



(12)发明专利申请

(10)申请公布号 CN 107291550 A

(43)申请公布日 2017. 10. 24

(21)申请号 201710481071.1

(22)申请日 2017.06.22

(71)申请人 华中科技大学

地址 430074 湖北省武汉市洪山区珞喻路1037号

(72)发明人 王芳 冯丹 李源

(74)专利代理机构 华中科技大学专利中心 42201

代理人 廖盈春 李智

(51) Int. Cl.

G06F 9/50(2006.01)

权利要求书3页 说明书8页 附图4页

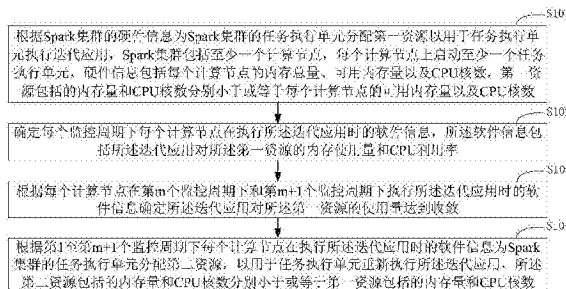
(54)发明名称

一种针对迭代应用的Spark平台资源动态分配方法及系统

(57)摘要

本发明公开了一种针对迭代应用的Spark平台资源动态分配方法及系统,包括:根据Spark集群的硬件信息为Spark集群的任务执行单元分配第一资源以用于任务执行单元执行迭代应用;确定每个监控周期下每个计算节点在执行所述迭代应用时的软件信息;根据每个计算节点在第m个监控周期下和第m+1个监控周期下执行迭代应用时的软件信息确定所述迭代应用对所述第一资源的使用量达到收敛;根据第1至第m+1个监控周期下每个计算节点在执行所述迭代应用时的软件信息为Spark集群的任务执行单元分配第二资源,以用于任务执行单元重新执行所述迭代应用。本发明在保证迭代应用正常而高效运行的同时,可以自动释放其占用的冗余系统资源,提高系统的整体资源利用率与应用的并发度。

CN 107291550 A



1. 一种针对迭代应用的Spark平台资源动态分配方法,其特征在于,包括:

根据Spark集群的硬件信息为Spark集群的任务执行单元分配第一资源以用于任务执行单元执行迭代应用,所述Spark集群包括至少一个计算节点,每个计算节点上启动至少一个任务执行单元,所述硬件信息包括每个计算节点的内存总量、可用内存量以及CPU核数,所述第一资源包括的内存量和CPU核数分别小于或等于每个计算节点的可用内存量和CPU核数;

确定每个监控周期下每个计算节点在执行所述迭代应用时的软件信息,所述软件信息包括所述迭代应用对所述第一资源的内存使用量和CPU利用率;

根据每个计算节点在第m个监控周期下和第m+1个监控周期下执行所述迭代应用时的软件信息确定所述迭代应用对所述第一资源的使用量达到收敛,m为正整数;

根据第1至第m+1个监控周期下每个计算节点在执行所述迭代应用时的软件信息为Spark集群的任务执行单元分配第二资源,以用于所述任务执行单元重新执行所述迭代应用,所述第二资源包括的内存量和CPU核数分别小于或等于第一资源包括的内存量和CPU核数。

2. 根据权利要求1所述的Spark平台资源动态分配方法,其特征在于,根据每个计算节点在第m个监控周期下和第m+1个监控周期下执行所述迭代应用时的软件信息确定所述迭代应用对所述第一资源的使用量达到收敛,包括:

若每个计算节点从第m个监控周期到第m+1个监控周期执行所述迭代应用时对所述第一资源所包括内存的使用量趋于稳定,则所述迭代应用对所述第一资源的使用量达到收敛。

3. 根据权利要求2所述的Spark平台资源动态分配方法,其特征在于,每个计算节点从第m个监控周期到第m+1个监控周期执行所述迭代应用时对所述第一资源的内存使用量趋于稳定,包括:

若每个计算节点从第m个监控周期到第m+1个监控周期执行所述迭代应用时内存使用量变化率满足如下公式,则每个计算节点对所述第一资源的内存使用量趋于稳定:

$$\delta_i < \alpha$$

其中, δ_i 表示计算节点i从第m到第m+1个监控周期下执行所述迭代应用时的内存使用量变化率,i表示计算节点的编号, α 表示预设变化率阈值;

δ_i 通过以下公式确定:

$$\delta_i = (\text{MEM}_{i(m+1)} - \text{MEM}_{im}) / \text{MEM}_{im}$$

其中, MEM_{im} 和 $\text{MEM}_{i(m+1)}$ 分别表示计算节点i在第m个监控周期下和第m+1个监控周期下执行所述迭代应用时的内存使用量。

4. 根据权利要求2所述的Spark平台资源动态分配方法,其特征在于,根据第1至第m+1个监控周期下每个计算节点在执行所述迭代应用时的软件信息为Spark集群的任务执行单元分配第二资源,通过以下公式确定:

$$\text{MEM}_{\text{sug}} = \lceil (1 + \beta_1) * \text{MEM}_{\text{max}} \rceil$$

$$\text{CPU}_{\text{sug}} = \lceil (1 + \beta_2) * \text{CPU}_{\text{max}} * \text{CPU_Core_NUM} \rceil$$

其中, MEM_{sug} 表示第二资源包括的内存量, CPU_{sug} 表示第二资源包括的CPU核数, β_1 和 β_2

2分别为内存量和CPU核数的资源需求浮动因子, MEM_{max}表示第m+1个监控周期下所有计算节点在执行所述迭代应用时内存使用量中的最大值, CPU_{max}表示从第1个监控周期到第m+1个监控周期中所有计算节点在执行所述迭代应用时CPU利用率中的最大值, CPU_Core_NUM表示每个计算节点的CPU核数。

5. 根据权利要求3所述的Spark平台资源动态分配方法, 其特征在于, MEM_m和MEM_{i (m+1)}分别通过以下公式确定:

$$\text{MEM}_{m} = (\text{MEM_USED}'_{im} - \text{MEM_USED}_{i})$$

$$\text{MEM}_{i (m+1)} = (\text{MEM_USED}'_{i (m+1)} - \text{MEM_USED}_{i})$$

其中, MEM_USED_i表示计算节点i无应用执行时的内存使用量, MEM_USED'_{im}与MEM_USED'_{i (m+1)}分别表示计算节点i在第m和第m+1个监控周期的内存总使用量, MEM_m和MEM_{i (m+1)}分别表示计算节点i在第m和第m+1个监控周期执行迭代应用时的内存使用量。

