

文章编号:1001-9081(2006)08-1776-03

## 移动 Ad hoc 网络中一种基于电池量的路由算法

张 毅<sup>1,2</sup>, 王小非<sup>2</sup>

(1. 哈尔滨工程大学 计算机科学与技术学院, 黑龙江 哈尔滨 150001;

2. 武汉数字工程研究所, 湖北 武汉 430074)

(zhangyi6898@yahoo.com)

**摘 要:**介绍了一种利用移动 Agent 来解决 Ad hoc 网络环境中基于电池量的路由问题的方法。首先通过移动 Agent 和各节点进行数据交换,了解网络中所有节点的连接信息,形成一个节点信息矩阵表;然后在该矩阵表的基础上,根据各节点电池余量,选择最合适的路径进行数据报文的发送。由于这种方法可以使用很少的 Agent 获得全局电池量的信息,因此可以减少维持节点信息而产生的开销。实验结果表明这种路由算法可以使各节点电池量的消耗趋于平衡。

**关键词:**Ad hoc; 电池量; 移动代理

**中图分类号:**TP393.03 **文献标识码:**A

## Battery-power-based routing algorithm for mobile Ad hoc networks

ZHANG Yi<sup>1,2</sup>, WANG Xiao-fei<sup>2</sup>

(1. College of Computer Science and Technology, Harbin Engineering University, Harbin Heilongjiang 150001, China;

2. Institute of Wuhan Digital Engineering, Wuhan Hubei 430074, China)

**Abstract:** A method which use mobile Agent to solve the battery-power-routing for mobile Ad hoc networks was introduced. A few mobile Agents communicated with every node to collect the network connection information to build the global information matrix of nodes. By information matrix, data packets were routed and transferred according to the remaining battery power. This approach needs fewer mobile Agents but gets more global battery power information, therefore it can reduce the cost of maintaining the routing information on each node. The experiment results show that the battery power on each node can be balanced with this routing algorithm.

**Key words:** Ad hoc; battery power; mobile Agent

## 0 引言

移动 Ad hoc 网络(Mobile Ad hoc Network, MANET)是由一组无线用户(节点)组成的自治集合<sup>[1,2]</sup>。它不需要固定的基站支持,无中心管理,可临时组织,并且具有高度移动性,可广泛应用于军事战术通信、应急通信、临时通信和传感器网络,以及其他需要快速布署、动态组网的通信场合。

Ad hoc 网络的特点是各节点地位平等,能自由移动,并且通过无线信道进行通信。由于移动终端本身由电池供电,因此总希望每个节点电量充足,如果某些节点因电池量不足而无法工作,它不仅影响本节点无法工作,同时会造成信号不能转发,从而影响整个 Ad hoc 网络的正常运转。因此,Ad hoc 节能机制的研究正成为一个热点,文献[3]利用电源休眠的方式节能,文献[4]通过路由方式节能。

利用移动 Agent 来解决基于电池量的路由问题是一种十分新颖的方法。移动 Agent 是一段程序代码,能控制自己在网络中移动,并能在每个节点独立地完成各种不同的任务<sup>[5]</sup>。移动 Agent 在分布式应用中十分有效,特别适用于动态的网络环境<sup>[6]</sup>。这些 Agent 在节点之间跳动,在节点中收集信息,并能将这些收集的信息给新的节点和 Agent。这样在

很短的时间内,每个节点都能接受到由 Agent 访问它们带来的更新信息。网络开始运行时,所有节点只知道它们自己和邻居的信息,而不知道别的节点的信息。当 Agent 开始路由时,这些节点就会得到别的节点的信息。

本文采用了基于移动 Agent 路由方式,使得网络中的每个节点可以知道节点之间的连接关系以及节点的电池量信息,从而不用消耗大量的网络资源,用很小的开销就可以实现在充分利用电池的基础上支持数据报文的正确传输。

## 1 Ad hoc 网络电池量消耗的基本原理

### 1) 跳数越少,电池量损耗越小

假定数据包  $j$  传送经过  $n_1, \dots, n_k$  节点,其中  $n_1$  是源节点,  $n_k$  是终点,  $T(a, b)$  代表从  $a$  到  $b$  一跳操作发送(接收)所消耗的能量。那么,数据包  $j$  传送消耗的中能量为:

$$e_j = \sum_{i=1}^{k-1} T(n_i, n_{i+1})$$

从上述公式可以看出,路由选择的跳数越少,消耗的能量就越少。

### 2) 总是耗用一个节点

如图 1 所示:如果按最小跳数选择,就会出现数据包在

收稿日期:2006-02-13;修订日期:2006-04-24

基金项目:“十五”国防预研计划资助项目(41306050103);国防科工委“十一五”预研计划资助项目(C0820061362)

