文章编号：1000-8608(2010)04-0591-06

WSN中基于融合代价和传输代价的分簇算法

王洪玉*[1]， 刘 爽[2]

（1.大连理工大学 信息与通信工程学院，辽宁 大连 116024；
2.中兴通信北京研发中心，北京 100083）

摘要： 考虑融合代价所带来的影响，提出CFTC算法，综合考虑了融合代价和传输代价，将二者作为判定簇头节点的主要条件，确保选出的簇头节点执行的簇内融合都是有效的，节省了能耗；同时又考虑了节点的剩余能量且为每个参与簇头竞争的节点增加能量阈值的限制，均衡了节点的能耗负担.仿真结果表明该算法的性能优于LEACH，不仅延长了网络的生命周期，而且还降低了网络的总能量消耗.

关键词： 无线传感器网络；数据融合；融合代价；传输代价；生命周期
中图分类号： TP393 **文献标志码：** A

0 引 言

无线传感器网络(wireless sensor networks, WSNs)是由具有感知、数据处理和短距离无线通信功能的传感器节点构成的一种自组织网络[1,2]，它的内部结构不同于传统的Ad-Hoc网络，节点的通信能力、计算能力、存储能力和电源能量都非常有限[3]，所以路由协议的研究集中在解决如何降低节点能耗而使网络的生存时间最长.现在无线传感器网络中的路由算法多采用分层的结构，将整个网络以簇的形式分层，在簇内采用数据融合技术[4,5]，即簇头节点将其簇内成员节点发送来的数据进行综合，去掉冗余信息，然后将信息量最小化后的数据传送到sink节点，这种将分簇、路由和数据融合结合在一起的方法减少了网内的通信流量，降低了网络的能耗，延长了网络的生命周期.

目前很多分簇算法都是在经典的分簇算法LEACH[6]基础上提出的，多数算法只考虑传输数据所带来的能量消耗，即分簇算法中只考虑到了传输代价，而忽略融合本身的能量开销，即数据融合代价.随着传感器网络的广泛应用，对以图像、多媒体、加密为感知的数据的融合处理算法复杂，能量开销高，甚至与数据传输开销相当[7].因此，为了使网络的能耗最小而进行簇头选举和路由选择时，不仅要考虑传输代价，而且还要考虑融合代价.本文在LEACH协议基础上综合考虑这二者的代价，提出一个新的基于融合代价和传输代价的分簇算法CFTC(clustering based on fusion cost and transmission cost)，以延长网络生命周期、降低网络总能量消耗.

1 理论基础

1.1 融合代价和传输代价

无线传感器网络可用图$G=(V,E)$表示，G是无向图，V表示节点的集合，E表示边的集合，集合E中的边e代表了两个节点对之间的通信链路.数据源节点的集合S是V的子集，它们将感知的数据周期发送到sink节点.

用权重$w(v)$表示集合S中每个节点v输出的信息量.集合E中的边e可表示成$e=(u,v)$，其中u是起始节点，v是终止节点.边e的权重等于起始节点的权重，即$w(e)=w(u)$.每条边用$t(e)$和$f(e)$来描述它的传输代价和融合代价.传输代价$t(e)$表示用于从节点u到节点v传输数据量为

收稿日期： 2008-03-04； **修回日期：** 2010-04-04.
基金项目： 国家自然科学基金资助项目(60272036).
作者简介： 王洪玉*(1968-)，男，教授，博士生导师，E-mail：whyu@dlut.edu.cn.

$w(e)$ 的数据所需要的能量，表达式为 $t(e)=w(e)c(e)d$，其中 $w(e)$ 为 u、v 链路上传输的信息数量，$c(e)$ 为链路单位信息的传输代价，它可以模拟不同的链路情况，d 表示两个节点之间的距离.

融合代价 $f(e)$ 表示在终端节点 v 进行融合过程中所需要的能量，其取决于被融合的数据量以及所采用的算法，为了避免混淆，用 $\tilde{w}()$ 表示融合之前节点的权重. 若节点 v 对 u、v 的数据进行融合，则融合后的权重 $w(v)=(w(u)+\tilde{w}(v))(1-det)$，其中 det 表示数据融合率. 假设边 e 的单位融合代价为 $q(e)$，则 $f(e)$ 可以表示为 $f(e)=q(e)(w(u)+\tilde{w}(v))$.

