Real-Time Routing in Wireless Sensor Networks: A Potential Field Approach

YINSHENG XU, FENGYUAN REN, TAO HE, and CHUANG LIN, Tsinghua University
CANFENG CHEN, Nokia Research Center
SAJAL K. DAS, University of Texas at Arlington

Wireless Sensor Networks (WSNs) are embracing an increasing number of real-time applications subject to strict delay constraints. Utilizing the methodology of potential field in physics, in this article we effectively address the challenges of real-time routing in WSNs. In particular, based on a virtual composite potential field, we propose the Potential-based Real-Time Routing (PRTR) protocol that supports real-time routing using multipath transmission. PRTR minimizes delay for real-time traffic and alleviates possible congestions simultaneously. Since the delay bounds of real-time flows are extremely important, the end-to-end delay bound for a single flow is derived based on the *Network Calculus* theory. The simulation results show that PRTR minimizes the end-to-end delay for real-time routing, and also guarantees a tight bound on the delay.

Categories and Subject Descriptors: C.2.1 [**Computer-Communication Networks**]: Network Architecture and Design—*Distributed networks*; C.2.2 [**Computer-Communication Networks**]: Network Protocols—*Routing protocols*

General Terms: Algorithms, Design, Experimentation, Performance

Additional Key Words and Phrases: Wireless sensor network, real-time routing, end-to-end delay, potential field, network calculus

ACM Reference Format:
Xu, Y., Ren, F., He, T., Lin, C., Chen, C., and Das, S. K. 2013. Real-time routing in wireless sensor networks: A potential field approach. ACM Trans. Sensor Netw. 9, 3, Article 35 (May 2013), 24 pages.
DOI: http://dx.doi.org/10.1145/2480730.2480738

1. INTRODUCTION

Wireless Sensor Networks (WSNs) are increasingly deployed to provide support for real-time applications subject to strict delay constraints. Such applications include medical care, emergency response, fire monitoring, border surveillance, and so on. These applications are desired to have low delay transmissions under resource constraints and environmental dynamics. For example, in the context of fire fighting,

This work was supported in part by the National Natural Science Foundation of China NSFC under grant no. 60971102, National Basic Research Program of China (973 Program) under grant nos. 2009CB320504, 2010CB328105, and 2012CB315803, and National Science and Technology Major Project of China (NSTMP) under grant no. 2011ZX03002-002-02.
Authors' addresses: Y. Xu (corresponding author), F. Ren, T. He, and C. Lin, Tsinghua National Laboratory for Information Science and Technology, Department of Computer Science and Technology, Tsinghua University, Beijing, 100084, China; email: xuysu08@csmet1.cs.tsinghua.edu.cn; C. Chen, NRC Radio Systems Lab, Nokia Research Center, Beijing, China; S. K. Das, Center for Research in Wireless Mobility and Networking (CReWMaN), Department of Computer Science and Engineering, the University of Texas at Arlington, Arlington, TX 76019.

DOI: http://dx.doi.org/10.1145/2480730.2480738

appropriate actions should be taken immediately in and around the event area, since any delay may cause huge damages including loss of lives. Similarly, a fire-fighter must rely on timely updates on temperature to remain aware of current fire conditions [Lu et al. 2005]. Thus, the data collection and delivery should still be valid at the time of decision making, otherwise invalid data could endanger the fire-fighter's life. Similarly, a surveillance monitoring system needs to alert appropriate authorities of an intruder immediately after an intrusion is detected [He et al. 2006]. In this article, we focus on the routing layer of WSNs to address the aforementioned real-time requirements.

However, it is very challenging to support real-time routing in resource-constrained WSNs. First, it is quite common that real-time (i.e., delay-sensitive) traffic as well as nonreal-time (i.e., nondelay-sensitive) traffic exist in the same WSN. The routing strategy should handle both of these packets, which may lead to congestions especially in a heavily loaded network. Second, the WSN has potentially lossy (wireless) links due to the environment around, which results in unpredictable transmission delay. In other words, if the routing layer is exploited for real-time transmissions, the unpredictable factors that may affect the delay in the Media Access Control (MAC) layer must be particularly removed. Besides, the end-to-end delay of real-time flows should be tightly bounded. Third, the wireless sensor nodes need to survive for a long time without additional energy supply and timely attention. Therefore, the routing algorithms in WSNs should satisfy real-time requirements with minimum energy consumption and high robustness, thus necessitating distributed routing algorithms rather than centralized ones.

In the literature, various methods have been proposed to solve the routing problems in WSNs, including those based on the concept of *potential field* in physics. The *gradient routing* in Poor [2000] is a novel approach to routing and control in wireless ad hoc networks. The packets are forwarded in a descending loop-free gradient from the origination to the destination. Utilizing the *steepest gradient search* method, a potential-based routing paradigm is proposed in Basu et al. [2003] in the traditional Internet. Nevertheless, it incurs huge management overhead. It is indeed expensive to build an exclusive virtual field for each destination in traditional Internet where numerous destinations are placed arbitrarily. On the contrary, the centralized traffic pattern in WSNs with a single destination (i.e., the sink) will spontaneously result in large decrease of the management cost when a potential-based routing algorithm is implemented, since only one field needs to be built. Another strategy, Yet Another Greedy Routing (YAGR) due to Na et al. [2007], utilizes the potential field to route in wireless ad hoc networks. YAGR eliminates the local maximum condition dynamically and achieves routing convergence, whereas it lacks further research on quality of service, especially under the resource limitations of WSN.

In our previous work [He et al. 2008][1] with the help of the potential technique, we proposed the Traffic-Aware Dynamic Routing (TADR) algorithm that routes packets around the congestion areas and caches the excessive packets along multiple underloaded paths. Although TADR performs well as a congestion alleviation solution, it may scatter packets from real-time applications into the longer paths and hence increase their end-to-end delays, which is unacceptable for real-time traffic. Considering the scenario of converge-cast (sending packets to the base station or sink) rather than the general point-to-point routing, the concept of potential field can still be useful and therefore investigated in depth. This motivates our work.

In this article, we propose a novel algorithm called Potential-based Real-Time Routing (PRTR) to primarily support real-time transmission in the WSN that carries *mixed* traffic composed of real-time and nonreal-time flows. To address the real-time routing

[1]The extended version of that paper can be found in IEEE TPDS [Ren et al. 2011].

challenges, PRTR builds up a virtual composite potential field through the convex combination of *node depth field* and *queue length field*. Both delay-sensitive and nondelay-sensitive packets are routed according to the corresponding fields, respectively. In the header of each packet, PRTR uses one bit *flag* to identify whether it demands low delay or not. Specifically, PRTR considers all the packets with flags 1 as delay sensitive, while those with 0 as nondelay sensitive. Moreover, an assistant mechanism called *priority queue* is used to decrease the queuing delay for real-time traffic, which cuts down the end-to-end delay further.

With the help of the virtual fields and the packet's identity, PRTR ingeniously provides real-time routing and simultaneously alleviates possible congestions. Since nondelay-sensitive packets are cached in the idle paths for multipath transmission, they are prevented from being dropped. Moreover, these packets provide an expeditious way (namely, the shortest path) to real-time flows, which minimizes the end-to-end delay of the latter and improves the overall throughput. The simulation results show that PRTR minimizes the end-to-end delay for real-time routing, and also guarantees a tight bound on the delay.

The rest of the article is organized as follows. We survey the related works in Section 2. The motivation of our work is given in Section 3 after introducing the basic scheme of the existing protocol TADR. Section 4 presents the fundamental idea of potential field technique and the construction procedure of the composite field. Section 5 describes in details how PRTR provides real-time routing and alleviates congestion. The end-to-end delay bound for a single flow is theoretically analyzed in Section 6. In Section 7, the integrated performance of PRTR is evaluated through simulation experiments in a randomly deployed network, with numerical examples of the delay bound illustrated. We draw conclusions in Section 8 with directions of future research.

