计算网格环境下一个统一的资源映射策略*

丁 箐[1,2], 陈国良[1,2], 顾 钧[3]

[1](中国科学技术大学 计算机科学与技术系,安徽 合肥 230027);
[2](国家高性能计算中心,安徽 合肥 230027);
[3](香港科技大学 计算机系,香港)
E-mail: charles@mail.hf.ah.cn; glchen@ustc.edu.cn
http://www.nhpcc.ustc.edu.cn

摘要: 由于资源具有广域分布、异构、动态等特性,计算网格环境下资源的管理和调度是一个非常复杂且具有挑战性的问题.提出了计算网格环境下一组相互独立的计算任务(meta-task)的资源映射策略.该策略采用重复映射方法,以更好地适应网格计算环境下的动态性和自治性.算法考虑到任务的输入数据位置对映射效果的影响;通过定义效益函数,该策略在追求较小的任务完成时间的同时兼顾任务的服务质量(QoS)需求.模拟实验结果显示,该映射策略更符合计算网格的复杂环境,能够更好地满足不同用户的实际需要.

关 键 词: 计算网格;资源映射;效益启发式
中图法分类号: TP393 文献标识码: A

计算网格(computational grid)是一个充满吸引力的高性能计算平台.由于网格环境下计算资源广域分布、异构、动态、有多个管理域、存在不同的存取花费模式,资源的管理和调度十分复杂,目前没有一种管理模式能够处理所有的网格应用需求.大量的网格项目试图提供一个合适的资源管理方法[1~4],一般可分为集中式控制和本地应用控制两种类型.前者可以获得系统所有的资源信息以优化资源的使用,但扩放性(scalable)很差且存在单点失败;而后者则不能有效地了解全局资源信息,无法实现具有“网格意识”(grid-aware)的应用.由于多机环境下的资源映射是众所周知的 NP 问题,对于 DAG 任务映射的研究现在一般都集中在单管理域的异构集群模式,并且必须采用各种启发式对映射问题进行简化[5,6].在网格环境下为了支持多个资源协同工作,资源预置(advanced resource reservation)是最常见的方法[7].目前在计算网格环境下主要考虑的是一组相互独立的任务(meta-task)的映射,所谓相互独立,即任务之间没有通信和数据依赖[8~11].

1 计算网格环境下的任务模式以及算法中使用的一些符号

假设网格系统由 m 个异构的集群(这里所说的集群也广义地包括单台的计算机)$C=\{c_1,c_2,c_3,c_m\}$、f 个文件服务器或数据存储系统 $S=\{s_1,s_2,s_3,s_f\}$所组成.现有 k 个任务,$T=\{t_1,t_2,t_3,t_k\}$,每个任务的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.一个文件或数据源可以为多个任务所共享,每个任务内部可能包含独立的或相互依赖的子任务.图 1 描述了文中的任务模式.

假设任务 T 在每台机器上的运行时间是已知的.对于每个需要映射的任务 t_i,我们定义以下参数:

* 收稿日期: 2000-11-01; 修改日期: 2001-05-09
基金项目: 国家重点基础研究发展规划 973 资助项目(G1998030403)
作者简介: 丁箐(1972－),男,安徽肥东人,博士生,主要研究领域为网格计算技术,分布式计算系统,计算机网络;陈国良(1938－),男,安徽颍上人,教授,博士生导师,主要研究领域为并行分布计算,并行体系结构,并行算法,计算机网络;顾钧(1956－),男,江苏大丰人,教授,博士生导师,主要研究领域为 NP 难解问题高效算法及其应用,计算机理论,快速算法,高性能应用软件.

(1) $ETC(t_i,c_{jk})$:任务 t_i 在集群 c_j 的第 k 个机器上的预期执行时间,若任务 t_i 不能在集群 c_j 上的第 k 个机器上执行,则为系统定义的最大值.

(2) $COMP(t_i,c_{jk},s_l)$:任务 t_i 在集群 c_j 的第 k 个机器上运行,存取数据存储系统 s_l 上的数据作为输入参数的预期完成时间.

(3) $START(c_{jk})$:集群 c_j 的第 k 个机器的最早可用时间.

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

(5) $TRAN(t_i,c_j,s_l)$:任务在集群 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.

(6) D_i:用户定义的完成任务 t_i 的最终期限.

