

AntNet的多路径 QoS路由算法研究

朱尚明, 高大启

(华东理工大学 信息科学与工程学院, 上海 200237)

摘要:以 AntNet 算法为基础,介绍了蚁群网络路由的问题模型和数据结构,通过引入 QoS 约束机制,提出了一种基于 AntNet 的多路径 QoS 路由算法.该算法采用具有带宽和时延 QoS 约束的新规则进行节点选择,并利用改进的节点信息更新规则以及根据路由表中概率值随机地选择相邻节点转发数据包.性能分析和模拟结果显示,基于 AntNet 的多路径 QoS 路由算法具有较快的收敛速度和较好的鲁棒性,能够自适应网络状态的动态变化,同时考虑了 QoS 约束和负载均衡问题.

关键词:蚂蚁网络;多路径路由;QoS;服务质量约束

中图分类号: TP393 **文献标识码:** A **文章编号:** 1673-4785 (2008) 04-0349-06

A multipath QoS routing algorithm based on Ant Net

ZHU Shang-ming, GAO Da-qi

(School of Information Science and Engineering, East China University of Science and Technology, Shanghai 200237, China)

Abstract: This paper examines a mathematical model and data structure for a multipath QoS routing algorithm based on the AntNet algorithm. The proposed algorithm selects nodes with a new rule considering both bandwidth and time-delay QoS constraints. Then it transfers data packets using the improved updating rule for nodal information and randomly chooses neighboring nodes to transfer data packets according to probabilities in the routing table. Performance analysis and simulation results show that the multipath QoS routing algorithm based on AntNet converges faster and is more robust than other algorithms. It can automatically adapt to dynamic variations in network status while taking into account QoS constraints and load balancing.

Keywords: AntNet; multipath routing; QoS; constraint on service quality

蚁群 (ant colony, AC) 算法是由意大利学者 M. Dorigo 等人于 20 世纪 90 年代初提出的一种新型的模拟进化算法^[1-2]. 蚁群算法从生物学和仿生学角度出发,模拟蚂蚁从蚁巢寻找到食物源的最优路径的自然行为,通过由候选解组成的群体的进化过程来寻求最优解.

蚁群算法作为一种求解复杂组合优化问题的计算智能方法,可以用于求解网络路由问题.近年来国内外学者先后提出了很多方案^[3-10],其中以 G Di Caro 和 M. Dorigo 提出的蚂蚁网络 (AntNet) 为典型代表^[3]. AntNet 算法通过 2 类基于移动 Agent 的前行蚂蚁和后行蚂蚁共同合作,动态地达到网络的自

适应路由.实验表明, AntNet 路由算法和其他路由算法 (如 OSPF、SPF 等) 相比具有一定的优势^[4]. 本文将 AntNet 算法为基础,介绍蚁群网络路由的问题模型和数据结构,对其进行改进,引入 QoS 约束机制,提出一种基于 AntNet 的多路径 QoS 路由算法,并对其性能进行分析.

1 问题模型和数据结构

1.1 问题模型

AntNet 本质上是一个基于移动 Agent 的系统,算法中构造了 2 类结构基本相同的人工蚂蚁:前行蚂蚁 F_{ant} 和后行蚂蚁 B_{ant} . 其中 F_{ant} 表示从源节点到目的节点的人工蚂蚁,在其向目的节点前进的过程中收集信息; B_{ant} 表示从目的节点返回源节点的人工蚂蚁,在返回途中根据 F_{ant} 收集的信息进行路由表

收稿日期: 2007-08-20

基金项目: 国家自然科学基金资助项目 (60373073).

通信作者: 朱尚明. E-mail: zhushm@ecust.edu.cn

的更新.

把通信网看成一个无向加权连通图 $G(N, L)$, 其中 N 为节点集合, L 为图中连接 2 个节点的链路集合. 不失一般性, 假设在任意 2 个节点之间至多存在一条链路, 每条链路 $l \in L$ 和一些 QoS 度量相关, 这些度量包括: 时延、可用带宽、包丢失率、费用和抖动等. 简化起见, 这里仅考虑带宽和时延约束.

