◎网络、通信与安全◎

# 一种动态能量感知的节点休眠调度算法

党小超[1,2],李 琦[1],郝占军[1,2]
DANG Xiaochao[1,2], LI Qi[1], HAO Zhanjun[1,2]

1.西北师范大学 计算机科学与工程学院,兰州 730070
2.甘肃省物联网工程研究中心,兰州 730070
1.College of Computer Science and Engineering, Northwest Normal University, Lanzhou 730070, China
2.Gansu Province Internet of Things Engineering Research Center, Lanzhou 730070, China

**DANG Xiaochao, LI Qi, HAO Zhanjun. Approach of sleep scheduling algorithm based on dynamic energy-aware. Computer Engineering and Applications, 2017, 53(20):61-67.**

**Abstract:** In order to reduce the end-to-end delay in a Low-Duty-Cycle Wireless Sensor Network (LDC-WSN) and balance the energy load of the network, a Dynamic Energy-aware Sleep Scheduling algorithm (DESS) is proposed. This algorithm adaptively increases the number of times of active time-slot to balance energy consumption of the nodes by perceiving dynamic residual energy of them. The simulation shows that, compared with the LES and TOSS, DESS can achieve significant performance improvement in terms of energy-saving and end-to-end delay in the situation of the same requirement delay and therefore prolong the network lifetime.
**Key words:** wireless sensor network; low-duty-cycle; node sleep scheduling; energy-aware; sleep delay

**摘 要:** 为减小低占空比无线传感器网络(LDC-WSN)中端到端的休眠延迟和均衡能量负载,提出了一种动态能量感知的节点休眠调度算法(DESS),该算法通过感知节点剩余能量的动态变化,自适应地增加苏醒时隙的次数,用以平衡网络中节点的能量消耗。仿真结果表明,与同类算法的LES和TOSS相比,DESS在休眠延迟以及能源消耗等方面带来明显的性能提升,有效地延长网络的生命周期。
**关键词:** 无线传感器网络;低占空比;休眠调度;能量感知;休眠延迟
**文献标志码:** A **中图分类号:** TP391 doi:10.3778/j.issn.1002-8331.1604-0390



无线传感器网络(Wireless Sensor Network,WSN)是近年来人工或自然环境中的一种新的数据采集技术[1],目前被广泛用于在远距离和危险环境中收集可靠和准确的信息。WSN由多个称为传感器节点的检测站组成,其中每一个传感器节点都是配有传感器、微处理器、收发器和电源的微小而轻巧的便携式设备[2]。而这种微小设备的电源只能提供有限的能量,如何最大化整个网络的工作时间也是目前的一个研究热点。

低占空比无线传感器网络(Low-Duty-Cycle Wireless Sensor Network,LDC-WSN)适应网络发展的恰当时机而产生[3-4]。低占空比模式下,每一个传感器节点都有自己的工作调度表,使节点的工作与睡眠的时间交替进行,极大地延长了整个网络的工作寿命,但在同一时间,相邻节点间的传输延迟会变大,从而影响数据的及时、有效传输。虽然目前针对传输延迟问题已经存在一些方法,但是传统方法在LDC-WSN环境下的节点剩余能量的动态变化方面的研究却有待进一步成熟。

文献[5-6]根据每条链路成功转发一个数据分组的时间是动态的、不确定的,对路径延迟的概率分布进行了分析。文献[7]提出了一种由发送节点发起建立连接

**基金项目:** 国家自然科学基金(No.61363059);西北师范大学青年教师科研能力提升计划项目(No.NWNU-LKQN-13-24)。
**作者简介:** 党小超(1963—),男,教授,硕士生导师,研究方向:计算机网络;李琦(1993—),女,硕士研究生,研究方向:无线传感器网络;郝占军(1979—),通讯作者,男,副教授,研究方向:计算机网络、无线传感器网络,E-mail:zhanjunhao@126.com。
**收稿日期:** 2016-04-28 **修回日期:** 2016-06-14 **文章编号:** 1002-8331(2017)20-0061-07
CNKI **网络优先出版:** 2016-08-22, http://www.cnki.net/kcms/detail/11.2127.TP.20160822.1003.018.html

的异步低占空比、低碰撞的PB-MAC算法，该算法在保持低延迟以及高传递率的情况下降低能耗。文献[8-9]为了降低端到端的延迟，分别提出了SDC算法和单一汇聚节点（Sink node）网络环境中的TOSS算法，其中SDC算法是使用通用发现的方法来降低邻居节点之间的传输时延。文献[10]为了均衡能量消耗，解决节点之间时延的问题，提出了一种PTW（Pipelined Tone Wake up）算法。但这些算法只能在特定条件下控制延迟，而且没有考虑链路质量引起的数据重传和节点能量的动态变化问题，不能有效地应用到实际生活当中。文献[11]则考虑了当链路不可靠时实现低占空比网络中节约能量和减少传输延迟的方法。文献[12]通过选择转发节点的序列，提出了一种优化端到端的期望延迟、能量消耗和传输成功率的方法。而文献[13]提出的一种被称为LES的休眠调度机制，考虑到链路质量问题，加入了能量感知技术，但是其采用的是使传感器节点最多只能增加一次苏醒时隙（active time slot）的方法，不能很好地使LDC-WSN中节点的能量均衡负载。

