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

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
4 February 2016 (04.02.2016)

WIPO | PCT

(10) International Publication Number
WO 2016/018352 A1

(51) International Patent Classification:
G06F 9/46 (2006.01)

(21) International Application Number:
PCT/US2014/049101

(22) International Filing Date:
31 July 2014 (31.07.2014)

(25) Filing Language: English

(26) Publication Language: English

(71) Applicant: **HEWLETT-PACKARD DEVELOPMENT COMPANY, LP** [US/US]; 11445 Compaq Center Drive W, Houston, Texas 77070 (US).

(72) Inventor: **CHERKASOVA, Ludmila**; 1501 Page Mill Rd., Palo Alto, California 94304-1100 (US).

(74) Agents: **KUIPERS, Luke W.** et al.; Hewlett-Packard Company, Intellectual Property Administration, 3404 E. Harmony Road, Mail Stop 35, Fort Collins, Colorado 80528-9599 (US).

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

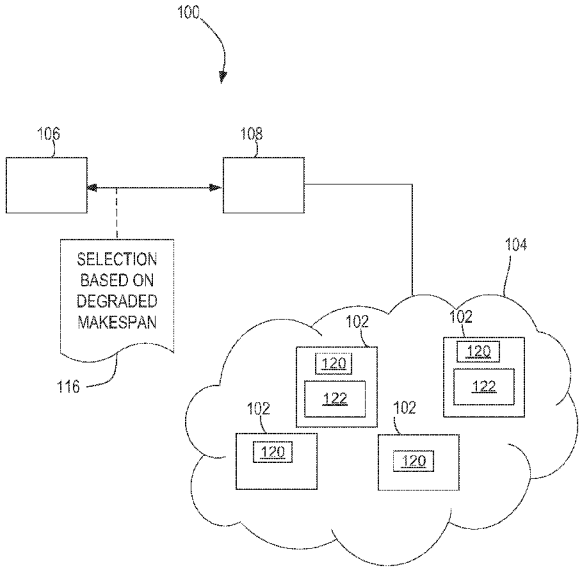

FIG. 1

(57) Abstract: A method, system, and computer-readable storage device for selecting a platform configuration in light of a degraded makespan is described herein. A job profile of a prospective job, a normal makespan goal, and a degraded makespan goal may be obtained. The job profile may include a job trace summary. A simulation result of the prospective job may be generated based on a first simulation of the job trace summary on a platform configuration and a second simulation of the job trace summary on a degraded version of the platform configuration. The simulation result may include a predicted normal makespan and a predicated degraded makespan. The platform configuration may then be selected. In some cases the platform configuration may be selected via a purchasing option sent to a tenant.

WO 2016/018352 A1

Declarations under Rule 4.17:

— *as to the identity of the inventor (Rule 4.17(i))*

Published:

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

PLATFORM CONFIGURATION SELECTION BASED ON A DEGRADED MAKESPAN

Background

[0001] A cloud infrastructure can include various resources, including computing resources, storage resources, and/or communication resources, that can be rented by customers (also referred to as tenants) of the provider of the cloud infrastructure. By using the resources of the cloud infrastructure, a tenant does not have to deploy the tenant's own resources for implementing a particular platform for performing target operations. Instead, the tenant can pay the provider of the cloud infrastructure for resources that are used by the tenant. The "pay-as-you-go" arrangement of using resources of the cloud infrastructure provides an attractive and cost-efficient option for tenants that do not desire to make substantial up-front investments in infrastructure.

Brief Description Of The Drawings

[0002] The following description illustrates various examples with reference to the following figures:

[0003] FIG. 1 is a schematic diagram of a cloud infrastructure service, according to an example;

[0004] FIG. 2 is a block diagram illustrating example components of the evaluation system, according to some implementations;

[0005] FIG. 3 is a flowchart that illustrates a method for selecting a platform configuration for a job or a workload of jobs, in accordance to an example;

[0006] FIG. 4 is a flowchart that illustrates the operation of generating simulation results of FIG. 3 in greater detail, according to an example implementation;

[0007] FIGS. 5 and 6A-6B illustrate execution orders of jobs, according to some examples;

[0008] FIG. 7 is a diagram which shows an scheduling queue that includes a number of entries in which jobs of the set $\mathcal{J}$ of jobs are to be placed; and

[0009] FIG. 8 is a block diagram of a computing device capable of selecting a platform configuration in light of a failure case, according to one example.

Detailed Description

[0010] A cloud infrastructure can include various different types of computing resources that can be utilized by or otherwise provisioned to a tenant for deploying a computing platform for processing a workload of a tenant. A tenant can refer to an individual or an enterprise (*e.g.,* a business concern, an educational organization, or a government agency). The computing platform (e.g., the computing resources) of the cloud infrastructure are available and accessible by the tenant over a network, such as the Internet, a local area network (LAN), a wide area network (WAN), a virtual private network (VPN), and so forth.

[0011] Computing resources can include computing nodes, where a "computing node" can refer to a computer, a collection of computers, a processor, or a collection of processors. In some cases, computing resources can be provisioned to a tenant according to determinable units offered by the cloud infrastructure system. For example, in some implementations, computing resources can be categorized into computing resources according to processing capacity of different sizes. As an example, computing resources can be provisioned as virtual machines (formed of machine-readable instructions) that emulate a physical machine. A virtual machine can execute an operating system and applications like a physical machine. Multiple virtual machines can be hosted by a physical machine, and these multiple virtual machines can share the physical resources of the physical machine. Virtual machines can be offered according to different sizes, such as small, medium, and large. A small virtual machine has a processing capacity that is less than the processing capacity of a medium virtual machine, which in turn has less processing capacity than a large virtual machine. As examples, a large virtual machine can have twice the processing capacity of a medium virtual machine, and a medium virtual machine can have twice the processing capacity of a small virtual machine. A

processing capacity of a virtual machine can refer to a central processing unit (CPU) and memory capacity, for example.

[0012] A provider of a cloud infrastructure can charge different prices for use of different resources. For example, the provider can charge a higher price for a large virtual machine, a medium price for a medium virtual machine, and a lower price for a small virtual machine. In a more specific example, the provider can charge a price for the large virtual machine that is twice the price of the medium virtual machine. Similarly, the price of the medium virtual machine can be twice the price of a small virtual machine. Note also that the price charged for a platform configuration can also depend on the amount of time that resources of the platform configuration are used by a tenant.

[0013] Also, the price charged by a provider to a tenant can vary based on a cluster size by the tenant. If the tenant selects a larger number of virtual machines to include in a cluster, then the cloud infrastructure provider may charge a higher price to the tenant, such as on a per virtual machine basis.

[0014] The configuration of computing resources selected by a tenant, such as a processor sizes, virtual machines, computer nodes, network bandwidth, storage capacity, and the like may be referred to as a platform configuration. The choice of the platform configuration can impact the cost or service level of processing a workload.

[0015] A tenant is thus faced with a variety of choices with respect to resources available in the cloud infrastructure, where the different choices are associated with different prices. Intuitively, according to examples discussed above, it may seem that a large virtual machine can execute a workload twice as fast as a medium virtual machine, which in turn can execute a workload twice as fast as a small virtual machine. Similarly, it may seem that a 40-node cluster can execute a workload four times as fast as a 10-node cluster.

[0016] As an example, the provider of the cloud infrastructure may charge the same price to a tenant for the following two platform configurations: (1) a 40-node

cluster that uses 40 small virtual machines; or (2) a 10-node cluster using 10 large virtual machines. Although it may seem that either platform configuration (1) or (2) may execute a workload of a tenant with the same performance, in actuality, the performance of the workload may differ on platform configurations (1) and (2). The difference in performance of a workload by the different platform configurations may be due to constraints associated with network bandwidth and persistent storage capacity in each platform configuration. A network bandwidth can refer to the available communication bandwidth for performing communications among computing nodes. A persistent storage capacity can refer to the storage capacity available in a persistent storage subsystem.