6. 一种针对迭代应用的Spark平台资源动态分配系统, 其特征在于, 包括:

第一资源分配单元, 用于根据Spark集群的硬件信息为Spark集群的任务执行单元分配第一资源以用于任务执行单元执行迭代应用, 所述Spark集群包括至少一个计算节点, 每个计算节点上启动至少一个任务执行单元, 所述硬件信息包括每个计算节点的内存总量、可用内存量以及CPU核数, 所述第一资源包括的内存量和CPU核数分别小于或等于每个计算节点的可用内存量以及CPU核数;

软件信息确定单元, 用于确定每个监控周期下每个计算节点在执行所述迭代应用时的软件信息, 所述软件信息包括所述迭代应用对所述第一资源的内存使用量和CPU利用率;

需求收敛确定单元, 用于根据每个计算节点在第m个监控周期下和第m+1个监控周期下执行所述迭代应用时的软件信息确定所述迭代应用对所述第一资源的使用量达到收敛;

第二资源分配单元, 用于根据第1至第m+1个监控周期下每个计算节点在执行所述迭代应用时的软件信息为Spark集群的任务执行单元分配第二资源, 以用于所述任务执行单元重新执行所述迭代应用, 所述第二资源包括的内存量和CPU核数分别小于或等于第一资源包括的内存量和CPU核数。

7. 根据权利要求6所述的Spark平台资源动态分配系统, 其特征在于, 需求收敛确定单元, 用于若每个计算节点从第m个监控周期到第m+1个监控周期执行所述迭代应用时对所述第一资源所包括内存的使用量趋于稳定, 则确定所述迭代应用对所述第一资源的使用量达到收敛。

8. 根据权利要求7所述的Spark平台资源动态分配系统, 其特征在于, 需求收敛确定单元, 用于若每个计算节点从第m个监控周期到第m+1个监控周期的内存使用量变化率满足如下公式, 则确定每个计算节点对所述第一资源的内存使用量趋于稳定:

$$\delta_i < \alpha$$

其中, δ_i 表示计算节点i从第m到第m+1个监控周期下执行所述迭代应用时的内存使用量变化率, i表示计算节点的编号, α 表示预设变化率阈值;

δ_i 通过以下公式确定:

$$\delta_i = (\text{MEM}_{i (m+1)} - \text{MEM}_{im}) / \text{MEM}_{im}$$

其中, MEM_m和MEM_{i (m+1)}分别表示计算节点i在第m个监控周期下和第m+1个监控周期下执行所述迭代应用时的内存使用量。

9. 根据权利要求7或8所述的Spark平台资源动态分配系统,其特征在于,第二资源分配单元,用于通过以下公式确定第二资源:

$$\text{MEMsug} = \lceil (1 + \beta 1) * \text{MEMmax} \rceil$$

$$\text{CPUusug} = \lceil (1 + \beta 2) * \text{CPUmax} * \text{CPU_Core_NUM} \rceil$$

其中,MEMsug表示第二资源包括的内存量,CPUusug表示第二资源包括的CPU核数, $\beta 1$ 和 $\beta 2$ 分别为内存量和CPU核数的资源需求浮动因子,MEMmax表示第m+1个监控周期下所有计算节点在执行所述迭代应用时内存使用量中的最大值,CPUmax表示从第1个监控周期到第m+1个监控周期中所有计算节点在执行所述迭代应用时CPU利用率中的最大值,CPU_Core_NUM表示每个节点的CPU核数。

10. 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上存储有计算机程序,所述计算机程序被处理器执行时实现如权利要求1至5任一项所述的Spark平台资源动态分配方法。

一种针对迭代应用的Spark平台资源动态分配方法及系统

技术领域

[0001] 本发明属于大数据技术领域,更具体地,涉及一种针对迭代应用的Spark平台资源动态分配方法及系统。

背景技术

[0002] 随着“互联网+”时代的来临,大数据日趋成为现今各行各业的热门话题。如何对海量的数据进行计算处理,使其价值最大化,是人类面临一个非常重大的挑战。AMP实验室提出了一种分布式内存抽象,称为弹性分布式数据集(RDD,Resilient Distributed Datasets),RDD允许用户显式地把工作集缓存在内存中,因此在未来重用时能够极大地提升速度。

[0003] AMP实验室在Spark系统中实现了RDD,并使用Spark来开发各种并行应用。Spark有诸多优异的特性:Spark最大的优点是能够将中间结果保存在内存中,计算速度比Hadoop MapReduce快100倍以上;Spark便于使用,如用户能够用Java、Scala、Python和R语言快速地编写应用程序;Spark具有通用性,能够在其上运行SQL查询、流计算以及机器学习和图计算等复杂的计算分析,同时Spark能够以多种模式运行,并能够从HDFS、Cassandra、HBase等多种数据流或文件系统中读取数据。

[0004] 应用程序提交到Spark集群后,会根据其中的action算子,将应用程序划分为多个Job,每个Job根据RDD的依赖关系划分为多个Stage,每个stage就是一个任务集,再分配到集群各个计算节点进行执行。Spark系统往往会有一个主节点(Master)以及一个或多个计算节点(Worker),应用运行时,会在Worker节点上启动一个或多个任务执行单元(Executor),Executor是Spark系统的任务执行单元。在Spark系统上启动一个应用程序后,默认的资源分配策略,会在每个Worker上启动一个Executor,并为每个Executor分配1GB内存以及全部的CPU资源。

[0005] 但是,默认的Spark资源分配策略是一种静态的方法,一方面,当应用需要的内存较大,超出Executor的内存容量时,应用执行效率极低,甚至无法执行;另一方面,为每个Executor分配的全部CPU资源不一定能够充分利用,可能导致CPU利用率低下,且无法在运行时释放系统CPU资源,系统中其他应用提交以后,只能等待当前应用执行完毕,释放占用的资源后才能继续执行。另外,用户可以手动配置为Executor分配的内存以及CPU资源,但是不同应用的特点不同,其对于资源的需求情况也有极大差异。同种应用当负载数据量不同时,对于资源的需求情况也不尽相同。因此,如何为Executor分配合适的资源,可能会对Spark用户带来极大的困扰。用户往往需要靠经验积累,甚至多次枚举各种配置参数组合运行应用程序,来获取针对特定应用程序的合适的资源分配量,而这种方法成本高、效率低。

[0006] 综上,Spark现有的资源分配策略是一种静态的方法,一方面可能导致应用执行效率低甚至无法执行,另一方面可能导致系统的资源利用率低下,同时如何为应用程序分配合适的资源并非易事,往往会给用户带来极大的困扰。

