An Autonomous Energy-Aware Routing Scheme: a Supplementary Routing Approach for Path-Preserving Wireless Sensor Networks

Fang-Yie Leu, Guo-Cai Li and Wen-Chin Wu

Department of Computer Science and Information Engineering,
Tunghai University, Taiwan

{leufy, g942815}@thu.edu.tw, altosax.tw@gmail.com

Abstract One of the hottest research topics is how to reduce energy consumption to prolong the lifetime of a WSN. Some nodes of a WSN may be isolated from others, particularly when all their neighbors have crashed or run out of energy, and are thus no longer helpful in collecting and relaying messages, even though their batteries still retain some level of energy. To avoid this, in this paper, we propose a supplementary routing approach, named Energy-aware routing (EnAR in short), which can help to extend the lifetime of a WSN by integrating itself with the routing scheme of the host WSN. With the EnAR, a node in a given WSN (denoted by host WSN) checks the amount of its remaining energy, each time the node receives a request, to see whether it should reject or accept the request. The WSN routes messages through other communication paths when nodes on the original path have consumed too much energy under the condition that alternate paths truly exist. One of the alternate paths will be chosen to relay messages to prevent nodes along the original path from exhausting all their energy so that the WSN can last longer. Our experimental results show that this scheme is applicable to and suitable for an environment with decentralized control, and is able to effectively prolong the lifetime of a WSN.

Keywords: wireless sensor networks; energy-aware routing; supplemental routing scheme; energy consumption; path-preserving routing scheme

1. Introduction

Due to the rapid advancement and development of manufacturing techniques, sensor nodes (sensors in short) have become tinier and cheaper. Those that work autonomously can be used in place of human workers to sense and monitor environmental change. In a wireless sensor network (WSN), sensors are usually put in locations that people can not or seldom reach. Hence, their batteries can not be replaced or can only be replaced infrequently. That is why prolonging the lifetime of batteries has become a hot research topic recently. Researchers have proposed ways to solve this problem, e.g., by decreasing message relaying counts, shortening communication distance, or using nodes with more energy to handle energy consuming tasks. LEACH [1], directed diffusion [2], SPIN [3], and SAR [4] concentrate on routing affection, whereas SMACS/EAR (Self-organizing Medium Access Control for Sensor networks/ Eavesdrop And Register) [4] uses a self-organizing and asynchronous approach to improve MAC protocols. All of these methods

decrease the energy consumption of a node and prolong system life. But let us consider the following situation.

The topology shown in Fig. 1 roughly consists of sections A, B and C. If too many events have occurred or there are many hot nodes in section B, nodes in this section will soon use up their batteries. This will isolate sections A and C. About two-thirds of the nodes still retain some level of energy, but only one-third of them (i.e., section C) are now in service. In other words, checking the total remaining energy in all nodes of a system will not enable us to estimate the remaining life of the system. The focus should be on whether system service is able to proceed or not, which can more accurately reflect the real situation. For example, when using sensors to monitor whether a forest/house is on fire, we do not consider that the system is still working if 30% of nodes have failed. Even if only 1% of nodes are not working, no one knows whether the forest/house is on fire if the fire occurs in the 1% unmonitored area. Hence, prolonging system service is more important than prolonging the life of a node, particularly when an entire monitored area should be completely sensed all the time. In the following, we consider that a system's service become unreliable when at least one node exhausts its energy.

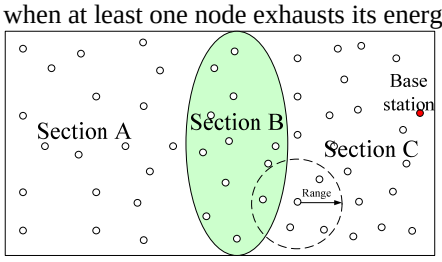

Fig. 1 Topology of a wireless sensor network in which there is an isolation section B

In this research, we propose a routing approach, named energy-aware routing (EnAR), which, as a supplemental mechanism, should be integrated with a path-preserving routing scheme of a WSN to effectively extend the WSN's service time. A node in such a system can reject service requests issued by other nodes, particularly when it has consumed too much energy since system start up. We call the original routing scheme the EnAR integrates with a host routing scheme. In the following, we use the terms routing and relaying messages interchangeably, even though some researchers define them differently. Also, a host routing scheme X, e.g., on-demand, after being integrated with the EnAR, is called X-EnAR, e.g., on-demand-EnAR, and the WSN that deploys an X-EnAR routing scheme is called EnAR-WSN.

