

SYSTEM AND METHOD FOR IMPLEMENTING FAILOVER IN SCHEDULING SERVICES

TECHNICAL FIELD

[0001] Various aspects of this disclosure relate to a system and a method for implementing failover in scheduling services.

BACKGROUND

[0002] The following discussion of the background is intended to facilitate an understanding of the present disclosure only. It should be appreciated that the discussion is not an acknowledgment or admission that any of the material referred to was published, known, or is part of the common general knowledge of the person skilled in the art in any jurisdiction as of the priority date of the disclosure.

[0003] In large systems, scheduling services often handle the core scheduling logic, and implementing failover in scheduling services may be a part of ensuring high service availability. Failover, briefly, may refer to a process where, when an exception (such as a failure) occurs on a certain node, the tasks undertaken and processed on that node may be scheduled to other nodes for completion, leveraging on the role of scheduling services in such scenarios.

[0004] Traditionally, for a system to achieve failover, a master-slave scheduling mode may be employed. Specifically, such mode may involve, a large number of nodes, wherein one or multiple master nodes may be selected for primary processing while other slave nodes maintain detection of the corresponding master nodes. When an exception (e.g., a failure) is detected on a master node, a corresponding slave node may promptly take over the task(s) of the master node to continue any processing. However, in this traditional mode, while a slave node is configured to support the corresponding master node so as to provide reliability, half of the node resources (e.g., the slave nodes) in the system tend to be idle at the same time, wherein each slave node remains idle so as to be on standby to take on a task specifically from its corresponding master node, but is not available to run any task from a non-corresponding master node, resulting in significant resource waste.

[0005] Accordingly, the present disclosure seeks to provide a system and a method that at least ameliorate aforesaid limitation(s).

SUMMARY

[0006] The disclosure seeks to provide a system and a method for implementing failover in scheduling services.

[0007] According to an aspect of the present disclosure there is provided a system configured to implement a failover in scheduling services, the system may comprise server apparatuses each may comprise a processing unit and a memory unit, wherein the system (e.g., one or more of the server apparatuses) is configured to obtain a task to be executed and to associate the task with a label from which requirements of the task is identifiable, wherein the processing unit and the memory unit of each server apparatus may be co-operable to configure the server apparatus as a service node capable of obtaining the task, and wherein the server apparatuses may be configured to enter a competition mode in which the server apparatuses compete for the label, and one of the server apparatuses, which may match the requirements, may obtain and may hold the label, and may convert from the service node to a target service node to execute the task, and if an exception happens to occur in the target service node, the server apparatus operating as the target service node may be configured to release the label to idling server apparatuses to compete for the label, avoiding any unnecessary repetition and/or delay from the server apparatuses in executing the task.

[0008] In various embodiments, the processing unit of the server apparatus operating as the target service node may write an identification information into the label held by the server apparatus to generate a held label, wherein the identification information may contain an identity of the server apparatus holding the label, and another server apparatus, which happens to obtain the held label, may be configured not to execute the task corresponding to the held label.

[0009] In various embodiments, the processing unit of the server apparatus operating as the target service node, based on a pre-defined interval, may write a usage time for holding the label and may aggregate the usage time written to periodically generate a heartbeat information, wherein the heartbeat information may be written into the label or the held label, and wherein the heartbeat information may be stored in (i) the memory unit of the server apparatus operating as the target service node and (ii) the label or the held label .

[0010] In various embodiments, the processing unit of the server apparatus operating as the target service node may render a signal identifying the heartbeat information is not updated for a period that exceeds a threshold defined by the system and/or a user, and wherein the server

apparatus operating as the target service node may be configured to release the held label and the idling server apparatuses then may compete for the label.

[0011] In various embodiments, one of the idling server apparatuses may successfully compete for the label, may obtain and hold the label, and may convert from a service node to a target service node to (i) continue execution of the task or (ii) execute the task from start of the task.

[0012] In various embodiments, the exception may comprise a disconnection of the server apparatus operating as the target service node from the system, and/or a hardware failure in the server apparatus operating as the target service node, and/or a software error in the server apparatus operating as the target service node, and/or insufficient resources in the server apparatus operating as the target service node.

[0013] In various embodiments, the system (e.g., one or more of the server apparatuses) may be configured to (i) obtain a number of tasks to be processed, (ii) identify a number of the server apparatuses available as service nodes, and (iii) compare the number of tasks with the number of the server apparatuses which are available as service nodes so as to operate with a sufficient number of the server apparatuses available as service nodes to commence the competition mode, wherein the sufficient number may comprise any number of server apparatuses more than the number of tasks but avoids a larger number of idling server apparatuses.

[0014] In various embodiments, each server apparatus configured as a service node may be operable as a monitoring node, wherein the server apparatus may obtain a distribution lock pre-set in the system, wherein the distribution lock may render the server apparatus to switch from the service node to the monitoring node which may be configured to monitor if the heartbeat information is updated, and if the heartbeat information is not updated for a period exceeding the threshold defined by the system and/or the user, the server apparatus operating as the monitoring node may render the signal (e.g., to one or more other server apparatuses, such as one or more other idling server apparatuses), informing the exception occurred, or in various embodiments, the system may further comprise server apparatuses each configured as a monitoring node, which may be configured to monitor if the heartbeat information is updated, and if the heartbeat information is not updated for a period exceeding the threshold defined by the system and/or the user, each server apparatus operating as the monitoring node may render the signal (e.g., to one or more other server apparatuses, such as one or more other idling server apparatuses), informing the exception occurred.

[0015] In various embodiments, the server apparatus operating as the target service node may be configured to check, prior to executing the task, (i) if the server apparatus previously obtained and may still be holding a prior label, and if the server apparatus is still holding the prior label, then the server apparatus may proceed to execute the task associated with the prior label, and (ii) if the server apparatus is idling and holds no prior label, then the server apparatus may avoid executing the task associated with the prior label .

[0016] In various embodiments, the server apparatus operating as the service node may successfully compete for the label and before converting to the target service node, checks if any prior identification information is written into the label to be obtained by the server apparatus, and if no prior identification information was written into the label, the server apparatus may convert from the service node to the target service node and the processing unit of the server apparatus operating as the target service node may write an identification information into the label.

[0017] According to another aspect of the present disclosure there is provided a method for implementing a failover in a system configured to schedule services, the method may comprise obtaining a task by the system (e.g., one or more of the server apparatuses), associating the task with a label from which requirements of the task is identifiable, entering a competition mode in which server apparatuses may compete for the label, wherein one of the server apparatuses, which may match the requirements, may obtain the label and may hold the label, converting the server apparatus that obtained and is holding the label from a service node to a target service node to execute the task, and if an exception happens to occur in the target service node, releasing the label from the server apparatus operating as the target service node to idling server apparatuses to compete for the label, avoiding any unnecessary repetition and/or delay from the server apparatuses in executing the task.