发明内容

[0007] 针对现有技术的缺陷,本发明的目的在于解决现有Spark资源分配策略是静态方法,可能导致应用执行效率低甚至无法执行或者系统的资源利用率低下,以及用户以手动配置Spark资源不能针对不同应用的特点分配合适资源的技术问题。

[0008] 为实现上述目的,第一方面,本发明实施例提供了一种针对迭代应用的Spark平台资源动态分配方法,包括:根据Spark集群的硬件信息为Spark集群的任务执行单元分配第一资源以用于任务执行单元执行迭代应用,所述Spark集群包括至少一个计算节点,每个计算节点上启动至少一个任务执行单元,所述硬件信息包括每个计算节点的内存总量、可用内存量以及CPU核数,所述第一资源包括的内存量和CPU核数分别小于或等于每个计算节点的可用内存量以及CPU核数;确定每个监控周期下每个计算节点在执行所述迭代应用时的软件信息,所述软件信息包括所述迭代应用对所述第一资源的内存使用量和CPU利用率;根据每个计算节点在第m个监控周期下和第m+1个监控周期下执行所述迭代应用时的软件信息确定所述迭代应用对所述第一资源的使用量达到收敛,m为正整数;根据第1至第m+1个监控周期下每个计算节点在执行所述迭代应用时的软件信息为Spark集群的任务执行单元分配第二资源,以用于所述任务执行单元重新执行所述迭代应用,所述第二资源包括的内存量和CPU核数分别小于或等于第一资源包括的内存量和CPU核数。

[0009] 本发明实施例提供的方法,在为Spark集群分配第一资源后,其迭代应用可能对第一资源的需求趋于稳定,且迭代应用仅需要第一资源中的一部分资源,则可通过自动监控对第一资源的使用情况,在对第一资源的使用收敛时,调整为Spark集群分配迭代应用实际需要的第二资源,以释放第一资源中冗余的资源,使得这些资源能够为集群上的其他应用程序利用,进而有效提高系统的整体资源利用率与应用的并发度。

[0010] 可选地,根据每个计算节点在第m个监控周期下和第m+1个监控周期下执行所述迭代应用时的软件信息确定所述迭代应用对所述第一资源的使用量达到收敛,包括:若每个计算节点从第m个监控周期到第m+1个监控周期执行所述迭代应用时对所述第一资源所包括内存的使用量趋于稳定,则所述迭代应用对所述第一资源的使用量达到收敛。

[0011] 可选地,每个计算节点从第m个监控周期到第m+1个监控周期执行所述迭代应用时对所述第一资源的内存使用量趋于稳定,包括:若每个计算节点从第m个监控周期到第m+1个监控周期的内存使用量变化率满足如下公式,则每个计算节点对所述第一资源的内存使用量趋于稳定: $\delta_i < \alpha$,其中, δ_i 表示计算节点i从第m到第m+1个监控周期下执行所述迭代应用时的内存使用量变化率,i表示计算节点的编号, α 表示预设变化率阈值; δ_i 通过以下公式确定:

$$[0012] \quad \delta_i = (\text{MEM}_i(m+1) - \text{MEM}_i(m)) / \text{MEM}_i(m)$$

[0013] 其中, $\text{MEM}_i(m)$ 和 $\text{MEM}_i(m+1)$ 分别表示计算节点i在第m个监控周期下和第m+1个监控周期下执行所述迭代应用时的内存使用量。

[0014] 可选地,根据第m+1个监控周期下每个计算节点在执行所述迭代应用时的软件信息为Spark集群的任务执行单元分配第二资源,通过以下公式确定:

$$[0015] \quad \text{MEM}_{\text{sug}} = \lceil (1 + \beta) * \text{MEM}_{\text{max}} \rceil$$

$$[0016] \quad \text{CPU}_{\text{sug}} = \lceil (1 + \beta_2) * \text{CPU}_{\text{max}} * \text{CPU_Core_NUM} \rceil$$

[0017] 其中, MEM_{sug} 表示第二资源包括的内存量, CPU_{sug} 表示第二资源包括的CPU核数, β_1 和 β_2 分别为内存量和CPU核数的资源需求浮动因子, MEM_{max} 表示第 $m+1$ 个监控周期下所有计算节点在执行所述迭代应用时内存使用量中的最大值, CPU_{max} 表示从第1个监控周期到第 $m+1$ 个监控周期中所有计算节点在执行所述迭代应用时的CPU利用率中的最大值, CPU_Core_NUM 表示每个计算节点的CPU核数。

[0018] 可选地, MEM_{im} 和 $\text{MEM}_{\text{i}(m+1)}$ 分别通过以下公式确定:

$$[0019] \quad \text{MEM}_{\text{im}} = (\text{MEM_USED}'_{\text{im}} - \text{MEM_USED}_{\text{i}})$$

$$[0020] \quad \text{MEM}_{\text{i}(m+1)} = (\text{MEM_USED}'_{\text{i}(m+1)} - \text{MEM_USED}_{\text{i}})$$

[0021] 其中, $\text{MEM_USED}_{\text{i}}$ 表示计算节点 i 无应用执行时的内存使用量, $\text{MEM_USED}'_{\text{im}}$ 与 $\text{MEM_USED}'_{\text{i}(m+1)}$ 分别表示计算节点 i 在第 m 和第 $m+1$ 个监控周期的内存总使用量, MEM_{im} 和 $\text{MEM}_{\text{i}(m+1)}$ 分别表示计算节点 i 在第 m 和第 $m+1$ 个监控周期执行迭代应用时的内存使用量。

[0022] 第二方面, 本发明实施例提供了一种针对迭代应用的Spark平台资源动态分配系统, 包括:

[0023] 第一资源分配单元, 用于根据Spark集群的硬件信息为Spark集群的任务执行单元分配第一资源以用于任务执行单元执行迭代应用, 所述Spark集群包括至少一个计算节点, 每个计算节点上启动至少一个任务执行单元, 所述硬件信息包括每个计算节点的内存总量、可用内存量以及CPU核数, 所述第一资源包括的内存量和CPU核数分别小于或等于每个计算节点的可用内存量以及CPU核数。

[0024] 软件信息确定单元, 用于确定每个监控周期下每个计算节点在执行所述迭代应用时的软件信息, 所述软件信息包括所述迭代应用对所述第一资源的内存使用量和CPU利用率。

[0025] 需求收敛确定单元, 用于根据每个计算节点在第 m 个监控周期下和第 $m+1$ 个监控周期下执行所述迭代应用时的软件信息确定所述迭代应用对所述第一资源的使用量达到收敛。