2. RELATED WORKS

This section presents a comprehensive survey on real-time routing protocols in wireless sensor networks. In Pothuri et al. [2006], a heuristic solution is designed to compute energy-efficient paths for delay-constrained packets in WSNs. A set of paths between the source nodes and the sink are identified and indexed in increasing order of energy consumption. After evaluating the end-to-end delay of each ordered path, the path which satisfies the delay requirement with the lowest index is selected. Another novel Real-Time routing protocol with Load Distribution (RTLD) is presented in Ahmed and Fisal [2008], which ensures high packet throughput with minimum packet overhead, thereby prolonging the lifetime of the wireless sensor nodes. RTLD makes optimal forwarding decisions depending on the link quality, packet delay time, and the remaining power of next-hop sensor nodes. The Real-Time Power Control (RTPC) proposed in Chipara et al. [2005] utilizes velocity with the most energy-efficient forwarding choice as the metric for selecting the next-hop node. A key feature of RTPC lies in its ability to send data while adapting to the transmission power. The aforementioned works typically focus on the power constraint and real-time functionality at the same time.

It is known that wireless links of better quality usually provide low packet loss and improve energy efficiency [Zhao and Govindan 2003]. A routing protocol in Couto et al. [2003] based on the link quality is called the expected transmission count metric (ETX), which finds paths with the minimum expected number of transmissions required to deliver a packet all the way to its destination. The primary goal of ETX is to find the path with high throughput despite packet losses whereas it does not consider the remaining energy and end-to-end delay. The Maximum Capacity Path scheme (MCP) presented in Huang and Jan [2004] is an energy-aware multipath routing algorithm. In this scheme, the sensor network is first layered, and then routing decisions are made by finding a shortest path with maximum buffers to approach the sink.

There also exist several real-time routing protocols that adopt the velocity assignment policy. For example, in Lu et al. [2002] the Real-time Architecture and Protocols (RAP) is developed based on velocity. In the routing layer, RAP utilizes an existing Geographic Forwarding (GF) routing protocol that makes a greedy decision to forward a packet to a neighbor if: (1) it has the shortest geographic distance to the packet's destination among all immediate neighbors; and (2) it is closer to the destination than the forwarding node. When such nodes do not exist, RAP uses another existing Greedy Perimeter Stateless Routing (GPSR) protocol to route packets around the perimeter of the void region. In the SPEED protocol [He et al. 2003], the packet deadline is mapped to a velocity in terms of the distance to the destination. A packet is forwarded by a node if it can meet the required velocity. When there is no neighboring node to meet the requirement, the packet is dropped probabilistically while regulating the workload. SPEED utilizes a Stateless Nondeterministic Geographic Forwarding (SNGF) protocol, which improves the pure GF protocol to achieve real time with theoretical delay bound and load balance. So we consider SPEED is superior to RAP in terms of end-to-end delay. Note that GF, GPSR, and SNGF incur extra overhead to maintain the coordinates of each sensor node. In order to be aware of the coordinates, extra overhead such as a locating device like GPS or signal strength information that is used to calculate the distance between nodes is required. Besides, SPEED handles congestion by throttling or rerouting the incoming traffic around the hot spot. The rerouted path, however, may not have a larger end-to-end channel capacity to accommodate the incoming traffic, hence leading to congestion. Since SPEED is a representative real-time routing protocol in the WSN, we reimplement it and set it as a benchmark for performance comparisons with our proposed PRTR protocol in Section 7.

As an extension of SPEED, MM-SPEED [Felemban et al. 2005] supports multiple communication speeds and provides differentiated reliability. MM-SPEED is also realized in a localized GF way without global network information, augmented with dynamic compensation for the local decision inaccuracy. The Real-time Power-Aware Routing (RPAR) protocol [Chipara et al. 2006] extends SPEED in another way. It achieves application-specified transmission delays at minimum energy cost by dynamically adapting transmission power and routing decisions. If none of the nodes can meet the velocity required, the transmission power will be adjusted to attempt another discovery. RPAR features a power-aware routing strategy and an efficient neighborhood manager which are optimized for resource-constrained wireless sensor nodes.

3. MOTIVATION OF THIS WORK

Based on the similar scenarios of the WSN, we make a simple comparative illustration to motivate this work. Figure 1 and Figure 2 depict the same part of a wireless sensor network that carries mixed traffic, but routed by the TADR and PRTR schemes, respectively. As illustrated in Figure 1, the basic policy of the TADR scheme [He et al. 2008; Ren et al. 2011] is briefly described as follows. Suppose the shortest path is congested due to numerous nondelay-sensitive packets from node B. To alleviate the congestion on that path, TADR will stop nodes A and C injecting packets into the heavily loaded shortest path, and find another path for these excessive packets to bypass the congested area. For example, nodes A and C will forward packets along other longer and lightly-loaded paths to reach the sink (e.g., A-4-5-6-sink and C-2-3-sink in Figure 1). TADR aims at intentionally finding underloaded paths, which is compliant with the idea of dynamic capacity planning to avoid congestion.

Unfortunately, TADR is not able to distinguish which packet demands real-time routing. Although this protocol performs well when congestion occurs, it does not take real-time requirements into account since node A or C can neither identify the delay-sensitive packets nor forward them intentionally to the shortest path. As

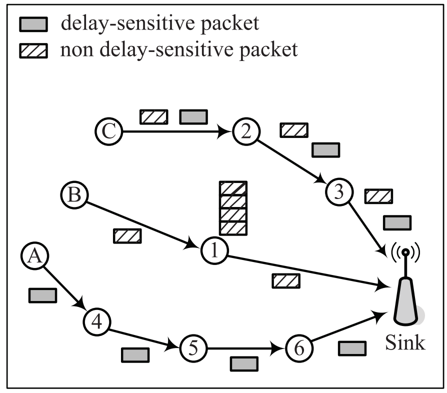

Fig. 1. Routing policy of TADR.

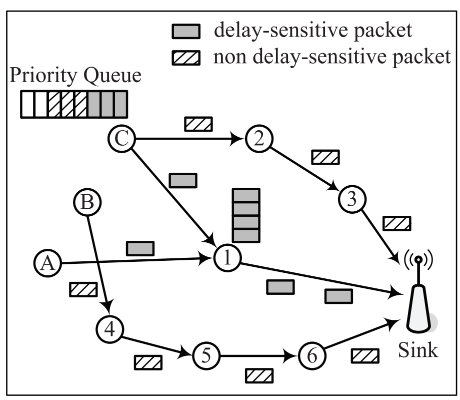

Fig. 2. Motivation of PRTR.

a consequence, TADR may scatter packets from real-time applications into longer paths when it takes actions on congestion alleviation (i.e., delay-sensitive packets are moved to paths A-4-5-6-sink and C-2-3-sink in Figure 1), and thus may increase the end-to-end delay of these packets. Motivated by this point, we propose in this article a novel strategy, called Potential-based Real-Time Routing (PRTR), with a goal to minimize the delay of real-time flows.

In the context of PRTR, which is able to identify each packets via the *flag* in its header, the same scenario is handled in a different way (as illustrated in Figure 2). PRTR ensures delay-sensitive packets traverse along the shortest path and in no case be scattered to other longer paths (namely, this kind of packets from A and C are only moved to the shortest path, as shown in Figure 2), which utterly minimizes their end-to-end delay. Meanwhile, the other kind of packets (from nonreal-time applications) will be scattered to some idle or lightly loaded paths to be cached for multipath transmissions to reach the sink. For example, in Figure 2, nondelay-sensitive traffic from B and C are scattered to path 4-5-6-sink or 2-3-sink to bypass the hot spot node 1. These actions minimize the delay of real-time traffic and alleviate congestion, hence providing real-time routing and improving the throughput of the whole network simultaneously. Furthermore, PRTR uses a *priority queue* to transfer the delay-sensitive packets ahead of the others, which further decreases their end-to-end delay. Specifically, the front of the buffering queue in node C always consists of real-time packets. In other words, PRTR distinguishes the two kinds of packets and performs the corresponding actions on them.

