文章编号:1001-506X(2011)11-2506-06

无线传感器网络的重叠分簇与边界搜索

廖 鹰[1,2],齐 欢[1],李伟群[1]
(1. 华中科技大学控制科学与工程系,湖北 武汉 430074;
2. 解放军信息工程大学理学院,河南 郑州 450001)

摘 要:无线传感器网络(wireless sensor networks, WSNs)由大量微小的传感器节点组成,分簇的网络架构能较好地处理大规模网络的自组织问题,因而成为 WSNs 提升性能和扩展性的标准方法。在拓扑发现、地理路由和目标追踪等应用中,重叠分簇能更好地满足要求,同时,辨别出 WSNs 的边界节点是重要的任务。与先前的基于节点的边界搜寻算法不同,提出了一种应对节点随机分布情况的自组织分簇算法。建立了重叠分簇,进而对重叠分簇进行分簇的边界融合,最后形成整个 WSNs 网络边界。仿真结果表明,该算法能够生成更为均衡的分簇,显著提高网络生存周期,并能有效的实现网络边界节点的搜索。

关键词:无线传感器网络;随机分布;自组织;重叠分簇;生存周期;边界搜寻
中图分类号:TN 915.04 **文献标志码**:A **DOI**:10.3969/j.issn.1001-506X.2011.11.31

Overlapping clustering and boundary search of wireless sensor networks

LIAO Ying[1,2], QI Huan[1], LI Wei-qun[1]
(1. *Department of Control Science and Engineering, Huazhong University of Science and Technology, Wuhan 430074, China*;
2. *Institute of Sciences, PLA Information Engineering University, Zhengzhou 450001, China*)

Abstract: Wireless sensor networks (WSNs) consist of a large number of sensor nodes. The clustering architecture can deal with self-organization of large-scale networks, so clustering is a standard approach to achieving efficiency and scalability. In the applications of topology discovering, geography routing, tracking and so forth, overlapping clusters are useful, and recognizing boundary nodes is important. Different from the former boundary search algorithm based on single node, a distributed self-organization overlapping clustering algorithm in a random network is proposed to generate overlapping clusters, by means of which the clusters borderline is fused to form boundary of WSNs. Moreover, the results of simulations indicate that the algorithm can construct balanced clusters, search the network boundary effectively and enhance the network survival period obviously.

Keywords: wireless sensor network; random distribution; self-organization; overlapping clustering; survival period; boundary search

0 引 言

无线传感器网络(wireless sensor networks, WSNs)[1]由大量能量有限的传感器节点组成,能够实现对外界物理状态变化的探测和监控。WSNs 的节点可通过人工或者其他方式随机布置(如飞机空投),通过节点之间的自组织形成监测网络,并利用分簇算法[2-9]形成层次化的网络拓扑,这是当前 WSNs 进行网络管理和实现数据汇聚通常采用的方法。在拓扑发现、地理路由和目标追踪等 WSNs 应用中,网络边界节点的确定和网络边界的生成是需要解决的问题。例如,在军事应用中,监控目标(如车辆、潜艇)是否进入或者离开监控区域就是一件非常有意义的工作,这需要准确划分 WSNs 边界,并且精确定位监控目标。

如果传感器节点具备精确的点位,那么确定边界节点将变得简化。但是,在实际应用中,为了降低网络成本,大规模布置的 WSNs 节点通常不具备全球定位系统(global positioning system, GPS)定位功能;此外,WSNs 通常是随机进行布置的,在外界环境的影响下,位置容易发生改变。这些因素都导致了实现边界节点的定位和 WSNs 边界搜寻成为一个困难的问题。

收稿日期:2010-06-23; **修回日期**:2011-07-28。
基金项目:国家自然科学基金(60774036);湖北省自然科学基金重点项目(2008CDA063);中央高校基本科研业务费专项资金(C2009Z025Y)资助课题
作者简介:廖鹰(1979-),男,讲师,博士研究生,主要研究方向为复杂网络理论与应用。E-mail:liaoying1979@163.com

文献[10]提出了无需定位信息,利用三角公式来建立网络边界的分布式算法。文献[11]提出一种分布式的算法,通过寻找定义为 flower 的结构进行边界融合,该算法需要较高的节点度(20～30),当网络的节点度较低的时候可能导致算法失败。文献[12]提出了一种基于分布式计算的方法,利用一系列的节点作为种子节点进行计算,由网络的连接性来求出网络的边界。文献[13]提出了一种利用近似邻居距离的启发式算法,并给出了集中式和分布式实现。上述算法进行边界搜寻是从节点出发,形成一定的节点结构或组合来进行融合,可能会因为网络的拓扑信息不全或某个节点的实效导致算法失败。

