

(51) International Patent Classification:
G06F 9/50 (2006.01)(21) International Application Number:
PCT/MY2014/000052(22) International Filing Date:
4 April 2014 (04.04.2014)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
PI2013001230 5 April 2013 (05.04.2013) MY

(71) Applicant: MIMOS, Berhad [MY/MY]; Technology Park Malaysia, 57000 Kuala Lumpur (MY).

(72) Inventors: Mohanavelu Mudaliar A/L Senapen; c/o MIMOS Berhad, Technology Park Malaysia, 57000 Kuala Lumpur (MY). Dr. Ettikan Kandasamy A/L Karuppiyah; c/o MIMOS Berhad, Technology Park Malaysia, 57000 Kuala Lumpur (MY). SOO, Saw Meng; c/o MIMOS Berhad, Technology Park Malaysia, 57000 Kuala Lumpur (MY). LIM, Oon Leong; c/o MIMOS Berhad, Technology Park Malaysia, 57000 Kuala Lumpur (MY). YONG, Keh Kok; c/o MIMOS Berhad, Technology Park Malaysia, 57000 Kuala Lumpur (MY).

(74) Agent: YAP, Kah Hong; Pyprus Sdn Bhd, Suite 8-02, 8th Floor, Plaza First Nationwide 161, Jalan Tun H.S. Lee, 50000 Kuala Lumpur (MY).

(81) Designated States (unless otherwise indicated, for every kind of national protection available): AE, AG, AL, AM, AO, AT, AU, AZ, BA, BB, BG, BH, BN, BR, BW, BY, BZ, CA, CH, CL, CN, CO, CR, CU, CZ, DE, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IR, IS, JP, KE, KG, KN, KP, KR, KZ, LA, LC, LK, LR, LS, LT, 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, 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).

[Continued on next page]

(54) Title: SCALABLE METHOD AND APPARATUS FOR PARALLEL COMPUTATION USING HYBRID HETEROGENEOUS PROCESSOR CAPABLE OF DYNAMIC RESOURCE CONFIGURATION

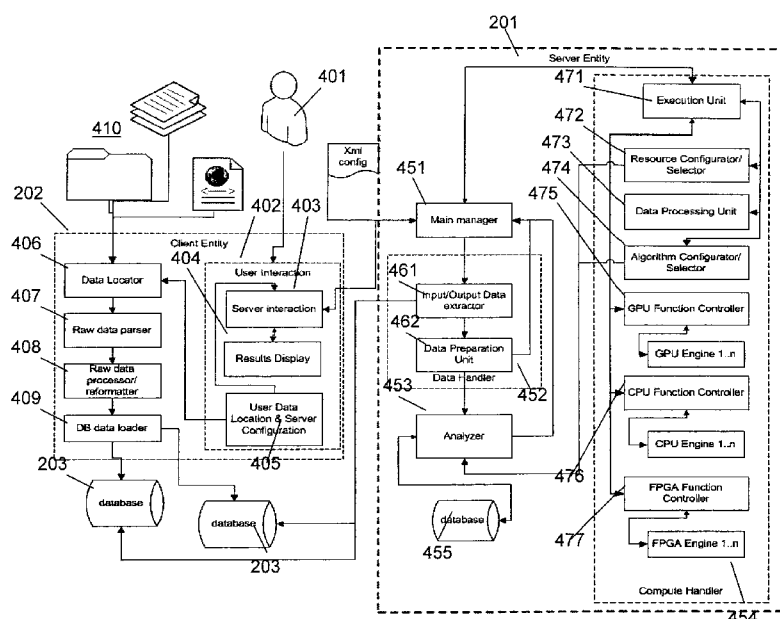


FIG. 4

(57) Abstract: The present invention provides a system and method for determining an application execution path on a parallel computing environment. It introduces a data-set analyzer (702), an algorithm analyzer (704), a compute-resource analyzer (706) and a resource selection and configuration unit (708) to manage the execution paths. The system analyses various parameters to determine an optimal execution path.



Declarations under Rule 4.17:

— *of inventorship (Rule 4.17(iv))*

Published:

— *with international search report (Art. 21(3))*

— *before the expiration of the time limit for amending the claims and to be republished in the event of receipt of amendments (Rule 48.2(h))*

Scalable Method And Apparatus For Parallel Computation Using Hybrid Heterogeneous Processor Capable Of Dynamic Resource Configuration

Field of the Invention

5 **[0001]** The present invention relates to parallel computing. Specifically, the present invention relates to system and method for parallel computation using hybrid heterogeneous processor.

Background

10 **[0002]** Conventional software techniques that relies on general purpose processor alone for computation analysis. These techniques generally suffer from latency issues due to computationally intensive nature of multiple applications acquiring massive amount of data such as financial instrument analytics modeling, which requires quick processing for rendering the required analytical results within a reasonable time frame.

Summary

15 **[0003]** In accordance with one aspect of the present invention, there is provided a method to determine an application execution path on a parallel computing environment. The method comprises analysing inputs data set characteristics and data-set parameters; analysing characteristics of algorithms and algorithm parameters;
20 determining available computing resources and compute-resource parameters; and selecting and configuring specific path for execution based on said outcome.

[0004] In one embodiment, the method further comprises receiving input from a data preparation unit; and receiving feedback from a resource selection and configuration unit adapted for the selecting and configuring the specific path. The receiving feedback further comprises reformatting data to suite the available resource
5 based on a ranked criteria. Further, selecting and configuring the specific path further comprises receiving inputs from data set analyser, algorithm analyser and compute resource analyser; receiving heuristic rules from heuristic rules database; and determining an optimal configuration selection based on the heuristic rules.

[0005] Further the method comprises identifying Available Data-Sets,
10 Algorithm List, Compute-Resource list and make an Initial Data-set List, Algorithm-List, Compute-Resource-List; selecting a parameter from an Initial Data-Set List; adding the parameter to an Ordered Data-set List based on the heuristic rules; selecting a parameter from an Initial Algorithm-Set List; adding the parameter to an Ordered Algorithm-set List based on the heuristic rules; selecting a parameter from an Initial
15 Compute Resource-Set List; adding the parameter to an Ordered Compute Resource-Set List based on the heuristic rules; selecting a parameter from the Ordered Data-set List starting from the first parameter against other parameter listed in the Ordered Algorithms List by satisfying specific heuristic rule; adding a node to a Rule Based Ordered List; checking the parameters matching with the specific heuristic rules;
20 selecting the ordered parameters from rule based ordered list; selecting one parameter from rule based ordered list satisfying the specific heuristic rule; adding a paired parameter to the rule based pair list for the data-set and the algorithm; selecting one paired parameter pair from the rule based ordered list of data-set and algorithm that satisfies a specific heuristic rule; adding the paired parameter to rule based triplet list

comprises the data-set, the algorithm and the computing resource; selecting a triplet parameter from rule based triplet list; selecting an optimum configuration; ranking the optimum configuration based on ranking rules with specific matching from rules database; selecting specific configuration if the specific configuration is available; and
5 selecting a lower rank configuration if the optimal configuration is not available. The method may further comprise returning status if the selected path is not available.

[0006] In another aspect of the present invention, there is provided a system for determining an application execution path on a parallel computing environment. The system comprises a data-set analyser that operably analyses input data-set
10 characteristics and determine data-set parameters; an algorithm analyser that operably analyses characteristics of algorithms and determine algorithm parameters; a compute-resource analyser that operably analyses available computing resources and determine compute-resource parameters; a resource selection and configuration unit that selects and configures specific path for execution based on the outcome rendered by the data-
15 set analyser, the algorithm analyser and the computer-resource analyser.

[0007] In one embodiment, the data-set analyser further comprises a data configuration selector operably selecting an appropriate data configuration based on input from resource selection and configuration executor; an algorithm configuration selector operably selecting an appropriate algorithm configuration based on input from
20 resource selection and configuration executor; and an compute-resource selector operably selecting an appropriate computing resource and configuration based on input from resource selection and configuration executor.

Brief Description of the Drawings

[0008] Preferred embodiments according to the present invention will now be described with reference to the figures accompanied herein, in which like reference numerals denote like elements;

5 [0009] **FIG. 1** illustrates a three dimensional chart of data-set configuration against algorithm configuration against compute resource configuration for a parallel computing framework that are considered in the present invention;

[0010] **FIG. 2** illustrates an overall computing network in accordance with one embodiment of the present invention;

10 [0011] **FIG. 3** illustrates schematically architecture of the computing network of **FIG. 2** in accordance with an embodiment of the present invention;

[0012] **FIG. 4** is a block diagram illustrating the operational flow of the computing network in accordance with one embodiment of the present invention;

[0013] **FIG. 5** exemplifies an execution flow in accordance with one
15 embodiment of the present invention;

[0014] **FIG. 6** illustrates an initialization process in accordance with one embodiment of the present invention;

[0015] **FIG. 7** illustrates a processing flow of the analyzer of **FIG. 4** in accordance with one embodiment of the present invention;

[0016] FIG. 8 illustrates a diagram of the data preparation unit of FIG. 4 preparing data for classification in accordance with one embodiment of the present invention;

[0017] FIG. 9 exemplifies a process flow of data formatting in accordance with one embodiment of the present invention;

[0018] FIG. 10 illustrates a block diagram of the algorithm preparation unit of FIG. 7 in accordance with one embodiment of the present invention;

[0019] FIG. 11 illustrates a process flow in accordance with one embodiment of the present invention;

[0020] FIG. 12 illustrates execution portion for data path of one embodiment of the present invention;

[0021] FIG. 13 illustrates a flow diagram of the computation resource selector in accordance with one embodiment of the present invention; and

[0022] FIG. 14 illustrates a process flow for algorithm selector in accordance with one embodiment of the present invention.

