

改进的蚁群算法在求解旅行 Agent 问题中的应用

邓江沙, 姚 刚

(长沙理工大学 计算机与通信工程学院, 湖南 长沙 410076)

摘 要: 蚁群算法作为一种新的生物进化算法, 具有并行、正反馈和启发式搜索等特点, 但它与其它进化算法同样存在易于陷入局部最小点等缺陷。为了克服这些缺陷, 介绍了一种改进的蚁群算法来求解旅行 Agent 问题, 解决移动 Agent 为完成用户指定任务, 在不同主机间移动时的迁移策略问题。实验结果表明了算法的可行性。

关键词: 蚁群算法; 迁移策略; 旅行 Agent 问题

中图分类号: TP301.6; O221.7

文献标识码: A

文章编号: 1673-629X(2006)07-0233-03

Application of Improved Ant Colony Optimization Algorithm
to Solve Traveling Agent Problem

DENG Jiang-sha, YAO Gang

(College of Computer and Communication, Changsha University of Science and Technology, Changsha 410076, China)

Abstract: Ant colony algorithm is a new evolutionary algorithm, has the characteristic of parallelism, positive feedback, heuristic search, but it has the limitation of stagnation like other evolutionary algorithms. To avoid the limitation, an improved ant colony optimization algorithm is introduced to solve the traveling agent problem, which is responsible for planning out an optimal migration strategy when agents migrate to several hosts for accomplishing its task. The experimental result shows that the algorithm is effective.

Key words: ant colony algorithm; migration strategy; traveling agent problem

0 引 言

移动 Agent 是一个代替人或其它程序执行某种任务的程序, 它在复杂的网络系统中能自主地从一台主机移动到另一台主机, 该程序能够选择何时、何地移动。在移动时, 该程序可以根据要求挂起其运行, 然后转移到网络的其它地方重新开始或继续其执行, 最后返回结果和消息^[1]。旅行 Agent 问题(TAP)就是移动 Agent 的迁移策略问题, 它根据移动 Agent 的任务、网络的软硬件环境和其它约束条件为 Agent 规划出最佳迁移路径^[2]。

蚁群算法是意大利学者 Dorigo 等人提出的一种模拟自然界蚁群行为的模拟进化算法^[3]。这种算法具有分布计算、信息正反馈和启发式搜索的特征, 是一种新型的启发式优化算法。蚁群算法在求解多种组合优化问题中获得了广泛的应用, 如调度、二次分配、网络路由等^[4]。旅行 Agent 问题为 NP-完全问题, 其时间、空间复杂度很高, 这就要求用于求解旅行 Agent 问题的方法一般必须具备自适应、自学习、分布式等特征。基于蚁群算法的特点, 用它来求解旅行 Agent 之类的组合优化问题极其适合。

1 旅行 Agent 问题

移动 Agent 在网络环境中执行用户指定的任务时, 当所在的主机无法满足其资源方面的要求时, 需要按照当前任务的需求及网络负载的情况选择移动到某台主机, 使用它提供的服务和资源, 之后再移动到其它的主机继续执行, 如此的交互过程反复进行, 直到整个任务执行完成或失败为止。旅行 Agent 问题的定义如下^[2]:

共有 n 台主机, Agent 在主机 i ($0 \leq i \leq n-1$) 上完成其任务的概率为 p_i , 这些概率彼此间相互独立。不论 Agent 在主机 i 上是否能完成任务, 它在主机 i 上因尝试执行任务而造成的时延均为 t_i , Agent 从主机 i 移动到主机 j 所需的时间为 $d(i, j)$ 。移动 Agent 从初始主机出发去执行某项任务, 若在某台主机完成任务, 则可由该主机直接返回初始主机, 而无需继续访问其余主机; 若未能完成任务, 则 Agent 需要继续移动到另一台主机直至其任务完成或遍历完所有主机均不能完成其任务即任务失败为止。在移动过程中每个主机至多访问一次, 对初始主机 0, 设置 $p_0 = t_0 = 0$ 。TAP 问题就是找出一条使得 Agent 完成任务所需时间的期望值最小的移动路径。