[0018] In various embodiments, the method may further comprise writing an identification information into the label held by the server apparatus operating as the target service node to generate a held label which may prevent another server apparatus from executing the task corresponding to the held label, wherein the identification information may contain an identity of the server apparatus holding the label.

[0019] In various embodiments, the method may further comprise writing a usage time for holding the label into the label or the held label, aggregating the usage time written to periodically generate a heartbeat information, and storing the heartbeat information in the (i)

memory unit of the server apparatus operating as the target service node and (ii) the label or the held label.

[0020] In various embodiments, the method may further comprise rendering a signal identifying the heartbeat information not being updated for a period that exceeds a threshold defined by the system and/or a user, and releasing the held label by the server apparatus operating as the target service node to allow the idling server apparatuses to then compete for the label.

[0021] In various embodiments, the method may further comprise (i) continuing execution of the task by one of the idling server apparatuses which may have successfully competed, obtained, and holds the label, and converted from a service node to a target service node, or in various embodiments, (ii) executing the task from start of the task by one of the idling server apparatuses which may have successfully competed, obtained, and holds the label, and converted from a service node to a target service node.

[0022] In various embodiments, the exception may comprise a disconnection of the server apparatus operating as the target service node from the system, and/or a hardware failure in the server apparatus operating as the target service node, and/or a software error in the server apparatus operating as the target service node, and/or insufficient resources in the server apparatus operating as the target service node.

[0023] In various embodiments, the method may further comprise (i) configuring a number of tasks to be processed and the system (e.g., one or more server apparatuses) to obtain the number of tasks, (ii) configuring the system (e.g., one or more server apparatuses) to identify a number of the server apparatuses available as service nodes, and (iii) configuring the system (e.g., one or more server apparatuses) to compare the number of tasks with the number of the server apparatuses available as service nodes so as to operate with a sufficient number of the server apparatuses available as service nodes to commence the competition mode, wherein the sufficient number may comprise any number of server apparatuses more than the number of tasks but may avoid a larger number of idling server apparatuses.

[0024] In various embodiments, the method may further comprise obtaining a distribution lock pre-set in a system, wherein the server apparatus that obtains the distribution lock switches from the service node to a monitoring node which may be configured to monitor if the heartbeat information is updated, and if the heartbeat information is not updated for a period exceeding the threshold defined by the system and/or the user, rendering, by the server apparatus operating as the monitoring node the signal to inform the exception occurred, or in various

embodiments, monitoring if the heartbeat information is updated by server apparatuses configured as a monitoring node, and if the heartbeat information is not updated for a period exceeding the threshold defined by the system and/or the user, rendering, by the server apparatus operating as the monitoring node, the signal to inform the exception occurred.

[0025] In various embodiments, the method may further comprise configuring the server apparatus operating as the target service node to check, prior to executing the task: (i) if the server apparatus previously obtained and may still be holding a prior label, and if the server apparatus is still holding the prior label, then may proceed to execute the task associated with the prior label by the server apparatus, and (ii) if the server apparatus is idling and holds no prior label, then avoiding execution of the task associated with the prior label by the server apparatus.

[0026] In various embodiments, the method may further comprise checking, by the server apparatus operating as the service node which successfully competed for the label and before converting to the target service node, if any prior identification information is written into the label to be obtained by the server apparatus, and writing an identification information into the label by the processing unit of the server apparatus that converts from the service node to the target service node.

BRIEF DESCRIPTION OF THE DRAWINGS

[0027] The disclosure will be better understood with reference to the detailed description when considered in conjunction with the non-limiting examples and the accompanying drawings, in which:

- FIG. 1 is a diagram showing a non-limiting example of how the failover is implemented in the system and method of the present disclosure.
- FIG. 2 is a flow diagram illustrating non-limiting examples of steps in the system and method of the present disclosure.

DETAILED DESCRIPTION

[0028] The following detailed description refers to the accompanying drawings that show, by way of illustration, specific details and embodiments in which the disclosure may be practiced. These embodiments are described in sufficient detail to enable those skilled in the art to practice the disclosure. Other embodiments may be utilized and structural, and logical

changes may be made without departing from the scope of the disclosure. The various embodiments are not necessarily mutually exclusive, as some embodiments can be combined with one or more other embodiments to form new embodiments.

[0029] Embodiments, which are non-limiting examples, described in the context of one of the enclosure systems, server devices, or methods are analogously valid for the other systems, devices, or methods. Similarly, embodiments described in the context of a system are analogously valid for a device or a method, and vice-versa.

[0030] Features that are described in the context of an embodiment may correspondingly be applicable to the same or similar features in the other embodiments. Features that are described in the context of an embodiment may correspondingly be applicable to the other embodiments, even if not explicitly described in these other embodiments. Furthermore, additions and/or combinations and/or alternatives as described for a feature in the context of an embodiment may correspondingly be applicable to the same or similar feature in the other embodiments.

[0031] In the context of various embodiments, the articles “a”, “an” and “the” as used with regard to a feature or element include a reference to one or more of the features or elements. Furthermore, as used in the present disclosure and the appended claims, the term “by” may also mean “from”, depending on the context. Furthermore, as used in the present disclosure and the appended claims, the term “if” may also mean “when” or “upon”, depending on the context. Furthermore, as used in the present disclosure and the appended claims, the words “and/or” may refer to and encompass any and all possible combinations of one or more of the associated listed items.

[0032] As used herein, the term “data” may be understood to include information in any suitable analogue or digital form, for example, provided as a file, a portion of a file, a set of files, a signal or stream, a portion of a signal or stream, a set of signals or streams, waveforms, and the like. The term data, however, is not limited to the aforementioned examples and may take various forms and represent any information as understood in the art.

[0033] Where used herein, the term “first”, “second”, “third”, “fourth”, “fifth”, etc. are used to distinguish one element/feature from another, and, unless otherwise stated, may not denote order, priority or sequence.

[0034] As used herein, the term “associate”, “associated”, and “associating” indicate a defined relationship (or cross-reference) between two items.

[0035] As used herein, the term “obtained” or its grammatical variants, in the context of obtaining data, broadly include pull technology used any time a transfer of data is initiated by

a request sent from a client (e.g., one of the server apparatuses) to a server (e.g., another server apparatus). Push technology, on the other hand, is implemented any time a transfer of information is initiated by a server without waiting on a request from a client. In various embodiments, the term “obtain” may include receive.

