第 34 卷 第 4 期 计 算 机 工 程 2008 年 2 月
Vol.34 No.4 Computer Engineering February 2008
·网络与通信· 文章编号：1000—3428(2008)04—0146—03 文献标识码：A 中图分类号：TP393

一种适合 P2P MMOG 的移动代理迁移策略

石祥滨[1,2]，王 越[2]，李 强[2]，刘 芳[2]，苏欣娜[2]
（1. 沈阳航空工业学院计算机学院，沈阳 110034；2. 辽宁大学信息科学与技术学院，沈阳 110036）

摘 要：目前的移动代理迁移策略不能满足 P2P MMOG 的实时性需求。该文提出了适合 P2P MMOG 的代理迁移策略，包括迁移时机计算和一种改进的蚁群算法。该策略根据 P2P MMOG 中的网络流量、代理服务需求度和目的主机负载状况动态决定迁移目的地。实验证明，该方案满足游戏实时性要求，降低了移动代理迁移时间和系统延迟，解决了 P2P MMOG 中玩家节点的负载均衡问题。

关键词：P2P MMOG 游戏；移动代理；迁移策略；蚁群算法

Suitable Migration Strategy of Mobile Agent for P2P MMOG

SHI Xiang-bin[1,2], WANG Yue[2], LI Qiang[2], LIU Fang[2], SU Xin-na[2]
(1. Department of Computer Science and Engineering, Shenyang Institute of Aeronautical Engineering, Shenyang 110034;
2. School of Information Science and Technology, Liaoning University, Shenyang 110036)

【Abstract】 As the present strategy of the mobile agent migration can not meet the real-time demand in P2P Massively Multiplayer Online Games (MMOG), this paper proposes a sort of agent migration strategy which contains migration opportunity calculation and a sort of improved ACO so that it fits the P2PMMOG better. The strategy chooses the migration opportunity dynamically based on three factors: the network traffic, the requirement degree of agent service and the load of host. Simulation results show that this strategy not only meets the real-time demand, but also reduces the migration time of the mobile agent and the system delay, so that it has solved the load balance problem of player nodes in P2P MMOG.

【Key words】 P2P MMOG; mobile agent; migration strategy; ant colony optimization

1 概述

基于P2P的MMOG通常采用混合式P2P结构，该结构能够高效地利用区域内所有玩家节点的计算资源，但是存在一些玩家节点经常由于任务过重导致超载，从而影响P2P MMOG的可伸缩性。移动代理凭借其能够根据运行环境的变化携带任务动态迁移的能力成功地解决了分布式系统中负载均衡问题[1]。因此，采用移动代理技术解决MMOG中负载均衡问题是十分有意义的。该方案中首要解决的问题是移动代理迁移问题，此外MMOG的实时性需求给移动代理迁移策略提出了更高的要求。

目前，一些研究者针对移动代理迁移策略进行了大量的研究。最佳解图策略[2]的路由表在代理移动初期就已经确定，代理在移动过程中严格按照路由表迁移。该策略适合短途迁移、静态路由、负载不发生变化的环境。但是该策略要求待访问的主机已知，而在多数情况下，设计者事先无法决定移动代理访问哪些主机；其次，该策略路径规划使用的负载信息在移动代理出发时采集，而环境的动态变化使这些信息过时。因此，最佳解图策略不能适应MMOG中玩家的动态加入、退出以及计算资源环境的不断变化。一步迁移策略[2]在移动代理出发之前并没有为之搜索出一个最佳解图，而是在迁移过程中根据环境变化不断调整其移动路径。该策略只能保证移动的每一步是“最佳的”。一步迁移策略适用于旅行计划不复杂、负载变化频率高、动态路由环境。但是当旅行图比较复杂时，该策略求得的解图是局部最优的。由于MMOG中移动代理迁移路线很复杂，因此一步迁移策略不适合MMOG。以上研究工作虽然可以解决迁移策略方面的一些问题，但是仍然存在缺陷，不适合MMOG中移动代理的迁移。此外，蚁群算法[3]也成功地解决了移动代理迁移问题。蚁群算法具有较强的稳定性、分散性和鲁棒性，算法中的信息素不断加强，采用了正反馈原理，并利用随机选择策略，加快进化过程。该策略可能导致移动代理在两个节点之间振荡迁移从而使得移动代理无法前进。