本文首先研究了LDC-WSN下的节点休眠调度和能量感知技术，其次提出了一种节点休眠调度算法（a Dynamic Energy-aware Scheduling Scheme，DESS）。该算法主要包括两部分：通过引入感知技术，对节点的剩余能量进行动态感知；根据感知到的结果对链路上节点的工作时间和休眠时间进行调度。该算法使得链路上的节点不局限于增加苏醒时隙的次数，每个节点可以根据感知到的剩余能量自适应地增加一次或多次苏醒时隙，不仅使得节点的能量尽其所用，而且均衡了整个网络中的能量负载，从而延长了整个网络的工作时间。最后，针对不同的参数，对DESS算法和其他经典算法通过仿真实验进行了对比和分析。

## 1　网络模型与相关定义

### 1.1　网络模型

在低占空比WSN中，把每个节点的寿命时间分为等长的周期，每个周期包括 $T$ 个相同的时隙（time slot）。如图1所示，每一个节点的工作调度表的一个周期都是由等长的时隙组成（在图中表示为固定长度的方框），其中阴影和空白的方框分别表示节点处于工作和休眠状态。

本文研究的网络是低占空比无线传感器网络，该网络需满足以下的一些要求：

（1）网络中的节点一旦随机部署完毕，其位置就不变，且本文中每个传感器节点的工作调度表都具有周期性。

（2）网络中选用FTSP协议来保证所有的节点在时间上保持同步。

（3）传感器节点之间在数据传输的过程中不考虑冲突问题[12]。

（4）本文采用和文献[13]相同的链路质量方法。

（5）在被部署前，传感器节点的工作调度表是随机选择的，之后相邻节点间会共享它们的工作调度表。网络中每个传感器节点在更新其工作调度表之前都会先通知它的邻居节点，如果确定它的邻居节点都知道这个即将更新的调度表，该节点才会在下一次苏醒的时刻更新其工作调度表。

### 1.2　相关定义

**定义1**（邻居节点）在一个节点所能感知的半径范围内的节点，也就是经一跳传输所能到达的节点，定义为该节点的邻居节点。节点 $i$ 的邻居集定义为：

$$N(i)=\{j|d(i,j)\leqslant R,j\neq i\} \tag{1}$$

其中，$i$ 和 $j$ 都表示传感器节点，$R$ 是指 $i$ 的感知半径，$d(i,j)$ 表示为 $i$ 和 $j$ 之间的直线距离。

**定义2**（工作节点和休眠节点[14]）在LDC-WSN中，

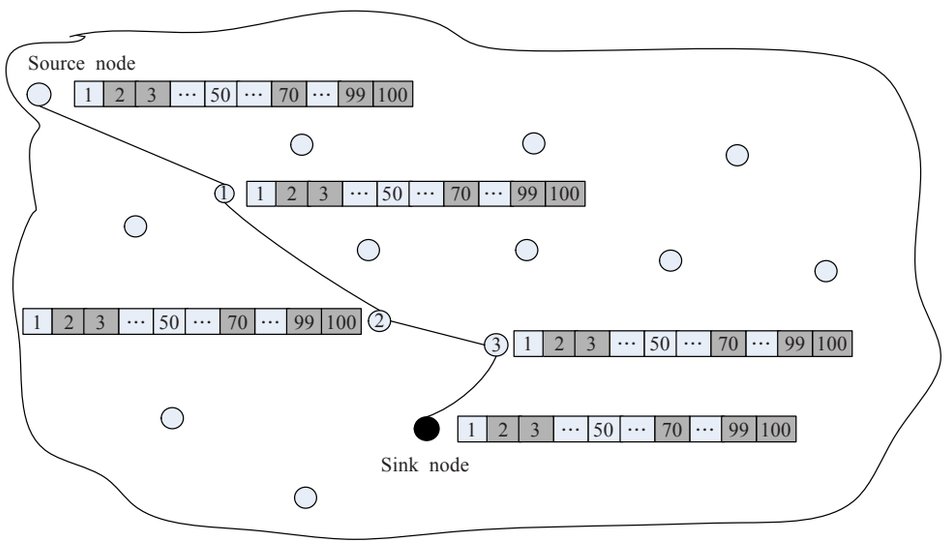



图1　低占空比无线传感器网络

传感器节点一般存在两种状态:工作状态或休眠状态。当节点处于前一种状态时,其不仅可以提供感知、检测、收发数据等各种功能,而且还可以提供空闲状态下的侦听功能;当节点处于后一种状态时,它只保留定时的功能。节点的这两种状态以轮替的方式交互进行,在一定的条件下相互转换。

网络中节点的工作时间是按周期 $T$ 进行划分。用数组 $(t_i^j,\tau)$ 表示传感器节点 $i$ 第 $j$ 次处于工作状态,且从第 $t_i^j$ 时刻开始处于工作状态的持续时间为 $\tau$,其中 $\tau$ 表示时隙的个数。不管节点是否处于工作状态,或处于休眠状态,时隙大小都统一不变。