2. Related Work

Several types of energy saving methods have been proposed for WSNs. The first type focuses on remaining energy, e.g., E-SPAN (energy-aware spanning tree algorithm) [9] considers energy that each device currently has when building a routing tree as an energy-saving topology. The one with the greatest amount of residual energy will be the root. Others choose one of their neighbors as the parent based on the residual energy the neighbor has and the distance between the underlying node and the root.

The second type emphasizes reducing the number of packet transmissions and receptions. LEACH [1] adopts a cluster-based routing approach in which nodes of a WSN are clustered into groups. Groups are self-organized in each round of data collection. Packets generated by nodes of a cluster (group) are centralized to their cluster head which is responsible for transmitting these packets to a base station. Therefore, how to choose cluster-heads is a critical issue in saving energy. The focus of MECH [5] is on the remaining

energy in that, the more energy a node currently has, the higher probability the node will be chosen as a cluster-head. TEEN (threshold sensitive energy efficient sensor network protocol) [6], LEACH-C (LEACH-centralized) [7] and PEGASIS [8] all refine LEACH. TEEN defines a clustering algorithm to classify WSNs into two types, proactive and reactive which are described above, and for saving energy. LEACH-C determines which nodes will be chosen as cluster-heads based on global information collected by base stations, including geographical topology and remaining energy. In PEGASIS, only one designated node rather than all cluster-heads sends aggregate data to a base station. It uses a greedy algorithm to establish a chain that includes all nodes of a WSN. Nodes transmit messages through a downstream chain for upstream nodes in each round of data collection.

The third type conducts sleeping-mode to nodes. Idle devices enter sleeping-mode in which devices turn off their antennas and some system components to save energy. S-MAC [10] added a sleeping-mode and a random schedule to IEEE802.11 MAC. Each sensor has a probability of entering its sleeping-mode [11] extended system life by preserving coverage and letting some system components go to sleep. LDS (linear distance-based scheduling) [12], BS (balanced-energy scheduling) [17], RS (randomized scheduling) [18], and DS (distance-based scheduling) [13] involve cluster-based routing and focus on a sleeping-mode algorithm. DS and LDS determine whether a sensor should go to sleep or not by calculating the distance between the sensor and its cluster-head. BS prolongs a system's life time by balancing system energy consumption.

The fourth type focuses on energy consumption and its impact on sensing coverage of sensor nodes. The DAPR protocol (distributed activation based on predetermined routes) [10] was proposed based on an application plane that considers the relationship between coverage area and available energy. Reducing power consumption may prolong the lifetime of whole WSNs, but the effective coverage area may not be in proportion to the number of surviving nodes due to the high density of remaining nodes. A CPCHSA algorithm (coverage-preserving cluster-head selection algorithm) [14] was devised to modify the LEACH [1] by deploying a threshold-adjusting formula based on effective coverage area to select a better cluster-head, which is one with a smaller effective coverage area. Those with larger effective coverage areas have lower probability of being a cluster-head. Carefully selected cluster-heads can physically extend the system lifetime and help a WSN maintain a higher effective coverage area.

3. System Model

3.1 Modification of routing

Table-driven and location-aware routing schemes maintain routing tables/paths at all times. Such maintenance consumes much energy and is harmful to system life. During routing, a node only needs to know the type of packets it is requested to relay or send. Routing paths are maintained or established periodically. This type of routing scheme is different from the on-demand approach in that a node with on-demand approach, before sending data, submits a path-discovery packet, that carries source and destination addresses, to find a routing path so that messages generated can reach a base station in its following stage, i.e., data transmission stage. To cope with all probable path-preserving routing schemes that host WSNs may deploy, the EnAR adopts on-demand's routing steps. A node in a EnAR-WSN can autonomously determine whether it should reject relaying messages for other nodes, continuously relay messages for other nodes, or reject a path-discovery request.

3.2 Modes

Fig. 2 shows the layer structure of the EnAR in which a node has two modes, normal and rejection. A node in its normal mode performs the designated tasks normally, including sensing environmental changes and relaying/transmitting message, etc. A node R enters its rejection mode when its remaining energy is less than a threshold, or in some WSNs (e.g., S-MAC, and LDS) it enters sleeping mode. R disappears when it enters rejection mode or leaves the WSN (e.g., failure or exhausting its energy). If a path-preserving WSN can accept that its nodes will disappear during its life time, the EnAR as a supplementary routing mechanism can then be smoothly integrated with the WSN. There are two cases when R will reject the request it receives. The first is if R, after entering its rejection mode, receives a path-discovery request. It rejects the request by replying with no messages, which will force the EnAR-WSN to choose another path that excludes R. The second is if R is now on a routing path. After receiving a packet, it finds that its remaining energy is insufficient to process (e.g., relay) the packet, R then switches itself to rejection mode. Such will cause the host routing scheme to issue a path-discovery request. Whether R should enter its rejection mode and the activities performed before and after entering rejection mode are all autonomously determined and done by R itself, i.e., it is a fully distributive and autonomous way without any central control (e.g., base station's interference and global information collected by base station).

