无结构动态适应无线传感器网络数据融合算法

乐俊，张维明，肖卫东，汤大权，唐九阳
（国防科学技术大学 信息系统工程重点实验室，湖南 长沙 410073）

摘 要：研究现有数据融合算法无法满足网络动态变化频繁和数据通信可靠性要求高的新应用需求的问题，提出一种无结构动态适应的数据融合算法，将网络划分为若干个栅格，传感器以栅格为单位进行数据通信并采用“多对多”的数据通信方式，传感器之间无需建立和维护结构。仿真实验结果表明，算法在实现数据融合的同时，能够适应网络的动态变化、提供高可靠性的数据通信。

关键词：无线传感器网络；数据融合；无结构；动态适应
中图分类号：TP393 **文献标识码**：A **文章编号**：1000-436X(2012)09-0053-13

Structure-free and dynamic-adaptive data fusion algorithm for wireless sensor networks

YUE Jun, ZHANG Wei-ming, XIAO Wei-dong, TANG Da-quan, TANG Jiu-yang
(Science and Technology on Information Systems Engineer ng Laboratory, National University of Defense Technology, Changsha 410073, China)

Abstract: Researches on the problem that the existing data fusion algorithms couldn't satisfy the new application requirements, ie, frequent dynamic change of network and hi demand with reliability of data communication, a structure-free and dynamic-adaptive data fusion algorithm was presented which divided the network into several grids, sensors perform data communication according to grids and adopt many-to-many data communication manner, there was no need for sensors to establish and maintain a structure. The results of simulation experiments show that this algorithm is able to adapt the dynamic change of network and offer high reliable data communication while performs the task of data fusion.

Key words: wireless sensor networks; data fusion; structure-free; dynamic-adaptive

1 引言

微机电系统、无线通信和数字电子等技术的发展促进了低成本、低功率和多功能传感器的发展，这些传感器由感知、数据处理和通信等模块组成，尺寸小并可以进行短距离无线通信，将大量传感器部署在监控区域，通过无线通信的方式相互连接，就可以组成无线传感器网络(WSN, wireless sensor networks)[1]。WSN 一般包括大量传感器节点（以下简称节点）和一个收集数据的基站，能够实时监测、感知和采集网络分布区域内的各种环境或监测对象信息，并对这些信息进行处理，然后传送到需要这些信息的用户。WSN 不需要固定网络支持，具有快速展开、抗毁性强等特点，可广泛应用于军事、工业、交通、环保等领域[2]。

在 WSN 的传统应用中，网络部署区域通常位于人无法轻易接近的远程环境中，如森林、战场、灾难现场等场合，节点一般位于感知对象周边，位置基本固定、电源不便更换，多个节点协同感知，基站往往位于网络部署区域之外。随着应用范围的

收稿日期：2011-07-05；修回日期：2012-07-22
基金项目：国家自然科学基金资助项目（60903225）
Foundation Item: The National Natural Science Foundation of China (60903225)

不断扩展，尤其是物联网的发展和应用，WSN 的应用也出现了新的模式，网络部署在日常环境如停车场、商场、仓库中，更换节点电源相对可行，节点通常随着物体移动，某个节点往往只负责感知其所对应物体及物体周围环境的相关信息，基站位于网络内部或边缘。在这些新应用中，WSN 具有网络动态变化频繁、对数据通信的可靠性要求高等特点。另一方面，WSN 的一些本质特性在这些新应用中并没有改变，由于节点体积受限、使用有限电源、采用无线通信方式等原因，节点的能量、通信距离和带宽等资源都受到限制，所以，如何节约能量，提高通信效率仍然是需要面临的问题。

数据融合是 WSN 解决资源受限问题所普遍采用的一种技术，它的基本思想是：当节点收到其他节点的数据后，不是分别转发出去，而是对这些数据进行融合处理，将融合后的结果转发出去，从而减少数据通信量[3]。在 WSN 中，数据通信所消耗的能量要大大高于数据采集和数据处理所消耗的能量。所以，减少数据通信量就能够有效降低能量的消耗，同时还能在一定程度上避免无线信道中的数据碰撞，从而提高通信效率。

在 WSN 新应用中，仍然可以采用数据融合节约能量和提高通信效率。本文针对 WSN 新应用中所出现的新需求，提出了一种无结构动态适应数据融合 (SFDA, structure-free and dynamic-adaptive data fusion) 算法，该算法不需要在传感器之间建立和维护数据通信的结构、采用“多对多”的数据通信方式，以适应网络的动态变化，提供高可靠性的数据通信。

2 相关工作

WSN 中的数据融合就是在数据从节点向基站传输的过程中，由中间节点对来自多个节点的数据进行融合[4]。它需要完成 2 项工作：数据收集和融合处理。根据应用的不同，数据收集的方式可以分为事件触发式、查询式和周期性汇报式。融合处理则包括数据计算、数据分组合并或数据压缩等方式。

现有的数据融合算法可以归纳为基于平面结构、基于分簇结构、基于树状结构、基于链路结构、基于混合结构和无结构几大类。下面分别简单介绍。

定向扩散(directed diffusion)[5]和 SPIN (sensor protocol for information via negotiation)[6] 是基于平面结构算法的典型代表。在定向扩散算法中，基站周期性地向邻近节点广播兴趣消息，兴趣包括任务类型、目标区域、时间戳等参数。每个节点维护一个兴趣列表，在接收到兴趣消息后，将兴趣的信息保存在兴趣列表中，以建立该节点指向基站的梯度关系，并向邻居节点转发兴趣消息。这样，随着兴趣消息的转发，每个收到兴趣消息的节点都能建立起指向基站的梯度。当节点收集到与兴趣相匹配的数据时，可以通过梯度信息将数据由其他节点依次转发至基站。SPIN 是一组基于协商的算法，节点在发送数据之前与其他节点进行协商，从而建立数据传输的路径，然后再按路径向基站发送数据。

LEACH(low energy adaptive clustering hierarchy)[7]是一种经典的基于分簇结构算法，它按照轮周期性运行，每轮都包括设置和稳定 2 个阶段。在设置阶段，节点以自组织的方式随机选择一定数目的簇首，然后，簇首广播自己成为簇首的消息，其他节点依据接收该消息的信号强度选择加入一个簇。在稳定阶段，簇成员将数据发送至簇首，由簇首对数据进行融合处理后将结果发送至基站。很多分簇算法在 LEACH 的基础上对簇首选择方法和分簇的方式进行改进，例如，HEED(hybrid energy-efficient distributed clustering)[8]在簇首选择和分簇的过程中综合考虑节点剩余能量和节点与邻居节点的邻近性或节点的度，使剩余能量较高的节点更有机会成为簇首，并能够平衡簇间的负载。

基于树状结构算法通常以基站为根节点，将网络中的节点组织成树状结构，数据沿着该结构向基站传输，并由非叶子节点进行数据的融合处理。这类算法的研究主要集中在如何建立树状结构方面，例如文献[9]所提到的 3 种方法：GIT(greedy incremental tree)、SPT(shortest paths tree)和 CNS (center at nearest source)。

