(19)中华人民共和国国家知识产权局

(12)发明专利

(10)授权公告号 CN 106681820 B
(45)授权公告日 2020.05.01

(21)申请号 201611252002.5

(22)申请日 2016.12.30

(65)同一申请的已公布的文献号
申请公布号 CN 106681820 A

(43)申请公布日 2017.05.17

(73)专利权人 西北工业大学
地址 710072 陕西省西安市友谊西路127号

(72)发明人 汤小春 田凯飞 段慧芳

(74)专利代理机构 西北工业大学专利中心 61204
代理人 王鲜凯

(51)Int.Cl.
G06F 9/48(2006.01)
G06F 9/50(2006.01)

(56)对比文件
CN 104978232 A,2015.10.14,
CN 105426255 A,2016.03.23,
CN 105100267 A,2015.11.25,
CN 104904160 A,2015.09.09,
CN 105930360 A,2016.09.07,
US 2014/0222745 A1,2014.08.07,
汤小春,李洪华.分布式系统中计算作业流的均衡调度算法.《计算机工程》.2010,

审查员 刘董敏

权利要求书1页 说明书7页 附图4页

(54)发明名称

基于消息组合的可扩展大数据计算方法

(57)摘要

本发明公开了一种基于消息组合的可扩展大数据计算方法,用于解决现有大数据计算方法实用性差的技术问题。技术方案是首先建立抽象的大数据计算作业拓扑结构,再部署大数据计算作业,生成大算数据计算作业中计算任务执行计划,调度大数据计算任务,进行计算任务执行过程的负载均衡,执行大数据计算任务。当作业执行完毕,作业生命结束。由于将计算任务的拓扑结构与计算任务分割,拓扑结构定义大数据计算作业的抽象,计算任务代表真实的执行过程,消息传递作为抽象任务与计算任务之间的控制纽带。采用主从结构模式,主节点上的调度器根据抽象任务的拓扑结构来控制计算任务在各个从集群节点上执行,实用性好。

消息对象

```
Parameter{
mode:client;
classpath:/usr/bin/jdk;
Username:test
…}
msg:mysort.compute
stagein:/home/test
stageout:/home/out/test
State{
Done:0-10;
Error:other
}
```

计算对象

```
public class mysort  {
    public static int compute( ){
        FileReader in new FileReader(file);
        List<String> list = new ArrayList<>();
        BufferedReader br = new BufferedReader(in);
        while (br.ready()) {
            list.add(br.readLine());
        }
        ……
         return ret;
    }
}
```

CN 106681820 B

1.一种基于消息组合的可扩展大数据计算方法,其特征在于包括以下步骤:

步骤一、抽象的大数据计算作业拓扑结构的建立:首先将大数据计算作业分割成一系列的小任务,小任务之间使用定制的顺序、循环、条件选择以及并行结构来组合;顺序、循环、条件选择以及并行结构用来控制小任务执行过程;采用文本文件方式建立大数据作业;

步骤二、大数据计算作业的部署:用户将抽象任务对应的计算任务部署到集群节点上;

步骤三、大数据计算作业中计算任务执行计划的生成:用户的抽象计算任务被提交到管理节点;管理节点检查大数据计算定义文件的语法以及语义,形成作业计划并保存;

步骤四、大数据计算任务的调度:大数据计算作业引擎获得任务执行计划,形成一个一个的消息发送序列;消息发送计划按照并发、顺序以及条件关系形成不同的阶段,同一阶段内的消息同时发送;上一个阶段的回复消息到达后,开启下一个阶段的消息发送;

步骤五、计算任务执行过程的负载均衡:任务分配模块根据任务之间的依赖关系,对消息发送的顺序进行优化,选择最优的任务执行节点来启动计算对象;管理节点根据各个工作节点的负载进行负载均衡,以提高性能;每个工作节点维护分配给它的计算对象的状态,计算对象的状态又会反映到管理节点的消息对象上;

步骤六、大数据计算任务的执行:作业引擎取得各个任务执行需要的参数后,就向执行节点发送执行请求;执行节点在收到请求后,开始执行计算对象包含的方法,同时向作业引擎返回计算对象状态为R,作业引擎将状态记录到系统共享缓冲区中;一旦计算对象执行完成,计算对象通过集群节点向主控节点返回Do状态或者De状态,并将计算结果也返回到主控节点;主控节点的引擎得到计算对象的计算结果后,更改抽象对象的状态,进行下一条请求的发送;其中,R代表运行状态,Do代表正常终了状态,De代表错误终了状态;