When the total network load consisting of mixed traffic is light, PRTR indeed performs well. However, if the nonreal-time traffic load starts to increase, PRTR will scatter them to bypass the shortest path, thus alleviating possible congestions, because the shortest path is dedicated to the real-time traffic. On the other hand, if the real-time traffic starts to increase, congestion may occur along the shortest path, implying some real-time packet loss. This loss may be endurable for low real-time traffic up to a certain threshold. But if the real-time traffic increases beyond the specified threshold, some delicate flow or congestion control mechanisms need to be designed in order to reduce the sending rate of the source. The measure of specifying threshold or control mechanisms is beyond the scope of this work, because our main focus in this article is how to exploit low delay transmission in the routing layer, especially in *mixed* traffic scenarios. Next, the potential field technique is introduced to achieve the aforementioned goals.

4. DESIGN OF POTENTIAL FIELDS

We design a virtual composite potential field in three steps. First, two independent potential fields using node depth and queue length are constructed, respectively. Second, their convex combination is obtained to form one composite potential field. Third, the maximum force rule is used to choose the next-hop node. These are described next.

4.1. Potential Field Model

Similar to the concept of potential field in physics, every node v residing in the WSN is assigned a scalar value $V(v)$ provided that a scalar potential field V is established. Supposing a packet p at node v attempts to reach the sink, p should be forwarded to one of the neighbors of v (in the rest of the article, we denote $nbr(v)$ as the neighbor set of node v). Here, $nbr(v)$ only comprises all the nodes that are one radio hop away from v. To select the next-hop node and drive the packets flowing in accordance with our motivation, the force acting on the packet p at node v roots in the potential difference between v and its neighbor $w \in nbr(v)$, which is defined as

$$F(v, w) = \frac{V(v) - V(w)}{D(v, w)}, \tag{1}$$

where $D(v, w)$ denotes the distance between two nodes v and w.

As far as $w \in nbr(v)$ is concerned, $D(v, w)$ is considered as 1 since in PRTR the distance between nodes is measured by radio hops. Hence Eq. (1) can be rewritten as

$$F(v, w) = V(v) - V(w). \tag{2}$$

Given this field model, we bring it into effect step by step.

4.2. Node Depth Potential Field

To actualize the basic routing functionality, the node depth field is built to ensure all the packets move toward the sink. The depth potential value $V_d(v)$ at node v is defined as

$$V_d(v) = \sqrt{D(v)}, \tag{3}$$

where $D(v) \geq 0$ and $D(v) \in \mathbb{Z}$. Generally, the sink is the only node at depth 0, that is, $D(Sink) = 0$. For any other nodes x, its depth $D(x)$ indicates the distance between x and the sink. By this notion, nodes in $nbr(Sink)$ are at depth 1, their farther neighbors are at depth 2, and so on. Thus according to Eq. (2), the force $F_d(v, w)$ produced by the

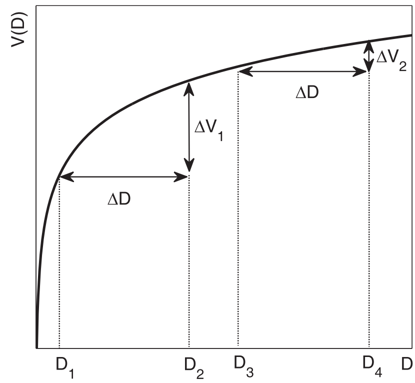

Fig. 3. The depth potential value $V(D)$ as a square root function of node depth D.

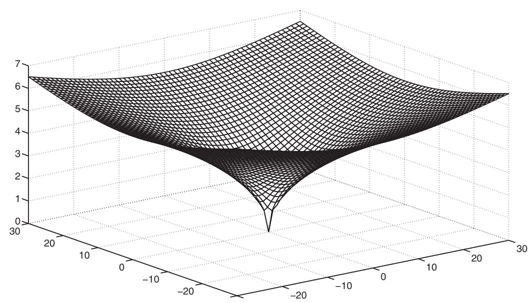

Fig. 4. The depth potential field model of three dimensions.

depth potential difference between nodes v and $w \in nbr(v)$ is given by

$$F_d(v, w) = V_d(v) - V_d(w). \tag{4}$$

The reason why $V_d(v)$ is defined by a (square root function lies in the fact that the curve of this function (see Figure 3) will become steeper as $D(v)$ decreases. In other words, the depth potential difference $\Delta V(D)$ is calculated as

$$\Delta V(D) = \sqrt{D + \Delta D} - \sqrt{D}, \tag{5}$$

which can be rewritten as

$$\Delta V(D) = \frac{\Delta D}{\sqrt{D + \Delta D} + \sqrt{D}}. \tag{6}$$

Evidently, $\Delta V(D)$ is inversely proportional to the node depth D if ΔD is fixed (e.g., if $D_2 < D_4$ then $\Delta V_1 > \Delta V_2$ in Figure 3). Namely, the node depth field tends to be a dominated factor when the packets approach the sink, while other factors may dominate the protocol at places that are far away from the sink. This property spontaneously improves the performance of PRTR, since it is possible for nondelay-sensitive packets to be scattered from the shortest path and give the way to the real-time flows.

Figure 4 illustrates the depth potential field of three dimensions, which intuitively looks like a bowl. The sink resides at the bottom, and all data packets flow down along the surface just like water drops. The PRTR algorithm constructs this smooth bowl using node depth which is defined as the least hops away from the sink. In lightly loaded networks with the surface of the bowl smooth, the routing algorithm just needs to move packets along the shortest path. Nevertheless, in heavily loaded situations such as a burst of data caused by detection of an outburst event, the traffic congestion emerges and forms some bulges on the surface of the bowl, which will stop the excessive packets from going directly down to the bottom. At that moment, it is necessary to find an appropriate detour path with idle nodes for those packets, and this forwarding force actually roots in another potential field built in the next subsection. When the congestion disappears, the bowl surface resumes smooth with packets forwarded along the shortest path again.

4.3. Queue Length Potential Field

As mentioned earlier, the node depth field ensures all the packets flow toward the sink, thus leaving the congestion problem unaddressed. To intentionally find an appropriate detour path to forward excessive packets and help them bypass the hot spots when congestion occurs, the queue length potential value at node v is defined as

$$V_q(v) = Q(v), \tag{7}$$

where $Q(v)$ denotes the normalized queue length at node v and is given as

$$Q(v) = \frac{\text{number of packets in the queue}}{\text{buffer size at node } v}. \tag{8}$$

Likewise, the force $F_q(v, w)$ generated by the queue length potential difference between nodes v and $w \in nbr(v)$ is

$$F_q(v, w) = V_q(v) - V_q(w), \tag{9}$$

where the range of $Q(v)$ is obviously [0, 1], and hence we obtain $F_q(v, w) \in [-1, 1]$.

Driven by this potential field, the packets will always be forwarded toward the underloaded nodes with smaller $Q(v)$, and accordingly bypass the congested areas.

4.4. Composite Potential Field

The preceding two independent potential fields are built respectively to provide two basic functions, namely forwarding packets toward the sink and toward the underloaded nodes. A composite potential field integrating them is derived through a convex combination as

$$V_c(v) = (1 - \alpha)V_d(v) + \alpha V_q(v), \tag{10}$$

where $V_c(v)$ is the potential value of the composite field at node v, and the weighing parameter $\alpha \in [0, 1]$. Then the composite force between nodes v and $w \in nbr(v)$ in this field is given by

$$F_c(v, w) = V_c(v) - V_c(w) \tag{11}$$

which can be rewritten as

$$F_c(v, w) = (1 - \alpha)F_d(v, w) + \alpha F_q(v, w). \tag{12}$$

The two independent fields properly interact together to support the required routing functionalities. By adjusting the weighing parameter α, various composite fields can be derived to solve the corresponding routing issues.

4.5. Maximum Force Rule

Now that all the packets are driven by the composite force, they will be spontaneously forwarded to the neighbor in the direction of the maximum force as defined next.

Maximum Force Rule : w is the next-hop neighbor of node v iff

$$w \in \mathop{arg\ max}_{x \in nbr(v)} F_c(v, x). \tag{13}$$

Also, this way is exactly the direction of the steepest gradient of the potential decline. The PRTR algorithm makes reasonable routing decisions based on this principle. As a whole, in a lightly loaded network, the maximum force will move packets along the shortest paths. Otherwise, it will dynamically choose multiple paths except for the shortest one, due to the queue length potential field dynamically changing. Consequently, it is possible for PRTR to find an appropriate detour path to cache excessive packets when the congestion occurs, which decreases the packet loss at nodes in the shortest path dedicated to the real-time traffic.

