(19)中华人民共和国国家知识产权局

(12)发明专利申请

(10)申请公布号 CN 110673945 A
(43)申请公布日 2020.01.10

(21)申请号 201810718590.X

(22)申请日 2018.07.03

(71)申请人 北京京东尚科信息技术有限公司
地址 100195 北京市海淀区杏石口路65号西杉创意园四区11号楼东段1-4层西段1-4层
申请人 北京京东世纪贸易有限公司

(72)发明人 罗勤 赵杰 邵伟 张永峰

(74)专利代理机构 北京成创同维知识产权代理有限公司 11449
代理人 范芳茗

(51)Int.Cl.
G06F 9/50(2006.01)

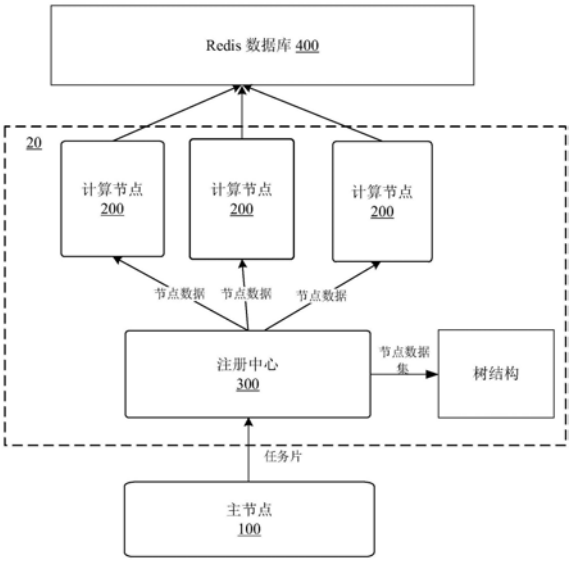
权利要求书2页 说明书7页 附图5页

(54)发明名称

分布式任务管理方法和管理系统

(57)摘要

本发明实施例提供一种分布式任务管理系统,包括:注册中心,接收多个任务,并且将所述多个任务结构化和存储为节点数据集;多个计算节点,从所述注册中心读取节点数据集中的节点数据,并且根据节点数据执行相应的任务。通过将任务结构化为节点数据集,并使计算节点从数据集中读取节点数据,计算节点能够根据需要读取节点数据,从而减轻计算节点的性能压力,提高数据处理效率。本发明同时提供一种分布式任务管理系统。



CN 110673945 A

1.一种分布式任务管理方法,其特征在于,包括:

接收多个任务片;

将所述多个任务片结构化和存储为节点数据集;

读取节点数据集中的节点数据;以及

根据节点数据执行相应的任务。

2.根据权利要求1所述的分布式任务管理方法,其特征在于,还包括:将任务分为多个任务片;

所述将所述多个任务片结构化和存储为节点数据集包括:每个任务片组成一个节点数据。

3.根据权利要求2所述的分布式任务管理方法,其特征在于,所述多个任务片结构化为树结构,每个节点数据存储在相应的一个树节点中。

4.根据权利要求3所述的分布式任务管理方法,其特征在于,所述将所述多个任务片结构化和存储为节点数据集还包括:将所述多个计算节点的IP地址存储在所述树结构的树节点中。

5.根据权利要求3所述的分布式任务管理方法,其特征在于,所述将所述多个任务片结构化和存储为节点数据集还包括:将每个任务片的执行状态存储在所述树结构的树节点中。

6.根据权利要求5所述的分布式任务管理方法,其特征在于,所述每个任务片的执行状态存储在所述树结构中的树节点中包括:

将待执行的任务片、执行失败的任务片、执行成功的任务片和正在执行的任务片的信息存储在所述树结构的树节点中。

7.根据权利要求6所述的分布式任务管理方法,其特征在于,当读取待执行的任务片时,在相应的树节点中增加正在执行的任务片的信息,当执行任务失败时,在相应的树节点中增加执行失败的任务片的信息,当任务执行结束后,在相应的树节点中增加执行成功的任务片的信息。

8.根据权利要求1所述的分布式任务管理方法,其特征在于,还包括:在任务执行过程中,将任务快照存储到数据库中。

9.根据权利要求1所述的分布式任务管理方法,其特征在于,还包括:根据任务快照恢复任务。

10.一种分布式任务管理系统,其特征在于,包括:

注册中心,接收多个任务片,并且将所述多个任务片结构化和存储为节点数据集;

多个计算节点,从所述注册中心读取节点数据集中的节点数据,并且根据节点数据执行相应的任务。

11.根据权利要求10所述的分布式任务管理系统,其特征在于,所述将所述多个任务片结构化和存储为节点数据集包括:每个任务片组成一个节点数据。

12.根据权利要求11所述的分布式任务管理系统,其特征在于,将所述多个任务片结构化为树结构,每个节点数据存储在相应的一个树节点中。

13.根据权利要求10所述的分布式任务管理方法,其特征在于,所述多个计算节点通过定时器触发执行相应的任务。

14.根据权利要求10所述的分布式任务管理系统,其特征在于,所述将所述多个任务片结构化和存储为节点数据集还包括:将所述多个计算节点的IP地址存储在所述树结构的相应树节点中。

15.根据权利要求12所述的分布式任务管理系统,其特征在于,所述将所述多个任务片结构化和存储为节点数据集还包括:将所述多个计算节点的任务片的执行状态存储在所述树结构中的树节点中。

16.根据权利要求15所述的分布式任务管理系统,其特征在于,将所述多个计算节点的任务片的执行状态存储在所述树结构中的树节点中包括:

在相应树节点中,存储待执行的任务片、执行失败的任务片、执行成功的任务片和正在执行的任务片的信息。

17.根据权利要求16所述的分布式任务管理系统,其特征在于,当所述多个计算中心读取待执行的任务片时,所述注册中心增加正在执行的任务片的信息,当所述多个计算中心执行任务失败时,所述注册中心增加执行失败的任务片的信息,当所述多个计算中心的任务执行结束后,所述注册中心增加执行成功的任务片的信息。

18.根据权利要求10所述的分布式任务管理系统,其特征在于,还包括:主节点,接收任务并将所述任务分为多个任务片。

19.根据权利要求10所述的分布式任务管理系统,其特征在于,所述计算节点将任务快照存储到数据库中。

20.根据权利要求10所述的分布式任务管理系统,其特征在于,所述计算节点根据任务快照恢复任务。

21.根据权利要求10所述的分布式任务管理系统,其特征在于,所述注册中心基于ZooKeeper。

22.一种计算机可读存储介质,其特征在于,所述计算机可读存储介质存储有计算机指令,所述计算机指令被执行时实现如权利要求1至9任一项所述的分布式任务管理方法。

23.一种分布式任务管理装置,其特征在于,包括:

存储器,用于存储计算机指令;

处理器,耦合到所述存储器,所述处理器被配置为基于所述存储器存储的计算机指令执行实现如权利要求1-9中任一项所述的分布式任务管理方法。

分布式任务管理方法和管理系统

技术领域

[0001] 本发明涉及计算机技术领域,具体涉及一种分布式任务管理方法和管理系统。

背景技术

[0002] 智能调拨是指从部署在一线城市的区域仓(RDC)把商品补给处于二线城市的前置仓(FDC)。这样可以使商品覆盖全国,而且供应商只需往区域仓(RDC)送货,而不用往全国各地的仓库送货。计算补货量需要对每个SKU(Stock Keeping Unit,库存量单位)进行补货计算,而由于基于基于SKU的补货计算涉及到数据量非常庞大,因此需要将该计算任务分配到多台服务器上同时执行。现有技术中,经常采用RPC(Remote Procedure Call,远程过程调用)技术和分片方法是经常采用的两种任务分配方法。

[0003] RPC技术例如使用JSF等远程方法调用方法。图1所示,在主节点master上,将任务分成多个任务片并调用RPC接口将多个任务片放到三台计算服务器Calc1-Calc3上进行运算,获得返回结果。RPC技术的缺点在于:由于网络带宽等限制,每次RPC请求携带的数据量必须适当,因而对于大数据量的任务,被分成多个任务请求,导致瞬间产生数量庞大的任务请求,造成对计算服务器很大的性能压力。例如现有100万数据,假设按一万条数据一组分发任务请求,那么需要分发100次到3台服务器,由于分发的过程非常快,导致每台计算服务器瞬间有了接近30个任务请求,如果每个任务请求的处理过程是一个非常耗时的工作,那么JVM(虚拟机)会得不到释放,导致计算服务器过载过热,影响整个计算集群。

[0004] 分片方法例如ElasticJob开源框架对数据进行分片,各数据节点取得各自任务片进行计算。分片方法是比较粗粒度的分片,容易发生数据倾斜的问题。例如,300万条数据,对应于上述的计算服务器Calc1-Calc3,则每个服务器将一次性分到100万数据,容易把JVM给撑爆。

[0005] 因此,发明人发现,上述的RPC技术和分片方法均存在不能有效处理大数据量的任务的问题。

发明内容

[0006] 有鉴于此,本发明实施例提供分布式任务管理系统和方法,其中计算节点从注册中心读取结构化的节点数据,以提高数据处理效率。

[0007] 根据本发明实施例的第一方面,提供一种分布式任务管理方法,包括:

[0008] 接收多个任务片;

[0009] 将所述多个任务片结构化和存储为节点数据集;

[0010] 读取节点数据集中的节点数据;以及

[0011] 根据节点数据执行相应的任务。

[0012] 优选地,还包括:将任务分为多个任务片;

[0013] 所述将所述多个任务片结构化和存储为节点数据集包括:每个任务片组成一个节点数据。

[0014] 优选地，所述多个任务片结构化为树结构，每个节点数据存储在相应的一个树节点中。

[0015] 优选地，所述将所述多个任务片结构化和存储为节点数据集还包括：将所述多个计算节点的IP地址存储在所述树结构的树节点中。

[0016] 优选地，所述将所述多个任务片结构化和存储为节点数据集还包括：将每个任务片的执行状态存储在所述树结构的树节点中。

[0017] 优选地，所述每个任务片的执行状态存储在所述树结构中的树节点中包括：

[0018] 将待执行的任务片、执行失败的任务片、执行成功的任务片和正在执行的任务片的信息存储在所述树结构的树节点中。

[0019] 优选地，当读取待执行的任务片时，在相应的树节点中增加正在执行的任务片的信息，当执行任务失败时，在相应的树节点中增加执行失败的任务片的信息，当任务执行结束后，在相应的树节点中增加执行成功的任务片的信息。