步骤七、大数据计算作业消亡:当作业执行完毕,或执行条件已不满足,或作业被用户取消,该作业将从系统中撤离,作业生命结束。

基于消息组合的可扩展大数据计算方法

技术领域

[0001] 本发明涉及一种大数据计算方法，特别涉及一种基于消息组合的可扩展大数据计算方法。

背景技术

[0002] 文献“架构大数据:挑战、现状与展望，计算机学报，2011，Vol34(10)，p1741-175”公开了目前存在的三种重要的大数据计算模型:Hadoop的MapReduce计算模式;综合批处理计算、流式计算、迭代计算和图计算的Spark系统;以及基于内存计算的计算模式。Hadoop在批处理领域应用的非常好，而在图数据计算方面性能较差;Spark系统作为一个混合计算模式;基于内存的计算对于流式数据性能很高。但是，这些计算模式在某些领域尚存在一些不足之处:(1)编程一旦完成，计算规模基本确定，无法动态扩展;(2)一旦用户的大数据处理程序做一个小的变动，就需要重新编译整个计算过程;(3)任务之间的拓扑结构由数据关系决定，用户无法自主控制;(4)计算任务的类型单一，不能直接利用已经存在的传统计算业务，无法与其它异构业务混合。

发明内容

[0003] 为了克服现有大数据计算方法实用性差的不足，本发明提供一种基于消息组合的可扩展大数据计算方法。该方法首先建立抽象的大数据计算作业拓扑结构，再部署大数据计算作业，生成大算数据计算作业中计算任务执行计划，调度大数据计算任务，进行计算任务执行过程的负载均衡，执行大数据计算任务。当作业执行完毕，或执行条件已不满足，或作业被用户取消，该作业将从系统中撤离，作业生命结束。由于将计算任务的拓扑结构与计算任务分割，拓扑结构定义大数据计算作业的抽象，计算任务代表真实的执行过程，消息传递作为抽象任务与计算任务之间的控制纽带。采用主从结构模式，主节点上的调度器根据抽象任务的拓扑结构来控制计算任务在各个从集群节点上执行，实用性好。

[0004] 本发明解决其技术问题所采用的技术方案:一种基于消息组合的可扩展大数据计算方法，其特点是包括以下步骤:

[0005] 步骤一、抽象的大数据计算作业拓扑结构的建立:首先将大数据计算作业分割成一系列的小任务，小任务之间使用定制的顺序、循环、条件选择以及并行结构来组合。顺序、循环、条件选择以及并行结构用来控制小任务执行过程。采用文本文件方式建立大数据作业。

[0006] 步骤二、大数据计算作业的部署:用户将抽象任务对应的计算任务部署到集群节点上。

[0007] 步骤三、大算数据计算作业中计算任务执行计划的生成:用户的抽象计算任务被提交到管理节点。管理节点检查大数据计算定义文件的语法以及语义，形成作业计划并保存。

[0008] 步骤四、大数据计算任务的调度:大数据计算作业引擎获得任务执行计划，形成一

个一个的消息发送序列。消息发送计划按照并发、顺序以及条件关系形成不同的阶段，同一阶段内的消息同时发送。上一个阶段的回复消息到达后，开启下一个阶段的消息发送。

[0009] 步骤五、计算任务执行过程的负载均衡：任务分配模块根据任务之间的依赖关系，对消息发送的顺序进行优化，选择最优的任务执行节点来启动计算对象。管理节点也可以根据各个工作节点的负载进行负载均衡，以提高性能。每个工作节点维护分配给它的计算对象的状态，计算对象的状态又会反映到管理节点的消息对象上。

[0010] 步骤六、大数据计算任务的执行：作业引擎取得各个任务执行需要的参数后，就向执行节点发送执行请求。执行节点在收到请求后，开始执行计算对象包含的方法，同时向作业引擎返回计算对象状态为R，作业引擎将状态记录到系统共享缓冲区中。一旦计算对象执行完成，计算对象通过集群节点向主控节点返回D状态或者E状态，并将计算结果也返回到主控节点。主控节点的引擎得到计算对象的计算结果后，更改抽象对象的状态，进行下一条请求的发送。