现在很多分簇算法中的数据融合模型是簇头节点一旦收到成员节点发送过来的数据便进行融合，然后将融合后的数据传送到 sink 节点，可是在这一过程中若簇头节点执行融合后的传输代价和融合代价之和大于直接中继这个数据到 sink 的传输代价，那么簇头节点的融合就没有任何意义并且还浪费了能量，所以为了减少能量的消耗，本文选择出的簇头节点必须满足 $f(e)+t(e)<\tilde{t}(e)$，$\tilde{t}(e)$ 为簇头节点未执行融合而直接传输的代价.

1.2 网络模型

本文中 N 个传感器节点随机分布在一个正方形区域 A 内，并且对该网络作如下假设：

(1)传感器节点部署后不再移动；

(2)sink 节点部署在区域 A 内的一个固定位置，并且是唯一的；

(3)网络中所有的节点是同构的，并且有相同的初始能量；

(4)网络形成的簇和簇之间没有干扰；

(5)簇内节点到簇头之间、簇头节点到 sink 之间都是 $1-hop$.

1.3 能量模型

本文中假设所有节点具有相同的初始能量，但是在每一轮内，每个节点的能量消耗是不相同的，对于成员节点，每一轮的能量消耗包括发送数据和接收数据两部分，而簇头节点还要另外包括融合数据所带来的能量消耗，本文采用文献[8]给出的传感器节点的无线能量模型. 在此模型中，传感器节点发送 k 数据所消耗的能量为

$$E_{tx}(k,d)=E_{elec}k+\varepsilon_{amp}kd^2 \tag{1}$$

传感器节点接收 k 数据所消耗的能量为

$$E_{rx}(k)=E_{elec}k \tag{2}$$

式中：E_{elec} 表示发送电路和接收电路每发送和接收 1 bit 数据的能量损耗，ε_{amp} 表示信号放大器将每 1 bit 数据传送单位距离的能量损耗，d 是信号传输的距离.

簇头节点将其成员发送的数据和自身的数据融合成一个有效数据信号所消耗的能量为

$$E_d=(M+1)E_{da}k \tag{3}$$

其中 E_{da} 表示融合 1 bit 数据所消耗的能量，M 表示簇头节点接收到的成员节点发送数据的个数.

2 算法描述

CFTC 算法是基于融合代价和传输代价来选择簇头节点的，根据前面所述，为了使选出的簇头节点是有效的，执行的数据融合是有意义的，它必须满足下面不等式：

$$dc_0(w(c)+w(v))(1-det)+q_0(w(c)+w(v))<dc_0(w(c)+w(v)) \tag{4}$$

式中：$w(c)$、$w(v)$ 分别为簇头节点和成员节点的权重；c_0、q_0 分别为簇头和 sink 所形成链路的单位信息的传输代价和融合代价；d 为簇头和 sink 之间的距离. 为了判定一个簇头节点是否满足式(4)，应首先选出可能的簇头的节点(possible_ch_node)，只有满足式(4)的可能簇头节点才可成为真正的簇头节点(true_ch_node). 对于可能簇头节点个数的确定采用文献[9]中给出的覆盖给定区域所需的最小节点数的公式：

$$P_{area}/Nr^2\pi=2\pi/\sqrt{27} \tag{5}$$

其中 P_{area} 为整个网络区域的面积，r 为节点的通信半径，由此可以得到可能的簇头节点个数：

$$N=\sqrt{27}P_{area}/2\pi^2r^2 \tag{6}$$

由于会存在一些可能的簇头节点与其通信范围内的所有普通节点都不满足式(4)，那么这些可能簇头节点便成为普通节点，然后进行所属簇的选择，但是最后有些节点仍未能选择到合适的簇头节点，则将直接成为没有成员节点的真正簇头节点，所以真正簇头节点的个数是不确定的，但实验表明与 N 还是很接近的.

