METHOD FOR SCHEDULING A TASK, RESOURCE SHARING USE METHOD AND SYSTEM BASED ON PREEMPTIVE SCHEDULING, SCHEDULER, DEVICE, AND STORAGE MEDIUM

The present application claims priority to a Chinese patent application No. 201810659298.5 filed with the CNIPA on Jun. 25, 2018, the disclosure of which is incorporated herein by reference in its entirety.

TECHNICAL FIELD

The present application relates to the field of computer technologies, and for example, to a method for scheduling a task, a resource sharing use method and system based on preemptive scheduling, a scheduler, a device, and a storage medium.

BACKGROUND

In a resource sharing distributed system, multiple tenants share and use resources, and meanwhile resources used by each tenant need to be limited to a certain extent, so as to ensure that each tenant can have resources to be used, and avoid the phenomenon of resource starvation of the tenant. A scheduler in the distributed system effectively schedules jobs or tasks of multiple tenants, thereby ensuring jobs or tasks of each tenant to be stably and quickly executed and meanwhile resources in the distributed system to be fully utilized.

A traditional distributed management system provides multiple scheduling strategies to ensure that the tasks of multiple tenants may be evenly distributed to physical nodes in the distributed system, and then the distributed tasks are run by the physical nodes. However, there is a phenomenon that resources are not fully utilized in the manner of processing the tasks.

SUMMARY

Embodiments of the present disclosure provide a preemptive scheduling based resource sharing use method, a system and a device, which can improve the utilization rate of resources.

In a first aspect, an embodiment of the present application provides a method for creating a task. The method includes: an application programming interface (API) server acquires a creation request of the task; and when the API server detects that a quota of a tenant to which the task belongs contains a resource matching with a priority of the task, and the matching resource satisfies a creation condition of the task, the task is created according to the creation request.

In a second aspect, an embodiment of the present application provides a method for scheduling

physical node needs a label of the GPU, and the second-stage optimization condition is the matching condition between the resources required by the current task and the remaining resources on the physical node. The scheduler selects physical nodes having the GPU, according to the node-task mapping table and the acquired information of the physical node, to form the node group. The scheduler determines whether the remaining resources on a physical node in the node group satisfy the resource condition required by the current task, physical nodes which do not satisfy the condition will be removed, and physical nodes which satisfy the resource condition required by the current task are scored, the more the remaining resources on the physical node are, the higher the score of the physical node is, and the physical node with the highest score is the target physical node. The method for screening the target physical node includes but is not limited to the method described above.

Through the above two times of screening, the physical node with the highest score is screened out to be served as the target physical node, that is, the physical node with the highest score is screened out to serve as the most appropriate physical node. Compared with a case of only one time of screening, the processing amount of data in each time of screening may be reduced, and the efficiency of the task scheduling may be improved.

In S230, the scheduler binds the current task with the target physical node, and sends binding information to the API server.

The scheduler binds the current task with the screened physical node (namely, the target physical node) best satisfying the preset screening condition, and sends the binding information to the API server, so that each physical node may read the task executed by itself from the API server.

According to the embodiments of the present application, the scheduler screens the multiple physical nodes based on the priority of the task and the preset screening condition to screen out the most appropriate physical node (namely, the target physical node) corresponding to each task, and schedules the current task to be scheduled to the most appropriate physical node. When the resources are in short supply, only the logical resource preemption is performed and the resources are not immediately preempted, so that the method of delaying a preemptive scheduling can logically yield resources for the task having the high priority, and continuously reserve the preempted task when the resources are not fully utilized, thereby improving the utilization rate of the resources.

physical node to form a node-pod mapping table.

Correspondingly, the binding module 830 is configured to bind the current pod with the target physical node, and send binding information to the API server.

A structure of the scheduler may also be other structural forms, so that the method for scheduling the task may be executed. For example, a scheduling system may be included in the scheduler, as shown in FIG. 8b, the scheduling system may include four parts: a node information list 840, a screening algorithm library 850, an optimization algorithm library 860, and an unscheduled queue 870.

The node information list 840 is configured to record currently available physical node information, including resource information (all resources and available resources) on the physical node, as well as a task queue that is already run on the physical node. This part of the information is key information for appointing the scheduling method, and needs to be synchronized in real time to ensure that the scheduling system has comprehensive cognition on resources and tasks

The screening algorithm library 850 is configured to predefine a variety of algorithms for screening the physical nodes to ensure physical nodes not satisfying an execution condition of the task to be removed.

The preferred algorithm library 860 is configured to predefine algorithms of multiple optimization nodes and weights of the algorithms, and select the physical node with the highest score calculated by the optimization algorithm as a scheduling node, namely the target physical node.

The unscheduled queue 870 is configured to be a queue formed by unscheduled tasks and is a priority queue to ensure a task having a high priority to be firstly scheduled.

The apparatus described above may execute the method for scheduling the task provided by any embodiment of the present application, and has function modules and beneficial effects corresponding to the method for scheduling the task.

FIG. 9 is a structural block diagram of an apparatus for preempting a task provided by an embodiment of the present application. As shown in FIG. 9, the apparatus for preempting the task includes a task list acquisition module 910, a detection module 920, a preemption module

CLAIMS

1. A method for scheduling a task, comprising:

acquiring, by a scheduler, a current task to be scheduled from a task scheduling queue, and acquiring a task, having a priority greater than or equal to a priority of the current task, on each physical node to form a node-task mapping table;

determining, by the scheduler, a target physical node satisfying a preset screening condition according to the mapping table and the preset screening condition; and

binding, by the scheduler, the current task with the target physical node, and sending binding information to an application programming interface (API) server.