**定义**3(节点工作调度表) 传感器节点 $i$ 的工作调度表 $\Gamma_i$ 的一个周期,是由一系列有限的工作和休眠两种状态组成的集合。结合上述的定义2,如果节点 $i$ 在一个周期中被唤醒了 $n$ 次,则它的一个周期的工作调度表就用公式(2)的形式表示。

$$\Gamma_i=\{(t_i^1,\tau),(t_i^2,\tau),\cdots,(t_i^n,\tau)\} \tag{2}$$

**定义**4(节点占空比(Duty Cycle,DC)) 在LDC-WSN中,一个周期内传感器节点处于工作状态的时间与周期总时间长度的比值。结合公式(2),可以得出节点 $i$ 的占空比:

$$DC_i=\frac{\sum_{j=1}^{n}\tau}{T} \tag{3}$$

**定义**5(休眠延迟) 在LDC-WSN中,发送节点 $i$ 收到数据信息到其将要发送到的接收节点 $j$ 处于工作状态所等待的时间,定义为休眠延迟。本文中,用 $d_{ij}(t)$ 表示节点 $i$ 在 $t$ 时刻收到消息后,即将发送给邻居节点 $j$ 的休眠延迟;$D_{ij}(t)$ 表示非相邻节点间的休眠延迟;用 $E[D_{ij}^{m,h}(t)]$ 表示从一个发送节点到最终接收节点的路径上,节点增加的苏醒时隙次数为 $h$ 时的最小延迟期望值。$m$ 为该条路径上从 $i$ 到 $j$ 所需要的跳数。

## 2　基于动态能量感知的节点休眠调度算法(DESS)

本文提出的DESS算法主要解决的问题是:在LDC-WSN中,如何在达到一定延迟要求的前提下,均衡节点能量负载,从而使网络的工作寿命最大化。DESS算法主要包括两部分:节点剩余能量的动态感知和休眠调度机制。这两部分是一起进行的,针对一条链路进行节点休眠调度,通过对节点的剩余能量进行感知来自适应地增加该条路径上节点苏醒时隙的次数。

### 2.1　节点剩余能量感知

在现实的网络应用中,传感器节点的剩余能量是有很大不同的,如果过多地使用剩余能量值较低的节点,必然会导致整个网络能量负载失衡,减短网络的生存周期[15]。

本文提出的DESS算法对节点的剩余能量进行动态感知,根据感知到的结果使得剩余能量多的节点可以多增加几次苏醒时隙,而剩余能量少的节点少增加或者不增加苏醒时隙次数来适应网络的实际需要。如果用 $E_{\text{avg}}$ 表示整个传感器网络的平均能量,用 $E_{\text{res}}^j$ 表示传感器节点 $j$ 本身的剩余能量,那么可以根据以上所述得出以下公式。

$$E[D_{ij}^{m,h}(t)]=\min\begin{cases}E[D_{ij}^{m,h}(t)]\\ E[d_{i1}(t)]\oplus E[D_{1j}^{m-1,h}(t')],h\geqslant 1,m>0\\ E[d_{i1}^{h_1}(t)]\oplus E[D_{1j}^{m-1,h-h_1}(t')],E_{\text{res}}^1>\alpha E_{\text{avg}}\end{cases} \tag{4}$$

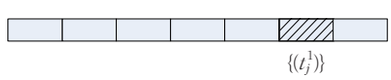
在公式(4)中,$\alpha$ 为动态权重值。从公式(4)可以看出,在源节点之后的第1个节点(简称节点1)尝试增加苏醒时隙之前,首先将比较剩余的能量和网络本身的平均能量。节点1为了增加苏醒状态的时隙,必须满足 $E_{\text{res}}^1>\alpha E_{\text{avg}}$(其他节点依次类推需满足 $E_{\text{res}}^i>\alpha E_{\text{avg}}$),否则还是按照原来的工作调度表工作。这里取 $\alpha=1.2$,因为经查阅许多文献和大量的实验发现,当 $\alpha=1.2$ 时,能够保证在对原本的节点休眠调度算法影响较小的情况下延长网络的工作寿命。

其中,$h_j$ 表示接收节点 $j$ 所增加的苏醒时隙的次数,$h$ 表示整条链路上所有节点增加的苏醒时隙次数的总和,且 $h\geqslant h_j\geqslant 0$。$h$ 的计算方式如公式(5)所示。

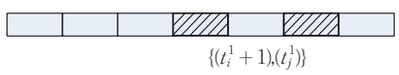
$$h=\sum_{i=1}^{m}h_i \tag{5}$$

### 2.2　节点休眠调度算法

对于邻居节点 $i$ 和 $j$ 来说,当 $j$ 需增加一次苏醒时隙来减少端到端的延迟时,则 $j$ 的工作周期就在原来的工作周期基础上再增加一个苏醒时隙,即 $\Gamma_j=\{(t_j^1)\}$ 变为 $\Gamma_j=\{(t_i^1+1),(t_j^1)\}$,如图2所示。




