node,

and for the root node, $\psi_j$ describes the internal state of a subsystem $P_j$ having $k = N$ elementary parts $e_i$ that are each comprised in a child node of the root node;

receiving input data to the leaf nodes specifying respective position vectors and respective internal state vectors of the N elementary parts E;

for each given non-leaf node, computing $\psi_j$ from the position vectors and internal states of all the child nodes of the given non-leaf node according to a covariant aggregation rule that represents $\psi_j$ as a tensor object that is covariant to rotations of the rotation group SO(3);

applying a Clebsch-Gordan transform to reduce a tensor product of the state vectors of the nodes to irreducible covariant vectors; and

computing $\psi_j$ of the root node as output of the ANN to determine the internal state of the N–body physical system.

12. The computing device of claim 11, wherein the tensor products of the state vectors and application of the Clebsch-Gordan transform comprise mathematical operations that are nonlinear,

and wherein applying the Clebsch-Gordan transform to reduce the tensor products of the state vectors of the nodes to irreducible covariant vectors comprises applying the nonlinear operations in Fourier space.

13. The computing device of claim 11, wherein the $m \geq 2$ leaf nodes form an input layer of the hierarchical ANN, the $m = 1$ non-leaf root node forms an single-node output layer of the hierarchical ANN, and the $m \geq 1$ intermediate non-leaf nodes are distributed among $m \geq 1$ intermediate layers of the hierarchical ANN,

wherein the hierarchical ANN is one of:

a strict tree structure, each successive layer after the input layer comprising one or more parent nodes of one or more child nodes that reside only in an immediately preceding layer; or

a non-strict tree structure, each successive layer after the input layer comprising one or more parent nodes of one or more child nodes that reside among more than preceding layer,

and wherein each given non-leaf node computing $\psi_j$ from the position vectors and internal states of all the child nodes of the given non-leaf node comprises the given non-leaf node receiving the activation of each of its child nodes, the activation of each given child node comprising the internal state of the given child node.

14. The computing device of claim 11, wherein the J subsystems correspond to a hierarchy of substructures of the compound object X, from smallest to largest, the largest corresponding to the entirety of X,

wherein each of the $P_j$ subsystems that has just a single elementary part $e_i$ corresponds to a single one of the smallest substructures,

wherein the subsystem $P_j$ that has $k = N$ elementary parts $e_i$ corresponds to the largest substructure,

and wherein the $P_j$ subsystems that have $2 \leq k < N$ parts $e_i$ correspond to substructures between the smallest and largest.

15. The computing device of claim 11, wherein the J subsystems correspond to a hierarchy of substructures of the compound object X, and wherein each node of the hierarchical ANN corresponds to one of the substructures of the compound object X,

wherein each respective non-leaf node corresponds to a respective substructure of the compound object X comprising the substructures of all of the child nodes of the respective non-leaf node,

wherein each respective leaf node corresponds to a particular substructure of the compound object X comprising a single elementary part $e_i$,

and wherein the internal state of each given subsystem corresponds to a respective potential energy function due to physical interactions among the substructures of the child nodes of the node corresponding to the given subsystem.

16. The computing device of claim 15, wherein the hierarchical ANN comprises adjustable weights shared among two or more of the nodes,

and wherein the computational operations further comprise training the ANN to learn the potential energy functions of all of the subsystems by adjusting the weights of the nodes corresponding to the subsystems.

17. The computing device of claim 16, wherein training the ANN to learn the potential energy functions comprises:

providing training data to the input layer, the training data comprising for the N–body physical system one or more known training sets, each including: (i) a given configuration of position vectors, and (ii) a known potential function for the given configuration;

for each of the training sets, comparing a computed potential function output from the non-leaf root node with the known potential function for the given configuration; and

based on the comparing, adjusting the weights to achieve agreement, to within a threshold level, between the computed potential functions and the known potential functions across the training sets.

18. The computing device of claim 17, wherein each of the training sets is at least one of empirical measurements of the N–body physical system, or ab initio computations of forces and energies of the N–body physical system

19. The computing device of claim 11, wherein the compound object X is comprised of molecules, wherein each elementary part $e_i$ is an atom,

and wherein $\psi_j$ for each node represents atomic potentials and forces experienced by each corresponding subsystem $P_j$ due the presence and relative positions of each of the other $P_j$ subsystems.

20. An article of manufacture comprising a non-transitory computer readable media having computer-readable instructions stored thereon for computationally simulating an N–body physical system, wherein the N–body physical system is represented mathematically as a compound object X having N elementary parts $E = \{e_i\}$, $i = 1,\ldots,N$, each $e_i$ representing one of the N bodies of the N–body physical system, wherein X is hierarchically decomposed into J subsystems, $P_j$, $j = 1,..,J$, each $P_j$ comprising one or more of the elementary parts of E, and wherein each $P_j$ is described by a position vector $\mathbf{r}_j$ and an internal state vector $\psi_j$, and wherein the instructions, when executed by one or more processors of a computing device, cause the computing device to carry out operations including:

constructing a hierarchical artificial neural network (ANN) having J nodes each corresponding to one of the J subsystems, the J nodes including $m \geq 2$ leaf nodes as ANN inputs, $m = 1$ non-leaf root node as ANN output, and $m \geq 1$ intermediate non-leaf nodes, wherein each node is a neuron of the ANN and is configured to compute an activation corresponding to a different one of the internal state vectors $\psi_j$, and wherein:

for each leaf node, $\psi_j$ describes the internal state of a respective one of the $P_j$ subsystems having just a single elementary part $e_i$,

for each given intermediate non-leaf node, $\psi_j$ describes the internal state of a respective one of the $P_j$ subsystems having $2 \leq k < N$ parts $e_i$ that are each comprised in a child node of the given intermediate non-leaf node,

and for the root node, $\psi_j$ describes the internal state of a subsystem $P_j$ having $k = N$ elementary parts $e_i$ that are each comprised in a child node of the root node;

receiving input data to the leaf nodes specifying respective position vectors and respective internal state vectors of the N elementary parts E;

for each given non-leaf node, computing $\psi_j$ from the position vectors and internal states of all the child nodes of the given non-leaf node according to a covariant aggregation rule that represents $\psi_j$ as a tensor object that is covariant to rotations of the rotation group SO(3);

applying a Clebsch-Gordan transform to reduce a tensor product of the state vectors of the nodes to irreducible covariant vectors; and

computing $\psi_j$ of the root node as output of the ANN to determine the internal state of the N–body physical system.

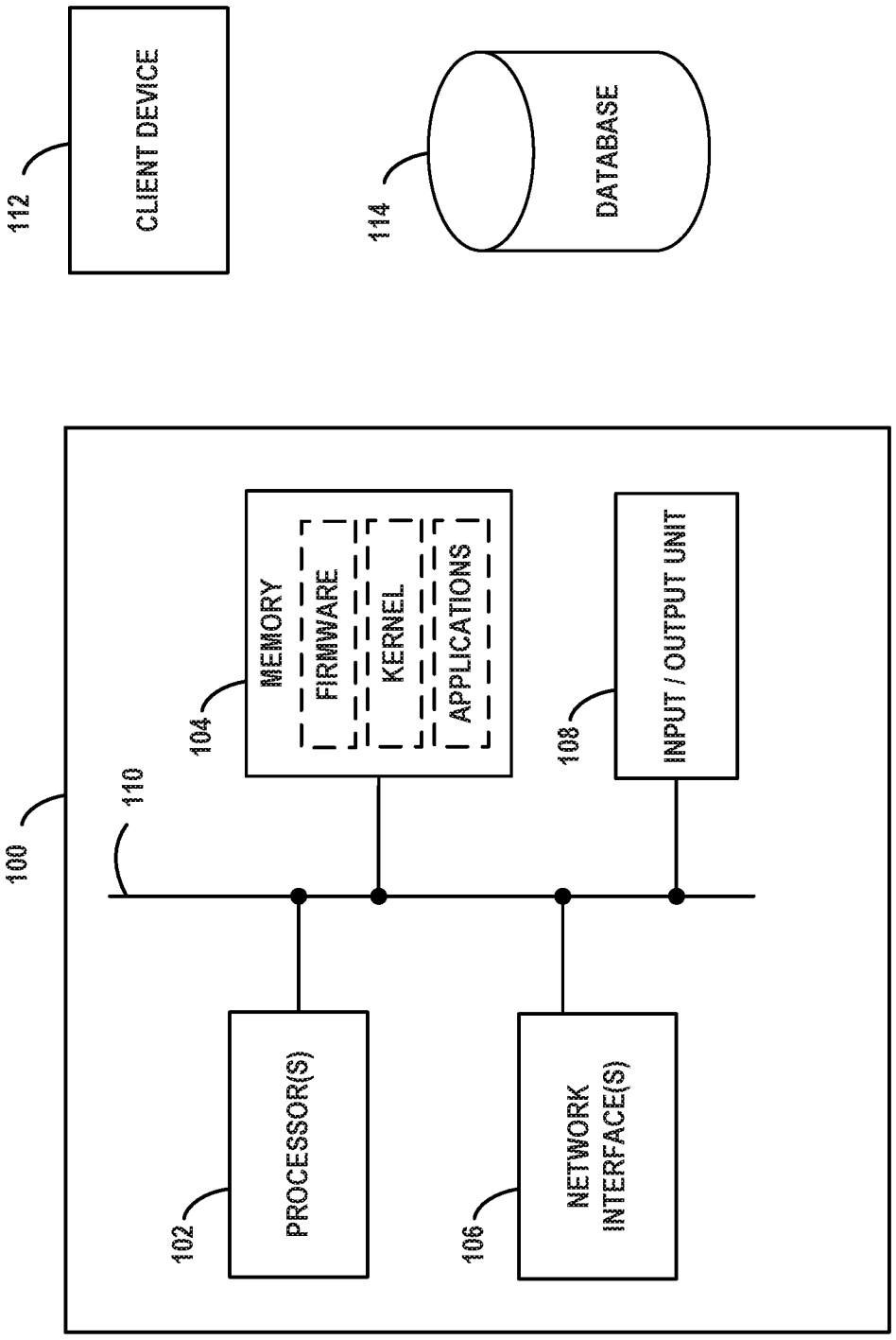

112
CLIENT DEVICE
114
DATABASE
100
110
104
MEMORY
FIRMWARE
KERNEL
APPLICATIONS
108
INPUT / OUTPUT UNIT
102
PROCESSOR(S)
106
NETWORK INTERFACE(S)