[0020] 优选地，还包括：在任务执行过程中，将任务快照存储到数据库中。

[0021] 优选地，还包括：根据任务快照恢复任务。

[0022] 根据本发明实施例的第二方面，提供一种分布式任务管理系统，包括：

[0023] 注册中心，接收多个任务片，并且将所述多个任务片结构化和存储为节点数据集；

[0024] 多个计算节点，从所述注册中心读取节点数据集中的节点数据，并且根据节点数据执行相应的任务。

[0025] 优选地，所述将所述多个任务片结构化和存储为节点数据集包括：每个任务片组成一个节点数据。

[0026] 优选地，将所述多个任务片结构化为树结构，每个节点数据存储在相应的一个树节点中。

[0027] 优选地，所述多个计算节点通过定时器触发执行相应的任务。

[0028] 优选地，所述将所述多个任务片结构化和存储为节点数据集还包括：将所述多个计算节点的IP地址存储在所述树结构的相应树节点中。

[0029] 优选地，所述将所述多个任务片结构化和存储为节点数据集还包括：将所述多个计算节点的任务片的执行状态存储在所述树结构中的树节点中。

[0030] 优选地，将所述多个计算节点的任务片的执行状态存储在所述树结构中的树节点中包括：

[0031] 在相应树节点中，存储待执行的任务片、执行失败的任务片、执行成功的任务片和正在执行的任务片的信息。

[0032] 优选地，当所述多个计算中心读取待执行的任务片时，所述注册中心增加正在执行的任务片的信息，当所述多个计算中心执行任务失败时，所述注册中心增加执行失败的任务片的信息，当所述多个计算中心的任务执行结束后，所述注册中心增加执行成功的任务片的信息。

[0033] 优选地，还包括：主节点，接收任务并将所述任务分为多个任务片。

[0034] 优选地，所述计算节点将任务快照存储到数据库中。

[0035] 优选地，所述计算节点根据任务快照恢复任务。

[0036] 优选地，所述注册中心基于ZooKeeper。

[0037] 根据本发明实施例的第三方面，提供一种计算机可读存储介质，所述计算机可读存储介质存储有计算机指令，所述计算机指令被执行时实现上述的分布式任务管理方法。

[0038] 根据本发明实施例的第四方面，提供一种分布式任务管理装置，包括：

[0039] 存储器，用于存储计算机指令；

[0040] 处理器，耦合到所述存储器，所述处理器被配置为基于所述存储器存储的计算机指令执行实现上述的分布式任务管理方法。

[0041] 本发明的实施例具有以下优点或有益效果：将任务片结构化为节点数据集存储在注册中心，并使计算节点从节点数据集中读取节点数据，从而可以减轻计算节点的性能压力，提高数据处理效率。

[0042] 本发明的优选实施例具有以下优点或有益效果：将节点数据存储为结构树，利用树的遍历获得任务片的执行状态，从而可以监控任务执行情况。

[0043] 本发明的另一优选实施例具有以下优点或有益效果：将任务快照存储到数据库中，在任务失败利用任务快照自动恢复任务执行，从而可以提高系统可靠性。

附图说明

[0044] 通过参照以下附图对本发明实施例的描述，本发明的上述以及其它目的、特征和优点将更为清楚，在附图中：

[0045] 图1是现有技术的RPC技术的结构示意图；

[0046] 图2是现有技术中的服务集群部署图；

[0047] 图3是根据本发明实施例的分布式任务管理方法的流程图；

[0048] 图4是根据本发明实施例的分布式任务管理系统的结构示意图；

[0049] 图5是根据本发明实施例的分布式任务管理方法的节点数据集的示例图；

[0050] 图6是根据本发明实施例的分布式任务管理装置的结构图。

具体实施方式

[0051] 以下基于实施例对本发明进行描述，但是本发明并不仅仅限于这些实施例。在下文对本发明的细节描述中，详尽描述了一些特定的细节部分。对本领域技术人员来说没有这些细节部分的描述也可以完全理解本发明。为了避免混淆本发明的实质，公知的方法、过程、流程没有详细叙述。另外附图不一定是按比例绘制的。

[0052] 如图2所示，多个服务器构成的服务器集群10，在各个服务器集群上部署有集群组件。通过集群组件为集群中的服务器提供一致性管理服务。参考图2，当客户端client向服务器集群发送请求，集群内部决定哪台或那些台服务器最终处理client的请求，但是对于所有客户端client来说，展示给客户端client都是同一个视图。集群组件例如ZooKeeper组件，它是一个分布式的，开放源码的分布式应用程序协调服务，为分布式应用提供一致性服务，提供的功能包括：配置维护、域名服务、分布式同步、组服务等。ZooKeeper不仅提供性能高效、功能稳定的系统而且提供简单易用的接口给用户。通过例如ZooKeeper这样的集群组件，服务器集群以一个整体面向多个客户端client的请求。

[0053] 基于上述的服务器集群，根据本发明实施例的分布式任务管理的方法得以实施。

[0054] 图3是本发明实施例的分布式任务管理方法的流程图，具体包括以下步骤。

[0055] 在步骤S101中，接收多个任务片。

[0056] 在步骤S102中，将多个任务片结构化和存储为节点数据集。

[0057] 在步骤S103中，读取节点数据集中的节点数据。

[0058] 在步骤S104中，根据节点数据执行相应的任务。

[0059] 具体地，对于上述服务集群，接收多个任务片，所述任务片可以来自另外一个节点，该节点将任务分割为多个任务片。相应地，每个任务片需要处理的数据减少。例如，300万条数据分为30个任务片，每个任务片处理10万条数据。再例如背景技术中所述的按照SKU计算补货量，则将涉及到的数据按照10万条一组进行分割，得到多个任务片。然后将这些任务片存储为结构化的节点数据集。每个任务片可对应存储一个节点数据。可以采用数据队列存储节点数据集，数据队列设置为具有先入先出的特性，使任务片只能串行领取和处理。由此确保任务片不会被重复领取。节点数据集存储集群中的一台服务器上，其他的服务器依次从节点数据集中获得节点数据，并根据节点数据执行对应的任务片。