作者简介:张毅(1968-),男,湖北武汉人,高级工程师,博士研究生,主要研究方向:网络管理、网络安全; 王小非(1957-),男,湖北武汉人,研究员,博士生导师,主要研究方向:计算机网络安全。

0→3, 1→4, 2→5 之间传输都要通过节点 6, 这样节点 6 的电池消耗会很大, 它的电池量会首先消耗光, 因而应避免总是使用同样一个节点来作为路由的中转。为节省电池的消耗, 宁可选择多跳, 如 0→3 可选择 0→1→2→3。

综上两点所述, 在电池量高于设定的最小阈值时, 尽量采用跳数少的路径; 在跳数相同的情况下, 避免选择电池量较少的那一条路径(比较路径节点电池的最小值); 若某一跳内某些节点电池量低于了设定的最小阈值时, 选择多一跳操作。

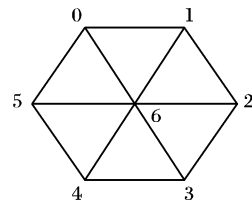


图1 电池能量图例

## 2 基于移动 Agent 的路由算法

首先通过移动 Agent 和各节点进行数据交换, 通过基于移动 Agent 的路由算法了解网络中所有节点的连接信息, 形成一个节点信息矩阵表, 然后在该矩阵表的基础上进行数据报文的传送。由于节点信息矩阵表是通过移动 Agent 移动来实现的, 而 Agent 的数量很少, 所以可以大大地减少维持节点信息而产生的开销。

### 2.1 相关概念

(1) 相邻关系( $a_{ij}$ ): 在无线环境下, 每个节点都有一个传输范围。我们定义在节点  $n$  传输范围  $R$  内的节点为  $n$  的邻居。假定节点  $n$  发送一个数据包, 它就可以传输到它所有的邻居节点。我们定义如果节点  $i$  为节点  $j$  的邻居, 则  $a_{ij} = 1$ ; 如果不是邻居, 则  $a_{ij} = 0$ 。

(2) 计数器(counter): 每个节点上设置一个 counter, 当网络开始运行时它们全部计数为 0。Agent 从一个节点跳到另一个节点收集分布节点的信息, 当一个 Agent 完成信息交换任务离开节点时, 就将该节点的 counter 增加 1。显然, counter 的大小代表了该节点被 Agent 访问过的次数。同样说明如果两个 Agent 有许多关于相同节点数据, 例如节点  $i$ , 那么 counter 更大的 Agent 携带的关于  $i$  节点的数据会更新。

(3) Agent 的数量: Agent 数量过少, 会导致节点的信息交换过慢, 特别是在节点速度较慢时; Agent 数量过多, 会使得 Agent 交换的信息变得雷同, 另外增加 Agent 数量还会增加网络拥塞。因而应合理地选择 Agent 的数量, 达到既能较好地满足快速信息交换的需求, 又保持网络流量合适。通过实验发现 Agent 的数量为系统中节点数量的一半比较合适。

(4) Agent 的数据结构: 一个 Agent 由下列几部分组成: Agent 标识符 id; Agent 程序 P; Agent 数据包(包含各节点的状态变量)。

Agent 数据包里包含了一些状态参数, 如  $a_{ij}$ , counter 等, Agent 能够和其他 Agent 和节点分享数据包里的内容。这些状态变量在 Agent 离开节点之前需要更新。

### 2.2 基于移动 Agent 的路由算法

Agent 的首要任务是传送每个节点的信息到另外一些节点。Agent 上有数据包, 每个 Agent 移动的同时, 用从别的节点或 Agent 上收集的最新信息更新 Agent 上数据包里的内容; 每个节点上有一个共享 cache, 节点通过共享 cache 了解整个网络的情况, 节点利用 Agent 数据包里最新的信息更新覆盖 cache 过时的信息。为了使开销达到最小, 采用“最先访问访

问次数最少的节点”的算法。在这个算法中, Agent 选择最小 counter 值的节点作为其下一次访问的目标节点, 因为该节点被访问过的次数最少。

Agent 的路由算法如下(如图 2 所示):

(1) Agent 到达节点  $n$ , 将 Agent 里数据包里存储的节点的 counter 值与存在当前节点中的所有其他节点的 counter 值进行比较, counter 值大的显然数据会更新, 于是将数据包里更新的信息复制到节点的 cache 上;

(2) 在该节点的 cache 上比较所有邻居的 counter 值, 取最小值, 该值的节点就是下一个可能要访问的节点;