为了利用蚂蚁进行路由选择, 网络中每个节点的路由表用信息素 (pheromone) 表来代替, 表中的信息素浓度是以概率值的形式表示. 蚂蚁在一条路径上前进时沿途释放信息素, 在此相当于对节点信息素表中的相应条目进行更新. 时延小、拥塞程度低的链路应该具有更大的概率值.

假设 P_{jd} 表示在当前节点 k 以 d 为目的节点时选择下一个节点 j 的概率, 再假设 N_k 为节点 k 的相邻节点集合, 即 $N_k = \{ \text{neighbors}(k) \}$, 那么有

$$P_{jd} = 1, \quad d \in N_k, \quad j \in N_k. \quad (1)$$

对于任意一条从源节点 s 到目的节点 d 的路径 p , 设 B_0 为从源节点 s 到目的节点 d 路由一个业务流所需的最小带宽, D_0 为所容许的最大时延, 则多路径 QoS 路由可以表示为在源节点 s 到目的节点 d 之间寻找多个路径, 且每条路径满足:

$$\min \{ \text{bandwidth}(l), l \in p(s, d) \} \geq B_0, \quad (2)$$

$$\sum_{l \in p(s, d)} \text{delay}(l) \leq D_0. \quad (3)$$

1.2 数据结构

为了实现寻路和更新路由表 (信息素表), 把蚂蚁分成 2 类, 一类是收集节点间时延和路径状态信息的前行蚂蚁 F_{ant} , 一类为进行路由表和状态信息更新的后行蚂蚁 B_{ant} . 2 类蚂蚁具有相同的数据结构, 可以保存节点、时延等信息.

蚂蚁的数据结构定义为一个数据表和一个堆栈 $S_{s-d}(k)$. 数据表包含了蚂蚁标识、源节点、目的节点以及经过的跳数, 堆栈 $S_{s-d}(k)$ 包含了依次经过的网络节点标识 k , 从源节点 s 到此节点经历的时延 t_k 以及次序关系. 蚂蚁 F_{ant} 前行到节点 k , 其 $S_{s-d}(k)$ 包含信息的形式为

$$\{ (j, t_1), (j, t_2), \dots, (j, t_k) \}.$$

网络节点的数据结构主要由 2 个表构成: 路由表 R_k 和节点状态数组 $\{ \text{Stat}(1), \text{Stat}(2), \dots, \text{Stat}(N) \}$. 为了实现多路径路由, 本算法对路由表进行了改进, 增加了一个路径标识 $p_i (i = 1, 2, \dots, m)$, 表示从源节点 s 到目的节点 d 的多个路径. 节点状

态数组存储了本地的流量统计, 描绘了当前节点到其他目的节点的本地自适应拥塞状态模型. 以节点 k 为例, $\text{Stat}(k) = M_k$ 表示从当前节点 k 所观察到的整个网络各个节点距离的统计模型. AntNet 算法的 M_k 采用了时延统计表, 它由三元组 (μ_d, σ_d^2, W_d) 组成. μ_d 和 σ_d^2 表示了从当前节点到目的节点 d 所经历的时间的均值和方差, W_d 是一个移动观察窗口, 窗口大小表示了最近经过节点 k 的、以 d 为目的节点的蚂蚁个数, 用来计算行程的最少时间. 时延统计表 M_k 的计算非常复杂, 涉及的参数较多, 本算法对其进行了改进, 引入压缩和强化机制, 简化利用时延来调整信息素的计算方法.

2 AntNet 的多路径 QoS 路由算法

2.1 路由表的初始化

在 AntNet 算法中没有具体定义路由表的初始化方法. 为了能够充分反映网络的初始拓扑状况, 考虑到要充分利用网络节点局部的信息, 本算法按照式 (4) 对路由表进行初始化:

$$P_{jd} = \begin{cases} \frac{1}{|N_k|} + \frac{(|N_k| - 1)}{|N_k|^2}, & j = d, j \in N_k, j = d; \\ \frac{1}{|N_k|} - \frac{1}{|N_k|^2}, & j \neq d, j \in N_k, j \neq d, |N_k| > 1; \\ \frac{1}{|N_k|}, & j \in N_k, d \notin N_k. \end{cases} \quad (4)$$

式 (4) 根据目的节点的不同类型 (邻接点、非邻接点) 而设置不同的初始概率值. 其中 $\frac{1}{|N_k|}$ 是提出的路由表先验因子, 代表概率增减量与原概率的权重, 可以通过实验确定; $|N_k|$ 代表该网络节点的邻居个数.

