文章编号:1001 - 9081(2006)01 - 0065 - 05

网格计算中基于信任机制的动态任务调度

黄宝边[1],曾文华[2]
(1. 厦门大学 计算机科学系,福建 厦门 361005;
2. 厦门大学 软件学院,福建 厦门 361005)
(baobianh@sohu.com)

摘 要:提出了一种基于信任机制的动态任务调度模型,该模型通过 MDS(Monitoring and Discovery Service)和 NWS(Network Weather Service)组件完成资源信息的收集与反馈,并借鉴现实人类社会中人与人之间的信任关系模型引入信任机制,对数据存储系统采用 DSRL(Dynamic Self-adaptive distributed Replica Location)方法,对出错节点上的任务采用动态迁移方法。在任务调度策略中对 Min-min 算法进行改进,提出了基于信任机制的 Trust-Min-min 算法,提高了网格计算的有效性。最后,采用 SimGrid 工具包对该模型和算法进行了仿真,验证了算法的合理性和高效性。

关键词:网格计算;调度;信任机制;Trust-Min-min 算法
中图分类号:TP393 **文献标识码**:A

Trust mechanism-based dynamic task scheduling in grid computing

HUANG Bao-bian[1], ZENG Wen-hua[2]
(1. *Department of Computer Science, Xiamen University, Xiamen Fujian* 361005, *China;*
2. *School of Software, Xiamen University, Xiamen Fujian* 361005, *China*)

Abstract: A kind of trust mechanism-based dynamic task scheduling model was presented, which collected and feeded back resource information through the use of MDS and NWS. It imported trust mechanism by using interpersonal trust relationship in human society for reference, and adopted the method of DSRL(Dynamic Self-adaptive distributed Replica Location method) for the management of data repository and the method of dynamic transfer for the tasks in the trouble nodes. For the task scheduling strategy, the Min-min algorithm was modified and the Trust-Min-min algorithm was proposed which enhanced the validity of grid computing. At last this algorithm was simulated with the aid of SimGrid toolkit and it was proved reasonable and efficient.

Key words: grid computing; schedule; trust mechanism; Trust-Min-min

0 引言

在过去的 10 年中,科学计算正从主机集中转移到分布方式;近年来,这一趋势更向着网格(Grid)计算延伸。网格是以资源共享为目的,支持对可计算资源的远程和并发访问,用高速网络连接地理上分布的可计算资源所组成的一个具有单一系统映像的高性能计算和信息服务环境[1]。若干正在进行的研究项目,如 Globus 正致力于建立地理分布资源的无缝连接和主观一体化。

网格资源主要包括计算资源(Computation Hosts)、通信资源(Network Bandwidth)和存储资源(Data Centers)。如何改进网格的计算有效性,使用网格资源高效地完成计算任务是网格系统的研究重点之一。

资源管理和任务调度是影响网格计算有效性的关键技术之一,目前有很多关于这方面的研究。从资源组织、定位和管理的角度来看,网格资源管理模型主要分为:集中式模型、分布式模型、层次模型和基于多 Agent 模型[2]。虽然这四种模型在网格计算环境中为资源发现和管理提供了许多策略,但是它们缺乏对资源变化的标准、控制和快速反应等各方面进行处理的有效综合策略。集中式模型对资源变化的标准和控制有较强的处理能力,但对计算网格的可测量性有较大的局限性;分布式模型能够较好的支持可测量性,但过度集中于局部特性,缺乏对全局的控制;层次模型是集中式模型和分布式模型的折中情况,它通过“树”结构实现对资源变化发现和管理,但更新资源信息需要层层上报给上层管理结点直到根结点,因此很难满足快速反应的要求;基于多 Agent 模型通过移动 Agent 收集资源的变化信息,具有较强的可测量性和适应性,但在计算网格环境下该模型对 Agent 本身和 Agent 所需的条件有很高的要求。