本文通过对以上的迁移策略优缺点的分析，综合 P2P MMOG 中的网络流量、代理服务需求度和目的主机负载状况几个因素改进了蚁群算法，同时提出了代理迁移时机计算算法。

2 一种 P2P-Agent 框架的网络游戏中间件体系结构

本文设计的代理迁移策略是基于目前正在实现的一种 P2P-Agent 框架的网络游戏中间件系统，该中间件体系结构如图 1 所示。

图 1 一种 P2P-Agent 框架的网络游戏中间件体系结构

基金项目：辽宁省自然科学基金资助项目“基于 P2P 的 MMOG 关键技术”(20052007)；辽宁省教育厅攻关计划项目“网络游戏引擎及其相关技术”(2004D116)

作者简介：石祥滨(1963－)，男，教授、博士，主研方向：分布式系统，网络游戏，数据库技术；王 越、李 强、刘 芳、苏欣娜，硕士研究生

收稿日期：2007-02-21 **E-mail：**sxb@lnu.edu.cn

在游戏系统设计中，游戏任务被抽象成同步代理服务[4]、兴趣管理代理服务[5]、逻辑判定代理服务和世界管理代理服务，这些代理是动态代理。此外，系统中的玩家代理服务是静态代理，它被部署在每个玩家机器上，负责玩家与游戏世界的沟通。游戏初始时，各个代理服务对象的位置信息被区域内所有玩家共享。游戏进行时，当随着任务不断增加使得其上计算环境不再满足代理服务对象需求时，代理服务对象将迁移到区域内其他合适的玩家节点上以保持负载均衡。在代理服务迁移的过程中需要耗费网络带宽资源和迁移途中经历的玩家节点的计算资源，这将增加系统的延迟，同时还将影响玩家游戏的实时性。因此，根据P2P MMOG的特点量身定做代理服务迁移策略是本文拟解决的关键问题。该策略中包含代理迁移时机计算和一种改进的蚁群算法。

3 迁移时机计算

MMOG 中各个代理服务不断感知外界计算资源状态(CPU、网络带宽等)的变化，并迁移到其他合适的玩家节点上。代理迁移时机计算完成了代理是否迁移的决定，相关内容如下。

代理服务对象在玩家节点上运行时所需要的资源集合记为 $R=<r_1,r_2,\cdots,r_m>$ (m 为资源数目，这里的资源指 CPU 资源，网络带宽等)，若 R 中资源均可用，则该玩家节点适合代理服务对象运行，r_i^{ω} 表示资源 r_i 的可用度阈值。p_{ri} 是资源 r_i 的耗用量。*Time* 表示单位时间段，代理在 *Time* 时间段内感知 n 次。当 *Time* 足够小并且 n 足够大时，每次感知到的结果是之前 *Time*/n 时间段内资源使用率的平均值。η_i 表示资源 r_i 在时间段 *Time* 内的期望，并且 $\eta_i=\sum_{i=0}^{n}\left(p_{ri}\times\frac{1}{n}\right)$。如果 $\eta_i<r_i^{\omega}$，则资源 r_i 可以提供正常的服务。$P^{\text{move}}=\sum_{i=0}^{s}\left(r_i^{\omega}\times\frac{1}{m}\right)$ 是代理服务对象是否迁移的基准，$P_0^{\text{move}}=\sum_{i=0}^{s}\left(\eta_i\times\frac{1}{m}\right)$ 表示 R 中资源使用率在 *Time* 时间段内的期望。

确定迁移时机算法

输入：$i\leftarrow 0$；$\eta_i\leftarrow 0$；$p_{ri}\leftarrow 0$；$move\leftarrow 0$；$P_0^{\text{move}}\leftarrow 0$；$m$；$P^{\text{move}}$；$r_i^{\omega}$；$n$

输出：*move*

步骤：

Step1 获得资源 r_i 的 p_{ri}。

Step2 如果 $p_{ri}<r_i^{\omega}$，则转到 Step3；否则 $move\leftarrow 1$，转到 Step7。

Step3 计算 $\eta_i=\sum_{i=0}^{n}\left(p_{ri}\times\frac{1}{n}\right)$。

Step4 重复执行 Step1~Step3 的步骤，直到 m 个资源的 η_i 全部计算出。

Step5 计算 $P_0^{\text{move}}=\sum_{i=0}^{s}\left(\eta_i\times\frac{1}{m}\right)$。

Step6 如果 $P_0^{\text{move}}\ge P^{\text{move}}$，则 $move\leftarrow 1$，否则 $move\leftarrow 0$。