基于链路结构算法的代表是 PEGASIS (power-efficient gathering in sensor information systems)[10]，它将网络中的所有节点连接成一条相邻节点之间距离最短的链路，然后随机选择一个节点作为首领，从链路的两端开始，节点依次向首领节点方向发送数据，中间节点将接收到的数据进行融合处理，并将融合结果发送至下一节点，最终由首领节点将数据发送至基站。

基于混合结构算法结合了上述结构中的 2 种或多种。例如，GSEN(group-based sensor network)[11]

融合了分簇和链路 2 种结构，将每个簇内的节点构造成链路，簇首之间也形成高一级的链路，MLC (multi-level clusteirng)[12]则融合了分簇和树状 2 种结构，在簇首和基站之间构造树状结构。

除了上述基于结构的数据融合算法外，也有文献对无结构的数据融合算法进行了研究。文献[13]针对事件触发式数据收集，首次提出无结构数据融合算法，在该算法中，需要发送数据的节点采用任意多播机制将数据发送至某个可进行融合处理的节点。文献[14]同样针对事件触发式数据收集，先动态选择一些节点担任融合节点，再由融合节点从其他节点收集数据进行融合后发送至基站。

3　相关模型及问题描述

3.1　网络模型

网络模型的建立是为了对算法的前提进行限定。假设 WSN 的部署区域是一个边长为 L、在二维坐标系中起始点（左下角顶点）坐标为(O_x,O_y)的二维正方形区域，节点均匀分布而且分布密度较大。除此之外，网络还具有如下性质：1)基站位于网络内部或边缘、位置固定、坐标为(BS_x,BS_y)；2)所有节点都是同构的，节点可以通过定位设备或定位算法等方法获得自己的位置信息；3)节点可以在网络中移动或退出网络，也可以有新的节点加入网络；4)各节点以及基站之间时间同步；5)节点的通信距离可控，节点能够根据通信距离调整发射功率；6)数据收集采用周期性汇报式，也就是说，每过一段时间，所有节点都向基站发送数据。

3.2　能耗模型

由于数据通信的能耗占传感器节点总能耗的绝大部分，远远大于其他部分的能耗，因此，和同类算法一样，本文只考虑数据传输和融合数据的能耗，节点获得位置信息的能耗不考虑。

本文采用与 LEACH 相同的能耗模型，如果发送距离小于能耗模型阈值 d_0，采用自由空间模型，否则，采用多路径衰减模型。节点发送和接收数据的能耗公式分别为

$$E_{\mathrm{Tx}}(l,d)=E_{\mathrm{Tx\text{-}elec}}(l)+E_{\mathrm{Tx\text{-}amp}}(l,d)=\begin{cases} lE_{\mathrm{elec}}+le_{\mathrm{fs}}d^2, & d<d_0 \\ lE_{\mathrm{elec}}+le_{\mathrm{mp}}d^4, & d\geq d_0 \end{cases} \tag{1}$$

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

其中，l 是数据的长度即比特数，d 是数据发送距离，$E_{\mathrm{Tx\text{-}elec}}(l)$是无线收发电路发送长度为 l 的数据的电路能耗，$E_{\mathrm{Tx\text{-}amp}}(l,d)$是将长度为 l 的数据发送至距离 d 的放大器能耗，E_{elec} 是无线收发电路发送或接收单位长度数据的电路能耗，e_{fs} 和 e_{mp} 分别是自由空间模型和多路径衰减模型的放大器能耗，$E_{\mathrm{Rx\text{-}elec}}(l)$是无线收发电路接收长度为 l 的数据的电路能耗。

节点对 k 个长度为 $l$bit 的数据进行融合处理所需要消耗的能量为

$$E_{\mathrm{A}}(k,l)=klE_{\mathrm{DA}} \tag{3}$$

其中，E_{DA} 是融合单位长度数据所需的能耗。

3.3　问题描述

与传统应用相比，WSN 的新应用尤其是其在物联网中的应用具有一些新的特点，主要表现在以下 2 个方面。

1) 节点的动态加入或退出以及节点在网络中的移动，导致网络动态变化更加频繁。传感器往往附着在物体表面或安装在物体内部，而这些物体往往具有较高的移动性。如果物体移动，传感器就会随之移动，物体从网络之外进入网络内、从网络内移动至网络外或者在网络内的移动以及传感器失效等都会造成网络的变化。

2) 对数据通信的可靠性要求更高。在传统应用中，多个节点协同工作，某些节点的数据传输失败并不会对网络的整体性能造成很大影响。而在新应用中，一个传感器通常只负责感知和收集其所对应物体及物体周围环境的相关信息。这就意味着，如果某个节点的数据无法传送至基站，就无法获得该节点对应物体的相关信息，从而产生不利的影响。

节点直接将数据发送至基站的方式虽然可以提高数据通信的可靠性，但是，由于节点的通信距离有限，网络的规模和可扩展性都会受到限制，而采用泛洪的方法则会出现“信息爆炸”。上面所介绍的各类基于结构的数据融合算法也都不能满足上述需求。首先，这些算法需要预先在节点之间构建某种结构，一个节点在发送数据之前需要与另外一个确定的节点建立连接，然后再按照该结构传输数据。而结构的建立需要节点间相互通信，节点会因为发送和接收控制消息而消耗较多的能量，如果采用固定的结构，必然无法适应网络的动态变化，如果动态维护结构，就需要再花费额外的能量开销；其次，基于结构的数据融合算法均采用“多对一”或者“一对一”的数据通信方式，传输路径上存在一些关键节点，例如分簇结构中的簇首、树状

结构中的非叶子节点以及链路结构中处于链路中间的节点等，如果这些节点失效，就会影响其他相关节点的数据传输。现有的无结构数据融合算法主要是针对事件触发式数据收集，不适用于周期性汇报式数据收集，而且这些算法也采用“多对一”或者“一对一”的数据通信方式，同样无法提供高可靠性的数据通信。

为了满足这些新应用需求，本文针对周期性汇报式数据收集的无线传感器网络，提出SFDA这种新的数据融合算法，在实现数据融合以节约能耗和提高通信效率的同时，还能够适应网络的动态变化、保证高可靠性的数据通信。

4 SFDA算法设计

SFDA是一种无需在节点之间建立和维护结构、采用“多对多”数据通信方式的数据融合算法，它按轮运行，既可以按照固定的周期运行，也可以由基站随时重新发起新的一轮，每轮包括设置阶段和稳定阶段。在设置阶段，网络被划分为多个正方形栅格，各个节点确定自己所属的栅格、传输数据的目的地和传输距离。在稳定阶段，各个节点按照一定的时序发送数据，如果基站位于节点所属栅格的邻近栅格，节点就直接将数据发送至基站，其他栅格内的节点则从邻近栅格中选择一个作为下一跳目的地，一个栅格内的所有节点分别接收其上一跳栅格所有节点发送的数据并和自身的数据进行融合处理，再将融合结果发送至下一跳目的地，从而使节点的数据能够通过一跳或多跳传输到达基站。