2 旅行 Agent 问题的求解

由 TAP 的定义可知, 当移动 Agent 在所有的主机上均未能完成其任务并最终返回初始主机时, TAP 就退化

收稿日期: 2005-10-25

作者简介: 邓江沙(1961-), 男, 湖南长沙人, 硕士生导师, 副教授, 研究方向为网络应用技术、面向对象技术。

为旅行商问题 (Traveling Salesman Problem, TSP), 所以 TSP 问题是 TAP 问题的一个特例^[5]。蚁群算法具有分布计算、信息正反馈和启发式搜索等特点, 在求解 TSP 等优化组合问题时, 在解的质量、收敛速度等方面具有较大的优势。笔者在蚁群算法的基础上, 通过修改它的全局和局部更新规则, 并引入自适应的信息素挥发系数^[6], 来求解 TAP 问题。

按照蚁群算法的思想, 可令每只蚂蚁代表一个执行用户指定任务的移动 Agent, 系统中同时有多只蚂蚁并行地求解同一问题, 每只蚂蚁的迁移路线就是所求 TAP 问题的一个解, 蚂蚁通过所得经验不断调整信息素强度, 使得 Agent 的迁移路线不断得到优化, 并最终得到最优解或近似最优解。

2.1 状态转移规则

蚁群算法在求解 TSP 问题时, 只需要考虑城市之间的距离, 但在 TAP 问题中, 对求解造成的影响不仅包括 Agent 在主机间移动的时间, 还包含各主机完成任务的概率和检测任务所费的时延。蚂蚁在选择下一个主机时, 除了倾向于花费时间短、信息素浓度高的路径外, 还会优先考虑那些完成任务概率高、计算时间短的主机。因为蚂蚁在访问过若干完成任务概率高、计算时间短的主机之后就已经完成了预定的任务, 不需要访问其它的主机就直接返回源主机。基于这样的思想, 定义状态转移规则如下:

第 k 只蚂蚁选择下一主机的概率:

$$p_k(r, s) = \begin{cases} \arg \max \{ [\tau(r, s)]^\alpha [\frac{p_s}{d(r, s) \cdot t_s}]^\beta \} & q \leq q_0, s \in J_k(r) \\ \frac{[\tau(r, s)]^\alpha [\frac{p_s}{d(r, s) \cdot t_s}]^\beta}{\sum_{u \in J_k(r)} [\tau(r, u)]^\alpha [\frac{p_u}{d(r, u) \cdot t_u}]^\beta} & q > q_0, s \in J_k(r) \\ 0 & \text{其它} \end{cases} \quad (1)$$

其中 r 是蚂蚁当前所在的主机, $\tau(r, s)$ 是主机 r 和 s 之间路径上的信息素浓度, $d(r, s)$ 为蚂蚁从主机 r 移动到 s 所需的时间, p_r 为蚂蚁在主机节点 r 完成任务的概率, t_r 为 r 处的时延, $J_k(r)$ 是蚂蚁 k 尚未访问过的主机集合。

2.2 信息素局部更新规则

当蚂蚁从主机 r 移动到主机 s 时, 路径 (r, s) 上的信息素按如下的公式进行局部更新:

$$\tau(r, s) = (1 - \rho) \cdot \tau(r, s) + \rho \cdot \Delta\tau(r, s) \quad (2)$$

$$\Delta\tau(r, s) = \sum_{k=1}^m \Delta\tau^k(r, s) \quad (3)$$

其中, $\Delta\tau^k(r, s)$ 表示第 k 只蚂蚁在本次循环中留在路径 (r, s) 上的信息素量, $\Delta\tau(r, s)$ 是所有经过路径 (r, s) 的蚂蚁留下的信息素的增量。

$$\Delta\tau^k(r, s) = \begin{cases} Q \cdot \frac{p_r + p_s}{d(r, s) \cdot t_s} & \text{若第 } k \text{ 只蚂蚁经过路径 } (r, s) \\ 0 & \text{否则} \end{cases} \quad (4)$$

其中 Q 为常数, p_r, p_s 为蚂蚁在主机节点 r, s 完成任务的概率, t_r, t_s 为 r, s 处的时延, $d(r, s)$ 为蚂蚁从主机 r 移动到 s 所需的时间。在蚁群算法中, $\Delta\tau(r, s)$ 常被设为一个初始的信息素强度 τ_0 , 这里采用类似蚁量系统中的局部

更新原则, $\Delta\tau(r, s)$ 考虑了主机 r, s 的状态, 与 p_r, p_s 成正比, 与 $d(r, s), t_r, t_s$ 成反比, 这样使蚂蚁在进行局部信息素更新时更有针对性, 在避免蚂蚁收敛到同一路径的同时, 也在一定程度上加快了搜索的速度。