好的任务调度算法同样可以改进网格计算有效性。作为计算网格的关键技术之一,国内外许多学者对网格任务调度算法做了大量的研究工作。其中较为经典的有:GA 算法、A* 算法、Min-min 算法、Max-min 算法等[3];较新的有:2-Phase 算法[4]、Co-RSPB 算法、Co-RSBF 算法、Co-RSPBP 算法[5]等。虽然目前已有较多的任务调度算法,但难以同时满足较高调度效率、计算网格环境动态性、计算有效性和任务 QoS 需求等要求。

基于上述分析,本文提出了一种基于信任机制的动态任务调度模型,并对 Min-min 调度算法进行改进,提出了基于信任机制的 Trust-Min-min 算法。

收稿日期:2005 - 07 - 25;**修订日期**:2005 - 10 - 05

作者简介:黄宝边(1981 -),男,福建晋江人,硕士研究生,主要研究方向:网格计算; 曾文华(1964 -)男,江苏兴化人,教授,博士,主要研究方向:神经网络、进化计算、智能控制.

1 基于信任机制的动态任务调度模型

首先要明确网格的使用模式,即用户通过向网格系统提交计算任务来共享网格资源,然后网格调度程序根据用户提交的 QoS 要求,按照某种策略把这些任务调度到合适的计算资源节点,并在网格系统中定位任务所需数据,计算资源节点在获得任务所需数据后开始执行。

1.1 调度模型

在网格调度的层次上,集群(广义上包括单台计算机)往往视为一个计算资源节点,因为它们是基于 SSI(Single System Image)构造的自治系统,自身有着较为完善的调度手段,所以干预集群内部的调度策略是没有必要的。因此,网格任务调度问题成为如何在各计算资源节点之间分配计算任务,并随时根据计算资源节点的变化情况做出调整。

基于上述分析,本文借鉴互联网的层次结构模型,根据计算资源节点的范围大小,将网格系统分为三个服务层次:网格服务层次、局域服务层次和集群服务层次。调度模型如图 1 所示。

基于该模型,任务的调度过程如下:

(1) 用户将元任务和 QoS 需求提交到网格系统"任务提交中心";

(2) "任务提交中心"将任务保存在"任务队列"中,并通过"QoS 查询系统"根据用户提交的 QoS 需求查询"网格服务资源信息中心",将结果返回给"QoS 控制模块";

(3) "QoS 控制模块"根据"QoS 查询系统" 返回的结果对资源进行预选择,得到满足 QoS 需求条件的资源子集;

(4) "任务调度器"按照调度策略将任务调度到各个"局域任务队列";

(5) 在局域服务层次上使用类似(1) ~(4)的过程将"局域任务队列"中的任务调度到各个集群上;

(6) 在集群服务层次上,资源节点在定位、获得所需数据后开始计算任务;

(7) 如果任务在执行过程中出错(包括超出规定时间、节点出错、退出网格系统等情况),则基于"信任机制"将出错节点上的任务转移到其他计算资源节点,直到任务完成为止。

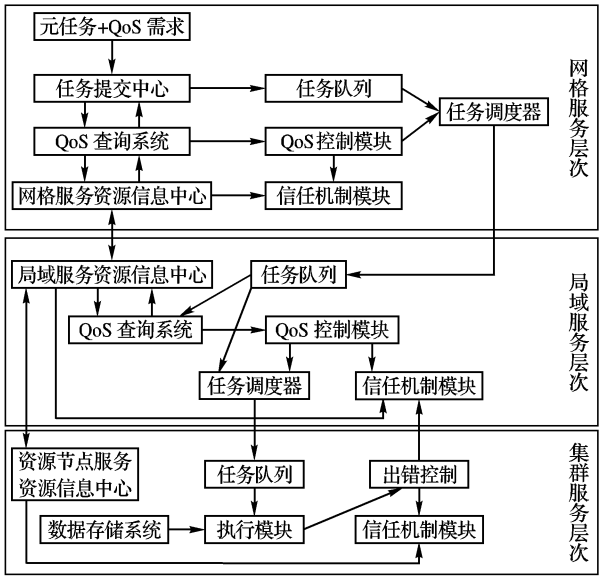

图 1 基于信任机制的动态任务调度模型

1.2 网格环境下的 QoS 层次结构

在网格环境中,根据网格资源类型的不同,QoS 类型也是多种多样。为了方便,本文讨论的 QoS 仅包括通信资源(通信带宽)QoS 和计算资源 QoS,其他类型的 QoS 都假设满足用户的 QoS 需求。

