密集型传感器网络 MAC 地址复用压缩算法①

龚玄辉，周四望

(湖南大学 软件学院，长沙 410082)

摘 要：针对传感器网络中 MAC 地址在通信时能量开销太大的问题，提出了一种新的适用于节点密集分布的 MAC 地址复用压缩算法。该算法将传感器网络节点进行区域网格分割，节点以区域为单位进行地址复用。为削弱因网络节点分布的非均匀性带给算法性能的负面影响，提出了通过标志位标识的非对称性 MAC 地址分配原则。理论和实验结果表明，该算法在密集型传感器网络中且节点分布不均匀情况下能很好的发挥地址压缩性能。

关键词：无线传感器网络；MAC 地址；密集型；地址压缩；网格划分

MAC Address Multiplexing Compression Algorithm for Intensive Wireless Sensor Networks

GONG Xuan-Hui, ZHOU Si-Wang

(Software School, Hunan University, Changsha 410082, China)

Abstract: Aiming to decrease the energy consumption of MAC address in wireless sensor networks, we propose a new MAC address multiplexing compression algorithm. In our proposed algorithm, nodes' addresses are multiplexed using regional grid segmentation, which on a regional basis. An asymmetry MAC address allocation principle allocation using bit flag is also presented, which can weaken the negative effects for irregular distribution of sensor nodes. Theoretic analysis and experimental result shows that our algorithm plays a good performance in the situation that the wireless sensor networks are intensively deployed and the distributions of nodes are irregular.

Key words: wireless sensor networks; Mac address; intensive; address compression; mesh generation

无线传感器网络只有有限的能量和通信带宽，但是通信数据报文中的报头控制信息却占据了太大的比例，消耗了大量能量，这使得无线传感器难以承载大数据量的传输，急需要对参与网络通信传输的数据报头进行压缩处理[1]。

对于报头压缩技术，这在传统网络中已取得良好的进展。从最初的 VJHC[2]、ROHC[3]到在 IPV6 协议下的基于 RFC4944 的压缩算法[4]，都为报头压缩技术打下了坚实的基础。然而，传统的报头压缩技术并没有考虑到压缩算法带来的能量消耗问题，且由于通信协议的不同，所以并不能完全适应于无线传感器网络。

近些年来，针对无线传感器网络报头压缩的研究并不是很多，仅有的算法虽然在一定程度上能较好的压缩地址空间，但是却或多或少存在着各自的缺陷。如文献[5]提出了一种基于密钥的 MAC 地址压缩算法：传输报文时不携带 MAC 地址，而是通过密钥异或计算数据来源。该协议由于不需要在数据包中携带 MAC 地址，有效地节省了传输能耗。但这种算法在节点密度较大时，依次使用同各个邻居节点的密钥进行计算会给网络带来较大时延，同时由于校验码被用来恢复地址，影响了其检错能力。

文献[6]中的 VGSR 压缩算法，以划分单元格的方式复用 MAC 地址，每一个节点被划分到一个单元格内，并确保其地址与其通信范围内的其他节点地址不同。该算法基于每个网格只能有一个节点且节点都在网格中心的假设，这显然过于理想，因为现实网络布局中很难确保节点的这种均匀等距分布。

本文采用了网格划分的方法对传感器网络节点进

① 基金项目:国家自然科学基金(60973127);湖南省自然科学基金(09JJ3123);博士后科学基金(20090461005)
收稿时间:2011-04-11;收到修改稿时间:2011-05-19

行区域网格分割，每一个网格作为一个独立的单元格，单元格选举出各自的簇头节点，簇头节点以复用区域为单位进行地址复用，而簇内节点则以单元格为单位进行地址复用的地址压缩算法。

1 MAC地址分配算法

1.1 网格划分

我们将网络在逻辑上划分成不同的单元格，并按一定的策略选举出每个单元格的簇头节点，剩下的节点为簇内节点。簇内节点只与自身所处单元格的簇头节点通信，单元格间通信则通过簇头节点。簇内节点因其通信的特殊性，所以它们以单元格为单元进行地址复用，而对于簇头节点，则以复用区域为单位进行地址复用。

复用区域为包含多个单元格的正方形区域，在这个区域中的簇头节点被分配的 MAC 地址号都是唯一的，如果能使整个网格中的其他所有簇头节点的都按这个区域中簇头节点的 MAC 地址分配原则进行排列的话，就能保证整个网络里每个具有相同 MAC 地址的簇头节点的通信范围都在 2R 之外。