CFTC 算法与 LEACH 类似，协议按轮运行，每轮也分为簇的建立和稳定传输两个阶段. 在簇的建立阶段，首先选出 possible_ch_node，进而确

定 true_ch_node，其具体过程如下：

(1)在对每个节点初始化时，首先要设置节点的权重 w，在本文中只有源节点才产生数据，所以初始时刻将源节点的权重值设为它产生数据的量，而非源节点权重值则设为 0.

(2)初始化之后开始选择 possible_ch_node，每个节点从(0,1)任选一个随机数，只有这个随机数小于门限值 $T(n)$，它才可能成为 possible_ch_node，而 $T(n)$的计算方法与 LEACH 相似，只是在原来的阈值中增加了节点的剩余能量因子[10]，$T(n)$变成

$$T(n)=\begin{cases}\dfrac{p}{1-p\left(r \bmod \dfrac{1}{p}\right)}\dfrac{E_{\text{res}}}{E_{\text{init}}}; & n\in G\\ 0; & \text{其他}\end{cases} \tag{7}$$

式中：E_{res} 表示该节点在本轮中当前的剩余能量，E_{init} 表示该节点的初始能量. 而概率 p 的表达式可以根据式(6) 获得，$p=\dfrac{N}{N_{\text{t}}}=\dfrac{\sqrt{27}P_{\text{area}}}{2\pi^2 r^2 N_{\text{t}}}$，$N_{\text{t}}$ 为整个网络节点的总个数.

由于考虑了节点当前的剩余能量，可以改善 LEACH 算法中当簇头能量过低而很快耗尽能量这种不利的情况，使能量负载均匀分布到网络内的所有节点上，推迟了节点的死亡时间，从而延长了网络的生命周期. 同时为了避免过低能量的节点被选为簇头，对每个节点增加了一个能量阈值 $E_{\min}$限制，只有能量不低于 $E_{\min}$ 的节点才可参与 possible_ch_node 的竞争. 一旦被选为 possible_ch_node 便要在整个网络内广播簇头选择请求 possible_ch_request 原语.

(3)收到簇头选择请求 possible_ch_request 原语的节点将其插入到簇选择请求表 request_table 中，然后中断一段时间 T_1 后选择所属的簇头节点 possible_ch_node，选择的依据是距离最短原则，即选择与 request_table 存有的簇头节点距离最短的节点作为自己可能所属的簇头，然后向其发送簇选择应答 ch_reply 原语，这个包中含有该节点的权重值.

(4)只有被选中的 possible_ch_node 会收到簇选择应答 ch_reply 包，收到此包后要判定是否满足式(4)，若满足，则此 possible_ch_node 便成为了真正的簇头节点，将标志位 CH_Identifier 赋值为 true，同时此节点的权重变成了融合之后的权重，即 $w(c)=(w^{\sim}(c)+w(v))(1-det)$. 然后该簇头节点要发送簇选择应答确认原语 reply_ack 来通知这个节点它已成为自己的簇内成员节点. 若不满足式(4)，则该节点要发送簇选择应答否决原语 cancel_reply 包. 设定中断时间 T_2，若在 T_2 内，该节点标志位 CH_Identifier 仍未转变成 true，则此节点转变成一个普通节点，并从它的 request_table 中重复上述过程选择自己的所属头节点，若在此期间，该节点标志位 CH_Identifier 成为 true，则中断停止.

(5)收到 reply_ack 的节点成为发送此包的簇头节点的簇内一员，member_identifier 赋值为 true，而收到 cancel_reply 的节点要将此 possible_ch_node 从它的 request_table 中删除，同时根据距离最短原则从表中继续选择 possible_ch_node，若删除后表的长度为 0，即不再有可选的节点，则此节点立刻变成 possible_ch_node，然后广播 ch_request.

(6) 上述过程一直持续到定时器时间 T_3，若在 T_3 之后有些节点的 CH_Identifier 和 member_identifier 仍均为 false，则这些节点便成为孤立的簇头节点，此时完成了簇的建立过程.