2.2 节点选择规则

在基本的 AntNet 算法中, 前行蚂蚁选择下一节点是根据路由表 (信息素表) 中的概率值随机选择的, 而数据流中数据包选择下一节点则采用比较本节点 k 的信息素表来选择下一个可能节点 v , 即按式 (5) 选择概率值最大的相邻节点进行转发.

$$p(k, v) = \max_{j \in N_k} P_{jd}. \quad (5)$$

式中: $d \in [1, N], N_k = \{ \text{neighbors}(k) \}$.

这种设计首先没有考虑到带宽、费用等其他的 QoS 约束, 其次当系统稳定时, 由于蚁群算法的正反馈自寻优能力, 若网络中各链路时延不变, 则经过一段时间后, 几乎所有的蚂蚁都会选择相同的路径. 当业务流量较大的时候就会发生拥塞. 为了防止这种

现象,本算法采用具有带宽和时延 QoS约束、并考虑相应链路队列状态的新规则进行节点选择.另外,还采用了改进的节点信息更新规则,以及根据路由表中概率值随机地选择相邻节点进行数据包转发来提高算法的性能.

在节点 k ,首先计算一个新的满足带宽约束要求的邻居集合 N_k ,节点 k 与 N_k 中的每个节点 v 的链路 $l(k, v)$ 的带宽都大于等于所需的最小带宽 B_0 ,即

$$N_k = \{v: v \in N_k, \text{bandwidth}(k, v) \geq B_0\}. \quad (6)$$

式中: $N_k = \{\text{neighbors}(k)\}$.然后再根据相应链路队列状态和概率表进行节点选择,选择规则描述如下:

1) 如果 N_k 中一个可行的相邻节点就是目的节点 d ,则蚂蚁将无条件选择这个相邻节点; 2) 如果 N_k 中存在以前所有蚂蚁都没走过的相邻节点,则按式(7)计算新概率 P_{jd} ,并在其中按概率 P_{jd} 的最大值随机地选择; 3) 如果 N_k 中的相邻节点都有以前的蚂蚁访问过,在尽量不选本身蚂蚁走过的节点的前提下,则在按式(7)计算新概率 P_{jd} ,并在其中按概率 P_{jd} 的最大值随机地选择.

$$\begin{cases} P_{jd} = \frac{P_{jd} + l_j}{1 + (\sum_{j \in N_k} l_j - 1)}, & j \in N_k; \\ l_j = 1 - \frac{q_j}{\sum_{j \in N_k} q_j}, & 0 \leq l_j \leq 1 \end{cases} \quad (7)$$

式中: l_j 称为启发式校正系数,与当前节点 k 的相邻节点 j 的缓冲区状态成比例, q_j 表示节点 k 与相邻节点 j 的链路队列长度,是链路队列状态与路由概率的权重因子,用于衡量启发式校正系数 l_j 与路由表中所存储的概率值 P_{jd} 相比的重要性.显然,等待队列越长的链路,被选择的概率就越小.的最佳取值由于不同的问题特性而有所变化,如果取值太小, l_j 的作用就体现不出来;如果太大,所得到的路由表会出现振荡,这两者都会使得性能降低.

2.3 节点信息更新规则

2.3.1 路由表更新规则

对于当前节点 k ,其路由概率表更新的规则为:如果相邻节点是后行蚂蚁过来的节点,那么下一节点的概率值按式(8)进行增加,否则按式(9)进行减少.

$$P_{id} = P_{id} + r(1 - P_{id}), \quad (8)$$

$$P_{jd} = P_{jd} - rP_{jd}, \quad j \neq i, j \in N_k. \quad (9)$$

式中:节点 i 是后向蚂蚁过来的节点,节点 j 是当前节点的相邻节点, N_k 为当前节点的相邻节点集合, d 是目的节点, r 为增强因子.每一步计算之后,都要

进行归一化处理.