[0060] 上述实施例将任务片结构化为节点数据集进行存储，并使多个计算节点分别从节点数据集中读取节点数据，从而减轻计算节点的性能压力，提高数据处理效率。如果服务器集群中包含低配置高配置的服务器，则每台服务器可按照处理能力动态获取节点数据，执行相应的任务，既保护了低配置的服务器，同时兼顾了处理效率。

[0061] 对于需要处理大数据量的任务，上述实施例尤其具有优势。在执行完一个任务片后再获取下一个任务片，如此能够有效保护服务器的内存资源。多个服务器根据自身的计算能力，获取不同数量的任务片，使得低配置的服务器获得有效保护。

[0062] 在可选的实施例中，用户可以预先指定任务片的大小，据此对任务进行分割，从而有效解决数据倾斜的问题。

[0063] 图4是根据本发明实施例的分布式任务管理系统的示意图。分布式任务管理系统20包括注册中心300和多个计算节点200。注册中心300接收多个任务片，并且将多个任务片结构化和存储为节点数据集。多个计算节点200从注册中心读取节点数据集中的节点数据，并且根据节点数据执行相应的任务。

[0064] 具体地，系统工作时，注册中心300接收多个任务片，并将任务片结构化为节点数据集，存储在注册中心，计算节点200从注册中心300分别读取节点数据，并执行相应的任务。在此过程中，注册中心监视计算节点200上的任务执行情况，并将其记录在结构化的节点数据集中。

[0065] 上述实施例将任务片结构化为节点数据集存储在注册中心，并使计算节点从节点数据集中读取节点数据，从而减轻计算节点的性能压力，提高数据处理效率。

[0066] 分布式任务管理系统20还可以进一步包括主节点100和Redis数据库400。主节点100用于将任务分成多个任务片。Redis数据库400用于存储过程数据。在一个实施例中，注册中心300接收任务，由主节点100将任务分成多个任务片。在另一个实施例中，由主节点100接收任务，将任务分成多个任务片后，将任务片发送给注册中心300。

[0067] 注册中心、计算节点和主节点均可以有部署集群组件，利用集群组件对进入到集群中的各个计算节点和主节点进行管理。同时，利用集群组件选出主节点和计算节点以执行不同的功能。而注册中心300上的节点数据集，其读写权限可以开放给多个计算节点200，使得每个计算节点200均可读取节点数据。

[0068] 在计算节点200处理任务片的过程中，可以将任务快照存放到Redis数据库400中，这个对于具有较多步骤的任务片尤其有效。如果某个步骤出现问题，可以快速的从快照数据中恢复到某一个步骤中，避免重复的计算量。通过任务快照，有效地提高了系统可靠性。

[0069] 图5示出了上述节点数据集的一个示例，存储任务片信息，以树结构表征一个具体任务，包括任务配置节点(task config)、实例节点(instances)、任务片节点(task shards)。任务配置节点存储多个任务片的元数据，实例节点存储计算节点和主节点的IP地址和主机名，任务片节点存储任务片的执行状态。在instances又包括节点master node和calc node，master node存储主节点的IP地址和主机名，calc node存储计算节点200的IP地址和主机名，task shards又包括节点task queen，task processing，task failover和task completed，其中，task queen存储待处理的任务片的信息，task processing存储正在处理的任务片的信息，task failover为处理失败的任务片的信息，task completed存储处理成功的任务片信息。在任务片节点处理过程中，任务片先在task queen等待计算节点200获取，一旦被计算节点200获取后，会转移到task processing中，并记录该任务片当前由哪台计算节点200(对应IP和host)处理，如果一个任务片处理失败后，该任务片会从task processing转移到task failover，并记录该任务片由哪台计算节点200在哪个步骤处理失败，如果该任务片处理成功，则该任务片会从task processing转移到task completed。Task failover里的任务片会被进一步记录到task queen，以待重新处理。

[0070] 可以看出，上述节点数据集以树结构存储。当然本发明实施例不限于此种存储结构的配置信息，例如，可以以关系型数据库存储。

[0071] 上述树结构的有益效果在于，将节点数据集存储为结构树后，利用树的遍历获得任务片的执行状态，方便监控任务执行情况。

[0072] 基于上述节点数据集的示例，在分布式计算失败时，可采用以下机制处理失败的任务片。具体地，在每一个计算节点200从Task Queen里读取到一个任务片的时候，相应地，在Task Processing目录下建立一个映射关系，指明任务片现在是由哪个服务器执行，如Shard1->10.168.90.80。当计算节点200和注册中心断开连接的时候，主节点100或注册中心300会得到相应的通知。此时找到断开连接相对应的任务片，把任务片写入到task failover中。通过上述机制，即使任务失败也可以从失败中自动恢复，从而保证了系统的可靠性。

[0073] 主节点100还可以以定时向计算节点200发送心跳信息以检查主节点100和计算节点200之间的连接。具体地，主节点100向计算节点200发出心跳信息，如果计算节点200有心跳，只是和注册中心断开的话，主节点100向计算节点200发出停止计算的命令，计算节点200根据命令把任务执行的快照存进Redis中。主节点100或注册中心300把taskfailover中的对应任务片移到task shards中的task queen里。如果计算节点确实挂掉，主节点100或注册中心300从Redis中取出计算任务运行的状态，写入task failover中以便未来的机器可以从失败的步骤直接开始。最终把task failover中的对应任务片移到task shards中的task queen里。