(a)接收点 $j$ 原来的工作调度




(b)接收点 $j$ 增加一次苏醒时隙后的工作调度表

**图2　接收节点 $j$ 增加一次苏醒时隙前后的工作调度表变化**

如果 $t_j^1>t_i^1+1$,那么可得到新增一次苏醒时隙后,节点间端到端的休眠延迟期望值,如公式(6)所示。

$$E[d_{ij}^1(t)]=\sum_{n=0}^{N_{\max}-1}p_{ij}(1-p_{ij})^n\{1+\left\lfloor n/2+\frac{1}{2}\right\rfloor\times [t_j^1-(t_i^1+1)]+\lfloor n/2\rfloor\times[T-(t_j^1-(t_i^1+1))]\} \tag{6}$$

这里考虑到链路质量问题,用 $p_{ij}$ 表示邻居节点 $i$ 和 $j$ 之间的链路质量。如果节点 $i$ 向节点 $j$ 发送数据的重

传次数 $n_{ij}$ 与算法所允许的最大重传次数 $N_{\max}$ 之间存在 $n_{ij}>N_{\max}$ 的关系，则丢弃该数据包。

对于网络中的节点，如果不增加苏醒时隙的数目，则节点之间的端到端的延迟仍然是原来的值。如果在一个工作周期中，增加接收节点的苏醒时隙次数，那么将会减少发送点等待接收点处于工作状态的时间，进而会减少节点之间端到端的延迟。由此可得到两个相邻节点之间的休眠延迟期望值公式：

$$E[D_{ij}^{m,h}(t)]=\begin{cases}E[d_{ij}(t)], m=1,h=0\\ E[d_{ij}^{h_j}(t)], m=1,h=h_j\end{cases}, h\geqslant h_i\geqslant 0, m>0 \quad (7)$$

对源节点 $i$ 和目标节点 $j$ 来说，$i$ 必须经过 $(m-1)$ 跳节点才能到达 $j$。那么对于目标节点 $j$ 来说，可分为以下两种情况，如公式(8)所示。

$$E[D_{ij}^{m,h}(t)]=\min\begin{cases}E[d_{i1}(t)]\oplus E[D_{1j}^{m-1,h}(t')]\\ E[d_{i1}^{h_j}(t)]\oplus E[D_{1j}^{m-1,h-h_1}(t')]\end{cases}, h\geqslant h_i\geqslant 0, m>0 \quad (8)$$

通过迭代的方法可得出一般节点之间增加 $h$ 次苏醒时隙后的延迟期望值，结合原来不增加任何苏醒时隙的初始延迟计算公式和 $h$ 值，可以得出一般情况下端到端休眠延迟的期望值，如公式(9)所示。

$$E[D_{ij}^{m,h}(t)]=\min\begin{cases}E[D_{ij}^{m,h}(t)]\\ E[d_{i1}(t)]\oplus E[D_{1j}^{m-1,h}(t')]\\ E[d_{i1}^{h_1}(t)]\oplus E[D_{1j}^{m-1,h-h_1}(t')]\end{cases}, h\geqslant 1, m>0 \quad (9)$$

其中，$t'$ 表示路径上源节点后面的第1个节点收到数据包的时刻。$E[d_{i1}(t)]\oplus E[D_{1j}^{m-1,h}(t')]$ 表示在该路径上的第1个节点处不增加任何苏醒时隙，后面路径上的节点到目标节点 $j$ 之间总共增加了 $h$ 次时隙的休眠延迟期望值；$E[d_{i1}^{h_1}(t)]\oplus E[D_{1j}^{m-1,h-h_1}(t')]$ 表示节点1处增加了 $h_1$ 次苏醒时隙，后面路径上的节点增加 $(h-h_1)$ 次的延迟期望值。

### 2.3　算法具体步骤

算法步骤如下：

**步骤**1　初始化低占空比无线传感器网络，节点随机分布。

**步骤**2　根据初始的工作调度表，在不增加任何苏醒时隙次数的情况下，计算出从源节点到目标节点这条路径上的休眠延迟期望值 $E[D_{ij}^{m,0}(t)]$。

**步骤**3　将步骤2得到的结果 $E[D_{ij}^{m,0}(t)]$ 和实际延迟要求值 $B$ 做比较。若是 $E[D_{ij}^{m,0}(t)]\leqslant B$，则满足实际延迟要求，没必要增加从源节点到目标节点这条路径上任何节点的苏醒时隙次数，算法退出；否则，进入步骤4。

**步骤**4　对该链路上节点的剩余能量进行感知。对单个节点 $i$ 来说，若 $E_{\text{res}}^{i}>\alpha E_{\text{avg}}$，则增加节点 $i$ 的苏醒时隙，遍历路径上的节点，每个节点从增加0次开始，依次计算出增加 $h_i(i=0,1,\cdots,m)$ 次所能达到的最小延迟期望值 $E[D_{ij}^{m,h_i}(t)]$，直到 $E[D_{ij}^{m,h_i}(t)]\leqslant B$。