5. MAIN FEATURES OF PROPOSED PRTR ALGORITHM

Before we introduce the main features of PRTR, it is important to identify the packets in the WSN. A one-bit *flag* is set in the header of each packet so as to sort the packets into two different categories. The flags of packets from real-time applications are set to 1 while the others are set to 0. Subsequently, every packet is appropriately routed in its corresponding composite field, which can be acquired by specifying the weighing parameter α.

5.1. Real-Time Routing

The primary goal of PRTR is to serve real-time traffic. Intuitively, when all the packets from real-time applications (i.e., those with flag 1) move along the shortest path to the sink, they can spontaneously obtain the minimum delay among all other paths. Accordingly, the parameter α in Eq. (10) is set to 0 and the composite field for real-time routing will only be built by the depth field, namely,

$$V_1(v) = V_d(v), \tag{14}$$

where $V_1(v)$ denotes the first composite field really implemented in PRTR. In this case, PRTR degenerates as a shortest path routing algorithm for delay-sensitive packets. Since $V_d(v)$ is a monotone increasing function of the depth of node v (see Figure 3), the following proposition can be stated.

PROPOSITION 5.1. *Given node v in depth d, let $S = \{x|D(x) = d, x \in nbr(v)\} \cup \{v\}$, $L = \{x|D(x) = d - 1, x \in nbr(v)\}$ and $H = \{x|D(x) = d + 1, x \in nbr(v)\}$. Then $l \in L$ is chosen as the next hop of v.*

Following the maximum force rule, it is easy to prove Proposition 5.1. As a consequence, all real-time flows are aggregated at the expeditious shortest path so that their end-to-end delays are minimized. If there exist more than one shortest path, a random choice will be made to obey the load-balance principle.

5.2. Congestion Alleviation

It is quite common that real-time and nonreal-time applications coexist in the same WSN. Provided that packets from both kinds of applications are all forwarded along the shortest path, the congestion problem occurs soon. Suppose the heavily loaded nodes with large queues reside in the shortest path. The traffic congestion will form some bulges on the surface of the bowl (see Figure 4), which will stop the excessive packets from flowing directly down to the sink and drive them out of the path to find other idle or underloaded paths. As discussed in Section 4, the congestion will be efficiently alleviated if the queue length potential field is also employed to find an appropriate detour path for nondelay-sensitive flows.

Therefore, the parameter α in Eq. (10) should neither be 0 nor 1, meaning the queue length field now participates in and the composite field for nonreal-time traffic is built, that is,

$$V_2(v) = (1-\alpha)V_d(v) + \alpha V_q(v), \tag{15}$$

where $0 < \alpha < 1$, and $V_2(v)$ denotes the second composite field implemented in PRTR.

It is necessary for a node to know exactly which neighbor is underloaded rather than where the lightly loaded paths are. Namely, if a node can forward excessive packets to its neighbor in the same depth with idle buffers, it succeeds in finding an appropriate detour path to cache packets and alleviate congestion. These are summarized in the following proposition.

PROPOSITION 5.2. *Given node v in depth d, let $S = \{x|D(x) = d, x \in nbr(v)\} \cup \{v\}$, $L = \{x|D(x) = d-1, x \in nbr(v)\}$, and $l \in L$ be the node with the minimal queue length in L. Then s can be chosen as the next-hop node of v if $\exists s \in S$ meets the following condition (for $0 < \alpha < 1$).*

$$Q(s) < Q(l) - \left(\frac{1}{\alpha} - 1\right)\frac{1}{2\sqrt{d}} \tag{16}$$

PROOF. Note that, if v does not choose l as the next hop, it will not choose any other nodes in L since the queue length of l is already the minimum. Now the potential values at each of these nodes are as fellow.

$$V_2(v) = (1-\alpha)\sqrt{d} + \alpha Q(v)$$
$$V_2(l) = (1-\alpha)\sqrt{d-1} + \alpha Q(l)$$
$$V_2(s) = (1-\alpha)\sqrt{d} + \alpha Q(s)$$

Then, the forces acting on the packets at node v are calculated as

$$F_2(v,l) = V_2(v) - V_2(l)$$
$$F_2(v,s) = V_2(v) - V_2(s).$$

According to the maximum force rule, if node v chooses s as its next hop instead of l, the force between v and s should be larger than that between v and l, such that

$$F_2(v,s) > F_2(v,l)$$
$$V_2(l) > V_2(s)$$

which can be rewritten as

$$\alpha Q(l) - \alpha Q(s) > (1-\alpha)(\sqrt{d} - \sqrt{d-1}) \tag{17}$$

$$(1-\alpha)(\sqrt{d} - \sqrt{d-1}) = (1-\alpha)\frac{1}{\sqrt{d} + \sqrt{d-1}} \tag{18}$$

$$(1-\alpha)\frac{1}{\sqrt{d} + \sqrt{d-1}} > (1-\alpha)\frac{1}{2\sqrt{d}}. \tag{19}$$

Combining Eqs. (17), (18), and (19), we obtain

$$\alpha Q(s) < \alpha Q(l) - (1-\alpha)\frac{1}{2\sqrt{d}} \tag{20}$$

when $\alpha \in (0,1)$ and both sides of the inequality (20) can be divided by α. This yields inequality (16) and proves the proposition. □

Remarks. Proposition 5.2 indicates that packets from node v can be forwarded to its neighbors at the same depth since they have more buffers to cache excessive packets. The caching of nondelay-sensitive packets helps alleviate the congestion on the shortest path. Let us review inequality (16). In a certain depth d, the weighing parameter α is actually the factor of the queue length difference between s and l that v begins to scatter its packets for multipath transmission. A corollary can be straightforwardly derived, implying different α results in different impacts on the performance of PRTR.

COROLLARY 5.3. *Nondelay-sensitive packets with flag 0 will be more easily driven out of the shortest paths to bypass the congested areas when α is set larger.*

Furthermore, it is also reasonable that nonreal-time packets from node v prefer to choose their next hops with nodes at higher level rather than at lower or same level. This phenomenon is similar to some kind of *counterflow*. Hence we derive the following proposition to clearly explain it.

PROPOSITION 5.4. *Given node v in depth d, let $S = \{x|D(x) = d, x \in nbr(v)\} \cup \{v\}$, $L = \{x|D(x) = d - 1, x \in nbr(v)\}$, $M = \{x|D(x) = d + 1, x \in nbr(v)\}$, $s \in S$ and $l \in L$ be the nodes with the minimal queue length in their depth. Then m can be chosen as the next-hop node of v if $\exists m \in M$ simultaneously meets the following two conditions (for $0 < \alpha < 1$).*

$$Q(m) < Q(l) - \left(\frac{1}{\alpha} - 1\right)\frac{1}{\sqrt{d+1}}$$

$$Q(m) < Q(s) - \left(\frac{1}{\alpha} - 1\right)\frac{1}{2\sqrt{d+1}}$$

PROOF. The proof is similar to that of Proposition 5.2, so we omit it here. □

Remarks. Proposition 5.4 implies the existence of *counterflow* phenomenon, which is another kind of caching operation similar to Proposition 5.2. Obviously, if the weighing parameter α is set at a large value, Proposition 5.4 will also lead to Corollary 5.3. However, if we set α small enough, the caching operation can be totally avoided.

Moreover, if d and α are both assigned certain values, an important threshold can be derived. According to inequality (16), we define

$$\triangle Q = Q(l) - Q(s) \tag{21}$$

$$\triangle Q_t = \left(\frac{1}{\alpha} - 1\right)\frac{1}{2\sqrt{d}}. \tag{22}$$

Apparently, $\triangle Q_t$ is the threshold of allowing neighbors in the same depth to compete as the next hop (if $\triangle Q > \triangle Q_t$, node v begins to scatter the packets). In the implementation of PRTR, the update message will be exchanged when the queue length variation of a node exceeds $\triangle Q_t$, which decreases the overhead for updating the queue length information.