FIG. 1

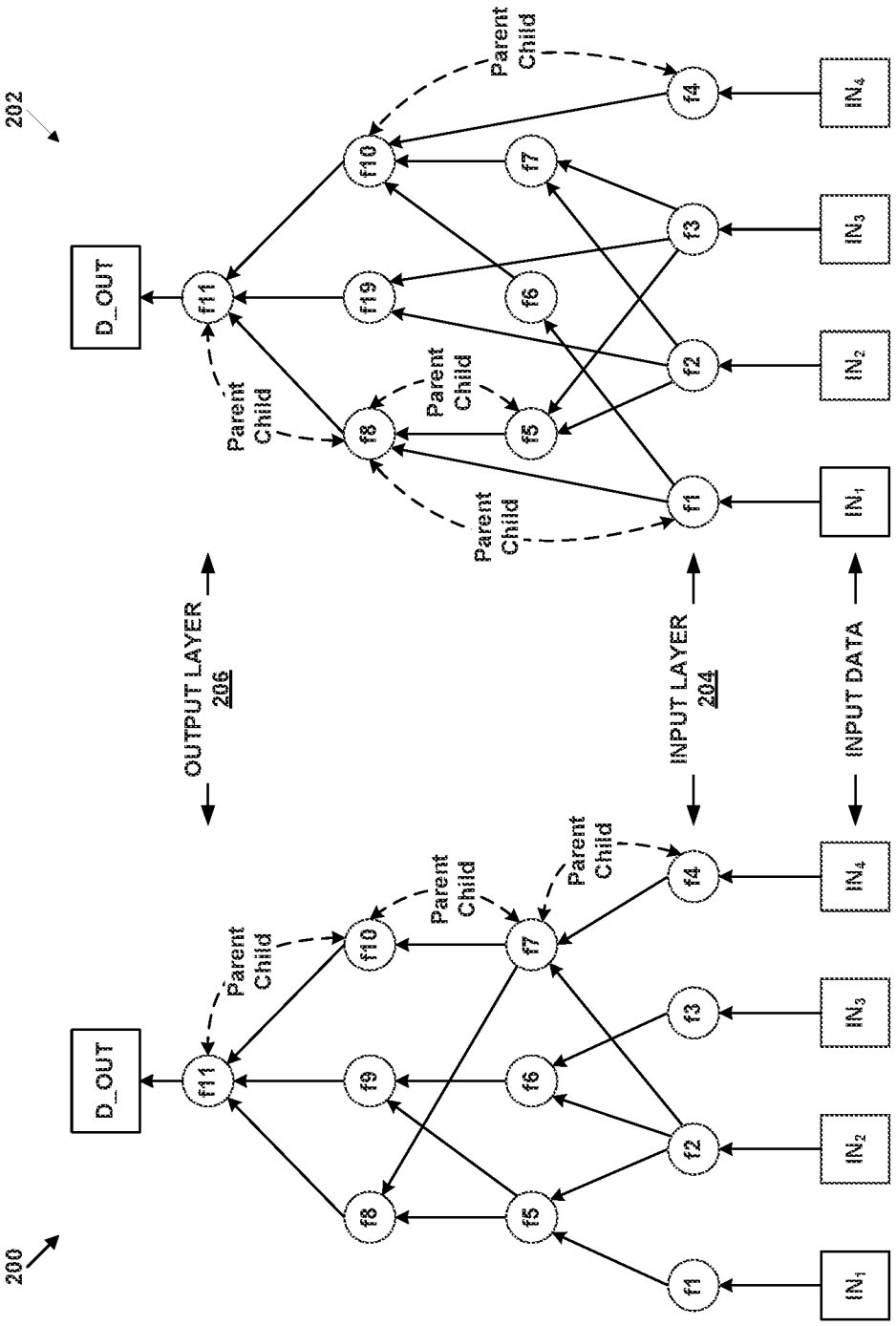

200
202
D_OUT
f11
f10
f9
f8
f7
f6
f5
f4
f3
f2
f1
IN4
IN3
IN2
IN1
Parent
Child
OUTPUT LAYER
206
INPUT LAYER
204
INPUT DATA