Detailed Description

[0023] Embodiments of the present invention shall now be described in detail, with reference to the attached drawings. It is to be understood that no limitation of the scope of the invention is thereby intended, such alterations and further modifications in the illustrated device, and such further applications of the principles of the invention as

illustrated therein being contemplated as would normally occur to one skilled in the art to which the invention relates.

[0024] FIG. 1 illustrates a three dimensional chart of data-set configuration against algorithm configuration against compute resource configuration for a parallel computing framework that are considered in the present invention. The cross markings “X” indicate possible optimal or desired matching conditions. At it can be shown, the optimal matching conditions may not necessary be set based on one specific configuration, but rather, a mixture of data-set, algorithm and compute resources based a determination in real-time, i.e. various use-case may have the cross marking at a different position.

[0025] FIG. 2 illustrates an overall computing network 200 in accordance with one embodiment of the present invention. The computing network 200 includes servers 201, client computers 202, and databases 203. These devices may be connected via a WAN (including internet) 205 or LAN 206 networks. Each of the servers may further includes one or more processing units, such as Central Processing Units (CPU) 212, Graphical Processing Units (GPU) 211, Field Programmable Grid Arrays (FPGA) 213, or the likes. Each of the processing units or arrays may be single or multi-core processing units. The databases 203 may be any one or more central databases that collate data from various sources. For example, a database that stores finance related information for computing value at risk (VAR) on a financial system may contains a vast amount of historical data, which requires high computational resources to process the data.

[0026] The servers **201** acquire a large amount of data from the databases **203**. The client computers **202**, or simply clients, include applications adapted for displaying or outputting results processed by the computing network. These applications further provide interaction and control module between the users and the servers **201**. The servers **201** can be used as a dedicated computational unit, or viewing client, or the database can be part of server **201**.

[0027] FIG. 3 illustrates schematically architecture of the computing network **200** of FIG. 2 in accordance with an embodiment of the present invention. Computations on the server **201** are handled by one or more processing units **350**, which include CPU, GPU and/or FPGA under a parallel computing environment. The server **201** includes a communication layer **311**, which allows the server **201** to connect to the computing network **200**, an interface layer **312** of the server **201**, which facilitates computation services to the client computer's **202** application, and a parallel computing library **313**, which facilitates services on the server **201**. The communication layer **311** may provide a web service allowing the client computers **202** to interact with the server **201**. The interface layer **312** includes a database interaction interface **321**, a data handler interface **322**, a task handler interface **323** and a resource handler interface **324**. For the purpose illustrating the present invention, the services of the parallel computing library **313** exemplifies herein includes equity VAR **331**, interest rate VAR **332**, Monte Carlo **333**, supporting functions **334** such as text sort/searches, and etc., which are to be processed by the processing units **350**.

[0028] The server **201** computes the data stored on the databases **203**, which can be a local database or a remote database as input source. The client computer **202**

accesses the services available on the server **201** through the web service, which also provide a user interface (UI) **351**.

[0029] **FIG. 4** is a block diagram illustrating the operational flow of the computing network in accordance with one embodiment of the present invention. As shown, a user **401** uses the client computer **202** to access services on the server **201** through the user interface available from the web service from the server **201**. The client computer **202** consists of user interaction components **402**, which include server interaction component **403**, a display **404**, and user data location and server configuration component **405**. The user data location and server configuration component **405** allows the user **401** to configure the server **201** to acquire computation services therefrom, and define the raw data location. Preliminary processing can be done at the client computer's **202** side to place the data in appropriate data form. Such preliminary processing is done through a data locator **406**, a raw data parser **407**, a raw data processor **408** and data loader **409**. It is readily understood to a skilled person that the data locator **406** allows user to select input data source, which can be local or remote data source. Local data source includes files and data stored on the client computer **202** storage while the remote data source may be references such as hyperlink, to a remote data source. The raw data parser **407** and the raw data processor **408** processes and converts the input data into a suitable format for processing by the server **201**. The processed data from the client computer **202** are transmitted to store as part of the database for processing by the server **201**.

[0030] The server **201** on the other hand consists of a main manager **451**, a data handler **452**, an analyzer **453** and a computation handler **454**. The main manager **451** handles the overall processing flow on the server. The main manager **451** triggers the

data handler 452 by passing data pointer to input/output extractor 461 of the data handler 452. Once data is extracted from database 203, a data preparation unit 462 of the data handler 452 provides a first level data processing for efficient data representation or standard format to support next level of analysis and to make eventual data processing more efficient. The first level data processing includes creating mapping tables, and identifying unique fields. The input/output data extractor 461 also decouples results and formats the results back to desired format to suite respective database table format. The analyzer 453 determines the input data types, algorithms to execute and the resources to use to achieve desired output. Once analysis is completed at the analyzer 453, main manager 451 is triggered to passed the analyzed results to an execution unit 471 of the compute handler 454 for handling the computations through the processing units under the parallel computing. The analyzed results are also stored on a server database 455. The compute handler 454 is adapted primarily to receive resources information, the requirements information, required operational instructions from the main manager and handle processing accordingly. The compute handler 454 comprises a resource configurator and selector 472, a data processing unit 473 and an algorithm configurator/selector 474. The resource configurator and selector 472 is a unit that determines the computational resources available on the server 201 and select an appropriate path accordingly. The data processing unit 473 determines the type and possibly the amount of data to be processed to manage the data to be processed by the respective processors. The data processing unit 473 may partition or segment the data, for example, segmenting the data for streaming based on main memory, local memory and register memory. Once execution done, pass back to data processing unit to merge back the partial results. The algorithm configurator and selector 474 is an algorithm functional unit. It manages and prepares algorithm for execution (i.e. initializes,

prepares hardware), partitions algorithm sequential, parallel, loops, and recursive – just the execution part. Considering the resources information, data to be processed and the algorithm required for processing data, a selected path can be established, the data will be channeled according for execution. When the path is not available, feedback is given
5 to analyzer 453.

[0031] The compute handler 454 further comprises function controllers 475, 476, 477 for the respective CPU, GPU and FPGA, for executes instruction from the execution unit based on the selected path. Each controller may be adapted to control one or more (1...n) processing engines, where available.

10 [0032] FIG. 5 exemplifies an execution flow in accordance with one embodiment of the present invention. Example is shown for a possible execution path selection for a given task that is to sort certain set of data, i.e. the request from user is from data set perspective at step 501. The request from user to sort certain set of data includes providing parameter field and orientation (i.e. ascending/descending), and the
15 server shall compute accordingly and provide with a sorted results. The users need not to be aware of the path to select as it will be determined automatically. Once request is made to sort data, the input data will be analyzed first at step 502. The input data includes parameters for the data set which will be used for heuristic rules matching are data field size 511, data field type 512 and number of records requires sorting 513.
20 Available algorithm will then be classified at step 504 based on optimized performance and execution capability within execution environment. In this example, three algorithms are available: Algorithm A1, A2 and A3.

[0033] Algorithm **A1** is provided for sorting text data type using an index to the record and the required string field data; Algorithm **A2** is provided for sorting float or integer data type using an index and the required integer or float field data; and Algorithm **A3** is provided for sorting text data type by splitting the input data into
5 different chunks to be processed independently using an index to the record and the required string field data.

[0034] The processing resource status will further be determined at a resource analysis **506**. There are two GPUs **532**, **534** available in this given example, one capable of supporting up to 2GB of data and the other capable of supporting up to 4GB
10 of data.

[0035] There are three possible paths to execute. The rules to select execution path for this example will be:

[0036] EP1 - When the user requires to sort data type integer, and the size of data to sort (i.e. field size (p) X number of records to sort (q)) is less than 2 GB of data,
15 thus it can be fitted to either one of the GPUs **532**, **534**, and the Algorithm **A2** shall be used for processing. Based on the resources analysis, the execution path will be utilizing GPU **532** for optimal result.

[0037] EP2 - When the user requires to sort data type text, and the size of data to sort (i.e. field size (p) X number of records to sort (q)) is more than 2 GB of data but
20 less than 4 GB, thus it can be handled only by the GPU **534** as GPU **532** does not have sufficient resource to handle this request. Algorithm **A1** shall be used in this case.

[0038] EP3 - When the user requires to sort data type text, and the size of data to sort (i.e. field size (p) X number of records to sort (q)) is more than 5 GB of data but less than 6 GB, both GPUs 532, 534 will be utilized in parallel to process the request. Algorithm A3 will be utilized to segment the data between the two GPUs for parallel processing.

[0039] When the conditions are not met to provide an execution path, an error alert will be returned to the user.

[0040] FIG. 6 illustrates an initialization process in accordance with one embodiment of the present invention. At step 601, initialization parameters are acquired based on XML/file configuration 602, such as server location and customization. The configuration file that can be resided in another machine will be obtained at step 603. At step 604, server initialization will be initiated based on input parameters such as database location etc. At step 605, the data is being prepared at the server under the parallel computing environment.