对上面的过程用伪码的形式进行简明描述，对于网络内的每个节点 n_i：

```
1.  Set timer T3{
2.  If Eres > Emin and random(0,1)<T(n) then
3.     {possible_ch_identifier = true
4.     Broadcast possible_ch_request(node_name,position)}
5.  On receiving a possible_ch_request from nj
6.     Add nj to request_table;Set timer T1,after T1
7.  If(sizeof(request_table) ==0 and possible_ch_identifier == false) then
```

```
8.    {possible_ch_identifier = true
9.    Broadcast possible_ch_request(node_name,position)}
10. Else
11.   {Choose possible_ch_node from request_table    // according to the shortest distance
12.   Send a ch_reply_packet(weight)}
13. On receiving a ch_reply_packet
14.   If satisfies the inequation(4) then
15.       {ch_identifier = true
16.       w(i)=(w(i)+ ch_reply_packet->weight)(1-det)}
17.     Add the node to member_table
18.     Send a reply_ack_packet
19.   Else
20.     {Set timer T2,after T2
21.     Send a cancel_reply_packet}
22. On receiving a reply_ack_packet
23.   member_identifier = true
24.   Send all the data in the queue to ch and Exit
25. On receiving a cancel_reply_packet
26.   Remove the possible_ch_node from request_table
27.   If sizeof(request_table)==0 then
28.     Broadcast possible_ch_request
29.   Else
30.     Choose possible_ch_node from request_table    //according to the shortest distance
31.     Send a ch_reply_packet }
32. If T3 timeout then the node is an isolated clusterhead node with no members
```

完成簇的建立后便进入稳定的数据传输阶段,簇头节点会基于簇内节点的数量建立一个TDMA时间列表来决定每个节点何时传输,簇中的成员节点根据这个TDMA时间列表把收集的数据传送给簇头,簇头将接收到的数据与自己产生的数据作融合处理,而对于孤立的簇头节点则不需要这些过程.簇头节点的数据传送本文采用的是直接发送数据到sink节点的方法,虽然这种方法与多跳的方法相比会加剧簇头节点的能量消耗,但是单跳传送的延时比较小,对于多跳的数据传送方法会在以后的工作中做深入研究.

3 算法的仿真实现与结果分析

根据本文前面所述的网络模型,采用仿真的方法对CFTC算法和LEACH算法的性能进行比较分析,仿真的模拟环境是在100 m×100 m内随机分布50个节点,sink节点的位置是(50,50).仿真中源节点以恒定的时间间隔6 s发送数据,数据的大小是1 024 bit,轮的定义为20 s的时间间隔.CFTC算法中的能量门限值E_{min}设为0.1 J,数据的压缩率det设为0.5,单位融合代价q_0与E_{da}相同,单位传输代价c_0设为1,本文认为传输代价仅由传输距离决定.其余的仿真参数与LEACH相同,$E_{elec}=50$ nJ/bit,$\varepsilon_{amp}=100$ pJ/bit/m^2,$E_{da}=5$ nJ/bit/signal.

仿真运行期间采用网络生存时间和消耗的能量作为算法节能的评价指标,网络的生存时间是以轮数R来表示的,可以定义为在达到规定的死亡节点个数时网络所持续的轮数.

图1是在随机分布50个节点都是源节点且初始能量为1 J的环境中进行仿真得到的曲线图,由图可以看到,CFTC算法在网络内的死亡节点个数满足不同的要求时性能一直优于LEACH,与LEACH相比可延长9%的生命周

期. 在这种环境中，网络内的所有节点都是源节点，都可以产生数据并且还要进行数据的转发，而 CFTC 算法可以确保在簇内进行的数据融合都是有效的，避免了不必要的数据融合所带来的更多的能量消耗，提高了能量的利用率，同时尽可能使剩余能量较多的节点充当簇头节点，均衡了节点的能量消耗，延长了网络的生命周期.

