

移动 Agent 在服务网格下的路由算法研究

高 瞻, 罗四维, 丁 丁

(北京交通大学计算机与信息技术学院, 北京 100044)

摘要: 以移动Agent作为任务载体, 代表用户应用程序在复杂多变的服务网格环境下智能地访问各种网格服务。该文引入了“最大效益旅行商问题”来描述移动Agent的行为方式, 给出了移动Agent的路由算法, 使得移动Agent在最大化用户效益的同时兼顾不同用户的服务质量需求。

关键词: 服务网格; 移动 Agent; 服务质量

Research on Mobile Agent Routing Algorithm in Service Grid

GAO Zhan, LUO Si-wei, DING Ding

(School of Computer and Information Technology, Beijing Jiaotong University, Beijing 100044)

【Abstract】 This paper proposes that mobile agents act as carriers of tasks on behalf of users' applications, accessing grid services intelligently in the dynamic complex service grid environment. The "most benefit traveling salesman problem (MBTSP)" is introduced to describe how agents act and a routing algorithm is also presented. Mobile agents are able to maximize the users' benefits as well as assure different QoS requirements.

【Key words】 service grid; mobile agent; QoS

“网格”(grid)是一门新兴的信息技术, 被称为下一代的Internet^[1]。它构建在互联网之上, 可以将高速互联网、高性能计算机、大型数据库、传感器、远程设备等融为一体, 实现大范围内的资源共享和系统集成。目前, 网格进入了迅速发展的阶段, 工业界与学术界联盟, 正致力于使网格计算在更广泛的领域得到推广和应用^[2]。2002年2月, GGF把影响最大的网格项目——Globus中的技术与当前影响广泛的商用的Web服务技术标准相融合, 提出了以开放网格服务结构(open grid services architecture, OGSA)^[3]作为标准的网格体系结构。OGSA以服务为中心, 把一切都抽象为服务, 服务既包括计算机设备、应用、数据, 也包括仪器、设备等。

在计算网格环境下, 资源具有地理位置上的广泛分布性、异构性、动态性并且可能属于不同的管理域, 许多网格计算项目, 比如Legion^[4], Condor^[5], AppLes^[6]等所采用的传统资源映射模式都是以系统为中心的, 即任务调度的目标是充分利用系统中的资源, 最大化系统的吞吐率。这在硬件资源比较稀缺, 且面临的主要是计算问题的情况下是非常必要的, 但在广泛应用于商业领域的服务网格下, 资源映射策略应该把用户的效益作为首要考虑的因素, 即在保证满足用户对服务需求的前提下最大化用户的效益。

针对服务网格的特点, 本文使用移动 Agent 作为访问网格服务的载体, 以最大化用户的效益为目的在服务网格环境下智能地移动。把移动 Agent 应用于网格环境可以避免在传统网格系统中使用的进程迁移机制缺乏灵活性、需要被控制的缺点, 减轻网格服务和应用程序开发者以及网格用户的负担, 增强应用程序对复杂多变的网格环境的适应性。

1 最大效益旅行商问题

经典旅行商问题(TSP)的定义是对于给定的 n 个城市, 城市之间有不同的距离, 旅行商从某一个城市出发不重复地遍

历其余城市后回到出发的城市, 要求找出一条旅行路线, 使该旅行路线的路程最短。在服务网格环境中移动 Agent 的迁移路径问题与 TSP 问题有一些差异, 具有自己的特点。

(1)可能有不同的网格节点提供相同的服务, 每个节点的服务能力以及服务价格可能不同。

(2)移动 Agent 不需要遍历所有的服务节点, 只要完成自己的任务就可以了。

(3)每个节点提供服务的能力和状态会发生变化。

(4)网格环境具有动态性, 网络负载会动态变化, 提供服务的节点也会动态变化, 随时会有提供服务的节点加入或退出。

(5)移动 Agent 的路由目标是总的费用最小而不是总的路径最短。

下面提出“最大效益旅行商问题(MBTSP)”来描述移动 Agent 的路由特点: 给定 n 个城市, 每个城市有自己对特定种类商品一定数量的需求, 并要对该种商品征收一定的税务。商人携带不同种类的商品, 从某一城市出发不重复地访问其它城市, 在不同城市之间旅行要花费不同的差旅费。要求找出一条旅行路径, 使得商人能够全部卖出自己的商品并能获得最大的利润。

该问题的数学形式描述为: 设有 N 个城市 $C = \{c_1, c_2, \dots, c_N\}$, 使用任意两个城市间的距离来表示商人在其间旅行需要花费的时间 $d(c_i, c_j) = t_{ij}, 1 \leq i, j \leq N$ 。商人有自己的效益函数 f , f 是关于商人旅行时间和其效益的函数, 携带有不同种类的商品