增强因子 $r \in (0, 1]$,并按式(10)进行动态计算:

$$r = c_1 \frac{t_{best_d}}{t_{-d}} + c_2 \frac{t_{sup} - t_{best_d}}{(t_{sup} - t_{best_d}) + (t_{k-d} - t_{best_d})}. \quad (10)$$

式中: c_1 和 c_2 为权重系数, t_{k-d} 为前行蚂蚁新近记录的从当前节点 k 到目的节点 d 的时延, t_{best_d} 是在观察窗口 W_d 内向目的节点 d 前行过程中所经历的最小时间, t_{sup} 是根据置信水平所确定的时延置信区间的上限,并由式(11)进行动态计算:

$$t_{sup} = \mu_d + \frac{d}{\sqrt{1 - \frac{1}{\sqrt{|W_{max}|}}}}. \quad (11)$$

式中: α 为置信水平, $|W_{max}|$ 为观察窗口的大小.

后向蚂蚁根据前向蚂蚁得到的时延值的“好坏程度”对概率表进行更新,式(10)的第1项反映了当前的时延值与最小时延值的比率,第2项反映了当前的时延值与最小时延的置信区间的相关度.当 t_{k-d} 的值不稳定时, r 的值不能完全反映网络的时延特性,因此本算法做了如下处理:使用压缩公式(12)来增加或减少 r 的值.这样处理使“好的程度”越大的时延值对路由表概率值的影响越大,而“好的程度越小”的时延值对概率值影响越小.

$$\begin{cases} s(x) = \frac{1}{1 + e^{\frac{a}{x|N_k|}}}, & x \in (0, 1], a \in R^+, \\ r = \frac{s(r)}{s(1)}. \end{cases} \quad (12)$$

式中: $s(x)$ 为压缩函数, $\frac{a}{|N_k|}$ 确定了压缩函数和相邻节点 N_k 的个数的相关度, a 为压缩系数,应为一个大于1的正实数.

2.3.2 时延表的更新

对于当前节点 k ,其时延表按式(13)、(14)进行更新:

$$\mu_d = \mu_d + (t_{k-d} - \mu_d), \quad (13)$$

$$\sigma_d^2 = \sigma_d^2 + ((t_{k-d} - \mu_d)^2 - \sigma_d^2). \quad (14)$$

式中: μ_d 为从本节点到目的节点 d 的时延平均值, σ_d^2 为时延方差, α 为权重参数,它反映了最近的几个样本对这个整体均值和方差的影响,通常与有效采样窗口 W 的关系可以表示为 $|W| = 5(\alpha / \sigma)$.

2.4 数据包路由和多路径选择

从源节点向目的节点发送路由数据包时,使用和前行蚂蚁不同的路由表.为了满足时延约束,在对数据包根据路由表中概率值随机地选择相邻节点时,首先比较相邻节点到目的节点的时延均值 μ_d ,

只有时延均值小于容许累积时延 D_0 的节点才有可能被选中. 然后, 对上述比较得到的数据包路由表通过式 (15) 的指数函数进行重映射, 并在映射后重新进行归一化处理.

$$g(x) = x, \quad 1 \quad (15)$$

式中: α 为强化系数, α 的值决定了数据包选择单路径路由还是多路径路由. $\alpha > 1$ 时, 指数函数 $g(x)$ 增强了高概率值部分, 减弱了低概率值部分, 避免了数据包选择概率过低的链路. α 值较大时 ($\alpha \gg 1$), 表示数据包仅能选择单一路径进行路由, $\alpha = 1$ 表示可以选择多路径路由, 即使概率较低的链路也有可能被选中.

为了改进网络性能, 实现不相交多路径路由, 在此引入链路剩余带宽和在瓶颈链路上互不相交的多路径的概念. 对于 AnNet 算法选出的从源节点 s 到达目的节点 d 的一条路径, 将对构成该路径的每个链路的剩余带宽进行调整. 为简单起见, 在当前节点 k , 可以在每次按概率表选路之后, 通过式对节点 k 和相邻节点 v 之间的链路的剩余带宽进行修正. 修正之后, 再根据式 (6) 计算一个新的满足带宽约束要求的邻居集合 N_k , 这样便实现了瓶颈链路上的多路径互不相交:

$$\text{bandwidth}(k, v) = \text{bandwidth}(k, v) - B_0. \quad (16)$$

以上过程在每次执行 AnNet 寻路算法之前执行一次, 直到多路径集的个数达到一定的阈值或满足要求的相邻节点集合为空.