[0026] 第二资源分配单元, 用于根据第1至第 $m+1$ 个监控周期下每个计算节点在执行所述迭代应用时的软件信息为Spark集群的任务执行单元分配第二资源, 以用于所述任务执行单元重新执行所述迭代应用, 所述第二资源包括的内存量和CPU核数分别小于或等于第一资源包括的内存量和CPU核数。

[0027] 可选地, 需求收敛确定单元, 用于若每个计算节点从第 m 个监控周期到第 $m+1$ 个监控周期执行所述迭代应用时对所述第一资源所包括内存使用量趋于稳定, 则确定所述迭代应用对所述第一资源的使用量达到收敛。

[0028] 可选地, 需求收敛确定单元, 用于若每个计算节点从第 m 个监控周期到第 $m+1$ 个监控周期的内存使用量变化率满足如下公式, 则确定每个计算节点对所述第一资源的内存使用量趋于稳定:

$$[0029] \quad \delta_i < \alpha$$

[0030] 其中, δ_i 表示计算节点 i 从第 m 到第 $m+1$ 个监控周期下执行所述迭代应用时的内存使用量变化率, i 表示计算节点的编号, α 表示预设变化率阈值;

[0031] δ_i 通过以下公式确定:

[0032] $\delta_i = (\text{MEM}_i(m+1) - \text{MEM}_i(m)) / \text{MEM}_i(m)$

[0033] 其中, $\text{MEM}_i(m)$ 和 $\text{MEM}_i(m+1)$ 分别表示计算节点*i*在第*m*个监控周期下和第*m+1*个监控周期下执行所述迭代应用时的内存使用量。

[0034] 可选地, 第二资源分配单元, 用于通过以下公式确定第二资源:

[0035] $\text{MEM}_{\text{sug}} = \lceil (1 + \beta_1) * \text{MEM}_{\text{max}} \rceil$

[0036] $\text{CPU}_{\text{sug}} = \lceil (1 + \beta_2) * \text{CPU}_{\text{max}} * \text{CPU_Core_NUM} \rceil$

[0037] 其中, MEM_{sug} 表示第二资源包括的内存量, CPU_{sug} 表示第二资源包括的CPU核数, β_1 和 β_2 分别为内存量和CPU核数的资源需求浮动因子, MEM_{max} 表示第*m+1*个监控周期下所有计算节点执行所述迭代应用时内存使用量中的最大值, CPU_{max} 表示从第1个监控周期到第*m+1*个监控周期中所有计算节点执行所述迭代应用时的CPU利用率中的最大值,

[0038] CPU_Core_NUM 表示每个计算节点的CPU核数。

[0039] 第三方面, 本发明实施例提供了一种计算机可读存储介质, 该计算机可读存储介质上存储有计算机程序, 所述计算机程序被处理器执行时实现上述第一方面所述的Spark平台资源动态分配方法。

[0040] 总体而言, 通过本发明所构思的以上技术方案与现有技术相比, 具有以下有益效果:

[0041] (1) 本发明提供的Spark资源动态分配方法是一个完全自动化过程, 对于用户执行应用程序来说完全是透明的, 用户无需了解底层设计, 无需与任何界面或接口进行交互, 大大降低了用户的使用门槛。

[0042] (2) 本发明解决了对于Spark集群上典型的迭代应用, 无法动态分配系统资源的问题。对于整个Spark集群系统而言, 本发明在保证该迭代应用正常而高效运行的同时, 可以释放其占用的冗余系统资源, 使得这些资源能够为集群上的其他应用程序利用, 进而有效提高系统的整体资源利用率与应用的并发度。

[0043] (3) 本发明不仅仅只适用于迭代应用, 大部分对于系统资源的需求量有上限值或者会逐渐收敛的应用, 本发明均可对其实行资源的动态分配方法, 进而提高系统的资源利用率与应用的并发度。

附图说明

[0044] 图1为本发明实施例提供的针对迭代应用的Spark平台资源动态分配方法流程示意图;

[0045] 图2为本发明实施例提供的针对迭代应用的Spark平台资源动态分配系统的架构图;

[0046] 图3为本发明实施例提供的针对迭代应用的Spark平台资源动态分配系统工作流程图;

[0047] 图4为本发明实施例提供的节点状态监测及建模评估模块的工作流程图;

[0048] 图5为本发明实施例提供的资源动态分配模块的工作流程图;

[0049] 图6为本发明实施例提供的针对迭代应用的Spark平台资源动态分配系统结构示意图。

具体实施方式

[0050] 为了使本发明的目的、技术方案及优点更加清楚明白,以下结合附图及实施例,对本发明进行进一步详细说明。应当理解,此处所描述的具体实施例仅仅用以解释本发明,并不用于限定本发明。

[0051] 图1为本发明实施例提供的针对迭代应用的Spark平台资源动态分配方法流程示意图;如图1所示,包括步骤S101至步骤S104。

[0052] S101,根据Spark集群的硬件信息为Spark集群的任务执行单元分配第一资源以用于任务执行单元执行迭代应用,所述Spark集群包括至少一个计算节点,每个计算节点上启动至少一个任务执行单元,所述硬件信息包括每个计算节点的内存总量、可用内存量以及CPU核数,所述第一资源包括的内存量和CPU核数分别小于或等于每个计算节点的可用内存量以及CPU核数。

[0053] S102,确定每个监控周期下每个计算节点在执行所述迭代应用时的软件信息,所述软件信息包括所述迭代应用对所述第一资源的内存使用量和CPU利用率。

[0054] S103,根据每个计算节点在第m个监控周期下和第m+1个监控周期下执行所述迭代应用时的软件信息确定所述迭代应用对所述第一资源的使用量达到收敛,m为正整数。

[0055] 可选地,若每个计算节点从第m个监控周期到第m+1个监控周期执行所述迭代应用时对所述第一资源的内存使用量趋于稳定,则所述迭代应用对所述第一资源的使用量达到收敛。

[0056] 可选地,若每个计算节点从第m个监控周期到第m+1个监控周期执行所述迭代应用时内存使用量变化率满足如下公式,则每个计算节点对所述第一资源的内存使用量趋于稳定: $\delta_i < \alpha$ 。

[0057] 其中, δ_i 表示计算节点i从第m到第m+1个监控周期下执行所述迭代应用时的内存使用量变化率,i表示计算节点的编号, α 表示预设变化率阈值。

[0058] δ_i 通过以下公式确定:

$$[0059] \quad \delta_i = (\text{MEM}_i(m+1) - \text{MEM}_i(m)) / \text{MEM}_i(m)$$

