软件学报 ISSN 1000-9825, CODEN RUXUEW
Journal of Software,2011,22(10):2335−2345 [doi: 10.3724/SP.J.1001.2011.03940]

E-mail: jos@iscas.ac.cn
http://www.jos.org.cn
Tel/Fax: +86-10-62562563

Ad Hoc 网络中改善拓扑控制性能的移动控制算法*

公维宾[1+], 常义林[2], 沈 中[2]

[1](长安大学 陕西省道路交通智能检测与装备工程研究中心,陕西 西安 710064)

[2](西安电子科技大学 信息科学研究所,陕西 西安 710071)

Movement Control Algorithms for Improving Topology Control Performance in Ad Hoc Networks

GONG Wei-Bin[1+], CHANG Yi-Lin[2], SHEN Zhong[2]

[1](Shaanxi Research Center for Road Traffic Intelligent Detection and Equipments Engineering, Chang'an University, Xi'an 710064, China)

[2](Information and Science Institute, Xidian University, Xi'an 710071, China)

+ Corresponding author: E-mail: wbgong@163.com, http://www.chd.edu.cn

Gong WB, Chang YL, Shen Z. Movement control algorithms for improving topology control performance in Ad Hoc networks. *Journal of Software*, 2011,22(10):2335−2345. http://www.jos.org.cn/1000-9825/3940.htm

Abstract: In Ad Hoc networks, topology control is designed to save energy and increase network capacity by enabling nodes to use proper transmission power, which is usually much smaller than the maximal transmission power. However, the randomicity of network deployment and node movement results in a non-uniform distribution of nodes. In a region where nodes are sparse, the distance between two nodes may be much longer than that in a dense region, so the transmission power is not able decrease and does not depend on which topology control algorithm is adopted. For the first time, this paper proposes movement control algorithms to improve performance of topology control by moving a subset of nodes to desiring positions. Based on the minimum spanning tree of the network topology graph, addition links are determined. Moreover, addition links are shortened and large communication range may be reduced by moving some nodes. Three movement algorithms, PMST-P, PMST-UV and LMST-LUV, are realized, and their performance is compared with each other with respect to maximum communication range, total moving distance and etc. using simulations.

Key words: topology control; movement control; deployment; transmission power; Ad Hoc network

摘 要: 在无线 Ad Hoc 网络中,拓扑控制算法能够使节点的传输功率小于最大传输功率,从而可以节省网络能量,提高网络容量.由于节点分布的随机性,在节点较为稀疏的区域,拓扑控制算法存在着局限性,因而提出了移动控制算法来改善拓扑控制算法的性能.在保证网络连通性的前提下,算法首先根据收集到的信息,通过构造网络最小生成树确定较长的通信链路,并移动网络中的部分节点使这些链路缩短,从而显著减小网络中较大的通信半径,提高了拓扑控制的性能.仿真实现了 PMST-P,PMST-UV 和 LMST-LUV 这 3 种移动控制算法,并对它们的性能进行了讨论和相

* 基金项目: 国家自然科学基金(61172069); 陕西省自然科学基础研究计划(2011JM8029); 中央高校基本科研业务费专项资金(CHD2009JC); 高等学校学科创新引智计划(B08038)

收稿时间: 2008-05-07; 修改时间: 2009-07-06; 定稿时间: 2010-08-27

互比较.

关键词: 拓扑控制;移动控制;部署;传输功率;Ad Hoc 网络
中图法分类号: TP393 文献标识码: A

在无线 Ad Hoc 网络中,节点能量受限、可能工作在战场等恶劣环境中、带宽较低等特点,使得网络的寿命、连通性、容错能力、负载均衡等热点问题受到广泛关注.拓扑控制(topology control,简称 TC)是解决这些问题的重要手段之一.拓扑控制[1−5]通过调节网络中各个节点的传输功率,构建一个优化的网络拓扑结构,达到节能[6−8]、增加网络寿命、增加网络容量[8]等目的.在节点较为密集的传感器网络中,还可以采用节点轮换休眠的方式进行拓扑控制[9−11].

由于无线 Ad Hoc 网络具有移动性特点,并且节点和链路的可靠性也较低,决定了其拓扑结构是动态的、随机的,因而使得以下几个问题备受关注:(1) 网络分区.即使所有节点都以最大传输功率工作,仍然不能保证网络的连通性;(2) 过大的传输功率.网络节点在部署区域内分布不均匀,在节点较为稀疏的区域,节点之间距离较远,无论采用何种拓扑控制算法,这些节点的传输功率都不会得到有效降低.而过大的传输功率不仅会使节点过快地耗尽能量并停止工作,影响网络寿命,还会严重干扰邻居节点通信,降低网络容量;(3) 抗毁性.由于网络中链路和节点的失效率都比较高,需要对网络进行容错性设计,使其具有抗毁性或者强连通性.

如前所述,传统的拓扑控制算法采用调节功率的方法进行拓扑控制,但都假定其最大功率拓扑没有出现网络分区或者已经具有了强连通性,并且不能避免出现过大的传输功率,因而无法解决网络中出现的上述问题.通过移动拓扑控制,即控制某些节点使其移动到指定的位置,可以有效解决上述问题[12−15].移动拓扑控制一般不单独使用,而是作为传统拓扑控制算法的补充和前提.在对网络进行移动拓扑控制后,再施以传统的拓扑控制算法,二者联合工作时,才能够有效地改善网络的性能.

移动拓扑控制研究的内容,就是通过有目的地重新部署部分受控移动节点,在保持网络连通性的基础上解决网络部署过程中出现的过大传输功率、抗毁性、覆盖率等问题,并且令受控节点移动的距离之和最小.该类问题是 NP 难解问题,只能寻找一些性能较好的启发式算法[16].移动拓扑控制算法最初引入到移动传感器网络中主要用于优化网络的覆盖率[17−19]及覆盖均匀性问题.在移动 Ad Hoc 网络中,则主要用于解决网络的抗毁性,可以构建两连通甚至多连通的网络.如 Basu 等人[12]的块移动算法(block move algorithm)和 Lin[13]的消除割点汇聚算法(cutvertex-elimination rendezvous algorithm,简称 CERA)等,都被用来消除网络中的割点,构造两连通网络.Kashyap 等人[14]则研究了在网络中引入额外的中继节点,通过移动中继节点构造出多连通的网络.如何解决网络分区和节点传输功率过大问题,则是该领域需要进一步研究的重要课题.本文提出了基于 MST(minimum spanning tree)的移动拓扑控制算法,能够均衡节点传输功率,解决了传输功率过大的问题.经过简单的演变,这些算法还可以将网络由不连通变为连通,以解决网络的连通性问题[20].