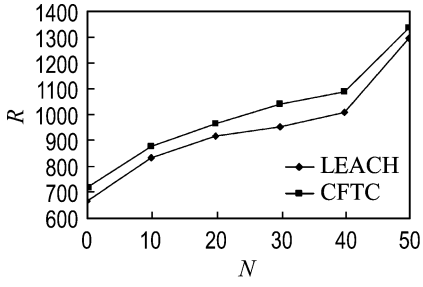

图 1　节点死亡的生命周期图

Fig. 1　The lifetime curve of dead node

图 2 展示了不同初始能量下两个算法的性能比较，这里的生命周期定义为一个节点死亡时网络所持续的轮数. 从图上的曲线可以看到，随着节点初始能量的增加，CFTC 的生命周期比 LEACH 的生命周期提高的幅度也相应增加，由起点的 10%到终点的 14%，这是因为在 CFTC 算法中簇头的选举考虑了节点剩余能量的因素，具有高剩余能量的节点成为簇头的概率更大，从而均衡了网络内节点的能耗，极大地延长了网络的生命周期，同时又对每个参加竞选 possible_ch_node 的节点增加了一个能量阈值的限制，使得能量较低的节点不能参与簇头节点的竞争，降低了低能量节点成为簇头的概率，进而延长了第一个死亡节点出现的时间，最大化了网络的生存时间. 而图 3 表明 CFTC 算法在提高网络生命周期的同时还节省了网络的能量消耗(E_{cons})，图 3 为节

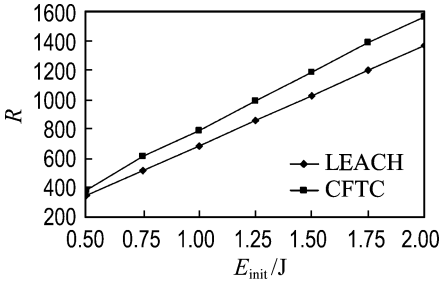

图 2　初始能量和网络寿命关系图

Fig. 2　The network lifetime vs. initial energy

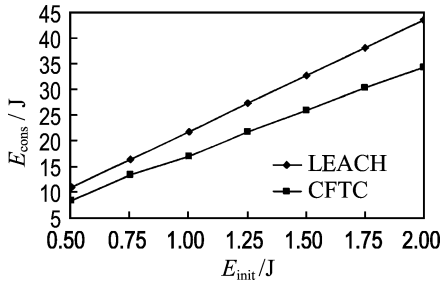

图 3　初始能量和能量消耗关系图

Fig. 3　The total energy cost vs. initial energy

点在不同的初始能量情况下，第一个节点死亡时整个网络的能量消耗，与图 2 是相对应的，可以很明显地看到：与 LEACH 算法相比，CFTC 不仅可以将网络的生命周期延长至少 10%，同时还可以节省近 25%的能量.

图 4 为不同单位融合代价(即 E_{da} 不同)情况下两个算法轮数的比较. CFTC 算法是在考虑了融合代价的条件下进行簇头选择的，适用于融合代价较高的应用环境. 从图上也可以看到 CFTC 在单位融合代价较高的环境下性能明显优于 LEACH，将网络生命周期延长了 50%，这是由于 CFTC 簇内的成员节点与簇头节点之间的数据融合都是有效的数据融合，而对于 LEACH 算法，其中有些成员节点与簇头节点融合后的数据传送到 sink 的能量消耗会大于直接发送，增加了节点的能耗，所以 CFTC 算法延长了网络的生命周期. 而从图上也可以看到对于融合代价较低的应用环境，CFTC 算法的性能还是优于 LEACH，将网络的生命周期延长了近 15%，由此可见，CFTC 算法的应用范围更加广泛. 而本文均采用简单的数据融合算法来模拟融合代价高低不同的环境，若采用复杂的融合算法，则 CFTC 算法的性能会更加优于 LEACH.