[0017] Increasing the number of computing nodes and the number of virtual machines may not lead to a corresponding increase in persistent storage capacity and network bandwidth. Accordingly, a workload that involves a larger amount of network communications would have a poorer performance in a platform configuration that distributes the workload across a larger number of computing nodes and virtual machines, for example. Since the price charged to a tenant may depend in part on an amount of time the resources of cloud infrastructure are reserved for use by the tenant, it may be beneficial to select a platform configuration that reduces the amount of time that resources of the cloud infrastructure are reserved for use by the tenant.

[0018] Selecting a platform configuration in a cloud infrastructure can become even more challenging when a performance objective is to be achieved. For example, one performance objective may be to reduce (or minimize) the overall completion time (referred to as a "makespan") of the workload. A makespan may be measured from the time a workload begins to when the workload is completed.

[0019] In some cases, a tenant may define a performance objective for cases where a failure occurs within the cloud infrastructure hosted by the cloud provider. Such may occur, for example, when the tenant executes a MapReduce cluster on virtual machines instantiated on the cloud infrastructure but one of those virtual machines fails. In some cases, tenants may have difficulty assessing how a given

platform configuration may operate in light of such a failure. Such may be the case because a failure of a large instance of a virtual machine that is a node within a Hadoop cluster might have a more severe performance impact compared to a loss of a small instance of a virtual machine in a Hadoop cluster.

[0020] In accordance with some implementations, techniques or mechanisms are provided to allow for selection of a platform configuration, from among multiple platform configurations, that is able to satisfy an objective of a tenant of a cloud infrastructure. For example, according to an example implementation, obtain a job profile of a prospective job, a normal makespan goal, and a degraded makespan goal. The job profile may include a job trace summary. A simulation result of the prospective job may be generated based on a first simulation of the job trace summary on a platform configuration and a second simulation of the job trace summary on a degraded version of the platform configuration. The simulation result may include a predicted normal makespan and a predicated degraded makespan. The platform configuration may then be selected. In some cases the platform configuration may be selected via a purchasing option sent to a tenant.

[0021] In another example, job profiles of prospective jobs in a workload, a normal makespan goal, and a degraded makespan goal may be obtained. The job profiles may include job trace summaries. A schedule of the workload may then be generated using the job trace summaries and a platform configuration. A simulation result of an execution of the workload according to the schedule and the platform configuration may be aggregated with a simulation result of another execution of the workload according to the schedule and a degraded version of the platform configuration. The aggregated simulation result including a predicted normal makespan and a predicated degraded makespan. The platform configuration may be selected based on the predicted normal makespan satisfying the normal makespan goal and the predicted degraded makespan satisfying the degraded makespan goal. Computing resources from a cloud infrastructure system may then be provisioned according to the selected platform configuration.

[0022] FIG. 1 is a schematic diagram of a cloud infrastructure service 100, according to an example. The cloud infrastructure service 100 includes a tenant system 106, an evaluation system 108, and a cloud infrastructure system 104. The cloud infrastructure system 104 can include computing nodes 102 communicatively coupled by a network. As described above, a computing node may be a computer, a collection of computers, a processor, or a collection of processors. A provider of the cloud infrastructure system 104 may partition computing resources from the computing nodes 102 and rent out those resources to the tenant system 106. For example, as shown in FIG. 1, each of the computing nodes 102 includes a number of virtual machines (VMs), such as virtual machines 120, 122. The virtual machines 120, 122 may be a partitioning of computing resource that is dedicated for a given tenant. The virtual machines may differ according to the underlying computer resources of the computing node that hosts the virtual machine. For example, the virtual machines 120 may be allocated a given amount of processor bandwidth, storage, or any other compute resource, while the virtual machines 122 may be allocated a different amount of processor bandwidth, storage, or any other compute resource.

[0023] The tenant system 106 is communicatively coupled to the cloud infrastructure system 104. A tenant system can refer to a computer or collection of computers associated with a tenant. Through the tenant system 106, a tenant can submit a request to the cloud infrastructure service 100 to rent the resources of the cloud infrastructure service 100 through, for example, virtual machines executing on the computing nodes 102. A request for resources of the cloud infrastructure service 100 can be submitted by a tenant system 106 to an evaluation system 108 of the cloud infrastructure service 100. The request can identify a workload of jobs to be performed, and can also specify target makespans (e.g., a normal case makespan or a degraded case makespan) and/or a cost the tenant is willing to spend on executing a workload.

[0024] The evaluation system 108 may be a computer system that interfaces with the tenant system 106 and the cloud infrastructure system 104. The evaluation system 108 may be a computer system that is configured to select a platform

configuration from among multiple platform configurations that can be hosted on the cloud infrastructure system 104 based on a degraded makespan target. In some cases, a selection of a platform configuration can be presented in a purchasing option 116 that the tenant can use to purchase computing resources from the cloud infrastructure system 104. The purchasing option 116 may include a selection of a platform configuration where the selection is based on a degraded makespan. Example methods and operations for selecting a platform configuration is discussed in greater detail below. Once the platform configuration is selected by the platform configuration selector 116 (as may be initiated by a tenant through the purchasing option 116), the selected resources that are part of the selected platform configuration (including a cluster of computing nodes 102 of a given cluster size, and virtual machines of a given size) are made accessible to the tenant system 106 to perform a workload of the tenant system 106.

[0025] By way of example and not limitation, the tenant system 106 may rent computing resources from the cloud infrastructure system 104 to host or otherwise execute a workload that includes MapReduce jobs. Before discussing further aspects of examples of the cloud infrastructure service 100, MapReduce is now discussed. MapReduce jobs operate according to a MapReduce framework that provides for parallel processing of large amounts of data in a distributed arrangement of machines, such as virtual machines 120, as one example. In a MapReduce framework, a MapReduce job is divided into multiple map tasks and multiple reduce tasks, which can be executed in parallel by computing nodes. The map tasks operate according to a user-defined map function, while the reduce tasks operate according to a user-defined reduce function. In operation, map tasks are used to process input data and output intermediate results. Reduce tasks take as input partitions of the intermediate results to produce outputs, based on a specified reduce function that defines the processing to be performed by the reduce tasks. More formally, in some examples, the map tasks process input key-value pairs to generate a set of intermediate key-value pairs. The reduce tasks produce an output from the intermediate key-value pairs. For example, the reduce tasks can merge the intermediate values associated with the same intermediate key.

[0026] Although reference is made to MapReduce jobs in the foregoing, it is noted that techniques or mechanisms according to some implementations can be applied to select platform configurations for workloads that include other types of jobs.

[0027] FIG. 2 is a block diagram illustrating example components of the evaluation system 108, according to some implementations. The evaluation system 108 shown in FIG. 2 includes a job tracer 210 to produce job trace summaries, a scheduler 212 to generate an execution order for jobs of a workload, a simulator 214 to simulate the execution of jobs of a workload on a candidate platform configuration that includes a given cluster of computing virtual machines executing on compute nodes, and a platform configuration selector 216 to select a platform configuration to achieve a target object.

[0028] Although FIG. 2 depicts the job tracer 210, the scheduler 212, the simulator 214, and the platform configuration selector 216 as being part of the evaluation system 108, it is noted that in other implementations, the job tracer 210, the scheduler 212, the simulator 214, or the platform configuration selector 216 can be modules of systems other than the evaluation system 108, such as the cloud infrastructure system 104.

[0029] FIG. 3 is a flowchart that illustrates a method 300 for selecting a platform configuration for a job (or a workload that includes multiple jobs), in accordance to an example. The method 300 may be performed by the modules, logic, components, or systems shown in FIGS. 1 and 2 and, accordingly, is described herein merely by way of reference thereto. It will be appreciated that the method 300 may, however, be performed on any suitable hardware.

[0030] The method 300 may begin at operation 302 when the platform configuration selector 216 obtains a job profile of a prospective job, a normal makespan goal, and a degraded makespan goal from the tenant system 106. In some cases, the job profile may include a job trace summary. A job trace summary may be data or logic that characterizes the execution properties of the jobs (or comprising tasks) that are part of the workload. For MapReduce frameworks, a job

trace summary can include data that represents a set of measured durations of map and reduce tasks of a given job on a given platform configuration.