4.1 设置阶段

在设置阶段，首先由各个节点向基站发送自己的位置坐标、节点标识等相关信息，基站在收集完所有节点的信息后，就能够获得当前网络的节点总数和节点分布信息，再结合节点可能失效的概率以及节点最大通信距离确定栅格的大小也就是栅格边长，然后划分栅格，确定每个栅格内节点的下一跳目的地，为每个栅格及栅格中的节点分配通信时隙。

栅格边长是SFDA算法的一个参数，其取值由具体的应用决定，在同一应用中，每轮的栅格边长也可以由基站动态确定。确定栅格边长需要考虑2个因素：一个是网络的节点分布和动态性，必须使每个栅格中均分布有可参与数据通信的节点以提高数据通信的可靠性；一个是节点的通信距离，必须保证节点能够完成下一跳数据传输。前一个条件要求栅格边长不能小于某个值，也就是限定了栅格边长的下界，后一个条件要求栅格边长不能大于某个值，也就是限定了栅格边长的上界。然后，可以在该下界和上界所确定的范围内选择一个合适的取值确定为栅格的边长。

假设当前网络中均匀分布有N个节点，需要确定的栅格边长为B，则栅格的总数G为$(L/B)^2$个，每个栅格中节点数目的期望值N_G为N/G，每个节点失效的概率相同并设为P_O，那么每个栅格可参与数据通信的节点（也就是有效节点）总数的期望值N_T为

$$N_T = N_G - N_G P_O \tag{4}$$

为了提高数据通信的可靠性，所以需要保证每个栅格中都分布着有效节点，也就是要求N_T大于1，所以栅格边长需要满足：

$$B \geq L/\sqrt{N(1-P_O)} \tag{5}$$

另外，在数据通信时，一个栅格内的节点需要向邻近某个栅格的所有节点发送数据，而节点与邻近栅格节点最大距离的情况就是这2个节点分别位于2个栅格对角线的两端，如图1所示。设节点最大通信距离为R，则栅格边长还应满足：

$$B \leq \sqrt{2}R/4 \tag{6}$$

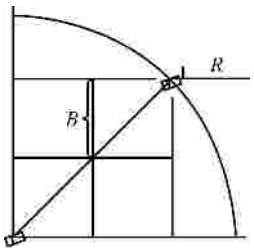


图1 相邻栅格节点最大距离示意

为了提高对网络的动态适应性，算法基于网络中节点的当前状态和动态变化趋势来预估其未来的状态。需要说明的是，式(5)给出的栅格边长下界是在节点均匀分布且分布密度较大以及每个节点失效概率均相等的条件下确定的，而式(6)给出的栅格边长上界是在所有节点同构也就是最大通信距离相等的条件下确定的，因而，只要网络能够满足上述条件，而且栅格边长能够同时满足式(5)和式(6)，算法就可以保证每个栅格中都分布着有效节点而且节点能够完成下一跳数据传输。在本文限定范

围之外的应用中，如果出现节点分布不均匀、节点失效概率不同或节点异构的情况，即使满足式(5)和式(6)的栅格边长也无法保证每个栅格中都分布着有效节点或节点能够完成下一跳数据传输，针对这些情况，将在下一步工作中进行研究并分别提出相应的解决方案，扩展算法以提高算法的健壮性和扩大算法的应用范围。还有，算法的性能如对网络动态变化的适应能力和能量消耗都与栅格边长的取值相关，未来，将通过定量的方法分析栅格边长的最佳取值，以优化算法的性能。

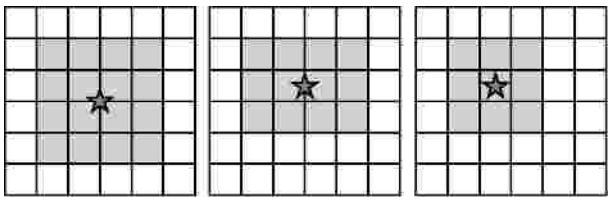
基站的另一个工作是确定每个栅格内节点的下一跳目的地以及每个栅格和栅格内节点传输数据的时隙。首先给出几个相关定义。

定义 1　（邻近栅格）和某个栅格有公共边或公共顶点的栅格称为该栅格的邻近栅格。

定义 2　（临近栅格）基站位于内部或边缘的栅格以及基站位于邻近栅格内部或边缘的栅格称为临近栅格。

定义 3　（临近区域）所有临近栅格组成的区域称为临近区域。

用二元组(G_x,G_y)作为栅格标识，其中，G_x是栅格横标识，G_y是栅格纵标识，坐标为(x, y)的位置所处的栅格为$(\lceil (x-O_x)/B \rceil, \lceil (y-O_y)/B \rceil)$，左下角栅格的标识为(1,1)、右上角栅格的标识为$(L/B, L/B)$，基站用(0,0)表示，基站所在栅格的标识为$(\lceil (BS_x-O_x)/B \rceil, \lceil (BS_y-O_y)/B \rceil)$。临近栅格和临近区域的分布分为基站位于某个栅格顶点、边界（不包括栅格顶点）或内部 3 种情况，如图 2 所示，其中，五角星代表基站，阴影区域代表临近区域。



(a)基站位于栅格顶点　(b)基站位于栅格边界　(c)基站位于栅格内部

图 2　临近栅格和临近区域示意

用$[LB_x{:}RT_x,LB_y{:}RT_y]$表示临近区域，其中，$LB_x$和$RT_x$分别为左下角和右上角临近栅格的横标识，$LB_y$和$RT_y$分别为左下角和右上角临近栅格的纵标识，可以通过算法 1 确定网络的临近区域。

算法 1　确定临近区域的算法伪码

$(BS\_G\_x, BS\_G\_y) \leftarrow$
$([(BS\_x-O\_x)/B], [(BS\_y-O\_y)/B])$;
If　$BS\_G\_x == (BS\_x-O\_x)/B$　or
　$BS\_G\_y == (BS\_y-O\_y)/B$
　If　$BS\_G\_x == (BS\_x-O\_x)/B$　and
　　$BS\_G\_y == (BS\_y-O\_y)/B$
$[LB\_x: RT\_x, LB\_y: RT\_y] \leftarrow [BS\_G\_x-1:$
$BS\_G\_x+2, BS\_G\_y-1: BS\_G\_y+2]$;
　End if
　If　$BS\_G\_x \neq (BS\_x-O\_x)/B$　and
　　$BS\_G\_y == (BS\_y-O\_y)/B$
$[LB\_x: RT\_x, LB\_y: RT\_y] \leftarrow [BS\_G\_x-1:$
$BS\_G\_x+1, BS\_G\_y-1: BS\_G\_y+2]$;
　End if
　If　$BS\_G\_x == (BS\_x-O\_x)/B$　and
　　$BS\_G\_y \neq (BS\_y-O\_y)/B$
$[LB\_x: RT\_x, LB\_y: RT\_y] \leftarrow [BS\_G\_x-1:$
$BS\_G\_x+2, BS\_G\_y-1: BS\_G\_y+1]$;
End if
Else
$[LB\_x: RT\_x, LB\_y: RT\_y] \leftarrow [BS\_G\_x-1:$
$BS\_G\_x+1, BS\_G\_y-1: BS\_G\_y+1]$;
End if