根据网格系统的三个服务层次,网格 QoS 可以划分为四个层次,如图 2 所示。

第一层为用户层,当用户需要使用网格服务进行计算时,由用户提出 QoS 需求;第二层为网格服务 QoS 层,主要描述网格服务的 QoS 需求和管理局域服务的 QoS 信息;第三层为局域服务 QoS 层,主要描述局域服务的 QoS 需求和管理集群服务的 QoS 信息;第四层为集群服务 QoS 层,主要描述和匹配任务在计算资源节点执行的 QoS 信息。

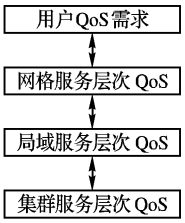

图 2 网格 QoS 层次结构

1.3 服务资源信息中心

服务资源信息中心主要负责监控和发现网格中的所有可用资源,收集并存储资源信息,同时保证同步更新,并且在需要的时候将这些信息提供给相应的请求。其主要包括 MDS(Monitoring and Discovery Service)和 NWS(Network Weather Service)两个组成部分。

MDS[6]是一个网络信息管理系统,用于收集和发布系统状态信息,可以从 MDS 获得的信息主要包括:可用资源集合和每一个资源节点的集群信息(如:CPU 类型、CPU 速度等)。

NWS[7]是一个设计用于跟踪现有资源和网络状况的分布式监控系统,能够提供短期的性能预测,它运行于每一个资源节点上以提供实时监控。

通过对资源信息的管理,反馈到控制模块和调度模块,为调度策略提供依据,从而尽可能的产生最优调度。

1.4 信任机制

借鉴于现实人类社会中人与人之间的信任关系模型,在计算网格环境中也可以建立局域与局域、集群与集群之间的信任机制。每一个局域都保存着该局域与网格环境中其他局域之间的信任关系值,每一个集群也都保存着该集群与其他集群之间的信任关系值,同时,上级层次保存着下级层次之间的信任关系值。

信任机制中的信任关系[2]主要包括两个方面:直接信任关系和间接信任关系。下面对集群之间的信任关系进行定义(局域之间的信任关系定义类似)。

直接信任关系指网格系统中具有直接联系的两个集群之间的信任关系。直接信任关系值主要是由这两个集群的 QoS 参数(如:计算能力、带宽等)比较关系来进行确定,可表示为:

$$Direct\_Trust(i,j) = \left(\sum_{k=1}^{N} \omega_k p_k^{(i,j)}\right)\left(1-\rho^{(i,j)}\right)$$

其中:N 表示需要考虑的 QoS 参数个数;ω_k 表示第 k 个 QoS 参数的权重(如:在局域服务层次,计算能力 QoS 的权重应小于带宽 QoS 的权重;在集群服务层次,计算能力 QoS 的权重应大于带宽 QoS 的权重);$\rho^{(i,j)}$ 表示将集群 i 上的任务转移到集群 j 的失败率,初始值 $\rho^{(i,j)}=0$;$p_k^{(i,j)}$ 表示集群 i 与集群 j 之间第 k 个 QoS 参数的比较关系值(假设第 k 个 QoS 参数为集群的计

算能力,若集群 i 的计算能力大于集群 j 的计算能力,则 $p_k^{(i,j)}=0$;若集群 i 的计算能力等于集群 j 的计算能力,则 $p_k^{(i,j)}=1$;若集群 i 的计算能力小于集群 j 的计算能力,则 $p_k^{(i,j)}=3$)。

间接信任关系指网格系统中通过第三者建立起联系关系的两个集群之间的信任关系。两个集群之间的间接信任关系值可如下表示:

$$Indirect\_Trust(i,j)=\sum_{\substack{k=1\\k\neq i,j}}^{M}\left(\beta_{(i,k)}\times Direct\_Trust(k,j)\right)$$