1 系统模型

假设 Ad Hoc 网络在部署时,所有节点都位于一个平面区域内,区域内部平坦,节点之间没有障碍物阻隔.网络中节点集合用 V 表示,每个节点分配唯一的标识号(如 IP 地址或 MAC 地址等).为方便起见,可令 $id(v_i)=i$.节点具有控制传输功率的能力.由于无线信号的传输模型比较复杂,为了研究方便,以 $r(p)$表示节点传输功率为 p 时信号的传播距离.节点的最大传输功率用 $p_{\max}$ 表示,对应的最大通信半径为 $r(p_{\max})$,简写为 $r_{\max}$.如果节点传输功率大于 $p_0(p_0<p_{\max})$,则认为传输功率过大,需要降低传输功率.p_0 称为临界功率,对应的通信距离为 $r(p_0)$,简写为 r_0,称为临界通信半径.

两个节点 u 和 v,如果它们相互在通信距离之内,则它们之间存在一条边,用 $e(u,v)$表示,权值为两个节点之间的欧氏距离 $d(u,v)$.当网络中所有节点的传输功率为 p 时,网络所形成的拓扑图可以表示为 $G(p)=(V,E(p))$,其中,$E(p)=\{e(u,v)|d(u,v)<r(p),u,v\in V\}$.很显然,最大功率拓扑图和临界功率拓扑图的关系满足 $G(p_{\max})\supseteq G(p_0)$,即后者是前者的生成子图.即使 $G(p_{\max})$是连通的,$G(p_0)$也可能不连通,它由几个不连通的子图组成,可以表示为

$$G(p_0)=\bigcup_{k=1}^{n}G_k(p_0).$$

$G_k(p_0)(k=1,2,\ldots,n)$为各个连通分量,本文称其为分割区域(partition),简称分区.如何移动部分节点,使这些不连通的分区重新构成一个连通图,即为本文所要解决的问题.在移动控制完成后,再进行拓扑控制,显然能够降低过大的传输功率.为了完成移动控制功能,节点必须具有受控移动能力和定位能力,并且能够通过路径规划算法从当前位置移动到任何指定的目的位置.

2 移动控制算法

设计移动控制算法必须遵循下列几条准则:(1) 连通性.算法执行后的$G(p_0)$要保持$G(p_{\max})$的连通性,在有些应用场合,可能还需要考虑算法执行过程中网络连通性的保持问题;(2) 移动距离.为了节省节点移动所消耗的能量,加快算法收敛速度,节点移动的距离要小;(3) 网络拓扑结构.算法对网络拓扑结构影响要小,尽量保持原来的拓扑结构.

寻找最优的移动控制算法比较困难,本文提出了启发式算法,遵循以下几个步骤:(1) 收集信息.每个节点通过与邻居节点交换信息,得到邻居节点的标识、位置、拓扑等信息;(2) 确定增补链路.确定传输功率大于临界功率的节点和需要增补的链路;(3) 确定移动节点.寻找距离增补链路最近的非割点或者叶子分区作为移动节点;(4) 节点移动.根据算法运行中网络连通性和移动距离的要求,选择合适的移动方式并令节点移动.上述几个步骤迭代运行,直到网络拓扑图 $G(p_0)$连通为止.

2.1 收集信息

定义 1. 当网络中所有节点的传输功率都为 p 时,对任意节点 $u\in V$,其邻居节点集合定义为 $V_u(p)=\{v|v\in V, v\neq u$ 且 $d(u,v)<r(p)\}$,其邻居节点图定义为 $NG_u(p)=(V_u(p),E_u(p))$,其中,$E_u(p)=\{e(w,v)|w\in V_u(p),v\in V_u(p),d(w,v)<r(p)\}$.

对网络中任意节点 u,收集信息分为 3 个阶段:(1) 分别以功率 $p_{\max}$ 和 p_0 广播 hello 包,hello 包的内容至少应包含位置坐标和节点标识,同时接收邻居节点发送的广播包,从而可以构造出相应传输功率下的邻居节点集合 $V_u(p_{\max})$和 $V_u(p_0)$;(2) 把邻居节点集合 $V_u(p_{\max})$和 $V_u(p_0)$的信息分别组成相应的 hello 包,以功率 $p_{\max}$ 和 p_0 广播,从而构造出邻居节点图 $NG_u(p_{\max})$和 $NG_u(p_0)$;(3) 将前两步得到的信息汇集起来,构造出网络拓扑图 $G(p_{\max})$和 $G(p_0)$.

如果无线信道的传播模型是已知的,可以直接收集各个节点的位置坐标,再通过计算确定每个节点的邻居节点,并进而构造整个网络的拓扑图.但在实际环境中,信道的传播模型很难确定,一些定位方法的精度也存在局限,因而采用广播 hello 包的方式更为可靠.

2.2 确定增补链路

通过收集信息,已经获得各个节点的位置坐标和网络拓扑图 $G(p_{\max})$和 $G(p_0)$.为了让不连通的 $G(p_0)$变成连通,必须在图中增加一些连接不同分区的边.很显然,这些边来自最大功率网络拓扑图 $G(p_{\max})$.为了保证节点移动时距离较小,这些边的权值之和也必须较小.下面给出与之相关的定义.

定义 2. 前文已经提到,$G(p_0)$可能由几个不连通的分区组成,可以表示为

$$G(p_0)=\bigcup_{k=1}^{n}G_k(p_0),G_k(p_0),\ k=1,2,\ldots,n.$$

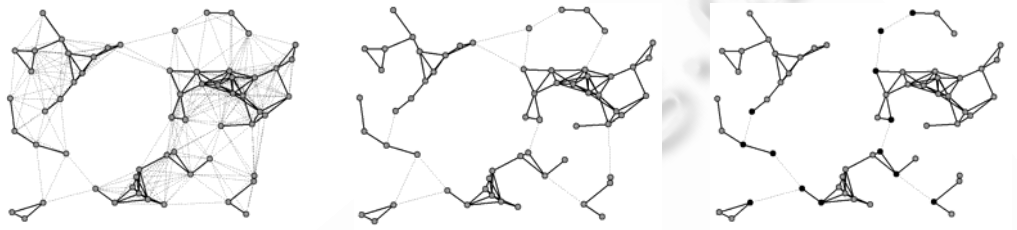
对于任意两个分区 $G_i(p_0)$和 $G_j(p_0)$,如果它们之间的边集 $E_{ij}(p_0)=\{e(u,v)|u\in G_i(p_0),v\in G_j(p_0),d(u,v)<r_{\max}\}$不为空,即这两个分区之间在 $G(p_{\max})$中是连通的,则称这两个分区之间有一条边,从边集 $E_{ij}(p_0)$选择权值最小的边来表示,记为 $e_{i,j}(p_0)$,权值为 $d_{ij}(p_0)$.