[0036] As used herein, “memory unit” may be understood as a non-transitory computer-readable medium in which data or information can be stored. References to “memory” included herein may thus be understood as referring to volatile or non-volatile memory, including random access memory (“RAM”), read-only memory (“ROM”), flash memory, solid-state storage, magnetic tape, hard disk drive, optical drive, etc., or any combination thereof. Furthermore, it is appreciated that registers, shift registers, processor registers, data buffers, etc., may be embraced herein by the term “memory”. It is appreciated that a single component referred to as “memory” or “a memory” may be composed of more than one different type of memory, and thus may refer to a collective component including one or more types of memory. It is readily understood that any single memory component may be separated into multiple collectively equivalent memory components, and vice versa. Furthermore, while memory may be depicted as separate from one or more other components (such as in the drawings), it is understood that memory may be integrated within another component, such as on a common integrated chip.

[0037] As used herein, the term “configured to” broadly refers to an arrangement, an adaptation, and/or a program to perform a specific function. It implies a purposeful arrangement or adaptation for achieving the stated functionality. For example, a processor configured to process data may include hardware (e.g., electronic circuitry, chips), and/or software working in tandem to process the data.

[0038] The present disclosure describes a system and a method for implementing a failover in scheduling services. Particularly, the method may be for implementing such failover in a system configured to schedule services. The present method and system are advantageous over traditional scheduling systems and methods, wherein some example(s) of traditional scheduling systems and methods have been described above. In traditional scheduling systems and methods, a scheduling unit (i.e., a centralized scheduling unit, may also be traditionally referred to as a scheduler) may be involved. The centralized scheduling unit may be a computer configured to receive all incoming tasks, schedule all the tasks, and distributes the tasks to various nodes of a scheduling system for the nodes to perform the tasks, wherein the nodes may include or may be a server apparatus. As opposed to such traditional scheduling systems

and methods, the system and method of the present disclosure are absent of such a scheduling unit. Advantageously, the system and method of the present disclosure offer a more effective and more efficient scheduling of services in a distributed network system through a competition mode among service nodes, even without involving a scheduling unit. Advantageously, the present system and method are configured to set a number of required service nodes based on the task quantity, and at the same time, avoid a situation of having a large number of idle nodes. Advantageously, based on the competition mode, another service node may promptly take over and continue executing a task from a service node that encounters an anomaly, ensuring reliability. Advantageously, with the competition mode, any unnecessary repetition and/or delay from the service nodes in executing the task can be circumvented. Particularly, the service nodes of the present system and method refer to server apparatuses (or “servers” for brevity) configured to be operable as service nodes. A server apparatus configured as a service node or operating as a service node is a server that may constitute a resource to be scheduled or ready to obtain a task. A server apparatus of the present disclosure may comprise a processing unit (i.e., processor) and a memory unit. The processing unit is capable of being configured to execute instructions stored in the memory unit to receive a request or task. The server apparatus is capable of communicating with another server apparatus.

[0039] The system and method of the present disclosure may be referred to as a distributed scheduling system and method, which relate to a field of scheduling service technology and offer an improved task scheduling solution. In summary, the system and method may involve the following:

[0040] (1) obtain at least one task to be processed, where each task may be preconfigured with a label; and

[0041] (2) determine a target service node holding the label based on competition for the label among multiple service nodes, wherein the service nodes may be processing nodes in a distributed network. The service nodes may obtain a label through a competitive process. For example, multiple service nodes may compete for the label, and the successful node becomes a target service node that holds the label. This process may not be a random assignment but may be dynamically determined based on network conditions and the processing capabilities of each service node; and

[0042] (3) execute the task corresponding to the label using the target service node based on the label held by the target service node, if an exception is detected in the target service

node, the label held by the target service node may be released to allow other idle service nodes to compete for the label to continue executing the task corresponding to the label.

[0043] The present system and method are advantageous for applications that indirectly achieve task scheduling through a competition mode, circumventing the use of a traditional scheduling unit. In the distributed network system of the present disclosure, each service node may obtain a label through a competition mode to execute a task associated with the label, thereby achieving effective and reasonable scheduling of each service node, avoiding a large number of idle nodes, and ensuring reliability by allowing timely takeover and continuation of task execution based on such a competition mode if an exception happens to occur in a service node or a target service node. In other words, the competition mode serves as the distributed task allocation mechanism, wherein service nodes compete for a task and each of the service nodes is configured to identify their availability and if their requirements match those of the task for carrying out the task.

[0044] The system and method are described in details below to facilitate understanding. Details regarding various embodiments of the system and method are described below. Embodiments and advantages described for the present system can be analogously valid for the method of the present disclosure, and vice versa. Where the various embodiments and advantages have already been described for the system or for the method, they shall not be iterated for brevity.

[0045] According to an aspect of the disclosure and with reference to the figures, particularly FIG. 1, there is provided a system configured to implement a failover in scheduling services. The system may be referred to as a distributed network system, as it may involve a network of servers, but for brevity referred to as the system. The system may be configured to obtain a task to be executed, and associate the task with a label from which requirements of the task is identifiable. The system may also comprise server apparatuses each comprising a processing unit and a memory unit. The processing unit and the memory unit of each server apparatus may be co-operable to configure the server apparatus as a service node capable of obtaining the task, and the server apparatuses may be configured to enter a competition mode in which the server apparatuses may compete for the label, and one of the server apparatuses, which matches the requirements, may obtain and may hold the label, and may convert from the service node to a target service node to execute the task, and if an exception happens to occur in the target service node, the server apparatus operating as the target service node may then be configured to release the label to idling server apparatuses to compete for the label, avoiding

any unnecessary repetition and/or delay from the server apparatuses in executing the task. The competition mode may be commenced by an internal event trigger mechanism within the system (e.g., via one or more of the server apparatuses). The internal event trigger mechanism may be, for example, a task queue, a message queue, etc., wherein the system (e.g., a processing unit of one or more server apparatuses) may be configured to first identify an input (e.g., a task to be executed), and then setting off the internal event trigger mechanism such that the server apparatuses, which may be configured to recognize a task has been input, enter the competition mode. The system does not involve any scheduling unit, or any separate control unit from the server apparatuses, that receives a signal and sends a command to start the competition mode. Said differently, in various embodiments, the system may be absent of any scheduling unit, which is traditionally configured to at least receive a task, and schedule and distribute a task to a server apparatus for the task to be carried out, and wherein the scheduling unit is not a server apparatus of the present disclosure.