[0031] The normal makespan goal may be data and/or logic that represents a tenant specified goal of a duration of time in which the cloud infrastructure system 104 can start and complete a job if the cloud infrastructure system 104 does not experience any faults during execution of the workload. The degraded makespan goal may be data and/or logic that represents a tenant specified goal of a duration of time in which the cloud infrastructure system 104 can start and complete a job where the cloud infrastructure system 104 experiences a fault during execution of the workload. The normal makespan goal and the degraded makespan goal may each be input supplied by a tenant.

[0032] At operation 304, the simulator 214 may generate a simulation result of the prospective job based on multiple simulations of the job trace summary, where each simulation of the job trace summary simulates an execution of the prospective job on a different version of a platform configuration. For example, the job trace summary may be simulated to execute on a version of the platform configuration that represents a normal case. In parallel, or sequentially, the job trace summary may be simulated to execute on another version of the platform configuration that represents a degraded case (e.g., where a node fails), relative to the version of the platform configuration representing the normal case. These simulations may be used to generate a predicted normal makespan and a predicated degraded makespan. To illustrate further, the simulator 214 may execute a simulation of the job on a platform configured with 20 small nodes. This platform configuration may represent a normal case platform configuration, and the simulation of the job on this platform configuration may produce a predicted normal makespan. The simulator 214 may execute another simulation of the job on a degraded version of the normal case platform configuration, such as a platform configuration specifying 19 small nodes, which may represent a single node failure of the normal case platform configuration. The simulation of the job on the degraded version of the normal case platform configuration may produce a predicted degraded makespan.

[0033] At operation 306, the platform configuration selector 216 may select a platform configuration for the tenant system 106. The platform configuration selector 216 may select the platform configuration based on the predicted normal makespan of the platform configuration satisfying the normal makespan goal and the predicted degraded makespan of the platform configuration satisfying the degraded makespan goal. The platform configuration selector 216 may communicate the selected platform configuration to the tenant system in a purchasing option (e.g., such as the purchasing option 116 in FIG. 1). In some cases, the purchasing option may be configured such that when the tenant system 106 selects or otherwise activates the purchasing option, the cloud infrastructure system 104 provisions VMs according to the platform configuration selected by the purchasing option.

[0034] Accordingly, the evaluation system 108 may provide a tenant with a comparatively simple mechanism to select a platform configuration to execute a job or a workload of jobs on a cloud infrastructure.

[0035] FIG. 4 is a flowchart that illustrates operation 304 in greater detail, according to an example implementation. In the example implementation shown in FIG. 4, the scheduler 212 may, at operation 406, generate a schedule of jobs in a workload. In some cases, to generate the schedule, the scheduler 212 uses a platform configuration 402 and a job trace summary 404. In some cases the platform configuration 402 specifies a cluster size and of a given instance type for a MapReduce cluster that will execute the workload. In some cases, properties of the platform configuration (e.g., cluster size (as may be represented by a number of virtual machines), instance type, or any other suitable type of computer resource) may be specified by the tenant, programmatically selected by the platform configuration selector 216, or the like.

[0036] The job trace summary 404 may include data or logic that characterizes the execution properties of the jobs that are part of the workload. For MapReduce frameworks, a job trace summary can include data that represents a set of measured durations of map and reduce tasks of a given job on a given platform configuration. The data or logic of the job trace summary can be created for the platform

configurations supported by the cloud infrastructure, which can differ, in some cases, by instance type (e.g. different sizes of virtual machines or physical machines) or by cluster sizes, for example. Using the job trace summary, data regarding the tasks of a job can be computed. For example, an average duration and/or maximum duration of map and reduce tasks of each job can be computed. The job trace summaries can be obtained in multiple ways, depending on implementation. For example, the job trace summaries may be obtained, from the job tracer 210: a) from the past run of this job on the corresponding platform (the job execution can be recorded on the arbitrary cluster size)], b) extracted from the sample execution of this job on the smaller dataset, or, c) interpolated by using a benchmarking approach.

[0037] In some implementations of operation 406, the scheduler 212 produces a schedule (that includes an order of execution of jobs and respective tasks) that reduces (or minimizes) an overall completion time of a given set of jobs. In some examples, a Johnson scheduling technique for identifying a schedule of concurrent jobs can be used. In general, the Johnson scheduling technique may provide a decision rule to determine an ordering of tasks that involve multiple processing stages. In other implementations, other techniques for determining a schedule of jobs can be employed. For example, the determination of an improved schedule can be accomplished using a brute-force technique, where multiple orders of jobs are considered and the order with the best or better execution time (smallest or smaller execution time) can be selected as the optimal or improved schedule.

[0038] With continued reference to FIG. 4, after the scheduler 112 generates a schedule for the workload, the simulator 214 may, at operation 408, execute a number of simulations of the schedule of jobs executing on the platform configuration 402 and variations of the platform configuration 402 that represent a degraded case. For example, the simulator 214 may execute a simulation of the schedule for the jobs on the platform configuration 402 and another simulation of the schedule for the jobs on a variation of the platform configuration 402, where the variation of the platform configuration specifies a node cluster with one less node to represent a one node failure case. As FIG. 4 shows, the simulator 214 may execute additional

simulations for other variants of the platform configuration to represent other degraded cases, such as a two node failure, a three node failure, and so on.

[0039] The results of the multiple simulations executed at operation 408 may form a data record 410. A data record may include one or more of the following fields: $(InstType, NumNodes, Sched, Makespan_{Nml}, Cost_{Nml}, Makespan_{Flt}, Cost_{Flt})$, where $InstType$ specifies an instance type (*e.g.*, a virtual machine size); $NumNodes$ specifies the cluster size (number of computing nodes in a cluster); $Sched$ specifies an order of the jobs of the workload; $Makespan_{Nml}$ specifies the predicted makespan of the workload of jobs in the normal case (no faults are present); $Cost_{Nml}$ represents the cost to the tenant to execute the jobs of the workload with the platform configuration (including the respective cluster size and instance type), where the cost can be based on the price charged to a tenant for the respective platform configuration for a given amount of time; $Makespan_{Flt}$ specifies the predicted makespan of the workload of jobs in a faulty case (e.g., one node fault); $Cost_{Flt}$ represents the cost to the tenant to execute the jobs of the workload with the platform configuration in the faulty case, where, again, the cost can be based on the price charged to a tenant for the respective platform configuration for a given amount of time.

[0040] In some cases, the operation 304 shown in FIG. 4 may iterate over different platform configurations to generate additional data records. For example, some implementations of the operation 304 may iterate over operations 406, 408 multiple times where each subsequent iteration updates the platform configuration by incrementing the size of the cluster. For each iteration of operation 406, 408, the scheduler 112 produces a new job schedule for the increased cluster size. Further, for each iteration of operation 406, 408, the simulator 214 simulates the job trace summary using the new schedule and the updated platform configuration and simulates the job trace summary using the new schedule and degraded versions of the updated platform configuration. These simulations generate predicted normal case makespans and degraded case makespans for the update platform configurations, which are added as a new data record in the search space. In this way, the operation 304 may add additional data records to the search space for

platform configurations with varying cluster sizes. The operation 304 can iterate over operations 406, 408 in this way until a stopping condition is detected. In some examples, a stopping condition can include one of the following: (1) the iterative process is stopped once cluster sizes from a predetermined range of values for a cluster size have been considered; or (2) the iterative process is stopped if an increase in cluster size does not improve the achievable makespan by greater than some specified threshold. The latter condition can happen when the cluster is large enough to accommodate concurrent execution of the jobs of the workload, and consequently, increasing the cluster size cannot improve the makespan by a substantial amount.

[0041] Aside from iterating over cluster size, the operation 304 can iterate over instance types. For example, the operations 406, 408 can be performed for another instance type (*e.g.* another size of virtual machines), which further adds data records to the search space that correlate various instance types with respective performance metrics (e.g., normal case makespan and degraded case makespans).