(7) β_i 和任务 t_i 相联系的效益函数,它定义任务在一定时间内完成后,用户所得到的收益.用户可根据自己的需求定义各种效益函数,最简单的情况下,效益函数是任务完成时间的反比函数.图 2 给出了几种常见的效益函数图形,本文主要采用图 2(c)的形式,式 1 是其数学表达式,其中 a,b,c 为用户定义的常数.

$$\beta=\begin{cases}a, & t<bD\\ a-c(t-bD), & t\ge bD\end{cases}. \tag{1}$$

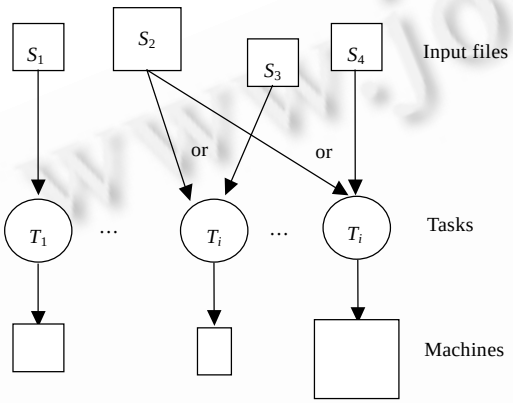

Fig.1 Task model

图 1 任务模式

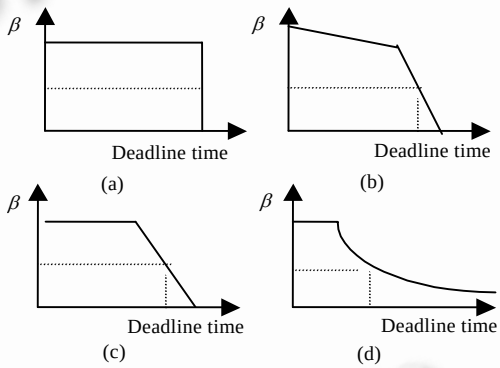

Fig.2 Benefit functions

图 2 效益函数

2 映射算法

映射启发式可分为两类[10]:在线模式(on-line mode)和批模式(batch mode).在线模式情况下,任务一旦到达,就立刻执行映射;而在批模式下,任务被收集为一个组,即所谓的 Meta-Task,当映射事件发生时才对整个任务组进行映射.批模式由于累积了较多的任务和资源信息,因此可以得到更有效的映射结果而被较多采用,但相对来说实时性较差.为了更好地反映网格环境的动态性和自治性,我们采用重复映射策略.当每次映射事件发生时,Superscheduler 通过网格信息查询服务获得当前网格所有组成集群的资源状态信息,Meta-Task 中既包含新到达的任务,也包含那些在上一次的映射事件中已经被映射但还没有开始执行的任务,算法使用更新的资源状态信息对所有这些任务进行映射,因此一个任务有可能被多次重复映射(见算法 1).这种重复映射策略虽然加重了系统负载,但算法可利用更新的资源状态信息得到更好的映射结果,与复杂、动态的网格计算环境是相适应的.

算法 **1**. 重复映射策略.

```
T=t0  //scheduler start time
      Δt        //inter-schedule time
      while (true)
            t=t+Δt
            while (current time<t)
                  collect arriving tasks into Meta-Task
            endwhile
```

collect cluster state in grid from Grid Information Sever
collect tasks which are mapped but not executed into Meta-Task
get ETC(t_i,c_{jk}),START(t_i,c_{jk}),TRAN(t_i,c_j,s_l)
schedule-Meta (M,M')
//mapping tasks using appropriate heuristics.
Endwhile

有多种调度(包括映射)相互独立任务的启发式存在[9~13],实验表明,其中 Min-min、GA、A*和 Sufferage 能够得到较好的性能.然而 GA 和 A*运行速度比较慢,不能适应大规模的计算环境,Min-min 的负载均衡性能不高,Sufferage 具有较好的综合性能.