[0041] FIG. 7 illustrates a processing flow of the analyzer 453 of FIG. 4 in accordance with one embodiment of the present invention. The analyzer comprises a data set analyzer 702 that analyses data set parameters, an algorithm preparation unit 704 that classifies and segregates available algorithms, a compute resource analyzer 706 that analyses computing resource 750 of the respective processing units, and a resource selection and configuration unit 708 that selects an optimal execution path for computing and processing the data. As shown in FIG. 7, the data set analyzer 702 received input 712 from the data preparation unit 462 of FIG. 4. During the path selections, the resource selection and configuration unit may feedback to the data set

analyzer and the data may be reformatted at the data set analyzer **702** when the selected path is not available. It is to be noted that the present embodiment provides a rule-based selection and rules are generated based on historical experience. When a rule is selected and the data set is formatted accordingly, it will be executed. However, the
5 selected configuration may be put in queue at the processor, and when a particular kernel (such as GPU execution unit) gets overloaded due to the runtime requirements (such as stack overuse), the selected configuration may fail to be executed, even though optimal configuration is selected after checking the available resource. The feedback to reformat the data set can be desired for new optimal configuration. Similarly, the
10 algorithm preparation unit **704** further receives sequential and parallel algorithms **714** operationally. The resource selection and configuration unit **708** further receives heuristic rules from a heuristic rules database **716**. Accordingly, the resource election and configuration unit **708** may process the results from the data set analyzer **702**, the algorithm preparation unit **704** and the compute resource analyzer **706** to select an
15 optimal execution path based on the heuristic rules. Thereafter the resource election and configuration unit **708** informs the main manager **451** of the selection outcomes at step **710**.

[0042] **FIG. 8** illustrates a diagram of the data preparation unit **462** of **FIG. 4** preparing data for classification in accordance with one embodiment of the present
20 invention. At step **801**, the data is processed by the data preparation unit **462**. The data will be grouped and when required, bit packing at the data preparation unit **462**. At step **802**, the data are being classified by a data classification unit. The data classification unit classifies data because GPU and FPGA process data in a specific data type. During the data classification, data type is checked at step **810** and data size is checked at step

820. The data types may includes, but not limited to, finance data **811**, text data **812**, integer data **813** and floating data **814**. At step **820**, the data size checking also includes checking data groupings at step **821**, verifying columns/field at step **822** and checking data alignments at step **823**.

5 **[0043]** Once the data is classified, the data is being further processed at data processing unit **473** at step **803**, then the control will be passed to resource selection & configuration unit **472** at step **804**, to get feedback if data formatting is appropriate at step **805**. The data will then be mapped to index the same at step **806** and an index mapping table will be created at step **807**. A data relationship is also mapped out at
10 step **808** and a relational mapping table is created at step **809**.

[0044] **FIG. 9** exemplifies a process flow of data formatting in accordance with one embodiment of the present invention. The illustrated diagram shows that data formatting is carried out on GPU, i.e. the resource selection and configuration unit selected GPU bit packing algorithm as the select the GPU as the execution path. Over
15 at the data handler, a huge data **901** is segmented into chunks. The data will be processed at the data classification and index mapping **902** and a specific data bit index table **904**. The data passes through the data structure analyser **906** and a data relationship map **908**, such as tree flow structure, is created. Preferably, these are carried out on the CPU. It is a one time processing to avoid moving data back and
20 forth. Accordingly, a bit packing table can be created to pack the bits based on relation and unique field bit mapping. Once the path is selected, the data is sent to compute handler for processing. And in the illustrated figure, as mentioned, a GPU is selected. The compute handler first determines if data chunks are available for handling by the GPU. Then it is being loaded onto the GPU **950**. The data will be formatter to fit for

GPU processing. The GPU **950** also receives the specific data bit index table **904**. The GPU **950** then indexes, maps, compresses/packs, and stores the newly presented data. The original chinks are deleted accordingly.

[0045] FIG. 10 illustrates a block diagram of the algorithm preparation unit **704**

5 of FIG. 7 in accordance with one embodiment of the present invention. The algorithm processing unit **704** has an algorithm classification unit **1001** for classifying the algorithm. The algorithm classification unit **1001** includes an algorithm analytical unit **1003**, a data dependency analytical unit **1004**, a performance analytical unit **1005** and a processor analytical unit **1006**.

10 **[0046]** The algorithm analytical unit **1003** further includes a functional specific algorithm unit **1011** adapted for sorting, matching and searching; a logical specific algorithm unit **1012** that provides logical functions such as recursive, procedural, serial, parallel, interactive; an implementation specific algorithm unit **1013** for handling device and conquer algorithm, and dynamic programming for implementation on sub-kernels;

15 and a complexity specific algorithm unit **1014** for handling algorithms in single and double precision. The data dependency analytical unit **1004** is adapted for string processing, initializing processing and floating point processing. The performance and processor analytical units **1005**, **1006** determine the available resources available to execute the relevant algorithms. Besides availability of the resources, the performance

20 and processor analytical units **1005**, **1006** will also determine the viability of the respective processors to handle the respective algorithms. The viability is determined based on the complexity in relation to the processor.

[0047] FIG. 11 illustrates a process flow in accordance with one embodiment of the present invention. The process starts with identifying an initial or available data-set, algorithm and resource list is made based on current available parameters at step 1101. The available parameters may include data-sets data type, size, number of records, total record size etc.; the available parameters for algorithm may include complexity, execution environment, data type processing capability etc.; the available parameters for resource may include number of processor available, global memory, local memory, registers etc. The parameters would be initial individual variable parameters within specific domain, i.e. data-set, algorithm or computing resource. At step 1102, the initial list will be ordered according to heuristic rules e.g. "select size, followed by data type". At step 1103, the ordered data set list is created based on heuristic rules from Heuristic Rule Database (HRDB). Example of the heuristic rule for data sets may include "for data type int multiply number of records by 4 to get total size" followed by "add index id of size 4 bytes per record". These ordered lists are created within specific domain based on respective rules from HRDB. At step 1104, the parameters are being checked if all parameters are matched with a specific HRDB rule. This cycle is iterated until the matching rules are rectified.

[0048] Next, the ordered parameters from the Rule Based Ordered List is selected at step 1105. At step 1106, first level matching is done to create a Rule Based Pair List between data-set and algorithm. The rules may include "for data type int algorithm should be able to support int or float data type".

[0049] At step 1107, from entries in first level matching, a second level matching is done to create a Rule Based Triplet List (RBTL) comprises matching rules for data-set, algorithm and compute resource configuration permutation based on

heuristic rules. Examples of the rules may include “for data set type int with total size less than 2GB and algorithm type int or float process compute resource capable of supporting less than or equal to 2GB data should be used.

[0050] At step **1108**, the triplet parameters from the RBTL are selected.

5 **[0051]** At step **1109**, an optimum path is selected for particular execution. Optimum path is selected based on hardcoded HRDB rules from previous execution experience. For example, a rule that defines: “for any data less than size of available FPGA and FPGA device algorithm and memory is available use FPGA device, rank it 1”. Through experience, it is recognized that FPGA is relatively faster, that form one
10 factor of the above rule. However, it is also known that when the data set is too small, CPU is suitable than the FPGA because moving data to FPGA is relatively costly. Accordingly, these two rules can be paired to formed said rule.

[0052] The step **1110** is provided to loop back to step **1109** to select the optimum path. At step **1111**, the path ranking is reviewed and checked. If the path
15 configuration is available at step **1112**, a lower rank path configuration is chosen at step **1113** and return to the step **1111**. The optimum path is ranked and verified if the selection is available at step **1112**, the information is passed to main manager at step **1114**.

[0053] In short, the Initial Data-Set List is selected to create Ordered Data-set
20 List” based on Heuristic Rules from HRDB. The same is done to create the ordered algorithm list and the ordered compute-resource list. Subsequently, a rule based ordered list is created for data-set, algorithm and the compute-resource. A rule based pair list is created to pair data-set with algorithm, data-set with compute-resource and

algorithm with compute-resource. Following that, the rule based triplet list combining the data-set, algorithm and the compute-resource is created from the pair list. The optimum configuration can be determined.

[0054] FIG. 12 illustrates execution portion for data path of one embodiment of the present invention. At the data processing unit **1201**, the data is being process by the data set configurator/selector **1203** to determine an optimal path for processing the data. Data is managed according to specific selected resource. Depending on the selected path, the data may be split or merged by a data partitioning unit **1205**. The data partitioning unit includes a specific data splitting unit **1212**, a specific data merging unit **1214**, a partial result combining unit **1216** and a final results producing unit **1218**. Through these units, the data can be split or merge as required, and similarly, the partial results can be combined and formed as one final results accordingly. The processed data is then channeled to the respective handler, which includes a GPU data handler **1222**, a CPU data handler **1224** and FPGA data handler **1226**, for processing. In the case of GPU, the data is to be segments into GPU streaming data **1232**, GPU global data **1234**, GPU local data **1236** and GPU register data **1238** for respectively placed on the GPU registers, local storage or global storage for processing. In the case of CPU, the data will be segmented for handling on the CPU RAM **1242** and I/O or hard disk drive **1244**.

[0055] FIG. 13 illustrates a flow diagram of the computation resource selector in accordance with one embodiment of the present invention. At step **1301**, a trigger is received for a specific computation resource configuration. At step **1302**, the compute-resource configuration and selection is triggered. At step **1303**, the input data set will be matched to perform specific computation resource configuration and this configuration will be selected. At step **1304**, if a matching path is available, the path

will be selected for execution at step 1306. At step 1305, status that the matching path not available will be returned if the path is not available.