FIG. 2

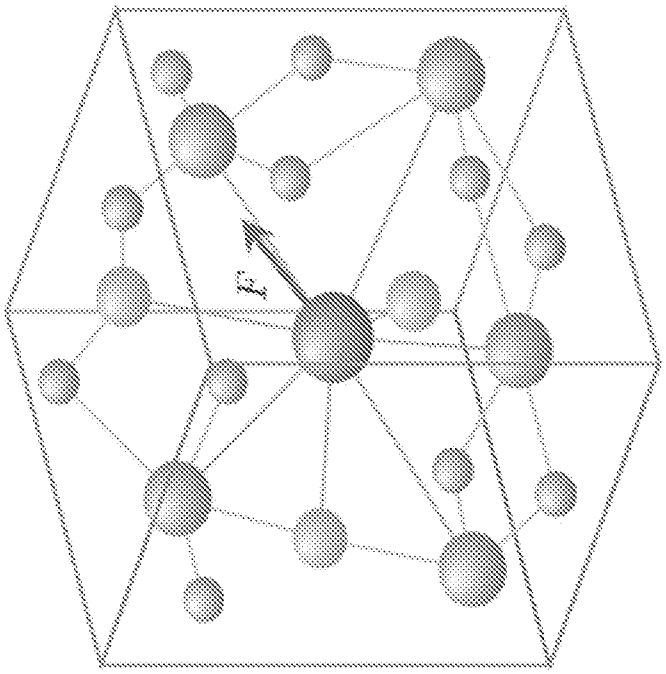


FIG. 3A

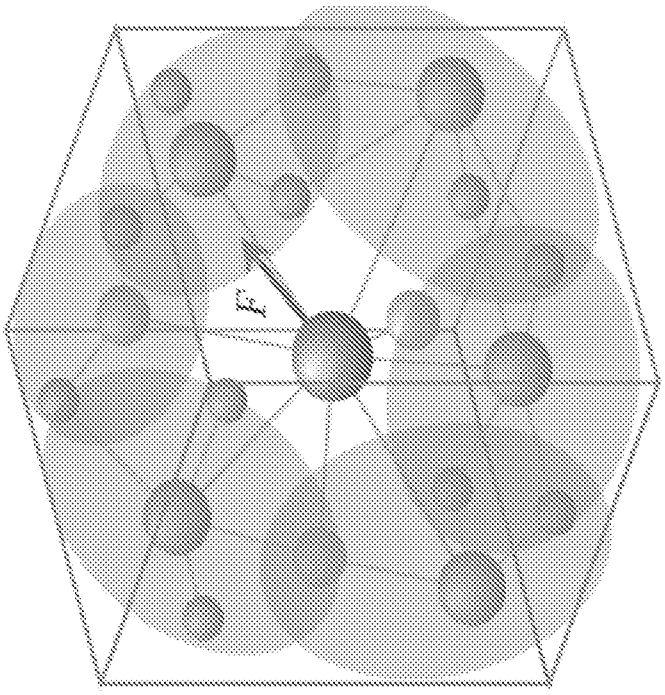


FIG. 3B

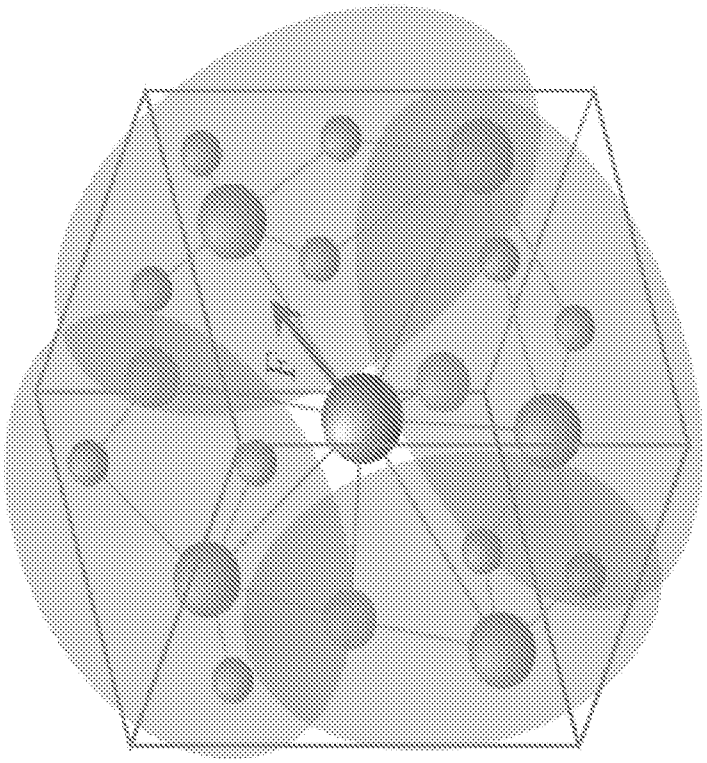


FIG. 3C

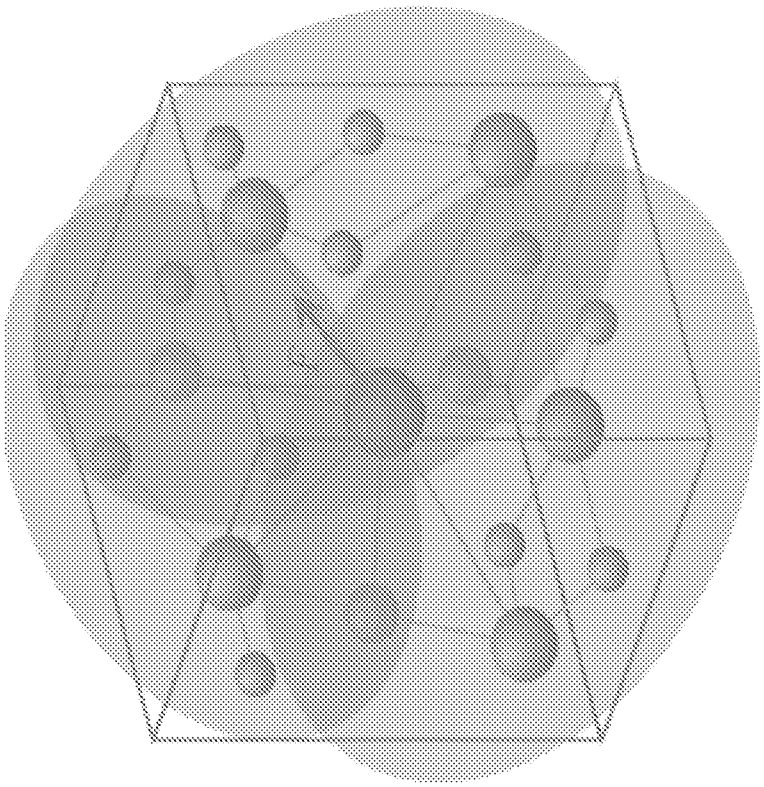


FIG. 3D

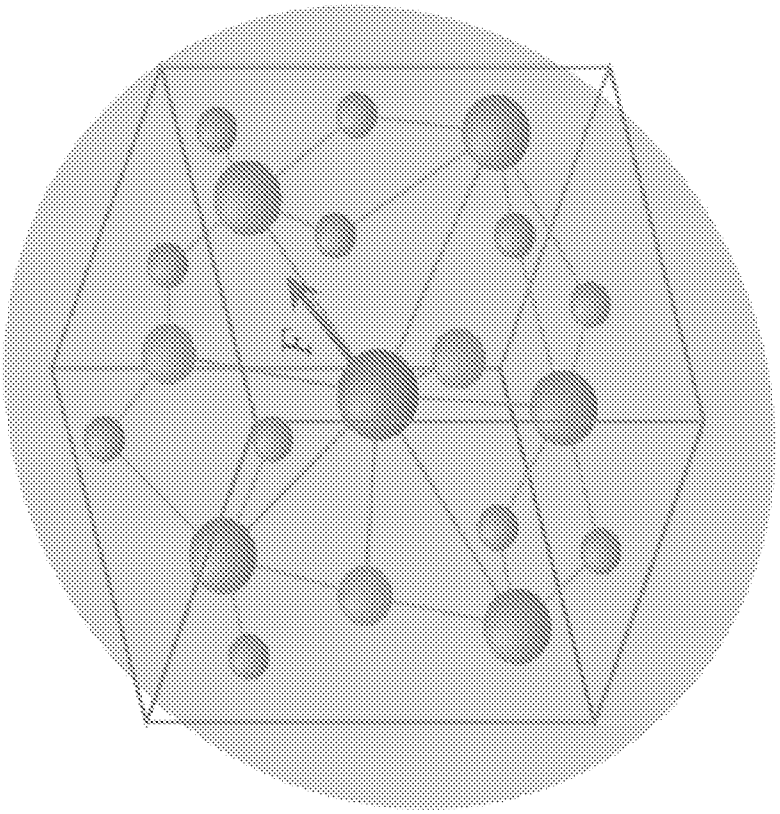