2.5 算法描述

基于 AnNet 的 QoS 路由算法的详细过程为

1) 首先对路由表初始化, 并以一定的间隔, 从每个节点 s (作为源节点), 随机选择一目的节点 d , 发送一个前行蚂蚁.

2) 前行蚂蚁按节点选择规则选择下一节点, 当蚂蚁从节点 i 到达下一节点 j 时, 它把链路 $l(i, j)$ 的时延保存到记忆栈中, 并把节点 i 加入其节点集合中, 节点跳数加 1.

3) 如果前向蚂蚁在寻路过程中出现环路, 则把该部分删除, 并从出现环路的开始节点开始进行搜索. 为防止前向蚂蚁无休止地进行循环, 根据网络规模设置一个蚂蚁的最大跳数, 如果蚂蚁经过的跳数大于这个最大值, 则蚂蚁死亡, 丢弃该蚂蚁.

4) 前行蚂蚁到达目的节点 d 时, 将产生一个后行蚂蚁, 并拷贝相应的信息到后行蚂蚁中, 后行蚂蚁

按原路返回到源节点 s , 并根据所述的信息更新规则对沿途各节点的信息进行更新. 如果后向蚂蚁不能按原路返回 (如出现链路故障等), 则将该蚂蚁丢弃.

5) 对于被选择的链路, 修正其剩余带宽, 以实现瓶颈链路上的多路径互不相交.

6) 对于多个多路径, 上述步骤重复进行, 直到多路径集的个数达到一定的阈值 m 或满足要求的相邻节点集合为空.

7) 每个网络节点得到一张具有多路径信息和选择下一节点的概率值的路由表, 当有呼叫请求接入时, 将对数据包选择一个路径并在该路径路由表中根据概率值随机地选择相邻节点进行转发.

3 算法分析与模拟

3.1 复杂度分析

对于从给定的源节点到目的节点前行的单个蚂蚁, 由于前行时需要每个节点的相邻节点计算转移概率, 设前行蚂蚁的最大跳数为 M , 最坏情况下需要在所有 M 个节点上进行搜索和计算, 其计算复杂度为 $O(M|N_k|)$. 单个后行蚂蚁由于需要更新每个节点的相邻节点路由表, 最坏情况下也需要在所有 M 个节点上计算和更新, 其计算复杂度也是 $O(M|N_k|)$.

显然, 对于从给定的源节点到目的节点前行的 m 个前行或后行蚂蚁, 本算法最坏情况下的计算复杂度为 $O(mM|N_k|)$. Dijkstra 最短路径算法计算一条路径的复杂度是 $O(L + N \log N)$, 其中 L 是网络中链路的个数, N 是网络中节点的个数. 可以看出, 当 M 和 $|N_k|$ 与节点个数 N 相比较小时, 基于 AnNet 的路由算法和 Dijkstra 最短路径算法在计算复杂度上是相当的.

3.2 模拟结果

为了对基于 AnNet 的 QoS 路由算法的性能进行分析, 本文对算法进行了模拟实验. 模拟实验首先产生一个随机网络 $G_p(N)$, 该网络由 N 个节点和以概率 p 选择的独立链路构成. 链路权重表示时延, 统一分布在 $(0, 1]$, 单位为 ms, 每个链路的带宽容量为 8 Mbps.

每次模拟都定义节点 1 为源节点, 节点 N 为目的节点, 并将模拟运行结果和 Dijkstra 最短路径算法进行比较. 模拟时间分为训练和测试 2 部分, 训练时仅产生和传输网络蚂蚁, 测试时网络蚂蚁和数据包同时产生. 数据包产生过程服从泊松分布, 数据包

大小服从负指数分布. 算法运行时的各种参数设置如表 1 所示.