[0046] The requirements may include the type/nature of task (or a combination of tasks) to be carried out (e.g., to make a phone call, to send an email or a document), task priority, task execution time needed, any task dependency, and/or resource availability of the server apparatus (e.g., the processing unit's and memory unit's capacity/space, network bandwidth). The terms "task" and "service" in the context of an action to be scheduled and carried out by the present system and method may be used interchangeably.

[0047] The label, as identifiers and allocation tags for tasks, may contain information (such as task ID (task identification number), task type, etc.), which may be stored in a key-value pair format. Said differently, the label may contain data in the form of key-value pairs. For example, a key-value pair denoting the task and its type, such as {"Task": "phone call"}, {"Task Token": "Token A"}, {"Server Status": "Idle"}, {"Server Number": "1234"}, etc. However, if the need arises, the label may also be used as an independent judgement marker during the competition mode without recording any information. The label may include a token, wherein the token facilitates acquisition of a permission to execute tasks by competing for a label based on the association between the task and the label. In other words, the token may be an authentication token, such as a security token (e.g., JWT – a JSON Web Token, wherein JSON denotes JavaScript object notation) that is a digitally signed piece of data that may be exchanged between a client (e.g., a server apparatus) and a server (e.g., a server apparatus) for verifying identity or granting access. The label may be assigned sequentially or randomly to tasks without the need for additional control by any traditional scheduling unit.

[0048] The server apparatus may comprise a processing unit and a memory unit. The processing unit (e.g., a central processing unit, i.e., “CPU”) of the server apparatus is capable of being configured to execute instructions stored in the memory. For example, a server apparatus may obtain a task input by the user or from a memory unit of another server apparatus.

[0049] FIG. 1 shows an example of how the system (and the method) is configured to operate. Understandably, FIG. 1 also illustrates an example of the method carried out by the system for implementing the failover via the competition mode.

[0050] As shown in FIG. 1, tokens such as Token A, Token B, and Token C, are illustrated, and each token may be associated with a task to be processed. For the purpose of illustration, and not to limit the system and method, 5 service nodes (denoted Servers 1 to 5) are shown in the distributed network system. After a round of competition for tokens, Server 1 holds Token A, Server 3 holds Token B, Server 4 holds Token C, and the remaining Servers 2 and 5 may be idle. If a new task to be processed is created and associated with Token D, the two idle servers then continue to compete for Token D. In order to execute the task associated with Token D, if Server 5 holds Token D, then Server 2 remains idle at this time. If Server 3 is detected as abnormal, Token B held by Server 3 is released. Since only Server 2 is idle at this time, Server 2 competes for Token B to continue executing the task associated with Token B. The Tokens A to D may be assigned sequentially or randomly to tasks without the need for additional control by any traditional scheduling unit.

[0051] According to an aspect of the disclosure and with reference to the figures, particularly FIG. 2, there is provided a method for implementing a failover in scheduling services. FIG. 2 is a flow diagram illustrating an example of how the system described above may implement the method (e.g., for implementing a failover in a system configured to schedule services). Embodiments, features, and advantages described for the system in an aspect of the present disclosure can be analogously valid for the method described in an aspect of the present disclosure, and vice versa.

[0052] The method may comprise obtaining a task by the system, associating the task with a label from which requirements of the task is identifiable, entering a competition mode in which server apparatuses compete for the label, wherein one of the server apparatuses, which matches the requirements, obtains the label and holds the label, converting the server apparatus that obtained and is holding the label from a service node to a target service node to execute the task, and if an exception happens to occur in the target service node, releasing the label

from the server apparatus operating as the target service node to idling server apparatuses to compete for the label, avoiding any unnecessary repetition and/or delay from the server apparatuses in executing the task. The method may be absent of (may not involve) any traditional scheduling unit. Said differently, in various embodiments, the method may not involve (may be absent of) any scheduling unit, which is traditionally configured to at least receive a task, and schedule and distribute a task to a server apparatus for the task to be carried out. As an example, the method (and understandably the system) may involve:

[0053] Step S100: Obtaining at least one task to be processed, where each task is preconfigured with a corresponding label. To facilitate understanding, scenarios are provided as an example application that are non-limiting. Said differently, the scenarios are non-limiting examples meant to facilitate understanding of the scheduling capabilities of the present system and method. The scenarios may include intelligent outbound call scenarios. In such scenarios, the real-time requirements for tasks may not be high. For example, a task may be set to make intelligent outbound calls to 1000 users within 2 days. Since the voice for intelligent outbound calls may be pre-edited, such as voices for advertising and marketing, there may be no need for user responses. Such call tasks may have low real-time requirements, and each service node (e.g., server apparatus) in the distributed network may be capable of executing and completing such tasks. In this example application, a task pool may be pre-defined and multiple tasks to be processed may be obtained from the task pool. It may only be necessary to ensure that the number of service nodes set is greater than the number of tasks. In addition, in this application example, each task may be preconfigured with a corresponding label, which may include, but may not be limited to, a token. The purpose of doing so, as mentioned above, may be to facilitate the acquisition of permission to execute tasks by competing for labels based on the association between tasks and labels.

[0054] Step S200: Determining a target service node holding the label based on competition for the label among multiple service nodes, wherein the service nodes may be processing nodes in the distributed network system.

[0055] In step S200, after a service node contends for a label and becomes a target service node, preferably, the following two operations may also be required to establish a binding relationship between the target service node and the label and to indicate to other service nodes that the label is being held. The two operations may be explained as follows:

[0056] Step S200a: One operation is to write the identification information of the target service node into the held label to establish a binding relationship between the target service

node and the held label. Thus, even if another service node acquires this label again, based on the identification information written in the label, it can determine that the label has already been held by another service node and is being used for task execution, and this other service node does not repeat the execution of the task under this label. In the various embodiments, this operation is used to determine the holder of the label and to indicate to other service nodes that the label has already been held.

[0057] Step 200b: The other operation is that, during the period when the target service node holds the label, the target service node writes the usage time to the held label according to a preset time period, and aggregates the usage time written in different time periods as heartbeat information maintained between the target service node and the held label. The usage time refers to the duration for which the service node (or the target service node) holds the label, rather than solely the time required to complete the task. During this period, the service node (or the target service node) may periodically write the usage time to the label at predefined intervals, which may be aggregated as the heartbeat information maintained between the service node and the label. The preset time period (i.e., interval) does not refer to a set time for task execution but rather to the periodic intervals at which the service node (or target service node) writes the usage time to the label. As a non-limiting example, the interval can be set to every 10 minutes, 5 minutes, 2 minutes, 1 minutes, etc., during which the service node (or target service node) regularly (i.e., periodically) writes the usage time to the label.