FIG. 3E

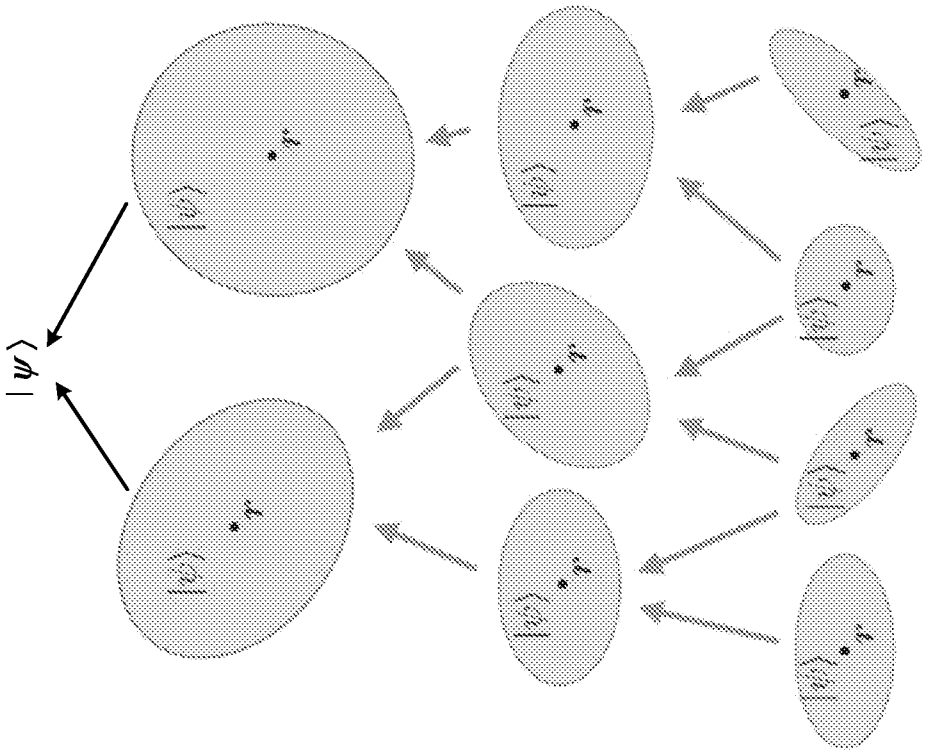



FIG. 3F

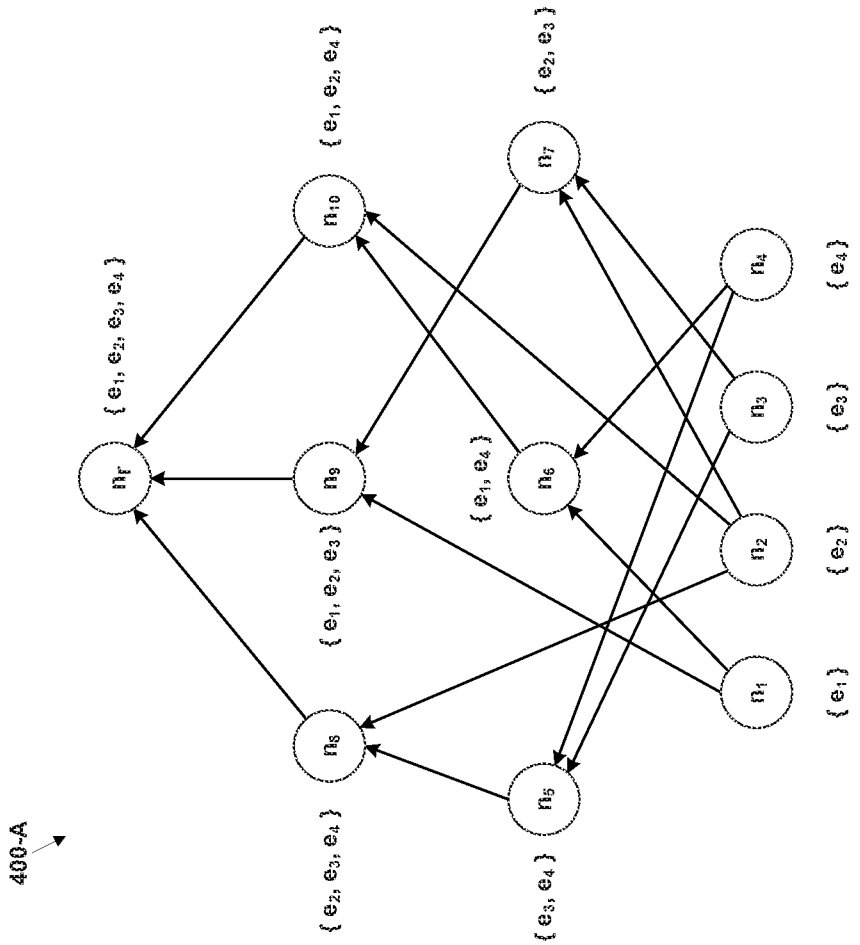

400-A
$n_1$
$n_2$
$n_3$
$n_4$
$n_5$
$n_6$
$n_7$
$n_8$
$n_9$
$n_{10}$
$n_F$
{ $e_1$ }
{ $e_2$ }
{ $e_3$ }
{ $e_4$ }
{ $e_3$, $e_4$ }
{ $e_1$, $e_4$ }
{ $e_2$, $e_3$ }