[0074] 如果主节点100失败，利用集群特性，所有和注册中心连接的服务器立马进行主节点100的选举，如果主节点100在生成task shards时失败，被选出的主节点100根据最后处理的任务片处理的步骤进行恢复，如果被选出的主节点100(之前作为计算节点200)有任务

在执行,则把任务执行的快照存进Redis中,把task processing中的信息移到task queen中。当任务结束时,主节点100清空Redis保存的中间信息,以及在ZooKeeper里面保存的目录信息,方便下次任务启动的时候不用再次清理。通过上述机制,即使任务失败也可以从失败中自动恢复,从而保证了系统的可靠性。

[0075] 进一步地,在计算节点200的计算的过程中也可以多线程处理,以提供按数据处理效率。

[0076] 应该指出的是,虽然上述实施例以ZooKeeper为例介绍本发明的一些方面,但是并不能以此限制本发明。因为其他的集群管理软件或者自主研发一款新的集群管理软件,也可以在其中实践本发明实施例提供的分布式任务管理方法。

[0077] 图6是根据本发明实施例的分布式任务管理装置的结构图。图6示出的设备仅仅是一个示例,不应对本发明实施例的功能和使用范围构成任何限制。

[0078] 参考图6,该分布式任务管理装置包括通过总线连接的处理器601、存储器602和输入输出设备603。存储器602包括只读存储器(ROM)和随机访问存储器(RAM),存储器602内存储有执行系统功能所需的各种计算机指令和数据,处理器601从存储器602中读取各种计算机指令以执行各种适当的动作和处理。输入输出设备包括键盘、鼠标等的输入部分;包括诸如阴极射线管(CRT)、液晶显示器(LCD)等以及扬声器等的输出部分;包括硬盘等的存储部分;以及包括诸如LAN卡、调制解调器等的网络接口卡的通信部分。存储器602还存储有以下的计算机指令以完成本发明实施例的分布式任务管理方法规定的操作:接收多个任务片;将所述多个任务片结构化和存储为节点数据集;读取节点数据集中的节点数据;以及根据节点数据执行相应的任务。

[0079] 相应地,本发明实施例提供一种计算机可读存储介质,该计算机可读存储介质存储有计算机指令,所述计算机指令被执行时实现上述分布式任务管理方法所规定的操作。

[0080] 附图中的流程图、框图图示了本发明实施例的系统、方法、装置的可能的体系框架、功能和操作,流程图和框图上的方框可以代表一个模块、程序段或仅仅是一段代码,所述模块、程序段和代码都是用来实现规定逻辑功能的可执行指令。也应当注意,所述实现规定逻辑功能的可执行指令可以重新组合,从而生成新的模块和程序段。因此附图的方框以及方框顺序只是用来更好的图示实施例的过程和步骤,而不应以此作为对发明本身的限制。

[0081] 系统的各个模块或单元可以通过硬件、固件或软件实现。软件例如包括采用JAVA、C/C++/C#、SQL等各种编程语言形成的编码程序。虽然在方法以及方法图例中给出本发明实施例的步骤以及步骤的顺序,但是所述步骤实现规定的逻辑功能的可执行指令可以重新组合,从而生成新的步骤。所述步骤的顺序也不应该仅仅局限于所述方法以及方法图例中的步骤顺序,可以根据功能的需要随时进行调整。例如将其中的某些步骤并行或按照相反顺序执行。

[0082] 根据本发明的系统和方法可以部署在单个或多个服务器上。例如,可以将不同的模块分别部署在不同的服务器上,形成专用服务器。或者,可以在多个服务器上分布式部署相同的功能单元、模块或系统,以减轻负载压力。所述服务器包括但不限于在同一个局域网以及通过Internet连接的多个PC机、PC服务器、刀片机、超级计算机等。