其中:M 表示网格系统中集群个数;$\beta_{(i,k)}$ 表示集群 i 与 k 之间关系的评价值,$\sum\beta_{(i,k)}=1$,初始值 $\beta_{(i,k)}=\frac{1}{m}$,随着网格环境的不断变化,$\beta_{(i,k)}$ 的值也不断变化。

综上,集群 i,j 之间的信任关系值可表示如下:

$$Trust(i,j)=\alpha\times Direct\_Trust(i,j)+\beta\times Indirect\_Trust(i,j)$$

其中:α、β 分别表示直接信任关系值和间接信任关系值的权重。

1.5　数据存储系统

在本模型中,每个计算资源节点都有一个数据存储系统,用来存储计算任务所需的数据。在网格系统中,不同的数据存储系统可保存同一数据副本,同一资源节点也可保存不同数据资源的副本。本文中,对数据存储系统的管理和数据定位采用一种可扩展、动态自适应的分布副本定位方法——DSRL (Dynamic Self-adaptive distributed Replica Location method)[8]。

DSRL 中提出了宿主结点的概念,支持对同一数据多个副本同时高效定位,且每次查询最多只经过一个中间结点,在查询性能和存储、更新开销上有较好的折中。同时 DSRL 中提出了一种动态均衡映射方法,将副本定位信息均衡分布在多个宿主结点上,并且能够很好的自适应结点的动态加入或退出。

对于计算任务所需数据的获取,如果数据在局域内外都存有副本,则基于带宽 QoS 顺序从局域内的某一数据存储系统获取数据;如果数据只在局域外存有副本,则根据带宽 QoS 顺序,从具有最大带宽 QoS 的数据存储系统获取数据。

1.6　任务的动态迁移[9,10]

在任务执行过程中若有资源节点出错,则基于信任机制将出错资源节点上未完成的任务迁移到其他合适资源节点上执行。任务迁移方法主要有两种:将出错资源节点上所有未完成的任务作为一个整体任务一次性迁移到另一个合适资源节点上执行;将出错资源节点上所有未完成的任务逐个地迁移到其他合适资源节点上执行。

1.7　模型特点

基于信任机制的动态任务调度模型通过基于“资源信息中心”反馈的信息对 QoS 进行控制,对资源进行预选择,从而达到满足任务 QoS 需求的目的。同时,该模型支持对资源节点的出错处理,支持对任务的动态迁移,通过信任机制缩短任务动态迁移的反应时间,这符合网格环境的自治性、动态性和异构性等特点。

2　基于信任机制的动态调度策略

在任务调度算法中,有多种调度相互独立任务的启发式算法存在,Tracy D. Braun 等人已经做了详细的研究[3]。实验结果表明,Min-min、GA 和 A* 能够得到较好的性能,然而,GA 和 A* 的运行速度较慢,不能适应大规模的计算环境。Min-min 算法仍然是目前网格调度算法的研究基础之一。

本文对 Min-min 算法进行改进,提出基于信任机制的 Trust-Min-min 调度策略。该调度策略既考虑到执行任务所需数据的获取时间、任务的 QoS 需求,又兼顾到计算有效性。

2.1　任务模式及参数定义

假定根据 QoS 需求对资源预选择后,符合条件的网格系统由 m 个异构的集群 $C=\{c_1,c_2,\cdots,c_m\}$(组成 r 个局域,$L=\{l_1,l_2,\cdots,l_r\}$)和 n 个数据存储系统 $S=\{s_1,s_2,\cdots,s_n\}$ 所组成。在网格环境下主要考虑的任务模式为元任务[3],即一组相互独立、任务之间没有通信和数据依赖的任务集合,设有 k 个任务组成,$T=\{t_1,t_2,\cdots,t_k\}$,每个任务的执行分为两部分:数据传输和任务计算。其中数据传输代表获得计算所需输入的数据通信,即该任务中需要消耗存储资源和通信资源的部分,输入为一组存储在某个数据源的文件或数据,数据量的大小已知,当任务在不同地理位置的集群上运行时,存取输入数据的花费有可能相差很大;任务计算是该任务中需要消耗计算资源的部分。图 3 描述了文中的任务模式。