{ $e_1$, $e_2$, $e_3$ }
{ $e_2$, $e_3$, $e_4$ }
{ $e_1$, $e_2$, $e_4$ }
{ $e_1$, $e_2$, $e_3$, $e_4$ }

FIG. 4A

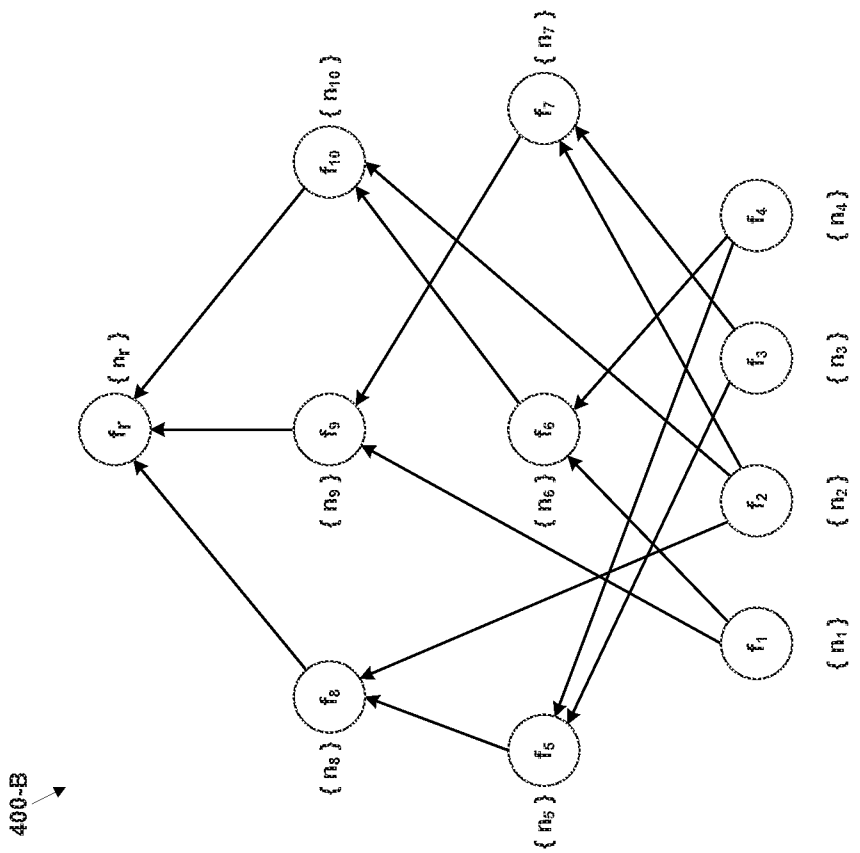

400-B
f1
{n1}
f2
{n2}
f3
{n3}
f4
{n4}
f5
{n5}
f6
{n6}
f7
{n7}
f8
{n8}
f9
{n9}
f10
{n10}
fr
{nr}