**步骤**5　计算 $h=\sum_{i=1}^{m}h_i$。这时，增加的苏醒时隙总次数 $h$ 是在满足实际延迟要求下的最小值，增加时隙所消耗的能量也是在满足时延要求下的最小值。

## 3　性能分析

### 3.1　能耗分析

网络节点的能量消耗主要有两个方面：处于工作状态的能量消耗 $E_{\text{W}}$ 和处于休眠状态的能量消耗 $E_{\text{S}}$。那么节点的总能耗 $E_{\text{Total}}$ 可用公式(10)表示。

$$E_{\text{Total}}=E_{\text{W}}+E_{\text{S}} \quad (10)$$

节点处于工作状态的时候执行感知，数据的采集、处理和通信等功能，执行这些功能的时候都需要消耗能量，且由节点的传感单元、处理单元和通信单元分别负责[15]。设单位时间内节点发送和接收数据的能耗分别为 $U_{\text{MS}}$ 和 $U_{\text{MR}}$，消息产生的平均时间间隔为 $TI$(time interval)，单位时间内数据采集和处理的能耗分别用 $U_{\text{WG}}$ 和 $U_{\text{WP}}$ 表示；用于采集和处理的时长分别用 $T_{\text{WG}}$ 和 $T_{\text{WP}}$ 表示，节点的总生命周期为 $T$，则工作状态的总能耗为采集、处理和通信的能耗之和，可用公式(11)表示。

$$E_{\text{W}}=U_{\text{WG}}T_{\text{WG}}+U_{\text{WP}}T_{\text{WP}}+\frac{U_{\text{MS}}T}{TI}+\frac{U_{\text{MR}}T}{TI} \quad (11)$$

节点处于休眠状态时只保留了定时功能，设休眠时单位时间内的能耗为 $U_{\text{S}}$，休眠时长为 $T_{\text{S}}$，则休眠状态消耗的能量可用公式(12)表示。

$$E_{\text{S}}=U_{\text{S}}T_{\text{S}} \quad (12)$$

将公式(11)、(12)代入公式(10)，可得：

$$E_{\text{Total}}=2U_{\text{WG}}T_{\text{WG}}+2U_{\text{WP}}T_{\text{WP}}+2\frac{U_{\text{MS}}T}{BI}+2\frac{U_{\text{MR}}T}{BI}+U_{\text{S}}T_{\text{S}} \quad (13)$$

从公式(13)可以看出，节点处于工作和休眠这两种状态的时间彼此不存在交集，也就是说一个节点在同一时刻能且只能处于一种状态，且根据节点的工作调度表分别持续一定的时间。本文提出的调度算法实质上就是调度节点处于各个状态的时间。整体来看，无线传感器网络中的节点一旦部署完毕，网络中节点的总体能耗也就随之确定，不会改变。在这样的一个大前提下，尽量使网络的生命周期延长[16]。

而LES算法中的能量感知和休眠延迟期望值的公式如下所示。

$$E[D_{ij}^{m,h}(t)]=\min\begin{cases}E[D_{ij}^{m,h}(t)]\\ E[d_{i1}(t)]\oplus E[D_{1j}^{m-1,h}(t')], h\geqslant 1, m>0\\ E[d_{i1}^{1}(t)]\oplus E[D_{1j}^{m-1,h-1}(t')], E_{\text{res}}^{1}>\alpha E_{\text{avg}}\end{cases} \quad (14)$$

从公式(7)、(9)和公式(14)能够看出,考虑到节点的剩余能量,增加的苏醒时隙次数中,对于每个节点$i$,DESS增加的次数不局限于其他算法中的0和1。本文提出算法当中的$h=\sum_{i=1}^{m}h_i$与TOSS和LES算法中的$h$相比,能够动态自适应地增加传输路径上苏醒时隙的次数,从而均衡网络中的能量负载,延长网络的工作寿命。

### 3.2　延迟分析

在LDC-WSN中,节点之间的延迟情况除了端到端的休眠延迟,还包括通信延迟[14]。本文中假设$n$为从源节点到目标节点的一条链路上所有节点的数目。设节点之间端到端的休眠延迟为$T_{SD}$,通信延迟为$T_{CD}$,那么总时延可用公式(15)表示。

$$T_{Total}=T_{SD}+T_{CD} \tag{15}$$

根据公式(15),DESS算法,TOSS算法和LES算法的总时延$T_{DESS}$,$T_{TOSS}$,$T_{LES}$分别用式(16)~(18)表示:

$$T_{DESS}=mT_{CD}+\sum_{i=1}^{h}(i\times T_{SD}) \tag{16}$$

$$T_{TOSS}=mT_{CD}+\sum_{i=1}^{n}(i\times T_{SD}) \tag{17}$$

$$T_{LES}=mT_{CD}+\sum_{i=1}^{m}(i\times T_{SD}) \tag{18}$$

用式(17)减去式(16),得到:

$$T_{TOSS}-T_{DESS}=\sum_{i=h}^{n}(i\times T_{SD})>0 \tag{19}$$

用式(18)减去式(16),得到:

$$T_{LES}-T_{DESS}=\sum_{i=h}^{m}(i\times T_{SD})>0 \tag{20}$$

因为$n,m>h$,所以公式(19)、(20)大于0。即无论是TOSS算法还是LES算法,节点之间端到端的延迟都大于本文提出的DESS算法,因此DESS算法的时间延迟较低。另外,该算法时间复杂度与从源节点到目标节点路径上的总节点个数$n$有关,还与路径上增加苏醒时隙的节点之间的跳数$m$有关,显然$m\leqslant n$。由算法的具体步骤可知,步骤1~步骤2的时间复杂度为$T_1(n)$;步骤3~步骤5的时间复杂度为$T_2(m)$;根据求和法则,算法总的时间复杂度为$T_1(n)+T_2(m)=O(\max\{n,m\})=O(n)$。可见,该算法在满足同等延迟要求的情况下,计算复杂度较低。