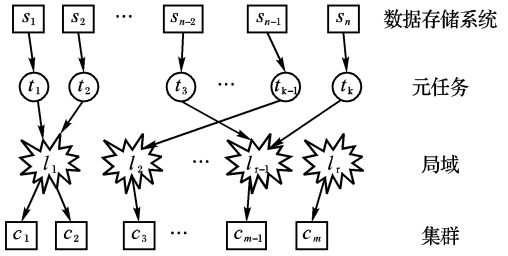

图 3　任务模式

假设任务 T 在每个节点上的运行时间是已知的。对于每个需要调度的任务 t_i,定义以下参数:

(1) $ETC(t_i,c_j)$:任务 t_i 在集群 c_j 上的预期执行时间,若任务 t_i 不能在集群 c_j 上执行,则为系统定义的最大值;

(2) $MCT(t_i,c_j,s_l)$:任务 t_i 在集群 c_j 上运行,存取数据存储系统 sl 上的数据作为输入参数的预期完成时间,则 k 个任务在 m 个不同集群上存取不同数据存储系统上的数据作为输入参数的预测最小完成时间是一个 $k\times m\times n$ 矩阵 Cluster-MCT;

(3) $MCT(t_i,l_k)$:任务 t_i 在局域 l_k 上运行的预期最小完成时间,则 k 个任务在 r 个不同局域上的预测最小完成时间是一个 $k\times r$ 矩阵 LA-MCT;

(4) $START(c_j)$:集群 c_j 的最早可用时间;

(5) $Data\text{-}START(s_l)$:数据存储系统 s_l 的最早可用时间;

(6) $TRAN(t_i,c_j,s_l)$:任务 t_i 在集群 c_j 上运行时在数据存储系统 s_l 上存取数据所需的传输时间。

这些参数数据可以通过调度模型的“资源信息服务中心”获取。

因此,若 $Data\text{-}START(s_l)>START(c_j)$,则 $MCT(t_i,c_j,s_l)=Data\text{-}START(s_l)+TRAN(t_i,c_j,s_l)+ETC(t_i,c_j)$;否则,$MCT(t_i,c_j,s_l)=START(c_j)+TRAN(t_i,c_j,s_l)+ETC(t_i,c_j)$。那么,$MCT(t_i,l_k)=\min(MCT(t_i,c_j,s_l))(\forall c_j\in l_k)$。

2.2　调度策略

2.2.1　元任务的划分

根据任务 QoS 需求,可将元任务分为四类:高计算能力高带宽需求的任务、高计算能力低带宽需求的任务、低计算能力高带宽需求的任务和低计算能力低带宽需求的任务,分别命名为 $QoS(h,h)$、$QoS(h,l)$、$QoS(l,h)$、$QoS(l,l)$。在进行任

务调度时，先调度 $QoS(h,h)$，再依次调度 $QoS(h,l)$、$QoS(l,h)$、$QoS(l,l)$，这样做的目的是为了避免出现高性能资源被 QoS 需求低的任务所占用，而那些 QoS 需求高的任务由于满足不了其执行条件所需要的资源，而停滞在那里，导致整个任务的完成时间延迟。先调度 $QoS(h,l)$ 然后调度 $QoS(l,h)$，可使得 $QoS(h,l)$ 的数据传输部分尽快结束，计算部分尽快开始，以便与 $QoS(l,h)$ 的数据传输部分并行执行，使得元任务中占用绝大部分时间的任务执行部分能够并行执行，从而缩短总的任务完成时间。

2.2.2 Trust-Min-min 算法描述

局域服务层次调度：

当需要调度的元任务 U 非空时，反复执行如下操作直至元任务 U 为空：

（1）对集合中每一个等待分配的任务 t_i，分别计算出把该任务分配到每个局域上的最小完成时间。假设该任务在第 j 个局域上的完成时间为最小，记为 $Min\text{-}LA\text{-}Time(i) = MCT(t_i,l_k)$，可得到一个含有 k 个元素的一维数组 Min-LA-Time；

（2）对于任务 t_i，计算

$$Trust\text{-}Min\text{-}LA\text{-}Time(i) = \frac{Min\text{-}LA\text{-}Time(i)}{Trust(k,j)}$$

