软件学报 ISSN 1000-9825, CODEN RUXUEW
Journal of Software,2016,27(7):1822−1840 [doi: 10.13328/j.cnki.jos.004926]

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

传感网中延迟限定的非汇聚数据移动式收集*

梁俊斌[1], 邹绍军[1,2], 陈宁江[1], 李 韬[3]

[1](广西大学 计算机与电子信息学院 广西多媒体通信与网络技术重点实验室,广西 南宁 530004)

[2](中南大学 信息科学与工程学院,湖南 长沙 410083)

[3](Department of Computing, Hong Kong Polytechnic University, Hong Kong)

通讯作者: 邹绍军, E-mail: zoushj@csu.edu.cn

摘 要: 在大规模的无线传感器网络中收集数据,不仅需要考虑节点的能量消耗,而且还需要考虑数据收集延迟.如何有效地均衡节点的能量消耗,同时最小化数据收集延迟,是一个具有挑战性的问题.为了均衡节点的能量消耗,利用移动数据收集器收集数据.以此为基础,提出一种 DC-Collection 算法来解决数据收集延迟和能耗的问题.首先,在网络中构造最短路径树,网络非连通时,不同的网络子图可以构造多棵最短路径树,它们构成一个最短路径树集合;其次,在每一棵最短路径树上选取部分节点作为采集节点和逗留节点,使得以采集节点为根的限高树的高度不超过 h,且在每个采集节点的通信区域内至少有一个逗留节点;再次,在每棵限高树内调整树的结构,让能量高的节点承担更多的子孙节点,最大化限高树的生命周期;最后,移动数据收集器从 Sink 出发,遍历逗留节点所在位置收集数据,最终回到起点,并将数据发送给 Sink.通过理论分析和大量仿真实验,其结果表明:与现有的数据收集协议相比,DC-Collection 不仅能够均衡各节点的能量消耗从而延长网络生命周期,而且能够缩短移动数据收集器收集数据行走的路径长度,从而缩短数据收集延迟.

关键词: 无线传感器网络;非汇聚数据收集;限高树;延迟限定;网络生命周期

中图法分类号: TP393

中文引用格式: 梁俊斌,邹绍军,陈宁江,李韬.传感网中延迟限定的非汇聚数据移动式收集.软件学报,2016,27(7):1822−1840. http://www.jos.org.cn/1000-9825/4926.htm

英文引用格式: Liang JB, Zou SJ, Chen NJ, Li T. Delay-Constrained data collection without aggregation in wireless sensor network with mobile collector. Ruan Jian Xue Bao/Journal of Software, 2016,27(1):1822−1840 (in Chinese). http://www.jos.org.cn/1000-9825/4926.htm

Delay-Constrained Data Collection Without Aggregation in Wireless Sensor Network With Mobile Collector

LIANG Jun-Bin[1], ZOU Shao-Jun[1,2], CHEN Ning-Jiang[1], LI Tao[3]

[1](Guangxi Key Laboratory of Multimedia Communications and Network Technology, School of Computer and Electronic Information, Guangxi University, Nanning 530004, China)

[2](School of Information Science and Engineering, Central South University, Changsha 410083, China)

[3](Department of Computing, Hong Kong Polytechnic University, Hong Kong)

* 基金项目: 国家自然科学基金(61562005, 61363003); 香江学者计划(XJ2013028); 广西自然科学基金(2015GXNSFAA139286); 广西高校优秀人才资助计划([2011]40); 广西高校科技研究项目(KY2015YB486); 广西高等学校优秀中青年骨干教师培养工程(GXQG012013034)

Foundation item: National Natural Science Foundation of China (61562005, 61363003); Hong Kong Scholars Program (XJ2013028); Natural Science Foundation of Guangxi Province, China (2015GXNSFAA139286); Excellent Talents Scheme of Guangxi Universities ([2011]40); Scientific and Technological Research Projects of Guangxi (KY2015YB486); Project of Outstanding Young Teachers' Training in Higher Education Institutions of Guangxi (GXQG012013034)

收稿时间: 2014-10-19; 修改时间: 2015-03-10, 2015-05-18, 2015-07-30, 2015-09-10; 采用时间: 2015-09-30; jos 在线出版时间: 2015-11-27

CNKI 网络优先出版: 2015-11-26 16:06:10, http://www.cnki.net/kcms/detail/11.2560.TP.20151126.1606.003.html

Abstract: For data gathering in large-scale wireless sensor networks, not only energy consumption of nodes but also data collection delay should be considered. It is a challenging problem to achieve the goal of balancing energy consumption of nodes and minimizing data collection delay in the network at the same time. In order to balance energy consumption of nodes, this paper utilizes a mobile data collector to collect data in the network, and proposes an algorithm named DC-Collection to solve the problem of minimizing data collection delay and energy consumption. First, DC-Collection constructs a shortest path tree. If the network is not connected, there are more than one shortest path trees (i.e., there is a set of shortest path trees in the network). Second, some nodes in the trees are selected as collective nodes or lingering nodes, where a collective node is the root of a height-limited tree that will receive data sent from its descendant nodes and a lingering node is a normal node that the mobile data collector will visit at a given time to collect sensing data. The mobile data collector can collect the data of all nodes in the network as long as it traverses the locations of all lingering nodes. The heights of the trees that rooted at the collective nodes are limited to be smaller than h. There is at least a lingering node exists in the communication area of each collective node. Third, DC-Collection adjusts the structures of the height-limited trees. It makes nodes with higher energy level possess more descendants, so as to maximize the network lifetime by balancing the energy consumption of nodes. Finally, the mobile data collector starts from a Sink, and traverses locations of the lingering nodes in sequence to collect data. After collecting all the data, it returns to the starting point and uploads the data to the Sink. Compared with existing algorithms, theoretical analyses and simulations show that DC-Collection can not only balance the energy consumption of nodes to prolong the network lifetime, but also shorten the path length that the mobile data collector walks to reduce the data collection delay.

Key words: wireless sensor network; data collection without aggregation; height-limited tree; delay-constrained; network lifetime

近年来,很多研究者利用移动节点[1−6]在无线传感器网络(wireless sensor network,简称 WSN)中收集数据.移动节点充当数据收集器(mobile data collector,简称 MDC)的角色[7].利用 MDC 收集数据主要有两大好处:一是尽管网络不连通,移动数据收集器在监视区域内行走也可以收集到全网节点感知的数据;二是节点可以采用单跳或较小跳数的方式将数据转发给移动数据收集器,有利于保存节点的能量.简言之,基于移动数据收集器的 WSN 能够更好地应用于实际环境中.

考虑到 MDC 有诸多优点,与文献[1−6]一样,本文利用一个 MDC 在 WSN 中定期地收集数据.然而在实际应用中,移动数据收集器的移动速度较低,如果没有一个较好的数据收集协议或机制,MDC 需要行走较长的路径才能收集完全网节点感知的数据.这样容易导致数据收集延迟过大,难以适用于延迟敏感的应用中.例如在森林火险监测、矿道瓦斯监测等应用中,希望在尽可能短的时间内收集到全网节点感知的数据并进行处理.因此,在研究如何设计数据收集协议时,不仅需要考虑节点的能量消耗,而且还需要考虑数据收集延迟的问题.

为了均衡节点的能量消耗和最小化数据收集延迟,本文提出了延迟限定的非汇聚数据收集 DC-Collection (delay-constrained data collection without aggregation)算法.该算法是在最短路径树的基础上,选取部分节点为采集节点和逗留节点.采集节点是限高树的根节点,用于收集其子孙节点感知的数据.根据 DC-Collection 算法,在逗留节点的通信范围内通常有多个采集节点,形成以逗留节点为中心的缓存池.当移动数据收集器到达逗留节点所在位置时,即可收集到逗留节点通信范围内所有采集节点收集到的数据.通过理论分析和大量的仿真实验,其结果表明:与现有的数据收集协议相比,DC-Collection 不仅能够有效地均衡各节点的能量消耗从而延长网络生命周期,而且能够有效地缩短数据收集路径从而缩短数据收集延迟.

本文第 1 节主要介绍 WSN 中数据收集协议的研究现状和相关工作.第 2 节是网络模型、问题描述以及问题建模.第 3 节详细阐述 DC-Collection 算法.第 4 节对 DC-Collection 算法进行理论分析.第 5 节通过仿真实验,验证 DC-Collection 算法的有效性.第 6 节总结全文.

1 相关工作

早期的 WSN 研究主要是围绕静态 WSN(即,节点和 Sink 部署后静止不动)进行.在静态 WSN 中,Sink 周围的邻居节点需要转发大量数据而过快地耗尽了能量,从而缩短了网络的生命周期[8].为了解决节点能量消耗不均衡的问题,Shah 等人较早利用移动数据收集器在稀疏的网络中收集传感器的数据[9].文献[9]指出:任何移动并且带有通信功能设备的物体都可作为移动数据收集器,例如带有通信设备的人、动物或车辆等.下面根据移动数据收集器的功能特性将 WSN 分为以下几大类.

1.1 利用移动数据收集器收集和处理数据的研究

移动数据收集器不仅要存储节点感知的数据,而且还要对收集到的数据进行处理并将处理结果传送给用户.这类数据收集器一般是指具有移动能力的 Sink,简称移动 Sink.根据网络中的节点是否以单跳的方式将数据发送给移动 Sink,可以进一步将 WSN 分为以下几个子类.

(a) 节点以单跳的方式将数据发送给移动 Sink 的研究

为了最大限度地保存节点的能量从而最大化网络生命周期,节点将自身感知的数据存储在本地的存储空间.当移动 Sink 与节点能够直接通信时,节点以单挑的方式直接将数据发送给 Sink.该类数据收集协议比较典型的工作有 OSCDG[3]和 ASTLLDG[4].

OSCDG[3]利用一个数据骡子(一种移动 Sink)收集数据.Sugihara 等人主要解决的问题是:如何控制数据骡子的移动,使得数据骡子能够在尽可能短的时间内收集到所有节点的数据.他们将所需解决的问题进一步划分成 3 个子问题:

1) 选取数据骡子行走的路径;
2) 数据骡子动态改变在行走过程中的速度;
3) 数据骡子确定收集各节点的数据的时间表.