FIG. 4B

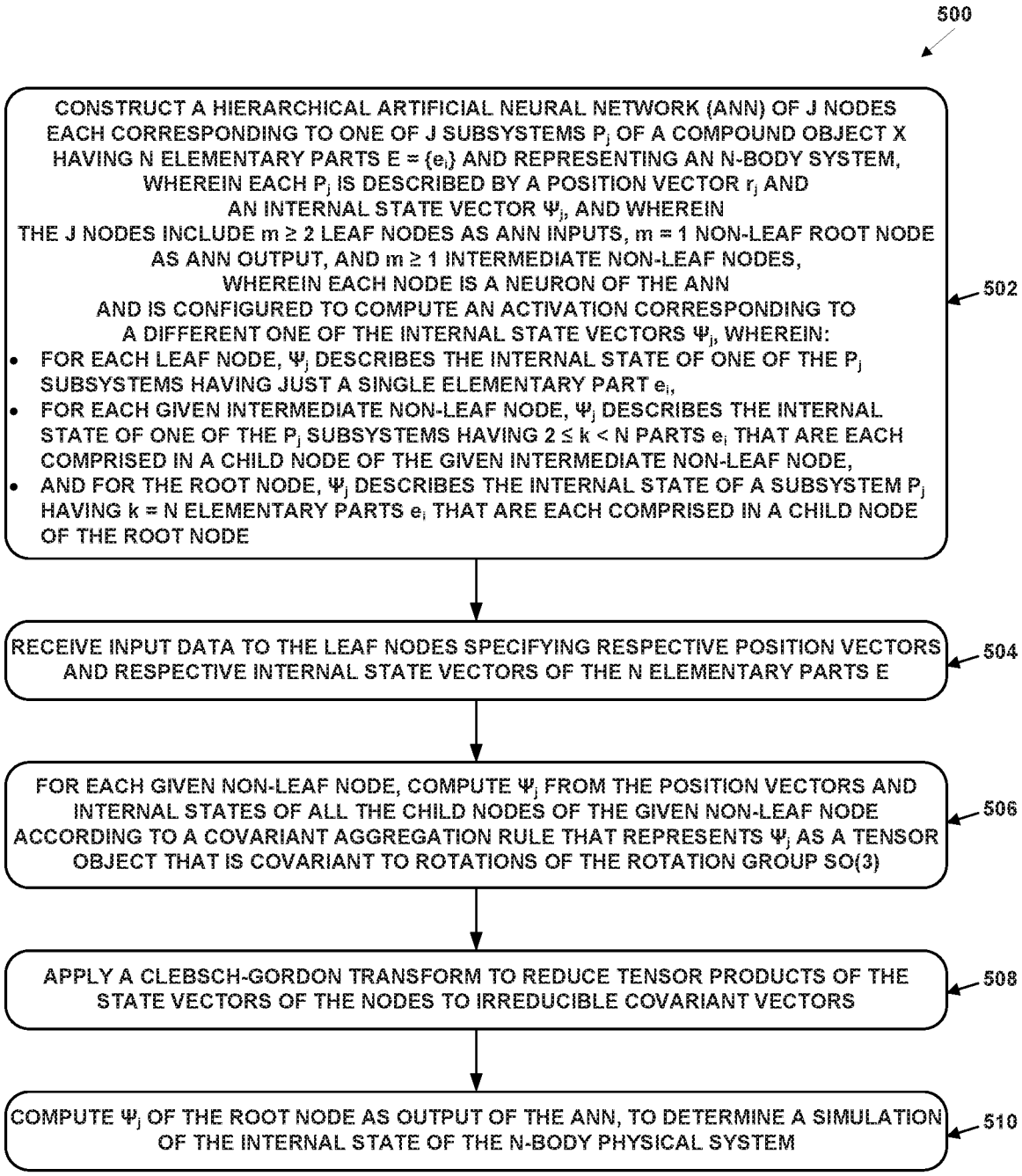
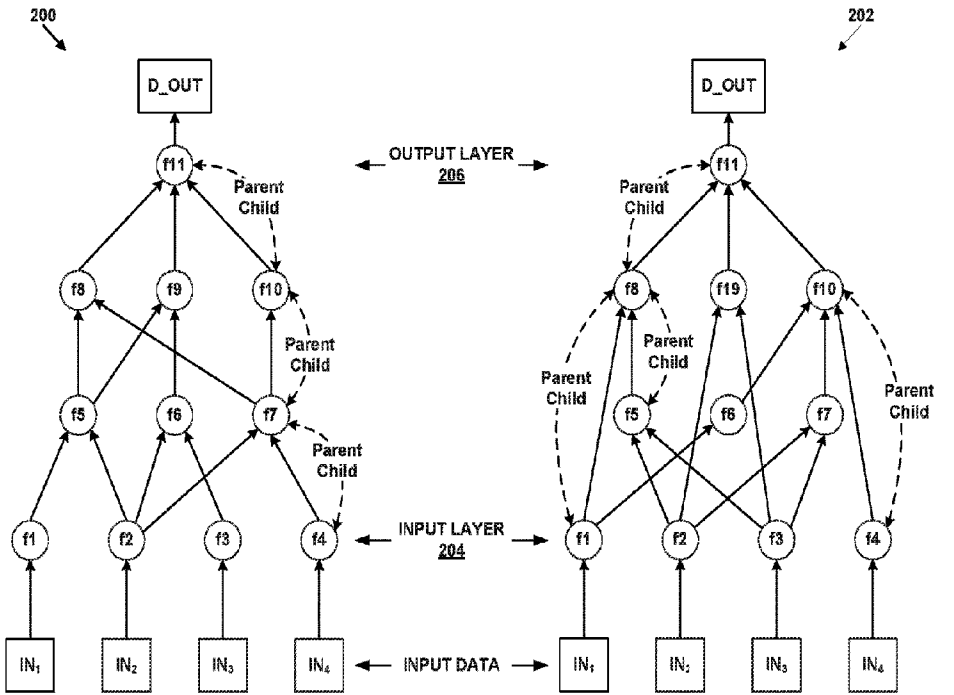