本文提出的启发式称为效益函数启发式,该启发式考虑到任务的输入数据位置对映射效果的影响,即任务/主机的亲和性(affinity),计算任务完成时间时将任务的输入数据传输时间考虑在内,更加符合网格环境下的任务实际执行情况.更重要的是,传统映射算法的目标一般都是得到最短的任务完成时间(makespan).然而在网格环境下,用户行为和系统资源状况都相当复杂,简单地追求最短完成时间并不合适,我们应对用户情况加以区分.该启发式在追求较短的任务完成时间的同时,兼顾到任务的服务质量(QoS)需求.对于一个计算任务,我们最关注的是其完成时间,目前我们主要将完成时间作为服务质量参数.算法通过定义任务的效益函数来评估任务的 QoS 需求,即用户在提交任务时根据其需求同时提交一个具体的效益函数,这个效益函数反映出任务经过一定的时间完成后,用户可以得到的效益.大多数的效益函数图形如图 2(c)所示,即任务只要在一定的时间期限内完成,用户的收益没有变化,当任务的预期完成时间接近定义的最终期限时,其效益函数值将有很大的下降.算法将 Meta-Task 中的任务按其最小完成时间以递增排序,当任务刚开始映射时,所有的任务预期完成时间距离用户定义的最终期限一般都还有较多的空余,也就是说所有任务的 Benefit Sufferage 都等于 0,算法实际上等同于 Min-min 启发式.随着时间的增加,有些任务的效益函数值会下降,算法将优先映射这些任务.如果用户有紧急的任务需要马上进行映射,则可将其效益函数设计成图 2(b)的形式,这样当映射事件发生时,此任务的 Benefit Sufferage 不等于 0,因此任务将被首先映射.算法 2 是详细的启发式算法.

算法 **2**. 启发式算法.

function schedule-Meta (Meta-Task M,Meta-Task M') {
for all tasks t_i in Meta-Task M
for all machines c_{jk}
for all available data repositories s_l of a task
calculate COMP(t_i,c_{jk},s_l)
do until (all tasks in M are scheduled)
for each task t_i in M
calculate minimum complete time
if the minimum complete time of task t_i larger then D_i
delete t_i from M, insert it into M'
// The task can not be scheduled in this event. User
must modify D_i, then scheduled in next event
endfor
sort the tasks in Meta-Task M in ascending order by their minimum complete time
mark all data repositories as available
for each task t_i in M
find machine c_{jk} and data repository s_l that gives the minimum complete time
calculate related benefit value in terms of minimum COMP(t_i,c_{jk},s_l) and
the second minimum COMP(t_i,c_{jk},s_l)
sufferage value = the best benefit value - the second best benefit value

```
            if data repository s_l is available
                assign t_i to data repository s_l, delete t_i from M
                mark s_l unavailable
            else
            if sufferage value of task t_k already assigned to s_l is less than
                the sufferage value of task t_i
            unassign t_k, add it back to M
                    assign t_i to data repository s_l, delete t_i from M
        endfor
        update Data-START( ) matrix
        update START( ) matrix
enddo}
```

3 模拟实验结果

效益函数启发式的算法复杂度和 Sufferage 相同,因此具有相近的执行效率.为了评估映射效果,我们设计了一个模拟程序,模拟的网格环境由 4 个集群和 5 个数据源组成,4 个集群分别包含 5,6,7,8 个主机.模拟应用由 500 个任务组成,每个任务必须存取一个数据源来获得输入数据,任务的运行时间已知.首先我们测试输入数据位置对映射结果的影响,图 3 显示随着任务输入数据量的增大,不考虑输入数据位置的 Min-min 启发式的 Makespan 急剧地增加,与之相比,Benefit 启发式的 Makespan 则增加缓慢.图 4 显示 Benefit 启发式在映射资源时考虑到任务的 QoS 需求,优先映射接近最终期限的任务和紧急的任务,是以 Makespan 的增加为代价的,但是和 Sufferage 启发式相比增加的幅度不是很大,基本上是常数级的增长.同时图 5 显示 Benefit 启发式对用户情况加以区分,优先映射接近最终期限的任务,因任务预期完成时间已超过最终期限而被抛弃的任务数目得到减少.

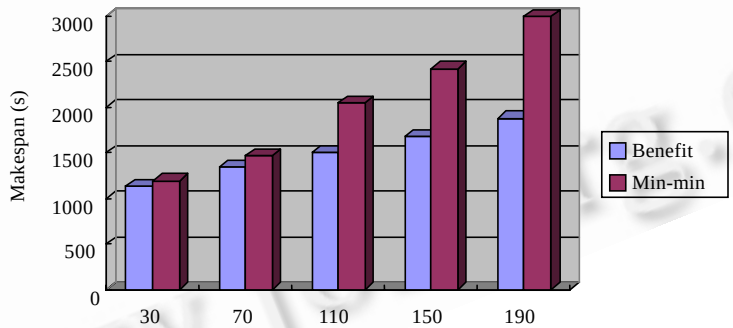

Fig.3 Makespan and input data (whether considering input data)

图 3 任务输入数据量和任务完成时间(是否考虑输入数据分布)