[0011] 步骤七、大数据计算作业消亡：当作业执行完毕，或执行条件已不满足，或作业被用户取消，该作业将从系统中撤离，作业生命结束。

[0012] 本发明的有益效果是：该方法首先建立抽象的大数据计算作业拓扑结构，再部署大数据计算作业，生成大算数据计算作业中计算任务执行计划，调度大数据计算任务，进行计算任务执行过程的负载均衡，执行大数据计算任务。当作业执行完毕，或执行条件已不满足，或作业被用户取消，该作业将从系统中撤离，作业生命结束。由于将计算任务的拓扑结构与计算任务分割，拓扑结构定义大数据计算作业的抽象，计算任务代表真实的执行过程，消息传递作为抽象任务与计算任务之间的控制纽带。采用主从结构模式，主节点上的调度器根据抽象任务的拓扑结构来控制计算任务在各个从集群节点上执行，实用性好。测试表明，减少了企业的IT基础设施投入的成本达到40％，减少企业人工操作成本30％以上。

[0013] 下面结合附图和具体实施方式对本发明作详细说明。

附图说明

[0014] 图1为消息对象和计算对象样例；

[0015] 图2为大数据计算作业的图形化表示；

[0016] 图3为一个大数据计算作业的定义文件；

[0017] 图4为大数据计算方法的计算模型；

[0018] 图5为一个强连通图的例子；

[0019] 图6为大数据计算方法的系统总体结构；

[0020] 图7为K-means计算模型样例；

具体实施方式

[0021] 参照图1-7。本发明基于消息组合的可扩展大数据计算方法具体步骤如下：

[0022] 1.与本发明有关的定义。

[0023] 定义1.消息对象：参照图1左侧部分，消息对象包含了parameter,stagein,stageout等与计算相关的固定的属性，是一个具体计算的抽象。一个消息对象的描述如下：

[0024] o＝(m,parameter,stagein,stageout,code,state)

[0025] 其中,m表示消息名,消息用来控制一个可执行组件或模块的执行。parameter是消息的输入参数。Stagein是可执行组件或模块执行时需要的输入数据描述。Stageout是可执行组件或模块的结果输出描述。code和state是输出参数。code是接收的可执行组件或模块的返回值。state是消息对象的执行状态,o.state＝{W,R,D},准备发送消息时,state为W;向可执行组件或模块发送消息后,state为R;可执行组件或模块执行结束后,state为终了态D。如果可执行组件或模块正确返回,状态是Do,反之,状态就是De。

[0026] 定义2.计算对象:可执行组件或模块称为计算对象,它也可以是一个完整的程序或者简单的命令。

[0027] 参照图1右侧部分,该计算对象是mysort类的一个实例,该类中包含了一个名为compute()的静态方法,实际上一个计算对象可能包含1个或者k个方法,这些方法由程序的逻辑来决定,计算对象的所有方法全部执行完后,消息对象的状态就为终了态。当然计算过程中任何一个方法发生错误,就会导致计算对象的终了状态为De。

[0028] 计算对象在进行一次计算之后,可能产生一个输出,这个输出用来作为其它计算对象或者自身进入下次计算的输入。

[0029] 定义3.并行对象:可以触发多个消息对象进入等待状态。它用来控制多个消息对象进行并行的发送消息,其自身不向执行组件发送消息。

[0030] 定义4.跳转对象:转跳对象常常用来控制某些消息对象的循环执行。

[0031] 定义5.条件对象:条件对象是并发对象的特例。同时触发多个抽象对象或者消息的状态变化,满足条件的消息对象进入等待状态,不满足条件的消息对象直接进入终了状态。

[0032] 定义6.消息的组合结构:大数据计算作业包含大量的计算对象,因此就对应着大量的消息对象。如何组织这些消息,是一个重要的问题。参照表1,为了表现消息的交互过程,采用顺序、循环、条件选择以及并行来定义消息的组合结构,用来控制消息对象的消息交互顺序。

[0033] 表1消息的组合结构

[0034]