## 4　仿真实验

为了验证本文所述算法的性能上优于同类算法TOSS和LES,本章设计了仿真实验。并且基于系统性能参数统计和占空比设置的灵活性等多方面的考虑,经过反复仿真实验,确定本文算法的参数默认值按表1设置更有利于仿真实验的进行。

**表1　仿真平台默认参数信息**

| 平台参数 | 取值 |
|---|---|
| 节点个数$m$ | 100 |
| 节点占空比$dc$/% | 1 |
| 节点通信、侦听半径$r_c$/m | 10 |
| 每个时隙持续时间$\tau$/s | 1 |
| 节点工作周期$T$/s | 100 |
| 链路质量$q$/% | 60~100 |
| 最大重传次数$N_{max}$ | 10 |
| 节点初始能量$E_{Init}$/J | 20 000~50 000 |
| 节点发送数据消耗能量$E_S$/J | 1 |
| 节点接收数据消耗能量$E_R$/J | 0.8 |
| 节点空闲状态消耗能量$E_L$/J | 0.75 |
| 平均消息产生时间间隔$BI$ | 1 200 |

仿真过程中,为了对比DESS算法和其他两种算法性能之间的区别,分别设置在不用节点占空比和不用休眠延迟要求的情境下做实验,每次只改变其中的一个参数,其他的参数则保持默认值;然后针对本文提出的算法单独做仿真,观察链路质量变化和节点数量变化分别对该算法的性能产生怎样的影响,说明DESS算法最适应的参数值范围。

### 4.1　算法的节点占空比仿真与分析

本次仿真实验是在保持休眠延迟要求固定不变的条件下,考察占空比对DESS算法中所增加的时隙次数的最小值和网络生命周期长度的影响,并与TOSS和LES这两种传统算法做比较,如图3所示。

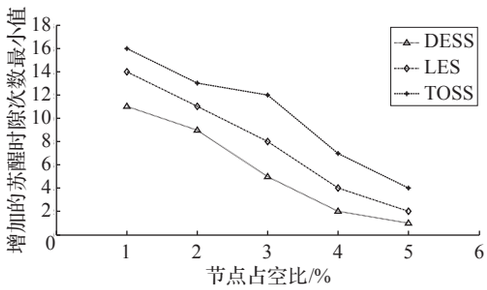



**图3(a)　不同占空比下的增加苏醒时隙次数的最小值**

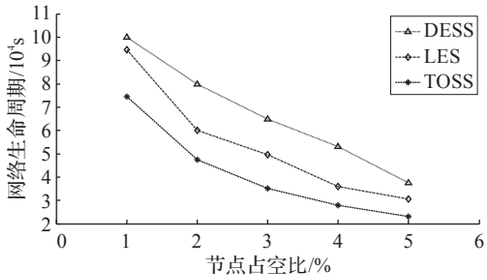



**图3(b)　不同占空比下的网络生命周期长度**

从图3(a)的仿真结果可以看到,在节点占空比越来越高(从1%~5%)的情况下,网络要满足延迟要求需要增加的苏醒时隙次数最小值也会越来越小。这是因为占空比越高,节点周期中的苏醒时隙个数就越多,处于工作状态的持续时间就越长。在相同节点占空比下,本

文提出的算法与另外两种算法相比,增加的苏醒时隙次数的最小值比较小。这是因为虽然在源节点后第一个节点处增加的苏醒时隙次数 $h_1$ 可能会多,但是就整条传输路径而言,路径上所有节点增加的苏醒时隙次数 $h_i$ 都是自适应确定的,每一次增加的苏醒时隙都是有效的,所以增加的总苏醒时隙次数 $h$ 相应变少。在节点占空比为3%时,DESS算法最少需要增加5次苏醒时隙就可以达到给定延迟要求,而LES和TOSS算法分别得增加8次和12次才能达到延迟要求。

从图3(b)可以看到,在节点占空比越来越高的情况下,网络的生命周期将会大大降低,这也是为什么目前的低占空比无线传感器网络(LDC-WSN)正变得比以往的一直处于工作状态的网络(always active network)越来越受青睐的原因。在同一占空比下,本文提出的算法更能延长网络的生命周期。在节点占空比由低到高(1%~5%)变化的过程中,本文提出的DESS算法都比其他两种算法更能有效地延长网络的工作时间,原因是路径上节点根据自身剩余能量自适应增加苏醒时隙,均衡了能量负载,使网络不会陷入局部负载失衡而减短生命周期,同时这也符合实验前的分析。

### 4.2　算法的休眠延迟仿真与分析

休眠延迟要求是指算法需要满足的实际给定延迟值,DESS算法需要根据不同的延迟要求动态自适应地增加不同的苏醒时隙次数,而增加时隙次数的多少,也会影响到网络生命周期的时间长短。仿真结果如图4所示。