[0042] After the search space has been built, the platform configuration selector 216 may, at operation 412, select a data record from the search space. In some examples, the platform configuration selector 216 can be used to solve at least one of the following problems: (1) given a target makespan *T* specified by a tenant, select the platform configuration that minimizes the cost; or (2) given a target cost *C* specified by a tenant, select the platform configuration that minimizes the makespan.

[0043] To solve problem (1), the following procedure can be performed.

1) Sort the data set
$Data(J) =$
$(InstType, NumNodes, Sched, Makespan_{Nml}, Cost_{Nml}, Makespan_{Flt}, Cost_{Flt})$
by the $Makespan_{Nml}$ values in non-descending order.

2) Form a subset $Data_{Makespan_{Nml} \leq T_{Nml}}(J)$ of the data set $Data(J)$, in which the entries of the subset $Data_{Makespan_{Nml} \leq T}(J)$ satisfy $Makespan_{Nml} \leq T_{Nml}$, where T_{Nml} is a target makespan specified by a tenant. Stated

differently, the entries of the data set $Data(J)$ whose $Makespan_{Nml}$ values exceed T_{Nml} are excluded from the subset $Data_{Makespan_{Nml} \leq T_{Nml}}(J)$.

3) Sort the data set $Data_{Makespan_{Nml} \leq T_{Nml}}(J)$ by the $Makespan_{Flt}$ values in non-descending order.

4) Form a subset $Data_{Makespan_{Flt} \leq T_{Flt}}(J)$ of the data set $Data_{Makespan_{Nml} \leq T_{Nml}}(J)$, in which the entries of the subset $Data_{Makespan_{Flt} \leq T_{Flt}}(J)$ satisfy $Makespan_{Flt} \leq T_{Flt}$, where T_{Flt} is a target makespan specified by a tenant. Stated differently, the entries of the data set $Data_{Makespan_{Nml} \leq T_{Nml}}(J)$ whose $Makespan_{Flt}$ values exceed T_{Flt} are excluded from the subset $Data_{Makespan_{Flt} \leq T_{Flt}}(J)$.

5) Sort the subset $Data_{Makespan_{Flt} \leq T_{Flt}}(J)$ by the $Cost_{Flt}$ values in non-descending order.

6) Select an entry (or entries) in the subset $Data^{minCost}_{Makespan \leq T}(J)$ with a low cost. The selected entry (or entries) represent(s) the solution, *i.e.* a platform configuration of a corresponding instance type and cluster size. Each selected entry can also be associated with a schedule, which can also be considered to be part of the solution. The solution satisfies the target makespan T_{Nml} while reducing (or minimizing) the cost in such a way that if a node faults, the jobs are processed (e.g., completed) within a degraded time limit (e.g., T_{Flt}).

[0044] The foregoing further describes determining a schedule of jobs of a workload, according to some implementations, which was introduced above with reference to operation 406. For a set of MapReduce jobs (with no data dependencies between them), the order in which the jobs are executed may impact the overall processing time, and thus, utilization and the cost of the rented platform configuration (note that the price charged to a tenant can also depend on a length of time that rented resources are used—thus, increasing the processing time can lead to increased cost).

[0045] The following considers an example execution of two (independent) MapReduce jobs J_1 and J_2 in a cluster, in which no data dependencies exist between the jobs. As shown in FIG. 5, once the map stage (m_1) of J_1 completes, the reduce stage (r_1) of job J_1 can begin processing. Also, the execution of the map stage (m_2) of the next job J_2 can begin execution, by using the map resources released due to completion of the map stage (m_1) of J_1. Once the map stage (m_2) of the next job J_2 completes, the reduce stage (j_2) of the next job J_2 can begin. As shown in FIG. 5, there is an overlap in executions of map stage (m_2) of job J_2 and the reduce stage (r_1) of job J_1.

[0046] A first execution order of the jobs may lead to a less efficient resource usage and an increased processing time as compared to a second execution of the jobs. To illustrate this, consider an example workload that includes the following two jobs:

- Job $J_1 = (m_1, r_1) = (20s, 2s)$ (map stage has a duration of 20 seconds, and reduce stage has a duration of two seconds).
- Job $J_2 = (m_2, r_2) = (2s, 20s)$ (map stage has a duration of two seconds, and reduce stage has a duration of 20 seconds).

[0047] There are two possible execution orders for jobs J_1 and J_2 shown in Figs. 6A and 6B:

- J_1 is followed by J_2 (FIG. 6A). In this execution order, the overlap of the reduce stage of J_1 with the map stage of J_2 extends two seconds. As a result, the total completion time of processing jobs J_1 and J_2 is $20s + 2s + 20s = 42s$.

- J_2 is followed by J_1 (FIG. 6B). In this execution order, the overlap of the reduce stage of J_2 with the map stage of J_1 extends 20 seconds. As a result, the total completion time is $2s + 20s + 2s = 24s$, which is less than the first execution order.

[0048] More generally, there can be a substantial difference in the job completion time depending on the execution order of the jobs of a workload. A workload

$\mathcal{J} = \{J_1, J_2, \ldots, J_n\}$ includes a set of n MapReduce jobs with no data dependencies between them. The scheduler 214 generates an order (a schedule) of execution of jobs $J_i \in \mathcal{J}$ such that the makespan of the workload is minimized. For minimizing the makespan of the workload of jobs $\mathcal{J} = \{J_1, J_2, \ldots, J_n\}$, the Johnson scheduling technique can be used.

[0049] Each job J_i in the workload $\mathcal{J}$ of n jobs can be represented by the pair (m_i, r_i) of map and reduce stage durations, respectively. The values of m_i and r_i can be estimated using lower and upper bounds, as discussed above, in some examples. Each job $J_i = (m_i, r_i)$ can be augmented with an attribute D_i that is defined as follows:

$$D_i = \begin{cases} (m_i, m) & if\ min(m_i, r_i) = m_i, \\ (r_i, r) & otherwise. \end{cases}$$

[0050] The first argument in D_i is referred to as the stage duration and denoted as D_i^1. The second argument in D_i is referred to as the stage type (map or reduce) and denoted as D_i^2. In the above, (m_i, m), m_i represents the duration of the map stage, and m denotes that the type of the stage is a map stage. Similarly, in (r_i, r), r_i represents the duration of the reduce stage, and r denotes that the type of the stage is a reduce stage.

[0051] An example pseudocode of the Johnson scheduling technique is provided below.

Johnson scheduling technique

Input: A set $\mathcal{J}$ of n MapReduce jobs. D_i is the attribute of job J_i as defined above.

Output: Schedule σ (order of execution of jobs).

1: Sort the set $\mathcal{J}$ of jobs into an ordered list L using their stage duration attribute D_i^1
2: $head \leftarrow 1, tail \leftarrow n$
3: **for each** job J_i in L **do**
4: if $D_i^2 = m$ then
5: *// Put job J_i from the front*
6: $\sigma_{head} \leftarrow J_i$, head ← head + 1
7: **Else**
8: *// Put job J_i from the end*
9: $\sigma_{tail} \leftarrow J_i$, tail ← tail – 1
10: **end if**
11: **end for**

[0052] The Johnson scheduling technique (as performed by the scheduler 212) depicted above is discussed in connection with FIG. 7, which shows an scheduling queue 702 that includes a number of entries in which jobs of the set $\mathcal{J}$ of jobs are to be placed. Once the scheduling queue 702 is filled, then the jobs in the scheduling queue 702 can be executed in an order from the head ($head$) of the queue 702 to the tail ($tail$) of the scheduling queue 702. At line 2 of the pseudocode, $head$ is initialized to the value 1, and $tail$ is initialized to the value n (n is the number of jobs in the set $\mathcal{J}$).

[0053] Line 1 of the pseudocode sorts the n jobs of the set $\mathcal{J}$ in the ordered list L in such a way that job J_i precedes job J_{i+1} in the ordered list L if and only if $min(m_i, r_i) \leq min(m_{i+1}, r_{i+1})$. In other words, the jobs are sorted using the stage duration attribute D_i^1 in D_i (stage duration attribute D_i^1 represents the smallest duration of the two stages).