[0058] Based on the periodically written heartbeat information by the target service node to the label, it indirectly proves that the target service node is in a normal working state. That is, if the heartbeat information is not updated for a long time, it may indicate that the target service node is probably abnormal. If it still occupies and holds the label, this may make it difficult for the task associated with the label to be completed normally. Based on this, the various embodiments of further verifies whether the target service node holding the label is in a normal working state based on the heartbeat information. The “long time” may refer to a situation where the time interval between the latest usage time retrieved from the heartbeat information and the current system time exceeds a pre-defined threshold. The pre-defined threshold may be set (e.g., configured) based on system capabilities and actual requirements to determine if a service node is anomalous.

[0059] From steps S200, S200a and S200b, it can be understood that in various embodiments, the processing unit of the server apparatus operating as the target service node writes an identification information into the label held by the server apparatus to generate a

held label, wherein the identification information contains an identity of the server apparatus holding the label, and another server apparatus, which happens to obtain the held label, is configured not to execute the task corresponding to the held label.

[0060] From steps S200, S200a and S200b, it can be understood that in various embodiments, the processing unit of the server apparatus operating as the target service node, based on a pre-defined interval, writes a usage time for holding the label and aggregates the usage time written to periodically generate a heartbeat information, wherein the heartbeat information is written into the label or the held label, and wherein the heartbeat information is stored in (i) the memory unit of the server apparatus operating as the target service node and (ii) the label or the held label.

[0061] From steps S200, S200a and S200b, it can be understood that in various embodiments, the method may further comprise writing an identification information into the label held by the server apparatus operating as the target service node to generate a held label which prevents another server apparatus from executing the task corresponding to the held label, wherein the identification information contains an identity of the server apparatus holding the label.

[0062] From steps S200, S200a and S200b, it can be understood that in various embodiments, the method may further comprise writing a usage time for holding the label into the label or the held label, aggregating the usage time written to periodically generate a heartbeat information, and storing the heartbeat information in the (i) memory unit of the server apparatus operating as the target service node and (ii) the label or the held label.

[0063] Step S300: Executing the task corresponding to the label using the target service node based on the labels held by each target service node. The service nodes may be processing nodes (nodes that become target service nodes that may eventually execute tasks) in the distributed network system. The number of tasks to be processed is less than the number of service nodes, and the difference between the two is very small, in order to avoid a large number of idle service nodes. Based on this very small difference, the various embodiments of such application implement task allocation in a contention mode. Particularly, if a service node contends for and holds a label, it indicates that the service node has obtained the permission to execute the task associated with the label. Based on the number of tasks to be processed, a corresponding number of target service nodes may also be required. Each target service node may execute the task associated with the label it holds.

[0064] From step S300, it can be understood that in various embodiments, the system (e.g., the one or more server apparatuses) may be configured to: (i) obtain a number of tasks to be processed, (ii) identify a number of the server apparatuses available as service nodes; and (iii) compare the number of tasks with the number of the server apparatuses which are available as service nodes so as to operate with a sufficient number of the server apparatuses available as service nodes to commence the competition mode, wherein the sufficient number comprises any number of server apparatuses more than the number of tasks but avoids a larger number of idling server apparatuses (e.g., having a number of idling server apparatuses that is 10 more than, or 5 more than, or 4 more than, or 3 more than, etc., the number tasks).

[0065] From step S300, it can be understood that in various embodiments, the method may further comprise (i) configuring a number of tasks to be processed and the system (e.g., the one or more server apparatuses) to obtain the number of tasks, (ii) configuring the system (e.g., the one or more server apparatuses) to identify a number of the server apparatuses available as service nodes, and (iii) configuring the system (e.g., the one or more server apparatuses) to compare the number of tasks with the number of the server apparatuses available as service nodes so as to operate with a sufficient number of the server apparatuses available as service nodes to commence the competition mode, wherein the sufficient number comprises any number of server apparatuses more than the number of tasks but avoids a larger number of idling server apparatuses (e.g., having a number of idling server apparatuses that is 10 more than, or 5 more than, or 4 more than, or 3 more than, etc., the number tasks).

[0066] Step S400: If an exception is detected in the target service node, releasing the label held by the target service node. This may involve monitoring the heartbeat information maintained between the target service node and the held label.

[0067] In the context of the present disclosure, an "abnormal" service node or target service node may refer to any abnormality that prevents the service node or target service node from performing as required, not necessarily a physical failure of the server. The anomalies may include software errors, network issues, insufficient resources, etc., all of which may cause the service node to malfunction. In various embodiments, non-limiting examples of the exception may comprise a disconnection of the server apparatus operating as the target service node from the system, and/or a hardware failure in the server apparatus operating as the target service node, and/or a software error in the server apparatus operating as the target service node, and/or insufficient resources in the server apparatus operating as the target service node. For instance, examples of abnormal service nodes and target service nodes may include, but are not limited

to, call disconnections (perhaps due to network issues or user disconnections), server hardware failures (such as damaged hard drives, memory failures, etc.), software errors (like program crashes, infinite loops, etc.), and insufficient resources (like high processing unit (e.g., CPU) usage, low memory, etc.), all of which may lead to the service node or the target service node being unable to function normally.

[0068] S500: Allowing other idle service nodes to compete for the label to continue executing the task corresponding to the label. In various embodiments, the processing unit of the server apparatus operating as the target service node may render a signal identifying the heartbeat information is not updated for a period that exceeds a threshold defined by the system and/or a user, and the server apparatus operating as the target service node may be configured to release the held label and the idling server apparatuses may then compete for the label. In various embodiments, one of the idling server apparatuses that successfully competes for the label, obtains and holds the label, and converts from a service node to a target service node to (i) continue execution of the task, or in various embodiments, (ii) execute the task from start of the task. In various embodiments, the method of may further comprise rendering a signal identifying the heartbeat information not being updated for a period that exceeds a threshold defined by the system and/or a user, and releasing the held label by the server apparatus operating as the target service node to allow the idling server apparatuses to then compete for the label. In various embodiments, the method may further comprise (i) continuing execution of the task by one of the idling server apparatuses which successfully competed, obtained, and holds the label, and converted from a service node to a target service node, or in various embodiments, (ii) executing the task from start of the task by one of the idling server apparatuses which successfully competed, obtained, and holds the label, and converted from a service node to a target service node. Embodiments of this step may involve determining whether the target service node is abnormal based on the heartbeat information.