In other words, both Propositions 5.2 and 5.4 explain the caching operations that only impact on the nondelay-sensitive packets. When congestions are alleviated or eliminated with the help of caching, those packets will be forwarded toward the sink because of the monotonic depth field.

5.3. Coexistence of Two Composite Fields

PRTR eventually establishes two specific composite fields. $V_1(v)$ only serves packets from real-time applications while $V_2(v)$ only impacts on the nonreal-time packets. In practice, two independent routing tables are implemented respectively at each node to represent the corresponding fields. Only for the first time the node receives a packet, PRTR checks its flag and thereby determines which routing table it relates to. Although PRTR utilizes two routing tables, it is not required to maintain them across the whole network all the time (details of the update table are provided in Section 7.1). Besides, nodes in PRTR just need local information of depth and queue length that can be easily obtained within one-hop update exchanges, which significantly decreases the overhead and eases the implementation.

5.3.1. A Word on Local Optima. The local optima denote the situation that at some instants, some nodes are better than all of their neighbors. Consequently, the local optimal node cannot find its next hop at that moment. As for the potential field V_1, this situation will not happen because real-time packets routed by V_1 just choose their next hop at the next lower level (i.e., V_1 is a monotonic field). However, as for the potential field V_2, at some instants, the local optima could appear. The local optimal node may have relatively small queue length (i.e., lightly loaded nodes), in which case it will not send the nonreal-time packets to any neighbor but cache them and wait. For example, the node v in Proposition 5.2 can also choose itself as the next hop, which means that the packets will stay in v rather than being sent out (otherwise the next-hop congested node may drop these packets). Once the queue length of that node becomes large enough, the local optimum will disappear automatically. This caching process is reasonable and acceptable for nondelay-sensitive applications. Besides, due to the timely updates of queue length information, both the appearance and disappearance of local optima will be fast. Therefore, the routing convergence will not be harmed. In other words, the local optima caused by V_2 will not incur serious performance degradation of PRTR over the whole network in the long run.

5.4. Priority Queue

To further decrease the queuing delay for real-time traffic, PRTR exploits an assistant mechanism priority queue to allow these packets to transmit before others. Naturally, the flag of each packet is used as the priority identifier. Packets with flag 1 occupy the first several places while the others queue in turn. Packets with the same flag are ordered according to their arrival time, namely a FIFO rule. Since the priority queue impacts the performance of PRTR, a comparative simulation is carried out in Section 7.3 to emphasize its importance.

6. DELAY BOUND ANALYSIS

Since the delay bound is extraordinarily important, the end-to-end delay bound for a single flow routed by PRTR is derived based on the *Network Calculus theory* [Boudec and Thiran 2001]. Similar to Lenzini et al. [2006], our methodology applies network calculus iteratively so as to obtain a set of end-to-end service guarantees for a flow. Afterwards, the least upper bound on delay is computed with the help of all the bounds derived from each single end-to-end service guarantee. We first introduce the necessary backgrounds.

6.1. Network Calculus Background

The *Network Calculus* is a theory for deterministic queueing systems analysis [Boudec and Thiran 2001]. The fundamental concept of *Service Curve (SC)*, allows one to model a network node (or a tandem of network nodes) in terms of worst-case relationships between the input and output of a flow traversing that node. Meanwhile, the worst-case traffic arrivals for a flow in any time interval is represented by means of its *Arrival Curve* (AC). By measuring the maximum horizontal distance between a flow's arrival curve $\lambda(t)$ and a node's service curve $\beta(t)$, we are able to calculate the delay bound $h(\lambda, \beta)$ for the flow at that node. The fundamental definitions and theorems applied to this section are complemented shortly, all of which refer to Boudec and Thiran [2001].

Definition 6.1 (*Min-Plus Convolution and Deconvolution*). As the primary operators, the min-plus convolution and deconvolution of two functions f and g are defined as

$$(f \otimes g)(t) = \inf_{0 \le s \le t} f(t-s) + g(s) \tag{23}$$

$$(f \oslash g)(t) = \sup_{u \ge 0} f(t+u) - g(u). \tag{24}$$

Definition 6.2 (*Arrival Curve*). Given a flow with input function R, a function $\lambda(t)$ is an arrival curve for R iff $\forall t, 0 \le s \le t$,

$$\begin{aligned} & R(t) - R(t-s) \le \lambda(s) \\ \Leftrightarrow\ & R \le R \otimes \lambda \\ \Leftrightarrow\ & \lambda \ge R \oslash R. \end{aligned} \tag{25}$$

As an example, a flow regulated by a *leaky bucket* regulator, with rate r and burst size b, is constrained by the arrival curve $\lambda_{r,b}(t)$ and

$$\lambda_{r,b}(t) = (rt + b) * 1_{t>0}, \tag{26}$$

where the indicator function 1_{expr} is equal to 1 if *expr* is true, and 0 otherwise.

Definition 6.3 (*Service Curve*). Let us suppose that a server process is able to cope with $C(t)$ bits of input data until time t. Then, $\beta(t)$ is a minimum service curve iff $\forall t, 0 \le s \le t$,

$$C(t) - C(t-s) \ge \beta(s). \tag{27}$$

Given the rate $R \ge 0$ and the latency $\theta \ge 0$, a typical *rate-latency* service curve can be defined as

$$\beta_{R,\theta}(t) = R[t-\theta]^+, \tag{28}$$

where notation $[x]^+$ denotes $\max(0, x)$.

Utilizing Theorem 6.4, the delay bound can be calculated.

THEOREM 6.4 [DELAY BOUND]. *Consider a system S that offers service curve β and stores input data in a FIFO queue. Assume a flow R traversing the system that has an arrival curve λ. Then the delay is bounded by horizontal deviation*

$$h(\lambda, \beta) = \sup_{t \ge 0}[\inf d \ge 0 : \lambda(t-d) \le \beta(t)] \tag{29}$$

which actually can be computed through

$$h(f, g) = \sup_{t \ge 0} g^{-1}(f(t)) - t, \tag{30}$$

where f and g denote two functions.

One of the strongest results of Network Calculus (albeit being a simple consequence of the associativity of $\otimes$) is the concatenation Theorem 6.5, which enables us to investigate tandems of systems as a single system.

THEOREM 6.5 [CONCATENATION THEOREM FOR TANDEM SYSTEMS]. *Consider a flow that traverses a tandem of systems S_i ($i = 1, \ldots, n$). Assume that S_i offers a service curve*

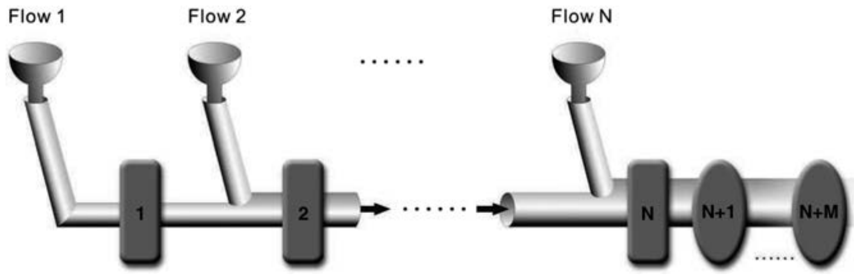

Fig. 5. The system model. A sink-tree tandem along the shortest path in PRTR. Rectangles and ellipses represent nodes. Nodes labeled from 1 to N aggregate flows while nodes from N+1 to N+M just relay packets. The funnels represent injection of traffic flows and pipelines denote the aggregate of flows.

β_i ($i = 1, \ldots, n$) to the flow. Then the concatenation of the two systems offers a service curve $\bigotimes_{i=1}^{n} \beta_i$ to the flow.

Another fundamental result given by Boudec and Thiran [2001] deals with the equivalent service curve of FIFO multiplexing. Suppose flows 1 and 2 are FIFO multiplexed into the same node, which is characterized by the service curve $\beta(t)$. Let $\lambda_2(t)$ be an arrival curve for flow 2. Then, the service received by flow 1 can be described in terms of an equivalent service curve β_1^{eq}, as follows.