设复用区域的边长为 d，单元格的边长为 r，节点通信半径为 R，对于三者的关系，本文取：$r=\frac{2}{3}R$ 即 R=1.5r 此时 d=4r。

网络单元格的划分的划分情况如图 1 所示，此图中，每个复用区域中簇头的个数为 16 个，最短簇头 MAC 集为 $\Phi_{cluster}$={0,1,2,….., 15}。

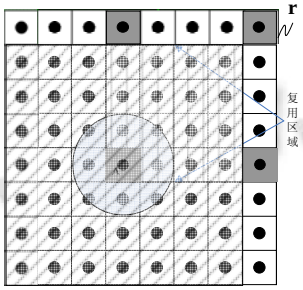

图 1 复用区域示意图

在实际运行中，对于网络中每一个节点，都必须为之建立一个位置—地址的映射关系，使其能根据自身的节点坐标信息计算出字的的所属单元格编号，而簇头节点则以此编号作为自己的地址。

设某节点坐标为(x,y),复用区域边长为 d,单元格边长为 r，则与之相对应的单元格编号 U 为：

$$\begin{cases} a=\lfloor x/r \rfloor \bmod(d/r) \\ b=\lfloor y/r \rfloor \bmod(d/r) \end{cases} \tag{1}$$

$$U=a+(d/r)b \tag{2}$$

节点计算出 U 后，将其保存。继续根据公式(3)、(4)求出自身位置离所属单元格中心的距离 s。此值将用来做簇头节点选举算法中的反比权重值。

$$\begin{cases} m=x \bmod r \\ n=y \bmod r \end{cases} \tag{3}$$

$$s=\sqrt{(m-\frac{r}{2})^2+(n-\frac{r}{2})^2} \tag{4}$$

1.2 簇头节点选举算法

我们定义在簇头选举过程中，节点有如下四种状态：

1)初始状态 (state=init)

2)等待状态 (state=wait)

3)簇头节点状态 (state=cluster)

4)簇内节点状态 (state=node)

算法的基本步骤如下：所有节点计算出 U 和 s 后，设置自身状态为初始状态，单元格进入簇头选举阶段。此阶段由单元格中随机一节点发起，此节点将自身保存的 U 和 s 绑定在 M_{com} 消息中广播出去。其他节点接到 M_{com} 消息后，根据自身状态和消息中距离值 s`与自身 s 比较结果执行相应步骤：

1)若节点状态为初始状态(line 08)，且 s`>s，节点进入等待状态,广播自身 M_{com} 消息并同时启动定时器 1，设置值为 $\Delta\phi$。否则节点进入簇内节点状态，转化为簇内节点，本节点算法结束。

2)若当前节点状态为等待状态(line 10)，如果 s`>s，节点维持自身状态不变。如果 s`=s，节点回退至初始化状态,并在 0~$\Delta\tau$ 中随机选择一指重置定时器 2，准备重新广播自身 M_{com} 消息，以便再次进入等待状态。如果 s`<s，则节点关闭定时器，进入簇内节点状态，本节点算法结束。

处于等待状态的簇头节点，如果在 $\Delta\phi$ 时间段内未接收到 s`<=s 的 M_{com} 消息，则设置自身状态为簇头状态(state=cluster)，将单元格编号 U 保存为自身的 MAC 地址，并广播 $M_{cluster}$ 消息(line 2)，宣告自己竞选簇头节点成功，本节点算法结束。而其他节点接收到

此消息，则进入簇内状态(line 13)，单元格内簇头选举阶段结束。

算法伪代码如下：

```
01 while(state=init||state=wait): wait for event
02 case event==timer1End:state=cluster;send(M_clust- er); break;
03 case event ==timer2End:send(M_com);break;
04 case event ==getMessage:etMessage(message);
05    end while
06
07 function getMessage(message)
08   if(message==M_com&&state==init)
09 switch(compareDist(M_com))
10   else if(message==M_com&&state==wait)
11 switch(compareDist(M_com))
12   else if(message==M_cluster)
13       state=node;
14   else   delete(message);
15       end if
16       end function
```