[0060] 其中, $\text{MEM}_i(m)$ 和 $\text{MEM}_i(m+1)$ 分别表示计算节点i在第m个监控周期下和第m+1个监控周期下执行所述迭代应用时的内存使用量。

[0061] 其中,预设变化率阈值可设为 α , α 取经验值0.05。

[0062] S104,根据第1至第m+1个监控周期下每个计算节点在执行所述迭代应用时的软件信息为Spark集群的任务执行单元分配第二资源,以用于所述任务执行单元重新执行所述迭代应用,所述第二资源包括的内存量和CPU核数分别小于或等于第一资源包括的内存量和CPU核数。

[0063] 具体地,为Spark集群分配第一资源后,其迭代应用可能对第一资源的需求趋于稳定,且仅需要第一资源中的一部分资源,则可通过调整为Spark集群分配迭代应用实际需要的第二资源,以释放第一资源中冗余的资源,使得这些资源能够为集群上的其他应用程序利用,进而有效提高系统的整体资源利用率与应用的并发度。

[0064] 可选地,根据第m+1个监控周期下每个计算节点在执行所述迭代应用时的软件信息为Spark集群的任务执行单元分配第二资源,通过以下公式确定:

$$[0065] \quad \text{MEMsug} = \lceil (1 + \beta_1) * \text{MEMmax} \rceil$$

$$[0066] \quad \text{CPUusg} = \lceil (1 + \beta_2) * \text{CPUmax} * \text{CPU_Core_NUM} \rceil$$

[0067] 其中, MEMsug表示第二资源包括的内存量, CPUusg表示第二资源包括的CPU核数, β_1 和 β_2 分别为内存量和CPU核数的资源需求浮动因子, MEMmax表示第m+1个监控周期下所有计算节点在执行迭代应用时中内存使用量中的最大值, CPUmax表示从第1个监控周期到第m+1个监控周期中所有计算节点在执行迭代应用时CPU利用率中的最大值, CPU_Core_NUM表示每个计算节点的CPU核数。

[0068] 本发明实施例对于整个Spark集群系统而言,在保证该迭代应用正常而高效运行的同时,可以释放其占用的冗余系统资源,使得这些资源能够为集群上的其他应用程序利用,进而有效提高系统的整体资源利用率与应用的并发度。

[0069] 如图2所示,本发明实施例提供的针对迭代应用的Spark平台资源动态分配系统架构为三方架构包括:客户端、Spark集群和监控服务器。其中用户在客户端提交Spark迭代应用程序,Spark集群包括一个主节点(Master)和一个或多个计算节点(Worker),主节点接受建模反馈信息以及任务的执行状态信息,负责任务调度与资源分配;计算节点接受调度信息,并在任务执行单元(Executor)中运行任务;监控服务器监控计算节点的状态信息,并反馈给主节点。

[0070] 如图3所示,本发明中,针对迭代应用的Spark平台资源动态分配系统的工作流程如下:

[0071] 步骤301,启动Spark集群,采集集群的硬件信息,监控服务器在特定端口接受、汇总集群的硬件信息,每一条硬件信息记录表示为:

$$[0072] \quad \text{Record_Hardware} = (\text{Hostname}, \text{MEM_Total}, \text{MEM_USED}, \text{MEM_AVA}, \text{CPU_Core_NUM})$$

[0073] 其中,Hostname表示该计算节点主机名, MEM_Total表示该计算节点的总内存大小, MEM_USED表示该计算节点无应用执行时的内存使用量, MEM_AVA表示该计算节点无应用执行时的可用内存大小, CPU_Core_NUM表示该计算节点的逻辑CPU核数。其中, MEM_Total = MEM_USED + MEM_AVA。

[0074] 步骤302,为Spark的任务执行单元Executor分配足够的系统资源执行迭代应用,其中足够的系统资源即为图1步骤所提及的第一资源,第一资源可以为全部的可用内存大小,即MEM_AVA,以及全部的逻辑CPU核数,即CPU_Core_NUM。第一资源也可以为MEM_AVA和CPU_Core_NUM中的部分资源。

[0075] 步骤303,主节点实时监控集群各计算节点上的迭代应用执行情况信息,即应用当前所处的迭代轮数以及当前轮次的迭代计算是否结束。其中,Spark源码中主节点的CoarseGrainedSchedulerBackend类通过调用receive函数,接收从计算节点的CoarseGrainedExecutorBackend类传回的任务执行信息,然后调用TaskSchedulerImpl类中statusUpdate方法,判断当前迭代计算任务是否执行完毕从做出相应处理,对此过程进行监控,即可获取当前轮次的节点迭代计算状态信息。

[0076] 步骤304,同时监控服务器启动节点状态监控,在特定端口定期(每隔30s)接收、汇总各个计算节点在运行迭代应用时产生的软件信息,每一条软件信息记录表示为:

$$[0077] \quad \text{Record_Software} = (\text{Hostname}, \text{Mointor_ID}, \text{MEM_USED}', \text{CPU_UTI})$$

[0078] 其中,Hostname同样表示该计算节点主机名,Moitor_ID表示该计算节点当前所处监控周期的序列号,MEM_USED'表示当前时刻该计算节点的内存使用量,CPU_UTI表示当前时刻计算节点的CPU利用率,其中,当前时刻即为当前监控周期。

[0079] 如图4所示,本发明实施例提供的节点状态监测及建模评估模块的工作流程如下:

[0080] 步骤401,监控服务器汇总、解析采集到的硬件、软件信息,计算相邻监控周期中各计算节点的内存使用量变化率,假设有n个计算节点,第m个和第m+1个监控周期的内存使用量变化率计算公式如下:

$$[0081] \quad \text{MEM}_{im} = (\text{MEM_USED}'_{im} - \text{MEM_USED}_i)$$

$$[0082] \quad \text{MEM}_{i(m+1)} = (\text{MEM_USED}'_{i(m+1)} - \text{MEM_USED}_i)$$

$$[0083] \quad \delta_i = (\text{MEM}_{i(m+1)} - \text{MEM}_{im}) / \text{MEM}_{im}$$

[0084] 其中, $i=1,2,\dots,n$,MEM_USED_i表示计算节点i无应用执行时的内存使用量,MEM_USED'_{im}与MEM_USED'_{i(m+1)}分别表示计算节点i在第m和第m+1个监控周期的内存总使用量,则MEM_{im}和MEM_{i(m+1)}分别表示计算节点i在第m和第m+1个监控周期的迭代应用的内存使用量, δ_i 表示计算节点i在第m到第m+1个监控周期的内存使用量变化率。