表 1 算法运行时的各种参数设置
Table 1 Various parameters in simulations

Name and simbol	Values
前行蚂蚁初始大小 /bit	192
后行蚂蚁大小 /bit	500
先验因子	1.5
新样本权重	0.1
观察窗口大小 w	50
队列长度权重	0.2
最小时延比率权重 c_1	0.7
时延置信区间权重 c_2	0.3
压缩系数 a	$N \times p$
强化系数	3
置信区间	0.95
最小可用带宽 B_0 /Mbps	4
最大累积时延 D_0 /ms	2
数据包到达时间均值 /ms	12.5
数据包生存时间 /ms	10
数据包大小均值 /bit	4 096

图 1~3 分别给出了基于 AnNet 的路由算法和 Dijkstra 最短路径算法的关于路径跳数的概率密度函数 (probability density function, PDF), 可以看出, 基于 AnNet 的路由算法都可以达到较好的收敛性.

图 1 所示的模拟实验中, 在链路选择概率 p 不变的情况下, 比较节点个数 N 不同时对算法的影响, 设置了 $N=50, p=0.2$ 和 $N=25, p=0.2$ 等 2 种场景. 可以看出, 在节点规模较小时, 基于 AnNet 的路由算法的表现出来的性能也较好, 基本接近 Dijkstra 最短路径算法. 在 $N=50, p=0.2$ 时, Dijkstra 算法和 AnNet 算法路径跳数的均值分别为 3.60 和 5.27, 而在 $N=25, p=0.2$ 时, Dijkstra 算法和 AnNet 算法路径跳数的均值则为 3.02 和 3.41.

图 2 所示的模拟实验中, 在节点个数 N 不变的情况下, 比较链路选择概率 p 不同时对算法的影响, 设置了 $N=25, p=0.1$ 和 $N=25, p=0.2$ 等 2 种场

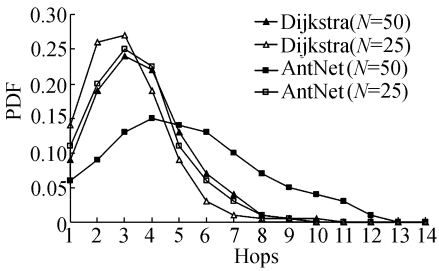


图 1 AnNet 算法在 $N=50, p=0.2$ 和 $N=25, p=0.2$ 时
路径跳数的概率密度函数

Fig 1 PDF of the hopcount of AnNet algorithm paths
for $N=50, p=0.2$ and $N=25, p=0.2$

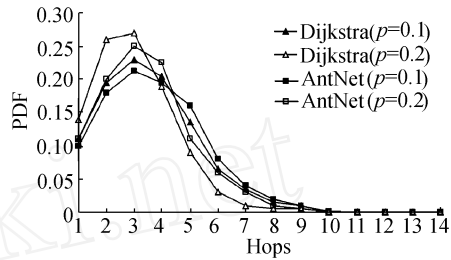


图 2 AnNet 算法在 $N=25, p=0.1$ 和 $N=25, p=0.2$ 时
路径跳数的概率密度函数

Fig 2 PDF of the hopcount of AnNet algorithm paths
for $N=25, p=0.1$ and $N=25, p=0.2$

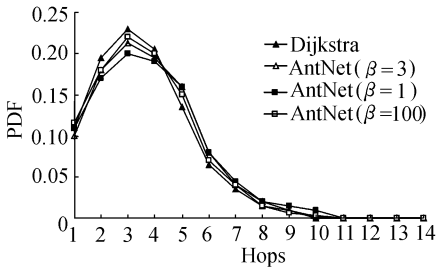


图 3 AnNet 算法在 $\beta=1, 3$ 和 100 时
路径跳数的概率密度函数

Fig 3 PDF of the hopcount of AnNet algorithm paths
for $\beta=1, 3$ and 100

景. 可以看出, 在节点规模一定时, 如果链路选择概率增大 (或减小), 由于可供路由的链路随之增大 (或减小), 2 种算法所选择路径的跳数会相应变小 (或变大). 在 $N=25, p=0.1$ 时, Dijkstra 算法和 AnNet 算法路径跳数的均值分别为 3.53 和 3.71, 而在 $N=25, p=0.2$ 时, Dijkstra 算法和 AnNet 算法路径跳数的均值则为 3.02 和 3.41.