[0083] 以上所述仅为本发明的优选实施例,并不用于限制本发明,对于本领域技术人员

而言，本发明可以有各种改动和变化。凡在本发明的精神和原理之内所作的任何修改、等同替换、改进等，均应包含在本发明的保护范围之内。

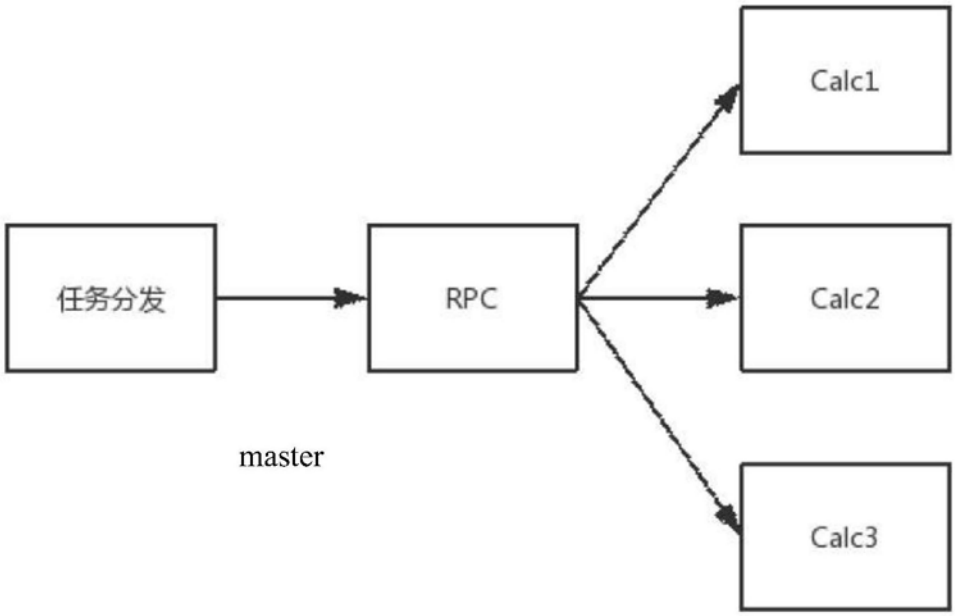

Calc1
任务分发
RPC
Calc2
master
Calc3

图1

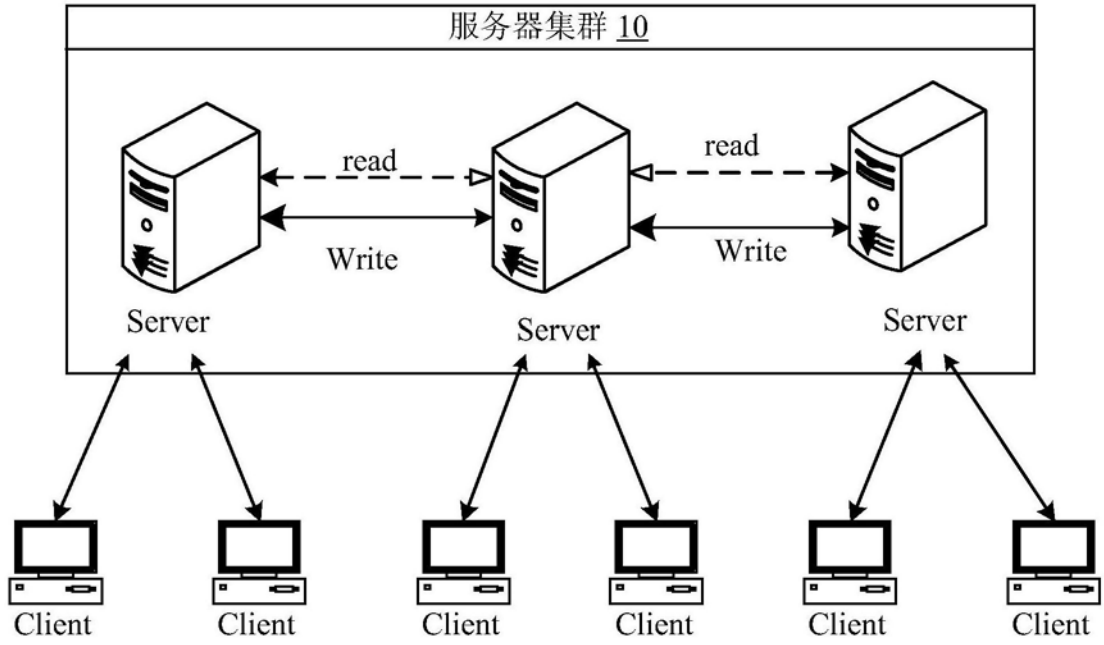

服务器集群 10
read
Write
read
Write
Server
Server
Server
Client
Client
Client
Client
Client
Client