假设任务 t_i 是从局域 k 提交到网格系统的，分配到第 j 个局域上的完成时间为最小；

（3）选择 *Trust-Min-LA-Time* 数组中具有最小完成时间的任务 a，并将它插入到相对应的局域任务集合中；

（4）从任务集合 U 中把任务 a 删除，同时更新 LA-MCT 矩阵。

集群服务层次调度：

当局域中需要调度的任务集合非空时，反复执行如下操作直至任务集合为空：

（1）对集合中每个等待分配的任务 t_i，分别计算出把该任务分配到该局域中每个集群上并存取每个数据存储系统上的数据作为输入参数的最小完成时间。假设该任务在第 j 个集群上并存取数据存储系统 sl 上的数据作为输入参数的完成时间为最小，记为 $Min\text{-}Cluster\text{-}Time(i) = MCT(t_i,c_j,s_l)$，可得到一维数组 Min-Cluster-Time；

（2）对于任务 t_i，计算

$$Trust\text{-}Min\text{-}Cluster\text{-}Time(i) = \frac{Min\text{-}Cluster\text{-}Time(i)}{Trust(k,j)}$$

假设任务 t_i 是从集群 k 提交到网格系统的，分配到第 j 个集群上并存取数据存储系统 s_l 上的数据作为输入参数的完成时间为最小；

（3）选择 Trust-Min-Cluster-Time 数组中具有最小完成时间的任务 a，并将它分配给相对应的集群；

（4）从任务集合中把任务 a 删除，同时更新 Cluster-MCT 矩阵。

2.2.3 基于信任机制的调度策略

基于上述分析，基于信任机制的调度策略主要包括下面三个步骤：

（1）定义 QoS 参数的阈值，根据任务 QoS 需求，将元任务按上述方法分成 4 个任务集合：$QoS(h,h)$、$QoS(h,l)$、$QoS(l,h)$、$QoS(l,l)$；

（2）运用 Trust-Min-min 算法，依次对任务集 $QoS(h,h)$、$QoS(h,l)$、$QoS(l,h)$、$QoS(l,l)$ 进行调度。先进行局域服务层次调度，再将已调度到各个局域内的任务进行集群服务层次调度。进行层次性调度的原因是：在局域服务层次，通信带宽 QoS 对任务完成时间的影响比计算能力 QoS 大；在集群服务层次，计算能力 QoS 对任务完成时间的影响比通信带宽 QoS 大。因此，两次调度对信任关系值的计算方法侧重点不一样，有利于区分局域服务层次和集群服务层次对 QoS 的要求不一样。

（3）若在任务执行过程中资源节点 c_k 出错，则对 c_k 上所有未完成任务进行迁移。若采用第一种方法，则首先计算整体任务 t 在其他所有可用资源节点上执行的完成时间 $MCT(t,c_j,s_l)$，然后计算

$$Trust\text{-}Time(j) = \frac{MCT(t,c_j,s_l)}{Trust(k,j)}$$

最后选择最小的 $Trust\text{-}Time(j)$，并将整体任务 t 迁移到对应的资源节点；若采用第二种方法，则按照 Trust-Min-min 算法对 c_k 上的未完成任务集进行重新调度。

2.3 调度策略特点

基于信任机制的调度策略考虑了任务的 QoS 需求并且进行了层次性调度，这有利于更合理地对任务进行调度。而且该策略考虑了局域与局域、资源节点与资源节点之间的信任关系，能够尽可能地避免将任务调度到不稳定的资源节点，并可对出错节点上的任务进行迁移，加强了网格计算的有效性。

3 模拟实验结果

Trust-Min-min 算法的复杂度与 Min-min 算法相同，因此具有相近的执行效率。由于 SimGrid 提供了一系列 API，可以方便地随机生成不同的主机处理能力、网络带宽、通信延迟、数据传输量和计算量等参数，我们将采用 SimGrid 网格任务调度模拟器对这两种算法进行模拟实验，分别对考虑任务 QoS 需求和