定义 3. 令 $PV(p_0)=\{G_1(p_0),\ldots,G_k(p_0),\ldots,G_N(p_0)\}$为 $G(p_0)$中所有分区的集合,$PE(p_0)$为分区之间边的集合,则分区图可以定义为 $PG(p_0)=(PV(p_0),PE(p_0))$.在分区图的基础上,引入分区最小生成树的概念.分区最小生成树定义为 $PT(p_0)=(PV(p_0),TE(p_0))$.其中,$TE(p_0)\subseteq PE(p_0)$,为最小生成树中的边集.与 $TE(p_0)$中的边相关联的节点分别在

两个分区内,称为网关节点.

为了找到增补边,首先需要根据 $G(p_{max})$和 $G(p_0)$构造出网络的分区图和分区最小生成树.在分区图 $PG(p_0)$的所有连通子图中,分区最小生成树 $PT(p_0)$的边的数目最小,权值之和也最小,这对减小移动节点的数目和移动距离非常有利.很显然,分区最小生成树 $PT(p_0)$中的边即可作为增补边.$PT(p_0)$的生成算法有很多,本文采用的是 Prim 算法.这种确定增补边的算法称为 PMST(partition minimum spanning tree)增补链路算法.

在如图 1 所示的网络中,共有 60 个节点.图 1(a)表示的是网络拓扑图,其中,实线部分表示的是临界功率拓扑图,虚线表示的是最大功率拓扑图.可以看出,临界功率拓扑图是不连通的,共有 8 个分区.图 1(b)是根据图 1(a)导出的分区拓扑图,虚线表示分区之间的边.与最大功率拓扑图相比,分区之间的链接已显著减少,但仍存在冗余.图 1(c)为图 1(b)的分区最小生成树,实心圆点表示网关节点,虚线表示增补链路.$G(p_0)$的几个非连通分区,通过分区最小生成树的边构成了一个连通图,这些边就是算法所要寻找的增补边.



(a) Full transmission power topology (a) 最大功率拓扑 (b) Partitions (b) 分区图 (c) Spanning tree of partitions (c) 分区最小生成树

Fig.1 Determination of addition links

图 1 增补链路的确定

2.3 确定移动节点

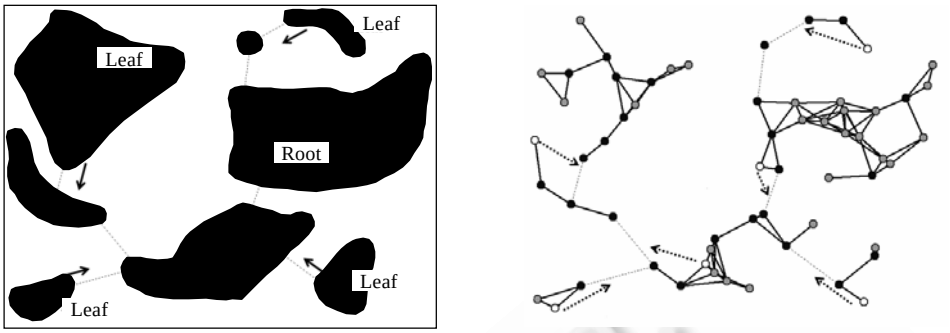
增补链路的传输功率都是大于临界传输功率的,必须缩短增补链路的长度才能使相应的节点降低传输功率,而缩短链路需要移动网络中的部分节点.最简单的确定移动节点的方法是把与增补链路相连的分区作为一个整体进行移动,称为分区(partition)移动算法.在移动过程中和移动过程结束后,分区内节点没有相对位移,分区内节点的连通性未发生改变.假设在分区最小生成树中,分区 $G_k(p_0)$分别与分区 $G_m(p_0)$和 $G_n(p_0)$相连,如果 $G_k(p_0)$向 $G_m(p_0)$移动一定的距离,则两个分区就可以合并成一个分区.但有可能产生新的问题,即 $G_k(p_0)$可能与 $G_n(p_0)$变成不连通的,从而破坏了整个网络的连通性.为了避免发生这种情况,只有生成树中度数为 1 的分区,即叶子分区可以移动,移动方向为其父分区.为了尽量减小节点的移动数目和移动距离,应该避免移动节点较多的分区,因此指定节点最多的分区为根分区,不能参与移动.

在分区移动算法中,节点之间的连接关系未发生变化,对上层路由协议的影响较小.但是,为了缩短一条链路,整个分区的节点都参与了移动,因而总的移动距离较大.此外,由于每次只能移动叶子分区,在分区较多的情况下,算法需要较多的迭代过程.为了解决上述问题,对于每一条增补链路,只选择一个节点移动到增补链路处,把一条增补链路分为两条链路,从而达到缩短链路的目的.被选择节点移动后,不能改变原网络拓扑的连通性,分区中的非割点可以满足这个条件.为了减少节点移动距离,在存在多个非割点的情况下,选择移动距离最小的非割节点作为移动节点.在图论中,求割点最常用的算法是深度优先遍历算法(depth first search,简称 DFS).这种确定移动节点的算法称为非割节点(uncut-vertex,简称 UV)算法,具体步骤如下:

- 根据增补链路、网关节点的位置和临界半径 r_0 确定移动节点需要到达的位置;
- 在与增补链路相连的两个分区中,寻找增补链路位置距离最短的非割节点作为移动节点;
- 为了不影响分区的连通性,一个分区一次只能移动一个非割节点.

图 2 显示了在图 1 所示的网络中,利用不同算法确定了移动节点,箭头表示分区或者节点移动的方向.在图 2(a)中,黑点表示网关节点,灰点表示普通节点,每个分区用一个灰色区域突出地显示出来;在图 2(b)中,黑点表示

网络中的割点,灰点表示非割节点,白点是从非割节点中选择出来的移动节点.由图 2(a)可以看出,算法确定了 4 个叶子分区,移动方向沿着增补链路指向其父分区方向.确定的根分区不能参与移动.经过第 1 次迭代过程,分区移动算法缩短了 4 条增补链路,而非割节点移动算法则缩短了 6 条增补链路.