临近栅格将基站作为目的地，其他栅格(G_x, G_y)选择距离最近的临近栅格(A_x,A_y)作为目的地，并从邻近栅格中选择一个栅格(N_x,N_y)作为下一跳，确定所有栅格目的地和下一跳的算法分别描述如下。

算法 2　确定栅格目的地的算法伪码

$G\_x \leftarrow 1$;
While　$G\_x \le L/B$　do
　$G\_y \leftarrow 1$;
　While　$G\_y \le L/B$　do
　　If　$LB\_x \le G\_x \le RT\_x$　and
　　$LB\_y \le G\_y \le RT\_y$
　　$(A\_x, A\_y) \leftarrow (0,0)$;
　　End if
　　If　$G\_x < LB\_x$　and　$G\_y < LB\_y$
　　$(A\_x, A\_y) \leftarrow (LB\_x, LB\_y)$;
　　End if
　　If　$LB\_x \le G\_x \le RT\_x$ and　$G\_y < LB\_y$
　　$(A\_x, A\_y) \leftarrow (G\_x, LB\_y)$;
　　End if
　　If　$G\_x > RT\_x$　and　$G\_y < LB\_y$
　　$(A\_x, A\_y) \leftarrow (RT\_x, LB\_y)$;

End if
If $G\_x<LB\_x$ and $LB\_y \le G\_y \le RT\_y$
$(A\_x, A\_y) \leftarrow (LB\_x, G\_y)$;
End if
If $G\_x > RT\_x$ and
$LB\_y \le G\_y \le RT\_y$
$(A\_x, A\_y) \leftarrow (RT\_x, G\_y)$;
End if
If $G\_x < LB\_x$ and $G\_y > RT\_y$
$(A\_x, A\_y) \leftarrow (LB\_x, RT\_y)$;
End if
If $LB\_x \le G\_x \le RT\_x$ and $G\_y > LB\_y$
$(A\_x, A\_y) \leftarrow (G\_x, RT\_y)$;
End if
If $G\_x > RT\_x$ and $G\_y > LB\_y$
$(A\_x, A\_y) \leftarrow (RT\_x, RT\_y)$;
End if
$G\_y \leftarrow G\_y + 1$;
End while
$G\_x \leftarrow G\_x + 1$;
End while

算法 3 确定栅格下一跳的算法伪码

$G\_x \leftarrow 1$;
While $G\_x \le L/B$ do
$G\_y \leftarrow 1$;
While $G\_y \le L/B$ do
If $A\_x == 0$ and $A\_y == 0$
$(N\_x, N\_y) \leftarrow (0,0)$;
Else
$H\_x = (A\_x == G\_x \ ?0:$
$(A\_x - G\_x)/|(A\_x - G\_x)|)$;
$H\_y = (A\_y == G\_y \ ?0:$
$(A\_y - G\_y)/|(A\_y - G\_y)|)$;
$(N\_x, N\_y) \leftarrow$
$(G\_x + H\_x, G\_y + H\_y)$;
End if
$G\_y \leftarrow G\_y + 1$;
End while
$G\_x \leftarrow G\_x + 1$;
End while

在确定了每个栅格的下一跳目的地之后，整个网络就形成了以栅格为单位的从栅格到基站的有向连通结构，每个栅格都能够直接或间接与基站连通，如图 3 所示。

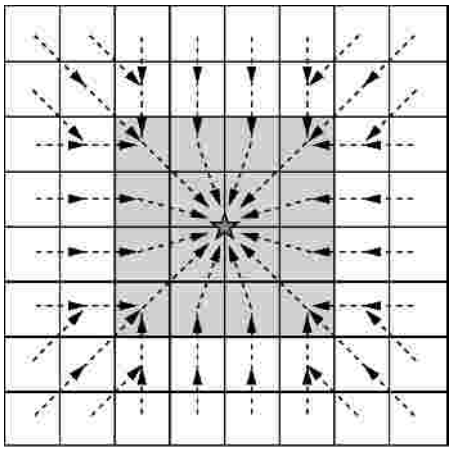


图 3 从栅格到基站的有向连通结构示意

为了能够实现数据的融合并提高通信效率，还需要确定通信时隙，在同一条链路上，按照由远及近的原则以栅格为单位发送数据，从而使中间栅格内的节点能够进行数据的融合处理。另外，如果相互之间不会产生干扰，不同链路的多个栅格可以同时进行数据的传输。

基于所收集的节点信息和栅格划分的结果，基站能够获得每个栅格中所分布的节点信息，从而可以为每个栅格以及栅格内的节点分配时隙。当然，分配时隙的时候必须要考虑网络的动态变化。

同一个栅格内节点的时隙先后顺序可以按照一定规则确定，如按照节点标识从小到大排列，设栅格(G_x,G_y)内的 i 个节点标识排列为 N_1，N_2，…，N_i，设基站为该栅格分配的时隙表示为(G_x，G_y，$T0$，TN，$T1$，i，N_1，N_2，…，N_i)，其中，$T0$ 为该栅格时隙开始的时刻，TN 为单个节点发送数据所需时间的期望值，$T1$ 为预留时隙长度。设置预留时隙是为了提高算法的动态适应能力，因为可能出现节点移动等情况使得某些栅格内分布的节点发生变化。栅格时隙的时间长度、节点 $N_k(1 \le k \le i)$时隙的开始时刻 Tk 和预留时隙开始时刻 Tr 分别为

$$T_G = iT_N + T_1 \tag{7}$$

$$T_k = T_N(k-1) + T_0 \tag{8}$$

$$T_r = iT_N + T_0 \tag{9}$$

预留时隙可以根据应用需求由基站动态确定，不同轮的预留时隙可以不同，同一轮中的各个栅格的预留时隙也可以不同。

上述工作完成之后，基站就向网络中的所有

节点广播当前轮数、部署区域起始点坐标、基站位置坐标、栅格边长、每个栅格的下一跳目的地、每个栅格以及栅格中节点的时隙等信息以发起一轮数据收集，节点接收广播消息并将这些信息保存在本地。

4.2　稳定阶段

在稳定阶段，每个节点首先根据本轮基站广播的相关信息以及自己的位置确定节点所属栅格、下一跳目的地、数据传输的距离、需要接收的上一跳栅格标识以及发送数据时隙等信息，在收发数据时，节点接收上一跳栅格内的所有节点发送的数据并和自己感知的数据进行融合处理，最后在相应的时隙中向下一跳目的地发送数据。和其他同类算法一样，SFDA 算法并不对采用何种融合处理方式进行限定，而是在应用中根据具体的需求，采用合适的融合处理方式。