THEOREM 6.6 [FIFO MULTIPLEXING SERVICE CURVES].

$$\beta_1^{eq}(t, \tau) = [\beta(t) - \lambda_2(t - \tau)]^+ 1_{t>\tau} \tag{31}$$

For ease of notation, $E(\beta, \lambda, \tau)$ is used to denote the right part of Eq. (31) in the subsequent sections.

6.2. System Model

The theoretical calculations start with modeling the system as illustrated in Figure 5. To derive the upper bound on the performance of PRTR, we analyze a sensor network tandem traversed only by real-time flows. Since PRTR makes all the real-time traffic forwarded along the shortest path, delay-sensitive data flows will aggregate as one at nodes on that path, which forms the system as a sink-tree tandem. The former part (rectangles) of the system model in Figure 5 depicts that tandem. The latter part (ellipses) indicates that, from node $N+1$ the aggregate is only relayed on the last hops of its trip without injecting new flows (suppose the next hop of node $N+M$ is the sink), which is in accordance with the real scenario in WSN using PRTR.

After injecting to the shortest path at node i ($1 \leq i \leq N$), the flow i's path through the network is a sequence of nodes from i to $N + M$. From node $i + 1$ to node N, the flow i is aggregated with other flows (i.e., from flow $i + 1$ to flow N) in a FIFO multiplexing order. If a real-time packet enters the queue after a non real-time one, the priority queue mechanism will immediately move it to the front of the latter one, which ensures the FIFO multiplexing order for all the real-time flows. Since the WSN is homogeneous, it is reasonable to model all the nodes with rate-latency service curves $\beta_{R,\theta}(t)$ in Eq. (28), and all the flows with leaky bucket arrival curves $\lambda_{r,b}(t)$ in Eq. (26), respectively.

6.3. Delay Bound Analysis

Like the techniques adopted by Lenzini et al. [2004], it takes two steps to derive the end-to-end delay bound for flow 1 traversing from node 1 to node $N + M$ in the system model. First, an equivalent end-to-end service curve β_1^{eq} is obtained for flow 1 through an iterative algorithm. Second, the lowest upper bound on delay is computed using Theorem 6.4.

As far as the first step is concerned, we begin with the rightmost node $N+M$. Applying Theorem 6.5, it is easy to obtain the equivalent service curve of nodes from $N+M$ to $N+1$, which simply relay the aggregate. By convoluting the service curve of each node for M times, the equivalent service curve $\beta_1^{eq_1}$ for the green nodes of the system model is obtained as

$$\beta_1^{eq_1} = \bigotimes_{i=N+1}^{N+M} \beta_i, \tag{32}$$

where $\bigotimes \beta_i$ denotes the min-plus convolution (see Eq. (23)) of function β_i.

To deal with the nodes from 1 to N, the procedures used to obtain another end-to-end equivalent service curve $\beta_1^{eq_2}$ can be formalized as the following five steps

(1) Set $j = N$, $\beta = \beta_N$.
(2) Calculate $\beta_j^{eq_2} = E(\beta, \lambda_j, \tau_j)$ by applying Theorem 6.6.
(3) Calculate $\beta = \beta_{j-1} \otimes \beta_j^{eq_2}$.
(4) Set $j = j - 1$.
(5) If $j = 1$, exit. Else go back to step (2).

Eventually, the equivalent end-to-end service curve β_1^{eq} for flow 1 is the convolution of $\beta_1^{eq_1}$ and $\beta_1^{eq_2}$

$$\beta_1^{eq} = \beta_1^{eq_1} \otimes \beta_1^{eq_2}. \tag{33}$$

Here, taking the properties of PRTR in WSN and simplifying computation into account, all the arrival curves can be assumed as

$$\lambda_i(t) = \lambda_{r,b}(t) = rt + b \qquad 1 \le i \le N, \tag{34}$$

where r denotes the sending rate of real-time traffic and b is the buffer size of nodes. Similarly, all the service curves are assumed as

$$\beta_j(t) = \beta_{R,\theta}(t) = R(t-\theta) \qquad 1 \le j \le N+M, \tag{35}$$

where R represents the rate of physical link and θ is the latency for scheduling packets. Besides, it is acceptable to assume that the sum of the sending rates of all the flows is no larger than the physical link rate, that is,

$$R \ge \sum_{i=1}^{N} r = Nr. \tag{36}$$

Based on the previous assumptions, β_1^{eq} is recalculated as

$$\beta_1^{eq} = [R - (N-1)r]t + r\sum_{i=2}^{N} \tau_i - (N-1)b - (N+M)R\theta. \tag{37}$$

Now that the equivalent end-to-end service curve β_1^{eq} for flow 1 is obtained, we move to the next step to compute the delay bound of flow 1 using Theorem 6.4 and Eq. (29), and hence derive the delay bound as

$$B = \frac{(N+M)R\theta + Nb}{R - (N-1)r}. \tag{38}$$

So far the end-to-end delay bound for flow 1 is obtained, and the delay bounds of flow i ($2 \le i \le N$) can also be derived in the same way. Notice that, as we only pay

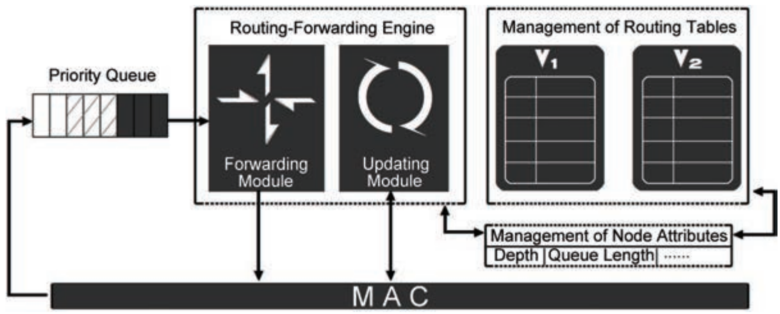

Fig. 6. The implementation of PRTR based on TinyOS. The arrows represent the data flows.

attention to the real-time flows in this methodology, B is the least performance upper bound (i.e., the minimum end-to-end delay) of PRTR. The bound is useful to evaluate routing protocols like PRTR which should guarantee a minimum end-to-end delay for real-time routing. In the next section, numerical examples of the delay bound combined with the simulation results demonstrate that the bound is tight.

7. PERFORMANCE EVALUATION

We conduct simulation experiments to evaluate the performance of PRTR on the TOSSIM [Levis et al. 2003] platform built in TinyOS. Comparative analysis is presented to investigate the impact of the weighing parameter α and the priority queue, omitting the influence of the subordinate parameters like buffer size and $\triangle Q_t$.

7.1. Implementation of Distributed Routing Algorithm

The implementation of PRTR based on TinyOS is depicted in Figure 6. It is mainly composed of four parts: the queuing module, the routing-forwarding engine, the management module of node attributes, and the management module of routing tables. The hardcore of PRTR lies in operating the routing tables, which interacts with the node attributes module frequently. The most important role of the attributes module is to decouple the routing engine and the routing tables. Tables V_1 and V_2 serve the corresponding composite fields $V_1(v)$ and $V_2(v)$, respectively. The updating packet is composed of 8-bit node depth and 8-bit queue length information. Once external events (such as the topology changing, the queue length variation exceeding $\triangle Q_t$) occur or the fixed timer (preventing redundancy updates and keeping the network connectivity) triggers, the correlative updating packets between neighbors are exchanged. To enhance the performance of PRTR, the assistant mechanism priority queue participates in queuing control through an open interface.

In Section 2, SPEED [He et al. 2003] was introduced as a typical real-time routing protocol in the WSN. We also reimplement its core mechanisms that take charge of real-time routing on the TOSSIM simulator, including the delay estimation scheme, the Neighborhood Feedback Loop (NFL), the Stateless Nondeterministic Geographic Forwarding (SNGF) algorithm, and the beacon exchange scheme.