[0085] 步骤402,判断该迭代应用对于系统资源(第一资源)的需求是否收敛,收敛的条件是n个计算节点的内存使用量变化率都满足以下公式:

$$[0086] \quad \delta_i < \alpha$$

[0087] 其中, $i=1,2,\dots,n$, α 为收敛因子,收敛条件是所有节点在相邻两个监控周期的内存使用量变化率均小于 α , α 取经验值0.05,如不满足收敛条件,执行步骤401。若收敛,执行步骤403,其中对第一资源的需求收敛指迭代应用对第一资源的使用量趋于稳定。

[0088] 步骤403,满足收敛条件,则计算系统资源的建议分配值,采用以下公式:

$$[0089] \quad \text{MEM}_{\max} = \text{MAX}\{\text{MEM}_{i(m+1)}\}$$

$$[0090] \quad \text{CPU}_{\max} = \text{MAX}\{\text{CPU_UTI}_{ik}\}$$

$$[0091] \quad \text{MEM}_{\text{sug}} = \lceil (1 + \beta_1) * \text{MEM}_{\max} \rceil$$

$$[0092] \quad \text{CPU}_{\text{sug}} = \lceil (1 + \beta_2) * \text{CPU}_{\max} * \text{CPU_Core_NUM} \rceil$$

[0093] 其中 $i=1,2,\dots,n$, $k=1,2,\dots,m+1$, β_1 和 β_2 分别为内存和CPU的资源需求浮动因子,MEM_{i(m+1)}表示计算节点i在第m+1个监控周期执行迭代应用时的内存使用量,CPU_UTI_{ik}表示计算节点i在第k个周期的CPU利用率,MEM_{max}表示第m+1个监控周期中各计算节点的迭代应用的内存使用量最大值,CPU_{max}表示从第1个监控周期到第m+1个监控周期中各计算节点的CPU利用率的最大值,MEM_{sug}表示系统内存资源的建议分配值,CPU_{sug}表示系统逻辑CPU核数的建议分配值, β_1 取经验值0.1, β_2 取经验值0.1。

[0094] 如图5所示,本发明实施例提供的资源的动态分配模块工作流程如下:

[0095] 步骤501,若迭代应用对系统资源的需求达到收敛,主节点读取各计算节点上的迭代应用执行情况信息,判断当前轮次的迭代信息是否结束,即步骤303中提到的主节点根据计算节点传回的任务执行信息,调用TaskSchedulerImpl类中的statusUpdate方法,判断当前迭代计算任务是否执行完毕,同时获取应用当前所处的迭代轮数,并等待当前轮次的迭代计算结束;

[0096] 步骤502,若当前轮次的迭代计算结束,调用Spark源码中主节点Master类的

killExecutor方法,结束当前执行进程,根据步骤303中得到的系统内存资源和CPU资源的建议分配值,为Spark集群的任务执行单元重新分配系统资源,格式为<"Memory:MEMsug", "core:CPUusg">。具体的步骤为,首先调用Master类的startExecutorsOnWorkers方法,然后在方法allocateWorkerResourceToExecutors中,Master向Worker发送启动Executor的消息,Worker在接收到LaunchExecutor消息之后,创建ExecutorRunner对象并最终在方法fetchAndRunExecutor中启动Executor进程。通过此步骤,在新的迭代周期中启用重新分配系统资源的任务执行单元,继续执行后续迭代计算。

[0097] 对于用户运行的该迭代应用而言,可能会由于执行单元的一次终止与重新分配系统资源并启动,以及部分缓存的中间数据结果的重计算而带来一些开销,但是对于多轮迭代而言,这些开销并不算大,且随着迭代轮数的增大,这些开销基本可以忽略不计。

[0098] 图6为本发明实施例提供的针对迭代应用的Spark平台资源动态分配系统结构示意图。如图6所示,包括:第一资源分配单元、软件信息确定单元、需求收敛确定单元以及第二资源分配单元。

[0099] 第一资源分配单元,用于根据Spark集群的硬件信息为Spark集群的任务执行单元分配第一资源以用于任务执行单元执行迭代应用,所述Spark集群包括至少一个计算节点,每个计算节点上启动至少一个任务执行单元,所述硬件信息包括每个计算节点的内存总量、可用内存量以及CPU核数,所述第一资源包括的内存量和CPU核数分别小于或等于每个计算节点的可用内存量以及CPU核数。

[0100] 软件信息确定单元,用于确定每个监控周期下每个计算节点在执行所述迭代应用时的软件信息,所述软件信息包括所述迭代应用对所述第一资源的内存使用量和CPU利用率。

[0101] 需求收敛确定单元,用于根据每个计算节点在第m个监控周期下和第m+1个监控周期下执行所述迭代应用时的软件信息确定所述迭代应用对所述第一资源的使用量达到收敛。

[0102] 第二资源分配单元,用于根据第1至第m+1个监控周期下每个计算节点在执行所述迭代应用时的软件信息为Spark集群的任务执行单元分配第二资源,以用于所述任务执行单元重新执行所述迭代应用,所述第二资源包括的内存量和CPU核数分别小于或等于第一资源包括的内存量和CPU核数。

[0103] 图6所示的系统还可包括更多或更少的部件,各部件的功能参见上述图1至图5所示的方法实施例,在此不做赘述。

[0104] 以上,仅为本申请较佳的具体实施方式,但本申请的保护范围并不局限于此,任何熟悉本技术领域的技术人员在本申请揭露的技术范围内,可轻易想到的变化或替换,都应涵盖在本申请的保护范围之内。因此,本申请的保护范围应该以权利要求的保护范围为准。