文献[14]提出了重叠多跳分簇算法的概念,提出了适用于传感器节点均匀分布的 k 跳重叠分簇算法(k-hop overlapping clustering algorithm, KOCA)算法,每个节点以等概率成为簇头,并且未考虑节点的剩余能量,并不能满足实际情况的需求。

本文在前人工作的基础上,提出了一种应对节点非均匀分布情况下的多跳自组织重叠分簇算法(self-organization overlapping algorithm for clustering, SOAC),将 SOAC 分簇算法应用到边界搜索中,建立 WSNs 的重叠分簇,并在重叠分簇的基础上进行分簇的边界融合,最终形成网络边界。

1　系统模型

已知传感器网络节点组成非空有限集合 V,整个网络可以抽象为一个无向图 $G=(V,E)$。其中 V 表示一系列的无线传感器节点,$E\subseteq V^2$ 表示为节点间有效连接(边)的集合。E 中的一条边:$e=(u,v)$ 表示节点 u 与节点 v 能直接相互通信。本文研究的对象为非均匀分布的多跳无线传感器网络,每一个节点 u 都被分配一个独一无二的标识符。

传感器节点并未装备 GPS 装置,无法自身进行定位。本文作如下假定:所有节点进行自组织形成网络拓扑,进行数据汇聚;所有节点以同样强度的信号传递数据,具备同样的最大传输距离 L,在同一频率上传输信号(共享信道),并且信道是自由竞争且无错的;所有节点间进行双向连接,不考虑单向连接的情况;分簇算法的刷新时间受节点的状态变化(如节点死亡)约束;所有的传感器节点在周期 $T(r)$(分簇刷新时间)内是固定不移动的,这也是传感器网络建模通常采用的方式。无线通信系统模型参考了文献[2],当发送 l 比特数据到距离 d 时,收发双方的能耗分别为 $E_{Tx}(l,d)$ 和 $E_{Rx}(l)$

$$E_{Tx}(l,d)=E_{Tx-elec}(l)+E_{Tx-amp}(l,d)=\begin{cases}lE_{elec}+l\varepsilon_{fs}d^2, & d<d_o\\ lE_{elec}+l\varepsilon_{mp}d^4, & d>d_o\end{cases}\tag{1}$$

$$E_{Rx}(l)=E_{Rx-elec}(l)=lE_{elec}\tag{2}$$

2　SOAC 分簇算法

传统的分簇算法(如 LEACH、HEED、K-clustering 等),它们的目标是形成一些不连接的分簇,从而形成网络的拓扑。然而,在传感器网络的实际应用中,如节点定位、目标追踪和网络鲁棒性提升,重叠分簇网络能够更好的满足要求[14]。在重叠分簇网络中,某个节点可能属于多个分簇,分簇与分簇之间的通信将变得更为可靠。

与应用于均匀分布传感器节点的 KOCA 算法不同,SOAC 算法能够在非均匀分布的节点中自组织生成 k-跳重叠分簇网络,其分簇中存在 3 种类型的节点:簇头节点、重叠节点和普通节点。图 1 给出了 SOAC 算法生成分簇后网络拓扑的变化,以及重叠节点在分簇中的作用。如图 1 所示,重叠节点可以属于多个分簇,连接不同的分簇并完成数据的传输和转发,能够有效提高网络的鲁棒性。图 1 中分簇 1 和分簇 2 的重叠节点为 b_1、b_2、b_3,b_3 同时属于分簇 3;b_2 的分簇信息表中 HID 有 A 和 B。SOAC 算法的实现分为簇头选择阶段和分簇形成两个阶段。

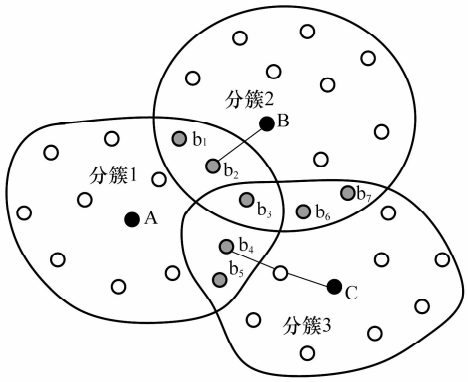

图 1　SOAC 算法生成重叠分簇示例

2.1　簇头选择阶段

SOAC 算法采用分布式的方式进行分簇,每个节点不需要中央控制,自组织形成分级的结构。分簇的簇头由选举产生,k-跳邻居中权重最高的节点当选为簇头。节点的权重利用式(3)计算,本文将节点的剩余能量和节点连接密度考虑在内,这样能够生成能量和位置分布上更为均衡的分簇。

$$W(u)=a\times P[D_k(u)]+b\times P\left[\frac{R_e(u)}{E(u)}\right]-c\times P[H(u)]$$
$$0\leqslant a,b,c\leqslant 1;\ c<a+b<1\tag{3}$$