[0056] FIG. 14 illustrates a process flow for algorithm selector in accordance with one embodiment of the present invention. At step 1401, a trigger is received for a specific algorithm configuration. At step 1402, the algorithm configuration and selection is triggered. At step 1403, the input data set will be matched to perform specific algorithm configuration and this configuration will be selected. At step 1404, if a matching path is available, the path will be selected for execution at step 1406. At step 1405, status that the matching path not available will be returned if path not available.

[0057] While specific embodiments have been described and illustrated, it is understood that many changes, modifications, variations, and combinations thereof could be made to the present invention without departing from the scope of the invention.

Claims

1. A method to determine an application execution path on a parallel computing environment, the method comprising:
 - analysing (502) input data-sets characteristics and data-set parameters;
 - 5 analysing (504) characteristics of algorithms and algorithm parameters;
 - determining (506) available computing resources and compute-resource parameters; and
 - selecting and configuring specific path (708) for execution based on the outcome of the aforesaid steps.
- 10 2. The method according to claim 1, wherein analysing input data-sets characteristics and data-set parameters further comprising:
 - receiving input from a data preparation unit (462); and
 - receiving feedback from a resource selection and configuration unit (472)
 - 15 adapted for the selecting and configuring the specific path.
3. The method according to claim 2, wherein receiving feedback further comprises reformatting data to suite the available resource based on a ranked criteria.
- 20 4. The method according to claim 1, wherein selecting and configuring the specific path further comprising:
 - receiving inputs (712) from data set analyser (702), algorithm analyser (704) and compute resource analyser (706);
 - receiving heuristic rules from heuristic rules database (716); and
 - 25 determining an optimal configuration selection based on the heuristic rules.
5. The method according to claim 4, further comprising:
 - identifying (1101) Available Data-Sets, Algorithm List, Compute-Resource list and make an Initial Data-set List, Algorithm-List, Compute-Resource-List;

- selecting (1102) a parameter from an Initial Data-Set List;
- adding (1103) the parameter to an Ordered Data-set List based on the heuristic rules;
- selecting a parameter from an Initial Algorithm-Set List
- 5 adding the parameter to an Ordered Algorithm-set List based on the heuristic rules;
- selecting (1102) a parameter from an Initial Compute Resource-Set List;
- adding the parameter to an Ordered Compute Resource-Set List based on the heuristic rules;
- 10 selecting (1103) a parameter from the Ordered Data-set List starting from the first parameter against other parameter listed in the Ordered Algorithms List by satisfying specific heuristic rule;
- adding a node to a Rule Based Ordered List;
- checking (1104) the parameters matching with the specific heuristic rules;
- 15 selecting (1105) the ordered parameters from rule based ordered list;
- selecting (1106) one parameter from rule based ordered list satisfying the specific heuristic rule;
- adding a paired parameter to the rule based pair list for the data-set and the algorithm;
- 20 selecting (1107) one paired parameter pair from the rule based ordered list of data-set and algorithm that satisfies a specific heuristic rule
- adding the paired parameter to rule based triplet list comprises the data-set, the algorithm and the computing resource;
- selecting (1108) a triplet parameter from rule based triplet list;
- 25 selecting (1109) an optimum configuration;
- ranking the optimum configuration based on ranking rules with specific matching from rules database;

selecting (1112) specific configuration if the specific configuration is available;
and

selecting (1113) a lower rank configuration if the optimal configuration is not available.

5

6. The method according to claim 2, wherein receiving feedback from the resource selection and configuration unit further comprises returning status if the selected path is not available.

10 7. A system for determining an application execution path on a parallel computing environment, the system comprising:

a data-set analyser (702) that operably analyses input data-set characteristics and determine data-set parameters;

15 an algorithm analyser (704) that operably analyses characteristics of algorithms and determine algorithm parameters;

a compute-resource analyser (706) that operably analyses available computing resources and determine compute-resource parameters; the

20 a resource selection and configuration unit (708) that selects and configures specific path for execution based on the outcome rendered by the data-set analyser (702), the algorithm analyser (704) and the computer-resource analyser (706).

8. The system according to claim 7, wherein the data-set analyser further comprises:

25 a data configuration selector (1203) operably selecting an appropriate data configuration based on input from resource selection and configuration executor;

an algorithm configuration selector (474) operably selecting an appropriate algorithm configuration based on input from resource selection and configuration executor; and

30 an compute-resource selector (472) operably selecting an appropriate computing resource and configuration based on input from resource selection and configuration executor.