[0054] The pseudocode takes jobs from the ordered list L and places them into the schedule σ (represented by the scheduling queue 702) from the two ends (head

and tail), and then proceeds to place further jobs from the ordered list L in the intermediate positions of the scheduling queue 702. As specified at lines 4-6 of the pseudocode, if the stage type D_i^2 in D_i is m, *i.e.*, D_i^2 represents the map stage type, then job J_i is placed at the current available head of the scheduling queue 702 (as represented by $head$, which is initiated to the value 1. Once job J_i is placed in the scheduling queue 702, the value of $head$ is incremented by 1 (so that a next job would be placed at the next head position of the scheduling queue 702).

[0055] As specified at lines 7-9 of the pseudocode, if the stage type D_i^2 in D_i is not m, then job J_i is placed at the current available tail of the scheduling queue 702 (as represented by $tail$, which is initiated to the value n. Once job J_i is placed in the scheduling queue 702, the value of $tail$ is incremented by 1 (so that a next job would be placed at the next tail position of the scheduling queue 702).

[0056] FIG. 8 is a block diagram of a computing device 800 capable of selecting a platform configuration, according to one example. The computing device 800 includes, for example, a processor 810, and a computer-readable storage device 820 including platform configuration selection instructions 822. The computing device 800 may be, for example, a memory node, a processor node, (see FIG. 1) or any other suitable computing device capable of providing the functionality described herein.

[0057] The processor 810 may be a central processing unit (CPU), a semiconductor-based microprocessor, a graphics processing unit (GPU), other hardware devices or circuitry suitable for retrieval and execution of instructions stored in computer-readable storage device 820, or combinations thereof. For example, the processor 810 may include multiple cores on a chip, include multiple cores across multiple chips, multiple cores across multiple devices, or combinations thereof. The processor 810 may fetch, decode, and execute one or more of the platform configuration selection instructions 822 to implement methods and operations discussed above, with reference to FIGS. 1-6. As an alternative or in addition to retrieving and executing instructions, processor 810 may include at least one integrated circuit ("IC"), other control logic, other electronic circuits, or

combinations thereof that include a number of electronic components for performing the functionality of instructions 822.

[0058] Computer-readable storage device 820 may be any electronic, magnetic, optical, or other physical storage device that contains or stores executable instructions. Thus, computer-readable storage device may be, for example, Random Access Memory (RAM), an Electrically Erasable Programmable Read-Only Memory (EEPROM), a storage drive, a Compact Disc Read Only Memory (CD-ROM), non-volatile memory, and the like. As such, the machine- readable storage device can be non-transitory. As described in detail herein, computer-readable storage device 820 may be encoded with a series of executable instructions for selecting a platform configuration in light of a degraded makespan.

[0059] As used herein, the term "computer system" may refer to one or more computer devices, such as the computer device 800 shown in FIG. 8. Further, the terms "couple," "couples," "communicatively couple," or "communicatively coupled" is intended to mean either an indirect or direct connection. Thus, if a first device, module, or engine couples to a second device, module, or engine, that connection may be through a direct connection, or through an indirect connection via other devices, modules, logic, engines and connections. In the case of electrical connections, such coupling may be direct, indirect, through an optical connection, or through a wireless electrical connection.

[0060] While this disclosure makes reference to some examples, various modifications to the described examples may be made without departing from the scope of the claimed features.

Claims

What is claimed is:

1. A method comprising:

obtaining, by a computing system, a job profile of a prospective job, a normal makespan goal, and a degraded makespan goal, the job profile including a job trace summary;

generating, by the computing system, a simulation result of the prospective job based on a first simulation of the job trace summary on a platform configuration and a second simulation of the job trace summary on a degraded version of the platform configuration, the simulation result including a predicted normal makespan and a predicated degraded makespan; and

communicating, by the computing system, a purchasing option that selects the platform configuration, the purchasing option selects the platform configuration based on the predicted normal makespan satisfying the normal makespan goal and the predicted degraded makespan satisfying the degraded makespan goal.

2. The method of claim 1, further comprising:

generating an additional simulation result of the prospective job based on a third simulation of the job trace summary on another platform configuration and a fourth simulation of the job trace summary on a degraded version of the another platform configuration, the additional simulation result including another predicted normal makespan and another predicated degraded makespan; and

selecting the simulation result to use in the purchasing option based on a comparison between a cost associated with the simulation result and a cost associated with the additional simulation result.

3. The method of claim 2, wherein the platform configuration and the another platform configuration differ in a cluster size.

4. The method of claim 2, wherein the platform configuration and the another platform configuration differ in an instance type.

5. The method of claim 1, further comprising:

generating an additional simulation result of the prospective job based on a third simulation of the job trace summary on another platform configuration and a fourth simulation of the job trace summary on a degraded version of the another platform configuration, the additional simulation result including another predicted normal makespan and another predicated degraded makespan; and

selecting the simulation result to use in the purchasing option based on a comparison between the predicted normal makespan and the another predicted normal makespan.

6. The method of claim 1, further comprising generating the degraded version of the platform configuration, generating the degraded version of the platform configuration includes decrementing a cluster size associated with the platform configuration.

7. The method of claim 1, further comprising: responsive to detecting a tenant activation of the purchasing option, provisioning computing resources within a cloud infrastructure according to the platform configuration.

8. The method of claim 1, wherein the normal makespan goal and the degraded makespan goal are inputs supplied by a tenant system.

9. The method of claim 1, wherein the prospective job is a MapReduce job.

10. The method of claim 1, further comprising generating a schedule that lists an execution order for the prospective job and other prospective jobs, wherein the first simulation and the second simulation operate according to the schedule.

11. A system comprising:

a processor to:

obtain job profiles of prospective jobs in a workload, a normal makespan goal, and a degraded makespan goal, the job profiles including job trace summaries;

generate a schedule of the workload using the job trace summaries and a platform configuration;

aggregate a simulation result of an execution of the workload according to the schedule and the platform configuration and a simulation result of another execution of the workload according to the schedule and a degraded version of the platform configuration, the aggregated simulation result including a predicted normal makespan and a predicated degraded makespan;

select the platform configuration based on the predicted normal makespan satisfying the normal makespan goal and the predicted degraded makespan satisfying the degraded makespan goal; and

provision computing resources from a cloud infrastructure system according to the selected platform configuration.

12. The system of claim 11, wherein the processor to select the aggregated simulation result from additional aggregated simulation results based on a comparison of a cost associated with the aggregated simulation result and costs associated with the additional aggregated simulation results.

13. The system of claim 12, wherein the processor further to remove aggregated simulation results from the additional aggregated simulation results based the removed aggregated simulation results including predicted normal makespans that violate the normal makespan goal.

14. The system of claim 12, wherein the processor further to remove aggregated simulation results from the additional aggregated simulation results based the removed aggregated simulation results including predicted degraded makespans that violate the degraded makespan goal.

15. A computer-readable storage device comprising instructions that, when executed, cause a processor of a computer device to:

receive, from a tenant system, a job profile of a prospective job, a normal makespan goal, and a degraded makespan goal, the job profile including a job trace summary;

generate a predicted normal makespan and a predicated degraded makespan for a platform configuration based on executing a first simulation of the prospective job executing on computing resources according to the platform configuration and a second simulation of the prospective job executing on computing resources according to a degraded version of the platform configuration; and

select the platform configuration based on the predicted normal makespan satisfying the normal makespan goal and the predicted degraded makespan satisfying the degraded makespan goal.