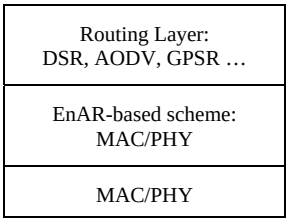

Fig. 2 The layer structure of the EnAR

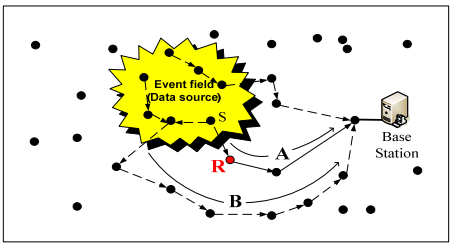

Fig. 3 Data generated in an event field is sent to a base station through different paths when a node R along a routing path refuses to relay packets. Neighbor which will receive a packet sent by node S is autonomously determined by S itself, which is either a source node of or an intermediate node along a routing path.

3.3 Isolation problem

How does an EnAR-WSN work? As shown in Fig.3, if a node R on path A enters its rejection mode, path A will be replaced by another path, e.g., by path B. For instance, DSR establishes path B, excluding R, with the least hop counts as the alternate shortest routing path. Although path B is farther away with longer delay, it may (may not) consume more energy than those nodes on path A if the bandwidth of the two paths is the same [19]. This is helpful not only in prolonging the life time of an entire system, but also in preventing node R from running out of energy too quickly in transmitting packets from the event field (data source) to a base station, thus avoiding the occurrence of isolation sections. A path-discovery request to establish path B is initiated by, e.g., node S in Fig. 3, which may be either the data source node or the disappearing node's upstream neighbor depending on the re-routing policy of the host WSN. The former (the latter) is the case when the host routing scheme is a static (dynamic) routing.

3.4 Types of packets

Node energy is often consumed by sensing environment, transmitting data and relaying packets. In this research, we divide total energy E_{total} that a node currently has into four types E_x, E_y, E_z and E_u, which are defined in Table 1 where $E_{total} = E_x + E_y + E_z + E_u$, in which the initial energy allocated to E_u will be described later. The way to allocate remaining energy $E(= E_{total} - E_u)$ is as follows. Transmission priorities of type *x*, *y* and *z*, denoted by *Pri(x)*, *Pri(y)* and *Pri(z)* respectively, are *Pri(x)>Pri(y)>Pri(z)*. The allocation function of the three types' initial energy E_i is defined as $E_i = E \times C_i$ where C_i is the weight of type *i*, *i* = *x*, *y*, *z* and $\sum_{\forall i} C_i = 1$, i.e., a node reserves E_i for processing and transmitting type *i* packets on receiving a request *Req*. Before relaying (when *Req* is type y or z), transmitting (when *Req* is type x), or replying with a packet, a node, e.g., node R, along a routing path checks which type the request is and how much energy it currently has. If *E* is less than a predefined threshold δ, R rejects the request. Otherwise, if the request is type x, y, or z, R further checks to see whether there is enough of the corresponding type of energy to serve the request. If not, a reallocation is performed. We assume that $E_p = E_{pi}$ where E_{pi} is the energy required to transmit a packet of type i, i=x,y,z. The reallocation policy is that when a node runs out E_x, $(E_y + E_z) \times C_x$ is then allocated to type x constrained on the fact that new $E_x \geq E_p$.

Table 1 Energy consumption types on a node R

Type	Energy consumption description
E_x	energy for sensing environmental data and transmitting data, which is the quota for node R itself, i.e., data source is node R.
E_y	energy for relaying packets for neighbors, which is the quota for neighbors. Data source is a neighbor of R.
E_z	energy for relaying packets for others, which is the quota for those other than node R and R's neighbors. Data source is neither node R nor R's neighbors.
E_u	energy for processing, including receiving and transmitting/relaying/replying, a path-discovery packet.

As the node runs out E_y, E_z will be reallocated to types y and z as E_y' $(= E_z \times \frac{C_y}{C_y + C_z})$ and E_z' $(= E_z \times \frac{C_z}{C_y + C_z})$, respectively, also constrained on $E_y' \geq E_p$. A node (e.g., node R), when running out E_z, will reject type-z requests. If R rejects a request, either due to $E < \delta$ or since E_j is less than E_{pj} of a type j request, even if reallocation has just been completed, j=x, y or z, re-routing will be performed. If node R is a throat point of the underlying WSN, the new path-discovery request, carrying the source and destination node addresses which are the same as those either of a previous

path-discovery request rejected by R or of a previous connection for the continuous relaying of messages going through R that R rejected for some reason, reaches R again. This time the node, except for exhausting E after receiving the request, should accept this request and relay the following arriving packets to make the WSN robust enough to serve requests. Therefore, a parameter that records a node's routing history should be referred to.