图2

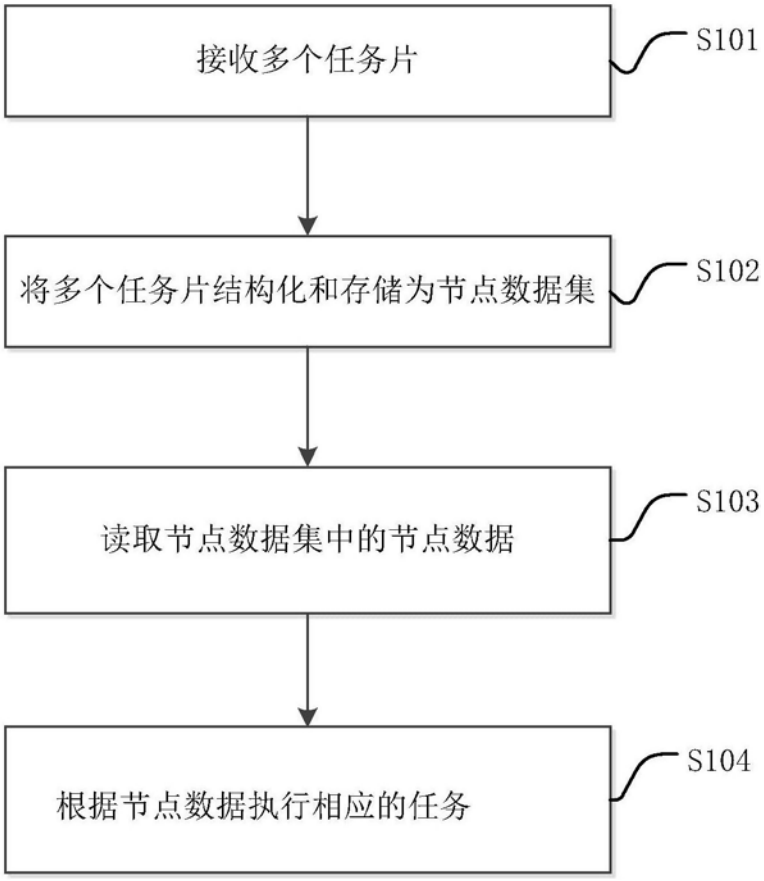

接收多个任务片
S101
将多个任务片结构化和存储为节点数据集
S102
读取节点数据集中的节点数据
S103
根据节点数据执行相应的任务
S104

图3

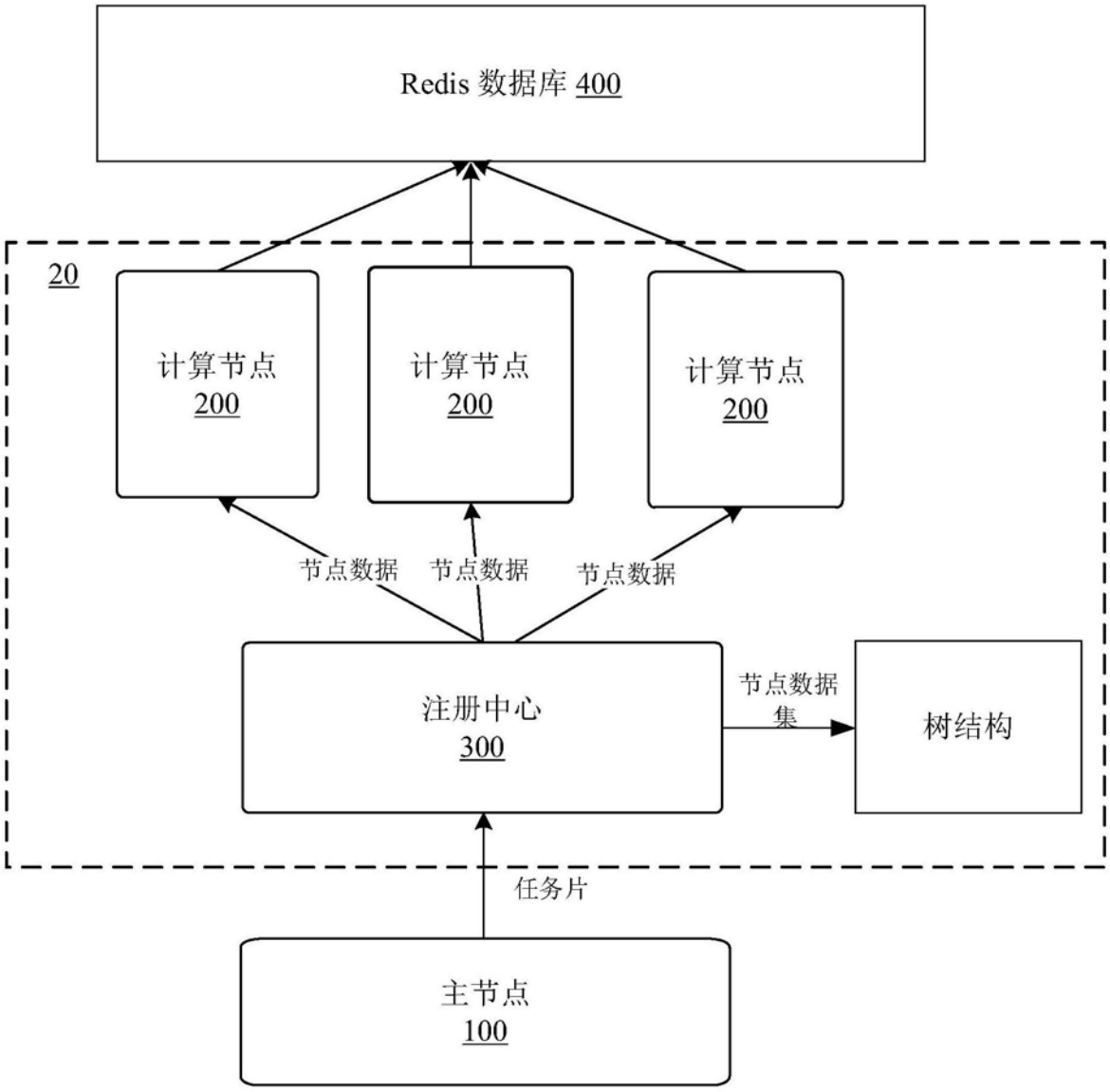

Redis 数据库 400
20
计算节点
200
计算节点
200
计算节点
200
节点数据
节点数据
节点数据
注册中心
300
节点数据
集
树结构
任务片
主节点
100