ASTLLDG[4]将整个网络区域划分成若干个大小相等的正方形区域.在数据收集阶段,移动 Sink 沿着划分好的正方形区域的边界从左到右行走,到达网络边界的时候逆向行走(即,从右到左行走)来收集数据.由于移动 Sink 所行走的路径穿越了网络中所有节点的通信范围,从而保证了节点能够以单跳的方式将数据发送给移动 Sink.

节点以单跳的方式将数据发送给移动 Sink 虽然可以最大限度地保存各节点的能量,但是由于 Sink 所行走的路径长度较大,容易造成较大的数据收集延迟.因此,该类数据收集协议难以适用于延迟敏感的应用中.

(b) 节点以多跳的方式将数据发送给移动 Sink 的研究

为了缩短数据收集延迟,节点通过多跳的方式将数据发送给移动 Sink.比较典型的做法是:将网络划分成多个簇,在每个簇中选取能量高的节点作为簇头,簇内的其他节点将数据发送给簇头,移动 Sink 只要访问簇头即可收集到全网的数据.该类数据收集协议比较典型的工作有 MobiCluster[5]和 EEBDG[6].

2012 年,Konstantopoulos 等人提出了分布式的 MobiCluster[5]数据收集协议:首先,借鉴文献[10]选取簇头的思想,将靠近移动 Sink 行走路径的节点分配到一个面积较小的簇中,而将距移动 Sink 行走路径较远的节点分配到一个面积较大的簇中,这样能够有效地均衡节点的能量消耗;然后,选取能量充足且能够在移动 Sink 行走的过程中保持与 Sink 通信的节点作为汇合节点;最后,簇头将收集到的数据发送到相应的汇合节点,汇合节点再将数据发送给移动 Sink.

针对节点密度高且节点是呈高斯混合分布的 WSN,EEBDG[6]利用改进后的最大期望值技术选取簇的图心;然后计算所有簇的图心所在位置构成的最优 TSP 路径;最后,移动 Sink 沿着 TSP 路径访问每个簇的图心来完成数据收集.

1.2 利用移动数据收集器收集但不处理数据的研究

为了均衡各节点的能量消耗,许多研究者利用移动数据收集器在网络中收集各节点感知的数据.移动数据收集器只是用于暂存节点感知的数据,最终将收集到的数据上传给静态 Sink 或基站.比较典型的数据收集协议有 LBCDG[11],TPDG[12],SPT-DGA[13].

为了延长网络生命周期和数据收集延迟,LBCDG[11]将网络中的节点划分成 3 层:节点在最底层,簇头在中间层,SenCar(数据收集器)在最顶层.LBCDG[11]与其他分簇的数据收集协议的不同之处在于,每个簇中不止一个簇头,这些簇头通过协调完成簇内的数据收集.对于簇中的每一个节点,在它的一跳范围之内至少有一个簇头.首先,SenCar 利用 MIMO(multiple-input and multiple-output)技术在每个簇中选取一些访问点;然后,SenCar 遍历这些访问点即可收集到存储在这些簇头中的数据;最后,SenCar 将收集到的数据发送给静态 Sink.

TPDG[12]是从节约能量的角度出发,网络中的节点通过单跳的方式将数据发送给移动数据收集器,最大化网络生命周期.首先,节点和 Sink 部署后,移动数据收集器通过发送“Hello”消息来收集网络中各节点的位置;然后,根据所收集到的位置信息,采用生成树覆盖算法从中找出部分位置,使得移动数据收集器只需到达这些位置即可收集到全网的数据;最后,移动数据收集器定期访问所找出的位置进行数据收集.但在大规模的 WSN 中,移动数据收集器需要访问大量的位置,使得数据收集延迟急速增长,难以适用于延迟敏感的应用.

Zhao 等人同时考虑了节能和数据收集延迟的问题,设计了均衡节点能量消耗和最小化数据收集延迟的数据收集协议 SPT-DGA[13].首先构造最短路径树,使得所有节点在最短路径树中;然后找到最短路径树中的最远叶子节点,沿着根节点的方向向上查找 h 跳,并将最后一跳节点选为轮询节点,其中,h 是一个正整数;最后,移动数据收集器定期地遍历所找到的轮询节点进行数据收集.与同类的数据收集协议相比,SPT-DGA[13]在节约节点能量和缩短数据收集延迟方面有很大的提升.但是,一方面,SPT-DGA[13]所找出的轮询节点数量仍然较多,使得数据收集延迟较大;另一方面,由于 SPT-DGA[13]算法局限于选取离 Sink 最近的节点为根节点,因此在网络拓扑或限高树的高度未发生变化的情况下,重新执行 SPT-DGA[13]算法所构造出来的最短路径树都是一样的,而且所选取的轮询节点(限高树的根节点)也不会发生变化.这样极易造成限高树的根节点能量消耗过快而死亡,从而缩短了网络生命周期.

本文借鉴 SPT-DGA[13]的思想,在 WSN 中引入一个移动数据收集器收集数据,提出一个节能省时的数据收集协议 DC-Collection.DC-Collection 算法主要对 SPT-DGA[13]算法进行了以下几个方面的改进.

(1) DC-Collection 算法构造最短路径树时,在静态 Sink 两跳范围内选取能量最高的节点作为最短路径树的根节点.如果在 Sink 的两跳范围内没有节点,则选取 3 跳范围内能量最高的节点作为最短路径树的根节点,以此类推.这样改进的好处有:(a) 同时兼顾了节点的能量水平和根节点的位置,既能避免选取能量低的节点作为根节点,又能避免根节点偏离 Sink 太远,有利于延长网络生命周期;(b) 在数据收集过程中,DC-Collection 算法能够根据当前网络情况动态选取能量高的节点作为最短路径树的根节点,从而使各节点能够在不同角色之间切换,有利于均衡节点的能量消耗;

(2) 在找到限高树的根节点后,DC-Collection 算法立即选取根节点的父节点为逗留节点;接着,在最短路径树上选取到达子孙节点的最大跳数不超过 h 且在逗留节点通信区域内的节点为采集节点,优先选取到达子孙节点的最大跳数小的节点为采集节点.此时,在逗留节点的通信区域内,多个采集节点形成以逗留节点为中心的缓存池.这样有利于减少逗留节点数量,使得移动数据收集器能以高概率获得较短的行走路径长度,从而缩短数据收集延迟;

(3) DC-Collection 算法对限高进行优化操作,使得能量大的节点承担更多的子孙节点.这样能够均衡节点的能量消耗,最大化限高树的网络生命周期,从而延长网络生命周期.

2 系统模型和问题描述

2.1 网络模型

在面积为 $A=M\times M$ 平方米的目标监视区域内随机部署一个 Sink 和 n 个节点,其中,符号 M 表示目标监视区域的宽度.整个传感器网络组成一个无向图 $G(V,E)$,图中可能包含一个或多个连通子图.其中,V 是 G 中节点的集合,E 是 G 中边的集合.网络中的 Sink 以及每一个节点用唯一的 ID 号进行标识,v_0 表示 Sink,$v_1,v_2,\ldots,v_{n-1},v_n$ 表示 ID 号分别为 $1,2,\ldots,n-1,n$ 的节点.因此,节点集合 $V=\{v_1,v_2,\ldots,v_{n-1},v_n\}$,$|V|=n$ 为节点集合的长度.如果两个节点 v_i 和 v_j 相互处于对方的通信半径内(即,它们能够相互通信),则边 $(v_i,v_j)\in E$.$|E|=e$ 为边的数量.在 WSN 中,节点感知的数据大小固定且不能与接收到的数据进行聚合.利用一个移动数据收集器在网络中收集数据.假定网络中有其他一些拥塞控制策略,以避免数据在传输过程中发生拥塞和重传.网络具有如下性质:

(1) Sink 和节点部署后不再移动;

(2) Sink 的能量充足且有足够的存储空间;

(3) 所有节点随机部署在目标监视区域,各节点的初始能量异构且不能补充;

(4) 移动数据收集器可以达到监视区域内的任何地方,且具有能量大、存储空间充足的特点;

(5) 移动数据收集器和节点的通信半径均为 r;

(6) 所有节点都可以在采集节点、逗留节点和普通节点这 3 种角色之间切换.特别地,Sink 属于逗留节点.

2.2 相关定义

定义 1(限高树). 在 WSN 中,构造多棵高度不超过 h 的树,称这些树为限高树.其中,h 是正整数.

定义 2(采集节点). 限高树的根节点称为采集节点.

定义 3(逗留节点). 在数据收集过程中,如果移动数据收集器移动到节点 a 所在位置进行数据收集,则称节点 a 为逗留节点.

定义 4(轮). 移动数据收集器遍历所有逗留节点所在位置来收集所有节点发送的数据,并且将收集到的数据发送给静态 Sink 的过程.

定义 5(节点的能量消耗). 在一轮数据收集中,节点 v_i 的能量消耗 $C(v_i)=KE_rD(v_i)+KE_t(D(v_i)+1)$.其中,$k$ 表示节点感知数据的大小,单位为 bit;E_r,E_t 分别表示接收和发送 1bit 数据所消耗的能量;$D(v_i)$表示节点 v_i 在树上的子孙节点数量.

定义 6(节点的存活轮数). 如果节点在第 m 轮数据收集后其剩余能量仍大于 0,在第 m+1 轮数据收集中其剩余能量小于 0,则称节点的存活轮数为 m.如果节点 v_i 的能量为 $E(v_i)$,则节点 v_i 在树中的存活轮数为

$$S_{node(v_i)}=\lfloor E(v_i)/C(v_i)\rfloor,$$

其中,$C(v_i)$为节点 v_i 在一轮数据收集中的能量消耗.

定义 7(限高树的生命周期). 在限高树 T 中,第 1 个节点死亡时的存活轮数称为限高树的生命周期.定义为

$$L_{tree}(T)=\min_{m\in V_T}\{S_{node(v_m)}\},$$

其中,$V_T=\{k_1,k_2,\ldots,k_t\}$为限高树 T 中节点编号的集合.