式中,a、b、c 为影响因子,影响算法的收敛速度,由网络规模和应用环境决定;$D_k(u)$ 表示节点的连接密度;$R_e(u)$ 表示节点 u 的剩余能量;$E(u)$ 为节点 u 的初始能量;$H(u)$ 为节点当选过簇头的次数。在式(3)中设置 $P[H(u)]$,可以降低当选过簇头的节点继续成为簇头的概率,均衡能量消耗。

距离节点 u 跳数不超过 k 的节点的集合记为 $N_k(u)$,即 k-跳邻居

$$N_k(u)=\{v\in V\mid v\neq u\wedge d(u,v)\leqslant k\}\tag{4}$$

式中,$d(u,v)$ 表示节点 u 与 v 之间的跳数,本文用跳数近似表示距离。

节点的连接密度通过式(5)计算,其中 $|N_k(u)|$ 为 u 的 k-跳邻居的总个数

$$D_k(u)=\frac{|(t,v)\in E\mid t,v\in N_k(u)\cup\{u\}|}{|N_k(u)|} \tag{5}$$

在初始阶段，一个随机节点触发分簇过程成为临时簇头，并向它的 k-跳邻居发送临时簇头广播。该节点和它的 k-跳邻居利用式(3)计算自身的权重，拥有最高权重的节点成为簇头。簇头节点采用广播的方式向 k-跳邻居发送簇头广播信息通告自己成为簇头，要求它们加入分簇。簇头广播信息包括簇头节点标识号、发送节点标识号和距离簇头的跳数。为了避免广播风暴，簇头首先向一跳邻居发送簇头广播，1 跳邻居接收了簇头广播后将自身的距簇头跳数设置为 1，如果 1 跳邻居向 2 跳邻居发送广播包时，1 跳节点收到这个广播包就将其丢弃。这样就节省了对距簇头跳数频繁设置的能量。多跳分簇的情况，就按上述过程扩散下去。当节点接收到簇头广播后，发送节点标识号可以用来维护一条到达簇头的路径；若当前广播信息中距簇头跳数为 k 则拒绝转发，这样保证分簇不超过 k-跳。所有 k-跳邻居节点接收到簇头广播后，即使它已经属于其他分簇，只要自己的权重低于簇头广播中簇头的权重，就发送同意加入分簇信息给簇头。簇头广播被限制在 k-跳内传播，因此有可能导致某些节点无法接收到任何邻近的簇头发出的簇头广播。SOAC 算法设置在经过 $T(t)$ 后，如果某个节点没有接收到该信息，则发出簇头广播，宣布自己成为临时簇头。其中，$T(t)$ 为分簇等待时间($T(t)T(r)$)。$T(t)$ 和 $T(r)$ 的设置应保证网络每个节点都能够找到自己的簇头，每经过 $T(r)$ 重新启动分簇过程。

2.2 分簇形成阶段

SOAC 算法对分簇的大小设置了门限，分簇中节点个数不能超过门限值，目的是避免形成过大分簇，否则会造成簇内通信量过大，导致网络的生存周期下降。当簇头节点接收到节点发送的加入分簇请求信息后，将管理的分簇的节点数与门限进行比较，若未达到门限，则同意新成员加入分簇，并更新分簇节点计数，否则拒绝请求。被拒绝的节点如果已经有其他的簇头，则分簇建立过程中止；如果没有加入分簇，则该节点成为临时簇头，进行 2.1 节中描述的处理过程。

每个分簇成员节点都维护一个分簇信息表，保存簇头节点标识号、发送节点标识号和距离簇头跳数等信息。如果某个节点接收到网络中传递的数据包，则会对应着更新自己分簇信息表的信息。例如，当接收到来自相同簇头的数据包，节点会检查其中的距离簇头跳数，如果低于自己分簇信息表中的距离簇头跳数值，则更新表中该值，并将发送节点标识号更新，表明已经找到一个到达簇头更近的路径，以新的发送节点标识号作为自己的转发节点。对于普通节点，其分簇信息表只有单一的簇头信息条目，表明其只属于某一个分簇；而重叠节点的分簇信息表中有多个簇头信息条目，将不同的邻近分簇连接起来。