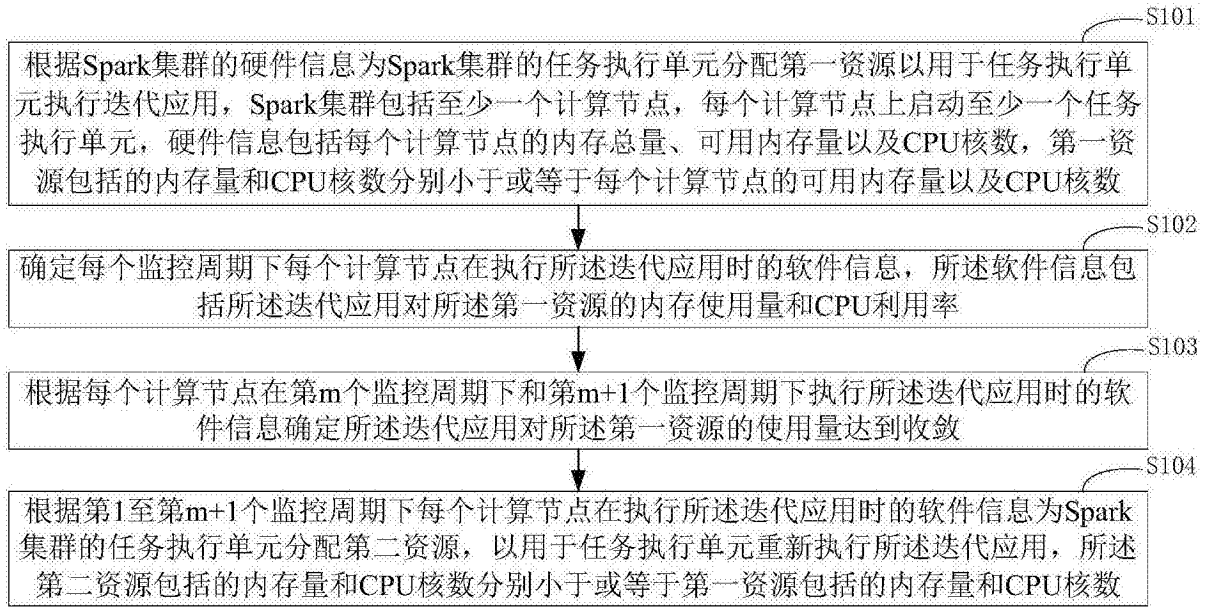


图1

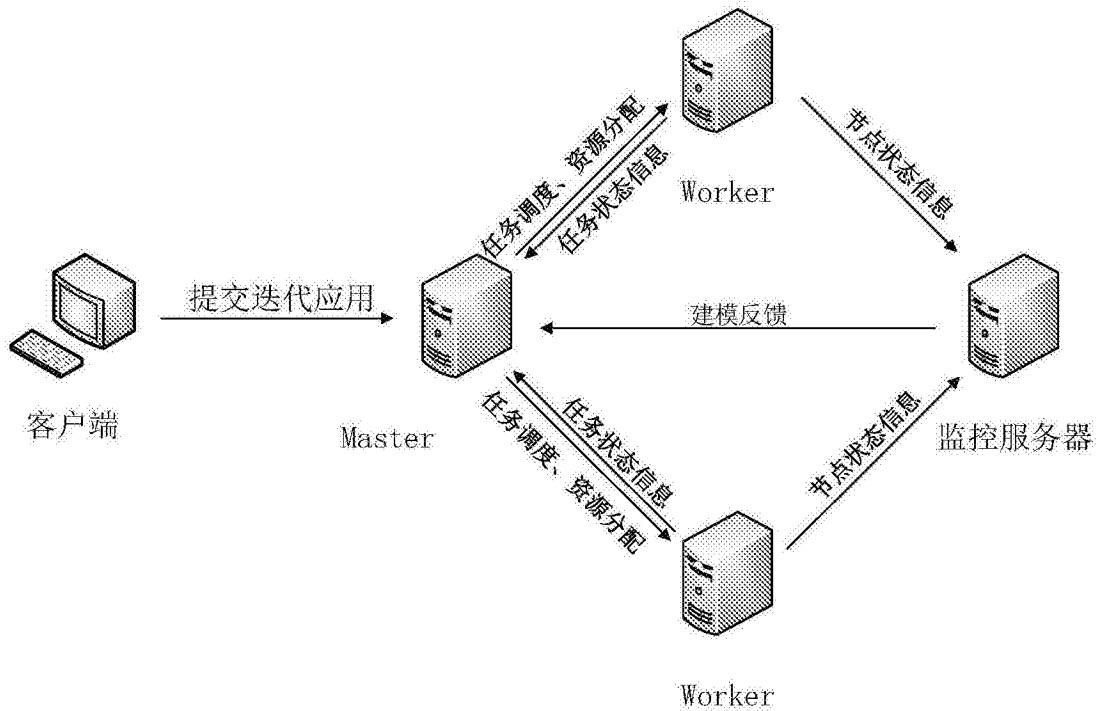


图2

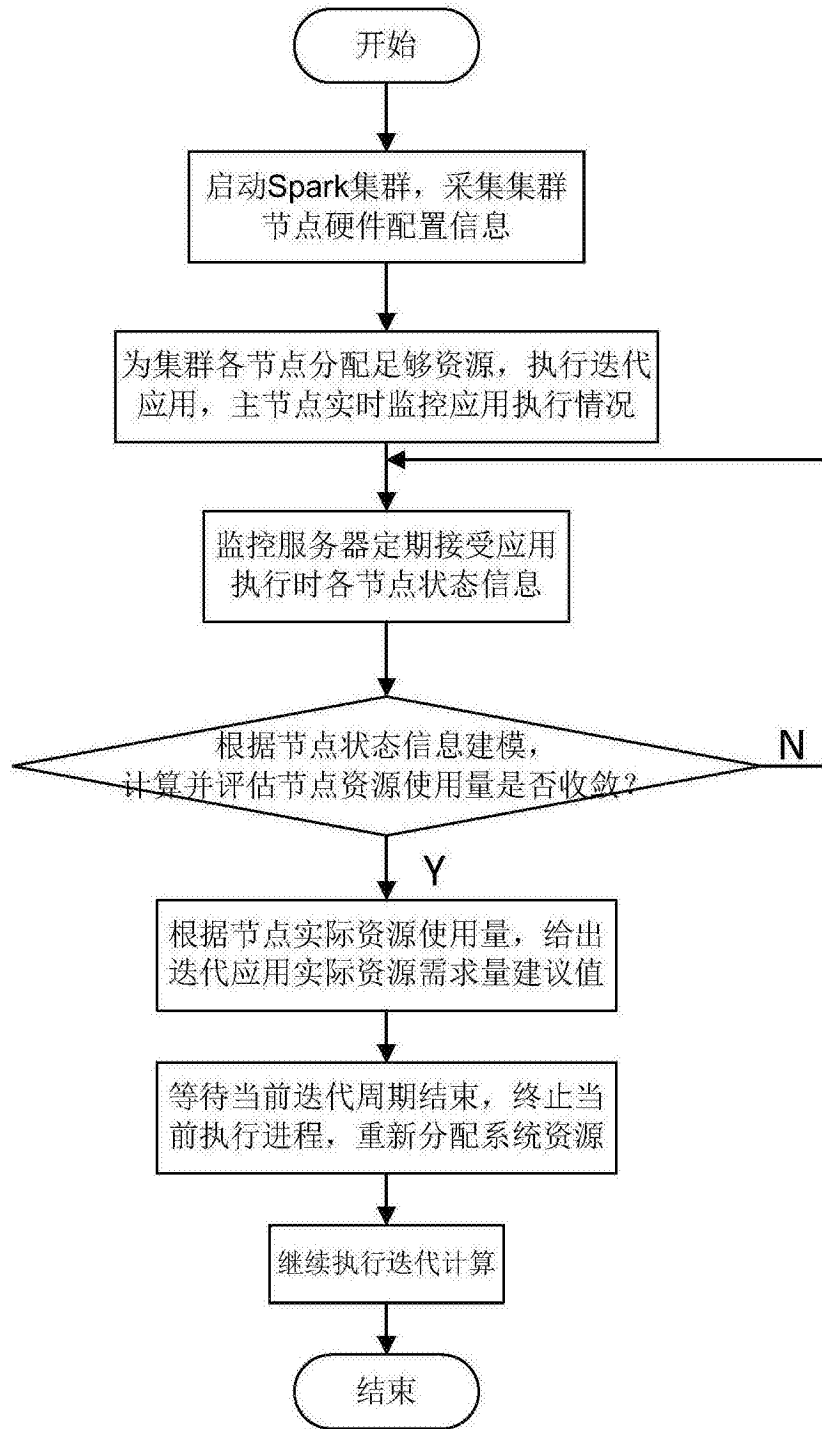


图3