1.3 簇内节点地址分配算法

当整个网络中簇头节点选举完毕后，网络开始进入簇内节点地址分配阶段。在此阶段中，我们同样规定簇内节点会经历如下三个状态。

1)预设值状态 (state=primary)

2)活动状态 (state=active)

3)就绪状态 (state=finish)

此阶段由簇内节点所处单元格的簇头节点通过连续广播 2 次 $M_{cluster}$ 发起。簇内节点连续两次接收到本单元格的 $M_{cluster}$ 消息后，进入预设值状态，预设自身 MAC 地址 D_1 为 0。节点间通过绑定了单元格编号与发送节点 MAC 地址 D_2 的 M_{node} 消息来分配地址。

我们将来自本单元格且消息中携带的 D_2 与接收节点的 D_1 比较关系为 D_2 >=D1 的 M_{node} 消息称为有效 Mnode 消息。算法的基本步骤如下：

1) 处于预设值状态的节点若在时间段内接收到有效 Mnode 消息(line 11)，则重新赋值 D_1: D_1= D_2+1，设置定时器 1，节点重新等待有效 M_{node} 消息。若未接收到有效 M_{node} 消息消息，则在 0~$\Delta\tau$ 时间内随机一时刻将自身 M_{node} 消息广播出去，节点进入活动状态(line 4)，并设置定时器 2 为 $\Delta\sigma$ 。

2)处于活动状态的簇内节点则继续等待时间，如果此期间内接收到有效 M_{node} 消息(line 12)，且 D_2=D_1，则在 0~$\Delta\sigma$ 时间内随机一时刻重新广播自身 Mnode 消息；否则，D_1= D2+1，节点重新进入预设值状态。如果在 $\Delta\sigma$ 时间未接收到有效 M_{node} 消息，则 D_1 保存为自身簇内 MAC 地址，节点进入就绪状态，本节点算法结束(line 5)。

算法伪代码如下：

```
01 set(timer1)
02 while(state=primary||state=active): wait for event
03   case event == timer1End:
04   state=active;send(comMessage);set(timer2);break;
05   case event ==timer2End: myAddr=D2;state=finish; break;
06   case event ==getComMessage: getComMessage (com
Message);
07   end while
08
09   function   getComMessage(comMessage)
10     if(state==primary)
11       switch(compareAddr(comMessage))
12     else if(state==active)
13       switch(compareAddr(comMessage))
14         end if
15     end function
```

算法结束后，簇头节点通过簇内节点地址的最大值来确定本单元格簇内节点地址长度。

这样分配完 MAC 地址后(图 2)，可以将每一个单元格看做一个单独的子网，子网号用本单元格簇头节点的 MAC 地址标识。簇内节点只与簇头节点通信，而簇头节点则负责将接收到得簇内节点信息按一定的路由策略通过簇头节点间的多跳传输传送到汇集节点。

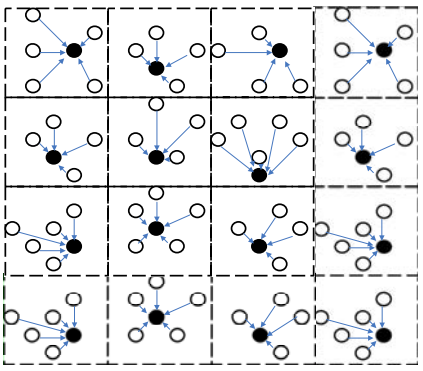

图 2　MAC 地址分配结果

1.4 讨论

簇头选举算法以节点自身位置离所属单元格中心的距离值 s 为反比权重，其值越小，竞选成功的可能性越大。这样才能使簇头节点最靠近单元格的中心点，从而使簇头节点间的连通率达到最高，算法中 $\Delta\phi=\frac{4s}{v}<<\Delta\tau$，v 为波速。簇内地址分配算法中 $\Delta\sigma=\frac{2\sqrt{2}r}{v}<<\Delta\tau$，r 为单元格边长。$\Delta\tau$ 之所以要远大于 $\Delta\phi$、$\Delta\sigma$ 是为了保证拥有相同值的节点间消息碰撞的几率大大减少。