定义 8(网络生命周期). 在 WSN 中,如果节点 a 的存活轮数最小,则节点 a 的存活轮数为网络生命周期.

定义 9(通信区域). 以节点所在位置为圆心、r 为半径的圆所覆盖的区域.

定义 10(节点跳数). 在一棵树 T 中,节点到其子孙的最大跳数.

定义 11(约束跳数). 用户给定的一个系统参数 h,它用于约束网络中所构造的限高树的最大树高.即,限高树的最大树高不超过 h.

定义 12(最优限高树). 网络中的任意一棵限高树 T,它的最优限高树定义为

$$T_{opt}=\{T\mid L_{tree}(T)=\max_{T'\in T_s(V_T)}L_{tree}(T')\},$$

其中,$V_T=\{k_1,k_2,\ldots,k_t\}$为限高树 T 中节点编号的集合;r_t 为限高树 T 的根节点;T'是由 V_T 组成的限高树,且根节点为 r_t;T_s 是所有限高树 T'的集合,在 T_s 中,最优限高树 T_{opt} 拥有最大的网络生命周期.

定义 13(完全非汇聚). 在 WSN 中,各个节点感知的数据相关性不强,节点发送的数据等于节点自身感知的数据和接收到的数据之和.

2.3 问题描述

根据以往的研究,WSN 中数据包的发送速度大约是几百米每秒[14].然而在实际应用中,典型的移动系统的移动速度大约在 0.1~2m/s[14,15].在大规模的 WSN 中,如果移动数据收集器与网络中每个节点直接通信(即,网络中的节点以单跳的方式将感知到的数据发送给移动数据收集器),能够最大限度地保存节点的能量,但是数据收集延迟会急速增长.因此在实际应用中,特别是对于一些延迟敏感的应用场景,不仅需要考虑节点的能量消耗,而且还需要考虑数据收集延迟的问题.为均衡各节点的能量消耗、延长网络生命周期和最小化数据收集延迟,与文献[16,17]一样,本文引入了一个能量充足且存储空间大的移动数据收集器来收集各节点感知的数据.

本文要解决的关键问题是:如何从节点集合中选取部分节点作为逗留节点,使得移动数据收集器遍历这些逗留节点时,既能达到均衡节点能耗的目的,又能实现最小化数据收集延迟的目标.为了解决这个问题,本文提

出了 DC-Collection 算法.

2.4 问题建模

本文通过 DC-Collection 算法来解决最小化数据收集路径的问题,该问题可以归为整数线性规划问题.为了便于描述整数线性规划问题,定义了表 1 所示的符号及其含义.

Table 1 Notation used in formulation of promblem

表 1 问题建模所使用的符号

符号	含义
v_0	Sink
$V=\{v_1,v_2,\ldots,v_{n-1},v_n\}$	网络中所有节点的集合
h	约束跳数
d_{ij}	两个节点 v_i 和 v_j 之间的距离
$p_{ij}=\{0,1\}$	如果节点 v_i 和 v_j 之间的距离包含在数据收集路径中,则 $p_{ij}=1$;否则,$p_{ij}=0$
$t_{ij}=\{0,1\}$	如果节点 v_i 在以 v_j 为根的限高树上,则 $t_{ij}=1$;否则,$t_{ij}=0$
$c_i=\{0,1\}$	如果节点 v_i 是采集节点,则 $c_i=1$;否则,$c_i=0$
$s_i=\{0,1\}$	如果节点 v_i 是逗留节点,则 $s_i=1$;否则,$s_i=0$
$t_{ij}^l=\{0,1\}$	如果节点 v_i 在以 v_j 为根节点的限高树的第 l 层,则 $t_{ij}^l=1$;否则,$t_{ij}^l=0$
$ns(i)$	节点 v_i 的邻居节点集合

目标函数:

$$Minimize \sum_{i,j\in V\cup(v_o)} d_{ij}p_{ij} \tag{1}$$

条件约束:

$$\sum_{j\in V,i\neq j} t_{ij}+c_i+s_i=1,\forall_i\in V \tag{2}$$

$$c_i\times\sum_{j\in ns(i)} s_j\geqslant c_i,\forall_i\in V \tag{3}$$

$$\sum_{j\in V,i\neq j} t_{ij}^l\leqslant h,\forall_i\in V \tag{4}$$

表达式(1)定义了本文求解最短数据收集路径的目标函数,公式(2)~公式(4)是目标函数的约束条件.

等式(2)约束了网络中的节点或为限高树上的节点(包括采集节点)或为逗留节点.

特别地,Sink 属于逗留节点.

不等式(3)约束了在每个采集节点的通信区域内至少有一个逗留节点,确保移动数据收集器移动到逗留节点所在位置时,采集节点和逗留节点能够将感知和接收到的数据发送给移动数据收集器.

不等式(4)约束了任意一棵限高树的高度不能超过 h.

3 DC-Collection 算法的设计

3.1 算法的基本思想

在 WSN 中,如果访问的节点数量越少且越集中,则移动数据收集器收集一轮数据所行走的路径长度越短,数据收集延迟就越短.考虑到最短路径树不仅有利于构造限高树,而且有利于缩短数据收集延迟,所以本文提出的 DC-Collection 算法首先构造最短路径树集合;然后,针对最短路径树集合的每棵最短路径树,在最短路径树上截取限高树并选取逗留节点,这样可以避免最短路径树的根节点过快消耗能量而死亡;接着,将限高树添加到限高树集合,并将逗留节点添加到逗留节点集合.此外,考虑到在完全非汇聚的模式下收集数据,树上节点的能量消耗与其子孙节点的数量息息相关,算法将对限高树集合中的每一棵限高树进行优化操作,使得能量大的节点被赋予更多的子孙节点,从而均衡了节点的能量消耗;最后,针对由逗留节点构成的集合,利用 Christofides algorithm[18]计算数据收集路径 tour,移动数据收集器沿着路径 tour 行走,收集存在采集节点和逗留节点中的数据.移动数据收集器最终回到 Sink,并将数据发送给 Sink.此时,一轮数据收集结束.

下面详细描述利用 DC-Collection 算法实现均衡节点的能量消耗同时最小化数据收集路径,从而达到缩短数据收集延迟的目的.

3.2 算法描述

3.2.1 第 1 阶段构造最短路径树集合

在不考虑约束跳数的情况下,构造最短路径树有利于缩短数据收集延迟.由于网络中所有节点以及边所构成的无向图 G 可能含有一个或多个连通分量,故最短路径树的数量可能有多个.我们称这些最短路径树所构成的集合为 TS.为了构造最短路径树集合 TS,Sink 需要获取各节点的位置等信息.然而,如果网络中的节点所组成的网络是非连通的,与 Sink 不在同一连通网络中的节点无法将自身的位置和能量信息通过无线通信发送给 Sink,这样就无法构造全网中的最短路径树集合 TS.为了解决这一问题,节点部署后,可以利用一个或多个移动数据收集器收集全网节点的位置和能量信息[12].各节点可以通过定位技术(如 GPS)来获取自身所在位置,当移动数据收集器能与节点直接通信时,节点将位置和能量信息发送给移动数据收集器;然后,移动数据收集器将收集到节点的位置和能量信息发送给 Sink.下面将详细描述构造最短路径树集合 TS 的过程.

首先,在 Sink 两跳通信范围内选取能量水平最高的节点 *rootNode* 作为最短路径树的根节点.如果在 Sink 的两跳范围内没有节点,则选取 3 跳范围内能量最高的节点作为最短路径树的根节点,以此类推.这样不仅能够使得逗留节点的集合更加靠近 Sink,而且有利于均衡节点的能量消耗.设根节点为最短路径树上的第 0 层.一旦确定根节点,则采用文献[13]中的方法来构造最短路径树,直至与根节点在同一连通分量中的所有节点都加入到树中.同理,对于网络中的其他连通分量,Sink 采用同样的方法选取根节点并构造最短路径树.当网络中的所有节点都加入到最短路径时,构造最短路径树集合 TS 的算法结束.

3.2.2 第 2 阶段构造限高树集合和逗留节点集合

在同一网络拓扑中,如果采用文献[13]中的方法构造限高树集合,约束跳数 h 越大,则移动数据收集器需要访问的节点数量越少,从而使获得的数据收集路径长度较短的概率越大.这样,移动数据收集器能够以高概率在更小的延迟内完成数据收集.因此,本文借鉴文献[13]中的思想构造限高树集合.构造限高树集合的问题可以进一步划分为如下两个子问题.

1. 如何选取采集节点,使得以采集节点为根的限高树的树高小于或等于 h;
2. 在确保能够收集到采集节点发送的数据的情况下,如何选取移动数据收集器访问的位置,使得移动数据收集器访问的位置较少从而以高概率获得较短的数据收集延迟.

为简化构造限高树的描述过程,我们将从最短路径树集合中选取一棵最短路径树进行详细的描述.

关于第 1 个子问题:

- 首先,找到最短路径树上距离根节点最远的一个叶子节点 x;
- 然后从节点 x 出发,沿最短路径树的根节点方向查找 h 跳.在向上查找的过程中,可能有以下两种情况:
  - (a) 在查找的过程中,如果没有遇到最短路径树的根节点 *rootNode*,则向上查找 h 跳后停止查找操作,并且将最后一跳的节点 y 选为采集节点;
  - (b) 在查找的过程中,若遇到最短路径树的根节点 *rootNode*,则终止查找操作,此时,根节点 *rootNode* 被选为采集节点.这样,可以确保网络中任意一棵以采集节点为根的限高树的高度不超过 h.

在回答第 2 个子问题之前,我们先思考这么一个问题:收集限高树上所有节点的数据,通常的做法是利用移动数据收集器访问网络中所有的采集节点来完成数据收集.然而在大规模的 WSN 中,如果采用这种数据收集方式,则移动数据收集器很有可能需要访问网络中大部分的区域才能收集到全网节点发送的数据,这样势必延长数据收集延迟.那么如何选取移动数据收集器访问的路径,使得该路径较短从而缩短数据收集延迟呢?