(3) 如果被选择的节点最近 3 次没有被访问过(最近 3 次访问的历史节点可以在 cache 表上访问到), 该节点就是 Agent 下一个要访问的节点; 如果被选的节点在最近三次被访问过, 则选择下一个最少访问的邻居, 以此类推, 从而保证从同一个节点出发的 Agent 不会连续地访问同样一个节点;

(4) 将节点  $n$  的 counter 值加 1, 即将访问的节点加入到节点  $n$  的历史信息中;

(5) Agent 离开前, 将节点  $n$  里 cache 上的内容存储到 Agent 的数据包里;

(6) Agent 继续访问下一个节点。

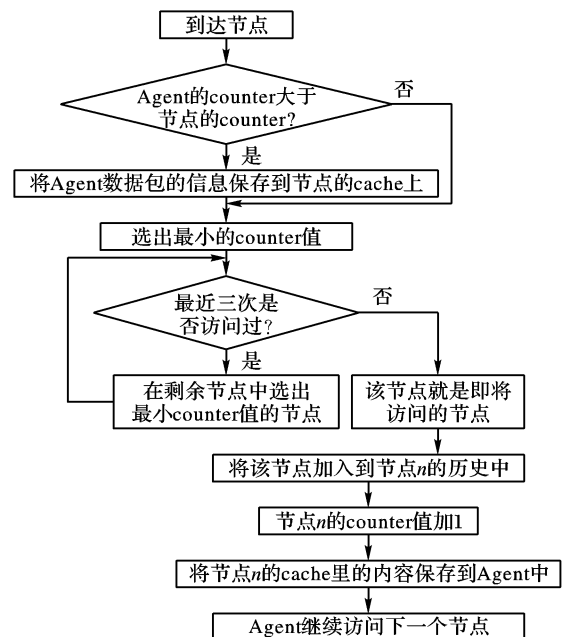


图2 Agent 的路由算法

## 3 基于电池量的数据报文路由算法

由基于 Agent 的路由算法得到了各节点的信息矩阵表, 当有数据报文需要传送时, 选择基于电池量的数据报文路由算法进行路由选择。

路由选择算法采用广度优先搜索算法, 以目的节点为树的根节点, 每一跳形成树的一层, 该树的最大层数为网络节点的数量。

如果节点  $i$  和节点  $j$  之间通信, 路径选择算法如下:

(1) 查看 cache 上节点  $i$  和节点  $j$  之间是否有连接, 即查看  $a_{ij}$  是否为 0, 如果不是 0, 说明有连接, 则路由结束, 信号直接发送; 如是 0, 说明不存在一跳操作, 则进入下一步。

(2) 查看 cache 上  $j$  节点的列式  $a_j$ , 找出  $a_j$  列中所有的非零项, 这些非零项就是节点  $j$  的子节点; 如果  $a_j$  列中所有值都

为零,则节点  $i$  和  $j$  不存在有效的路径,路由结束。

(3) 继续将上述子节点根据其相应的列找出其子节点,查看子节点中是否有  $i$  节点,若存在多个  $i$  节点,则形成多条路径,对多条路径的所有节点的电池余量进行排序,然后比较各条路径中电池余量的最小值,这些值中最大者,若大于最小电池量阈值  $\eta_0$ ,则该路径为所选最佳路径;若存在相同值,则继续比较每条路径次最小值中的最大者,以此类推;记录路径走法,路由结束;若选出值小于阈值  $\eta_0$ ,则多选一跳,到步骤(4)。

(4) 若这些子节点中未包括  $i$  节点或  $i$  节点的  $\eta < \eta_0$ ,则这些子节点继续根据其相应的列找出其子节点,同样判断这些子节点是否存在节点  $i$ ,形成一个循环,若在  $n$  (节点数量) 次循环中找到了  $i$  节点,则路由结束,若循环次数超过了  $n$  依旧未找到  $i$  节点,则路由路径不存在,  $i$  和  $j$  不通,路由结束。

## 4 实验

利用 NS2 仿真平台进行仿真实验<sup>[7]</sup>,在仿真中,环境大小设为  $(1500 \times 1500) \text{ m}^2$ ,节点随机分布。我们设定网络有 15 个移动节点,传输范围 400 m。每个节点的移动速度从 10 m/s 到 50 m/s。每个节点随机选择一个方向作为其目标方向,采用统一的速度移动,一旦它到达了目的地,在那等待一定的时间,然后随机地选择另一个方向,开始移动。