SOAC 算法避免了固定的簇头方案(簇头管理分簇并完成分簇数据转发，能量消耗得更快)，采用周期更换的方案来均衡节点的能量消耗。簇头选举完成，经过 $T(r)$ 后，簇头重新选举过程被触发：所有分簇节点向簇头发送自己的更新后的权重(经过一轮分簇运行后连接密度和剩余能量发生变化)；簇头接收到更新权重信息后选择权重最大的节点成为下一轮簇头；新簇头得到簇头通知后，发出簇头广播，进入新一轮的分簇形成阶段。通过该过程，原来的双向交互变成了单向通信，减少了随机节点发出权重询问广播的能量消耗，节省了一半的能量消耗。

在图 2 给出的 SOAC 算法的基本实现流程中，本文给出了基于针对传感器节点的实施细节。

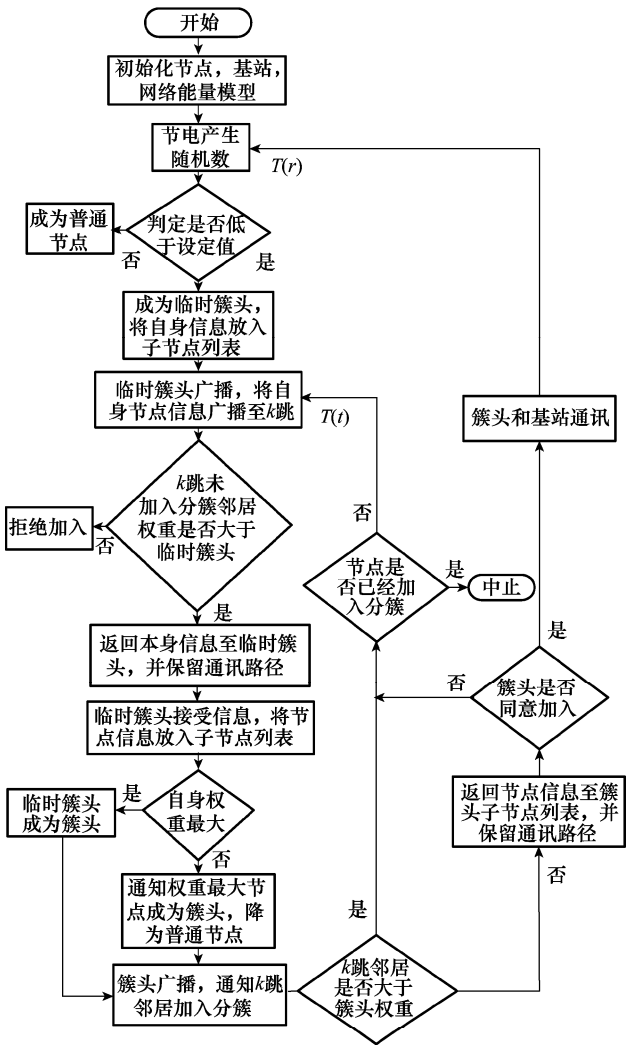

图 2 SOAC 算法实现流程图

3 边界搜寻算法

边界搜寻算法的基本思想是：基于 SOAC 算法形成分簇，进而生成每个分簇的边界，接着融合各分簇的重叠节点，将多个分簇进行整合。在 SOAC 分簇完成后，网络的拓扑已经形成，在互相重叠的分簇之间，重叠节点起到了衔接作用。边界搜寻算法将重叠的分簇边界进行融合，形成扩展边界，如图 3 所示，并以此递归下去，最终将整个 WSNs 的分簇边界融合。SOAC 算法生成的重叠分簇，能够有效

地形成 WSNs 边界。

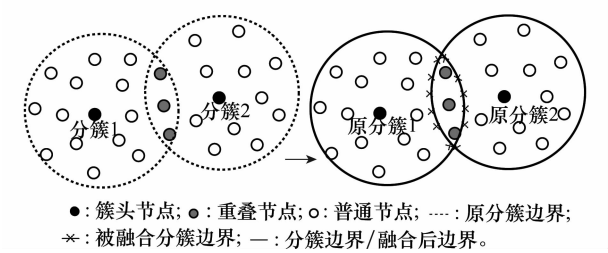

图 3　重叠分簇间进行边界融合示例

理想情况下,SOAC 算法可以形成完全重叠的分簇架构,即任意的两个分簇都具备重叠节点。但在实际应用中,受限制于分簇大小或者地理位置分布,很可能形成互相孤立的分簇。本文提出的边界搜寻算法很好地解决了该问题。对于孤立分簇的情况,分簇的边界节点向它所属分簇以外的一跳邻居发送查询请求,若相邻分簇的边界节点接收到查询请求,并且存在两个或两个以上的边界节点互为邻居节点,这两个相邻分簇可以进行边界融合,如图 4 所示。