(a) Movement of partitions
(a) 分区移动示意图

(b) Movement of uncut-vertex nodes
(b) 非割节点移动示意图

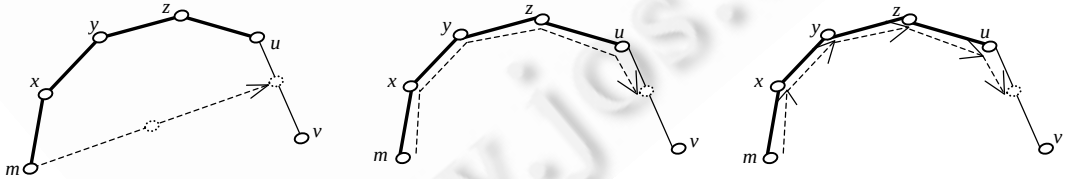
Fig.2 Determination of moving nodes candidates

图 2 移动节点的确定

2.4 节点移动

在非割节点算法中,移动节点可以采用直线移动(direct movement,简称 DM)方式到达目标位置(如图 3(a)所示中的移动节点 m).根据对算法的分析,在算法执行后,网络拓扑的连通性没有遭受破坏;但在算法执行过程中,则有可能出现该节点与其他节点不连通的情况.为了解决这个问题,可以采用回溯移动(traced movement,简称 TM)方式(如图 3(b)所示),令移动节点 m 沿着其与目标位置之间的最短路径移动,最短路径采用 Dijkstra 算法获得.回溯法可能导致一个节点的移动距离过大,从能量消耗的均匀性考虑,对移动节点不够公平,因而可以采用级联移动(cascaded movement,简称 CM)方式来加以改进(如图 3(c)所示).首先是节点 m 移动到下一跳节点 x 位置并停止,然后,节点 x 移动到 y,依此类推,直到节点 u 移动到目标位置.在级联方式和回溯方式中,移动节点在移动过程中始终与网络保持连通状态,两种移动方式的移动距离是相同的,但级联方式更好地解决了节点移动的公平性问题.同时不难看出,直线移动具有最小的移动距离.

在算法执行的 4 个过程中,确定增补链路和确定移动节点较为关键.因此,本文的移动控制算法以这两个步骤的算法进行联合命名.按照这种方法,我们把采用分区最小生成树算法和分区移动算法的移动控制算法称为 PMST-P,采用分区最小生成树算法和非割节点移动算法的移动控制算法称为 PMST-UV 算法,而下一节将要讨论的算法称为 LMST-LUV 算法.



(a) Direct movement style
(a) 直线移动方式

(b) Traced movement style
(b) 回溯移动方式

(c) Cascaded movement style
(c) 回溯级联移动方式

Fig.3 Movement style of nodes

图 3 节点的移动方式

3 移动控制算法的分布式实现

Ad Hoc 网络是一种自组织网络,可能部署在恶劣环境中,并且可能并不存在一个集中式的服务器来收集信

息,无法构造出整个网络的拓扑图.因而提出了把基于 MST 的移动控制算法应用到分布式环境中的 LMST-LUV (localized minimum spanning tree-localized uncut-vertex)移动控制算法,并对其正确性进行了说明.

3.1 LMST-LUV分布式移动控制算法

定义 4. 对任意节点 $u\in V$,根据其邻居节点图 $NG_u(p_{\max})$,可以求得邻居节点最小生成树 $NT_u(p_{\max})=(V_u(p_{\max}), TE_u(p_{\max}))$,$TE_u(p_{\max})$为最小生成树的边.在 $NT_u(p_{\max})$中,所有与节点 u 相邻的节点称为逻辑邻居节点集合,定义为 $LV_u(p_{\max})=\{v|v\in V_u(p_{\max}),e(u,v)\in TE_u(p_{\max})\}$;节点 u 与逻辑邻居节点集合及其它们之间的链路所组成的图称为逻辑邻居节点图,表示为 $LG_u(p_{\max})=(LV_u(p_{\max}),LE_u(p_{\max}))$,其中,$LE_u(p_{\max})=\{e(u,v)|v\in LV_u(p_{\max}),e(u,v)\in LE_v(p_{\max})\}$.

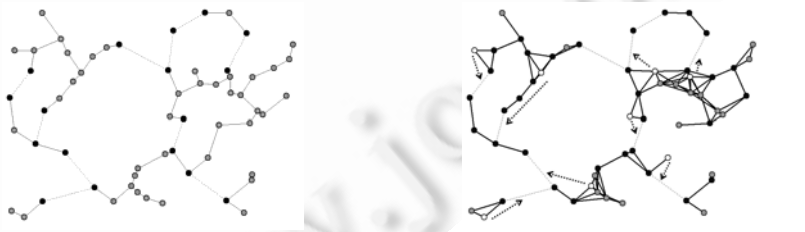
在 $LG_u(p_{\max})$中,如果存在大于临界半径 r_0 的边,则认为该边为增补边.检测到自己的逻辑邻居节点图中有增补链路的节点 u,首先向该链路的另一个关联节点 v 发起请求,查询该条链路是否存在于 $LG_v(p_{\max})$中.如果不存在,则说明该链路为单向链路,应该从 $LG_u(p_{\max})$中删除.如果 u 的逻辑邻居节点图中有增补链路,则称节点 u 为网关节点.这种确定增补链路的算法称为 LMST(local minimum spanning tree)增补链路算法.

在确定移动节点时,由于节点只有一跳邻居节点信息,无法得到网络的分区信息,因而无法采用分区移动算法,也无法根据全局信息确定哪个节点是非割节点,因而也不能直接采用非割节点移动算法.我们可以根据邻居节点图 $NG_u(p_0)$来确定非割节点,这就是本地非割节点(localized uncut-vertex,简称 LUV)算法.在本地非割节点算法中,网关节点必须作为割点来对待,不能移动,否则会破坏网络原有的连通性.确定非割节点的图 $NG_u(p_0)$中不能包含移动节点,也就是说,不能有相邻的两个节点同时参与移动,否则也会破坏网络的连通性.