Step7 结束。

如果算法输出的结果是 $move=1$，则表明代理服务对象需要迁移。

4 基于蚁群算法的代理迁移算法

蚁群算法的思想是蚂蚁群并行、异步地从某个节点转移到其他相邻节点，在转移过程中利用节点的局部信息素表，应用统计决策策略选择下一个移动的节点，从而逐步找到可行解。一旦找到某个可行解，蚂蚁按照给定的限制条件对解进行估计，同时根据所形成解的质量释放一定量的信息素，通过信息素的间接通信作用引导后来的蚂蚁向最优解方向搜索。本文将改进蚁群算法以保证游戏实时性和防止代理服务在两个玩家节点之间振荡迁移，并进一步根据玩家的网络流量衡量出最佳解。算法相关内容描述如下：

该算法的初始解是区域中所有的玩家节点。问题的最终解是区域中综合计算资源使用率、代理服务需求程度和网络带宽占用 3 个因素得出的最佳玩家节点集合。算法描述如下：

$A=\{A_i\},(i=0,1,2,\cdots,s\text{-}1)$ 表示代理服务对象的集合，其中，s 表示代理服务的数量。

Q_j^m 表示玩家节点 j 的 m 个邻居玩家。

$\lambda_k(A_i)$ 表示玩家 k 访问 A_i 的次数。

$\tau_{k,A_i}(t)$ 表示 t 时刻，玩家 k 上关于 A_i 的信息素浓度，这里用经过玩家 i 访问 A_q 的次数密度表示，由下式求出：

$$\tau_{k,A_i}(t)=\begin{cases}\dfrac{\lambda_k(A_i)}{\sum_{p=0}^{m-1}\lambda_p(A_i)},k=1,2,\cdots,m & t\neq 0\\ C & t=0\end{cases}$$

$\mu_{k,A_i}(t)$ 是与 $\tau_{k,A_i}(t)$ 相关联的基于问题的启发式信息值，这里 $\mu_{k,A_i}(t)$ 取为系统资源使用率。

$\zeta_{j,k}^{A_i}(t)$ 表示 t 时刻 A_i 由玩家节点 j 转移到玩家节点 k 的概率，由下式求出：

$$\zeta_{j,k}^{A_i}(t)=\begin{cases}\dfrac{\tau_{k,A_i}^{\alpha}(t)\mu_{k,A_i}^{\beta}(t)}{\sum_{s=0}^{m-1}\tau_{s,A_i}^{\alpha}(t)\mu_{s,A_i}^{\beta}(t)},k\in Q_j^m\\ 0,\ \text{others}\end{cases}$$

其中，α,β 分别是 $\tau_{k,A_i}(t)$ 和 $\mu_{k,A_i}(t)$ 在概率 $\zeta_{j,k}^{A_i}(t)$ 中权重的参数。如果 $\alpha=0$，那么最轻载的玩家节点容易被选中。如果 $\beta=0$，那么只有信息素起作用，于是最需要 A_i 的玩家节点容易被选中。

同时，本文为了避免代理服务在两个玩家节点之间发生振荡迁移引入了一个参数 $\hbar_{\text{k}}$，由下式求出：

$$\hbar_k=\begin{cases}\phi & \phi\in[0,1],\text{if Player } k \text{ is the last visited}\\ 1 & \text{if Player } k \text{ is not the last visited}\end{cases}$$

进而 $\zeta_{j,k}^{A_i}(t)$ 改进为 $\zeta_{j,k}^{A_i}(t)=\begin{cases}\dfrac{\tau_{k,A_i}^{\alpha}(t)\mu_{k,A_i}^{\beta}(t)\hbar_k}{\sum_{s=0}^{m-1}\left(\tau_{s,A_i}^{\alpha}(t)\mu_{s,A_i}^{\beta}(t)\hbar_k\right)} & k\in Q_j^m\\ 0 & \text{others}\end{cases}$。