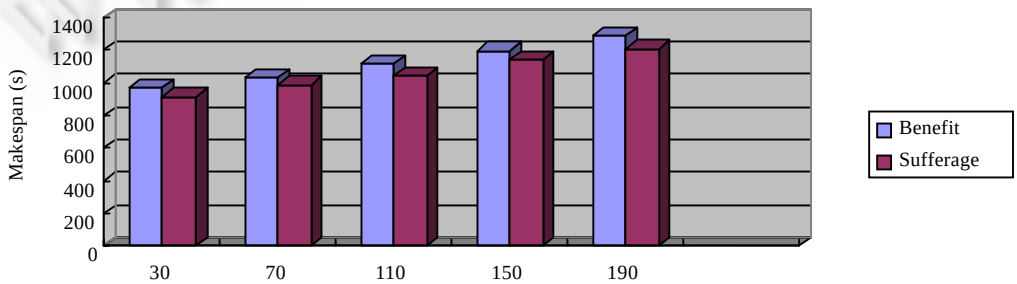

Fig 4 Makespan and input data (whether considering QoS of tasks)

图 4 任务输入数据量和任务完成时间(是否考虑任务的 QoS)

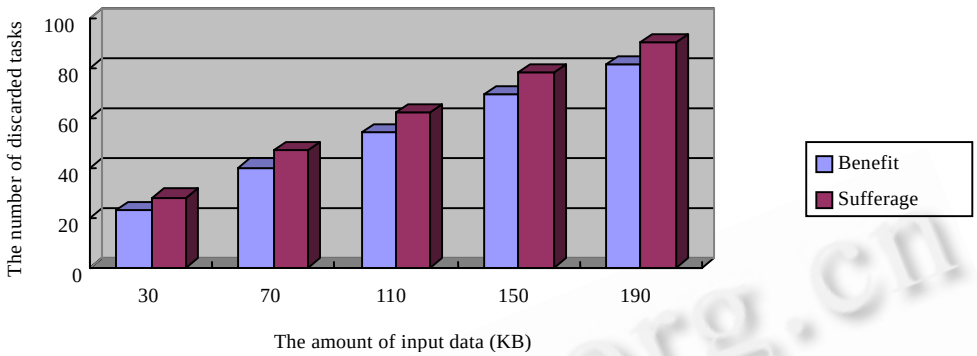

Fig.5 The abandoned task number and input data (whether considering QoS of tasks)

图 5 任务输入数据量和抛弃的任务数(是否考虑任务的 QoS)

4 结 论

本文针对计算网格环境下的复杂结构,提出了有效的映射一组相互独立任务的策略.该策略采用重复映射以更好地适应网格计算环境下的动态性和自治性.算法考虑到任务的输入数据位置对映射效果的影响.通过定义效益函数,该策略在追求较小的任务完成时间的同时兼顾任务的服务质量(QoS)需求,比较符合网格环境下任务的实际情况.今后要做的工作是对 QoS 进一步细化,开发包含多种 QoS 限制的算法策略以及在算法中更深入地引入经济性概念[14].由于网格环境具有动态性和复杂的用户行为,经济方法是资源配置的有效方法之一,我们正试图用计算市场来实现网格环境下的资源配置问题.

References:

[1] Foster, I., Kesselman, C. The Grid, Blueprint for a New Computing Infrastructure. San Francisco: Morgan Kaufmann Publishers Inc., 1998. 279~309.

[2] Czajkowski, K., Foster, I. A resource management architecture for metacomputing systems. In: Feitelson, D.G., Rudolph, L., eds. Proceedings of the 4th Workshop on Job Scheduling Strategies for Parallel Processing. LNCS 1459, Orlando: Springer-Verlag, 1998. 62~82.

[3] Sekiguchi, S., Sato, M. Ninf: network based information library for globally high performance computing. In: Proceedings of the Parallel Object-Oriented Methods and Applications (POOMA). 1996. 39~48. http://www.acl.lanl.gov/Pooma96/.

[4] Freund, R., Gherrity, M. Scheduling resources in multi-user, heterogeneous computing environments with Smarnet. In: Proceedings of the 7th Heterogeneous Computing Workshop (HCW'98), IEEE Computer Society Press, 1998. 184~199. http://dlib.computer.org/ conferen/ hcw/8365/pdf/83650003.pdf.

[5] Iverson, M., Ozguner, F. Dynamic, competitive scheduling of multiple DAGs in a distributed heterogeneous environment. In: Proceedings of the 7th Heterogeneous Computing Workshop (HCW'98). IEEE Computer Society Press, 1998. 70~78. http://dlib.computer. org/ conferen/hcw/8365/pdf/83650070.pdf.