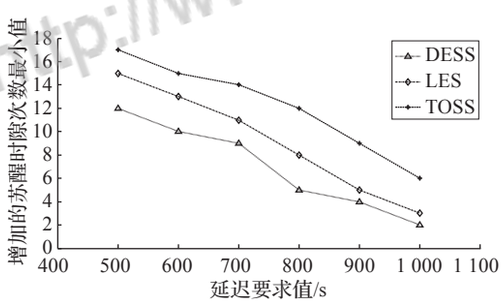



**图4(a)　不同延迟要求下的增加苏醒时隙次数的最小值**

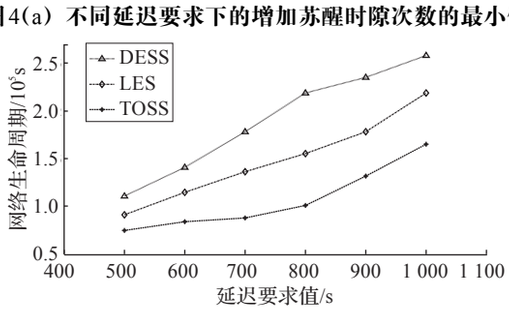



**图4(b)　不同延迟要求下的网络生命周期长度**

从图4(a)的仿真结果可以看到,给定的实际延迟要求值越大,满足此要求需要增加的时隙次数最小值就越小,反之亦然。这是因为网络对延迟的要求严格程度越高,需要增加更多次的苏醒时隙,才可以减少发送节点等待其下一个接收节点处于苏醒时隙的时间,进而才可以满足时延要求。

在同一休眠延迟要求下,DESS增加的时隙数的最小值比TOSS和LES增加的次数最小值都小,这与实验前的分析相吻合。在延迟要求值为500 s时,也就是说为了保证数据的时效性,在实际应用中要求网络节点间传输数据的时间延迟不得超过500 s时,该算法在链路上最少需要增加12次苏醒时隙,而LES算法和TOSS算法分别在该路径上最少增加15次和17次才能满足时延要求。当路径上的每个节点都增加且只增加一次的苏醒时隙时,三种算法的性能是等同的,但是本文所提出的DESS算法能够根据每个节点的剩余能量动态自适应地增加不同的时隙次数,这既是DESS算法与其他同类算法最不一样的地方,也是本文最重要的一个方法。从图4(b)可以看到,网络延迟要求越宽松,即延迟要求值越大,三种算法下的网络生命周期都会对应延长。当休眠延迟要求为800 s时,DESS算法比LES算法延长了29%的生命周期,比TOSS算法延长了54%的生命周期,说明在同一休眠延迟要求下,本文提出的DESS算法在延长生命周期方面优于LES和TOSS算法,主要原因是本文方法是动态增加苏醒时隙,很大程度上均衡了节点能量负载。

### 4.3　链路质量和节点数量的分析

除了节点占空比和延迟要求对网络性能有一定的影响,还有链路质量、节点数量以及许多其他因素也会对网络的性能造成不同程度的影响。

为了体现链路质量对网络性能的影响,保持路径上的节点数量不变,对休眠延迟期望值做仿真实验,结果如图5所示。

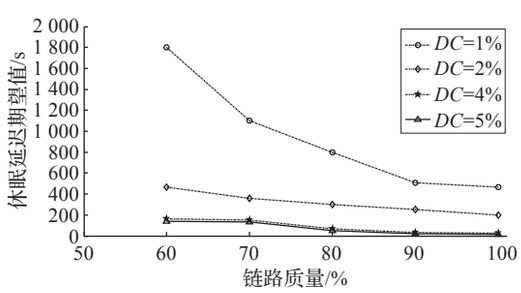



**图5　不同链路质量下的休眠延迟期望值**

从图5可以看出,随着链路质量的提高,即节点间传输成功率的提高,网络的休眠延迟期望值会减小,这是因为链路质量越可靠,数据传输成功率就越高,节点间由于传输失败而进行重传的次数就越少。另一方面可以看到,在同一链路质量下,占空比越大,延迟期望值会明显减小,其变化曲线也渐趋平缓。其中主要原因是节点占空比越大,节点之间传送数据需要等待的时间就会变短。

为了体现传感器节点的数量对网络性能的影响,做如下的仿真实验,结果如图6所示。

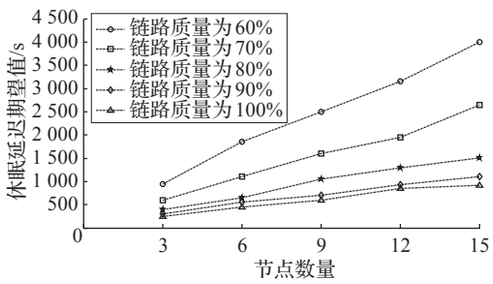



**图6　不同节点数量下的休眠延迟期望值**

从图6可以看出,链路上节点数目越多,休眠延迟期望值会越大。例如,在链路质量为90%时,路径上有6个节点和9个节点分别对应的休眠延迟期望值为550 s和1 100 s。其中主要原因是从源节点经过多跳路由才到达目标节点,跳数越多,节点之间的延迟时间就累加得越长,导致节点间端到端的休眠延迟期望值增大。