为了防止 $\tau_{k,A_i}(t)$ 的无限制累加和发现更好解，本算法采用了信息素挥发系数 χ，随着时间的推移，信息素调整规则采用如下公式 $\tau_{k,A_i}(t+1)=\chi\tau_{k,A_i}(t)+\Delta\tau_{k,A_i}(t),(\chi\in[0,1])$。这里 $\Delta\tau_{k,A_i}(t)=\begin{cases}\omega\mu_{k,A_i}(t) & \text{if } k \text{ needs } A_i\\ 0 & \text{otherwise}\end{cases},\omega\in[0,1]$，其中，$\Delta\tau_{k,A_i}(t)$ 表示 $[t,t+1]$ 时间段内信息素增量。

本文除了考虑综合计算资源使用率和代理服务需求程度两个参数之外还考虑了目的玩家节点处网络带宽资源占用因素。采用文献[6]中的游戏流量模型 $f(x)=\frac{1}{b}\mathrm{e}^{\frac{x-a}{b}}\mathrm{e}^{-\mathrm{e}^{\frac{x-a}{b}}}(b>0)$，其中，$a$, b 为参数；x 为数据包发送的时间间隔；$f(x)$ 表示 x 时间间隔内发包的概率。由该模型可以预测目的玩家节点的未来某时刻的网

络流量 Tra，这里 $Tra=\lim\limits_{x\to x_0}\left(f(x)\times package\_size\right)$，其中，$package\_size$ 为游戏中数据包大小。若目的节点能够满足网络流量的需求，应有 $Tra<\overline{Tra}$。Tra 最终可以化简为

$$Tra=\frac{e^{\frac{a}{b}}}{b}\times package\_size\times \lim_{x\to x_0}\left[\left(e^{-e^{\frac{a}{b}}}\right)^{e^{\frac{x}{b}}}e^{-\frac{x}{b}}\right]$$

最后本算法通过式子 $Opt=\frac{\zeta_{ij}^{A_q}(t)}{Tra}$ 选择出最佳的迁移目的地，该式保证代理服务对象总是迁移到具有最大 Opt 值的玩家节点处。

基于蚁群算法的代理服务迁移算法描述如下：

Step1 顺序取出 Q_j^m 中的玩家。

Step2 根据 $\tau_{k,A_i}(t)=\begin{cases}\dfrac{\lambda_k(A_i)}{\sum\limits_{p=0}^{m-1}\lambda_p(A_i)},k=1,2,\cdots,m & t\neq 0\\ C & t=0\end{cases}$ 计算出该玩家 k 关于 A_i 的信息素浓度。

Step3 如果 Q_j^m 中所有玩家均被遍历，那么转到 Step4，否则转到 Step1。

Step4 根据 $\zeta_{j,k}^{A_i}(t)=\begin{cases}\dfrac{\tau_{k,A_i}^{\alpha}(t)\mu_{k,A_i}^{\beta}(t)\hbar_k}{\sum\limits_{s=0}^{m-1}\left(\tau_{s,A_i}^{\alpha}(t)\mu_{s,A_i}^{\beta}(t)\hbar_k\right)} & k\in Q_j^m\\ 0 & \text{others}\end{cases}$ 计算出 A_i 转移到各个玩家节点上的概率。

Step5 根据 $Tra=\lim\limits_{x\to x_0}\left[\dfrac{e^{\frac{a}{b}}}{b}\left(e^{-e^{\frac{a}{b}}}\right)^{e^{\frac{x}{b}}}e^{-\frac{x}{b}}\times package\_size\right]$ 计算出各个目的玩家节点的 Tra。

Step6 通过 $Opt=\frac{\zeta_{ij}^{A_q}(t)}{Tra}$ 选择出最佳的(Opt 值最大的玩家节点)迁移目的地。

该算法根据 MMOG 中网络流量、代理服务需求程度和目的主机负载状况动态决定代理迁移目的地，符合 MMOG 运行时的网络状况和实际情况，适合 MMOG 中移动代理的迁移。