本文提出一个低延迟的方案:

- 首先,根据第 1 个子问题的解决方法选出采集节点 a;
- 然后选取逗留节点,并在逗留节点的通信范围内选取采集节点.

在选取逗留节点及其通信范围内采集节点的过程中,可能有以下两种情况.

(a) 如果采集节点 *a* 是最短路径树的根节点 *rootNode*,则执行以下操作:选取根节点 *rootNode* 作为逗留节点,并将其孩子节点选为采集节点.同时,将采集节点及其子孙节点构成的所有限高树保存到限高树集合,并将根节点 *rootNode* 添加到逗留节点集合;

(b) 如果采集节点 *a* 不是最短路径树的根节点,则执行以下操作:首先,选取采集节点 *a* 的父节点 *b* 作为逗留节点,并将节点 *b* 加入到逗留节点集合;其次,将采集节点及其子孙节点组成的限高树保存到限高树集合,并将限高树从最短路径树上移除;再次,在逗留节点 *b* 的通信区域内选取节点跳数不超过 *h* 且在最短路径树上的节点作为新的采集节点;最后,将新的采集节点及其子孙节点组成的限高树保存到限高树集合,并将限高树从最短路径树上移除.采用迭代的方式执行上述过程,直至所有节点从最短路径树上移除.此时,构造限高树的算法结束.需要注意的是:在逗留节点 *b* 的通信区域内,节点跳数不超过 *h* 且在最短路径树上的节点可能有多个.为了均衡各节点的能量消耗,延长网络生命周期,在逗留节点 *b* 的通信范围内选取采集节点时,优先选取节点跳数较小的节点作为新的采集节点.此外,由于节点的移除会引起最短路径树上节点的属性信息发生变化,属性信息包括节点的子孙节点、节点跳数以及在树中的层次等,因此在执行查找逗留节点通信区域内采集节点的操作之前,需要更新在逗留节点通信区域内且在最短路径树上节点的信息.

本文与文献[13]构造限高树的主要区别在于:本文选取采集节点后需进一步选取逗留节点,并在逗留节点的通信范围内选取采集节点.选取逗留节点的操作比较简单,就是选取采集节点的父节点作为逗留节点.接下来,本文将重点描述在逗留节点的通信区域内寻找采集节点的算法.具体描述见算法 1.

算法 1. *FindCollectingNodes*(*node*,*visitNode*,*hopCount*) //在逗留节点周边查找满足条件的采集节点.

```
1. if (~isempty(node(visitNode).neighbor))   //逗留节点的邻居节点不为空
2.     stack=node(visitNode).neighbor;
3.     while (~isempty(stack))
4.         a=stack(end);
5.         将最短路径树上的邻居节点 a 添加到候选节点集合 candidateNodes 中;
6.       stack=stack(1:end−1);
7.     end
8.     if (~ismember(candidateNodes))
9.         m=1;
10.        while (m≤length(candidateNodes))
11.            从 candidateNodes 选取到达节点跳数最短的节点 b;
12.            if (node(b).farLeaf≤hopCount&&node(b).tree~=1);
13.                collectingNodesSet=[collectingNodesSet,b];
14.                更新以 b 为根节点树的信息(如子孙、最远叶子节点、存活轮数等);
15.            end
16.            candidateNodes=setdiff(candidateNodes,b);
17.        end
18.    end
19. end
```

3.2.3 第 3 阶段优化限高树

在完全非汇聚的网络模式下进行数据收集,节点的能量消耗主要与其子孙节点的数量相关.然而,第 2 阶段构造限高树集合的操作完成后,各节点的属性信息发生了变化.此时,能量高的节点可能拥有的子孙节点数量较少;相反,能量低的节点可能拥有较多的子孙节点数量.这样不利于延长网络生命周期.为了最大化网络生命周期,需要对限高树集合中的每一棵限高树进行优化操作.优化限高树的问题可以进一步划分为以下几个子问题.

(1) 哪些节点需要优化?

(2) 如何优化?

关于第 1 个子问题,对于固定发射和接收功率的节点,Sink 可以根据节点的子孙节点的数量计算它在限高树中的存活轮数.显然,所有节点中最小存活轮数即为这棵限高树的生命周期.我们定义这棵限高树上存活轮数最小的节点为瓶颈节点.如果瓶颈节点无法进行优化,即使限高树上的其他节点能够进行优化操作,也无助于提高限高树的网络生命周期.因此,瓶颈节点是优化的首选节点.需要注意的是:由于优化后限高树的结构会发生变化,因此每个节点的存活轮数以及相应的信息会发生变化,如节点在限高树中的层次、孩子数量等.为了保证优化后的限高树的高度不超过 h,优化限高树的算法是迭代执行的,且每次迭代只对瓶颈节点进行优化.

关于第 2 个子问题,为最大化限高树的网络生命周期,需要将瓶颈节点的子孙节点转移到其他非瓶颈阶段上,以减轻瓶颈节点的负担,增加网络生命周期.对于任意一个瓶颈节点的子孙节点 a,它能够成功转移需要满足以下 4 个条件.

(1) 在节点 a 的通信区域内存在一个非瓶颈节点 b;

(2) 节点 a 和节点 b 在同一棵限高树上;

(3) 节点 a 及其子孙节点转移到节点 b 后,限高树的高度不能超过 h;

(4) 节点 a 及其子孙节点转移到节点 b 后,限高树的生命周期必须大于它们转移之前的网络生命周期.

满足以上条件的非瓶颈节点可能不止一个,为了最大化限高树的网络生命周期,必须选择能够获得最大网络生命周期的非瓶颈节点作为目标节点进行转移.每次转移操作成功后,均更新限高树上各节点的属性信息.采用迭代的方式查找目标节点并完成转移操作.如果找不到满足以上条件的节点,则终止优化限高树的操作.算法的具体描述见算法 2.

算法 2. *OptimizeLimitedTree*(*node*,*p*,*TDe*,*RDe*,*h*).

```
1. 更新树的信息并返回集合 survivorSetSort;
2. while (~isempty(survivorSetSort))
3.     选取存活轮数最小的节点 a;
4.     transferFlag=0;   //转移标志;0 表示不能进行转移,1 表示可以进行转移
5.     if (~isempty(node(a).descendant))
6.         for i=1:length(node(a).descendant)
7.             b=node(a).descendant(i);
8.             如果节点 c 满足转移条件,则将节点 c 加入候选节点集合 candidateNode 中;
9.         end
10.        if (~isempty(candidateNode))
11.            在 candidateNode 中选取节点 d 作为目标节点,使得节点 b 成功转移后,新的限高树拥有最大网
               络生命周期;
12.            node(d).children=union(node(d).children,b);   //转移到目标节点
13.            node(b).parent=d;
14.            node(a).children=setdiff(node(a).children,b);   //移除节点
15.            更新树的信息并返回集合 survivorSetSort;
16.            transferFlag=1;
17.        end
18.    end
19.    if (transferFlag==0)   //无法将节点 a 的孩子节点进行转移,停止优化限高树的操作.
20.        break;
21.    end
```

22. end

3.2.4 第 4 阶段计算数据收集路径

根据第 2 阶段获得的逗留节点集合来规划移动数据收集器的行走路径,以便移动数据收集器以尽可能小的延迟完成数据收集.显然,针对逗留节点集合,求解一条最短回路是典型的旅行商问题(traveling salesman problem,简称 TSP).到目前为止,求解 TSP 问题的最好算法是 Christofides algorithm[18],它的近似率达到 3/2.为了让移动数据收集器行走的路径尽可能短从而缩短数据收集延迟,本文采用了 Christofides algorithm[18]计算 TSP 的路径.Christofides algorithm[18]的具体实现过程如下.

- 首先,在完全图 $G'(V',E')$的基础上构造一棵最短路径树 T',其中,V'是所有逗留节点和 Sink 组成的集合,E'是 V'中各节点之间的欧几里德距离集合;
- 其次,在 V^0 中找到权值最小的完全匹配,其中,V^0 表示 T'中度为奇数的节点集合;
- 再次,在前两步的基础上,计算以 Sink 为起点和终点的欧拉回路;
- 最后,移除欧拉回路中重复的节点,即可形成哈密顿回路.哈密顿回路就是求解的数据收集路径 tour.

移动数据收集器从 Sink 出发,沿着数据收集路径 tour 行走,最终回到 Sink,并将数据发送给 Sink 来完成数据收集.

3.3 DC-Collection算法的执行

- 网络部署的初始阶段

为了选取采集节点、逗留节点以及构造最优限高树,Sink 需要收集到所有节点的分布信息.分布信息包括节点的位置、能量、邻居节点等.一方面,由于在网络部署的初始阶段节点是随机部署的,Sink 并不知道每个节点的分布信息;另一方面,因为节点也不知道到达 Sink 的路径,所以节点也无法将自身的分布信息准确地发送给 Sink.比较典型的做法是:构造一棵以 Sink 为根的层次结构树;接着,通过可靠的数据传送机制来收集分布信息,如逐跳确认机制[19].然而,这种方法只适用于网络连通的情况.若网络是非连通的,与 Sink 不在同一连通网络中的节点无法通过无线通信将分布信息发送给 Sink,这样就无法构造最优限高树.为了解决该问题,节点部署后:首先,利用移动数据收集器获取全网节点的分布信息[12];然后,移动数据收集器将分布信息发送到 Sink,Sink 通过执行 DC-Collection 算法可以得到限高树集合和逗留节点集合,进而得到最优限高树集合以及数据收集路径的相关信息;最后,利用移动数据收集器将这些相关信息发送给网络中的所有节点.这样,网络中的所有节点构成一个 3 层的数据收集拓扑:最高层是 Sink,中间层是采集节点和逗留节点,最底层是普通节点.此时,网络部署的初始阶段结束并且进入数据收集阶段.