3.5 Working algorithm

We assume that in an EnAR-WSN each packet carries destination and source addresses and all packets are of the same size because a WSN is an event report system and data conveyed on a packet is very often fixed in size. That is why we assume $E_{px} = E_{py} = E_{pz}$, which can simplify the following algorithms, their analyses, and the calculation of the energy consumed by a packet. A node, when receiving a request, checks to see whether its remaining energy is larger than a given threshold or not by invoking ChkReq(). If yes, the EnAR updates its energy by executing ChkServ(). ChkServ() will call ReallocEngy() to reallocate its remaining energy when E_x or E_y is used up. Furthermore, ChkReq() checks to see whether a request that a node receives has been rejected by the node within a period pre-defined of time or not by involving a timer. If yes, R should accept the request since re-routing after rejection selects R again.

ChkReq (p)

/** When node R receives a request (i.e., a packet) p which may be a path-discovery request (type u) or a packet of type i either from its upstream node M (i.e., type y or z) or from itself (i.e., type x), it checks to see whether p is acceptable or not; Also, assume that a node maintains a list L to keep its routing history ***/

1. {Let Sa and Da be the source and destination addresses of p, respectively, and L be a list of source and destination pairs (Sa, Da) with a counter $count_{Sa\_Da}$ and a timer t_{Sa_Da}. */*the counter counts the times having been rejected */*

2. if ((Sa, Da) is not in L) { $L = L \cup \{(Sa, Da)\}$; $count_{Sa\_Da}$=1; starts timer t_{Sa_Da}} else if (t_{Sa_Da} times out) $count_{Sa\_Da}$=0; */*(Sa, Da) already exists but the timer times out*/*

3. if ((E > threshold δ) or ($E > E_p$ and $count_{Sa\_Da} \geq 1$))
{Deci=ChkServ(p, $count_{Sa\_Da}$,t_{Sa_Da});

3.1. switch(Deci)

3.2. case (accepted): if p is a type-u packet, then reply to the request with an accept-ACK;
else {deliver the packet received/generated to the downstream node toward Da; $count_{Sa\_Da}$ =0;}

3.3. case (rejected): {reject the request; $count_{Sa\_Da}$++; enter rejection mode; start t_{Sa_Da} ;}

3.4. default: output an error message; */* p is a type other than u, x, y and z */* }

4. Else {reject p; $count_{Sa\_Da}$++; */* $E < \delta$, or the time point that R rejected last request was not within the pre-defined period of the time from right now */*}

ChkServ(p, $count_{Sa\_Da}$, t_{Sa_Da}) /*E_p is the energy required to transmit a packet p */

1. { switch (type(p))

2. case(u): if($count_{Sa\_Da} \geq 1$ and t_{Sa_Da} does not time out) return(accepted); */*a path-discovery request*/*

3. case(x): if ($E_x \geq E_p$){ $E_x = E_x - E_p$;return (accepted);}

else{ ReallocEngy(E_x , E_y , E_z , x, Decision); return(Decision);}

4. case(y): if ($E_y \geq E_p$){ $E_y = E_y - E_p$; return (accepted);}

else{ReallocEngy(E_x , E_y , E_z , y, Decision); return (Decision);}

5. case(z): if ($E_z \geq E_p$){ $E_z = E_z - E_p$; return (accepted);}

else return (rejected);

6. default: return (error);}

ReallocEngy(E_x **,** E_y **,** E_z **, Q, Decision) /*** to reallocate energy */

1. {if (Q==x) {

case($E_p \leq E < 2 \cdot E_p$):{ $E_x = E$; E_y =0; E_z =0; Decision=accepted;}

/* $E = E_x + E_y + E_z$ *is only sufficient to transmit a packet, then allocate energy to* E_x */

case($2 \cdot E_p \leq E < 3 \cdot E_p$):{ $E_x = 0.5 \times E$; $E_y = 0.5 \times E$; $E_z = 0$; Decision=accepted;} /*E is only sufficient to transmit two packets */

default: /* $E \geq 3 \cdot E_p$, *Distributed energy to* E_x , E_y *and* E_z *based on their weight* C_x , C_y *and* C_z */

{ $q = E - 3 \cdot E_p$; $E_i = q \times C_i + E_p$, $\forall i = x, y, z$; Decision=accepted;}}