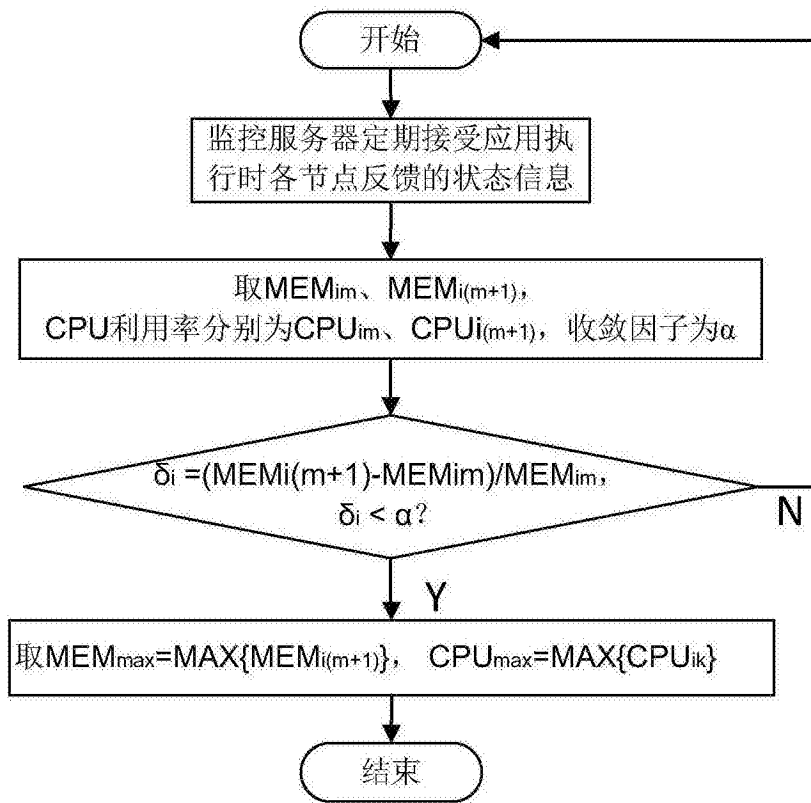


图4

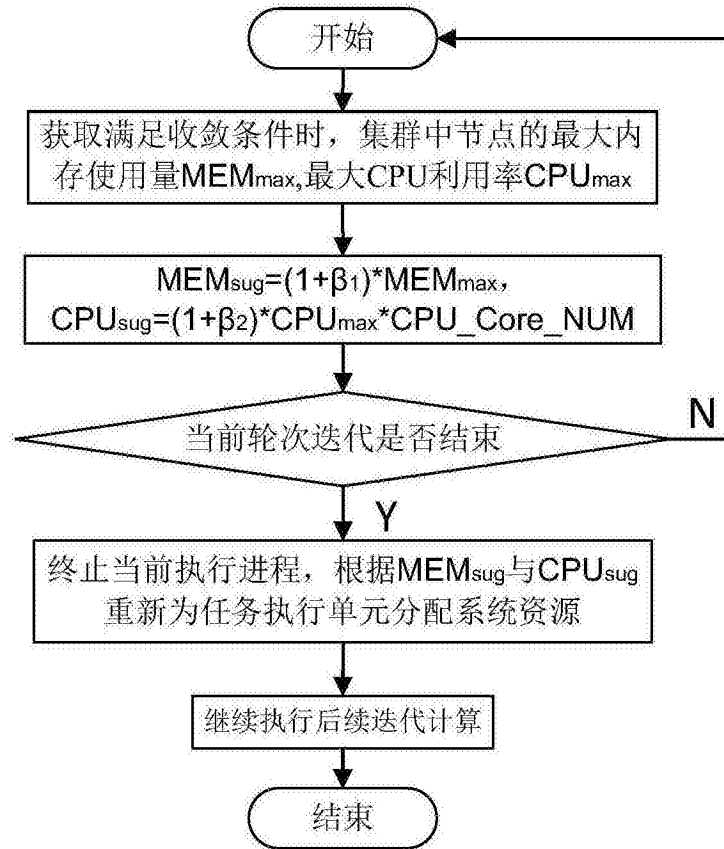


图5

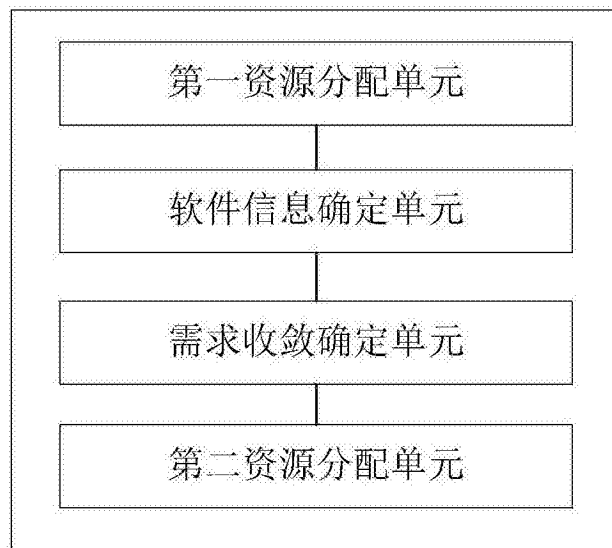


图6