- 数据收集阶段

移动数据收集器从静态 Sink 节点出发,沿着数据收集路径行走.移动数据收集器在行走的过程中发送数据请求消息.节点只有在首次接收到数据请求消息时,才将数据请求消息广播给它的邻居节点,否则丢弃该数据请求消息.这样能够避免节点多次接收并广播同一数据请求消息,大大降低了网络中的消息通信量,从而保存节点的能量.当限高树上的节点收集到数据请求消息并且收集到所有孩子节点发送的数据后,将自身感知的数据和接收的数据发送到父节点.以此类推,直至采集节点接收到所在限高树上所有子孙节点发送的数据.当移动数据收集器行走到逗留节点所在位置时,逗留节点将自身感知的数据发送给移动数据收集器.与此同时,在移动数据收集器通信区域内的采集节点将自身感知和收集到的数据发送给移动数据收集器.移动数据收集器沿着数据收集路径行走,遍历所有逗留节点所在位置并收集逗留节点和采集节点发送的数据,最终返回静态 Sink 并将收集到的所有数据发送给 Sink.此时,一轮数据收集结束.

在数据收集过程中,由于采集节点需要收集其子孙节点感知的数据,导致采集节点需要消耗的能量远比其他节点要多,使得这些采集节点成为影响网络生命周期的关键因素.因此,需要重新执行 DC-Collection 算法,使得节点能够在不同角色之间切换,以便均衡节点的能量消耗从而延长网络生命周期.此外,由于 Sink 在执行 DC-Collection 算法之前需要获取全网中节点的分布信息,整个过程需要消耗较多的能量用于节点间的通信.为了进一步减少消息开销并且保存节点的能量,我们采用局部更新和全局更新的机制进行处理.具体实现过程如下:以

采集节点开始收集数据时的能量为起点,当采集节点的能量消耗一半时,以采集节点为根的限高树开始执行优化限高树的操作;限高树完成优化操作后,开始新一轮的数据收集.这就是局部更新操作.当网络中的所有采集节点都执行限高树的优化操作后,重新执行DC-Collection算法,以便节点在不同角色之间切换.这样有利于均衡节点的能量消耗.这就是所谓的全局更新.尽管局部更新和全局更新的操作会增加一些额外的开销,但是我们充分考虑了能量的动态性.在数据收集过程中,根据采集节点能量消耗的具体情况进行局部更新和全局更新操作,这样能够有效地延长网络生命周期,其付出的代价还是值得的.

3.4 DC-Collection算法的执行实例

为了更好地理解DC-Collection算法,下面以图1为例对DC-Collection算法的执行过程进行详细的描述.

假设在WSN中有30个节点和一个位于监视区域中心的Sink,Sink节点已获得所有节点的分布信息且构造了一棵最短路径树,如图1(a)所示.设约束跳数 h=2.

在图1(a)中,由于在Sink两跳通信范围内能量水平最高的节点编号是18,故选取节点18作为最短路径树的根节点.最短路径树中的最远叶子节点为5,20,26,它们到达根节点18的跳数为5.对于节点5,沿着节点5向根节点的方向查找2跳,找到节点28.选取节点28为采集节点以及节点28的父节点16为逗留节点.将以采集节点28及其子孙节点组成的限高树从最短路径树中移除.以逗留节点16为中心,在其通信范围内的节点为节点2、节点8、节点21、节点28.由于节点8、节点21在最短路径树中,且它们的节点跳数都没有超过约束跳数2,故将节点8、节点21选取为采集节点,并将节点8、节点21以及对应的子孙节点从最短路径树中移除.尽管节点2在逗留节点16的通信范围内,但是节点2的节点跳数3大于约束跳数2,故其不能选取为采集节点.选取采集节点以及逗留节点的第1次迭代操作结束,得到图1(b)所示的结果.

在图1(b)中,移除部分节点和相应边后的最短路径树中的最远叶子节点为节点20、节点26,它们到达根节点18的跳数为5.对于节点20,沿着节点20向根节点的方向查找2跳,找到节点30.选取节点30为采集节点以及节点30的父节点4为逗留节点.将以采集节点30及其子孙节点组成的限高树从最短路径树中移除.以逗留节点4为中心,在其通信范围内的节点为3,11,24,30.由于节点3、节点11在最短路径树中,且它们的节点跳数都没有超过约束跳数2,故将节点3、节点11选取为采集节点,并将节点3、节点11以及对应的子孙节点从最短路径树中移除.此时,节点24的节点跳数由3变成1,故将其选为采集节点并从最短路径树中移除.选取采集节点以及逗留节点的第2次迭代操作结束,得到图1(c)所示的结果.

在图1(c)中,移除部分节点后的最短路径树中的最远叶子节点为1,它到达根节点18的跳数为4.对于节点1,沿着节点1向根节点的方向查找2跳,找到节点12.选取节点12为采集节点以及节点12的父节点2为逗留节点.将以采集节点12及其子孙节点组成的限高树从最短路径树中移除.以逗留节点2为中心,在其通信范围内的节点为12,16,18.由于节点12、节点16已不在最短路径树中,当前节点18的节点跳数为3,故它们不选取为采集节点.此时,将逗留节点2从最短路径树中移除.选取采集节点以及逗留节点的第3次迭代操作结束,得到图1(d)所示的结果.

在图1(d)中,移除部分节点后的最短路径树中的最远叶子节点为7,19,27,它们到达根节点18的跳数为3.对于节点7,沿着节点7向根节点的方向查找2跳,找到节点22.选取节点22为采集节点以及节点22的父节点18为逗留节点.将以采集节点22及其子孙节点组成的限高树从最短路径树中移除.由于选取的逗留节点18为最短路径树的根节点,故所有节点已从最短路径树上移除.构造限高树的算法结束,得到图1(e)所示的结果.

在图1(e)中,对限高树进行优化操作.由于在以节点3为根的限高树中,节点25是瓶颈节点.通过优化限高树的操作,将节点6转移到节点14下面,延长了限高树的生命周期,得到如图1(f)所示的结果.采用迭代的方法对所有限高树进行优化操作,得到如图1(g)所示的结果.

针对由逗留节点和Sink组成的集合,采用Christofides algorithm[18]计算起点和终点都为Sink的数据收集路径tour.如图1(h)所示,带有箭头的加粗线段表示移动数据收集器在数据收集过程中行走的方向.

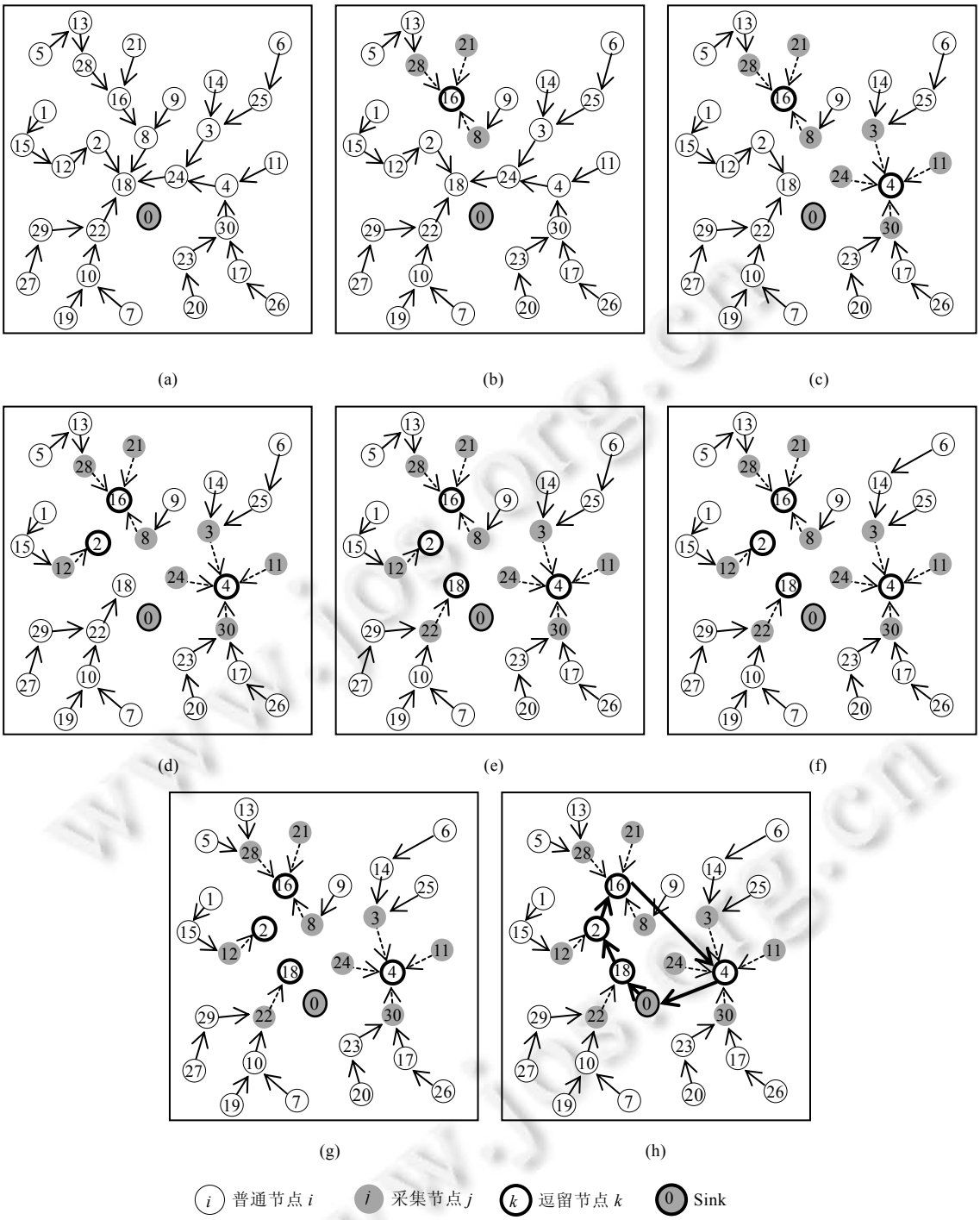


Fig.1 An implementation instance of the DC-Collection algorithm

图 1 DC-Collection 算法的实现实例

4 算法性能分析

4.1 优化限高树对网络性能的影响

在第 2 阶段完成构造限高树后,为延长网络生命周期,对限高树进行优化操作.然而,通过大量的仿真实验,其结果表明:如果网络部署后不再重新执行 DC-Collection 算法,则优化限高树的操作对延长网络生命周期的影响不大.主要原因有以下几点.