结构类型	表示	说明
顺序结构	MSG o1; parameter; stagein; stageout; code; state MSG o2; parameter; stagein; stageout; code; state MSG o3; parameter; stagein; stageout; code; state	表示消息之间按照时间顺序或者其它的次序要求，一条消息结束后才能启动下一条消息。
并行结构	Para{MSG o1 ... }{MSG o2 ... }{MSG o3 ... }	表示有并行处理的一系列消息。采用关键字 para 表示并行，每一对“{}”之间表明一个分支。下面的各分支分别包含一条消息 o1，o2 和 o3。
选择结构	If（code）{ MSG o1 ... }{ MSG o2 ... }	表示只有在给定的条件满足的时候才能执行的一系列消息。采用关键字 if 代表选择，code 代表警戒条件，每一对“{}”之间表明真/假分支。下面消息 o1 包含在真分支中，o2 在假分支中。

[0035]

跳转结构	MSG o1 … MSG o2 … MSG o3 … Goto o1 number	表示循环执行的一系列消息。采用 goto 来控制消息 o1，o2 和 o3 的循环执行。Goto 后面是转跳的目标消息名，number 表示循环的次数。

[0036] 定义7.大数据作业:大数据计算作业通常由大量的可执行组件组成。可执行组件之间按照一定的规则进行消息交互,最终实现用户的计算要求。因此,大数据计算作业就可以抽象为消息交互的集合,由消息的组合结构来控制消息的发送顺序。针对每个具体的大数据计算作业,可以在分布式机群上部署可执行计算组件,在统一管理节点上部署消息对象的组合。

[0037] 统一管理节点上部署消息对象的组合,可以采用文本文件的形式描述,称为作业定义文件。作业定义文件中包含消息对象、顺序对象、转跳对象、分支对象以及并行对象,其中顺序对象、转跳对象、分支对象以及并行对象总称为控制对象。消息对象之间通过控制对象形成一个大数据计算作业的定义文件。在定义文件中,每个消息对象及控制对象被唯一的字符串所标识。

[0038] 大数据作业的定义文件部署在一个管理节点上,而计算对象被分散到不同的工作节点上。作业执行时,调度引擎解析定义文件,根据消息对象和控制对象之间的逻辑关系,由消息对象向对应的计算对象发送消息,控制对象决定消息的发送次序。当所有的消息对象都发送完消息后,一次计算就结束。如果用户需要再次执行本次运算,可以再次启动消息发送。

[0039] 2.本发明的实现结构。

[0040] (1)大数据作业定义文件举例。

[0041] 图2描述了一个图形化表示的大作业,其中6个消息对象用圆圈加数字表示,(p1,p2)是一个并行对象,(f1,f2)是一个条件对象,g(3,2)是一个转跳对象。

[0042] 图3是上述大数据计算作业的定义文件，其中MSG是消息对象的关键字，Para是并行对象关键字，If是条件对象的关键字，它包含一个警戒条件condition，condition可以通过外部设置，也可以是If对象的先前对象的返回值表达式；满足警戒条件，执行第一个分支，否则执行第二个分支。Goto是转跳对象的关键字。成对的“{}”是代表分割消息组合的关键字。

[0043] (2)计算模型的结构。

[0044] 消息对象的状态从等待W、运行R到终了D的变迁过程叫一次计算。图4表示了一次计算的执行过程：一次计算过程是消息对象从W状态到D状态的变化，即(start,W)(s,R)(e,D)的序列。当消息对象接收到start事件时，进入W状态；处于W的状态的消息对象接收到s事件，它就向计算对象发送一条消息调用，自身状态变为R。计算对象接收到消息后，开始执行，运行结束后将返回消息发送给消息对象，消息对象接收到计算对象的返回值后，其状态被设置为D状态。如果计算对象的返回值在消息对象指定的范围内，状态为正常终了，即Do状态，否则，状态就被设置为错误终了，即De状态。

[0045] 一个计算对象可能包含1个或者k个方法，这些方法由程序的逻辑来决定，计算对象的所有方法全部执行完后，消息对象的状态就为终了态。当然计算过程中任何一个方法发生错误，就会导致计算对象的终了状态为De。