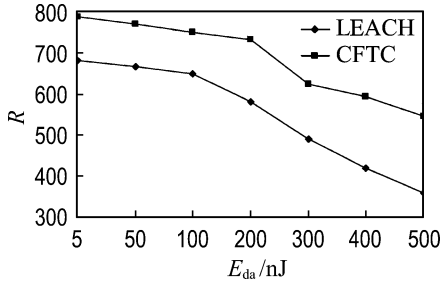

图 4　单位融合代价和网络寿命关系图

Fig. 4　The network lifetime vs. unit fusion cost

4 结 语

本文提出的CFTC簇头选择算法考虑了融合代价和传输代价,使得所分的簇的簇内成员节点与簇头节点之间的数据融合都是有效的,避免了成员节点与簇头融合后数据传输的能耗大于直接中继的能耗,保证了簇头节点不会由于执行无效的数据融合而带来不必要的能量消耗,进而节省了整个网络的能量损耗,延长了网络的生命周期.同时又考虑了节点的剩余能量以及为每个参与簇头竞争的节点增加能量阈值的限制,均衡了节点的能量消耗.仿真结果表明CFTC算法有效地提高了网络的生命周期,节省了能量消耗.

参考文献:

[1] AKYILDIZ I, SU W, SANKARASUBRAMANIAM Y, *et al*. A survey on sensor networks [J]. **IEEE Communications Magazine**, 2002, **40**(8):102-114

[2] LAMBROU T P, PANAYIOTOU C G. A survey on routing techniques supporting mobility in sensor networks [C] // **5th International Conference on Digital Mobile Ad-Hoc and Sensor Networks**. SA: IEEE Computer Society, 2009:78-85

[3] 孙利民,李建中,陈 渝,等. 无线传感器网络[M]. 北京:清华大学出版社, 2005

[4] AL KHDOUR T A, BAROUDI U. A generalized energy-aware data centric routing for wireless sensor network [C] // **IEEE International Conference on Signal Processing and Communications**. United Arab Emirates:IEEE Computer Society, 2007:117-120

[5] HUNG Li-ling, CHANG Sheng-wen, CHANG Chih-yung, *et al*. A data-centric mechanism for wireless sensor networks with weighted queries [C] // **Joint Conferences on Pervasive Computing (JCPC)**. Taiwan:IEEE Computer Society, 2009:131-136

[6] HEINZELMAN W, CHANDRAKASAN A, BALAKRISHNAN H. Energy-efficient communication protocol for wireless microsensor networks [C] // **The 33rd Annual Hawaii International Conference on System Sciences**. Hawaii: IEEE Computer Society, 2000:3005-3014

[7] LUO H, LUO J, DAS S K, *et al*. Routing correlated data with fusion cost in wireless sensor networks [J]. **IEEE Transactions on Mobile Computing**, 2006, **5**(11):1620-1632

[8] LUO H, LUO J, DAS S K, *et al*. Adaptive data fusion for energy efficient routing in wireless sensor networks [J]. **IEEE Transactions on Computers**, 2006, **55**(10):1286-1299

[9] SLIJEPCEVIC S, POTKONJAK M. Power efficient organization of wireless sensor networks [C] // **IEEE International Conference on Communications ICC'01**. Petersburg:IEEE Communication Society, 2001: 472-476

[10] 刘 明,龚海刚,毛莺池,等. 高效节能的传感器网络数据收集和聚合协议[J]. 软件学报, 2005, **16**(12):2106-2116

A clustering algorithm based on fusion and transmission cost for WSN

WANG Hong-yu*[1], LIU Shuang[2]

(1. School of Information and Communications Engineering, Dalian University of Technology, Dalian 116024, China; 2. Research Center of ZTC, Beijing 100083, China)

Abstract: To consider the fusion cost, the clustering based on fusion cost and transmission cost (CFTC) algorithm which takes into account fusion and transmission cost synthetically is proposed. Both of the costs are used to determine cluster head. It makes sure that the cluster head selected can perform data fusion effectively and save energy. Meanwhile, the algorithm also considers the residual energy and sets threshold value for every candidate. So energy load among the nodes in network can be balanced. Simulation results show that the CFTC algorithm not only prolongs network lifetime but also decreases total energy consumption compared with LEACH.

Key words: WSN (wireless sensor network); data aggregation; fusion cost; transmission cost; lifetime