(1) 执行完第 2 阶段构造限高树的算法后,网络中的节点或在以采集节点为根的限高树上或为逗留节点.由于本文考虑的是在完全非汇聚的网络中进行数据收集,因此节点的能量消耗与其子孙节点的数量密切相关且是成正比的关系.此外,在每棵限高树中,根节点的子孙数量最多.所以,根节点消耗的能量远比其子孙节点消耗的能量要多得多.这样就极易造成根节点的能量消耗过大,使得根节点具有较小的存活轮数.然而,网络中存活轮数最小的节点,其存活轮数决定了网络生命周期.因此,采集节点(限高树的根节点)的存活轮数在很大程度上决定了网络生命周期;

(2) 采集节点和逗留节点的选取是相互关联的,在优化限高树的过程中,不能更改限高树的根节点(采集节点).由于移动数据收集器在收集数据的过程中只遍历逗留节点所在位置进行数据收集,所以只有在逗留节点通信区域内的采集节点才能将数据发送给移动数据收集器.为了确保移动数据收集遍历所有逗留节点所在位置后能够收集到网络中所有节点发送的数据,在优化限高树的过程中,不能改变限高树的根节点(采集节点),只能对根节点的子孙节点进行转移.这个因素在很大程度上制约了优化限高树,从而达到延长网络生命周期的目的.

4.2 DC-Collection问题的难度

定理 1. DC-Collection 问题是 NP-hard 问题.

证明:由于 TSP 问题是典型的 NP-hard 问题,因此只要能将 TSP 问题等价转换成 DC-Collection 问题的一个特例,我们就可以证明 DC-Collection 问题是 NP-hard 问题.下面我们通过一个多项式时间的归约来证明.

- 首先,给定一个 TSP 的实例无向图 $G(V,E)$;
- 然后,构造一个拓扑结构与 G 一样的无向图 $G'(V',E')$,并将无向图 $G'(V',E')$作为 DC-Collection 的一个实例.其中,V'表示网络中所有传感器节点和 Sink,E'表示任意两个节点之间的边.

我们可以不断减小节点的通信半径,当节点的通信半径足够小时,网络中的任意两个节点无法直接相互通信.减少节点通信半径,使得各节点无法直接通信的操作非常直观,而且能够在多项式的时间内完成.在网络中的所有节点无法直接相互通信的情况下,各节点感知的数据都无法通过其他节点转发到 Sink.移动数据收集器必须访问所有节点才能完成一轮的数据收集,这就意味着所有节点和 Sink 都是逗留节点.此时,图 G'中数据收集路径长度与图 G 中 TSP 的总权值是相等的.当且仅当在图 G'中 DC-Collection 的数据收集路径长度最小时,图 G 中的 TSP 有最小距离代价的路径.因此,DC-Collection 问题是 NP-hard 问题.证毕. □

由于 DC-Collection 问题是 NP-hard 的,所以本文采用启发式算法来求解.

4.3 算法的时间复杂度分析

定理 2. 在最坏情况下,DC-Collection 算法的时间复杂度为 $O(n^3)$.

证明:下面我们将根据 DC-Collection 算法执行的 4 个阶段详细分析 DC-Collection 算法的时间复杂度.

- DC-Collection 算法的第 1 阶段,在具有多个连通分量的网络中构造最短路径树集合的时间复杂度为[13]

$$\sum_{k=1}^{t} O(n_k^2),$$

其中,t 表示网络中连通分量的数量且 $1\leqslant t\leqslant n$,k 表示第 k 个网络中连通分量,n_k 表示第 k 个连通分量中的节点数量.显然,第 1 阶段的最坏时间复杂度为 $O(n^2)$.

- DC-Collection 算法的第 2 阶段,在具有 n_k 个节点的最短路径树中选取采集节点的时间复杂度为[13]

$$O(n_k^2 + h \times n_k).$$

而从算法 1 易知:在逗留节点的通信区域内,寻找采集节点所需的时间复杂度是 $O(n)$.因此,在 t 棵最短路径树上,构造树高为 h 的限高树的时间复杂度为

$$\sum_{k=1}^{t}(O(n_k^2 + h \times n_k) \times O(n)),$$

其中,n_k 表示第 k 个最短路径树中的节点数量.显然,构造限高树的最差时间复杂度为 $O(n^3)$.

- DC-Collection 算法的第 3 阶段,优化限高树.

算法 2 的第 1 步~第 3 步是节点更新信息并选取存活次数最小的节点,第 4 步~第 22 步是将瓶颈节点的子孙转移到满足转移条件的节点,最大化限高树的网络生命周期.所以,优化限高树所需的时间复杂度是 $O(n^2)$.

- DC-Collection 算法的第 4 阶段,采用 Christofides algorithm[18]计算数据收集路径长度.

在最坏的情况下,网络中的所有节点都不能与其他节点进行通信.此时,网络中的所有节点将在数据收集路径上.由于 Christofides algorithm 最坏情况下的时间复杂度是 $3/2\times O(n^3)$,所以在最坏情况下,计算数据收集路径长度所需的时间复杂度是 $O(n^3)$.

综上所述,在最坏情况下,DC-Collection 算法的时间复杂度为 $O(n^2)+O(n^3)+O(n^2)+O(n^3)$.即,整个算法的时间复杂度为 $O(n^3)$.证毕. □

4.4 约束跳数和延迟之间的关系

定理 3. 在 WSN 中,约束跳数越大,DC-Collection 算法所选取的逗留节点数越少.这样,移动数据收集器获得较短数据收集路径长度的概率越大,使得移动数据收集器能够以高概率获得更短的数据收集延迟.

证明:假设目标监视区域的面积为 $M\times M$,节点的通信半径为 r.易知,目标监视区域内两个对角顶点的距离最大且该距离为 $\sqrt{2}\times M$.由此可以计算得出,网络中的节点至多经过 $k=\left\lceil \sqrt{2}\times M/r \right\rceil$ 跳即可将数据转发给根节点.此外,我们定义:如果节点能够与根节点之间通信,则称该节点为 1 跳节点;如果节点至少需要通过 2 跳才能将其数据转发到根节点,则称该节点为 2 跳节点.依次类推.如图 2 所示,最右边的 d_0 为最短路径树的根节点,d_i 为 i 跳节点($1\leqslant i\leqslant k$).

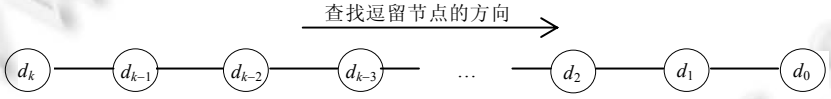


Fig.2 An analytic instance of the relationship between constrained hop-numbers and delay

图 2 约束跳数和延迟关系的分析图例

在利用移动数据收集器收集数据的 WSN 中,移动数据收集器收集一轮数据所需要的时间(即延迟)包括两部分:一是各采集节点收集数据的时间;二是移动数据收集器按照规划好的路径行走所需时间.然而根据以往的研究,WSN 中数据包的发送速度大约是几百米每秒[14],而在实际应用中,典型的移动系统的移动速度大约在 0.1~2m/s[14,15],故在数据收集过程中,可以忽略约束跳数对延迟的影响.因此,本文的延迟主要由数据收集路径长度决定.

为了便于描述约束跳数与延迟的关系,我们不妨先假设最短路径树的拓扑结构如图 2 所示.显然,当约束跳数 h 增大时,所选取的逗留节点越靠近根节点 d_0;而且选取的逗留节点数量也越少.推广到一般情形,网络中通常存在多个 d_i 跳节点被选为逗留节点.为简化证明过程,不妨假设这些处于 d_i 跳的逗留节点之间的路径长度是以 $d_i\times r$(根节点与 d_i 跳逗留节点之间的距离)为半径的圆周长,即 $L_i=2\pi\times(d_i\times r)$,且数据收集路径长度为 $\sum_{i=1}^{k}L_i\times P_i$.其中,$p_i$ 的取值为 0 或 1,且 $p_i=1$ 表示在 d_i 跳有逗留节点,$p_i=0$ 表示在 d_i 跳没有逗留节点.显然,数据收集路径长度会随着约束跳数的增加而减少.

综上所述,在移动数据收集器移动速度固定的情况下,约束跳数越大,算法所选取的逗留节点数越少.这样,移动数据收集器获得较短数据收集路径长度的概率越大,使得移动数据收集器能够以高概率获得更短的数据

收集延迟.证毕. □

5 仿真实验与结果分析

本文所提出的 DC-Collection 算法主要关注于如何解决网络中数据传输路由的选择和建立问题.此外,我们假设 MAC 层有可靠的包传输机制,网络层的丢包和拥塞控制问题可以得到较好的解决.由于 MATLAB 能够快速建立可视化的原型系统并进行算法性能分析,满足我们的实验仿真要求,所以我们选择在 MATLAB 平台上进行仿真实验.