[0046] 计算对象在进行一次计算之后，可能产生一个输出，这个输出用来作为其它计算对象或者自身进入下次计算的输入。

[0047] 图5是一个计算一个强连通图的例子，以此来大致描述一下计算模型。图5中的每个顶点有一个值。顶点A、B、C、D之间的连线代表图的边，顶点的数字代表图的顶点值。通过传播顶点的最大值到图的每个顶点的策略，来检查图的连通性。顶点A、B、C、D代表了4个计算对象，被部署在不同的工作节点上。S代表A、B、C、D的消息对象集合，它位于一个管理节点上，从S发出的实心线代表S中的消息对象向计算对象A、B、C、D发送消息，虚线代表计算时顶点值的传递。在这个计算模型中，S的4个消息对象按照并行的方式组合，并发的4条消息到达计算对象后，代表一次计算。图中的返回消息省略。

[0048] 在每一次的计算过程中，S向各个计算对象发送消息，计算对象接收到消息后经过计算，将结果返回到S，当顶点A、B、C、D的返回值不再发生变化的时候，计算结束，算法终止。图5的(a)中，S向计算对象A发送一条计算消息，A计算后，得到最大值为6，更改自身的值并返回消息；S同时向B、C、D也发送消息，各个顶点代表的计算对象进行了一次计算，其自身的状态变为图5(b)。图5中的虚线代表计算对象的输入关系。经过4次计算以后，图5的(c)和(d)中代表的顶点值不再发生变化，计算过程结束，得到了一个连通图。使用该方法进行大数据计算，在编程完成后，即计算业务确定之后，用户通过修改大数据作业定义文件，不仅可以将计算规模进行动态扩展，比如增加更多的计算节点，而且能够控制各个任务之间的拓扑结构，重复利用已有的计算业务，提高了大数据计算的拓扑结构灵活性以及计算作业与描述的独立性。

[0049] 图6是该计算模型的总体结构，包括两种类型的工作单元：管理节点及多个工作节点。管理节点负责作业的调度以及任务的分配。整个系统支持HDFS或者运行其它的存储系统，也可以是本地文件系统，主要用于数据的持久化。

[0050] (3)输入/输出数据。

[0051] 输入数据包括作业定义文件、计算对象以及计算数据。作业定义文件采用文本方式编写，存储在主控节点的本地文件系统中。计算对象可以是JAVA格式的字节码，也可以是其它二进制文件，或者一条SQL命令，它们可以保存在计算节点的本地文件系统中或者在HDFS文件系统中。计算对象包含的数据则采用本地文件或者分布式文件系统存储，但是工作节点上的计算对象能够访问这些数据。

[0052] 计算对象的输出信息保存在工作节点的本地磁盘或者HDFS中。日志信息包含消息对象和计算对象的开始结束时间，状态变化事件等。

[0053] (4)管理节点和工作节点的实现。

[0054] 管理节点通过向工作节点发送指令来协调工作节点之间的工作，具体包括：①启动一个新的计算；②终止计算；③计算对象的状态反馈；④查询状态等。管理节点等待所有工作节点的消息，并指导工作节点下一步要做什么。因此管理节点也控制计算的同步，在每个计算开始的时候，管理节点都会发送启动消息。

[0055] 工作节点存储计算对象的结果、状态标志，并维护当前和下一次计算所需要的消息队列。每个工作节点由3个线程组成：

[0056] ①一个计算线程用于对工作节点中的计算对象的执行，它还维护一个输出消息缓冲区。当缓冲区饱满后，它要么通过网络发送给通信线程，要么直接传递给本地消息解析线程。

[0057] ②通信线程用于发送和接受缓冲区中的消息，也用于简单地协调管理节点和工作节点之间的消息。当一个消息缓冲区接收到消息时，它将消息传递到解析器线程。

[0058] ③消息解析器线程对输入消息缓冲区中的消息进行解析，已发送消息放到相应计算线程的接受消息队列中去，用于下一次的执行。

[0059] (5)工作节点上的计算对象。

[0060] 在计算对象的compute()函数内部，可以访问外部的输入数据、配置文件和一个全局对象的操作。全局对象用于协调管理节点和工作节点之间的消息、数据共享和统计汇总。在每次计算开始，更新工作节点本地映射的对象，在计算结束的时候，计算对象向管理节点通知结束状态。

[0061] 3.本发明的实现样例。

[0062] 采用基于消息传递的大数据计算模型，对K-means计算过程进行了改进，算法步骤如下：

[0063] (1)从数据中随机抽取出k个点作为初始聚类中心，由这个中心代表各个聚类；

[0064] (2)原始数据分散到m个计算节点，计算其到这k个聚类中心的距离，将点归到离其最近的聚类中心；

[0065] (3)调整聚类中心。将聚类的中心移到聚类的几何中心；

[0066] (4)重复第2)步，直到聚类中心不再变化，此时算法收敛。

[0067] 表2是改进的K-means算法在不同数目计算节点上运行的加速比。

[0068] 表2 K-means算法的加速比

[0069]