2. else /* $Q==y$ */

if ($E_p \leq E_y + E_z < 2 \cdot E_p$){ $E_y = E_p$; $E_x = E - E_p$; E_z =0; Decision=accepted;}

/* $E_y + E_z$ *is only sufficient to transmit a packet, so give* E_p *to* E_y *and remaining energy to* E_x */

else /* $E_y + E_z \geq 2 \cdot E_p$ */

{ $q = E_y + E_z - 2 \cdot E_p$; $E_j = q \times \frac{C_j}{C_y + C_z} + E_p, \forall j = y, z$;

Decision=accepted; }}}

4. Experimental Results and Discussion

Three experiments were performed in this research. The first studied the difference between the lifetime of a WSN and an EnAR-WSN when different extreme amounts of initial energy are allocated to types x, y and z where system lifetime is the time period from the beginning of a system to the time point when the first node of the system dies. The second illustrated the ways to allocate initial energy to a node in a uniformly distributed WSN. The third compared system service time (in rounds) when different initial energy allocation approaches are used.

4.1 System Life Time

In the first experiment, each sensor was given only enough energy to transmit and receive 100 packets. The energy required for reallocation computation is one-tenth of that for transmitting a packet. In Fig. 4, *a*-*b*-*c* along x-axis means the energy reserves for types *x*, *y* and *z* of a node are *a*%, *b*% and *c*%, respectively, where $a+b+c=100$. It shows that the lifetimes of clustering approaches with EnAR and without EnAR (cluster with EnAR

and cluster without EnAR, in short, respectively) are similar on 0-0-100 because most nodes in both approaches exhausted most of their energy relaying packets for others (excluding a node itself and its neighbors). As compared with the cluster with EnAR in the cases of 0-100-0 and 100-0-0, the cluster with EnAR on 0-0-100 consumed the least energy (i.e., the difference between two poles of a pair, with and without the EnAR) to issue and process path-discovery packets. Its lifetime thus was longer than that of the other two.

The worst case occurs at 100-0-0. Because after a node enters its rejection mode, each time it receives a type-*y* or type-*z* request, a path-discovery request will be issued. Unfortunately, requests of these two types are the majority, particularly for those nodes near their base station. Such a node consumes a lot of energy to run ChkReq() and ChkServ(), and the system of such an allocation issues more path-discovery requests than 0-0-100 and 0-100-0. Both of these operations are harmful to system life. Our conclusion is that initial energy allocation significantly affects energy consumption.

To improve the situation, we deployed the *Pareto principle*, with which 20% of energy is reserved for a node itself and 80% for neighbors (30%) and others (50%). System lifetime (see 20-30-50 in Fig.4) was longer than that of the cluster approach without EnAR. Theoretically, its optimal initialization occurs when C_i (recall, weight of type *i*) is the probability that a node will receive a type-*i* request, *i=x,y,z,* during its lifetime. Formula (1) is an estimation.

$$C_i(t_k) = \frac{P_i(t_k)}{\sum_{j\in\{x,y,z\}} P_j(t_k)}, i = x, y, z \tag{1}$$

where $p_j(t_k)$ is the amount of type *j* requests received during the period of time from time points t_{k-1} to t_k , j=x, y, or z, and $\sum_{j\in\{x,y,z\}} P_j(t_k)$ is the total amount of requests of types x, y, and z received in this time period. The energy initially allocated to E_u is

$$E_u = E_{total} \times \frac{p_u}{\sum_{j\in\{x,y,z\}} p_j + \sum p_u} \tag{2}$$

where $\sum p_u (\sum_{j\in\{x,y,z\}} p_j)$ is the total amount of type-u (types x, y and z) packets a node has received in its historical records.

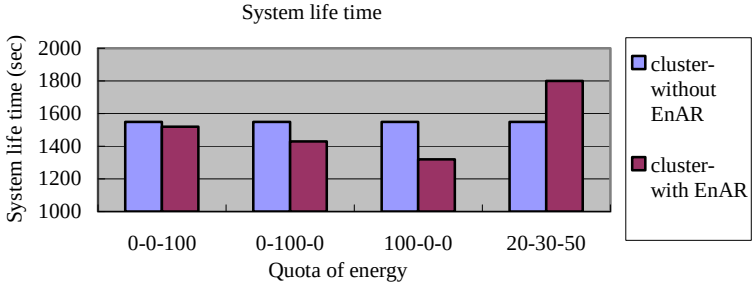

Fig. 4 *a*-*b*-*c* in x-axis means energy reserved for types *x*, *y* and *z* of a node is *a*%, *b*% and *c*%, respectively, and *a*+*b*+*c*=100. The difference between two poles of a pair (with and without the EnAR) is the energy consumed for reallocation and path-discovery.