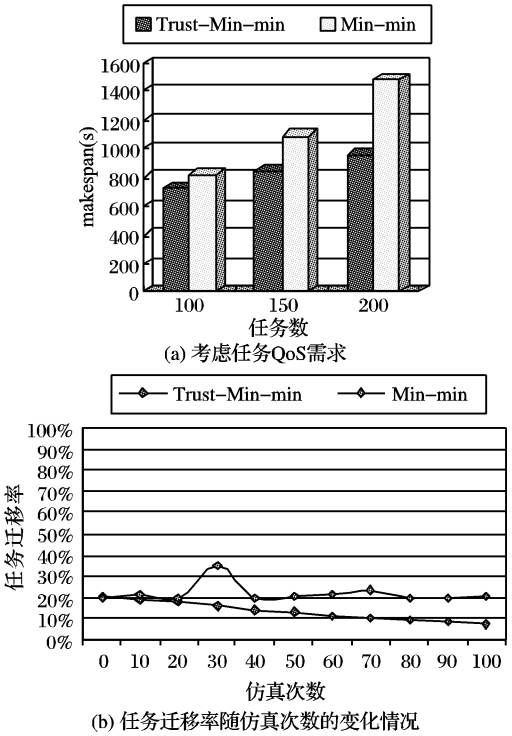

图 4 仿真比较结果

对任务迁移率等不同方面进行测试比较。模拟的网格环境由20个集群组成4个局域,初始信任关系值均设为1,每个集群上有各自的数据存储系统,每个集群的CPU速率(计算能力)在500MHz到800MHz之间随机给定,局域内部集群之间的通信带宽设为100Mbps,局域之间的通信带宽设为1Mbps。对每一次试验都进行了100次仿真,试验依次使用100、150和200个随机产生的任务,每个任务随机从某个集群上提交到网格系统,其QoS需求随机给定,且必须存取一个数据源来获得输入数据,每个任务的运行时间已知。每一次仿真集群的出错个数设定为集群数的20%(即2 020% =4个),对任务迁移采用第二种方法且不考虑迁移开销,最后计算这100次仿真的平均值作为结果。实验结果如图4所示。

图4(a)显示对任务QoS需求进行考虑调度的Trust-Min-min算法比不考虑任务QoS需求的Min-min算法执行效果好,随着任务的增加,Trust-Min-min算法的makespan增加速度比Min-min算法稍慢。图4(b)显示随着仿真次数的增加,信任机制的建立,Trust-Min-min算法的任务迁移率逐渐减小,而Min-min算法的任务迁移率则基本保持不变。

4　结语

网格计算中的任务调度问题是一个非常困难的NP完全问题。本文提出了一种基于信任机制的动态任务调度模型,并使用改进的基于信任机制的Trust-Min-min算法进行任务调度。与Min-min相比,它可以更合理的对任务进行调度,能够尽可能地避免将任务调度到不稳定的局域或资源节点,加强了网格计算的有效性。因此,基于信任机制的调度模型和策略是一种比较有效的任务调度解决方案。

参考文献:

[1] BU GY, XU ZW. A Grid System Theoretical Model[A]. Proc of HPCAsia 2001[C], 2001.

[2] LUO JZ, JI P, WANG XZ, *et al.* Resource management and task scheduling in grid computing[A]. CSCWD'2004[C], 2004.431 - 436.

[3] BRAUN TD, SIEGEL HJ, BECK N. *et al.* A Comparison of Eleven Static Heuristics for Mapping a Class of Independent Tasks onto Heterogeneous Distributed Computing Systems[J]. Journal of Parallel and Distributed Computing, 2001, 61(6): 810 - 837.

[4] CHEN HT, MAHESWARAN M. Distributed dynamic scheduling of composite tasks on grid computing systems[A]. Parallel and Distributed Processing Symposium, Proceedings International, IPDPS 2002[C], 2002.88 - 97.

[5] MIN R, MAHESWARAN M. Scheduling co-reservations with priorities in grid computing systems[A]. Cluster Computing and the Grid 2nd IEEE/ACM International Symposium CCGRID 2002[C], 2002.250 - 251.

[6] MDS document[EB/OL]. http://www.globus.org/mds.