改进的K-means算法	数据量				
	$1*10^4$	$1*10^5$	$1*10^6$	$2*10^6$	$4*10^6$
3台机器	0.05	1.61	2.37	2.84	2.86
5台机器	0.08	1.67	3.51	4.29	4.73
8台机器	0.09	1.71	4.37	6.03	7.13

[0070] 图7是实现K-means改进算法的过程,其中task2是一个并行对象,task4是一个转跳对象,其它都是消息对象。

消息对象

```
Parameter{
mode:client;
classpath:/usr/bin/jdk;
Username:test
…}
msg:mysort.compute
stagein:/home/test
stageout:/home/out/test
State{
Done:0-10;
Error:other
}
```

计算对象

```
public class mysort  {
    public static int compute( ){
       FileReader in new FileReader(file);
       List<String> list = new ArrayList<>();
       BufferedReader br = new BufferedReader(in);
       while (br.ready()) {
            list.add(br.readLine());
       }
       ……
        return ret;
   }
}
```

图1

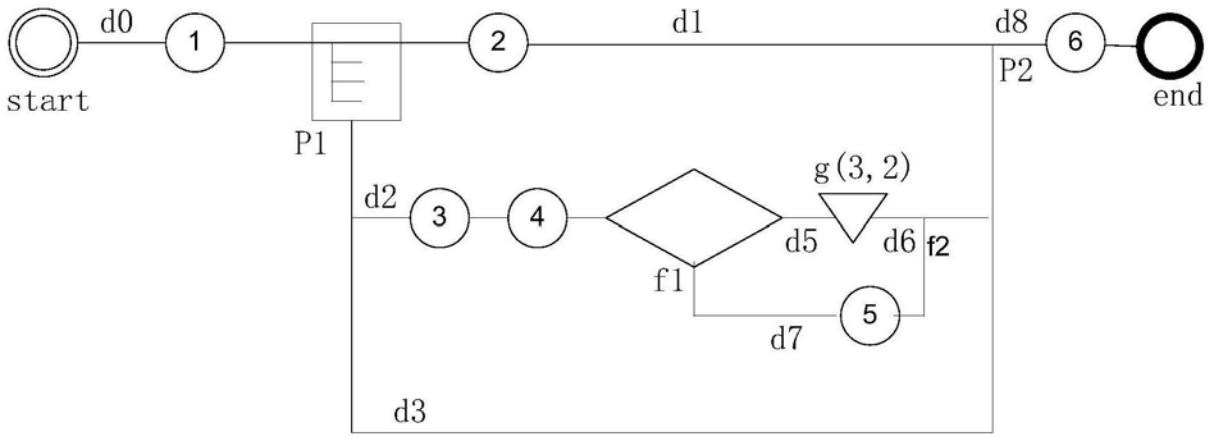

图2

```
MsgObj{
    MSG m=o1;msg=serv.count(100);;;;
    Para { MSG m=o2; msg=serv.count(100);;;;}
        { MSG m=o3; msg=serv.count(100);;;;
         MSG m=o4; msg=serv.count(100);;;;
         If (cond){ Goto o3,2}
                { MSG m=o5; msg=serv.count(100);;;;}
        }
        { }
    MSG m=o6; msg=serv.count(100);;;;
}
```