[0069] Based on the difference in quantity between the number of tasks to be processed and the number of service nodes, after a round of contending for labels, the service nodes that hold labels may execute tasks, while the service nodes that do not hold labels may be in an idle state. In order to avoid excessive idle resources, this difference in quantity can be reduced to a very small and controllable level. This way, if a target service node is detected to be abnormal, it can be made to release the held label, allowing idle service nodes to continue contending for the label, take over the task associated with the label, and continue executing and completing

the task. Therefore, even if there is an abnormal service node in the distributed network, it may not affect the normal execution and completion of tasks.

[0070] Since the target service node should periodically write the usage time to the label during the task execution period, and these usage times written in different periods are aggregated as heartbeat information, the various embodiments can indirectly judge whether the target service node is in a normal working state based on the heartbeat information. The specific judgment measures may be as follows:

[0071] Firstly, obtain the latest usage time written from the heartbeat information, compare the latest usage time with the current system time to obtain the time interval; secondly, judge whether the time interval is greater than a preset threshold, if yes, determine that the target service node is abnormal, if no, determine that the target service node is working normally.

[0072] Step S600: If the target service node is detected to be abnormal, the label held by the target service node is released. If the target service node is abnormal, the purpose of releasing the label it holds is to allow other service nodes to have the opportunity to hold the label, and to avoid the task associated with the label being shelved.

[0073] It may be noted that, steps S400 to S600 provide monitoring and service operations for the target service node. The execution subject of this operation can be each service node in the distributed network, or a separately set service node that provides monitoring services (i.e., such a service node is not used to execute tasks to be processed). Based on the different execution subjects adopted, the various embodiments may provide the following two monitoring service schemes, which are specifically explained as follows:

[0074] One monitoring service scheme is: if the execution subject providing the monitoring service operation is each service node, then to avoid operation conflicts between multiple service nodes, a distributed lock can be preset in the distributed network. Accordingly, if a service node is the first to acquire the distributed lock, it has the permission to perform a monitoring service operation, and after completing a monitoring service operation, it should release the distributed lock. Additionally, it may be noted that in the preferred implementation, a time period for the monitoring service operation can be set to avoid frequent operations that waste processing costs and fail to achieve the monitoring effect. Moreover, to avoid the situation where a certain service node occupies the distributed lock for too long without releasing it, an upper limit for the processing duration of a single monitoring operation can also be set to ensure reasonable occupation of the distributed lock.

[0075] Another monitoring service scheme is: if the execution subject providing the monitoring service operation is a separately set service node that is not used to execute tasks to be processed, it can be referred to as a monitoring service node to distinguish it from other service nodes. In this case, there is no need to set a distributed lock as in the previous scheme. However, to ensure the reliability of operation execution (for example, to avoid the situation where the monitoring service cannot be provided due to an abnormality of the monitoring service node), multiple monitoring service nodes can be set, and further, primary and secondary nodes can be set among these monitoring service nodes. Furthermore, multiple high and low levels can be set among the primary and secondary nodes, so that the low-level nodes serve as backups for the high-level nodes, ensuring the normal provision of monitoring services. Additionally, it may be noted that the various embodiments may also consider the idle cost of service nodes, so the number of secondary nodes and the corresponding levels can be reasonably set based on actual demand and experience.

[0076] In various embodiments, monitoring services may be provided by a single service node or a group of service nodes for monitoring the status of all service nodes. The present system and method are advantageous in offering such a setup, which is versatile to cater to the system's scale and complexity, as well as the granularity of monitoring requirements. For example, in practical applications, a one-to-many or distributed monitoring architecture may be adopted to improve monitoring efficiency and reliability.

[0077] Accordingly, from steps S400 to S600, it can be understood that in various embodiments, each server apparatus configured as a service node is operable as a monitoring node, wherein the server apparatus obtains a distribution lock pre-set in the system, wherein the distribution lock renders the server apparatus to switch from the service node to the monitoring node which is configured to monitor if the heartbeat information is updated, and if the heartbeat information is not updated for a period exceeding the threshold defined by the system and/or the user, the server apparatus operating as the monitoring node renders the signal (e.g., to the other one or more server apparatuses), informing the exception occurred, or in various embodiments, the system may further comprise server apparatuses each configured as a monitoring node, which is configured to monitor if the heartbeat information is updated, and if the heartbeat information is not updated for a period exceeding the threshold defined by the system and/or the user, each server apparatus operating as the monitoring node renders the signal (e.g., to the other one or more server apparatuses), informing the exception occurred.

[0078] Accordingly, from steps S400 to S600, it can be understood that in various embodiments, the method may further comprise obtaining a distribution lock pre-set in a system, wherein the server apparatus that obtains the distribution lock switches from the service node to a monitoring node which is configured to monitor if the heartbeat information is updated, and if the heartbeat information is not updated for a period exceeding the threshold defined by the system and/or the user, rendering, by the server apparatus operating as the monitoring node, the signal (e.g., to the other one or more server apparatuses) to inform the exception occurred, or in various embodiments, monitoring if the heartbeat information is updated by server apparatuses configured as a monitoring node, and if the heartbeat information is not updated for a period exceeding the threshold defined by the system and/or the user, rendering, by the server apparatus operating as the monitoring node, the signal (e.g., to the other one or more server apparatuses) to inform the exception occurred.

[0079] Based on the two provided monitoring service schemes, the following monitoring operations may be adopted:

[0080] If the target service node is detected to be abnormal, the identification information of the target service node is deleted from the label held by the target service node, thereby determining the deletion of the binding relationship between the target service node and the held label, and releasing the label held by the target service node.

[0081] Additionally, it may be noted in the examples of this application that due to the low real-time requirements for task processing in the various embodiments, tasks on the target service node that has acquired the label can be scheduled to be executed at a predetermined time. Furthermore, the actual execution subjects of the two monitoring schemes mentioned above are other service nodes different from the target service node. Therefore, based on the time difference that occurs, there may be situations where a certain target service node no longer has a binding relationship with the previously held label before the scheduled task execution time arrives. This situation is specifically explained as follows:

[0082] For example, after the target service node acquires the label, it starts to maintain heartbeat information with the label. However, before the scheduled task execution time, the task has not yet started due to some factors (which are not limited in this application). During this time gap, the monitoring scheme mentioned above detects and determines that the target service node is abnormal due to some factors, forcing the target service node to release the label. However, since the factors leading to the abnormality have not been actually confirmed, and these factors may include non-fault abnormalities, such as temporary slow network speed

resulting in the failure to successfully write one or several usage times into the label, which means that the target service node itself has not actually failed. However, due to the monitoring scheme, the target service node releases the label. Therefore, at the scheduled task execution time, the target service node no longer has a binding relationship with the previously held label.