2.3 信息素全局更新规则

为了使搜索过程更具有指导性, 使蚂蚁倾向于完成任务概率高、网络延时少的主机移动, 对全局新规则进行修改。当所有的蚂蚁完成一次循环后, 不仅对最优的路径进行增强, 而且对最差的路径进行削弱, 使得属于最优路径的边与属于最差路径的边之间的信息素强度的差异进一步增大, 从而使蚂蚁的搜索行为更集中于最优解的附近。

当 (r, s) 属于最优路径的一条边时:

$$\tau(r, s) = (1 - \rho) \cdot \tau(r, s) + \rho \cdot \Delta\tau \quad (5)$$

其中: $\Delta\tau = \frac{1}{T_{\text{best}}}$ 。

当 (r, s) 属于最差的路径上的一条边且不是最优路径上的一条边时:

$$\tau(r, s) = (1 - \rho) \cdot \tau(r, s) - \epsilon \frac{T_{\text{worst}} - T_{\text{best}}}{T_{\text{worst}} + T_{\text{best}}} \quad (6)$$

其中: T_{best} 为所有的蚂蚁在完成一次循环后的最优路径上所费的总时间, T_{worst} 为本次循环中最差路径上所费的总时间, ϵ 是一个常数。每次进行全局更新时, 最优的路径按 (5-14) 进行增强, 而最差的路径按 (5-15) 被减小了, 这样更有利于蚂蚁发现全局最优解。

2.4 自适应地调整 ρ 值

为了避免算法易于陷入局部最优解, 通过自适应地改变算法的挥发系数, 在保证收敛速度的条件下提高解的全局性。当问题规模较大时, 由于信息量的挥发系数 ρ 的存在, 那些从未被搜索到的路径信息素强度会减少至接近 0, 降低了算法的全局搜索能力, 当 ρ 过大, 以前搜索过的解被选择的可能性过大, 减少 ρ 虽然可以提高全局搜索能力, 但又会使算法的收敛速度降低, 因此对 ρ 值自适应改变, 在最优值在一定循环次数内没有明显改进时, 降低 ρ 值, 但不小于一个最小值。

$$\rho(t) = \begin{cases} \epsilon \cdot \rho(t-1) & \text{当 } \epsilon \cdot \rho(t-1) > \rho_{\min} \\ \rho_{\min} & \text{否则} \end{cases} \quad (7)$$

其中 ρ_{\min} 为 ρ 的最小值, 可以防止 ρ 过小降低算法的收敛速度。通过对挥发系数自适应地改变, 既可以提高解的全局性, 又可以保证收敛的速度。

求解 TAP 问题的算法步骤描述如下:

(1) 初始化。

设 $N_c = 0$, 迭代次数初值为 0;

$\tau(r, s) = \tau_0, \Delta\tau(r, s) = 0$, 为每条主机间路径 (r, s)

设置一个信息素强度初值, 信息素强度增量初值为 0; $\text{tabu}_k = \emptyset$, 将禁表置空;

将 m 只蚂蚁随机地置于 n 个主机节点上;

将蚂蚁的初始主机加到当前各自的禁表中;

(2) for $i = 1$ to n

for $k = 1$ to m

if 蚂蚁 k 没有完成指定的任务

按照公式(1) 计算 $p_k(r, s)$, 选择下一个将要访问的主机 s ;

将蚂蚁移到 s , 将 s 加入到它的禁表;

计算 $\Delta\tau^k(r, s), \Delta\tau(r, s)$ 的值, 根据公式(2)、(3) 对路径 (r, s) 上的信息素强度进行局部更新;

(3) for $k = 1$ to m

根据禁表求出本次循环中的最优解和最差的解;

if 最优解与 N 个迭代前的最优解相等

由公式(7) 更新 ρ 值;

由公式(5)、(6) 对最优路径与最差路径的信息素强度进行全局更新;

if 不满足终止条件

清空所有蚂蚁的禁表;

对每一条路径 (r, s) , 置 $\Delta\tau(r, s) = 0$;

$N_c = N_c + 1$;

返回步骤(2)

else 输出最优解

2.5 算法的性能评估

为了验证算法的性能, 依照实际网络拓扑结构产生所需要的数据来进行仿真实验。需要的数据包括: 主机数 n , 主机间的移动的时间 $d(i, j)$, 完成任务的概率 p_i , 主机的时延 t_i 等。