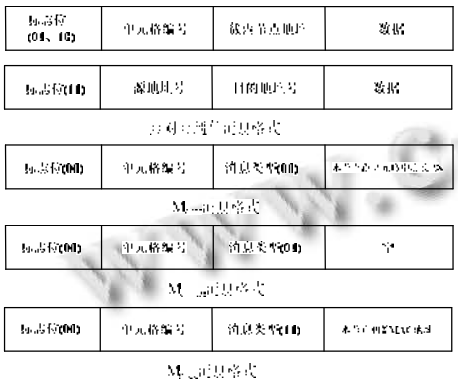

图 3 通信消息格式

由于在每一个单元格内，簇内节点被分配的 MAC 地址与簇头节点的 MAC 地址有可能相同，为避免通信混淆且使本文算法能有效抵御因节点不均匀性带来的负面影响，本文采用两比特的通信标志。考虑到点到点通信时一般会有簇头到簇头、簇内到簇头、簇头到簇头这三种状况，所以我们使用两比特的标志位区分，分别为 11、01、10，而剩下的 00 用来表示节点间的广播通信。图 3 为消息的通信格式，其中的 M_{com} 消息、$M_{cluster}$ 消息、M_{node} 消息都是广播消息，所以其标志位都设置为 00，而标识位为 00、01、10 的消息因为都属于单元格内通信，所以它们都需统一携带自身所处的单元格编号。

节点在接收到消息时，先根据通信标志位来决定是否需要进行进一步的地址判别，如果需要进一步地址判别，则根据自身保存的地址长度分别取出源地址与目的地址进行判别。

2 算法的分析与模拟

本次仿真实验基于网络仿真平台 OMNet++ 4.0，MAC 层协议采用 802.11。实验模拟在指定区域 L=120m 分别随机布置 N 个通信半径 R=18m 的节点。着重考察本文算法在节点密集分布情况下的压缩性能。

此时，整个网络被分割成 100 个单元格，图 4 为仿真系统演示界面局部截图，展示了网络中某一复用区域中单元格的分布情况与簇头选举情况。

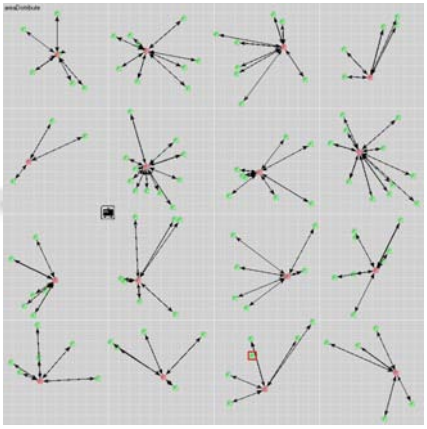

图 4 仿真系统演示界面(局部)

2.1 算法性能分析

2.1.1 平均连通度对地址平均长度的影响

假设传感器网络中节点通信半径为 R，逻辑划分的单元格边长为 r，某复用区域的网络平均连通度为 $n=\frac{\lambda r^2}{\pi R^2}$，则此区域内的单元格中的节点平均个数为：$r=\frac{2}{3}R$，而本文中 r 与 R 存在关系：，所以有 $n=\frac{4\lambda}{9\pi}$。则簇内节点 MAC 地址的平均长度为则为：

$$L=\left\lceil \log_2(\frac{4\lambda}{9\pi}-1)\right\rceil \tag{5}$$

公式中 $\lambda>14.13$，当 λ 小于此值时，L=1。

整个复用区域内，节点的 MAC 地址平均长度 AVG 为：

$$AVG=\frac{4\lambda\left\lceil \log_2(\frac{4\lambda}{9\pi}-1)\right\rceil+36\pi}{4\lambda+9\pi} \tag{6}$$