首先，每个节点如节点 j 根据自己当前的位置坐标(j_x,j_y)确定自己所处栅格的标识，然后从保存的基站广播信息中查找该栅格对应的下一跳目的地和所有以其为下一跳目的地的栅格。如果下一跳目的地的标识为$(0,0)$，则说明该节点需要直接向基站发送数据，它的最小数据传输距离 D_min 为节点与基站之间的距离，否则，它的下一跳目的地就是某个栅格，设该栅格的标识为(G_x,G_y)，则该节点的最小数据传输距离为它与这个栅格的 4 个顶点距离中的最大值。在理想条件下，最小数据传输距离就能够满足数据传输的需求，但在实际应用中，数据的实际传输距离会小于节点发送数据时所使用的数据传输距离，因而，需要在最小数据传输距离之上再增加一定的预留距离 D_r，以确保数据传输的可靠性。同样的，预留距离的大小也取决于具体的应用，每个节点的预留距离也可由节点分别确定。节点 j 确定数据传输距离 D 的算法描述如下：

算法 4　节点确定数据传输距离的算法伪码

$(G\_x, G\_y) \leftarrow getNexthop(j\_x, j\_y)$;
If $G\_x == 0$ and $G\_y == 0$
　$D\_\min \leftarrow \sqrt{(BS\_x - j\_x)^2 + (BS\_y - j\_y)^2}$;
Else
　$(P1\_x, P1\_y) \leftarrow$
　　$(B(G\_x - 1) + O\_x, B(G\_y - 1) + O\_y)$;
　$(P2\_x, P2\_y) \leftarrow$
　　$(BG\_x + O\_x, B(G\_y - 1) + O\_y)$;
　$(P3\_x, P3\_y) \leftarrow$
　　$(BG\_x + O\_x, BG\_y + O\_y)$;
　$(P4\_x, P4\_y) \leftarrow$
　　$(B(G\_x - 1) + O\_x, BG\_y + O\_y)$;
　$d1 \leftarrow \sqrt{(P1\_x - j\_x)^2 + (P1\_y - j\_y)^2}$;
　$d2 \leftarrow \sqrt{(P2\_x - j\_x)^2 + (P2\_y - j\_y)^2}$;
　$d3 \leftarrow \sqrt{(P3\_x - j\_x)^2 + (P3\_y - j\_y)^2}$;
　$d4 \leftarrow \sqrt{(P4\_x - j\_x)^2 + (P4\_y - j\_y)^2}$;
　$D\_\min \leftarrow MAX(d1, d2, d3, d4)$;
End if
$D \leftarrow D\_\min + D_r$;

节点接收到其他节点发送的数据消息时，首先检查该消息是否来自其上一跳栅格，如果不是，不接收该消息，否则，接收该消息。节点在自己发送数据之前将所有接收的数据与自己感知的数据进行融合处理，融合处理的方式可以根据应用需求选择。

发送数据时，各个节点根据时隙信息在相应的时隙内发送数据。首先，节点从所属栅格的时隙信息中查找自己的节点标识，如果节点标识在时隙信息中，则节点计算自己时隙的开始时刻，并在该时刻到来时检测信道是否可用，如果信道可用，则发送数据，如果信道不可用，节点先随机生成一个延迟，待该延迟结束时再检测信道是否可用，然后决定是否发送数据，如果还不能发生数据，则重复上述操作，直至完成数据发送或者节点的时隙结束；如果节点标识不在所属栅格的时隙信息中，该节点就在栅格的预留时隙内发送数据，在栅格的预留时隙开始时，节点首先随机生成一个延迟，待该延迟结束时再检测信道是否可用，如果信道可用，则开始发送数据，否则，就重复上述随机等待，检测信道，决定是否发送数据的操作，直至完成数据发送或者栅格时隙结束。算法 5 描述了节点在相应时隙中发送数据 $Data$ 的操作。

算法 5　节点发送数据操作的算法伪码

$isTimesliceEnd \leftarrow testTimeslice(T_0, T_G)$;
$isChannelIdle \leftarrow false$;
$isDataHavebeensent \leftarrow false$;
If $hasTimeslice == true$ and
　$isTimesliceEnd == false$
　$isChannelIdle \leftarrow testChannel()$;
　If $isChannelIdle == true$
　　$sendData(Data, D)$;
　　$isDataHavebeensent \leftarrow true$;

```
    End if
End if
While  isTimesliceEnd == false  and
   isChannelIdle == false and
   isDataHavebeensent == false do
      timeforWait ← random_time();
      sleep(timeforWait);
      isTimesliceEnd ← testTimeslice(T0,TG);
   If  isTimesliceEnd == false
      isChannelIdle ← testChannel();
      If  isChannelIdle == true
         sendData(Data,D);
         isDataHavebeensent ← true;
      End if
   End if
End While
```

节点在发送数据时，附带将节点标识、所属栅格标识等信息一起发送。另外，为了提高动态适应性，所有在本轮中未发送数据的节点都定期检测自己的位置并确定所属栅格信息，如果节点的所属栅格改变，则节点立即更新相关信息，并准备在相应时隙发送数据，从而尽量确保其仍然能够发送数据。

5 SFDA 算法分析

本节主要从网络动态适应性和能量消耗 2 个方面对 SFDA 算法进行分析。

在基于结构的算法中，必须预先在节点之间建立某种结构，然后再按照结构传输数据，这种结构可以看作是节点之间的一个有向连通图，每个节点与其下一跳节点连接，也就是说，每个节点在发送数据之前都必须清楚自己需要将数据发送至哪个节点、数据传输距离是多远。而在 SFDA 算法中，设置阶段将网络划分为多个正方形栅格后，节点的数据传输就可以按照栅格为单位分类进行，每个节点（直接向基站发送数据的节点除外）只需知道将数据发送至哪个栅格，而不需要知道接收数据的是哪个或哪些节点，也就是说，节点在发送数据之前，只需要知道数据的传输距离，而不需要知道目的地节点，也就没有基于结构算法所建立的那种由每一个节点与另外一个节点连接而构建的确定结构。如果网络中的节点动态变化，在基于结构的算法中，只要结构还没有更新，节点之间的连接就不会改变，也就对动态变化做出反应。而在 SFDA 算法中，节点则可以通过定期检测来更新自己的下一跳栅格。

WSN 的新应用中，网络动态变化的原因主要分为节点从网络中退出、节点在网络中移动以及新的节点加入网络 3 种。

节点从网络中退出的情况有 2 种，一种是节点移动到了网络部署区域之外，另一种则是节点虽然还在网络中但由于节点故障、能量耗尽等因素造成节点无法正常工作。在采用“多对一”或“一对一”数据通信方式的数据融合算法中，如果某些节点退出网络，则可能会造成以这些节点为转发节点的所有节点都无法将数据发送至基站。而 SFDA 以栅格为单位进行数据通信，是一种“多对多”的数据通信方式，这样，即使某些节点退出网络，只要栅格中还分布有节点，则仍然可以实现数据的有效收集。

对于节点在网络中移动的情况，SFDA 使节点定期检测位置信息，如果节点没有离开原来的栅格，则不会产生任何影响，否则，节点能够及时更新相关信息从而仍然可以发送数据，所以，节点在网络中的移动对数据的收集影响很小。而在基于结构的算法中，如果节点的移动破坏了所建立的结构，相关节点的数据都无法发送至基站，从而会影响网络的数据收集。