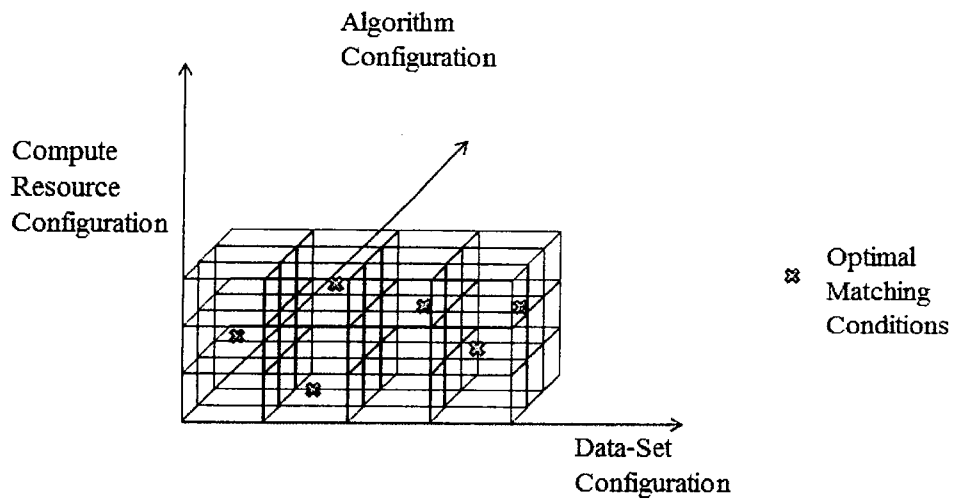


FIG. 1

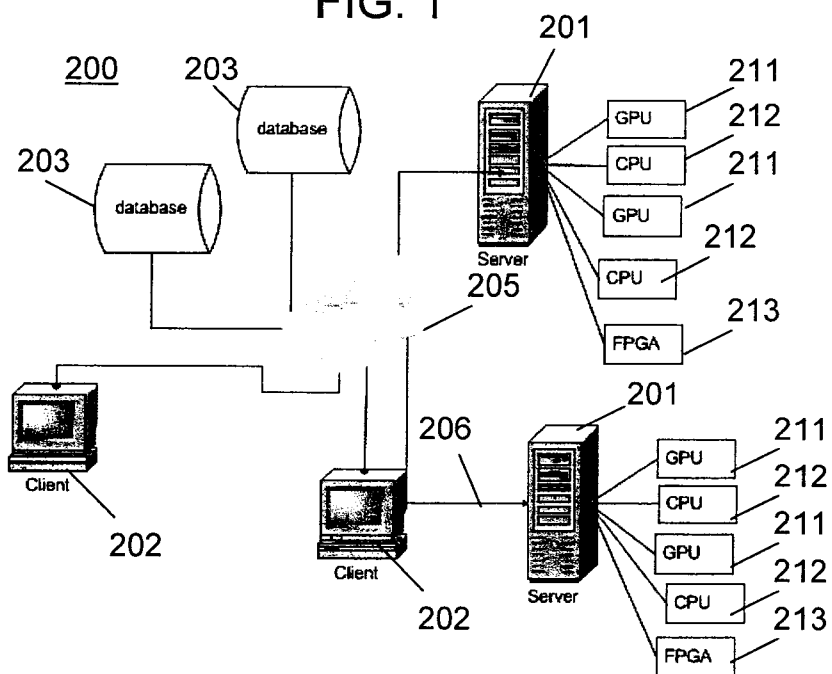


FIG. 2

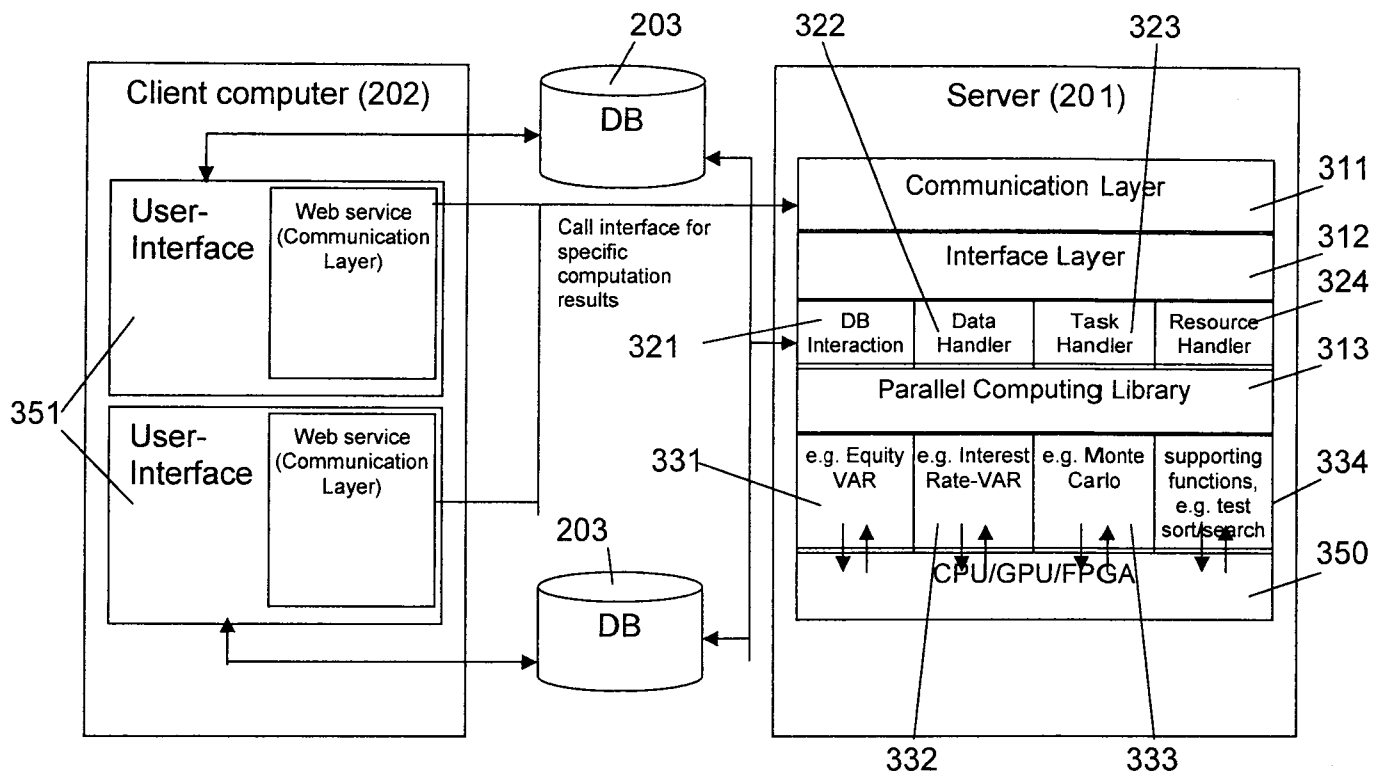


FIG. 3

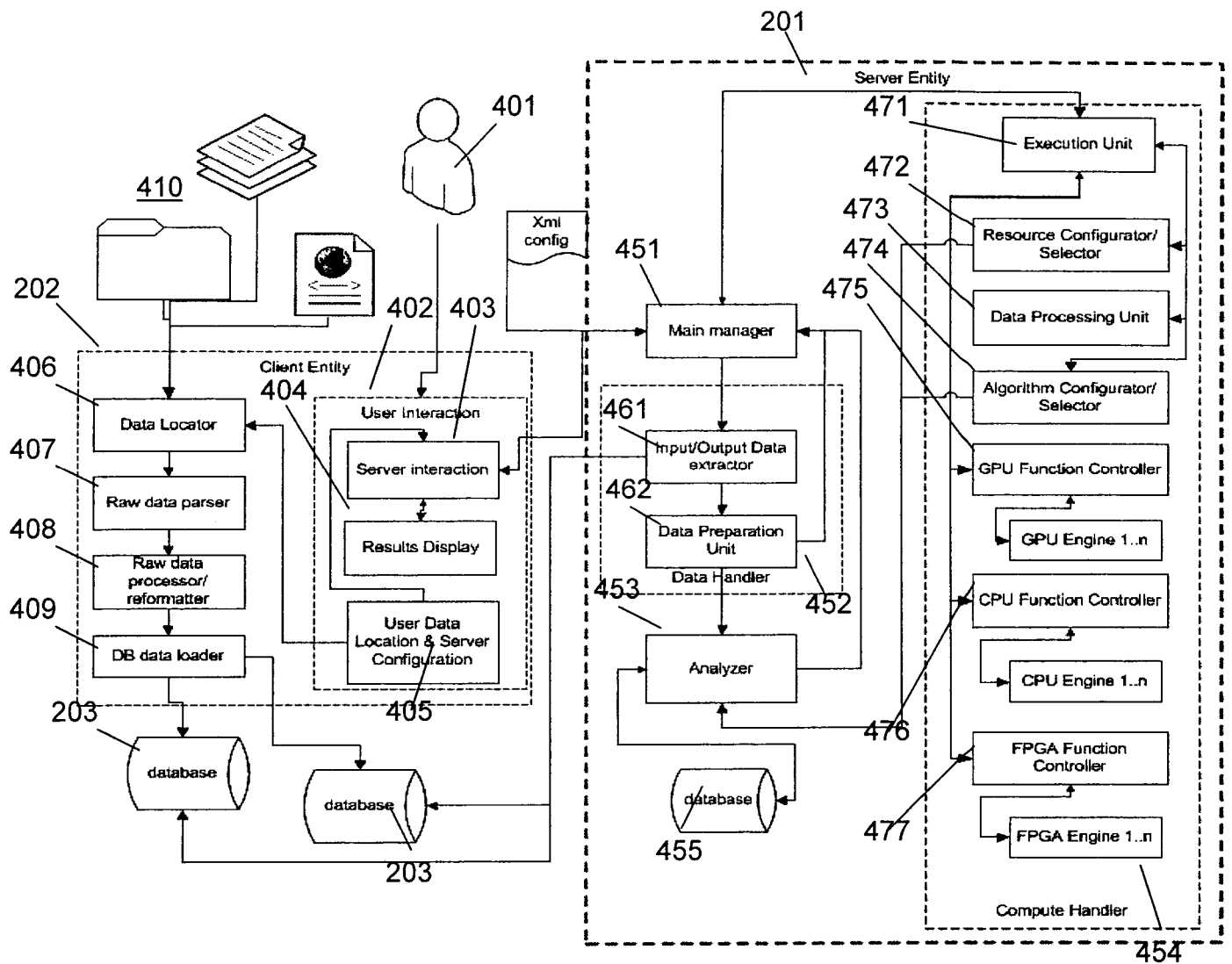


FIG. 4

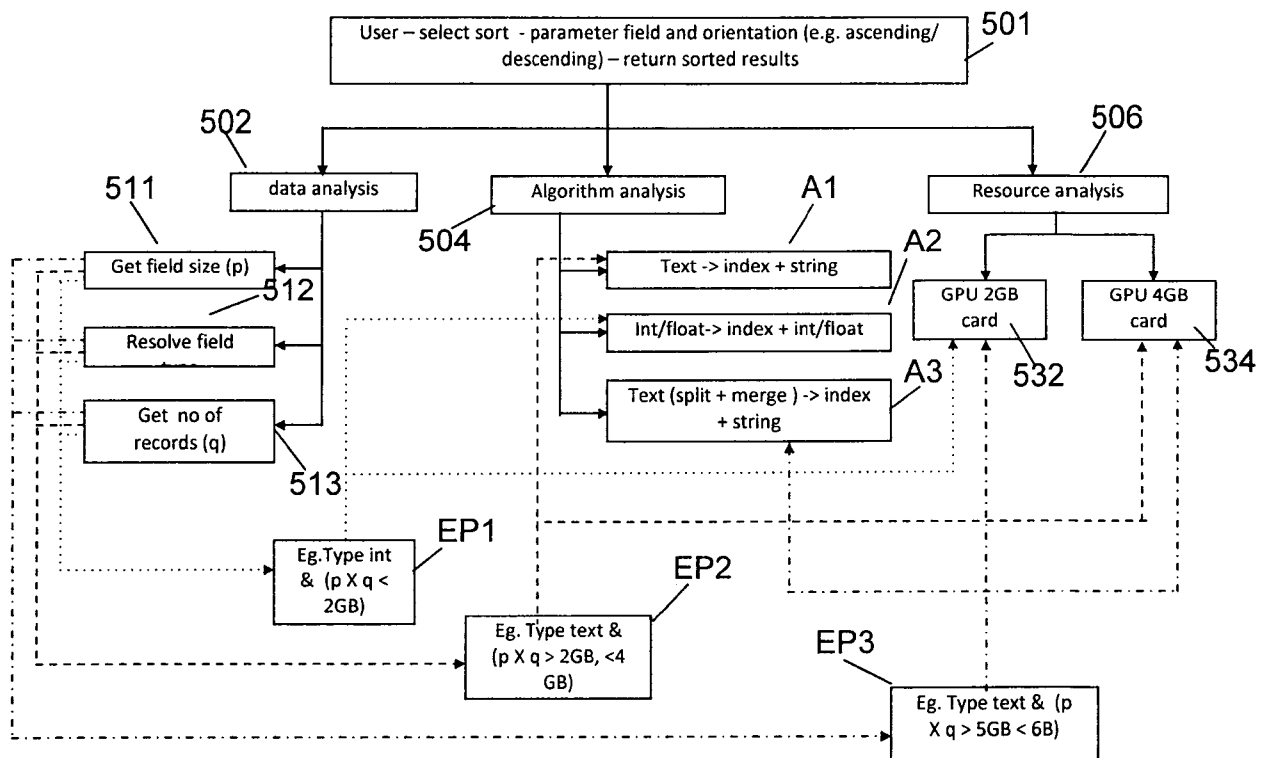


FIG. 5

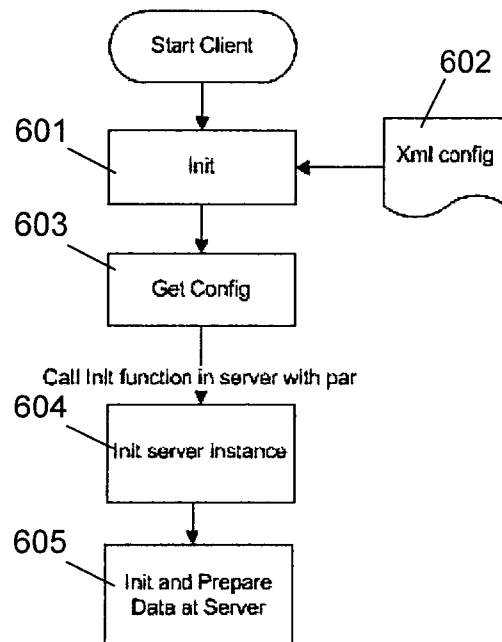


FIG. 6

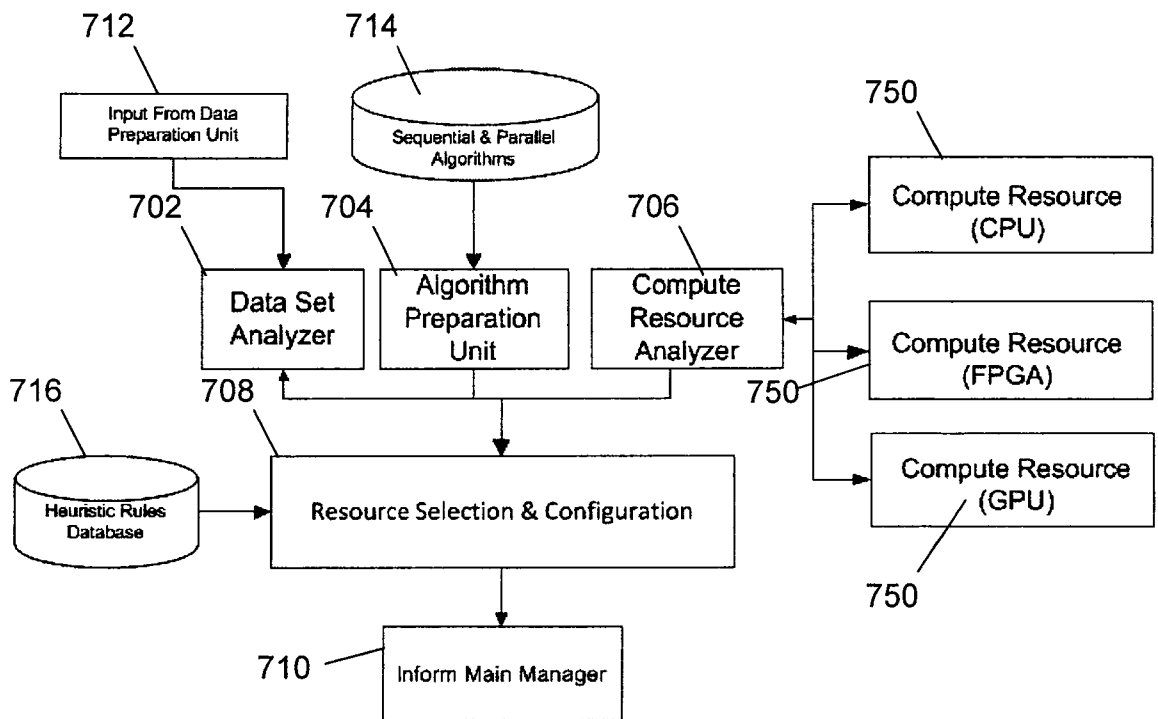


FIG. 7

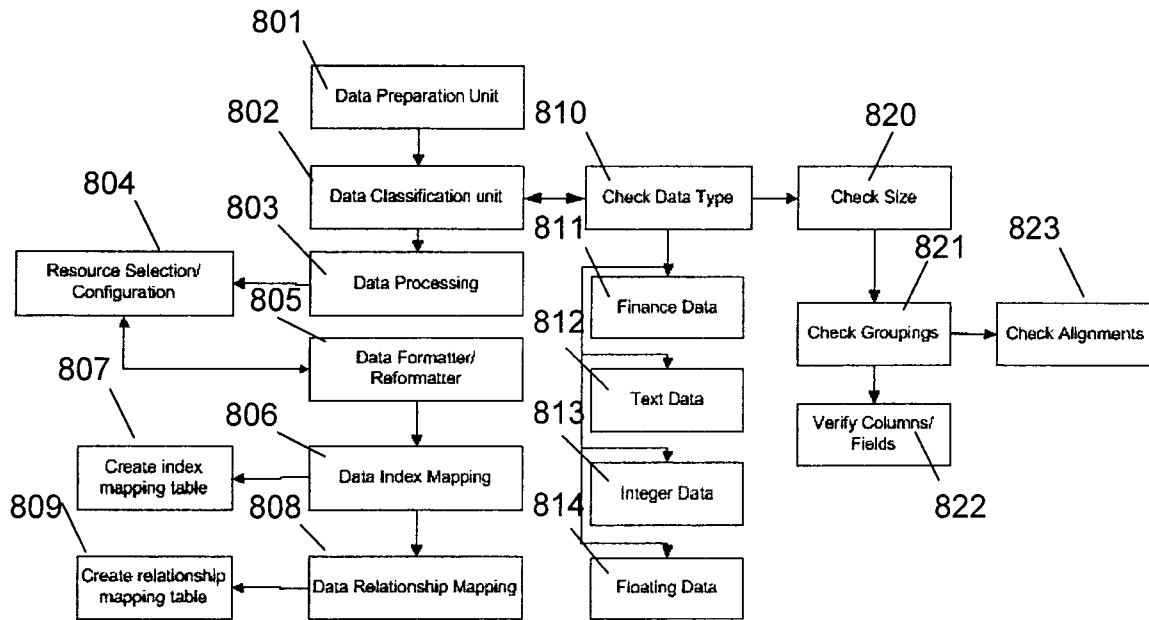


FIG. 8

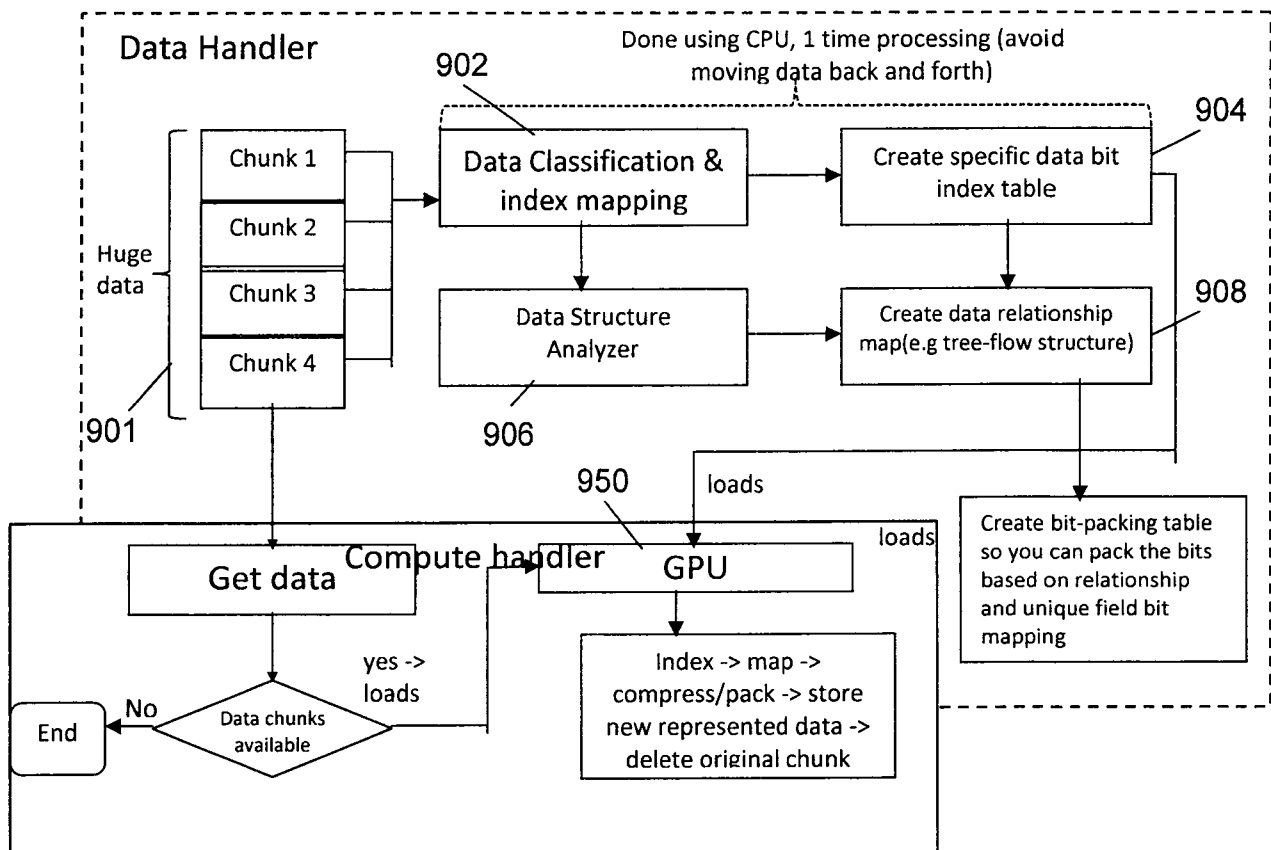


FIG. 9

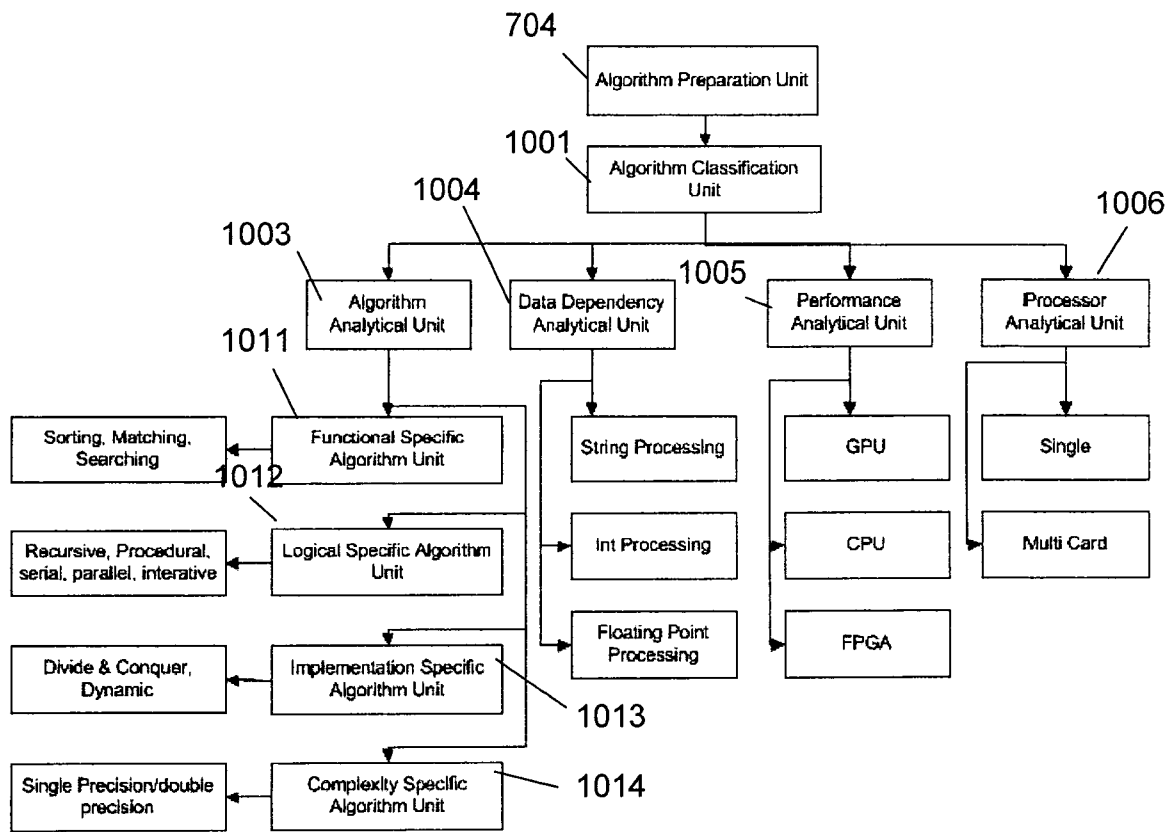


FIG. 10

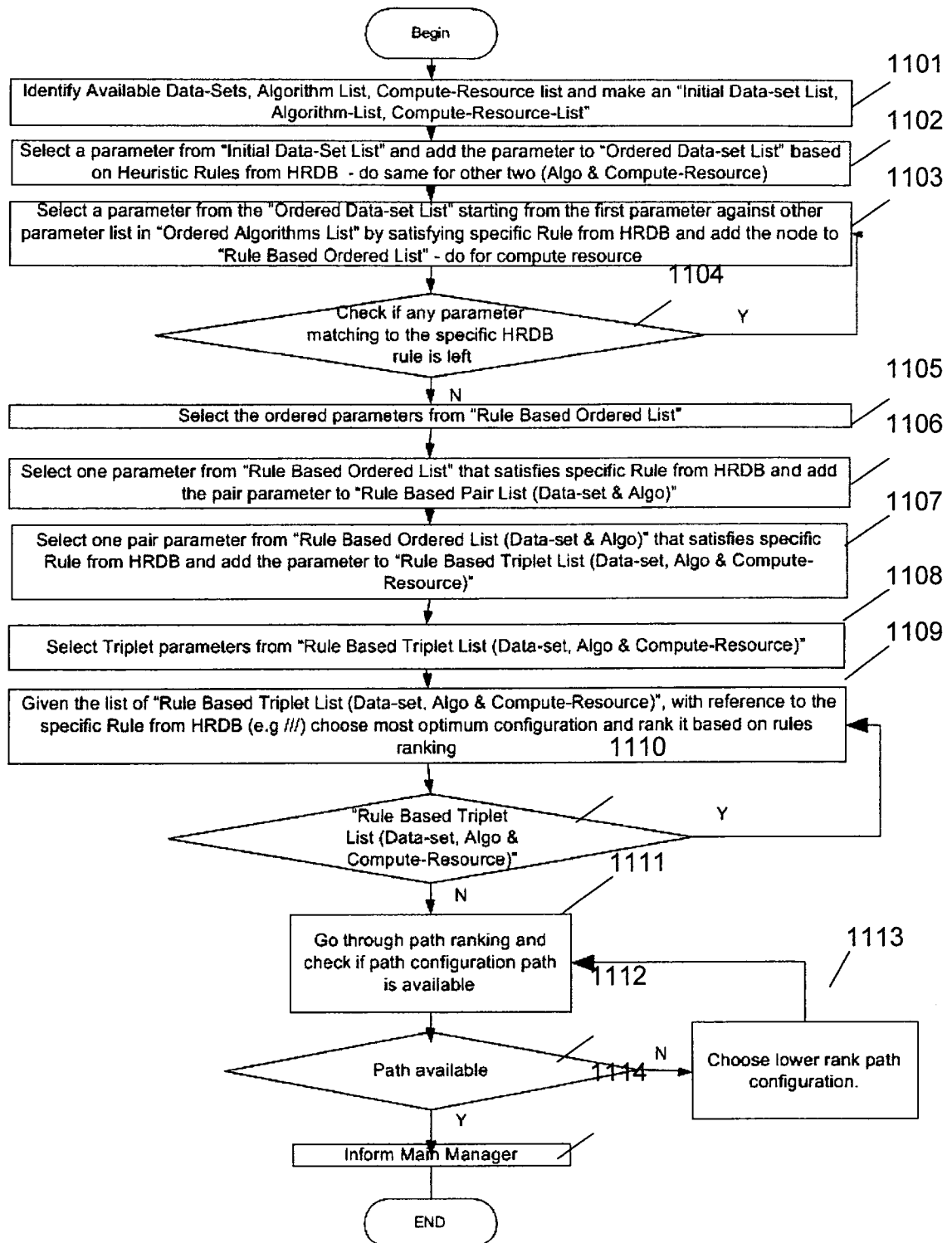


FIG. 11

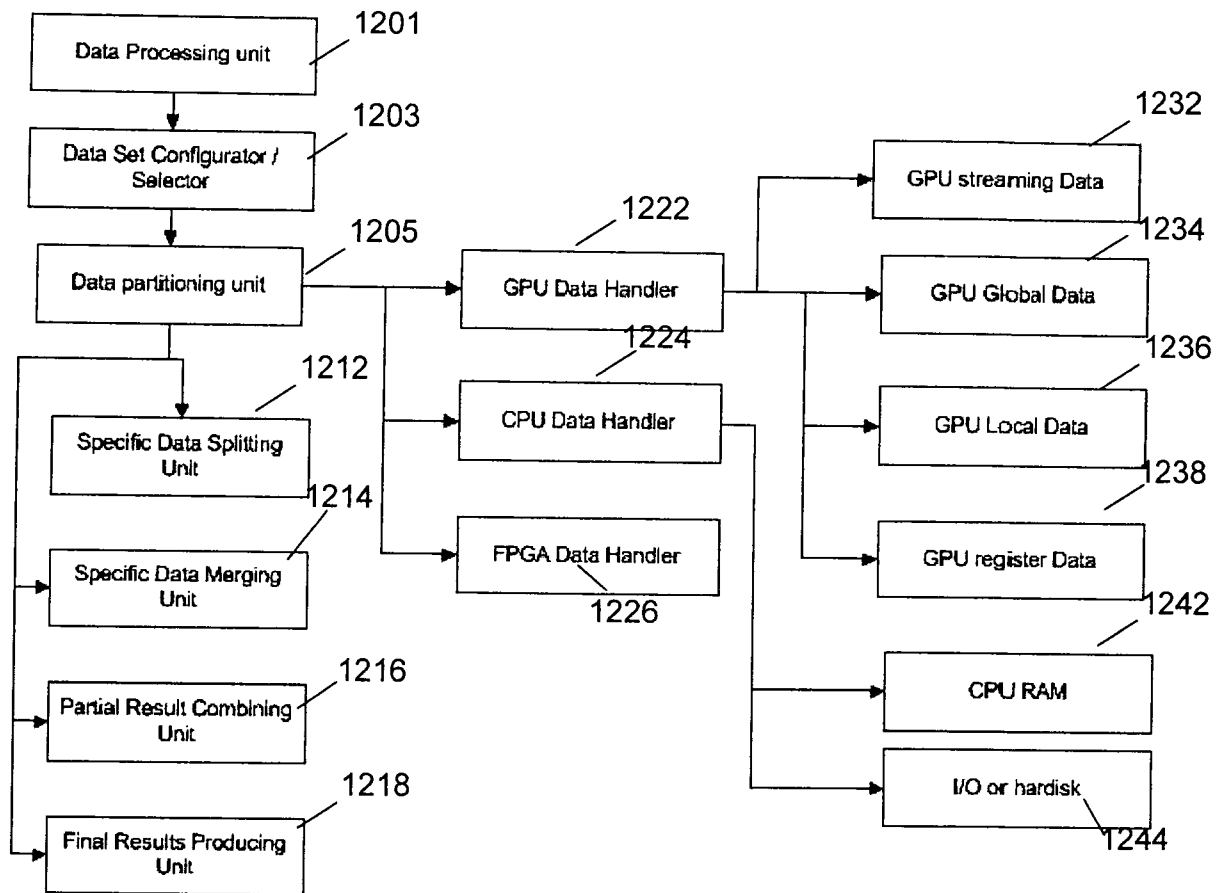


FIG. 12

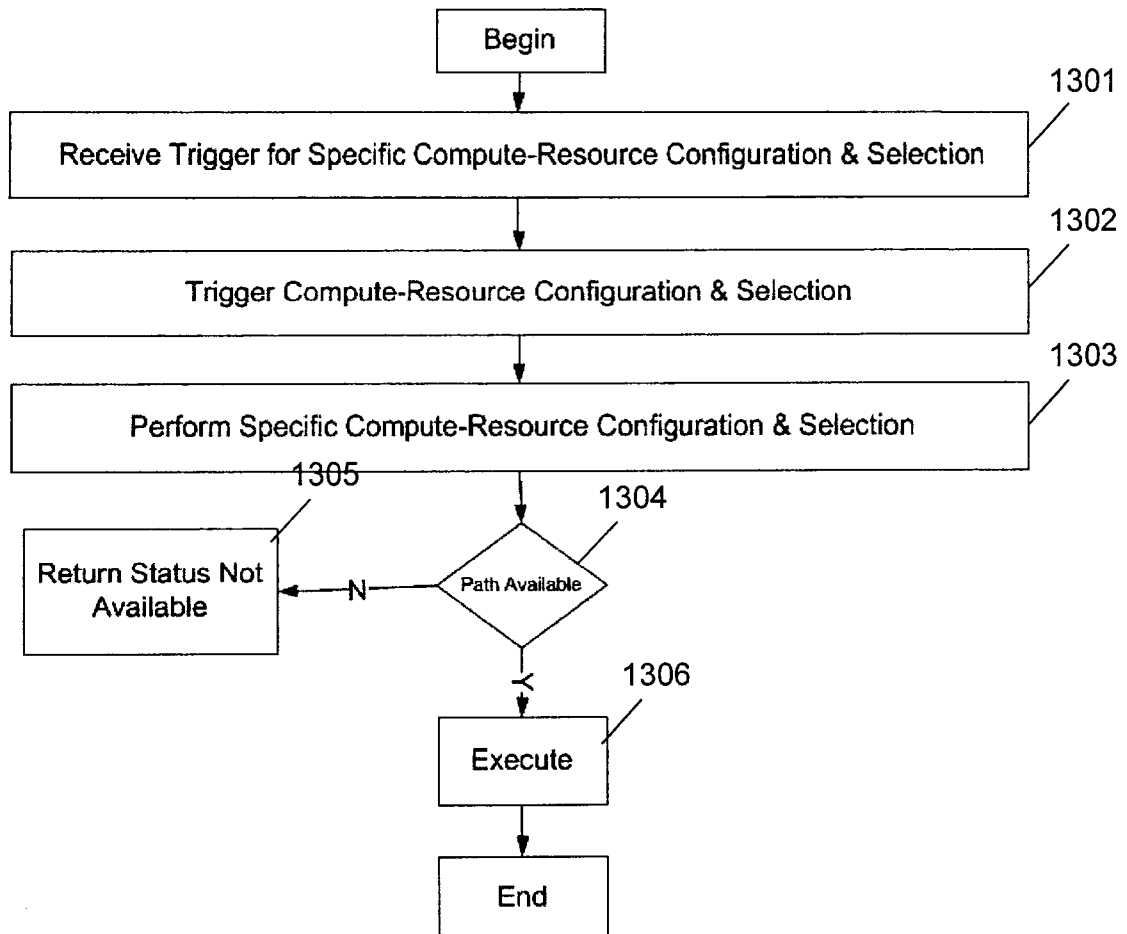


FIG. 13

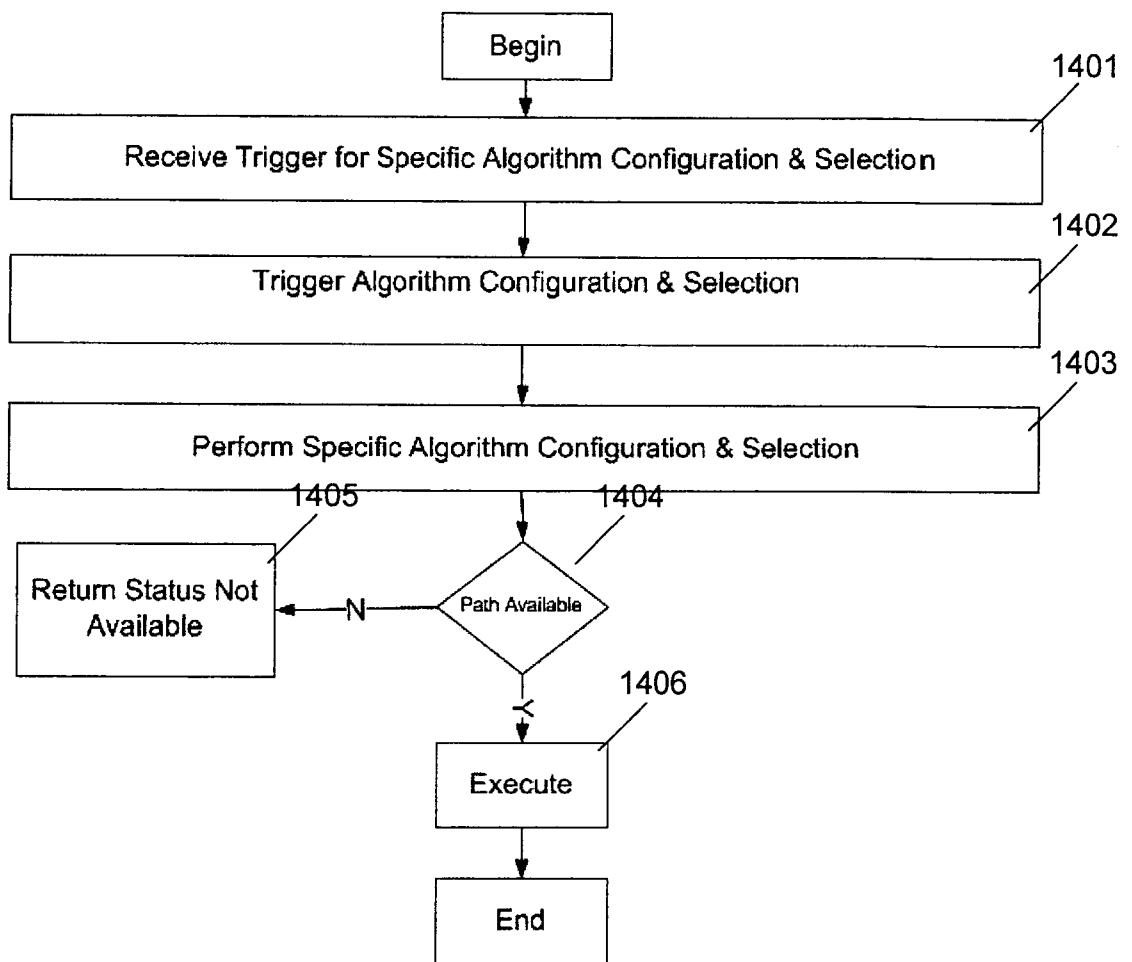


FIG. 14