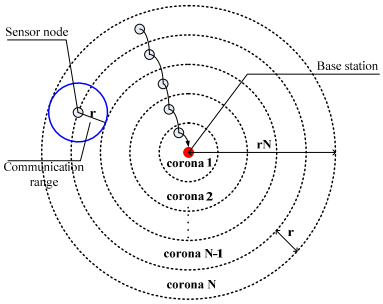

Fig. 5 Delineation of an EnAR-WSN for initial energy allocation

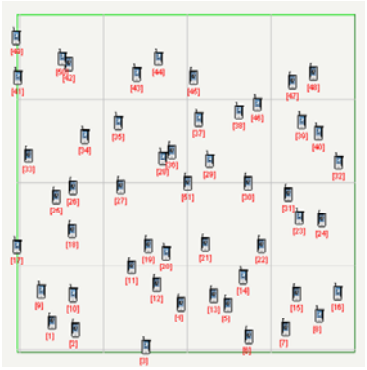

Fig. 6 Topology of a simulation environment of 200x200m^2 in which nodes are uniformly distributed and node 51 (at the center of this field) is a base station

4.2 Initial Energy Allocation

In the second experiment, we assume that there is a uniform distribution system of which, as shown in Fig. 5, the communication range of a node is *r* geographical distance units, and the data stream is forwarded to a base station hop by hop, i.e., a node in a corona, e.g., corona c, relays packets generated by nodes in outer corona, i.e., corona c+1, c+2, …, N, to its direct inner corona, i.e., corona c-1, where c=1,2,3 …, N-1 and corona 0 is a base station, This framework is the same as that illustrated in [20]. Assume that the probability of data generation is equal to all nodes, if we can preload different ratios of initial energy to sensor batteries in different coronas, the sensor network should have a better balance energy consumption model as we expect, so that initial energy allocated to a node is a function of c, i.e., $E_i = f_i(c),\ i = x, y, z$.

Let r_y and r_z be respectively the numbers of neighbors and others which a node has to relay packets for. When sensor density in the underlying field is high, r_y and r_z can be expressed as

$$r_y = \frac{(r(c+1))^2 - (rc)^2}{(rc)^2 - (r(c-1))^2} = \frac{2c+1}{2c-1} \tag{3}$$

$$r_z = \frac{(rN)^2 - (r(c+1))^2}{(rc)^2 - (r(c-1))^2} = \frac{N^2 - (c+1)^2}{2c-1} \tag{4}$$

Here, c also means that a node is c hops (c coronas) away from a base station and N is the total number of circles in the underlying system. Then, theoretically the optimal initial energy allocated to a node is as follows.

$$E_x = \frac{E}{1 + r_y + r_z} \tag{5}$$

$$E_y = \frac{r_y E}{1 + r_y + r_z} \tag{6}$$

$$E_z = \frac{r_z E}{1 + r_y + r_z} \tag{7}$$

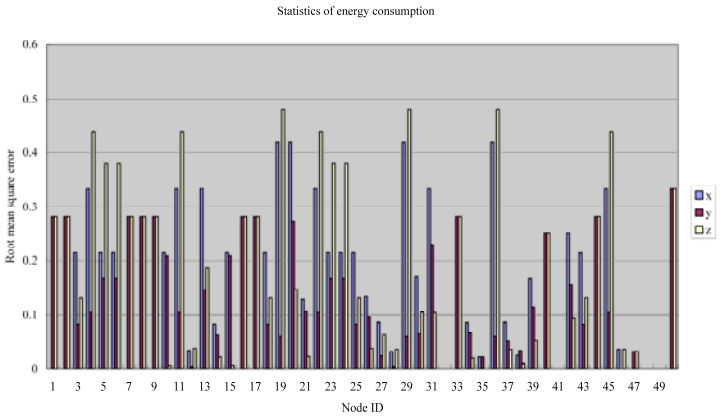

Fig. 7 Root mean square errors between simulated results by deploying formulas (5) to (7) and their theoretically optimal values in a static network topology deploying AODV without EnAR

In this experiment, we deployed 51 nodes for a sensing/event field and relayed packets generated by using static AODV without EnAR. The topology shown in Fig. 6 is produced by using QualNet [15], and is a field in which nodes are uniformly distributed. Fig. 7 illustrates the root mean square errors (RMSEs) of energy physically consumed by a node for processing requests of types x, y and z when initial energy is individually allocated by using formulas (5) to (7) and by using the theoretically estimated optimal values (i.e., formula (1)) where RMSE(x)=0.14317, RMSE(y)=0.132233, and RMSE(z)=0.18803, showing that the RMSEs do not very approximate to zero, owing to the steady path of static AODV. Some nodes one hop (circle) or a few hops (circles) away from a base station do not relay packets for other nodes, e.g., nodes 29 and 36 in Fig. 6, since most packets generated by their upstream nodes flow through nodes 30 and 28, respectively.