[6] Wang, L., Siegel, H.J., Roychowdhury, V.P., *et al*. Task matching and scheduling in heterogeneous computing environments using a genetic algorithm based approach. Journal of Parallel and Distributed Computing, 1997,47(1):8~22.

[7] Foster, I., Roy, A., Winkler, L. A quality of service architecture that combines resource reservation and application adaptation. In: Proceedings of the 8th International Workshop on Quality of Service (IWQOS 2000). 2000. 181~188. http://www.globus.org/documentation/ incoming/iwqos_adapt1.pdf.

[8] Armstrong, R., Hensgen, D., Kidd, T. The relative performance of various mapping algorithm is independent of sizable variance in run-time predictions. In: Proceedings of the 7th Heterogeneous Computing Workshop (HCW'98). IEEE Computer Society Press, 1998. 79~87. http://dlib.computer.org/conferen/hcw/8365/pdf/83650079.pdf.

[9] Braun, T.D., Siegel, H.J., Beck, N., *et al*. A comparison study of static mapping heuristics for a class of meta-tasks on heterogeneous computing systems. In: Proceedings of the 8th IEEE Heterogeneous Computing Workshop (HCW'99). IEEE Computer Society Press, 1999. 15~29. http://dlib.computer.org/conferen/hcw/0107/pdf/01070015.pdf.

[10] Maheswaran, M., Ali, S., Siegel, H.J., *et al*. Dynamic matching and scheduling of a class of independent tasks onto heterogeneous computing systems. In: Proceedings of the 8th IEEE Heterogeneous Computing Workshop (HCW'99). IEEE Computer Society Press, 1999. 30~44. http://dlib.computer.org/conferen/hcw/0107/pdf/01070030.pdf.

[11] Maheswaran, M., Siegel, H.J. A dynamic matching and scheduling algorithm for heterogeneous computing systems. In: Proceedings of the 7th IEEE Heterogeneous Computing Workshop (HCW'98). IEEE Computer Society Press, 1998. 57~69. http://dlib.computer.org/conferen/hcw/8365/pdf/83650057.pdf.

[12] Maheswaran, M. Quality of service driven resource management algorithms for network computing. In: Proceedings of the 1999 International Conference on Parallel and Distributed Processing Technologies and Applications (PDPTA'99). Computer Science Research, Education, and Applications Press, 1999. 1090~1096. http://www.cs.umanitoba.ca/~anrl/PUBS/qos_based_rms.ps.gz.

[13] Alhusaini, A.H., Prasanna, V.K. A unified resource scheduling framework for heterogeneous computing environments. In: Proceedings of the 8th Heterogeneous Computing Workshop (HCW'99). IEEE Computer Society Press, 1999. 156 ~165. http://dlib.computer.org/conferen/hcw/0107/pdf/01070156.pdf.

[14] Buyya, R., Abramson, D., Giddy, J. A case for economy grid architecture for service oriented grid computing. In: Proceedings of the 10th IEEE International Heterogeneous Computing Workshop (HCW 2001) 2001. 776~790. http://www.csse.monash.edu.au/~rajkumar/ papers/ecogrid.pdf.

A Unified Resource Mapping Strategy in Computational Grid Environments*

DING Qing[1,2], CHEN Guo-liang[1,2], GU Jun[3]

[1](*Department of Computer Science and Technology, University of Science and Technology of China, Hefei* 230027, *China*);

[2](*National High Performance Computing Center at Hefei, University of Science and Technology of China, Hefei* 230027, *China*);

[3](*Department of Computer Science, Hong Kong University of Science and Technology, China*)

E-mail: charles@mail.hf.ah.cn; glchen@ustc.edu.cn

http://www.nhpcc.ustc.edu.cn

Abstract: The management of resources and scheduling computations in a grid environment is a complex undertaking, mainly due to resource's geographic distribution, heterogeneity, distributed ownership with different policies and priorities, varying loads, reliability, and availability conditions. A unified resource mapping strategy in computational grid environments is presented, which considers the input data repositories and QoS of tasks to mapping a set of independent tasks (meta-task) to resources. The repetitive mapping algorithm is more suitable for the dynamic adaptability and domain autonomy in the grid. The benefit function heuristic adopted in the algorithm can assure the QoS of tasks more effectively.

Key words: computational grid; resource mapping; benefit heuristic

* Received November 1, 2000; accepted May 9, 2001
Supported by the National Grand Fundamental Research 973 Program of China under Grant No.G1998030403