2. The method of claim 1, wherein determining, by the scheduler, the target physical node satisfying the preset screening condition according to the mapping table and the preset screening condition comprises:

screening out, by the scheduler, physical nodes satisfying a first-stage screening condition from the mapping table to form a node group; and

scoring, according to the mapping table and a second-stage optimization condition, the physical nodes of the node group, and screening out a physical node with a highest score to serve as the target physical node.

3. The method of claim 1 or 2, wherein acquiring, by the scheduler, the current task to be scheduled from the task scheduling queue, and acquiring the task, having the priority greater than or equal to the priority of the current task, on each physical node to form the node-task mapping table comprises:

acquiring, by the scheduler, a current pod to be scheduled from a pod scheduling queue, and acquiring a pod, having a priority greater than or equal to a priority of the current pod, on each physical node to form a node-pod mapping table; and

wherein binding, by the scheduler, the current task with the target physical node, and sending the binding information to the API server comprises:

binding, by the scheduler, the current pod with the target physical node, and sending the binding information to the API server.

4. A resource sharing use method based on preemptive scheduling, comprising:

acquiring, by an application programming interface (API) server, a creation request of a task;

in a case where the API server detects that a quota of a tenant to which the task belongs contains a resource matching with a priority of the task, and the matching resource satisfies a creation condition of the task, creating the task according to the creation request;

acquiring, by a scheduler, the task created by the API server to form a task scheduling queue;

acquiring, by the scheduler, a current task to be scheduled from the task scheduling queue, and acquiring a task, having a priority greater than or equal to a specified priority of the current task, on each physical node to form a node-task mapping table;

determining, by the scheduler, a target physical node satisfying a preset screening condition according to the mapping table and the preset screening condition;

binding, by the scheduler, the current task with the target physical node, and sending binding information to the API server;

monitoring, by the physical node, the binding information between the task and the physical node in the API server, and acquiring the corresponding task based on the monitored binding information to form a task queue;

in a case where the physical node processes a target task to be run in the task queue, acquiring a task list of tasks running on the physical node;

detecting, by the physical node, whether remaining resources obtained after the physical node executes the tasks in the task list satisfy resources required for running the target task;

in a case where the resources required for running the target task are not satisfied, sequentially moving, by the physical node, at least one task having a priority lower than the target task in the task list into a queue to be removed in a priority order from low to high until the remaining resources of the physical node satisfy the resources required for running the target task, and adopting the target task to preempt the at least one task in the queue to be removed; and

calling, by the physical node, an execution environment to run the target task.

5. The method of claim 4, wherein acquiring, by the API server, the creation request of the task comprises: acquiring, by the API server, a creation request of a pod; and

wherein in a case where the API server detects that a quota of a tenant to which the task belongs contains a resource matching with a priority of the task, and the matching resource satisfies the creation condition of the task, creating the task according to the creation request comprises:

in a case where the API server detects that a quota of a namespace to which the pod belongs contains a quota value of a resource matching with a priority corresponding to the pod, and the quota value satisfies a creation condition of the pod, creating the pod according to the creation request of the pod.

6. The method of claim 4, further comprising:

acquiring, by the physical node, resource usage information every set time interval; and

in a case where the physical node determines that the resource usage information reaches a preset limitation condition, sequentially moving at least one task in the task list into the queue to be removed in the priority order from low to high until the resource usage information, determined in a case where the physical node executes tasks in the task list, does not reach the preset limitation condition, and stopping the at least one task in the queue to be removed.

7. The method of claim 4 or 6, wherein the target task is a target pod, the task list is a pod list, and tasks in the task list are pods running on the physical node.

8. A scheduler, comprising:

a mapping table forming module, which is configured to: acquire a current task to be scheduled from a task scheduling queue, and acquire a task, having a priority greater than or equal to a priority of the current task, on each physical node to form a node-task mapping table;

a screening module, which is configured to determine a target physical node satisfying a preset screening condition according to the mapping table and the preset screening condition; and

a binding module, which is configured to: bind the current task with the target physical node, and send binding information to an application programming interface (API) server.

9. A resource sharing use system based on preemptive scheduling, comprising an application

programming interface (API) server, the scheduler of claim 8, and an apparatus for preempting a task; wherein

the API server comprises:

a request acquisition module, which is configured to acquire a creation request of a task; and

a task creation module, which is configured to create the task according to the creation request, in a case of detecting that a quota of a tenant to which the task belongs contains a resource matching with a priority of the task, and the matching resource satisfies a creation condition of the task; and

the apparatus for preempting the task comprises:

a task list acquisition module, which is configured to acquire a task list of tasks running on a physical node in a case of processing a target task to be run;

a detection module, which is configured to detect whether remaining resources on the physical node satisfy resources required for running the target task;

a preemption module, which is configured to: in a case where the detection module detects that the remaining resources on the physical node do not satisfy the resources required for running the target task, sequentially move at least one task in the task list into a queue to be removed in a priority order from low to high until the remaining resources, obtained in a case where the physical node executes remaining tasks in the task list, satisfy the resources required for running the target task, and a priority of each task in the queue to be removed is less than a priority of the target task, and adopt the target task to preempt the at least one task in the queue to be removed; and

a task execution module, which is configured to call an execution environment to run the target task.

10. A device, comprising:

one or more processors; and

a storage apparatus for storing one or more programs;

wherein the one or more programs, when executed by the one or more processors, cause the one or more processors to implement the method for scheduling the task of any one of claims

1-3.

11. A computer-readable storage medium, storing a computer program, wherein the program, when executed by a processor, implements the method for scheduling the task of any one of claims 1-3.