7.1.1. Implementation of MAC. Since we focus on the routing layer in this work, we design and implement an appropriate MAC protocol on top of TOSSIM. The so-called *appropriate* MAC is modified based on TDMA, which is noncompetitive and all the unpredictable factors that may affect the delay are particularly removed. Furthermore, the MAC protocol is improved to ensure that the packets will not be dropped in the MAC layer. Therefore, all packet losses observed in PRTR are only due to buffer overflow. In

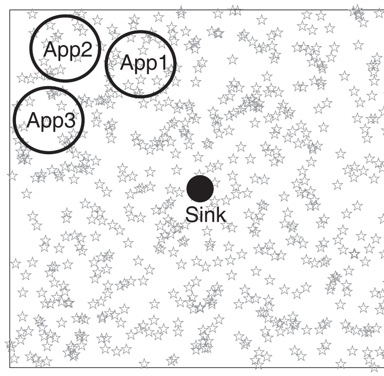

Fig. 7. A randomly deployed sensor network. The pentagrams represent wireless sensor nodes and the black round denotes the sink.

other words, the enhanced MAC that we use will not influence either the end-to-end delay or the network throughput.

Since the suggested TDMA-like MAC is noncompetitive, within each transmission slot, the transmission opportunity is assigned to only one sender. The sender will broadcast its packets to all the neighbors that can hear it. However, only the receiver that is in accordance with the destination of each packet will accept and handle this packet. All the other receivers will drop that packet. In PRTR, if some node intends to change its next-hop neighbor, it only needs to change the destination of each packet. Thus, no MAC problem will be generated when diverting packets toward new paths.

Besides, in Hull et al. [2004] the congestions in wireless sensor networks are classified into two categories. The first type is node-level congestion caused by buffer overflow in the node and can result in packet loss, and increased delay. The second type is link-level congestion that is related to the wireless channels shared by several nodes using the competitive MAC protocol, such as CSMA/CA. In the link-level congestion, collisions could occur when multiple active sensor nodes attempt to seize the channel at the same time. However, in this work, the implemented appropriate MAC is noncompetitive and no packets will be dropped due to collision that results in the link-level congestion. Once the link-level congestion is totally avoided, we can concentrate on alleviating the node-level (due to buffer overflow) congestion in PRTR.

7.2. Simulation Setup and Performance Metrics

Figure 7 illustrates a randomly deployed network with 600 nodes densely distributed over a 100m$\times$100m square area to form a flat multihop network. In this WSN, only one sink resides in the center with all data packets forwarded toward it. Therefore, the data flows have obvious directional characteristic in the WSN. We are only concerned about the transmission quality for single-direction link and omit the feedback link when PRTR is designed. Although the experimentations conducted for PRTR require symmetric links, it is still reasonable with asymmetric links which are common in practice. The detailed configurations of simulation parameters are summarized in Table I.

Three independent applications are deployed on the nodes residing in the monitoring area (identified as events with black circles in Figure 7), one of which is a real-time application with a relatively higher sampling rate while the others generate nondelay-sensitive packets with lower sampling rates. More details on these applications can be

Table I. Simulation Configurations

Deployment	Area Size	100×100
	Deployment Type	Randomly
	Network Architecture	Homogeneous, Flat
	Number of Nodes	600
	Largest Depth	26
	Average Node Degree	8
	Sink	(50, 50)
	Radio Range	5.5
	Link Layer Transmission Rate	8Kbps (40 packets/s)
Task	Application Type	Event-driven
	Packet Size	25 Bytes
PRTR	Buffer Size	31 packets
	α	0.9 and 0.5
Simulation	Time	400 seconds

Table II. Applications in simulation experiments

	Flag	Type	Active Time	Rate
App1	1	real-time	$100s \sim 140s$	$40 packets/s$
App2	0	non real-time	$100s \sim 160s$	$30 packets/s$
App3	0	non real-time	$100s \sim 160s$	$25 packets/s$

found in Table II. Specifically, the *Active Time* in this table indicates the on/off time of each application.

Three important performance metrics are defined as follows.

(1) End-to-End Delay (EED). The EED is defined as the difference between the time when a packet is injected to the WSN and the time it is received by the sink.
(2) Throughput Ratio (TR).

$$TR = \frac{\text{number of packets received by the Sink}}{\text{number of packets sent by source nodes}}. \tag{39}$$

(3) Energy Consumption per Received Packet (ECRP). The ECRP is defined by the sum of all the energy consumption for sending (3 units energy) and receiving (2 units energy) of one packet that is received by the sink.

7.3. Comparative Analysis

We choose TADR [He et al. 2008] and SPEED [He et al. 2003] as the performance benchmarks for throughput and delay, respectively. To provide comparative analysis, five experiments using different routing protocols but based on the same event configurations (as in Table II) are carried out separately, including PRTR ($\alpha = 0.5$), PRTR ($\alpha = 0.9$), PRTR ($\alpha = 0.9$ without priority queue), TADR, and SPEED. For simplifying notations, we denote $\alpha = 0.9$ without priority queue as $\alpha^* = 0.9$, unless specified otherwise.

7.3.1. Real-Time Routing. As the primary goal of PRTR is to provide real-time routing, the simulation results of real-time application 1 (App1) are shown in Figure 8. It illustrates the average EEDs of arrival packets using different routing protocols in a 10s period (e.g., the first bar group represents 100∼110s). Since the delays of the variant PRTR ($\alpha = 0.5$) and PRTR ($\alpha = 0.9$) are small enough, all the packets routed by them get to the sink before 160s, leaving the data of the last three bar groups empty (see 160s∼190s in Figure 8). As a real-time routing protocol, PRTRs with both $\alpha = 0.5$ and $\alpha = 0.9$ minimize the EED of packets, whereas PRTR without priority queue and

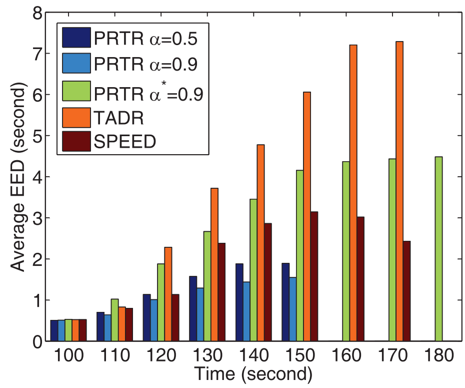

Fig. 8. The average EED of packets from App1 within a 10s period of simulation.

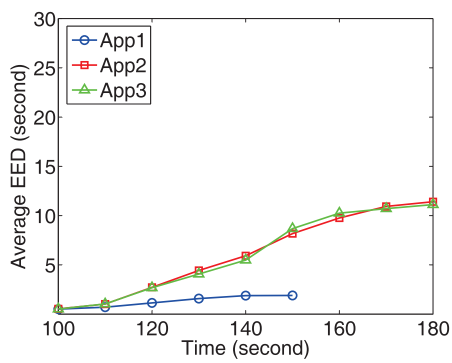

Fig. 9. The average EED of different applications in PRTR when $\alpha = 0.5$.

TADR increase the delay significantly (nearly two times of those using the priority queue after 130s). Particularly, the first two variants of PRTR achieve less delay than SPEED.

7.3.2. Impact of α and Priority Queue. To further explore the impact of the weighing parameter α and the priority queue, three similar experiments are performed using different PRTRs. The average EEDs of packets from all three applications are shown in Figures 9, 10, and 11, respectively. It is indicated that PRTR with lager α ($\alpha = 0.9$ in Figure 10 relative to $\alpha = 0.5$ in Figure 9) will increase the EED of nondelay-sensitive packets, which validates Corollary 5.3 since these packets will be more easily driven out of the shortest paths to be cached. Note that smaller α will weaken the capability of real-time transmission of PRTR as shown in Figure 8 and Table III. Thus, there is a trade-off between real-time transmission and caching capability when we adjust the value of α. Moreover, the assistant mechanism priority queue significantly impacts on the routing of the real-time packets (i.e., PRTR with $\alpha^* = 0.9$ causes more delay to App1 as shown in Figure 11).

7.3.3. Throughput Ratio and Energy Consumption. Another two metrics should be mentioned since they are always involved in the goals of a routing protocol in the WSN.