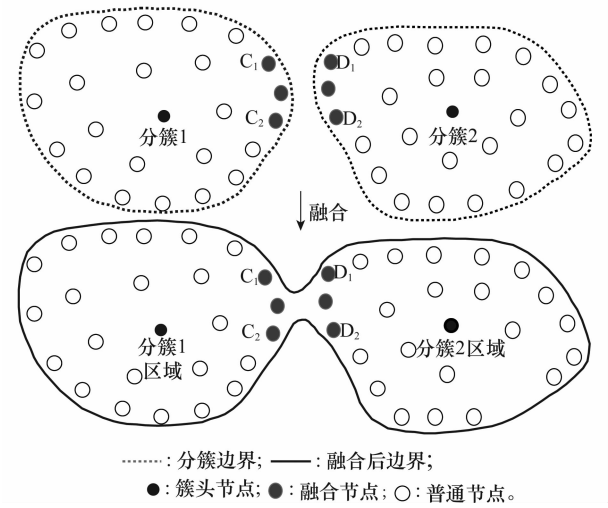

图 4　非重叠分簇进行边界融合示例

在通过握手通信确定了相邻分簇的邻居节点后,算法分别找寻出在各自分簇中距离最远并在相邻分簇中拥有邻居节点的边界节点,如图 4 所示,算法得出分簇 1、2 中的 C_1 和 D_1、C_2 和 D_2 互为邻居节点,且 C_1 和 C_2 在分簇 1 的边界节点中边界距离最远,D_1 和 D_2 边界距离最远,那么不论 C_1 和 C_2 以及 D_1 和 D_2 之间还有没有互为邻居的边界节点,两个分簇以 C_1 和 D_1 以及 C_2 和 D_2 组成新边界的一部分,即边界节点为{……C_1、D_1…… C_2、D_2……}。

边界搜寻算法的实现过程如下:在 SOAC 完成分簇后,簇头根据实时更新的分簇信息表区分出边界节点并生成本分簇的边界,接着检查重叠节点,试图找寻相邻分簇中的重叠节点,按照图 3 所示进行第一步的边界融合;如果相邻分簇没有两个或两个以上的互为邻居的连续重叠节点,那么启动孤立分簇边界融合进程,即分簇的边界节点(包括融合后的新分簇边界节点)向其一跳邻居发送查询请求,寻找相邻分簇中的邻居节点。本文的融合算法要求至少存在两个一跳邻居才有可能进行边界融合,当发现两个相邻分簇中存在互为邻居节点的节点组合后,按照图 4 所示进行边界融合,算法需要找寻在各自分簇中距离最远并在相邻分簇中拥有邻居节点的边界节点;分簇边界融合的对象是互相靠近的两个分簇边界,而不是相互远离的分簇边界。

4　仿　真

4.1　分簇算法仿真

在仿真实验中,WSNs 节点随机分布在 50 m×50 m 的区域中;为了便于比较,本文采用了文献[2]中的模型,并参考了文献[15]对多跳分簇仿真的实现,仿真工具选用 Matlab-2009 a;在仿真中,将目标区域设置为[0,50]×[0,50],基站放置在[25,100],分簇门限设为 15。

SOAC 算法应先考虑的是如何设置跳数 k。在仿真中,k 分别被设置为 1、2、3 和 4($k>4$ 会造成分簇过大或过多节点无法加入分簇,网络的生存周期下降)。图 5 和图 6 分别给出了 160 个和 200 个节点随机分布的分簇仿真结果。为了方便统一进行比较,本文将网络生存时间用 WSNs 与基站通信的轮次数来表示。在进行算法比较时,与基站通信的轮次一定的情况下,死亡节点数所占全部节点数的比例越低的算法被认为是更优。

从图 5 和图 6 看出,$k=2$ 在两种随机分布的情况下均取得了较佳的性能,分簇跳数的增加带来的结果是平均分簇节点个数的增大,分簇管理和通信消耗的能量过多,造成了性能的下降。