基金项目: 国家“863”计划基金资助项目(2004AA1Z2530)

作者简介: 高 瞻(1982-), 男, 博士研究生, 主研方向: 网格计算, 计算机并行处理; 罗四维, 教授、博士; 丁 丁, 讲师、博士研究生

收稿日期: 2007-01-07 **E-mail:** bjtugaozhan@gmail.com

$S = \{s_1, s_2, \dots, s_K\}$, 商品的价格为 $P = \{p_1, p_2, \dots, p_K\}$ 。城市 c_i 有其对某一种商品 s_j 的需求, 需求量 $r_i(s_j) = r_{ij}$, $1 \leq i \leq N, 1 \leq j \leq K$ 。现在要求找出一个城市的排列顺序 $\{c_1, c_2, \dots, c_m\}$, $1 \leq m \leq N$, 使得旅行商的货物量

$$G = \sum_{i=1}^m r_{ij}$$

且旅行商的效益

$$B = f\left(\sum_{i=1}^{m-1} t_{i,i+1}\right) - \sum_{i=1}^m r_{ij} p_j$$

最大。

2 服务网格下的移动 Agent 路由算法

在服务网格下, 以最大化用户效益为目标的移动 Agent 路由问题就是一个“最大效益旅行商问题”。为了表示不同用户对网格服务的 QoS 需求(本文考虑的是任务完成时间), 给每个移动 Agent 赋予一个用户效益函数, 它表示用户效益和任务完成时间之间的关系。常见的用户效益函数如图 1 所示。

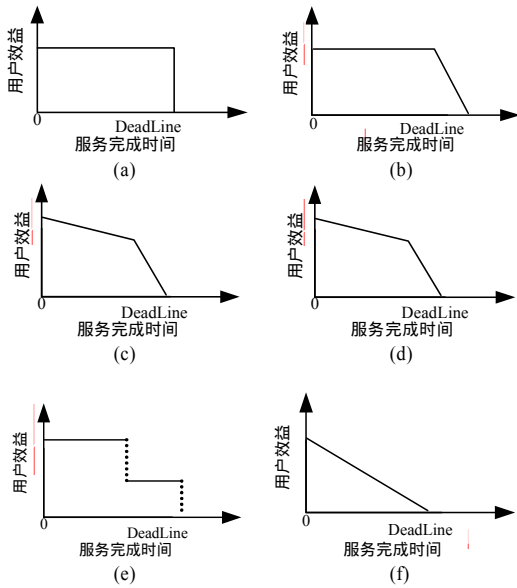


图 1 几种常见的效益函数

由于网格节点的地理位置广泛分布, 节点之间的网络带宽和负载各不相同, 因此 Agent 在不同的节点之间移动所需要的时间也有很大差异。为了最大化用户的效益, Agent 应该找到一条任务完成时间尽量小的路由路线。除了任务的完成时间之外, Agent 还应考虑路由路线上的节点服务价格。路径最短的路线可能经过服务价格较高的节点, 它可能并不是能够最大化用户效益的 Agent 路由选择。

在算法中使用移动 Agent 对路由路径进行评价的方法来指导 Agent 的路由。起始时 Agent 从某一个节点出发, 以一定的概率选择下一个要到达的节点。当 Agent 完成自己的任务后根据自己的评价函数对自己所经过的不同节点间的路径进行评价, 至此 Agent 完成第 1 次的迭代。在第 2 次迭代时, Agent 利用上一次对不同节点间路径的评价值进行选路, 任务完成后使用评价函数对各个路径进行评价。当移动 Agent 选择的路径不再发生变化或达到预先定义的最大迭代次数时, 算法认为 Agent 找到了能够最大化用户效益的路由路线。

下面给出对算法中所需使用的参数的定义:

- (1) r_{ij} : 网格节点 n_i 和 n_j 之间的路径。
- (2) $ActualSet$: 移动 Agent 在路由时所经过的节点集合。
- (3) R : 移动 Agent 在路由时所经过的节点间路径的集合。

(4) t_{ij} : 移动 Agent 由网格节点 n_i 迁移到节点 n_j 所需要的时间。

(5) μ_j : 节点 n_j 所能提供的服务量。

(6) e_j : 移动 Agent 使用节点 n_j 所提供的服务需要花费的服务运行时间。

(7) w_j : 移动 Agent 在节点 n_j 上的服务等待时间。

(8) p_j : 节点 n_j 的服务价格。

(9) C : 移动 Agent 完成自己的任务后所经过的节点服务价格之和, 即

$$C = \sum_{j \in ActualSet} p_j$$

(10) T : 移动 Agent 完成自己的任务共需花费的时间, 包括在节点间的移动时间 t_{ij} , 在节点上的运行时间 e_j 及等待时间 w_{j0} 。