500
CONSTRUCT A HIERARCHICAL ARTIFICIAL NEURAL NETWORK (ANN) OF J NODES EACH CORRESPONDING TO ONE OF J SUBSYSTEMS $P_j$ OF A COMPOUND OBJECT X HAVING N ELEMENTARY PARTS E = {$e_i$} AND REPRESENTING AN N-BODY SYSTEM, WHEREIN EACH $P_j$ IS DESCRIBED BY A POSITION VECTOR $r_j$ AND AN INTERNAL STATE VECTOR $\Psi_j$, AND WHEREIN THE J NODES INCLUDE m ≥ 2 LEAF NODES AS ANN INPUTS, m = 1 NON-LEAF ROOT NODE AS ANN OUTPUT, AND m ≥ 1 INTERMEDIATE NON-LEAF NODES, WHEREIN EACH NODE IS A NEURON OF THE ANN AND IS CONFIGURED TO COMPUTE AN ACTIVATION CORRESPONDING TO A DIFFERENT ONE OF THE INTERNAL STATE VECTORS $\Psi_j$, WHEREIN:
• FOR EACH LEAF NODE, $\Psi_j$ DESCRIBES THE INTERNAL STATE OF ONE OF THE $P_j$ SUBSYSTEMS HAVING JUST A SINGLE ELEMENTARY PART $e_i$,
• FOR EACH GIVEN INTERMEDIATE NON-LEAF NODE, $\Psi_j$ DESCRIBES THE INTERNAL STATE OF ONE OF THE $P_j$ SUBSYSTEMS HAVING 2 ≤ k < N PARTS $e_i$ THAT ARE EACH COMPRISED IN A CHILD NODE OF THE GIVEN INTERMEDIATE NON-LEAF NODE,
• AND FOR THE ROOT NODE, $\Psi_j$ DESCRIBES THE INTERNAL STATE OF A SUBSYSTEM $P_j$ HAVING k = N ELEMENTARY PARTS $e_i$ THAT ARE EACH COMPRISED IN A CHILD NODE OF THE ROOT NODE
502
RECEIVE INPUT DATA TO THE LEAF NODES SPECIFYING RESPECTIVE POSITION VECTORS AND RESPECTIVE INTERNAL STATE VECTORS OF THE N ELEMENTARY PARTS E
504
FOR EACH GIVEN NON-LEAF NODE, COMPUTE $\Psi_j$ FROM THE POSITION VECTORS AND INTERNAL STATES OF ALL THE CHILD NODES OF THE GIVEN NON-LEAF NODE ACCORDING TO A COVARIANT AGGREGATION RULE THAT REPRESENTS $\Psi_j$ AS A TENSOR OBJECT THAT IS COVARIANT TO ROTATIONS OF THE ROTATION GROUP SO(3)
506
APPLY A CLEBSCH-GORDON TRANSFORM TO REDUCE TENSOR PRODUCTS OF THE STATE VECTORS OF THE NODES TO IRREDUCIBLE COVARIANT VECTORS
508
COMPUTE $\Psi_j$ OF THE ROOT NODE AS OUTPUT OF THE ANN, TO DETERMINE A SIMULATION OF THE INTERNAL STATE OF THE N-BODY PHYSICAL SYSTEM
510

FIG. 5


200
202
D_OUT
D_OUT
OUTPUT LAYER
206
f11
f11
Parent
Child
Parent
Child
f8
f9
f10
f8
f19
f10
Parent
Child
Parent
Child
Parent
Child
Parent
Child
f5
f6
f7
f5
f6
f7
Parent
Child
f1
f2
f3
f4
INPUT LAYER
204
f1
f2
f3
f4
IN1
IN2
IN3
IN4
INPUT DATA
IN1
IN2
IN3
IN4