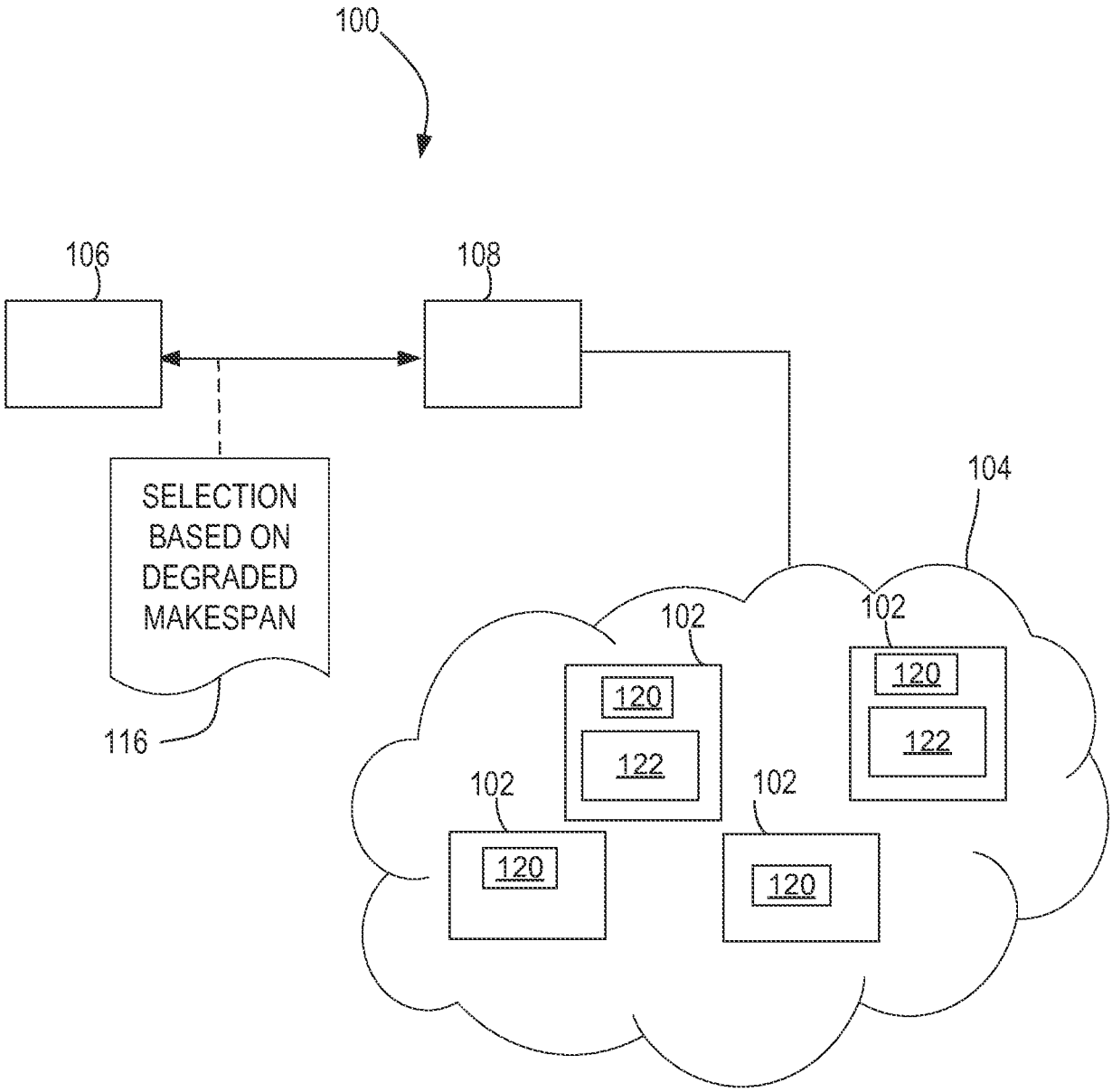
100
106
108
SELECTION BASED ON DEGRADED MAKESPAN
116
104
102
120
122
102
120
122
102
120
102
120

FIG. 1

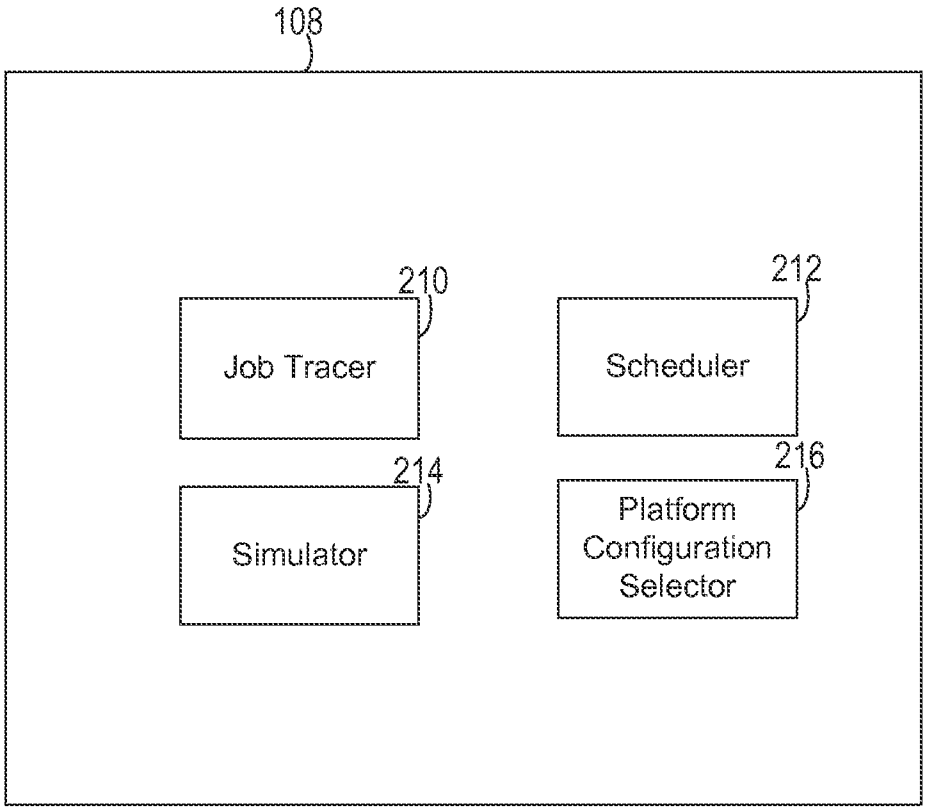
108
210
Job Tracer
212
Scheduler
214
Simulator
216
Platform Configuration Selector

FIG. 2

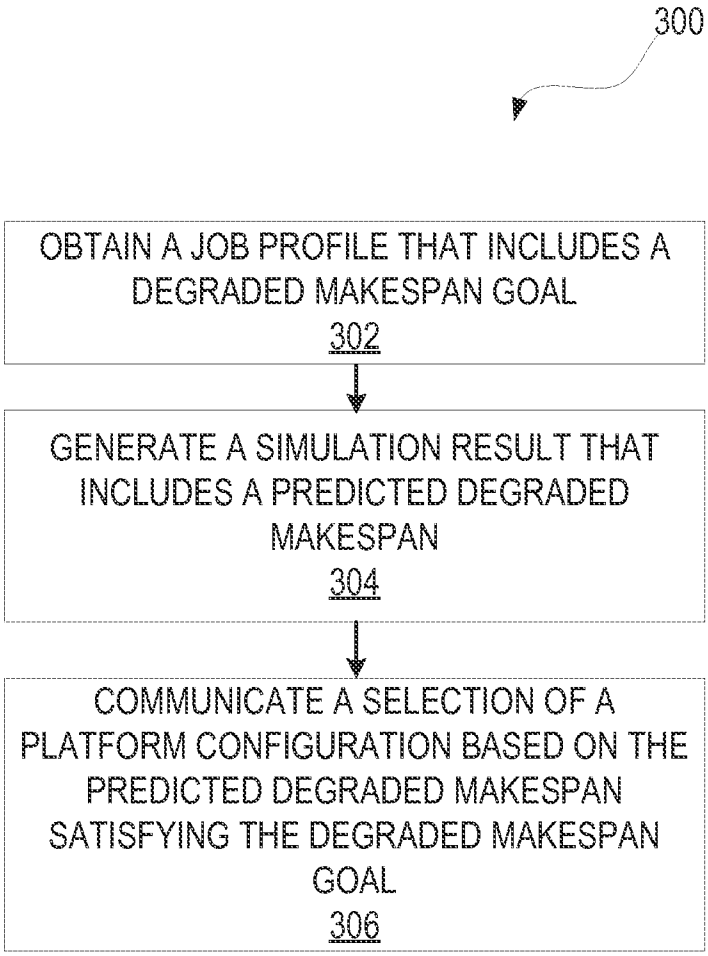
300
OBTAIN A JOB PROFILE THAT INCLUDES A DEGRADED MAKESPAN GOAL
302
GENERATE A SIMULATION RESULT THAT INCLUDES A PREDICTED DEGRADED MAKESPAN
304
COMMUNICATE A SELECTION OF A PLATFORM CONFIGURATION BASED ON THE PREDICTED DEGRADED MAKESPAN SATISFYING THE DEGRADED MAKESPAN GOAL
306