如果考虑标志位给 MAC 地址平均长度带来的影响，则节点 MAC 地址的平均长度 AVG’为：

$$AVG'=AVG+1 \tag{7}$$

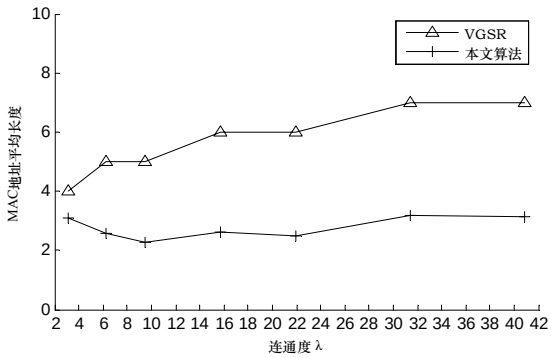

图 5　不同连通度情况下的 MAC 地址长度比较

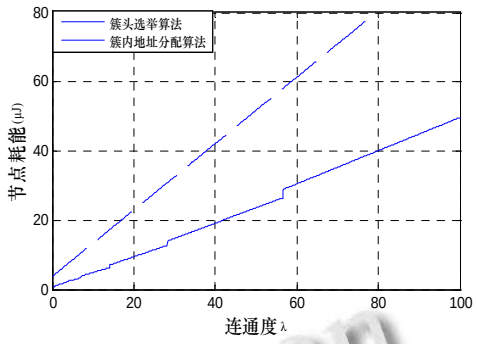

图 6　节点在算法中的平均耗能

图 5 比较了在不同平均连通度的情况下，本文算法和 VGSR 算法所需的 MAC 地址大小。从图中可以看出，本文算法所需的地址长度明显低于 VGSR 算法所需的地址长度。

2.1.2 能量的开销与收益

首先讨论通信消耗。本文算法存在三种通信消息：M_{com} 消息、$M_{cluster}$ 消息、M_{node} 消息。其消息格式如图 3 所示，其中通携带数据长度 M_{com} 消息为 16bit，$M_{cluster}$ 消息为 0bit，M_{node}e 消息则为 $\left\lceil \log_2(\frac{4\lambda}{9\pi}-1) \right\rceil$，所以总的消息长度：$M_{com}$ 消息为 24bit，$M_{cluster}$ 消息为 8bit，M_{node} 消息则为 $8+\left\lceil \log_2(\frac{4\lambda}{9\pi}-1) \right\rceil$ bit。

假设节点接收和发送一比特数据所需要的能量分别为 Er 和 Es。在簇头选举算法中，每一个节点平均会接收到来自其通信范围内的每一个节点广播的 M_{com} 消息和一个 $M_{cluster}$ 消息，且自身发送一个 M_{com} 消息，考虑到网络中每个节点通信范围内平均节点个数即为网络平均连通度，则簇头选举算法中给节点带来的平均能量消耗 $E_{cluster}$ 为：

$$E_{cluster}=[24(\lambda-1)+8]E_r+24E_s \quad (8)$$

而簇内节点地址分配算法中在平均情况下会接收到来自其通信范围内的每一个节点广播的 M_{node} 消息，并且自身广播一个 M_{node} 消息，则簇内节点地址分配算法给节点带来的平均能量消耗 E_{node} 为：

$$E_{node}=(8+\left\lceil \log_2(\frac{4\lambda}{9\pi}-1) \right\rceil)[(\lambda-1)E_r+E_s] \quad (9)$$

如果通信耗能的参数取值为：E_r=0.18 μ J，E_r=0.04 μ J，则节点在算法中的耗能情况如图 6 所示。

接下来分析算法执行完毕后节点的能量收益：

假设网络存在 t 个复用区域，为计算方便，设复用区域的网络平均连通为 $\lambda=18\pi\approx56.5$，则单元格内节点平均个数为 8，网络总的节点个数为 128*t 个，采用全局唯一 MAC 地址通信时地址长度为 $2(7+\lceil \log_2 t \rceil)$ bit。而本文中通信量最大的单元格间通信，平均地址长度也仅为 9bit，节点每次发送消息时减少 $(5+2\lceil \log_2 t \rceil)$ bit。由公式(8)、(9)可知每个节点额外能量消耗为：84.36 μ J。因此，每个节点节约的能量 E_{save} 与发送消息个数 k 直接的关系为：