INTERNATIONAL SEARCH REPORT

International application No
PCT/MY2014/000052

A. CLASSIFICATION OF SUBJECT MATTER
INV. G06F9/50
ADD.

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)
G06F

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)

EPO-Internal, WPI Data

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	US 2009/204789 A1 (GOODING THOMAS M [US] ET AL) 13 August 2009 (2009-08-13) abstract paragraph [0013] paragraph [0034] - paragraph [0035] paragraph [0052] paragraph [0058] - paragraph [0059] paragraph [0064] paragraph [0067] - paragraph [0069] -----	1-8
A	US 6 959 372 B1 (HOBSON RICHARD F [CA] ET AL) 25 October 2005 (2005-10-25) abstract paragraph [0007] - paragraph [0012] -----	1-8



Further documents are listed in the continuation of Box C.



See patent family annex.

* Special categories of cited documents :

"A" document defining the general state of the art which is not considered to be of particular relevance

"E" earlier application or patent but published on or after the international filing date

"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)

"O" document referring to an oral disclosure, use, exhibition or other means

"P" document published prior to the international filing date but later than the priority date claimed

"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention

"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone

"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art

"&" document member of the same patent family

Date of the actual completion of the international search

15 August 2014

Date of mailing of the international search report

21/08/2014

Name and mailing address of the ISA/

European Patent Office, P.B. 5818 Patentlaan 2
NL - 2280 HV Rijswijk
Tel. (+31-70) 340-2040,
Fax: (+31-70) 340-3016

Authorized officer

Archontopoulos, E

INTERNATIONAL SEARCH REPORT

Information on patent family members

International application No

PCT/MY2014/000052

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
US 2009204789	A1	13-08-2009	NONE

US 6959372	B1	25-10-2005	US 6959372 B1 25-10-2005
			US 2006129777 A1 15-06-2006
			US 2007113038 A1 17-05-2007
			US 2011047354 A1 24-02-2011