5 实验与分析

通过应用移动代理的 P2P MMOG 的中间件系统开发了一个简单的 P2P 游戏，玩家采用一个简单的球表示，魔法物体用简单的圆锥或者八面体表示。玩家通过吃掉魔法物体来改变自己的形态和颜色。P2P 环境采用 JXTA 来实现，代理平台采用 IBM Aglets V1.0。实验采用校园网中的 200 个计算机资源，分别在这些计算机上部署代理服务。实验将检测在应用 3 种代理迁移算法时各迁移算法的执行时间、网络延迟和某个区域内的 11 个玩家的 CPU 使用率的方差来比较 3 种算法作用下的负载情况。实验结果如图 2、图 3 以及表 1 所示。

从图 2 中可以看到，最佳解图算法的波动较大，受环境影响较多，其执行时间普遍偏高。一步迁移策略虽不能每次都达到最佳，但其平均执行时间要明显优于最佳解图算法。改进的蚁群算法的执行效率最高，随着执行次数的增加，达到一个较为平稳的状态，其执行时间普遍偏低。因此，该实验从算法的执行时间方面证明改进的蚁群算法更适合 MMOG。

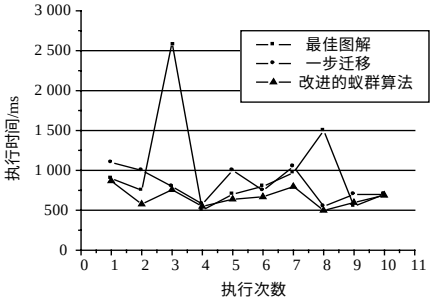

图 2　3 种算法执行时间的比较

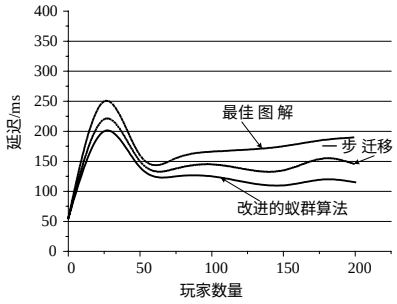

图 3　3 种算法对应延迟的比较

表 1　CPU 使用率方差比较

算法	PC1	PC2	PC3	PC4	PC5	PC6	PC7	PC8	PC9	PC10	PC11	方差
改进的蚁群算法	0.60	0.65	0.58	0.72	0.81	0.68	0.73	0.76	0.70	0.74	0.73	0.31
最佳图解	0.55	0.67	0.85	0.75	0.76	0.70	0.80	0.65	0.60	0.58	0.81	0.39
一步迁移	0.60	0.65	0.72	0.76	0.62	0.78	0.58	0.77	0.76	0.68	0.76	0.33

从图 3 中可以看到，在改进的蚁群算法作用下，代理服务能够快速地在网络中玩家节点间迁移，使得该代理服务对象能够快速地服务玩家，使得玩家的请求被快速地解决，降低了系统的延迟。因此，本实验从系统延迟方面证明改进的蚁群算法更适合 P2P MMOG。

表 1 记录了某区域内 11 个玩家的 CPU 使用率在 3 种迁移策略作用下第 12 分钟时刻的方差比较。方差越小，越能够符合负载均衡的需求。该实验结果表明改进的蚁群算法对应的方差值最小。因此，本实验从负载均衡方面证明改进的蚁群算法更能满足 MMOG 的需求。综上 3 个实验结果得出改进的蚁群算法更适合 P2P MMOG。

6 结束语

本文提出了一种适合 P2P MMOG 的移动代理迁移策略。实验证明，这种移动代理迁移策略提高了移动代理的迁移速度，降低了系统延迟，均衡了系统负载，满足 MMOG 实时性需求。本文在今后的研究中将就代理的内部处理机制进行更深入的研究。

参考文献

[1] Craus M, Bulancea C. Using Agent Technology Combined with ACO Metaheuristics on Load Balancing Algorithms[J]. Advances in Electrical and Computer Engineering, 2004, 4(2): 55-60.

[2] 刘大有, 杨　博, 杨　鲲, 等. 基于旅行图的移动 Agent 迁移策略[J]. 计算机研究与发展, 2003, 40(6): 838-845.

[3] Michlmayr E, Pany A, Graf S. Applying Ant-based Multi-Agent Systems to Query Routing in Distributed Environments[C]//Proceedings of the 3rd IEEE Conference on Intelligent Systems. London, UK: [s. n.], 2006.

(下转第 153 页)