(11) $f(t)$: 用户的效益损失函数。其值可由时间为 t 时移动 Agent 所代表用户的效益函数值减去用户提交任务时的效益函数值得到。

(12) $g_{ij}(n)$: 移动 Agent 对节点 n_i 和 n_j 之间路径的评价函数, 自变量 n 代表算法的迭代次数。则移动 Agent 的路由算法如式 (1) 所示。

$$p_{ij}(n) = \begin{cases} g_{ij}(n) \left[\frac{u_j}{f(t_{ij} + e_j + w_j) + p_j} \right]^\alpha & m \in S \\ \frac{\sum_{m \in S} g_{im}(n) \left[\frac{u_m}{f(t_{im} + e_m + w_m) + p_m} \right]^\alpha}{otherwise} & \\ 0 & \end{cases} \quad (1)$$

其中, $p_{ij}(n)$ 表示在第 n 次迭代中, 移动 Agent 由网格节点 n_i 移动节点 n_j 的概率。在每次迭代中 Agent 开始找路时 S 由所有提供移动 Agent 所需服务的网格节点组成, 当 Agent 访问过其中一个节点后就该节点从 S 中删除, 当 Agent 满足了某种服务需求后就该所有提供该种服务的节点从 S 中删除。

在式 (1) 中

$$\frac{u_j}{f(t_{ij} + e_j + w_j) + p_j}$$

的意义是节点 n_j 的服务能力与 Agent 由节点 n_i 移动到 n_j 并访问完 n_j 所提供的服务共需的花费之间的比值, 把它称为节点 n_j 的性价比, 记为 ppr_{ij} (performance-price ratio)。可见参数 α 控制着路径的评价值与节点的性价比对移动 Agent 选路的影响。移动 Agent 对路径的评价函数如式 (2) 所示。

$$g_{ij}(n) = \begin{cases} \rho g_{ij}(n-1) + \frac{Q}{f(T) + C}, r_{ij} \in R \\ \rho g_{ij}(n-1), otherwise \end{cases} \quad (2)$$

其中, ρ 和 Q 为常量; ρ 代表惩罚系数; $g_{ij}(0)$ 的初始值为一个随机的小正数。

3 模拟实验结果及分析

使用 TSPLIB 库提供的 eil51 地图来模拟网格环境对移动 Agent 的路由算法进行试验。首先进行迭代 500 次得到一条最优的 Agent 路由线路 $path_1$, 然后任意选取线路中的一个节点 n , 增加它的服务等待时间, 用来模拟该节点的负载过大或网络延迟较大的情况等, 使得任务完成时间超出用户定义的任务截止时间。然后, 在此基础上继续进行 800 次迭代, 结果表明 Agent 会找到另外一条路由线路 $path_2$, 尽管 $path_2$ 的花费比 $path_1$ 要大, 但它避开了节点 n 并使得任务能够在用户定义的截止时间之前被完成, 从而保证了网格服务的 QoS。

另外测试了参数 α 对移动 Agent 路由算法的影响。使用

如图 1(b)所示类型的用户效益函数,对 α 的值分别为 1,3,5,7 这 4 种情况进行了试验。在每种情况下,进行 10 次试验,每次试验的迭代次数为 800 次,试验结果如表 1 所示。

表 1 在 eil51 图中路由算法的试验结果

	$\alpha=1$	$\alpha=3$	$\alpha=5$	$\alpha=7$
最小收敛迭代次数	283	237	172	194
平均收敛迭代次数	291.3	249.6	186.1	205.7
最小花费	141.2	131.6	93.5	145.8
平均最小花费	145.4	138.7	98.2	151.3

在图 2 中每条曲线描述的是相应条件下 10 次试验的平均结果。

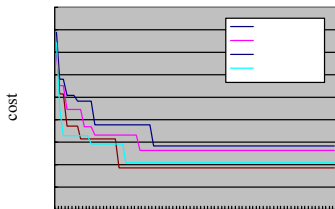


图 2 在 eil51 图中的 Agent 路由结果

从图 2 中可以看出,在 α 的值由 1 增加到 3 以及由 3 增加到 5 时,算法的收敛速度加快,所得到的结果也变得更好;但当 α 的值由 5 增加到 7 时,不但算法的收敛速度变慢,而且所得到的结果也在变坏。因此,要想使移动 Agent 能得到较好的路由结果必须给 α 选择合适的值。另外,可以根据网格的实际情况适当减少移动 Agent 的路由迭代次数,尽管这样可能会使移动 Agent 找到的并不是最优的路径,但是可以

(上接第 36 页)