约束跳数、通信半径以及节点密度是影响算法性能(即,网络生命周期和数据收集路径长度)的主要因素,为了评估和分析本文所提出的 DC-Collection 算法的性能,本节将对提出的 DC-Collection 算法与 SPT-DGA[13]算法和 LBCDG[11]算法进行一系列的仿真实验,进而验证 DC-Collection 算法的性能.由于节点感知数据能耗和计算能耗可以忽略不计[20],此外,在每轮数据收集过程中,节点只接收并广播发送一次数据请求消息,这些数据请求消息小且数量不大,故其消耗的能量也可以忽略不计,所以在 WSN 中,数据包的发送和接收所消耗的能量成为主要的通信能耗.为使 DC-Collection 算法、SPT-DGA[13]算法以及 LBCDG[11]算法能够进行公平的对比,节点采用固定发射功率.节点进行通信时,发送数据所消耗的能量大约是接收数据所消耗能量的 2 倍[21].此外,在 LBCDG[11]算法中,节点直接将数据发送给簇头.为测量约束跳数 h 对网络性能的影响以及进行公平的实验对比,我们对 LBCDG[11]算法进行拓展,使得节点能够通过多跳的方式将数据转发给簇头.仿真实验中所使用的部分实验参数见表 2.节点的能量消耗主要用于发送数据和接收数据,所以在整个数据收集过程只考虑节点发送和接收数据的能量消耗.实验的结果均是算法执行 20 次后的平均结果

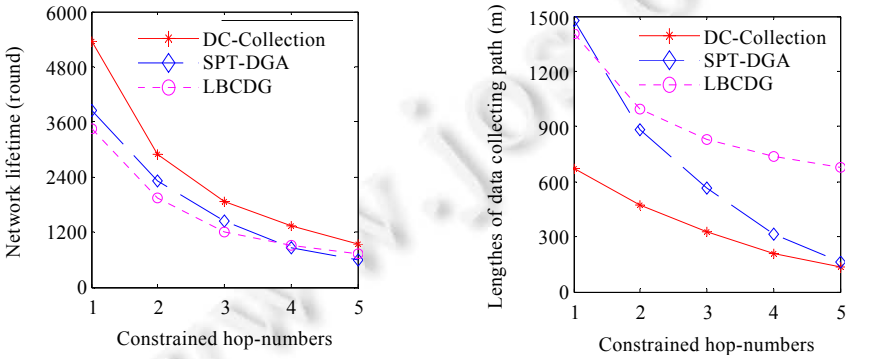
Table 2 Experimental paramaters

表 2 实验参数

目标监视区域的面积	200×200m^2
节点的初始能量	[0.5,1]J
数据包的大小	128bit
发送 1bit 数据的能量消耗	100nJ/bit
接收 1bit 数据的能量消耗	50nJ/bit

5.1 约束跳数对网络生命周期和数据收集路径长度的影响

本小节将在一个随机确定的网络实例中,测试约束跳数 h 在不同取值时对 DC-Collection 算法、SPT-DGA[13]算法以及 LCDG[11]算法性能的影响.假设在目标监视区域内随机部署 200 个节点,节点的通信半径为 30m,其他实验参数见表 2.我们分别测试约束跳数 h 为 1,2,3,4,5 时,DC-Collection 算法、SPT-DGA[13]算法以及 LBCDG[11]算法的网络生命周期和数据收集路径长度.实验结果如图 3(a)、图 3(b)所示.



(a) 网络生命周期随约束跳数变化的场景 (b) 数据收集路径长度随约束跳数变化的场景

Fig.3 Scene with different constrained hop-numbers

图 3 约束跳数变化的场景

从图 3(a)可知:随着约束跳数 h 的不断增大,各算法的网络生命周期都在不断减小,且 DC-Collection 算法获得较大的网络生命周期.主要有两个方面的原因.

(1) 随着约束跳数 h 的增大,限高树上的节点数量也随之增加,导致节点的能量消耗增加,进而缩短了网络生命周期;

(2) 在 DC-Collection 算法中,在逗留节点通信区域内所选取的采集节点,其子孙节点到达采集节点的跳数一般小于 h,使得限高树上的节点数量较少,而且 DC-Collection 算法对限高树进行优化,有利于均衡节点的能量消耗;然而 SPT-DGA[13]算法没有进行优化,而且在 LBCDG[11]算法中节点随机选取簇头转发数据,没有考虑邻居节点的能量,不利于均衡节点的能量消耗.

以上这些因素,使得 DC-Collection 算法获得比 SPT-DGA[13]算法以及 LBCDG[11]算法更大的网络生命周期.

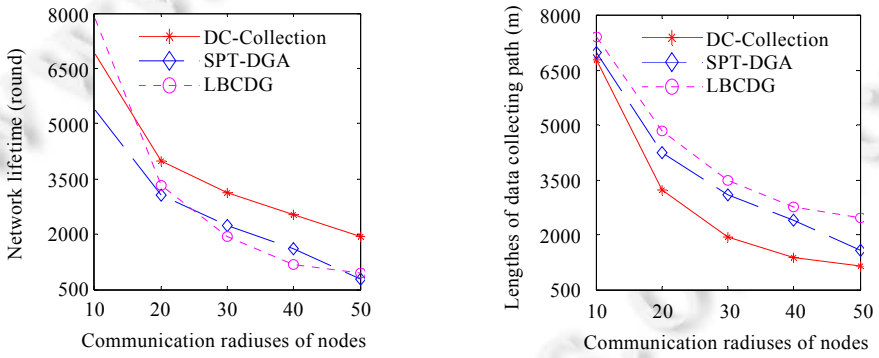
从图 3(b)可知:随着约束跳数 h 的不断增大,各算法的数据收集路径长度都在不断减小,且 DC-Collection 算法获得较短的数据收集路径长度.主要有两个方面的原因.

(1) 随着约束跳数 h 的增大,各算法所选取的逗留节点数量不断减少,有利于缩短数据收集路径长度;

(2) 在 DC-Collection 算法中,在逗留节点通信区域内选取部分节点为采集节点,使得更多的节点从最短路径树上移除.这样,DC-Collection 算法选取的逗留节点的数量比 SPT-DGA[13]算法和 LBCDG[11]算法更少,从而获得较短的数据收集路径长度.

5.2 通信半径对网络生命周期和数据收集路径长度的影响

本小节将在目标监视区域的面积、约束跳数、节点数量固定的条件下,测试在节点的通信半径发生变化时,网络生命周期和数据收集路径长度的变化情况.假设约束跳数 h=2,节点的数量 n=200,且在目标监视区域内随机部署,其他实验参数见表 2.我们在网络中分别测试节点的通信半径为 10,20,30,40 和 50m 时,DC-Collection 算法、SPT-DGA[13]算法以及 LBCDG[11]算法的网络生命周期和数据收集路径长度.实验结果如图 4(a)、图 4(b)所示.



(a) 网络生命周期随节点的通信半径变化的场景 (b) 数据收集路径长度随节点的通信半径变化的场景

Fig.4 Scene with different communication radiuses of nodes

图 4 节点通信半径变化的场景

从图 4(a)可知:节点的通信半径在 10m 时,LBCDG[11]算法获得了较高的网络生命周期;但当节点的通信半径在 20m~50m 之间不断增大时,DC-Collection 算法获得了比 SPT_DGA[13]算法和 LBCDG[11]算法更大的网络生命周期.主要有 3 个原因.

(1) 随着节点的通信半径的增大,DC-Collection 算法和 SPT-DGA[13]算法所构造的限高树以及 LBCDG[11]构造簇中的节点数量往往会随之增加.这样,根节点(采集节点)或簇头需要转发较多的数据,从而缩短了各算法的网络生命周期;

(2) 与 SPT-DGA[13]算法相比,DC-Collection 算法在逗留节点的通信范围内选取较多节点作为采集节点,且

以这些采集节点为根的限高树通常拥有较少节点,有利于保存节点的能量.此外,DC-Collection 算法的第 3 阶段对限高树进行了优化操作,有利于延长网络生命周期,最终使得 DC-Collection 算法能够获得比 SPT-DGA[13]算法更长的网络生命周期;

(3) 对于 LBCDG[11]算法,节点在约束跳数(h=2)范围内随机选取一个簇头并将数据转发给它.节点的通信半径越大,簇头通常需要转发的数据越多,而且节点转发数据时没有考虑邻居节点的能量水平,容易造成节点的能量消耗不均衡,从而缩短了网络生命周期.

以上这些因素,使得 DC-Collection 算法获得了比 SPT-DGA[13]算法和 LBCDG[11]算法更大的网络生命周期.

从图 4(b)可以看出:随着节点通信半径的不断增大,各算法的数据收集路径长度均在不断减少,且 DC-Collection 算法获得了比 SPT-DGA[13]算法及 LBCDG[11]算法更短的数据收集路径长度.主要有两个方面的原因.

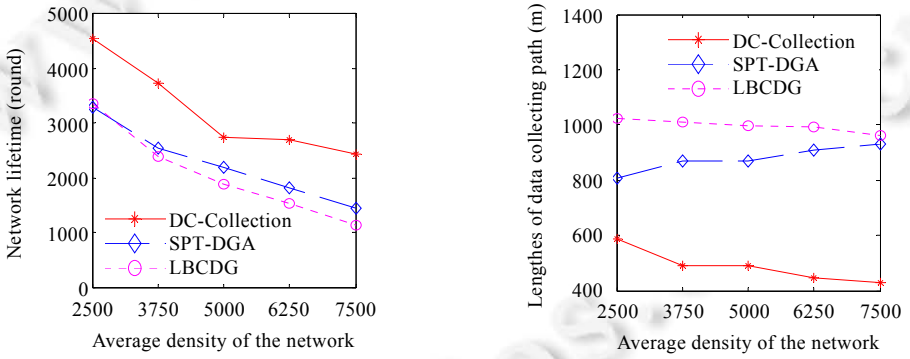
(1) 随着节点通信半径的增大,各算法所选取的逗留节点数量不断减少,使得各算法能够以高概率获得较短的数据收集路径长度;

(2) 在 DC-Collection 算法中,随着节点通信半径的不断增大,在逗留节点内能够选取更多节点作为采集节点,使得更多的节点从最短路径树上移除.

与 SPT-DGA[13]算法和 LBCDG[11]算法相比,DC-Collection 算法所选取的逗留节点数量要少得多,使得 DC-Collection 算法能够以高概率获得比 SPT-DGA[13]算法和 LBCDG[11]算法更短的数据收集路径长度.

5.3 网络平均密度对网络生命周期和数据收集路径长度的影响

在固定面积的目标监视区域内,网络平均密度与网络中节点数量成正比.所谓的网络平均密度是指节点数量与目标监视区域面积的比值.本节将在目标监视区域的面积、约束跳数、节点的通信半径固定的条件下,通过改变节点数量获得不同网络的平均密度,并在不同网络平均密度下,测试网络生命周期和数据收集路径长度的变化情况.假设约束跳数 h=2,节点的通信半径 r=30m,节点在目标监视区域内随机部署,其他实验参数见表 2.我们分别测试网络平均密度为 2 500,3 750,5 000,6 250 和 7 500 个/km^2 时,DC-Collection 算法、SPT-DGA[13]算法以及 LBCDG[11]算法的网络生命周期和数据收集路径长度.实验结果如图 5(a)、图 5(b)所示.