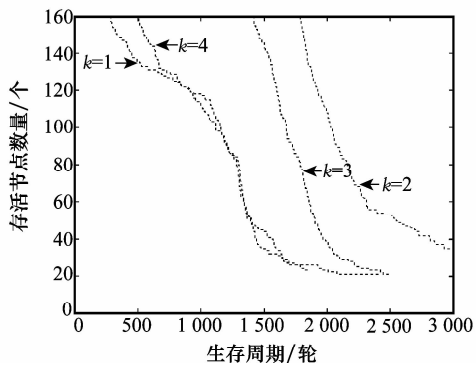

图 5　不同跳数 k 下生存周期的比较(160 个节点)

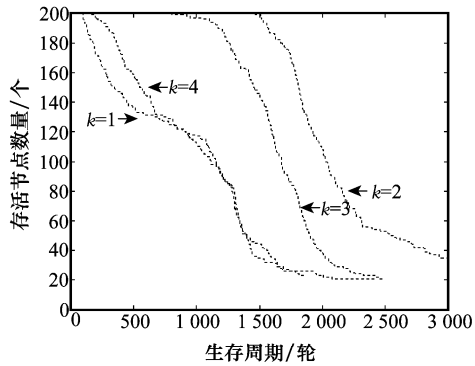

图 6　不同跳数 k 下生存周期的比较(200 个节点)

本文将 SOAC 与经典的低能耗自适应层次分簇算法(low-energy adaptive clustering Hierarchy, LEACH)、混合能量有效分布式分簇(hybrid energy-efficient distributed clustering, HEED)算法和权重分簇算法(weighted clustering algorithm, WCA)作比较,重点观察不同比例节点死亡时和节点数量变化时各种算法生存周期。

图 7 给出了节点数变化的情况下,各种算法生存周期的变化。仿真条件:$k=2$、100、150、200 和 250 个传感器节点随机分布,各选取 10 次随机分布,对结果求平均值。

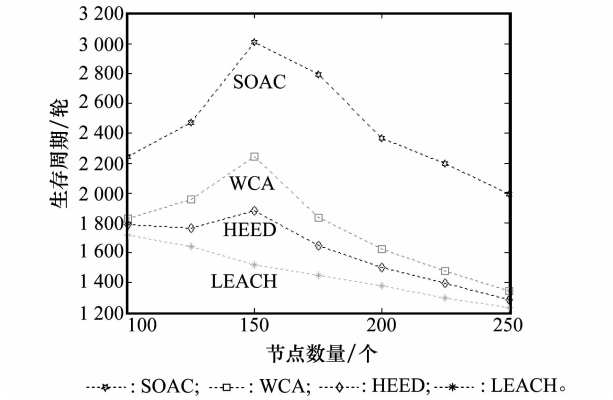

图 7 节点数量变化时算法生存周期的比较

从图 7 可以得出,随着节点数的增加,各种算法表现的差异较大,在节点数为 150 的情况下,SOAC、WCA 和 HEED 均取得了各自的最佳性能,而 LEACH 算法随着节点数增多持续下降。当节点数继续增加(>150),4 种算法均出现了性能下降。这是由于过于密集布置的节点会导致分簇的节点数量增加,造成分簇内部能量消耗增大,反而造成了性能的降低。

图 8 给出了当 10%、20%、30%、40%、50%、60%、70%和 80%的节点死亡时,各种算法的与基站通信的轮次数。仿真条件:$k=2$,随机选取 10 次 150 个节点的随机分布对结果求平均值。

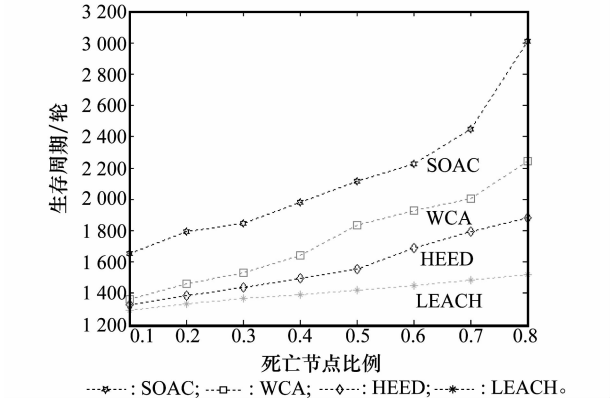

图 8 不同比例节点死亡时算法生存周期的比较

由图 8 可以看出,当 10%的节点死亡时,SOAC 与基站通信了 1 655 轮,优于其他算法;而当 70%节点死亡时,SOAC 进行了 2 450 轮,也是优于其他算法;在其他比例下,SOAC 同样表现更优。这是由于 SOAC 生成的分簇更加均匀,减少了簇内通信的能量开销,在网络生存周期上表现得更优。同时,SOAC 算法并未因存在重叠节点导致分簇性能有过多的降低,还能够更好的应用边界搜寻算法。