分别利用 VOD 和 LOF 方法分析股票的异常波动,设定前 15 个异常因子最大的点为异常点,用圆圈标出,实验结果如图 3 所示。

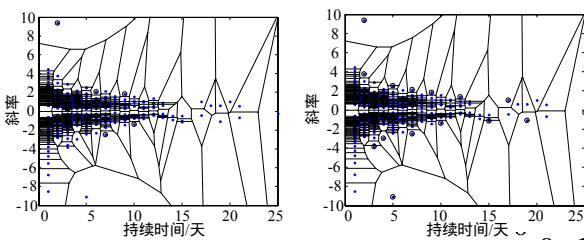


图 3 VOD和LOF方法的比较

从图 3 中不难看出,LOF 方法检测出的 10 个异常点在密度较高的区域中,显然是错误的,而 VOD 方法给出了正确的结果。

5 结束语

针对基于密度的异常检测方法存在的问题,本文利用 Voronoi 图的基本原理,定义了一种新的 V 邻域异常因子 VNOF,提出了一种基于 Voronoi 图的异常检测算法 VOD,解决了目前基于密度方法存在的缺陷。

对于高维空间中的异常检测,受“维灾”问题的影响,随着数据维数的升高,算法的效率急剧下降,目前主要采用近似算法。文献[6]提出了一种高维空间中近似 Voronoi 图的构建方法,对于 d 空间中 n 个点的点集,建立近似 Voronoi 图的时间为

在一定程度上避免移动 Agent 路由超时,减轻网络的负担。

4 结束语

本文研究了以移动 Agent 为任务载体,最大化用户效益为目标的服务资源映射问题。提出了兼顾用户效益和 QoS 需求的 Agent 路由算法,并对算法进行了模拟实验。下一步的工作包括进一步细化用户的 QoS 需求,开发出针对多种 QoS 需求的算法,增强算法在避免 Agent 路由超时方面的鲁棒性,并把它应用到实际的网格环境中。

参考文献

- 1 Foster I. Internet Computing and the Emerging Grid[DB/OL]. (2000-12-07). <http://www.nature.com/nature/webmatters/grid/grid.html>.
- 2 都志辉,陈渝,刘鹏.以服务中心的网格体系结构 OGSA [DB/OL]. (2002-09). <http://www.chinagrid.net/grid/talksanddocs.htm>.
- 3 徐志伟,冯百明,李伟.网格计算技术[M].北京:电子工业出版社,2004:47-48.
- 4 Chapin S, Karpovich J, Grimshaw A. The Legion Resource Management System[C]//Proc. of the 5th Workshop on Job Scheduling Strategies for Parallel Processing. 1999: 162-178.
- 5 Litzkow M, Livny M, Mutka M W. Condor—A Hunter of Idle Workstation[C]//Proc. of International Conference on Distributed Computing Systems. 1988-06.
- 6 Casanova H, Obertelli G, Berman F, et al. The AppLe Sparger Sweep Template: User-level Middleware for the Grid[C]//Proc. of Super Computing. 2000: 75-76.

$$O\left(n \frac{\log n}{\epsilon^d} \log^2 \frac{n}{\epsilon}\right)$$

Alpha = 1
Alpha = 3
Alpha = 5
Alpha = 7

最近邻近的查询时间为 $O(\log(h/\epsilon))$,其中, ϵ 为阈值,同其他高维空间中基于密度的异常检测相比,保证了检测结果的准确性,同时算法的时间复杂性也比较低。

参考文献

- 1 Breunig M M, Kriegel H P, Ng R, et al. LOF: Identifying Density-based Local Outliers[C]//Proceedings of the ACM SIGMOD International Conference on Management of Data, Dallas, Texas, USA. 2000: 93-104.
- 2 Jin W, Tung K H A, Han Jiawei, et al. Ranking Outliers Using Symmetric Neighborhood Relationship[C]//Proceedings of the 10th Pacific-Asia Conference on Advances in Knowledge Discovery and Data Mining, Singapore. 2006: 577-593.
- 3 蒋盛益,李庆华,王卉.一种增强的局部异常挖掘方法[J].计算机研究与发展,2005,42(2): 210-216.
- 4 Pei Yaling, Zaiane O R, Gao Yong. An Efficient Reference-based Approach to Outlier Detection in Large Datasets[C]//Proceedings of the 6th International Conference on Data Mining, Washington, DC, USA. 2006: 478-487.
- 5 Preparata F P, Shamos M I. Computational Geometry—An Introduction[M]. [S. l.]: Springer-Verlay, 1985.
- 6 Har-Peled S. A Replacement for Voronoi Diagrams of Near Linear Size[C]//Proceedings of the 42nd IEEE Symposium on Foundations of Computer Science, Las Vegas, Nevada, USA. 2001: 94-103.