算法实现时,网关节点以临界功率 p_0 发送查询非割点的广播包,广播包的 TTL 字段设置为 MAX_HOP_COUNT;接收到查询包的节点若是割点,则继续以临界功率广播;若是非割点,则首先按照反向路径对根节点进行响应,然后继续进行广播;网关节点根据收集到的非割节点信息,确定最小移动距离的非割节点,并把割点标识号和距离发送到另一个网关节点.网关节点收到邻居网关节点的通知信息后,把其中的距离与本网关节点查询得到的距离相比,如果本网关节点的距离较小,则向本分区中移动距离最小的非割节点发出移动命令,非割节点接收到移动命令后,向目标位置移动,并把自己的移动状态以临界功率进行广播,从而通知到其他邻居节点.

上述移动控制算法简称 LMST-LUV 算法,由于是分布式算法,节点只能依赖本地信息,因而在性能上与 PMST-UV 算法存在一定的差距.特别是在确定移动节点时,网关节点可能在 MAX_HOP_COUNT 跳之内不能找到非割节点,从而使有些链路得不到缩短.

图 4 说明了算法的执行过程.图 4(a)显示了确定的增补链路,用虚线加以表示.由于信息的本地化,LMST 增补链路算法所构造出的并不是一个全局的最小生成树,而是构造出了一个存在许多环路的拓扑结构,所确定的增补链路数目也多于 PMST 算法,但只是存在冗余,并没有遗漏的情况.图 4(b)中,黑点表示的是采用本地割点算法确定的割点,灰点是非割节点,白点是从非割节点中挑选出来的移动节点,箭头是节点移动方向.



(a) Dtermination of addition links (a) 增补链路确定

(b) Determination of moving nodes candidates (b) 移动节点确定

Fig.4 LMST algorithm

图 4 LMST 算法

3.2 正确性证明

本节首先给出了图论中常用的最小生成树 MST 的性质定理及其逆定理,然后证明了 LMST-LUV 算法的正

确性.

MST 性质定理. 设 $G=(V,E)$是一个连通网络,U 是顶点集 V 的一个真子集.若(u,v)是 G 中所有的一个端点在 $U(u\in U)$里、另一个端点不在 $U(v\in V-U)$里的边中具有最小权值的一条边,则一定存在 G 的一棵最小生成树包含此边(u,v).

MST 性质逆定理. 设 $G=(V,E)$是一个连通网络,U 是顶点集 V 的一个真子集.若 $u\in U,v\in V-U$,且(u,v)是最小生成树中的一条边,则(u,v)是 G 中所有的一个端点在 $U(u\in U)$里、另一个端点不在 U(即 $v\in V-U$)里的边中权值最小的一条边.

定理 1. 已知 $G(p_{\max})$是最大功率网络拓扑图,V 是顶点集,$T_{\mathrm{LMST}}(p_{\max})=(V,TE_{\mathrm{LMST}}(p_{\max}))$为其 LMST 生成树,其中,$TE_{\mathrm{LMST}}(p_{\max})=\bigcup_{u\in V}LE_u(p_{\max})$,$T_{\mathrm{MST}}(p_{\max})=(V,TE_{\mathrm{MST}}(p_{\max}))$为其最小生成树,那么 $T_{\mathrm{MST}}(p_{\max})$是 $T_{\mathrm{LMST}}(p_{\max})$的子图.

证明:由已知条件可知,$T_{\mathrm{MST}}(p_{\max})$和 $T_{\mathrm{LMST}}(p_{\max})$的顶点集是相等的,要证明 $T_{\mathrm{MST}}(p_{\max})$是 $T_{\mathrm{LMST}}(p_{\max})$的子图,只需证明 $TE_{\mathrm{LMST}}(p_{\max})\supseteq TE_{\mathrm{MST}}(p_{\max})$即可.对于任意 $e\in TE_{\mathrm{MST}}(p_{\max})$,设 $e=(u,v)$,根据树的定义,删除该边后,$TE_{\mathrm{MST}}(p_{\max})$成为不连通的两个子图.不防设这两个子图的顶点集分别为 U 和 $V-U$,且有 $u\in U,v\in V-U$,根据 MST 性质逆定理,边 e 是 $G(p_{\max})$中所有的一个端点在 U 里、另一个端点不在 U 里的边中具有最小权值的边.对于节点 u,设其邻居节点图为 $NG_u(p)$,邻居节点最小生成树为 $NT_u(p)$.显然,如果删除 e 边后,$NT_u(p)$也成为不连通的两个子图.不妨设这两个子图的顶点集合分别为 U_u 和 V_u,不难得到 $U_u\subseteq U$ 和 $V_u\subseteq V-U$,且有 $u\in U_u$ 和 $v\in V_u$,e 是 U_u 和 V_u 之间权值最小的边.根据 MST 性质,e 边必定包含在邻居节点最小生成树 $NT_u(p)$中.根据逻辑邻居节点图 $LG_u(p)$的定义,该边也必定包含在 $LG_u(p)$中.因此,对于任意 $e\in TE_{\mathrm{MST}}(p_{\max})$,有

$$e\in TE_{\mathrm{LMST}}(p_{\max}),TE_{\mathrm{LMST}}(p_{\max})\supseteq TE_{\mathrm{MST}}(p_{\max}).$$

即 $T_{\mathrm{MST}}(p_{\max})$是 $T_{\mathrm{LMST}}(p_{\max})$的子图. □

定理 2. 已知 $G(p_0)$是临界功率网络拓扑图,$C_{\mathrm{MST}}(p_0)$为割点集,$C_{\mathrm{LMST}}(p_0)$为 LMST 算法割点集,则有

$$C_{\mathrm{LMST}}(p_0)\supseteq C_{\mathrm{MST}}(p_0).$$

证明:对于任意 $u\in C_{\mathrm{MST}}(p_0)$,如果删除该顶点,则由割点定义可知,该连通分区成为两个不连通子图,邻居节点集 $V_u(p_0)$也分成不连通的两部分,所以节点 u 也是邻居节点图 $NG_u(p_0)$中的割点,因而有 $u\in C_{\mathrm{LMST}}(p_0)$.可得,

$$C_{\mathrm{LMST}}(p_0)\supseteq C_{\mathrm{MST}}(p_0).$$ □

定理 3. 如果 $G(p_{\max})$是连通的,且通过 LMST-LUV 算法所得到的增补链路都缩短到小于临界通信半径,则最终形成的拓扑图 $G_{\text{LMST-LUV}}(p_0)$也是连通的.