SOAC 能够得到更合理的分簇结构,同时在完成第一次簇头选举后,避免了节点再次交互权重、选举簇头的广播通信,从而有效减少簇内通信的能量消耗。LEACH、HEED、WCA 等算法每轮分簇后分簇结构都发生改变,SOAC 算法每一轮分簇的结构保持相对的稳定,簇头切换发生在簇内节点之间。

LEACH 是随机生成簇头没有考虑节点的剩余能量,实际分簇的效果不佳;HEED 算法考虑了剩余能量,但形成的分簇结构与实际网络不符,不能有效提高网络生存周期;而 WCA 算法需要每个节点保持所有的节点信息,计算和通信开销过大。SOAC 分簇结构与簇头的选择综合考虑节点的分布和剩余能量,生成的簇结构更符合实际网络节点不均匀分布的情况,因而在节点数变化的情况下依然取得了更好的性能。

4.2 边界搜索算法仿真

为了分析边界搜索的效果,本文利用算法最终生成的网络边界所包含的节点数与网络总节点数的比值(覆盖率)来衡量:生成的边界覆盖率越高,边界搜索算法的性能越好。

在仿真中,为更好的分析边界搜索在不同分布下的性能,本文分别选取网络节点数为 125、150、175、200、225、250 和 275 进行仿真,节点分布区域设置为[0,50]×[0,50]。

在边界搜索仿真中,本文分别对每一类网络节点总数(125、150、175、200、225、250 和 275)各选取 10 次随机分布进行计算求平均值,得到的仿真结果如图 9 所示。当节点数为 200 个时,边界搜索的性能最佳,平均覆盖率达到了 94.2%;当节点数大于 200 个,随着节点数量的增加,覆盖率略微下降,但保持在 93%以上;在 125 个节点的情况下,平均覆盖度是最低,达到了 90.7%。仿真结果说明,本文设计的边界搜索算法达到了设计算法的要求,这主要是由于该算法是基于良好的分簇结构,以分簇为单位进行边界搜索,提升了可靠性。

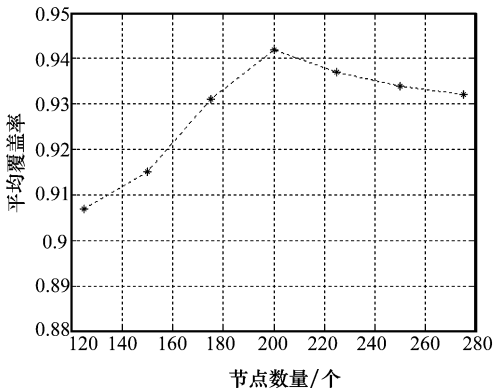

图 9 节点数变化时边界搜索覆盖率对比

图 10 给出了在 50 m×50 m 区域中对 150 个随机分布节点进行 SOAC 分簇的仿真拓扑图。从图中可以看出,

SOAC 生成均衡的分簇,而且重叠节点较好的衔接了多个分簇,便于进行分簇边界融合。

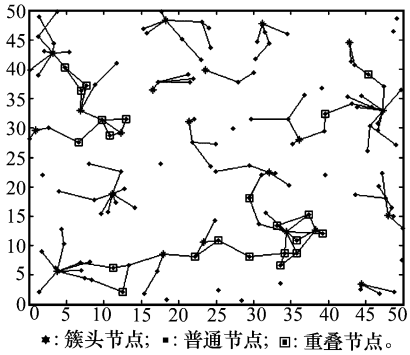

图 10　SOAC 分簇结构仿真结果(k=2 节点数 150)

图 11 给出了对图 10 的分簇结构进行边界搜索的仿真结果,可以看出边界搜索算法能够形成封闭的整个 WSNs 网络边界,达到了设计算法的目的。SOAC 分簇优化了能量消耗,保证了生存周期,在非均匀的分布的不利情况下,充分利用了有限的重叠节点,交互分簇信息,并尽可能克服了无重叠节点分簇边界融合的问题,最大限度的确立了整个 WSNs 的边界。