(a) 网络生命周期随网络平均密度变化的场景 (b) 数据收集路径长度随网络平均密度变化的场景

Fig.5 Scene with different average density of the network

图 5 网络平均密度变化的场景

由图 5(a)可知:随着网络平均密度的不断增大,各算法的网络生命周期均在不断地减小,且 DC-Collection 算法获得了比 SPT-DGA[13]算法和 LBCDG[11]算法更大的网络生命周期.主要有 3 个原因.

(1) 随着网络平均密度的增大,单位面积内的节点数量通常会增多.这样,DC-Collection 算法和 SPT-DGA[13]算法所构造的限高树以及 LBCDG 算法[11]构造簇中的节点数量往往会随网络平均密度的增大而增加,使得根节点(采集节点)或簇头需要转发较多的数据,从而缩短了各算法的网络生命周期;

(2) SPT-DGA[13]算法没有对限高树进行优化.此外,对于 LBCDG[11]算法,节点在约束跳数(h=2)范围内随机

选取一个簇头,并将数据转发给它.这样,能量较低的簇头可能转发较多数据而过快耗尽能量,缩短了网络生命周期.然而,DC-Collection 算法所构造的限高树的高度通常小于 h,且对限高树进行优化操作,有利于延长网络生命周期.

以上这些因素,使得 DC-Collection 算法获得比 SPT-DGA[13]算法以及 LBCDG[11]算法更大的网络生命周期.

由图 5(b)可知:随着网络平均密度的增加,各算法的数据收集路径长度均在不断地减少,且 DC-Collection 算法获得了比 SPT-DGA[13]算法以及 LBCDG[11]算法更短的数据收集路径长度.主要有两个方面的原因.

- 一方面,网络平均密度越大,单位面积内拥有节点数量越多的概率越大.这样,各算法所构造的限高树或簇中的节点数量往往会随网络平均密度的增大而增加,选取靠近 Sink 的节点作为逗留节点或簇头的概率也随之增大,从而缩短了各算法的数据收集路径长度;
- 另一方面,DC-Collection 算法充分利用采集节点在移动数据收集器的通信范围内,采集节点可以将接收到的数据以及自身感知的数据发送给移动数据收集器这一特性,使得它所选取的逗留节点数量比 SPT-DGA[13]算法和 LBCDG[11]算法得到的逗留节点数量更少.所以,DC-Collection 算法获得了比 SPT-DGA[13]算法和 LBCDG[11]算法更短的数据收集路径长度.

6 结 论

在大规模的无线传感器网络中收集数据,不仅需要考虑节省节点的能量消耗,而且还需要考虑降低数据收集的延迟.如何有效地均衡节点的能量消耗和最小化数据收集延迟,是一个 NP-hard 问题.通过对数据收集模型进行分析,本文提出一种节能省时的数据收集协议 DC-Collection 来解决这个问题.仿真实验结果表明:与 SPT-DGA[13]算法和 LBCDG[11]算法相比,无论是在约束跳数、节点的通信半径,还是在节点的数量发生变化的情况下,DC-Collection 算法在网络生命周期和数据收集路径长度方面均有较大的优势.

DC-Collection 算法是一种集中式的算法.在构造最优限高树时,DC-Collection 算法需要预先利用移动节点获取全网中的节点的分布信息,这个过程需要消耗较多的能量用于节点间的通信和协调.因此,我们下一步的工作将研究采用分布式算法构造限高树以及选取逗留节点.与此同时,在构造限高树和选取逗留节点时,将考虑更多的参数(如节点的能量等),使得算法更好地应用于真实环境.

References:

[1] Wang Y, Wu HY. DFT-MSN: The delay/fault-tolerant mobile sensor network for pervasive information gathering. In: Proc. of the IEEE INFOCOM. New York: IEEE Press, 2006. 1−12. [doi: 10.1109/INFOCOM.2006.272]

[2] Guo LJ, Beyah R, Li YS. SMITE: A stochastic compressive data collection protocol for mobile wireless sensor networks. In: Proc. of the IEEE INFOCOM. New York: IEEE Press, 2011. 1611−1619. [doi: 10.1109/INFCOM.2011.5934953]

[3] Sugihara R, Gupta RK. Optimal speed control of mobile node for data collection in sensor networks. IEEE Trans. on Mobile Computing, 2010,9(1):127−139. [doi: 10.1109/TMC.2009.113]

[4] Kinalis A, Nikoletseas S, Patroumpa D, Rolim J. Biased sink mobility with adaptive stop times for low latency data collection in sensor networks. In: Proc. of the IEEE GlobeCom. New York: IEEE Press, 2009. 1−6. [doi: 10.1109/GLOCOM.2009.5425600]

[5] Konstantopoulos C, Pantziou G, Gavalas G, Mpitziopoulos A, Mamalis B. A rendezvous-based approach enabling energy- efficient sensory data collection with mobile sinks. IEEE Trans. on Parallel and Distributed Systems, 2012,23(5):809−817. [doi: 10.1109/TPDS.2011.237]

[6] Takaishi D, Nishiyama H, Kato N, Miura R. Towards energy efficient big data gathering in densely distributed sensor networks. IEEE Trans. on Emerging Topics in Computing, 2014,99:1−10. [doi: 10.1109/TETC.2014.2318177]

[7] Zhang XW, Dai HP, Xu LJ, Chen GH. Mobility-Assisted data gathering strategies in WSNs. Ruan Jian Xue Bao/Journal of Software, 2013,24(2):198−214 (in Chinese with English abstract). http://www.jos.org.cn/1000-9825/4349.htm [doi: 10.3724/SP.J.1001.2013.04349]

[8] Olariu S, Stojmenovic I. Design guidelines for maximizing lifetime and avoiding energy holes in sensor networks with uniform distribution and uniform reporting. In: Proc. of the IEEE INFOCOM. New York: IEEE Press, 2006. 1−12. [doi: 10.1109/INFOCOM.2006.296]

[9] Shah RC, Roy S, Jain S, Brunette W. Data mules: Modeling a three-tier architecture for sparse sensor networks. In: Proc. of the ACM SNPA. New York: IEEE Press, 2003. 30−41. [doi: 10.1109/SNPA.2003.1203354]

[10] Chen GH, Li CF, Ye M, Wu J. An unequal cluster-based routing protocol in wireless sensor networks. Wireless Networks, 2009, 15(2):193−207. [doi: 10.1007/s11276-007-0035-8]

[11] Zhao M, Yang YY. A framework for mobile data gathering with load balanced clustering and MIMO uploading. In: Proc. of the IEEE INFOCOM. New York: IEEE Press, 2011. 2759−2767. [doi: 10.1109/INFCOM.2011.5935108]

[12] Ma M, Yang YY, Zhao M. Tour planning for mobile data-gathering mechanisms in wireless sensor networks. IEEE Trans. on Vehicular Technology, 2013,62(4):1472−1483. [doi: 10.1109/TVT.2012.2229309]

[13] Zhao M, Yang YY. Bounded relay hop mobile data gathering in wireless sensor networks. IEEE Trans. on Computers, 2012,61(2): 265−277. [doi: 10.1109/TC.2010.219]

[14] Xing GL, Wang T, Jia WJ, Li M. Rendezvous design algorithms for wireless sensor networks with a mobile base station. In: Proc. of the ACM MobiHoc. New York: ACM Press, 2008. 231−240. [doi: 10.1145/1374618.1374650]

[15] Pon R, Batalin MA, Gordon J, Kansal A, Liu D, Rahimi M, Shirachi L, Yu Y, Hansen M, Kaiser WJ, Srivastava M, Sukhatme G, Estrin D. Networked infomechanical systems: A mobile embedded networked sensor platform. In: Proc. of the IEEE IPSN. New York: ACM/IEEE Press, 2005. 376−381. [doi: 10.1109/IPSN.2005.1440952]

[16] Ma M, Yang YY. Data gathering in wireless sensor networks with mobile collectors. IEEE Trans. on Parallel and Distributed Systems, 2008. 1−9. [doi: 10.1109/IPDPS.2008.4536269]

[17] Guo ST, Wang C, Yang YY. Mobile data gathering with wireless energy replenishment in rechargeable sensor networks. In: Proc. of the IEEE INFOCOM. New York: IEEE Press, 2013. 1932−1940. [doi: 10.1109/INFCOM.2013.6566993]

[18] An HC, Kleinberg R, Shmoys DB. Improving christofides' algorithm for the *s*- path TSP. In: Proc. of the ACM STOC. New York: ACM Press, 2012. 875−886. [doi: 10.1145/2213977.2214055]

[19] Wu Y, Wang C, Fahmy S, Shorff NB. On the construction of a maximum-lifetime data gathering tree in sensor networks NP-completeness and approximation algorithm. In: Proc. of the IEEE INFOCOM. New York: IEEE Press, 2008. 1−9. [doi: 10.1109/INFOCOM.2008.80]

[20] Heinzelman WR, Chandrakasan A, Balakrishnan H. Energy-Efficient communication protocol for wireless microsensor networks. In: Proc. of the IEEE HICSS. New York: IEEE Press, 2000. 1−10. [doi: 10.1109/HICSS.2000.926982]

[21] Intanagonwiwat C, Govindan R, Estrin D. Directed diffusion: A scalable and robust communication paradigm for sensor networks. In: Proc. of the ACM MobiCom. New York: ACM Press, 2000. 56−67. [doi: 10.1145/345910.345920]

附中文参考文献:

[7] 张希伟,戴海鹏,徐力杰,陈贵海.无线传感器网络中移动协助的数据收集策略.软件学报,2013,24(2):198−214. http://www.jos.org.cn/1000-9825/4349.htm [doi: 10.3724/SP.J.1001.2013.04349]

梁俊斌(1979－),男,广西南宁人,博士,教授,主要研究领域为无线传感器网络.

陈宁江(1975－),男,博士,教授,CCF 高级会员,主要研究领域为网络分布式计算.

邹绍军(1987－),男,博士生,主要研究领域为无线传感器网络,数据中心网络.

李韬(1986－),男,博士后,主要研究领域为信息物理系统.