图3

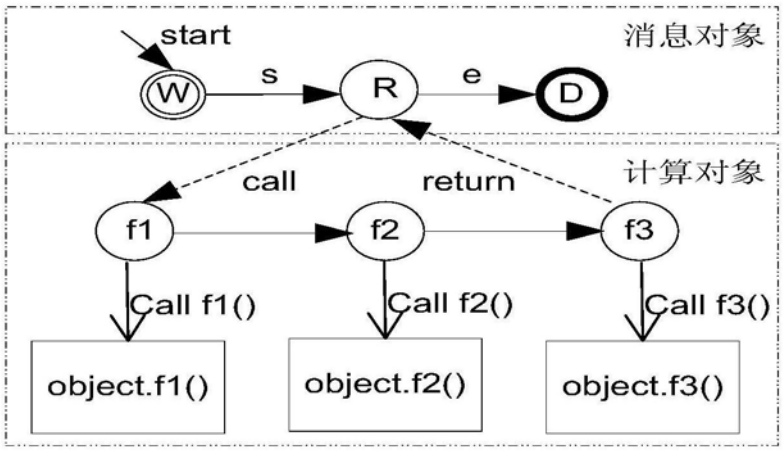

图4

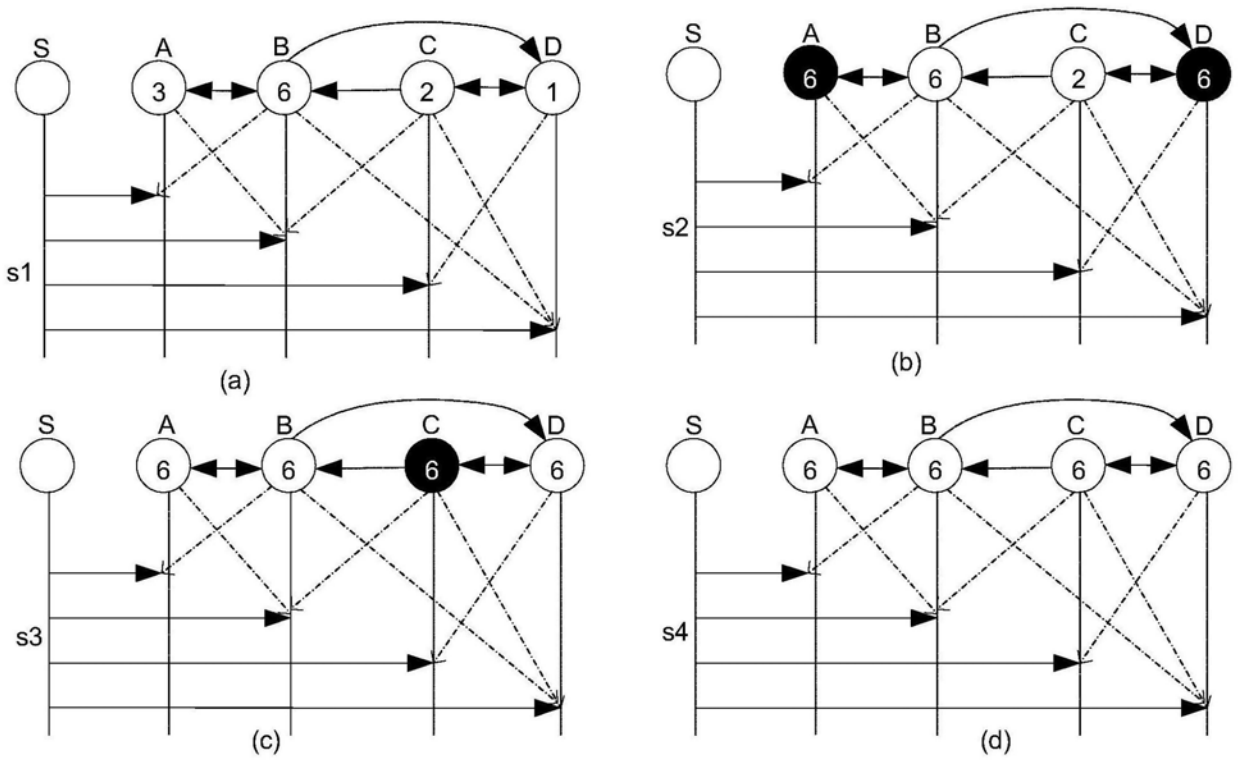
S
A
B
C
D
3
6
2
1
s1
(a)
6
6
2
6
s2
(b)
6
6
6
6
s3
(c)
6
6
6
6
s4
(d)