因为节点加入网络而引起的网络动态变化对 SFDA 的影响与其他算法类似，只有在新一轮开始之前加入网络的节点才能够发送数据。

在能量消耗方面，“多对多”的数据通信会增加节点的数据接收量，额外消耗一定的能量，实际上就是以能量为代价换取数据通信的可靠性。虽然如此，SFDA 以栅格为单位对进行数据的“多对多”传输，某个栅格的节点只将数据发送至一个邻近栅格内的节点，限制了重复接收数据节点的个数，另外，节点发送数据的距离只能覆盖其邻近栅格，限制了节点发送数据的距离，从而尽量降低了额外的能量开销。再者，算法只增加了数据接收次数而并没有增加数据发送次数，考虑到接收数据的能耗大大小于发送数据的能耗，所以，SFDA 的额外能耗并不高。

由于 SFDA 所针对的是 WSN 的新应用，这些网络部署在日常环境中，通过人工或其他方式为节点更换电源或为电源充电具有一定的可行性。所以，在这些应用中，节能位于次要位置，数据通信的可靠性才是首要考虑，因而，SFDA 对网络动态变化的适应能力强、可实现高可靠的数据通信，能够以较小的能耗代价满足这种应用需求。

需要说明的是，SFDA 算法采用了不同于现有数据融合算法的数据传输方式，因为任何 2 种数据融合算法可以被同一个无线传感器网络分别采用但却不能在一次数据收集中同时使用，所以，对于同一个无线传感器网络，可以采用 SFDA 算法的数据传输方式，也可以采用包括基于结构算法在内的其他数据传输方式，但是，不能在同一次数据收集中同时采用 2 种方式。此外，SFDA 算法的“无结构”是指不需要为数据传输而在节点之间建立每一个节点与另外一个节点连接的结构，并非指组网结构，所以，在有结构的无线传感器网络中，SFDA 算法并不会影响网络自身的数据传输，网络的结构对 SFDA 算法也没有影响。

6　仿真实验及结果分析

本文在 MATLAB 平台上进行对比仿真实验，考察 SFDA 算法对网络动态变化的适应能力以及能量消耗情况，并与其他几种算法进行比较。因为现有的无结构数据融合算法都是针对事件触发式，而在基于平面结构算法中，定向扩散算法是基于查询式、SPIN 算法是基于事件触发式，与本文 SFDA 算法所针对的周期性汇报式数据收集有所不同，所以，从基于分簇结构、树状结构和链路结构算法中分别选择一种具有代表性的、可应用于周期性汇报式数据收集的算法与本文的算法进行对比实验，它们分别是 LEACH 算法、GIT 算法和 PEGASIS 算法。

6.1　仿真实验环境

网络部署区域边长、基站位置、节点总数、SFDA 的栅格边长都作为可设置参数，LEACH 的每轮簇首期望数目依据文献[7]确定为节点总数的 5%，以使 LEACH 达到较好的性能。融合处理能够将若干个长度相同的数据消息分组融合为一个仍为该长度的数据消息分组，其他仿真实验参数的设置如表 1 所示。

表 1　　其他仿真实验参数的设置

参数	取值
网络部署区域起始点坐标	(0,0)
节点最大通信距离	300m
数据消息分组长度	800bit
控制消息分组长度	200bit
无线收发电路能耗	50nJ/bit
自由空间模型放大器能耗	$10pJ/bit/m^2$
多路径衰减模型放大器能耗	$0.001\ 3pJ/bit/m^4$
传输模型阈值	86.2m
融合单位长度数据能耗	5nJ/bit

6.2　仿真实验结果分析

首先比较各个算法对网络动态变化的适应能力。由于节点新加入网络对几种算法的影响相同，因此，仿真实验只考察算法对节点在网络中移动或退出网络所造成的网络动态变化的适应能力。如果节点因为在网络中移动或退出网络而导致节点无法正常收发数据，则称该节点失效。因为几种对比算法都是针对节点位置固定而非节点移动的应用场景而设计的，所以实验不考虑导致节点失效的具体原因，而只设置节点的失效概率，每次对比实验中，各个算法的初始状态、失效的节点（失效节点总数以及哪些节点失效）都是相同的，从而使实验结果具有可比较性。分别进行多次对比仿真实验，每次对比仿真实验中几种算法的节点失效概率相同，统计仿真实验中产生的连带失效节点的数目，所谓连带失效节点是指节点自身没有失效但是因为其他节点失效而无法将数据发送至基站的节点。相同条件下，算法的连带实效节点的数目越小，就说明算法越能够适应网络的动态变化。

每次对比仿真中节点的失效概率均相同，网络部署区域边长为 200m，基站位置为(100, 100)，网络中均匀分布有 1 200 个节点，SFDA 的栅格边长为 20m，每个算法的仿真分别进行 100 轮，节点失效概率为 0.01~0.09 和 0.1~0.9 的仿真结果分别如图 4 和图 5 所示。可以看出，在相同的条件下，SFDA、LEACH、GIT 和 PEGASIS 的连带失效节点平均数依次递减，而且，SFDA 的连带失效节点平均数都要远远小于其他几种算法。在节点失效概率小于 0.4 时，SFDA 几乎没有连带失效节点产生，随着节点失效概率的增大，SFDA 的连带失效节点也只是缓慢增加，而其他几种算法的连带失效节点则较多，尤其是 GIT 和 PEGASIS，在节点失效概率较大时，几乎没有节点能够成功将数据传输至基站，主要原因是这些算法的关键节点较多，而一个关键节点的失效就可能影响其他很多节点的数据传输。仿真结果证明，使用 SFDA 算法，网络动态变化对网络数据收集的影响比使用 LEACH、GIT 和 PEGASIS 要小得多，SFDA 对网络动态变化的适应能力更强、能够提供更为可靠的数据通信。

对于 SFDA 算法，在同一应用场景中，栅格边长取值的不同以及基站所处位置的不同都会给 SFDA 算法对网络动态变化的适应能力造成影响。保持上述实验场景的其他条件不变，分别设置不同