每个节点在仿真开始时产生一个随机数,可能是偶数或是奇数。如果该节点产生的数是偶数,则该节点产生一个 Agent,而产生奇数的节点就不产生 Agent。这意味着在网络开始运行时,Agent 的数量大约是节点数的一半。

在实验中发现,节点 14 的 cache 上的数据如表 1 所示。

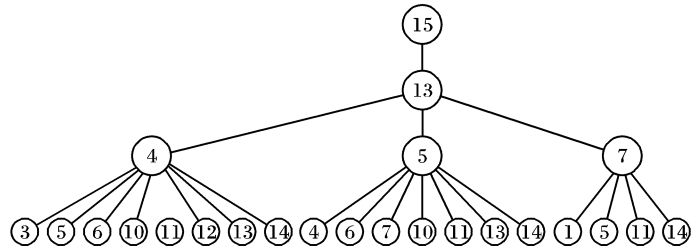


图3 报文路由算法树形图

从节点 14 传输数据到 15 节点,由图 3 的树形结构分析可知:存在的路径有四条:14→4→13→15,14→5→13→15,14→7→13→15。14 是起点,15 是终点不必考虑,节点 13 的  $\eta$  为最小,但它是所有路径必须经过的节点,则继续比较节点 4,5,7,由于则三个节点中节点 5 的  $\eta$  最小,节点 7 的  $\eta$  最大,所以最佳路径为 14→7→13→15,最差路径为 14→5→13→15。

仿真实验中采用了上述的 Agent 路由算法及基于电池量的路径算法,实验结果证明,该算法能快速发现最佳路由途径,使系统的电池的消耗趋于平衡。

表1 节点 14 cache 内的矩阵信息表

节点	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	Counter	历史	$\eta$ (%)
1	0	0	0	0	0	0	1	0	0	0	1	0	0	1	0	95	10	50
2	0	0	0	0	0	1	0	1	1	1	1	1	0	0	0	96	5	55
3	0	0	0	1	0	1	0	0	0	1	1	0	0	0	0	94	8	57
4	0	0	1	0	1	1	0	0	0	1	1	1	1	1	0	93		61
5	0	0	0	1	0	1	1	0	0	1	1	0	1	1	0	97		56
6	0	1	1	1	1	0	0	0	0	1	1	1	0	1	0	98		58
7	1	0	0	0	1	0	0	0	0	0	1	0	1	1	0	96		69
8	0	1	0	0	0	0	0	0	1	1	0	1	0	0	0	95		48
9	0	1	0	0	0	0	0	1	0	0	0	1	0	0	0	94		47
10	0	1	1	1	1	1	0	1	0	0	1	1	0	1	0	93		52
11	1	1	1	1	1	1	1	0	0	1	0	1	0	1	0	92		51
12	0	1	0	1	0	1	0	1	1	1	1	0	0	1	0	91		60
13	0	0	0	1	1	0	1	0	0	0	0	0	0	0	1	95		52
14	1	0	0	1	1	1	1	0	0	1	1	1	0	0	0	93		57
15	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	92		55

## 5 结语

本文介绍了一种在 Ad hoc 网络环境中基于电池量的路由算法。由于这种方法可以使用很少的 Agent 获得较多的全局信息,因此可以减少维持节点信息而产生的开销。实验结果表明这种路由算法有很高的效率和鲁棒性,可以使各节点电池量的消耗趋于平衡,为下一步 Ad hoc 网络管理的研究打下了基础。

### 参考文献:

- [1] IETF Working Group Charter. Mobile Ad hoc Network (MANET) [EB/OL]. <http://www.ietf.org/html.charters/manet-charter.html>, 2001-03-14.
- [2] CORSON S, MACKER J. Mobile Ad hoc Networking (MANET): Routing Protocol Performance Issues and Evaluation Considerations

[S]. RFC2501, 1999.

- [3] PALCHAUDHURI S. Power Mode Scheduling for Ad hoc Network Routing[D]. Masters Thesis, Computer Science, Rice University, 2002.
- [4] SINGH S, WOO M, RAGHAVENDRA C. Power-aware routing in mobile Ad hoc networks[A]. Proceedings of Mobicom '98[C]. 1998.
- [5] PHAM VA, KARMOUCH A. Mobile Software Agents: An Overview[J]. IEEE Communication Magazine, 1998, 36(7): 26-37.
- [6] GRAY R, KOTZ D, NOG S, et al. Mobile Agents for mobile computing[R]. Technical Report PCS-TR96-285, Department of Computer Science, Dartmouth College, Hanover, NH 03755, 1996.
- [7] FALL K, VARADHAN K. NS notes and documentation. The VINT Project, UC Berkeley, LBL, USC/ISI, and Xerox PARC[Z]. 1999.