## 5　结束语

针对LDC-WSN中能量负载不均衡、端到端延迟大等问题,本文提出了DESS算法。该算法在使用LDC和动态能量感知技术的基础上,使节点动态自适应地增加苏醒时隙的个数,均匀地消耗能量,但是该算法限定了最大数据重传次数,以此来降低计算复杂度,并用牺牲部分精确度作为代价来满足休眠延迟要求。通过仿真和实验结果验证,与其他休眠调度算法相比,DESS算法在平均能耗、时延等方面性能更优,能够大幅延长网络的生命周期。但本文仍有一定的局限性,DESS算法仅考虑了从源节点到目标节点的单向延迟,没有考虑双向延迟问题,这项内容有待进一步研究。

## 参考文献:

[1] Bhosle M V,Nighot M K,Das S S.An energy efficient and reliable location wise data aggregation in WSN[C]// International Conference on Computing Communication Control & Automation (ICCUBEA),2015.

[2] Hady A A,El-Kader S M A,Eissa H S.Intelligent sleeping mechanism for wireless sensor networks[J].Egyptian Informatics Journal,2013,14(2):109-115.

[3] Xu L,Chen G.LBcast:load-balanced broadcast scheduling for low-duty-cycle wireless sensor networks[C]//Global Communications Conference (GLOBECOM),2013:7-12.

[4] Wang F,Liu J.On reliable broadcast in low duty-cycle wireless sensor networks[J].IEEE Transactions on Mobile Computing,2012,11(5).

[5] Liu X,Zhang H,Xiang Q,et al.Taming uncertainties in real-time routing for wireless networked sensing and control[J].IEEE Transactions on Smart Grid,2013,4(1):288-301.

[6] Chen Q,Gao H.Link quality based path delay analysis in wireless sensor networks[J].Journal on Communication,2014,35(6):100-109.

[7] 李哲涛,朱更明,王志强,等.低占空比、低碰撞的异步无线传感器网络MAC协议[J].通信学报,2013(10):9-16.

[8] Gu Y,He T,Lin M,et al.Spatiotemporal delay control for low-duty-cycle sensor networks[C]//IEEE Real-time Systems Symposium,2009:127-137.

[9] Cao Q,Abdelzaher T,He T,et al.Towards optimal sleep scheduling in sensor networks for rare-event detection[C]// Proceedings of the 4th International Symposium on Information Processing in Sensor Networks,2010:20-27.

[10] Yang X,Vaidya N F.A wakeup scheme for sensor networks:achieving balance between energy saving and end-to-end delay[C]//Real-time & Embedded Technology & Applications Symposium,2004:19-26.

[11] Cheng L,Gu Y,He T,et al.Dynamic switching-based reliable flooding in low-duty-cycle wireless sensor networks[C]//Proceedings of INFOCOM,Turin,Italy,2013:1393-1401.

[12] Gu Y,He T.Dynamic switching-based data forwarding for low-duty-cycle wireless sensor networks[J].IEEE Transactions on Mobile Computing,2011,10(12):1741-1754.

[13] Chen L Y,Wang J L,Zhang J Y,et al.Scheduling scheme algorithm in low-duty-cycle WSN[J].Journal of Software,2014,25(3):631-641.

[14] Mao J,Tian Y,Jiang Y.Sleep scheduling algorithm for monitoring wireless sensor networks[J].Communications Technology,2012.

[15] Yu G,Tian H.Bounding communication delay in energy harvesting sensor networks[C]//IEEE International Conference on Distributed Computing Systems,2010:837-847.

[16] 温涛,张冬青,郭权.无线传感器网络冗余节点休眠调度算法[J].通信学报,2014(10).

[17] Boano C A,ZúÑiga M A,Voigt T,et al.The triangle metric:fast link quality estimation for mobile wireless sensor networks[C]//2010 Proceedings of 19th International Conference on Computer Communications and Networks (ICCCN),2010:1-7.

---

26. 一种光交换核心节点调度算法研究

27. 基于干扰感知的上行调度算法研究

28. 高三了,你还在能量休眠吗?

29. 一种动态频谱分配算法

30. 一种对等网络中动态随机中继节点路由算法

31. 一种基于决策树分类算法的家庭能量动态调度系统

32. 认知网络节点自适应休眠调度算法

33. 基于节点调度策略的能量有效覆盖算法

34. 一种基于能量感知的弹性光网络 RSA 算法

35. 一种费用敏感型家电能量优化动态调度算法

36. 一种基于网络感知的虚拟机再调度算法

37. 一种动态调度的延迟敏感流网络挖掘算法

38. 动态自由节点滞后的任务调度算法

39. 一种分布式动态负载均衡调度算法

40. 一种感知节点智能寻优与动态演进部署算法

41. 一种基于传感节点密度和能量的动态分组策略

42. 小规模WSN的休眠调度算法研究

43. 一种基于节点性能的Hadoop动态调度策略

44. 一种新的集卡动态调度模型及算法

45. 一种网格调度算法的研究

46. Ad-hoc网络节点休眠策略的设计

47. CDN带宽动态调整与跨节点调度

48. 基于队列感知和信道感知的上行调度算法研究

49. 一种基于节点性能的Hadoop动态调度策略

50. 一种基于Vxworks的多任务调度算法