$$E_{save}=(1.1+0.44\lceil \log_2 t \rceil)k-84.36 \quad (10)$$

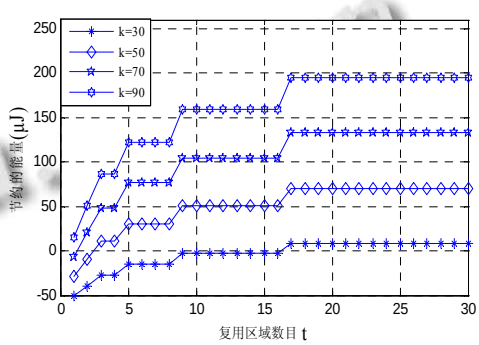

图 7　节点节约的能量与复用区域数目的关系

从图 7 中可以看出：相同 K 条件下，随着复用区域数目的增加，节点节约的能量也相对提升。

3　结语

在无线传感器网络中，随着网络节点的增多，报头的数据量也越大，极大的浪费了节点有限的能量。本文算法采用网格划分的 MAC 地址复用算法，并考

（下转第 63 页）

词到词之间 3 元语言模型我们也是借助 SRILM 得来。

测试结果如下：

表 1 不考虑上下文的实验结果

错误类型	错误单词数	纠正单词数	纠错率
单词错误	6956	6470	93.0%
真词错误	4581	0	0%
两者都有	11537	6470	56.1%

表 2 考虑上下文的实验结果

错误类型	错误单词数	纠正单词数	纠错率
单词错误	6956	6524	93.8%
真词错误	4581	3871	84.5%
两者都有	11537	10395	90.1%

从实验结果我们可以看出,考虑上下文时，即采用词到词之间的 3 元语言模型时，能纠正一些真词错误类型的单词；而不考虑上下文时，不能纠正真词错误类型的单词。不论考不考虑上下文，单词错误类型的单词纠错率比真词错误类型的单词纠错率高出很多。

6 结语

基于上下文的拉丁维文的拼写校对是一个难度比较大的研究课题。本系统实现了基于上下文的拉丁维文纠错功能，但是对于真词错误类型的单词的纠错率还是比较低的，光靠统计学的方法还是不能完全解决句法和语义上的某些真词错误的。我们应该针对拉丁维文的文本进行分析，加强句法和语义层次的校对策略研究，与目前的统计方法相结合，从而更进一步提高拉丁维文的校对准确率。语义问题是语言学与自然语言处理研究中的薄弱环节，语义错误的校对在拉丁维文的文本校对中仍未实现，需待进一步研究。

参考文献

1 龚小谨,罗振声,骆卫华.中文文本自动校对中的语法错误检查.计算机工程与应用,2003,8:98–100,127.

2 骆卫华,罗振声,龚小谨.中文文本自动校对的语义级查错研究.计算机工程与应用,2003,12:115–118.

3 玛依热·依布拉音,米吉提·阿不里米提,艾斯卡尔·艾木都拉.基于最小编辑距离的维语词语检错与纠错研究.中文信息学报,2008,22(3):110–114.

4 张仰森,俞士汶.文本自动校对技术研究综述.计算机应用研究,2006,6:8–12.

5 古丽拉·阿东别克,艾尔肯·伊米尔.维吾尔文校对中常见错误分析.计算机工程与应用,2005,27:181–183.

6 Kukich K. Techniques for Automatically Correcting Words in Text. ACM Computing Surveys, 1992,24(4): 377–438.

7 Merrett TH, Shang HP. Trie Methods for representing text. Proceedings of the International Conference on Foundations of Data Organization and Algorithms. Lecture Notes in Computer Science vol. 730, Springer Verlag,1993:130–145.

8 Stolcke, Andreas. Srilm-an extensible language modeling tookit. Proceedings of the International Conference on Spoken Language Processing,2002:311–318.

（上接第 74 页）

虑到节点的不均匀性，提出簇内节点以单元格为单位进行地址复用，簇头节点以复用区域进行地址复用，以通信标志位来避免消息混淆。这样能抵御节点的不均匀性带来的压缩算法的弱化。仿真试验的结果表明：在相同的地址长度下，本文算法能容纳更多的节点，在节点密集分布的情况下，其地址压缩性能较为突出。

参考文献

1 Kan BQ, Cai L, Zhu HS, et a1. Accurate energy model for WSN node and its optimal design.Journal of Systems Engineering and Electronics,2008,19(3):427–433.

2 Jacobson V. Compressing TCP/IP Headers for Low-Speed Serial Links. Request for Comments 1144, February 1990.

3 Bormann C, Burmeister C, Degermark M, et al. RObust Header Compression (ROHC): Framework and four profiles: RTP, UDP, ESP, and uncompressed. Request for Comments 3095. July 2001.

4 Montenegro G, Kushalnagar N. Transmission of IPv6 Packets over IEEE 802.15.4 Networks. 2007.

5 Chin KW, Lowe D, Sanchez RG. A new technique for reducing MAC address overheads in sensor networks.IEEE Communications Letters, 2006,10(5):338–340.

6 田野,盛敏,李建东.一种新的传感器网络 MAC 地址分配算法.西安电子科技大学学报,2006,33(5):716–720.