[0083] To address the possible scenario described above, before executing the task, each target service node that previously acquired the label should add a step to check whether the current binding relationship still exists. If the binding relationship exists, the task is executed; if the binding relationship does not exist, the service node is considered idle and no longer executes the task associated with the previously acquired label. This avoids executing the task directly based on a non-existent binding relationship, which could lead to error messages indicating abnormal task execution. If too many such error messages appear in the distributed network, it affects the normal task execution process in the distributed network and cause interference in the subsequent analysis of task results. Accordingly, in various embodiments, the server apparatus operating as the target service node is configured to check, prior to executing the task: (i) if the server apparatus previously obtained and is still holding a prior label, and if the server apparatus is still holding the prior label, then the server apparatus proceeds to execute the task associated with the prior label, and (ii) if the server apparatus is idling and holds no prior label, then the server apparatus avoids executing the task associated with the prior label. Accordingly, in various embodiments, the method may further comprise configuring the server apparatus operating as the target service node to check, prior to executing the task: (i) if the server apparatus previously obtained and is still holding a prior label, and if the server apparatus is still holding the prior label, then proceeding to execute the task associated with the prior label by the server apparatus, and (ii) if the server apparatus is idling and holds no prior label, then avoiding execution of the task associated with the prior label by the server apparatus.

[0084] In summary, based on steps S100 to S600, the distributed scheduling system and method of the present disclosure may involve pre-configuring corresponding labels for each task to be processed. This allows various service nodes within the distributed network to compete for these labels to acquire the right to execute the tasks associated with them. Furthermore, after a target service node acquires a label, the method also provides monitoring services for that target service node. Particularly, it monitors whether the label contains the identification information of the target service node to ensure the binding relationship between them, thereby preventing other service nodes from repeatedly acquiring the label. Additionally,

it monitors the heartbeat information between the target service node and its acquired label to determine whether the target service node is experiencing an anomaly. Based on the monitoring services provided above, if a target service node encounters an anomaly, it is promptly forced to release the label so that other idle service nodes can acquire and hold the label to continue completing the task.

[0085] The system and method circumvent the involvement of a traditional scheduling unit, and yet able to offer effective and reasonable scheduling of service nodes in a distributed network, due to the competition mode. While setting the number of required service nodes based on the task volume, the present system and method also avoid a situation of a large number of idle nodes. Moreover, if a service node encounters an anomaly, based on such a competition mode, it can promptly take over and continue executing the task, ensuring reliability.

[0086] S700: After other idle service nodes compete for the tag, it is determined whether any identification information of service nodes is written on the tag.

[0087] S800: If it is determined that no identification information of any service node is written on the tag, then its own identification information is written to the tag to establish a binding relationship between itself and the tag, and the task corresponding to the tag is completed according to the current execution progress.

[0088] All service nodes have the right to compete for each tag, but once the tag is held by a service node, other service nodes do not have the right to occupy it repeatedly. Therefore, even after a tag is released by the target service node, since every other idle service node has the right to compete for it, but it is unknown which one occupies it, if an idle service node grabs the tag, it needs to verify whether the identification information of a service node has been written on it (i.e., this write operation is used to prove tag ownership). If it is determined that no identification information has been written, then its own identification information is written to the tag, which means implementing the principle of "first write, first occupy". This ensures that even if other idle service nodes grab the tag, they cannot occupy it repeatedly, thereby avoiding the task under the tag being executed repeatedly by different service nodes.

[0089] Based on steps S700 and S800, it can be understood in various embodiments, the server apparatus operating as the service node successfully competes for the label and before converting to the target service node, may check if any prior identification information is written into the label to be obtained by the server apparatus, and if no prior identification information was written into the label, the server apparatus converts from the service node to

the target service node and the processing unit of the server apparatus operating as the target service node writes an identification information into the label.

[0090] Based on steps S700 and S800, it can be understood in various embodiments, the method may further comprise checking, by the server apparatus operating as the service node which successfully competed for the label and before converting to the target service node, if any prior identification information is written into the label to be obtained by the server apparatus, and writing an identification information into the label by the processing unit of the server apparatus that converts from the service node to the target service node.

[0091] The steps of the disclosed methods may not be restricted to the sequence set forth in the claims unless explicitly required. Steps may be performed in a different order, concurrently, or omitted entirely, depending on the embodiment.

[0092] The methods described herein may be performed and the various processing or computation units and the devices and computing entities described herein may be implemented by one or more circuits. In an embodiment, a "circuit" may be understood as any kind of a logic implementing entity, which may be hardware, software, firmware, or any combination thereof. Thus, in an embodiment, a "circuit" may be a hard-wired logic circuit or a programmable logic circuit such as a programmable processor, e.g., a microprocessor. A "circuit" may also be software being implemented or executed by a processor, e.g., any kind of computer program, e.g., a computer program using a virtual machine code. Any other kind of implementation of the respective functions which are described herein may also be understood as a "circuit" in accordance with an alternative embodiment.

[0093] While the disclosure has been particularly shown and described with reference to specific embodiments, it should be understood by those skilled in the art that various changes in form and detail may be made therein without departing from the spirit and scope of the disclosure as defined by the appended claims. The scope of the disclosure is thus indicated by the appended claims and all changes which come within the meaning and range of equivalency of the claims are therefore intended to be embraced.

CLAIMS

1. A system configured to implement a failover in scheduling services, the system comprising:

server apparatuses each comprising a processing unit and a memory unit,

wherein the system is configured to obtain a task to be executed and to associate the task with a label from which requirements of the task is identifiable,

wherein the processing unit and the memory unit of each server apparatus are co-operable to configure the server apparatus as a service node capable of obtaining the task, and

wherein the server apparatuses are configured to enter a competition mode in which the server apparatuses compete for the label, and one of the server apparatuses, which matches the requirements, obtains and holds the label, and converts from the service node to a target service node to execute the task,

and if an exception happens to occur in the target service node, the server apparatus operating as the target service node is configured to release the label to idling server apparatuses to compete for the label, avoiding any unnecessary repetition and/or delay from the server apparatuses in executing the task.

2. The system of claim 1,

wherein the processing unit of the server apparatus operating as the target service node writes an identification information into the label held by the server apparatus to generate a held label, wherein the identification information contains an identity of the server apparatus holding the label,

and

another server apparatus, which happens to obtain the held label, is configured not to execute the task corresponding to the held label.

3. The system of claim 2,

wherein the processing unit of the server apparatus operating as the target service node, based on a pre-defined interval, writes a usage time for holding the label and aggregates the usage time written to periodically generate a heartbeat information,

wherein the heartbeat information is written into the label or the held label, and

wherein the heartbeat information is stored in (i) the memory unit of the server apparatus operating as the target service node and (ii) the label or the held label.