To simplify the following description, we again assume that the probabilities of data generation of all nodes in a WSN are the same, and the information required, i.e., the total number of packets that a node has received within a period of time or since the system started up, can be gathered from the system in some ways so that we can accordingly allocate initial energy to a node. For example, Fig. 8 shows that the number of packets that a node has received during a given period of time can be obtained by analyzing its system topology. Hence, the node can realize how to allocate its initial energy, e.g., initial energy allocated to types x, y and z for node A is $(\frac{1}{7}, \frac{2}{7}, \frac{4}{7})$ since it has two neighbors and four others. Actually, the information gathered from a real static WSN is often general enough to meet all cases, no matter which distribution (e.g., normal, uniform or Poisson) nodes in the WSN are? That means the information is better than that derived from a uniform distribution, i.e., formulas (5) to (7), since it can help to allocate more accurate initial energy so as to prolong the system life of the WSN.

4.3 System Service Time

In the third experiment, we compared the two schemes, formulas (5) to (7) and formula (1), to see how they affect the system service time of a WSN. The simulation environment deployed AODV routing scheme and 50 nodes which are randomly distributed in system field. For simplicity, we assume the energy consumption of generating and relaying a packet is the same, even it is not always true in real world. Fig. 9 shows the service time before the first node exhausts all its energy. It is clear that AODV-EnAR using for-

mulas (5) to (7) outperforms AODV without EnAR, but does not perform as well as AODV-EnAR using formula (1). The reasons are mentioned above.

Unfortunately, collecting packet statistics for deriving formula (1) sometimes is infeasible since the collection always consumes energy and should be performed system wide. Also, before allocating initial energy, we have to know the value of the denominator of formula (1). Furthermore, some routing schemes can not predict or obtain their routing topologies beforehand, e.g., direct diffusion [16], particularly when the system is a decentralized-control WSN. Our conclusion is that formulas (5) to (7) are feasible if r_y and r_z can be derived from system topology, e.g., based on the number of upstream hops which a node has to relay packets for.

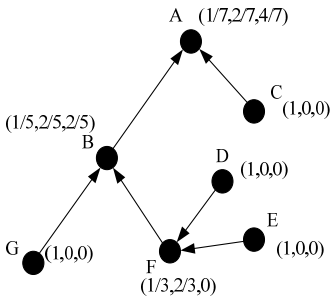

Fig. 8 A routing topology deploying AODV where (x,y,z) of a node represents the ratios of types x,y and z requests the node will receive

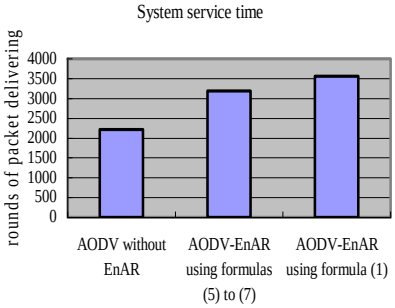

Fig. 9 Comparison of system service time when different initial energy allocation approaches are used

5. Conclusion and Future Research

In this study, we propose an energy-aware supplemental routing scheme, named the EnAR, to prolong a WSN's system life by routing packets through alternate paths with a decentralized approach. Nodes along the alternate paths will take over the task of relaying packets to prevent nodes on the original path not only from exhausting their energy quickly, but also from forming an isolated section no sooner after system start up. The EnAR should collaborate with path-preserving routing schemes. Experimental results showed that the collaboration can effectively prolong an underlying WSN's system lifetime. Also, the ways to allocate initial energy have been proposed and discussed. Among them formulas (5) to (7) are suitable for uniform distribution. Even if a base station is located at or near the edge of a field, they are still applicable. Formulas (1) and (2) can be applied to different distributions, uniform, normal, Poisson and so on, if the number of packets a node will receive can be obtained or predicted.

Our future work includes analyzing the cost and reliability models such as integrating the EnAR with different routing schemes, and studying other appropriate methods that can optimally allocate initial energy to requests of types x, y and z, besides the Pareto principle and formulas (1) and (2), particularly when nodes are distributed to a system field with different distributions, e.g., normal distribution and the stochastic approach, since in a WSN it is very difficult to collect global information, which also consumes a lot of energy and bandwidth. Furthermore, we will try to integrate MAC layer protocols, like sleeping mode scheduling, with the EnAR to further improve its system efficiency so as to save as much energy as possible, especially when the node density of a sensor network is high.