图5

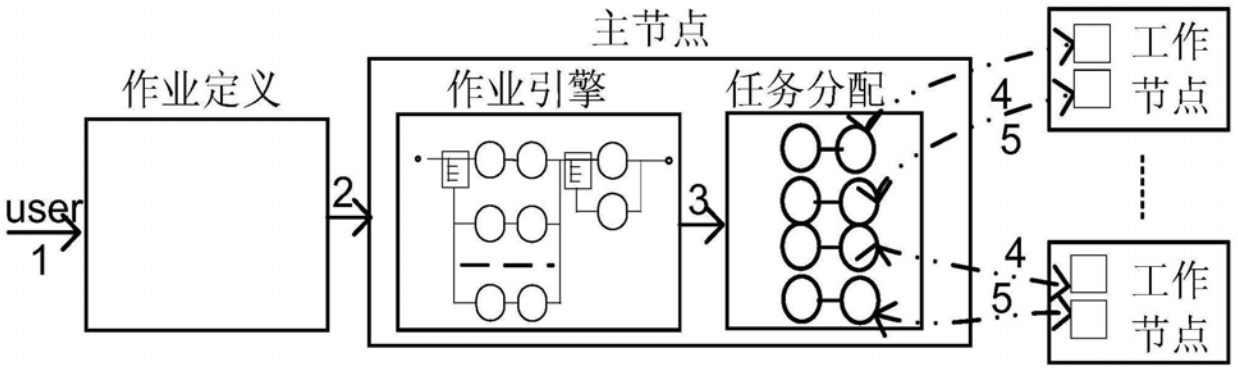
主节点
作业定义
作业引擎
任务分配
user
1
2
3
4
5
4
5
工作
节点
工作
节点

图6

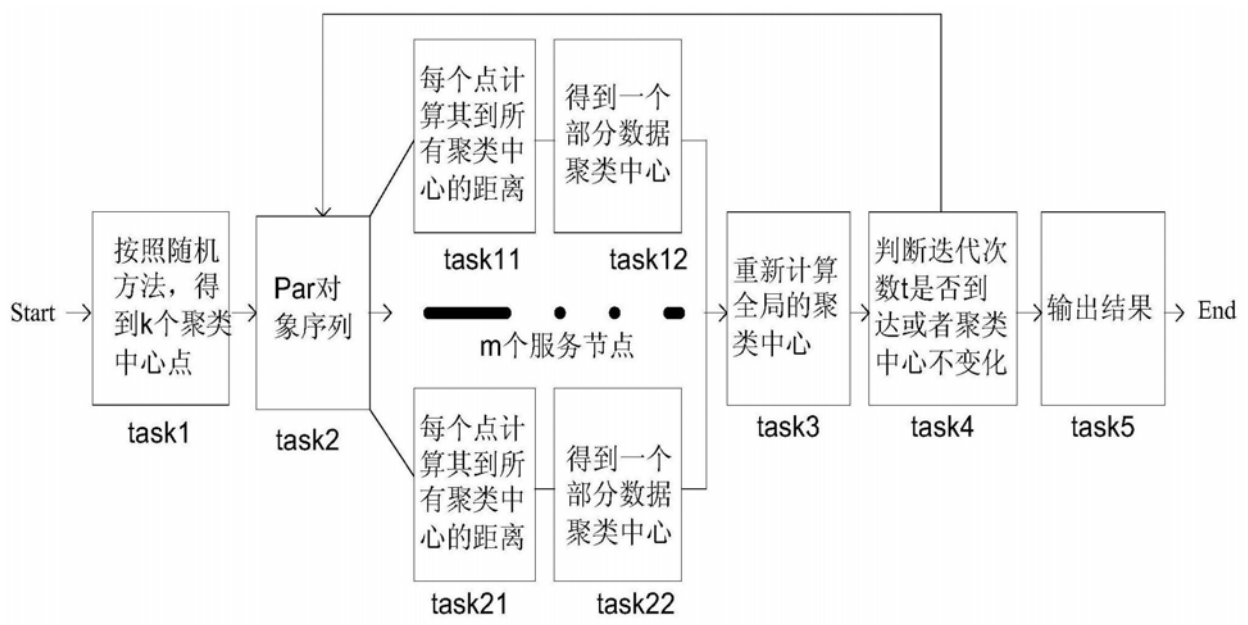
Start
按照随机方法，得到k个聚类中心点
task1
Par对象序列
task2
每个点计算其到所有聚类中心的距离
task11
得到一个部分数据聚类中心
task12
m个服务节点
每个点计算其到所有聚类中心的距离
task21
得到一个部分数据聚类中心
task22
重新计算全局的聚类中心
task3
判断迭代次数t是否到达或者聚类中心不变化
task4
输出结果
task5
End

图7