实验中需要设置的参数包括 α, β , 蚂蚁的数量等如下:

$\alpha = 1, \beta = 3$, 蚂蚁的数目笔者取与主机数目相同, ρ_{\min} 分别取 0.1 和 0.01。分别对蚁群算法和文中算法进行实验, 每项实验执行 10 次, 每次执行共运行循环 2000 次, 结果如表 1 所示。

由实验结果可以看出, 相对于传统的蚁群算法, 文中的算法具有更强的全局最优解搜索能力, 且收敛性更好。而通过调整 ρ_{\min} 值的大小, 能够在全局最优解和收敛性之

间寻求最佳的选择。

表 1 实验结果对比

	蚁群算法	文中算法($\rho_{\min} = 0.1$)	文中算法($\rho_{\min} = 0.01$)
最优解	134.1	112.7	108.9
迭代次数	1531	1206	1326

3 小结

蚁群算法作为一种新的生物进化算法, 具有分布计算、信息正反馈和启发式搜索等特点, 在求解一系列优化组合问题时, 在解的质量、收敛速度等方面具有较大的优势。但它也存在一些缺陷, 如需要较长的搜索时间, 当规模较大时, 还时可能陷入局部最优解, 产生过早收敛的问题。为克服蚁群算法的这些缺陷, 通过修改它的全局以及局部更新规则, 并引入自适应的挥发系数来求解 TAP 问题。实验结果表明, 相对于传统蚁群算法, 文中的改进算法不仅收敛速度较快, 而且具有更强的全局最优解搜索能力, 能很好地解决 TAP 问题。

参考文献:

- [1] 朱森良, 邱瑜. 移动代理系统综述[J]. 计算机研究与发展, 2001(1): 16-25.
- [2] Brewington B, Gray R, Moizumi K. Mobile agents in distributed information retrieval[A]. In: Klusch M. Intelligence Information Agent[C]. Berlin: Springer-Verlag, 1999. 355-395.
- [3] Dorigo M, Vittorio M, Alberto C. The Ant System: Optimization by a colony of cooperating agents[J]. IEEE Transactions on Systems, Man, and Cybernetics, 1996, 26(1): 1-13.
- [4] 李士勇. 蚁群算法及其应用[M]. 哈尔滨: 哈尔滨工业大学出版社, 2004.
- [5] 骆正虎. 移动 agent 系统若干关键技术问题研究[D]. 合肥: 合肥工业大学, 2002.
- [6] 王颖, 谢剑英. 一种自适应蚁群算法及其仿真研究[J]. 系统仿真学报, 2002, 14(1): 31-33.

(上接第 232 页)

程中关键的地物要素进行查询识别和数据浏览, 如了解指定物体的空间位置和属性信息, 从而实现相应的操作。

4 结束语

讨论了引江济太三维虚拟景观原型系统的设计和实现, 并介绍了系统场景可视化中采取的一些关键技术。目前该三维可视化系统已经投入使用, 能够在微机上流畅地运行, 图形的生成速度和质量还是令人满意的, 并提供了三维查询与空间分析功能。但系统在处理复杂场景时, 还存在一些不足之处, 有待于进一步改进和提高。

参考文献:

- [1] GB/T50095-98. 水文基本术语和符号标准[S]. 1998.

- [2] 李军, 景宁, 吴秋云, 等. 基于面向对象数据库的三维 GIS 实验系统[J]. 计算机辅助设计与图形学学报, 2003, 15(7): 880-885.
- [3] Garland M, Heckbert P. Surface simplification using quadric error metrics [A]. In: Computer Graphics (SIGGRAPH Proceedings) [C]. Los Angeles, CA, USA: [s. n.], 1997. 209-216.
- [4] Xia J C, Varshney A. Dynamic view-dependent simplification for polygonal models [A]. IEEE Visualization '96 [C]. San Francisco, CA, USA: [s. n.], 1996. 327-334.
- [5] 费广正, 乔林. Visual C++ 6.0 高级编程技术—OpenGL 篇[M]. 北京: 中国铁道出版社, 2000. 87-122.
- [6] 魏文礼, 沈永明. 二维溃坝洪水波演进的数值模拟[J]. 水利学报, 2003(9): 43-47.