的栅格边长和改变基站的位置，每次仿真 100 轮，考察连带失效节点平均数，分析栅格边长和基站位置对 SFDA 算法的动态变化适应性的影响。

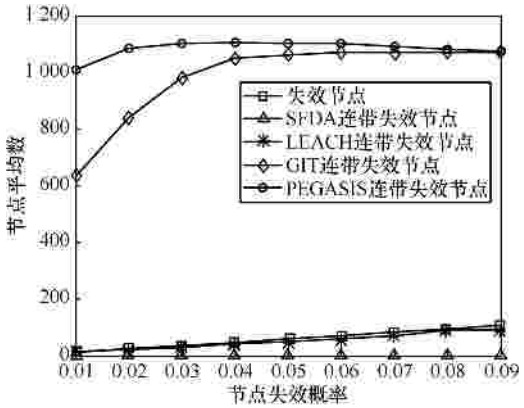


图 4 节点失效概率为 0.01~0.09 的仿真结果

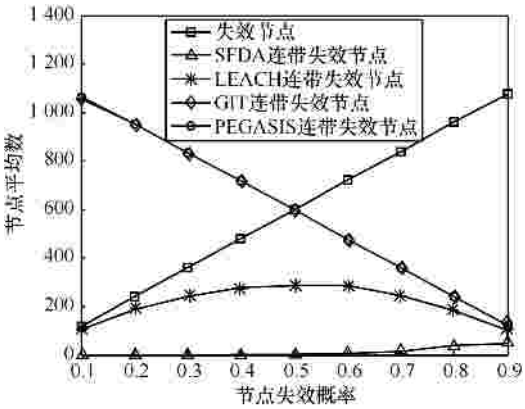


图 5 节点失效概率为 0.1~0.9 的仿真结果

栅格边长分别为 20m、25m 和 40m 时，在不同节点失效概率的情况下，实验中网络产生的连带失效节点平均数如图 6 所示。栅格边长的取值越大，产生的连带失效节点越少，这就说明，栅格边长越大，SFDA 算法对网络动态变化的适应能力越强。在 SFDA 算法中，连带失效节点的产生原因是处于中间的栅格内的所有节点都失效，从而造成以中间栅格内的节点为中继节点的其他相关栅格内的有效节点无法完成数据传输而失效。栅格边长取值越大，栅格就越大，在节点均匀分布的情况下，栅格内的节点也就越多，在节点失效概率相同时，栅格内所有节点都失效的可能性就越小，此外，栅格边长越大，网络中的栅格就越少，相同节点将数据发送至基站所需的跳数就越少，数据传输失败的可能性也越小。因而，栅格边长越大所产生的连带失效节点就越少，对网络动态变化的适应能力就越强。当然，栅格边长也不是越大就越好，在上述实验条件下，如果栅格边长达到 50m，则所有的栅格都是临近栅格，也就是说，所有的节点都直接将数据发送至基站，而且栅格越大，节点发送数据的距离就越大，能耗也越多，如何确定最优的栅格边长以平衡各方面的性能将是未来研究的重点。

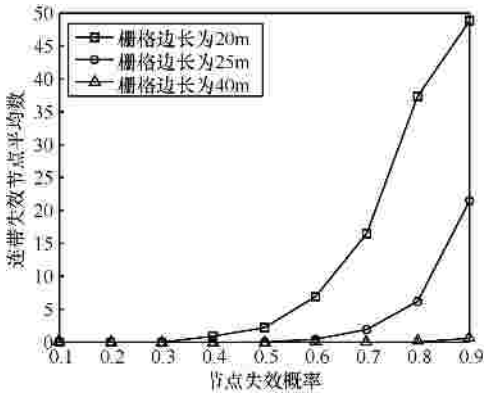


图 6 栅格边长对 SFDA 算法的动态变化适应性的影响

将基站位置分别设置为(100，100)、(100，110)和(110，110)，使基站分别位于栅格顶点、栅格边界和栅格内部，在不同节点失效概率的情况下，网络产生的连带失效节点平均数如图 7 所示。上述 3 种情况产生的连带失效节点依次递增，说明对网络动态变化的适应能力依次减弱。主要是因为网络中临近栅格数目不同而造成的，基站位于栅格顶点时临近栅格最多，位于栅格边界时次之，位于栅格内部时最少。临近栅格越多，则直接向基站发送数据的节点越多，可能出现连带失效的节点就越少。另外，临近栅格越多，为其他栅格内节点向基站传输数据提供中继服务的栅格就越多，以每个这样的栅格为中继的节点数目就越少，如果某个这样的栅格内的所有节点都失效，其所造成的连带失效节点也就越少。因此，应设置基站的位置使得临近栅格的个数尽量多，以提高网络的动态适应能力。

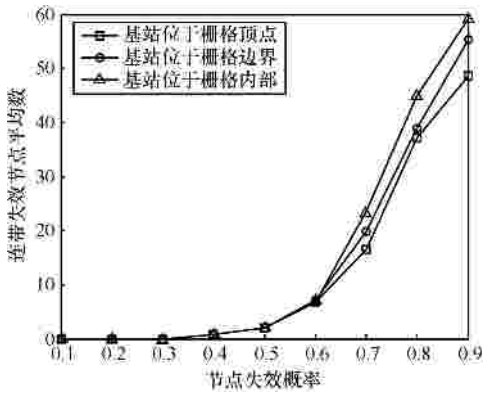


图 7 基站位置对 SFDA 算法的动态变化适应性的影响

再对比几种算法的能耗。分别为 SFDA 设置不同的栅格边长、基站位置、节点数目、网络部署区域边长，既比较几种算法的能耗，也对相关参数对 SFDA 能耗的影响进行考察。每次条件相同的仿真分别进行 100 轮，统计节点发送消息、接收消息和融合数据的能耗。最后，比较不同条件下几种算法平均每轮整个网络和每个节点的能耗，仿真结果如表 2 所示。从表中的实验结果可

表 2　　能量消耗仿真结果

算法	部署区域/mxm	节点数目	簇首数目	基站位置	栅格边长/m	网络平均能耗/J	节点平均能耗/μJ
SFDA	200×200	1 200	—	(100,100)	10	0.249 11	207.59
SFDA	200×200	1 200	—	(100,100)	20	0.621 19	517.66
SFDA	200×200	1 200	—	(100,100)	40	1.803	1 502.5
LEACH	200×200	1 200	—	(100,100)	—	1.077	897.5
GIT	200×200	1 200	—	(100,100)	—	0.427 37	356.14
PEGASIS	200×200	1 200	—	(100,100)	—	0.428 4	357
SFDA	200×200	1 200	—	(100,200)	20	0.778 59	648.83
LEACH	200×200	1 200	60	(100,200)	—	1.099 6	916.33
GIT	200×200	1 200	—	(100,200)	—	0.427 41	356.18
PEGASIS	200×200	1 200	—	(100,200)	—	0.428 83	357.36
SFDA	200×200	1 200	—	(200,200)	20	0.982 74	818.95
LEACH	200×200	1 200	60	(200,200)	—	1.132 6	943.83
GIT	200×200	1 200	—	(200,200)	—	0.427 43	356.19
PEGASIS	200×200	1 200	—	(200,200)	—	0.428 98	357.49
SFDA	200×200	800	—	(100,100)	20	0.300 04	375.05
LEACH	200×200	800	40	(100,100)	—	0.566 89	708.61
GIT	200×200	800	—	(100,100)	—	0.284 97	356.21
PEGASIS	200×200	800	—	(100,100)	—	0.285 64	357.04
SFDA	200×200	1 600	—	(100,100)	20	1.071 8	669.88
LEACH	200×200	1 600	80	(100,100)	—	1.738 9	1 086.8
GIT	200×200	1 600	—	(100,100)	—	0.569 77	356.1
PEGASIS	200×200	1 600	—	(100,100)	—	0.570 34	356.46
SFDA	120×120	1 200	—	(60,60)	20	1.119 2	932.67
LEACH	120×120	1 200	60	(60,60)	—	1.071 8	893.17
GIT	120×120	1 200	—	(60,60)	—	0.427 26	356.05
PEGASIS	120×120	1 200	—	(60,60)	—	0.427 49	356.25
SFDA	280×280	1 200	—	(140,140)	20	0.461 27	384.39
LEACH	280×280	1 200	60	(140,140)	—	1.090 3	908.58
GIT	280×280	1 200	—	(140,140)	—	0.427 53	356.28
PEGASIS	280×280	1 200	—	(140,140)	—	0.429 7	358.08