[7] RICH W. Dynamically forecasting network performance using the network weather service[J]. Journal of Cluster Computing, 1998, 1(1): 119 - 132.

[8] 李东升,李春江,肖依,等.数据网格环境下一种动态自适应的副本定位方法[J].计算机研究与发展,2003,40(12):1775 - 1780.

[9] WU M, SUN XH. A general self-adaptive task scheduling system for non-dedicated heterogeneous computing[A]. Cluster Computing, 2003. Proceedings. 2003 IEEE International Conference on 2003[C], 2003.354 - 361.

[10] RITCHIE G, LEVINE J. A Fast, Effective Local Search for Scheduling Independent Jobs in Heterogeneous Computing Environments[A]. Proceedings of the 22nd Workshop of the UK Planning and Scheduling Special Interest Group (PLANSIG 2003)[C], 2003.

(上接第64页)

4　结语

本文将网格环境中任务的执行分成数据传输和计算两个部分,对任务完成时间进行详细的分析,给出减少任务完成时间的方法,并根据任务的需求指定任务优先级,然后通过优先级对任务进行调度的算法,该算法降低了任务的总完成时间,能够提供较好的负载平衡。今后的工作将进一步研究调度事件的发生频率对网格性能的影响和根据任务的执行价格决定任务调度的策略。

参考文献:

[1] FOSTER I, KESSELMAN C. The Grid: Blueprint for a Future Computing Infrastructure[M]. Morgan Kaufmann Publishers, USA, 1999.

[2] BUYYA R, MURSHED M. A Deadline and Budget Constrained Cost-Time Optimisation Algorithm for Scheduling Task Farming Applications on Global Grids[R]. Technical Report, Monash University, March 2002.

[3] VISWANATHAN S, VEERAVALLI B. Design and Analysis of a Dynamic Scheduling Strategy with Resource Estimation for Large-Scale Grid Systems[A]. Proceedings of the Fifth IEEE/ACM International Workshop on Grid Computing (GRID'04)[C], 2004.

[4] 都志辉,陈渝,刘鹏.网格计算[M].北京:清华大学出版社,2002.

[5] CASANOVA H, LEGRAND A, ZAGORODNOV D, *et al.* Heuristics for scheduling parameter sweep applications in Grid environments[A]. Proc. of the 9th Heterogeneous Computing Workshop (HCW'2000)[C]. Cancun, Mexico, 2000. 349 - 363.

[6] MORENO R. Job scheduling and Resource Management Techniques in Dynamic Grid Environments[A]. 1st European Across Grids Conference[C], 2003.

[7] BUYYA R, ABRAMSON D, GIDDY J. Nimrod/G: An Architecture for a Resource Management and Scheduling System in a Global Computational Grid[A]. Proceedings of the HPC ASIA'2000[C], 2000.

[8] YARKHAN A, DONGARRA JJ. Experiments with Scheduling Using Simulated Annealing in a Grid Environment[A]. Workshop on Grid Computing[C], Baltimore, 2002.

[9] CASANOVA H, OBERTELLI G, BERMAN F, *et al.* The AppLeS Parameter Sweep Template: User-Level Middleware for the Grid[A]. the super computing conference[C], 2000.

[10] WU MY, SHU W, ZHANG H. Segmented min-min: a static mapping algorithm for meta-tasks on heterogeneous computing systems[A]. Heterogeneous Computing Workshop, (HCW 2000) Proceedings. 9th[C], 2000.375 - 385.

[11] HE XS, SUN XH, VON LASZEWSKI G. QoS Guided Min - Min Heuristic for Grid Task Scheduling[J]. The Journal of Computer Science and Technology, 2003, 18(4): 442 - 451.

[12] MUSUNOORI SB, ELIASSEN F. Richard StaehliSimula Research Laboratory. QoS-Aware Component Architecture Support for Grid[A]. Proceedings of the 13th IEEE International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises (WET ICE'04)[C], 2004.277 - 282.

[13] ROSENBERG AL, YURKEWYCH M. Guidelines for Scheduling Some Common Computation-Dags for Internet-Based Computing[J]. IEEE Trans. Computers, 2005, 54(4): 428 - 438.