证明:由定理 1 可知,$T_{\mathrm{MST}}(p_{\max})$是 $T_{\mathrm{LMST}}(p_{\max})$的子图,而 $T_{\mathrm{MST}}(p_{\max})$保持了 $G(p_{\max})$的连通性,因此 $T_{\mathrm{LMST}}(p_{\max})$也保持了 $G(p_{\max})$的连通性.增补链路集 $RE_{\mathrm{LMST}}(p_{\max})$由 $T_{\mathrm{LMST}}(p_{\max})$中长度大于临界通信半径的链路构成,如果通过 LMST-LUV 算法使所有增补链路的长度都小于临界通信半径,则此时 $T_{\mathrm{LMST}}(p_0)$保持了 $T_{\mathrm{LMST}}(p_{\max})$的连通性.在进行移动控制时,选择本地非割节点进行移动,由定理 2 可知,本地非割节点的移动不会影响网络的连通性.由以上结论可知,$T_{\mathrm{LMST}}(p_0)$保持了 $G(p_{\max})$的连通性.由于 $T_{\mathrm{LMST}}(p_0)$是 $G_{\text{LMST-LUV}}(p_0)$的生成子图,显然,$G_{\text{LMST-LUV}}(p_0)$保持了 $T_{\mathrm{LMST}}(p_0)$的连通性,即如果移动控制前 $G(p_{\max})$是连通的,则移动控制后形成的拓扑图 $G_{\text{LMST-LUV}}(p_0)$也是连通的. □

4 仿真结果

4.1 仿真方法

最大通信半径是评价本文移动控制算法的最重要的性能参数,算法执行前后,最大通信半径减小越多,则算法效果越显著.小的节点平均移动距离不仅反映了节点移动所消耗的能量较少,对网络拓扑结构影响较小,还表明算法的收敛速度较快.需要移动的节点比率,用实际移动节点数目与总节点数的比值来表示,是算法对拓扑结构影响程度的一个重要量度,移动节点数目越少,对拓扑结构的影响也越小.因此,首先对上述几项性能指标进

行了仿真研究.此外,还通过对算法执行时间和迭代次数的仿真研究了算法的收敛性.通过对算法的分析容易看出,在最大通信半径 r_{max} 一定的情况下,临界通信半径 r_0 对算法性能具有显著的影响,本文也对此作了仿真研究.节点移动距离、最大通信半径等指标与所采用的拓扑控制算法有关,为研究方便,本文采用的拓扑控制算法为 MST 算法[16].

为了研究各种移动控制算法的性能,利用 VC6.0 开发了仿真测试平台,实现了 PMST-P,PMST-UV 和 LMST-LUV 等算法.仿真区域为矩形,大小为 1600m×1200m,内部平坦,没有障碍物,移动节点可在其中自由移动.节点初始部署时满足均匀分布,且满足连通性.为了减小随机误差,每个仿真结果为 50 次仿真实验的平均值.节点具有全向天线,每个节点的通信能力都相同,最大通信半径为 360m.在 3 种节点移动方式中,直线移动方式的移动距离最短,其他两种方式的移动距离相等,但回溯级联方式需要移动的节点数要大些.这些结论都比较容易得到,这里不再进行针对性的仿真研究.在本文所有仿真实验中,节点都选择直线移动方式,且移动时其速度为 2m/s.

4.2 节点数目与算法性能的关系

仿真中保持临界通信半径为 180m 不变,节点数目从 10~100 依次增加,每次改变 10 个节点.算法执行过程中,网络分区数目决定了增补链路数的多少,而增补链路数直接影响了算法性能的各项指标.增补链路数、网络分区数与节点数目的关系如图 5 所示,其中,*PART* 表示网络分区的数目,*PMST* 表示利用分区最小生成树算法得到的增补链路数目,*LMST* 表示利用本地分区最小生成树算法得到的增补链路数目.PMST 增补链路算法得到的增补链路就是分区树中的边,因此数目总比分区个数小 1.与 PMST 算法相比,LMST 增补链路算法由于只能依靠邻居节点信息,所检测到的增补链路存在冗余,总是多于前者,因此,LMST-LUV 算法的性能总体上不如 PMST-UV 算法.

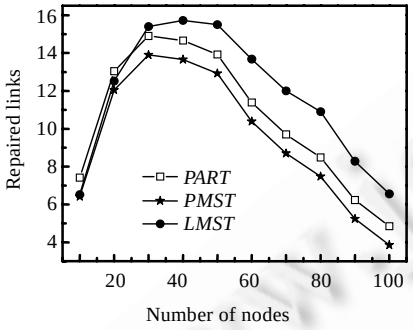
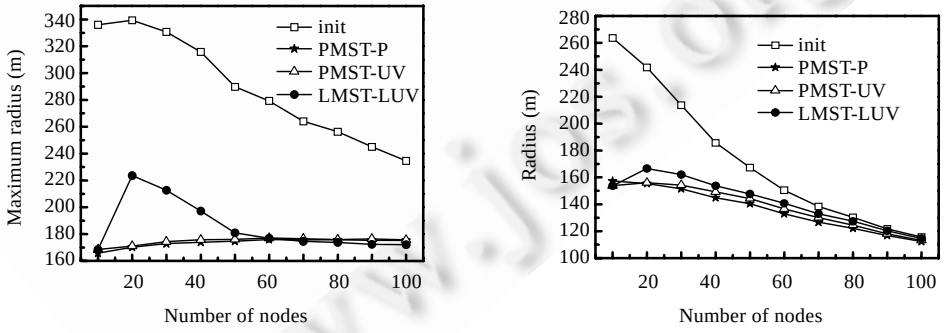


Fig.5 Addition links

图 5 增补链路

执行移动控制算法的目的是降低网络中较大的通信半径.从图 6(a)中可以看出,3 种算法都能有效减小网络中节点的最大通信半径.由于 PMST-P 算法和 PMST-UV 算法可以利用全网信息进行移动控制决策,因此总是能够把最大通信半径降低到临界半径之下.而 LMST-LUV 算法由于可能存在得不到缩短的链路,这是节点数为 20~40 时算法的最大通信半径要大于临界半径,性能不如 PMST-P 算法和 PMST-UV 算法的原因.从图 6(b)可以看出,3 种算法的通信半径基本相同,但都比初始部署时有所减小.特别是节点数目较少时,初始通信半径较大,需要缩短的链路较多;在算法执行后,通信半径显著减小.



(a) The maximum communication radius

(a) 最大通信半径

(b) The mean communication radius

(b) 平均通信半径

Fig.6 Communication radius

图 6 节点通信半径

从图 7 和图 8 可以看出,PMST-P 算法由于采用了分区移动算法,缩短一条链路需要某个分区的全部节点参与移动,因而平均移动距离较大,参与移动的节点数较多.在 PMST-UV 和 LMST-LUV 算法中,一条链路只需一个非割节点参与移动,因而移动节点数目较少,平均移动距离也较小.