4. The system of claim 3,

wherein the processing unit of the server apparatus operating as the target service node renders a signal identifying the heartbeat information is not updated for a period that exceeds a threshold defined by the system and/or a user, and

wherein the server apparatus operating as the target service node is configured to release the held label and the idling server apparatuses then compete for the label.

5. The system of claim 4, wherein one of the idling server apparatuses successfully competes for the label, obtains and holds the label, and converts from a service node to a target service node to (i) continue execution of the task or (ii) execute the task from start of the task.

6. The system of any one of claims 1 to 5, wherein the exception comprises:

a disconnection of the server apparatus operating as the target service node from the system; and/or

a hardware failure in the server apparatus operating as the target service node; and/or

a software error in the server apparatus operating as the target service node; and/or

insufficient resources in the server apparatus operating as the target service node.

7. The system of any one of claims 1 to 6, wherein the system is configured to:

(i) obtain a number of tasks to be processed,

(ii) identify a number of the server apparatuses available as service nodes; and

(iii) compare the number of tasks with the number of the server apparatuses which are available as service nodes so as to operate with a sufficient number of the server apparatuses available as service nodes to commence the competition mode, wherein the sufficient number comprises any number of server apparatuses more than the number of tasks but avoids a larger number of idling server apparatuses.

8. The system of claim 4,

wherein each server apparatus configured as a service node is operable as a monitoring node, wherein the server apparatus obtains a distribution lock pre-set in the system, wherein

the distribution lock renders the server apparatus to switch from the service node to the monitoring node which is configured to monitor if the heartbeat information is updated, and if the heartbeat information is not updated for a period exceeding the threshold defined by the system and/or the user, the server apparatus operating as the monitoring node renders the signal, informing the exception occurred;

or

further comprising server apparatuses each configured as a monitoring node, which is configured to monitor if the heartbeat information is updated, and if the heartbeat information is not updated for a period exceeding the threshold defined by the system and/or the user, each server apparatus operating as the monitoring node renders the signal, informing the exception occurred.

9. The system of claim 1, wherein the server apparatus operating as the target service node is configured to check, prior to executing the task:

(i) if the server apparatus previously obtained and is still holding a prior label, and if the server apparatus is still holding the prior label, then the server apparatus proceeds to execute the task associated with the prior label; and

(ii) if the server apparatus is idling and holds no prior label, then the server apparatus avoids executing the task associated with the prior label.

10. The system of claim 1 or 2, wherein the server apparatus operating as the service node successfully competes for the label and before converting to the target service node, checks if any prior identification information is written into the label to be obtained by the server apparatus, and if no prior identification information was written into the label, the server apparatus converts from the service node to the target service node and the processing unit of the server apparatus operating as the target service node writes an identification information into the label.

11. A method for implementing a failover in a system configured to schedule services, the method comprising:

obtaining a task by the system;

associating the task with a label from which requirements of the task is identifiable;

entering a competition mode in which server apparatuses compete for the label, wherein one of the server apparatuses, which matches the requirements, obtains the label and holds the label;

converting the server apparatus that obtained and is holding the label from a service node to a target service node to execute the task;

and

if an exception happens to occur in the target service node, releasing the label from the server apparatus operating as the target service node to idling server apparatuses to compete for the label, avoiding any unnecessary repetition and/or delay from the server apparatuses in executing the task.

12. The method of claim 11, further comprising:

writing an identification information into the label held by the server apparatus operating as the target service node to generate a held label which prevents another server apparatus from executing the task corresponding to the held label, wherein the identification information contains an identity of the server apparatus holding the label.

13. The method of claim 12, further comprising:

writing a usage time for holding the label into the label or the held label;

aggregating the usage time written to periodically generate a heartbeat information; and

storing the heartbeat information in the (i) memory unit of the server apparatus operating as the target service node and (ii) the label or the held label.

14. The method of claim 13, further comprising:

rendering a signal identifying the heartbeat information not being updated for a period that exceeds a threshold defined by the system and/or a user, and

releasing the held label by the server apparatus operating as the target service node to allow the idling server apparatuses to then compete for the label.

15. The method of claim 14, further comprising:

(i) continuing execution of the task by one of the idling server apparatuses which successfully competed, obtained, and holds the label, and converted from a service node to a target service node;

or

(ii) executing the task from start of the task by one of the idling server apparatuses which successfully competed, obtained, and holds the label, and converted from a service node to a target service node.

16. The method of any one of claims 11 to 15, wherein the exception comprises:

a disconnection of the server apparatus operating as the target service node from the system; and/or

a hardware failure in the server apparatus operating as the target service node; and/or

a software error in the server apparatus operating as the target service node; and/or

insufficient resources in the server apparatus operating as the target service node.

17. The method of any one of claims 11 to 17, further comprising:

(i) configuring a number of tasks to be processed and the system to obtain the number of tasks;

(ii) configuring the system to identify a number of the server apparatuses available as service nodes; and

(iii) configuring the system to compare the number of tasks with the number of the server apparatuses available as service nodes so as to operate with a sufficient number of the server apparatuses available as service nodes to commence the competition mode, wherein the sufficient number comprises any number of server apparatuses more than the number of tasks but avoids a larger number of idling server apparatuses.

18. The method of claim 14, further comprising:

obtaining a distribution lock pre-set in a system, wherein the server apparatus that obtains the distribution lock switches from the service node to a monitoring node which is configured to monitor if the heartbeat information is updated, and

if the heartbeat information is not updated for a period exceeding the threshold defined by the system and/or the user, rendering, by the server apparatus operating as the monitoring node, the signal to inform the exception occurred;

or

monitoring if the heartbeat information is updated by server apparatuses configured as a monitoring node, and

if the heartbeat information is not updated for a period exceeding the threshold defined by the system and/or the user, rendering, by the server apparatus operating as the monitoring node, the signal to inform the exception occurred.

19. The method of claim 11, further comprising:

configuring the server apparatus operating as the target service node to check, prior to executing the task:

- (i) if the server apparatus previously obtained and is still holding a prior label, and if the server apparatus is still holding the prior label, then proceeding to execute the task associated with the prior label by the server apparatus; and
- (ii) if the server apparatus is idling and holds no prior label, then avoiding execution of the task associated with the prior label by the server apparatus.

20. The method of claim 11 or 12, further comprising:

checking, by the server apparatus operating as the service node which successfully competed for the label and before converting to the target service node, if any prior identification information is written into the label to be obtained by the server apparatus; and

writing an identification information into the label by the processing unit of the server apparatus that converts from the service node to the target service node.