图4

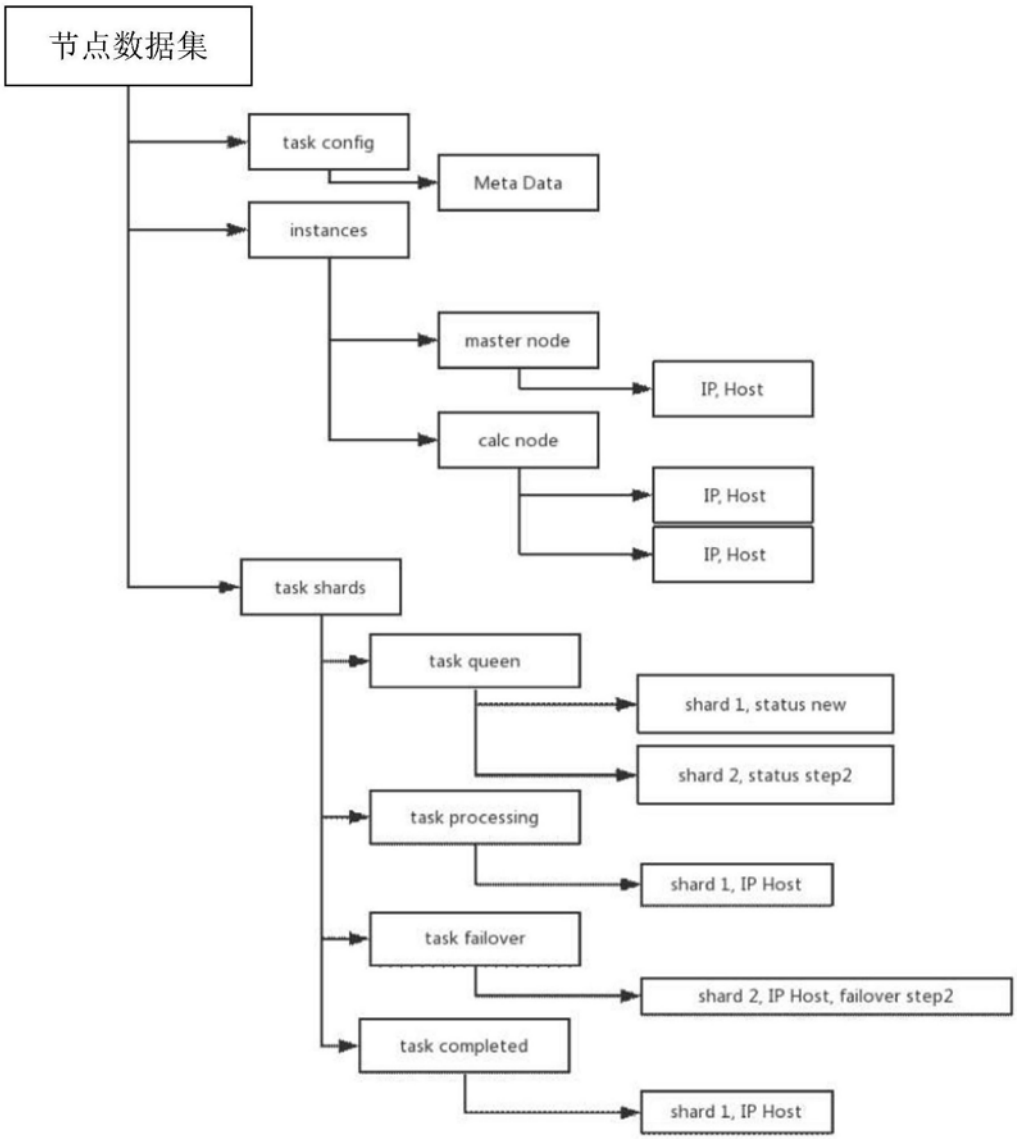

节点数据集
task config
Meta Data
instances
master node
IP, Host
calc node
IP, Host
IP, Host
task shards
task queen
shard 1, status new
shard 2, status step2
task processing
shard 1, IP Host
task failover
shard 2, IP Host, failover step2
task completed
shard 1, IP Host

图5

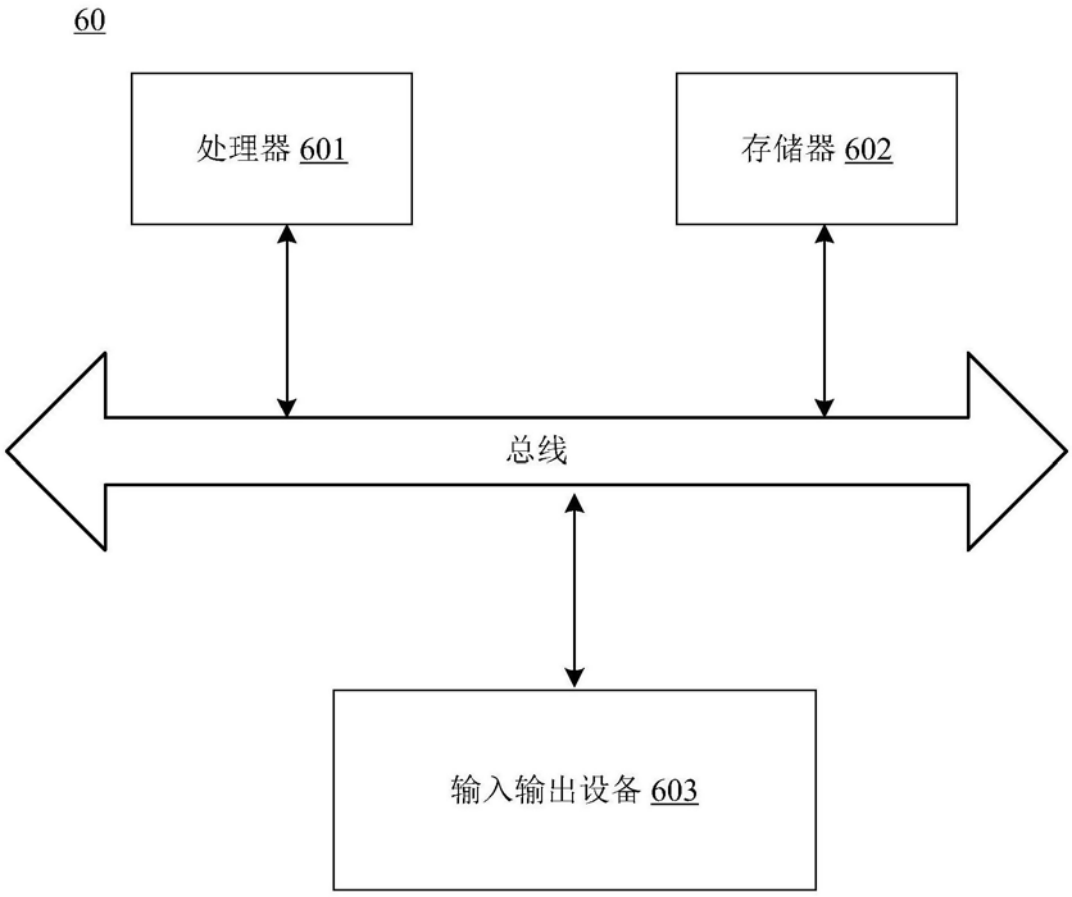


图6