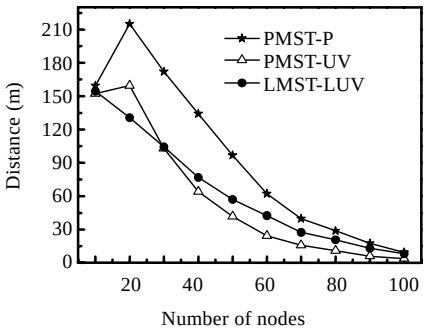


Fig.7 Moving distance

图 7 节点移动距离

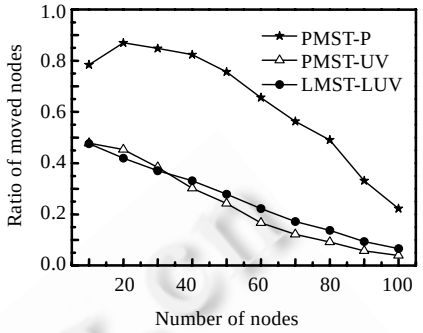
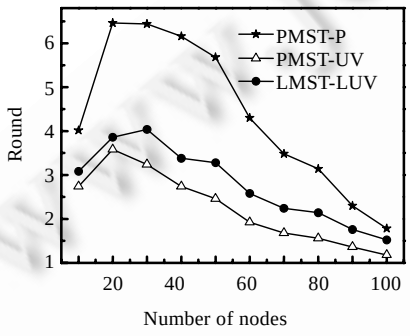


Fig.8 Ratio of moved nodes

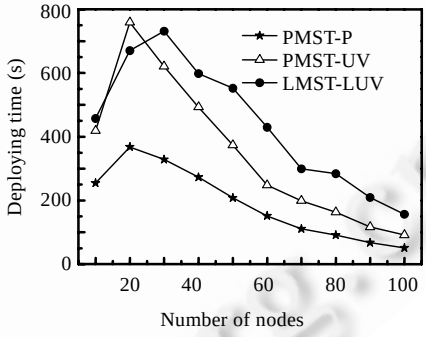
图 8 移动节点数

图 9 显示了算法的收敛性.PMST-P 算法由于每次只能增补与叶子分区相关联的链路,算法迭代次数较多;其他两种算法对链路的增补是同步进行的,因而算法迭代次数较少.PMST-UV 算法的增补链路不存在冗余,因而收敛速度比 LMST-LUV 算法要快,如图 9(a)所示.虽然 PMST-P 算法的迭代次数较多,平均移动距离也较大,但因为分区移动是分区内各个节点的同步移动,故其每个迭代过程的移动距离较短,所用时间也较短,其算法执行时间反而最小,如图 9(b)所示.



(a) Iteration times

(a) 迭代次数



(b) Deploying time

(b) 执行时间

Fig.9 Convergence of movement algorithms

图 9 算法收敛性

4.3 临界半径与算法性能的关系

在本节的仿真实验中,保持节点数目为 60,最大通信半径为 360m 不变,临界半径 r_0 从 150m~330m 依次增加,每次改变 30m.从图 10 中可以看出,网络分区数越少,经过算法确定的增补链路越少,因而会显著地减少移动节点个数和节点移动距离(如图 12 所示),加快算法的收敛(如图 13 所示),但网络的最大通信半径不会得到很大的改变(如图 11 所示).

对于不同的应用场合和不同类型的节点,选择合适的临界半径是非常重要的.临界半径较小时,节点平均移动距离较大,移动本身需要消耗大量的能量,对原来网络拓扑结构的改变也较大.因此在大多数情况下,临界半径过小是不合适的;而临界半径过大又不能达到移动控制的目标.因此在大多数应用场景下,增补链路的数目不应太多,应小于非增补链路的数目;临界通信半径应设置得比平均通信半径要大一些,例如大 0.5 倍左右,才能具有良好的性能.如在上述实验仿真条件下得到网络平均半径为 150.4m,那么临界通信半径可设为 225m.从结果

可以看出,此时算法具有较好的性能.

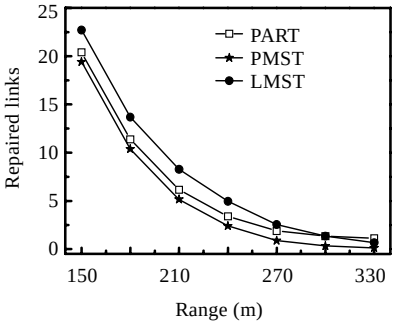


Fig.10 Addition links vs. critical range

图 10 增补链路数与临界半径的关系

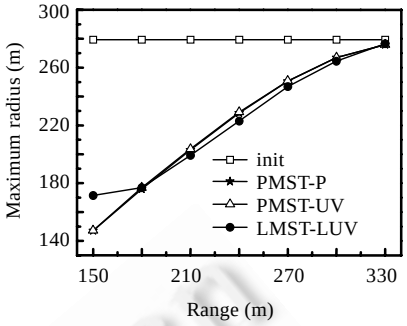


Fig.11 Maximum radius vs. critical range

图 11 最大通信半径与临界半径的关系

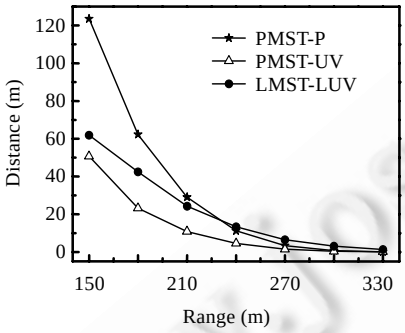


Fig.12 Moving distance and vs. critical range

图 12 节点移动距离与临界半径的关系

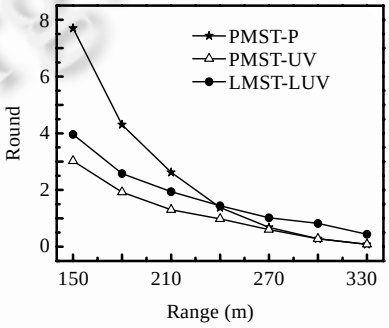


Fig.13 Iteration times vs. critical range

图 13 迭代次数与临界半径的关系

5 结 论

在无线 Ad Hoc 网络中,拓扑控制可以节省网络能量,提高网络容量.由于网络部署存在随机性,拓扑控制算法存在着局限性,因此提出了通过移动节点的方法来改善拓扑控制的性能.在本文提出的 PMST-P,PMST-UV 和 LMST-LUV 这 3 种移动控制算法中,前两种需要全局信息,适合集中式运行.而最后一种只需本地信息,是分布式运行的.算法的性能不仅与节点的疏密程度有关,还受到临界通信半径的影响.在大多数情况下,增补链路的数目不应太多;当临界通信半径比平均通信半径大 0.5 倍时,算法具有较好的性能.