FIG. 3

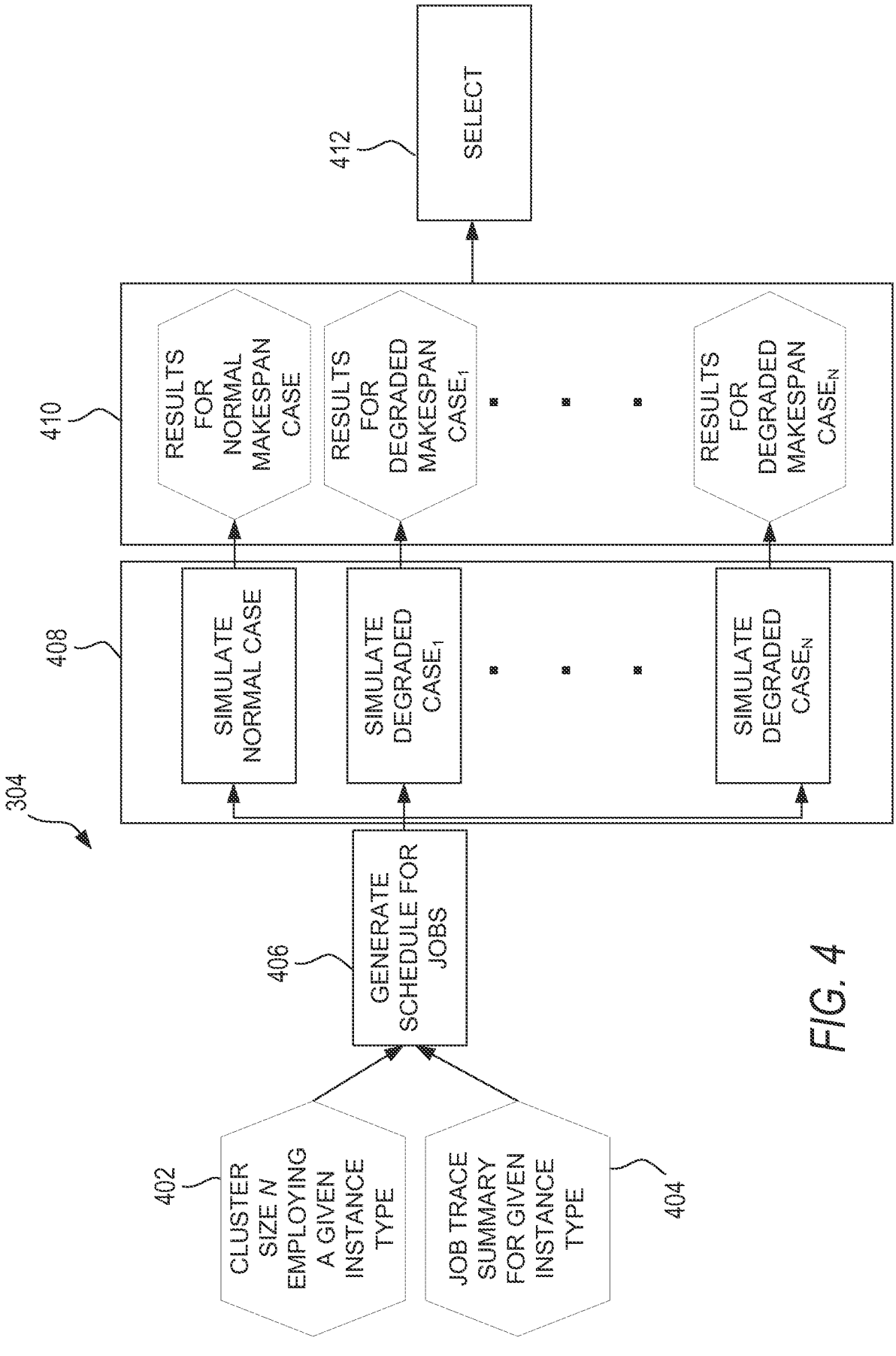
402
CLUSTER SIZE N EMPLOYING A GIVEN INSTANCE TYPE
404
JOB TRACE SUMMARY FOR GIVEN INSTANCE TYPE
406
GENERATE SCHEDULE FOR JOBS
304
408
SIMULATE NORMAL CASE
SIMULATE DEGRADED CASE1
SIMULATE DEGRADED CASEN
410
RESULTS FOR NORMAL MAKESPAN CASE
RESULTS FOR DEGRADED MAKESPAN CASE1
RESULTS FOR DEGRADED MAKESPAN CASEN
412
SELECT