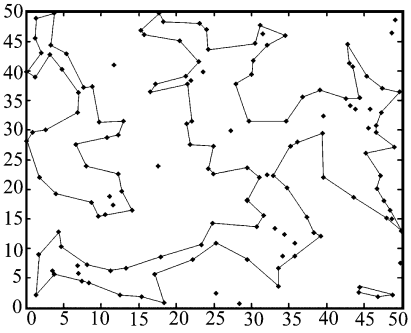

图 11　边界搜索仿真结果(k=2 节点数 150)

本文采用的边界搜索算法是基于分簇的,在形成 SOAC 分簇后的分簇边界融合相比单个节点发起的边界搜索要更为可靠和有效。

5　结　论

本文提出了针对非均匀分布的 WSNs 的多跳重叠分簇算法。同传统的分簇算法相比,可以形成更为稳定合理的簇结构,并且大幅度提高了网络生存周期,且分簇是重叠的方式构成的,有利于高效准确的进行分簇的边界融合,形成整个 WSNs 边界。基于分簇进行边界搜寻的方式,有别于以往基于节点进行边界搜寻的方式,能够提高搜寻的合理性和可靠性。在具体实现中,本文考虑了重叠分簇的边界融合的情况以及非重叠分簇的边界融合的情况,仿真实验也证明该算法具备可行性。

参考文献:

[1] Tilak S, Abu-Ghazaleh N B, Heinzelman W. A taxonomy of wireless micro-sensor network models[J]. *Mobile Computing and Communications Review*, 2002, 6(2): 1 - 8.

[2] Heinzelman W B, Chandrakasan A P, Balakrishnan H. An application-specific protocol architecture for wireless microsensor networks[J]. *IEEE Trans. on Wireless Communications*, 2002, 1(4): 660 - 670.

[3] Younis O, Fahmy S. HEED: a hybrid, energy-efficient, distributed clustering approach for ad hoc sensor networks[J]. *IEEE Trans. on Mobile Computing*, 2004, 3(4): 366 - 379.

[4] Xu Y, Heidemann J, Estrin D. Geography-informed energy conservation for ad hoc routing[C]// *Proc. of the 7th Annual Int's Mobile Computing and Networking*, 2001: 70 - 84.

[5] Chatterjee M, Das S K, Turgut D. WCA: a weighted clustering algorithm for mobile ad hoc networks[J]. *Cluster Computing Journal*, 2002, 5(2): 193 - 204.

[6] Basagni S. Distributed clustering for ad hoc networks[C]// *Proc. of the International Symposium on Parallel Architectures, Algorithms and Networks*, 1999: 310 - 315.

[7] Bettstetter C, Friedrich B F. Time and message complexities of the generalized distributed mobility-adaptive clustering (GD DMAC) algorithm in wireless multihop networks[C]// *Proc. of the IEEE Vehicular Technology Conference*, 2003: 176 - 180.

[8] Fernandess Y, Malkhi D. K-clustering in wireless ad hoc networks[C]// *Proc. of ACM International Workshop on Principles of Mobile Computing*, 2002: 31 - 37.

[9] Lehsaini M, Guyennet H, Feham M. A novel cluster-based self-organization algorithm for wireless sensor networks[C]// *Proc. of the Collaborative Technologies and Systems*, 2008: 19 - 26.

[10] Deogun J S, Das S, Hamza H S, et al. An algorithm for boundary discovery in wireless sensor networks[C]// *Proc. of the 12th High Performance Computing*, 2005: 343 - 352.

[11] Fekete S P, Kroller A. Geometry-based reasoning for a large sensor network[C]// *Proc. of the 22nd ACM Symposium Computational Geometry*, 2006: 475 - 476.

[12] Funke S, Klein C. Hole detection or: how much geometry hides in connectivity[C]// *Proc. of the 22nd ACM Symposium Computational Geometry*, 2006: 377 - 385.

[13] Sitanayah L, Datta A, Cardell-Oliver R. Heuristic algorithm for finding boundary cycles in location-free low density wireless sensor networks[J]. *Computer Networks*, 2010, 54(10): 1630 - 1645.

[14] Youssef M, Youssef A, Younis M. Overlapping multihop clustering for wireless sensor networks[J]. *IEEE Trans. on Parallel and Distributed Systems*, 2009, 20(12): 1844 - 1856.

[15] 姜华,郑春雷,刘海涛. 无线传感器网络的系统建模与仿真实现[J]. 系统工程与电子技术, 2007, 29(1): 92 - 95. (Jiang H, Zheng C L, Liu H T. Research of system model and simulation in WSN[J]. *Systems Engineering and Electronics*, 2007, 29(1): 92 - 95.)