References:

[1] Li N, Hou JC, Sha L. Design and analysis of an MST-based topology control algorithm. In: Proc. of the IEEE INFOCOM. San Francisco: IEEE Press, 2003. 1702−1712. [doi: 10.1109/INFCOM.2003.1209193]

[2] Li L, Halpern JY, Bahl P, Wang YM, Wattenhofer R. A cone-based distributed topology-control algorithm for wireless multi-hop networks. IEEE/ACM Trans. on Networking, 2005,13(1):147−159. [doi: 10.1109/TNET.2004.842229]

[3] Wattenhofer R, Zollinger A. XTC: A practical topology control algorithm for ad-hoc networks. In: Proc. of the 4th Int'l Workshop on Algorithms for Wireless, Mobile, Ad Hoc and Sensor Networks (WMAN). New Mexico: IEEE Press, 2004. 216−223. [doi: 10.1109/IPDPS.2004.1303248]

[4] Jones CE, Sivalingam KM, Agrawal P, Chen JC. A survey of energy efficient network protocols for wireless networks. Wireless Networks, 2001,7(4):343−358. [doi: 10.1023/A:1016627727877]

[5] Li L, Halpern JY. A minimum-energy path-preserving topology-control algorithm. IEEE Trans. on Wireless Communications, 2004, 3(3):910−921. [doi: 10.1109/TWC.2004.826324]

[6] Li N, Hou JC. Localized topology control algorithms for heterogeneous wireless networks. IEEE/ACM Trans. on Networking, 2005, 13(6):1313−1324. [doi: 10.1109/TNET.2005.860095]

[7] Wattenhofer R, Li L, Bahl V, Wang YM. Distributed topology control for power efficient operation in multihop wireless ad hoc networks. In: Proc. of the IEEE INFOCOM. Alaska: IEEE Press, 2001. 1388−1397. [doi: 10.1109/INFCOM.2001.916634]

[8] Cardei M, Du DZ. Improving wireless sensor network lifetime through power aware organization. Wireless Networks, 2005,11(3): 333−340. [doi: 10.1007/s11276-005-6615-6]

[9] Tian D, Georganas ND. A node scheduling scheme for energy conservation in large wireless sensor networks. Wireless Communications and Mobile Computing, 2003,3(2):271−290. [doi: 10.1002/wcm.116]

[10] Yan T, He T, Stankovic JA. Differentiated surveillance for sensor networks. In: Proc. of the ACM Int'l Conf. on Embedded Networked Sensor Systems. Los Angeles: IEEE Press, 2003. 51−62. http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.12.2637 [doi: 10.1145/958491.958498]

[11] Basu P, Redi J. Movement control algorithms for realization of fault-tolerant ad hoc robot networks. IEEE Networks, 2004,18(4): 36−44. [doi: 10.1109/MNET.2004.1316760]

[12] Lin J. Distributed mobility control for fault-tolerant mobile networks. In: Proc. of the 2005 Systems Communications. New York: IEEE Press, 2005. 61−66. [doi: 10.1109/ICW.2005.42]

[13] Kashyap A, Shayman M. Relay placement and movement control for realization of fault-tolerant ad hoc networks. In: Proc. of the 41st Annual Conf. on Information Sciences and Systems. Baltimore: IEEE Press, 2007. 783−788. [doi: 10.1109/CISS.2007.4298415]

[14] Frew EW, Brown TX, Dixon C, Henkel D. Establishment and maintenance of a delay tolerant network through decentralized mobility control. In: Proc. of the 2006 IEEE Int'l Conf. on Networking, Sensing and Control. 2006. 584−589.

[15] Wang GL, Cao GH, La Porta TF. Movement-Assisted sensor deployment. IEEE Trans. on Mobile Computing, 2006,5(6):640−652. [doi: 10.1109/TMC.2006.80]

[16] Zou Y, Chakrabarty K. Sensor deployment and target localization based on virtual forces. In: Proc. of the IEEE INFOCOM. San Francisco: IEEE Press, 2003. 1293−1303. [doi: 10.1109/INFCOM.2003.1208965]

[17] Lloyd EL, Liu R, Marathe MV, Ramanathan R, Ravi SS. Algorithmic aspects of topology control problems for ad hoc networks. In: Proc. of the IEEE Mobile Ad Hoc Networking and Computing (MOBIHOC). Lausanne: IEEE Press, 2002. 123−134. [doi: 10.1145/1046430.1046433]

[18] Shen Z, Chang YL, Cui C, Zhang X. Study of the minimum-energy path-preserving topology control algorithm for wireless Ad Hoc networks. Journal of Xidian University (Natrual Science), 2006,33(3):341−346 (in Chinese with English abstract).

[19] Gong WB, Chang YL, Shen Z, Zhang Y. Mobile deployment based on minimum coverage overlap in wireless sensor networks. Journal of system simulation, 2008,20(13):3604−3609 (in Chinese with English abstract).

[20] Gong WB. Study of the Node Mobility Techniques [Ph.D. Thesis]. Xi'an: Xidian University, 2009 (in Chinese)

附中文参考文献:

[18] 沈中,常义林,崔灿,张新.无线 Ad Hoc 网络中保留最小能量路径的拓扑控制算法.西安电子科技大学学报(自然科学版),2006,33(3):341−346.

[19] 公维宾,常义林,沈中,张颖.传感器网络中基于最小覆盖重叠的移动部署.系统仿真学报,2008,20(13):3604−3609.

[20] 公维宾.无线 Ad Hoc 网络拓节点移到技术研究[博士学位论文].西安:西安电子科技大学,2009.

公维宾(1974－),男,山东蒙阴人,博士,讲师,主要研究领域为无线 Ad Hoc 网络,网络管理.

沈中(1969－),男,博士,主要研究领域为无线 Ad Hoc 网络,无线传感网络.

常义林(1944－),男,教授,博士生导师,主要研究领域为网络管理,多媒体通信.