FIG. 4

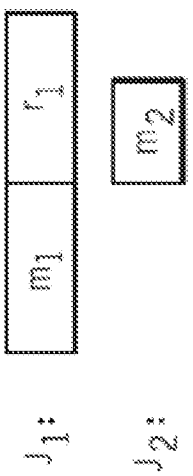

FIG. 5

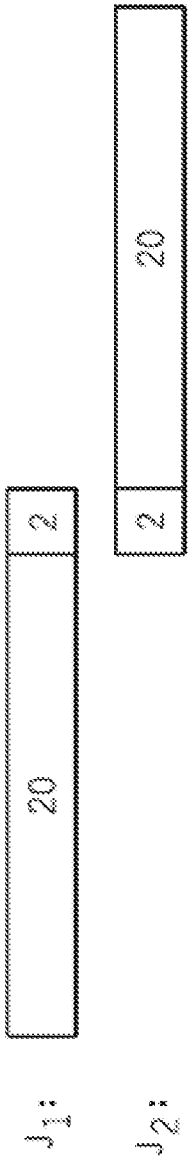

FIG. 6A

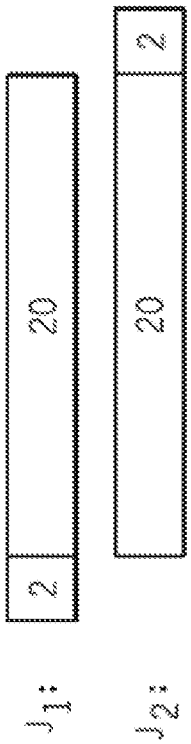

FIG. 6B

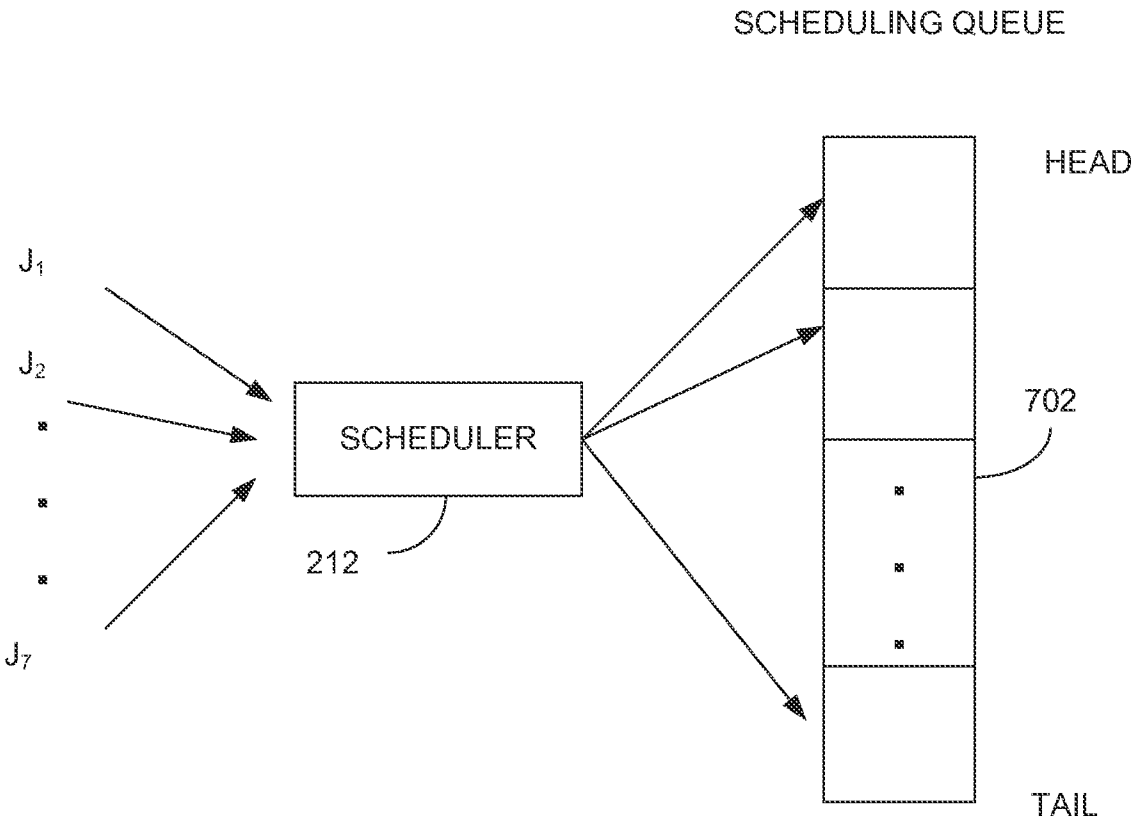

FIG. 7

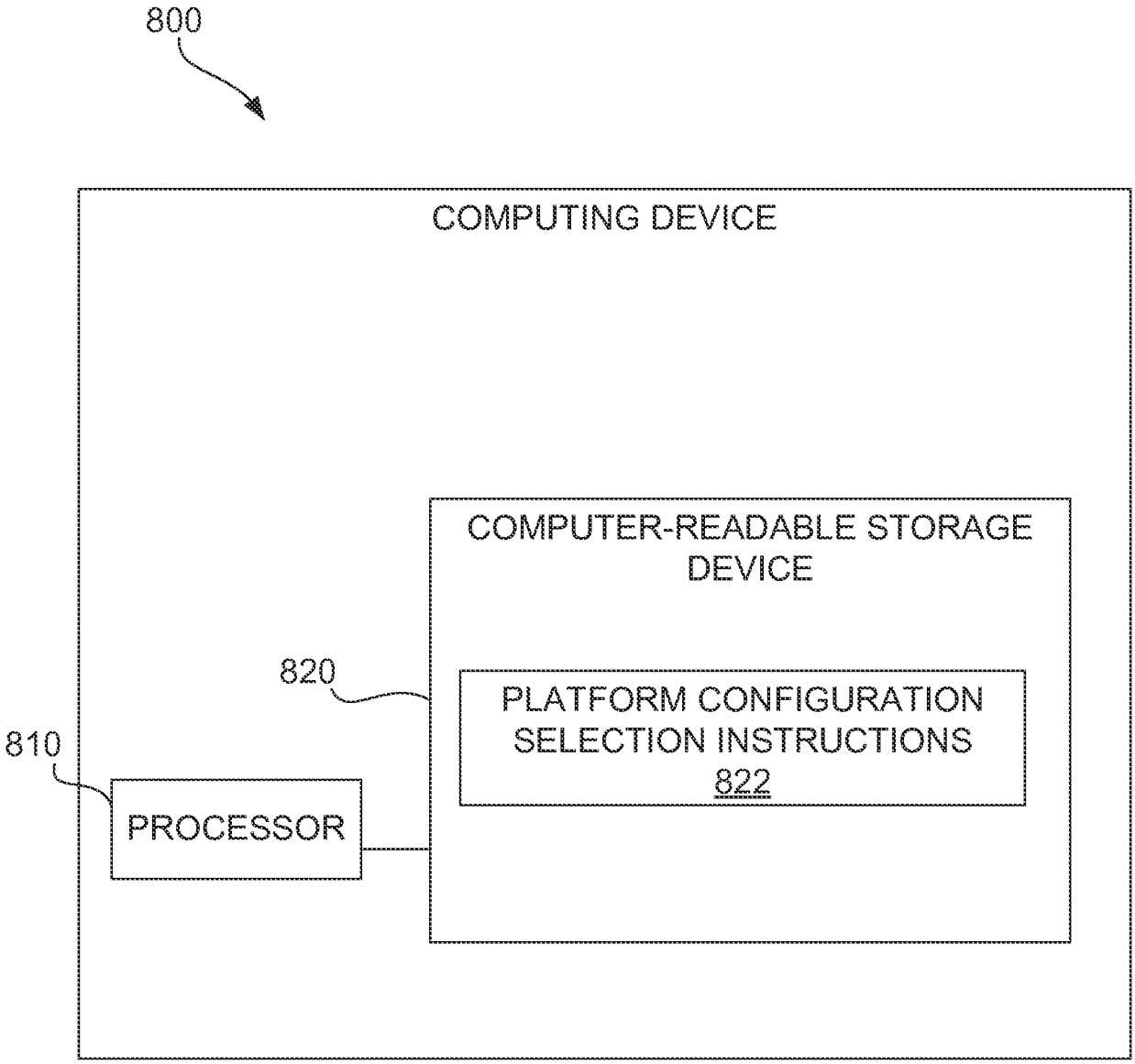
800
COMPUTING DEVICE
COMPUTER-READABLE STORAGE DEVICE
820
PLATFORM CONFIGURATION SELECTION INSTRUCTIONS
822
810
PROCESSOR

FIG. 8

INTERNATIONAL SEARCH REPORT

International application No.
PCT/US2014/049101

A. CLASSIFICATION OF SUBJECT MATTER

G06F 9/46(2006.01)i

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 9/46; G06F 9/455; G06F 1/32; G06F 9/50

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched
Korean utility models and applications for utility models
Japanese utility models and applications for utility models

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)
eKOMPASS(KIPO internal) & Keywords: computing, platform, configuration, selection

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
A	US 2012-0185868 A1 (BARTFAI-WALCOTT, K. K. et al.) 19 July 2012 See abstract, paragraphs [0055]-[0063], and fig. 4.	1-15
A	US 2010-0169072 A1 (ZAKI, S. M. et al.) 01 July 2010 See abstract, paragraphs [0099]-[0109], and figs. 11 and 12.	1-15
A	US 2012-0192186 A1 (BORNSTEIN, D. R. et al.) 26 July 2012 See abstract, paragraphs [0040]-[0047], and fig. 3.	1-15
A	US 2014-0019984 A1 (LI, W. et al.) 16 January 2014 See abstract, paragraphs [0055]-[0066], and fig. 2.	1-15
A	US 2008-0313640 A1 (LIU, J. et al.) 18 December 2008 See abstract, paragraphs [0027]-[0034], and figs. 2 and 3.	1-15

☐ 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	Date of mailing of the international search report
23 April 2015 (23.04.2015)	**24 April 2015 (24.04.2015)**

Name and mailing address of the ISA/KR
International Application Division
Korean Intellectual Property Office
189 Cheongsa-ro, Seo-gu, Daejeon Metropolitan City, 302-701,
Republic of Korea
Facsimile No. ++82 42 472 7140

Authorized officer
YU, Jintae
Telephone No. +82-42-481-8530

INTERNATIONAL SEARCH REPORT

Information on patent family members

International application No.
PCT/US2014/049101

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
US 2012-0185868 A1	19/07/2012	US 8868749 B2	21/10/2014
US 2010-0169072 A1	01/07/2010	US 8204734 B2	19/06/2012
US 2012-0192186 A1	26/07/2012	US 8429666 B2 WO 2012-103231 A1	23/04/2013 02/08/2012
US 2014-0019984 A1	16/01/2014	CN 103543987 A US 8914802 B2	29/01/2014 16/12/2014
US 2008-0313640 A1	18/12/2008	None	