6. References

[1] Heinzelman W, Chandrakasan A, and Balakrishnan H (2000) Energy-Efficient Communication Protocols for Wireless Microsensor Networks (LEACH). *In Proc. the 33rd Hawaii International Conference on Systems Science*, volume 8, pages 3005-3014.

[2] Intanagonwiwat C, Govindan R, Estrin D, Heidemann J, andSilva F (2003) Directed Diffusion for Wireless Sensor Networking. *IEEE/ACM Transactions on Networking*, 11(1):2-16.

[3] Heinzelman W, Kulik J, and Balakrishnan H (1999) Adaptive Protocols for Information Dissemination in Wireless Sensor Networks. *In Proc. the Fifth Annual ACM/IEEE International Conference on Mobile Computing and Networking*, pages 174-185.

[4] Sohrabi K et al (2000) Protocols for Self-Organization of a Wireless Sensor Network. *IEEE Personal Communication*, 7(5):16–27.

[5] Chang R, and Kuo C (2006) An Energy Efficient Routing Mechanism for Wireless Sensor Networks. *In Proc. the International Conference on Advanced Information Networking and Applications*, volume 2, pages 18-20.

[6] Manjeshwar A, and Agrawal D (2001) TEEN: a Routing Protocol for Enhanced Efficiency in Wireless Sensor Networks. *In Proc. the International Parallel and Distributed Processing Symposium*, pages 2009-2015.

[7] Heinzelman W B, Chandrakasan A P, and Balakrishnan H (2002) An Application-Specific Protocol Architecture for Wireless Microsensor Networks. *IEEE Transactions on Wireless Communication*, 1(4):660-670.

[8] Lindesy S, and Raghavendra C (2002) PEGASIS: Power-Efficient Gathering in a Sensor Information System. *In Proc. the IEEE Aerospace Conference*, volume 3, pages 1125-1130.

[9] Lee M, and Wong V W S (2005) An Energy-aware Spanning Tree Algorithm for Data Aggregation in Wireless Sensor Networks. *In Proc. the IEEE Pacific Rim Conference on Communications, Computers and Signal Processing*, pages 300-303.

[10] Perillo M and Heinzelman W (2004) Dapr: A Protocol for Wireless Sensor Networks Utilizing an Application-based Routing Cost. *In Proc. the IEEE Wireless Communications and Networking Conference (WCNC 2004)*, volume 3, pages 1540–1545.

[11] Tian D, and Georganas N D (2002) A Coverage-Preserving Node Scheduling Scheme for Large Wireless Sensor Networks. *In Proc. the first ACM International Workshop on Wireless Sensor Networks and Applications*, pages 32-41.

[12] Deng J, Han Y S, Heinzelman B, and Varshney P K (2005) Scheduling Sleeping Nodes in High Density Cluster-based Sensor Networks. *Mobile Networks and Applications*, 10:825-835.

[13] Deng J, Han Y S, Heinzelman B, and Varshney P K (2005) Balanced-Energy Sleep Scheduling Scheme for High Density Cluster-based Sensor Networks. *Computer Communications*, 28(14):1631-1642.

[14] Tsai Y R (2007) Coverage-Preserving Routing Protocols for Randomly Distributed Wireless Sensor Networks. *IEEE Transactions on Wireless Communications*, 6:1240-1245.

[15] Scalable Network Technologies, http://www.scalable-networks.com

[16] Intanagonwiwat C, Govindan R, Estrin D, Heidemann J, and Silva F (2003) Directed Diffusion for Wireless Sensor Networking. *IEEE/ACM Transactions on Networking*, 11(1):2-16.

[17] Chamam A and Pierre S (2007) Optimal Scheduling of Sensors' States to Maximize Network Lifetime in Wireless Sensor Networks. *In Proc. the IEEE International Conference on Mobile Ad-hoc and Sensor Systems (MASS 2007)*, pages 1-6.

[18] Eryilmaz A, Modiano E, and Ozdaglar A (2006) Randomized Algorithms for Throughput-Optimality and Fairness in Wireless Networks. *In Proc. the 45th IEEE Conference on Decision and Control*, pages 1936-1941.

[19] Leu F Y, and Li G C (2007) A Scalable Sensor Network Using a Polar Coordinate System. *Signal Processing*, 87(12):2978-2990.

[20] Yang Y and Cardei M (2007) Movement-Assisted Sensor Redeployment Scheme for Network Lifetime Increase. *In Proc. the 10th ACM Symposium on Modeling, analysis, and simulation of wireless and mobile systems (MSWiM '07)*, pages 13-20.