图 3 所示的模拟实验中, 在节点个数 N 和链路

选择概率 p 不变的情况下, 比较强化系数 不同时对算法的影响. 从图中可以看出, 在网络规模一定时, 的变化对算法性能的影响不大, 说明该算法具有较好的鲁棒性. 值较大时, AnNet算法的性能更接近 Dijkstra最短路径算法, 这和前面的分析是一致的. 在 $N = 25$ 、 $p = 0.2$ 的情况下, Dijkstra算法和 AnNet算法在 $\alpha = 1$ 、3 和 100 时路径跳数的均值分别为 3.53、3.80、3.71 和 3.59.

4 结束语

采用蚁群优化算法来搜集网络最新信息, 可以自适应网络状态的动态变化. 通过改进选择策略, 动态更新路由表项和节点状态信息, 可以解决网络的负载均衡问题, 提高了网络性能. 基于 AnNet 的多路径 QoS 路由算法具有较快的收敛速度和较好的鲁棒性, 同时考虑了满足 QoS 约束和负载均衡等问题. 基于蚁群算法来求解网络路由问题的研究, 尚处于刚刚起步阶段, 其作为一种智能计算方法, 具有很好的发展前景.

参考文献:

- [1] DORIGO M. Optimization learning and natural algorithm [D]. Vatican: Politecnico di Milano, 1992.
- [2] DORIGO M, MANIEZZO V, COLONIA A. The ant system: optimization by a colony of cooperating Agents [J]. IEEE Transactions on Systems, Man, and Cybernetics Part B, 1996, 26(1): 29-41.
- [3] CARO G, DORIGO M. AnNet distributed stigmergetic control for communications networks [J]. Journal of Artificial Intelligence Research, 1998(9): 317-365.
- [4] DHILLON S S, MIEGHEM P V. Performance analysis of the AnNet algorithm [J]. Computer Networks, 2007(51): 2104-2125.
- [5] BARÁN B. Improved AnNet routing [J]. ACM SIGCOMM Computer Communication Review, 2001, 31(2): 42-48.
- [6] BARAN B, SOSA R. A new approach for AnNet routing [C] // Proceedings of Ninth International Conference on

Computer Communications and Networks Las Vegas, NV, USA, 2000.

- [7] 王利, 郭巧. 基于理性的蚁群自适应路由 [J]. 通信学报, 2005, 26(1): 6-10.
- WANG Li, GUO Qiao. Rationality-based AnNet self-adaptive routing [J]. Journal on Communications, 2005, 26(1): 6-10.
- [8] 吕勇, 赵光宙, 苏凡军. 基于蚁群算法的自适应动态路由算法 [J]. 浙江大学学报 (工学版), 2005, 39(10): 1537-1540.
- LÜ Yong, ZHAO Guangzhou, SU Fanjun. Adaptive dynamic routing algorithm based on AnNet algorithm [J]. Journal of Zhejiang University (Engineering Science), 2005, 39(10): 1537-1540.
- [9] 卢正鼎, 刘会明. 基于蚁群算法的理性自适应路由研究 [J]. 计算机工程与科学, 2006, 28(12): 15-18.
- LU Zhengding, LIU Huiming. Research of rationally adaptive routing based on ant colony algorithms [J]. Computer Engineering & Science, 2006, 28(12): 15-18.
- [10] 潘达儒, 袁艳波. 一种基于 AnNet 改进的 QoS 路由算法 [J]. 小型微型计算机系统, 2006, 27(7): 1169-1174.
- PAN Daru, YUAN Yanbo. Improved QoS routing algorithm based on the AnNet [J]. MiniMicro Systems, 2006, 27(7): 1169-1174.

作者简介:



朱尚明, 男, 1969 年生, 副教授, 主要研究方向为计算机网络、多媒体通信和智能理论, 发表学术论文 40 余篇, 出版著作 3 部.



高大启, 男, 1957 年生, 教授, 博士生导师, 主要研究方向为模式识别、神经网络、计算机嗅觉和信号处理, 学科带头人, 先后主持了 10 多项科研项目, 发表学术论文 100 余篇, 其中被 SCI、EI 和 ISTP 等收录近 60 篇.