以看出，SFDA 的能耗在大多数情况下都高于 GIT 和 PEGASIS，只有在栅格边长较小时，SFDA 的能耗才低于 GIT 和 PEGASIS，但是，与 LEACH 相比，SFDA 的能耗性能在大多数情况下都更好，只有在栅格边长较大时，SFDA 的能耗才高于 LEACH。主要是因为在 GIT 和 PEGASIS 算法中，节点只需要向距离较近或距离最近的节点发送数据，所以它们的能耗性能较好。虽然 SFDA 算法的能耗性能在某些条件下不及一些能耗性能较好的基于结构算法，却仍然要优于其他一些基于结构算法。

另外，分析能耗仿真结果可以得出，在其他条件相同时，栅格边长越小能耗越小，因为栅格越小，每个栅格内的节点越少、节点传输数据的距离越短，所以接收和发送数据的能耗就越小；基站的位置越靠近网络的中心能耗越小，因为基站越靠近网络中心，负责转发数据的栅格就越少，从而降低了能量消耗；节点总数越少能耗越小，因为节点总数越少则分布在每个栅格中的节点也会越少，转发栅格内的节点接收的数据量就越小，从而减小了能耗；网络部署范围越大能耗越小，因为部署范围越大，每个栅格内的节点就越少，节点数据接收量减少，但是栅格大小没有变化，发送距离也没有变化，所以能耗也会减小。

7 结束语

本文分析了 WSN 在的一些新应用中出现的新特点——网络动态变化更加频繁、对数据通信的可靠性要求更高，针对现有数据融合算法在这些应用中的不足，提出了适用于周期性汇报式数据收集的 SFDA 算法，它是一种无需在节点之间建立和维护结构、采用“多对多”数据通信方式的无线传感器网络数据融合算法，对网络动态变化具有很强的适应能力，并能够提供高可靠性的数据通信。仿真实验结果表明，SFDA 在网络动态适应性方面具有较大的优势，额外的能耗代价也较小，能够满足新应用需求。

本文对无结构数据融合这类新颖的数据融合算法进行了探索式的研究，还有许多内容需要进一步探讨，也有较多问题亟待解决。下一步工作主要包括以下几个方面：1)针对不同形状的网络部署区域、不同的栅格形状等情况进行研究，以增强算法的适用性；2)通过定量的方法确定相关参数如栅格边长的最佳取值，以优化算法的性能；3)针对另外一些应用中可能出现的节点分布不均匀、节点失效概率不同或节点异构等情况，扩展算法以提高算法的健壮性和扩大算法的应用范围；4)研究提出节点数据传输的详细调度方法。

参考文献：

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

[2] 马祖长，孙怡宁，梅涛.无线传感器网络综述[J].通信学报，2004, 25(4): 114-124.
MA Z C, SUN Y N, MEI T. Survey on wireless sensor network[J]. Journal on Communications, 2004, 25(4): 114-124.

[3] KRISHANAMACHARI B, ESTRIN D, WICKER S. The impact of data aggregation in wireless sensor networks[A]. Proceedings of International Workshop of Distributed Event Based Systems[C]. Vienna, Austria, 2002. 575-578.

[4] RAJAGOPALAN R, VARSHNEY P K. Data-aggregation techniques in sensor networks: a survey[J]. IEEE Communications Surveys & Tutorials, 2006, 8(4): 48-63.

[5] INTANAGONWIWAT C, GOVINDAN R, ESTRIN D. Directed diffusion for wireless sensor networks[J]. IEEE/ACM Transcations on Networking, 2003, 11(1): 2-16.

[6] KULIK J, HEINZELMAN W R, BALAKRISHNAN H. Negotiation-based protocols for disseminating information in wireless sensor networks[J]. Wireless Networks, 2002, 8(1): 169-185.

[7] HEINZELMAN W R, CHANDRAKASAN A, BALAKRISHNAN H. Energy-efficient communication protocol for wireless microsensor networks[A]. Proceedings of International Conference on System Sciences[C]. San Francisco: IEEE Computer Society, USA, 2000. 3005-3014.

[8] YOUNIS O, FAHMY S. HEED: a hybrid, energy- efficient, distributed clustering approach for ad hoc sensor networks[J]. IEE Transcations on Mobile Computing, 2004, 3(4): 366-379.

[9] KRISHNAMACHARI B, ESTRIN D, WICKER S. Modeling data centric routing in wireless sensor networks[A]. Proceedings of IEEE International Conference on Information Communications New York: IEEE Computer Society, USA, 2000. 1-18.

[10] LINDSEY S, RAGHAVENDRA C. PEGASIS: power-efficient gathering in sensor information systems[A]. Proceedings of IEEE Aero-

space Conference[C]. Montana: IEEE Aerospace and Electronic Systems Society, USA, 2002. 1125-1130.

[11] TABASSUM N, URANO Y, HAQUE A. GSEN: an efficient energy consumption routing scheme for wireless sensor network[A]. Proceedings of IEEE International Conference on Networking[C]. Washington: IEEE Computer Society, USA, 2006. 117-122.

[12] OKEKE B C, LAW K L E. Multi-level clustering architecture and protocol designs for wireless sensor networks[A]. Proceedings of International Conference on Wireless Internet[C]. Hawaii USA, 2008. 1-9.

[13] CHAO C M, HSIAO T Y. Structure-free data aggregation in sensor networks[J]. IEEE Transactions on Mobile Computing, 2007, 6(8): 929-942.

[14] FAN K W, LIU S, SINHA P. Design of structure-free and energy-balanced data aggregation in wireless sensor networks[A]. Proceedings of IEEE International Conference on High Performance Computing and Communications[C]. Washington: IEEE Computer Society, USA, 2009. 222-229.

作者简介：

乐俊（1984-），男，湖北广水人，国防科学技术大学博士生，主要研究方向为无线传感器网络。

张维明（1962-），男，安徽合肥人，博士，国防科学技术大学教授、博士生导师，主要研究方向为无线传感器网络、信息系统与智能决策技术等。

肖卫东（1968-），男，湖南长沙人，博士，国防科学技术大学教授、博士生导师，主要研究方向为无线传感器网络、指挥信息系统等。

汤大权（1971-），男，天津人，博士，国防科学技术大学教授、硕士生导师，主要研究方向为无线传感器网络、信息资源管理等。

唐九阳（1978-），男，湖南邵东人，博士，国防科学技术大学副教授、硕士生导师，主要研究方向为无线传感器网络、P2P 网络等。