Tables III and IV show the statistical results of the Throughput Ratio (TR) of the network and average End-to-End Delay (EED) of all the packets. These statistics indicate that, according to the classification of packets, PRTR dedicates itself to minimizing the

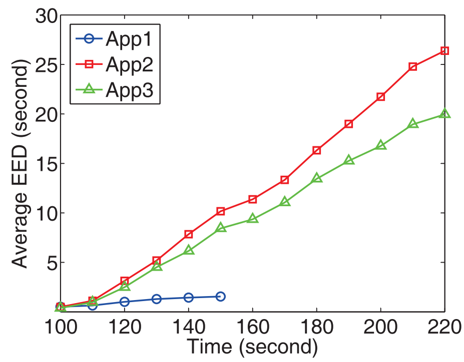

Fig. 10. The average EED of different applications in PRTR when $\alpha = 0.9$.

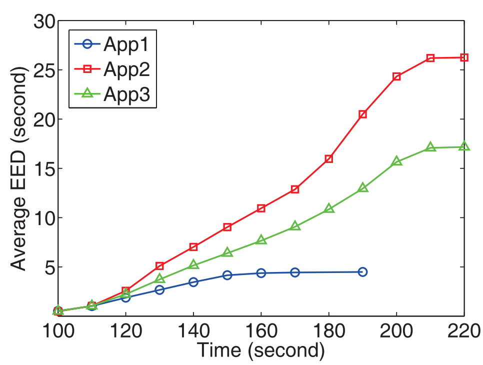

Fig. 11. The average EED of different applications in PRTR when $\alpha^* = 0.9$.

Table III. Throughput Ratio (TR) and Average End-to-End Delay (EED)

	$\alpha = 0.5$		$\alpha = 0.9$		$\alpha^* = 0.9$	
	TR(%)	EED(s)	TR(%)	EED(s)	TR(%)	EED(s)
App1	86.8	1.89	98.7	1.55	96.8	4.48
App2	97.0	11.41	98.3	26.38	98.7	26.25
App3	96.2	11.12	98.8	20.09	99.3	17.17

EED of real-time App1, leaving the EEDs of App2 and App3 unassured. PRTR with larger α tends to scatter nondelay-sensitive packets with larger EEDs. Without the priority queue, the EEDs of all applications increase significantly, emphasizing the importance of this mechanism. Note that PRTR still maintains high TR (compared with TADR) of all applications due to the congestion avoidance. In other words, although nonreal-time packets may be cached by neighbors at the same level (Proposition 5.2) or at the higher level (Proposition 5.4), they will be forwarded toward the sink at last. Since SPEED does not distinguish applications, it serves each one fairly with larger EED of App1 than PRTR. Besides, the throttling or rerouting that SPEED uses to handle congestion are less effective than PRTR.

The ECRP and the throughput of the whole network within a period of simulation are presented in Figures 12 and 13, respectively. Actually, PRTR with $\alpha = 0.5$ performs power control as well as TADR and SPEED (in Figure 12). Compared with TADR, our proposed algorithm PRTR achieves similar congestion avoidance and throughput improvement (100s$\sim$150s as shown in Figure 13). Specifically, since PRTRs with larger

Table IV. Throughput Ratio (TR) and Average End-to-End Delay (EED)

	$\alpha = 0.9$		TADR		SPEED	
	TR(%)	EED(s)	TR(%)	EED(s)	TR(%)	EED(s)
App1	98.7	1.55	92.1	6.19	83.7	2.91
App2	98.3	26.38	93.7	8.43	86.5	3.35
App3	98.8	20.09	92.5	8.08	85.4	3.34

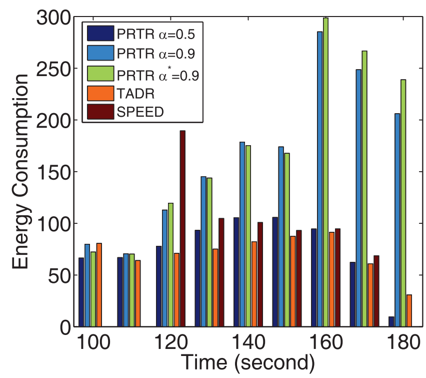

Fig. 12. The average energy consumed by every arrival packet within a 10s simulation.

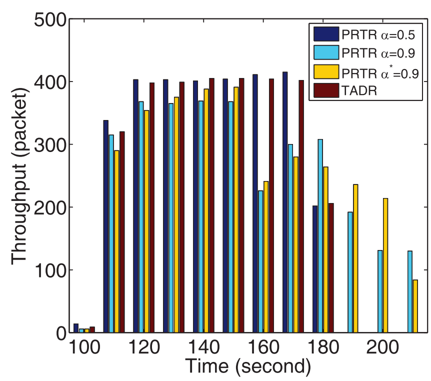

Fig. 13. The throughput of the whole network within a 10s simulation.

α (i.e., $\alpha = 0.9$) will induce more caching operations and thereby lager delays for App2 and App3 (see Figures 10 and 11), most packets from App2 and App3 reach the sink after 160s. Consequently, PRTRs with $\alpha = 0.9$ are highlighted in the last three bars of both Figures 12 and 13.

7.4. Numerical Examples

Let us now give numerical examples of the delay bound as obtained from Eq. (38). Following the general configurations in related works, M is set to 5, and the physical data rate R is practically set to 8Kbps (namely 40 packets/s) with node latency θ set to 0.1s. Moreover, the node buffer b is up to 31 packets with 25 bytes in each packet, while the flow rate is constrained to 4 packets per second. Six more simulations are conducted using PRTR ($\alpha = 0.9$), and we calculate the average EED of App1 in each simulation. The results and the function curve of B in Eq. (38) are jointly presented

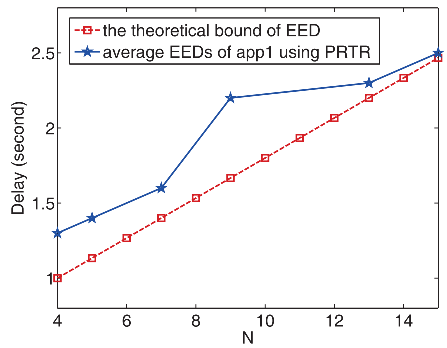

Fig. 14. The theoretical bound of EED ($M = 5$) and the average EED of App1 using PRTR in simulations.

in Figure 14. It can be illustrated that B is tight enough as the least upper bound on delay of PRTR.

8. CONCLUSION

This article focuses primarily on how to support real-time routing in wireless sensor networks. We propose a novel algorithm called Potential-based Real-Time Routing (PRTR), which is based on the concept of potential field to spontaneously make basic routing decisions. The composite potential field is defined as a convex combination of the node depth field and queue length field, with the help of a weighing parameter α. Taking the maximum force rule into account, α is given two specific values and hence two specified fields are established to provide different services, respectively. Specifically, $V_1(v)$ for $\alpha = 0$ degenerates PRTR as a shortest path routing algorithm, which exactly provides delay-minimized real-time routing. The other field $V_2(v)$ for $0 < \alpha < 1$ is used to alleviate congestion and help nondelay-sensitive flows bypass the hotspots, which scatters these excessive packets for multipath transmission.

Since the delay bound is highly important, in the theoretical analysis we utilize the Network Calculus theory to calculate the end-to-end delay bound for a single flow routed by PRTR. The simulation results show that the PRTR algorithm provides excellent real-time routing performance for delay-sensitive applications and improves the throughput of the whole network at the same time. In the future work, some testbeds should be set up to carry out experiments instead of simulations, and the optimal value of weighing parameter α may be derived theoretically.

To summarize, the important features of PRTR are as follows.

(1) PRTR only finds paths toward the sink and does not serve for data dissemination or point-to-point communication.
(2) The most attractive point of the virtual potential field is that, since there is only one destination (i.e., the sink) in the WSN, technically only one potential field needs to be built. Although we establish two different fields for two kinds of applications, it is unnecessary to maintain them across the WSN all the time, because update packets only exchange between neighbors under certain conditions. Therefore, only a few simple maintenance operations are needed with little extra overhead.
(3) In the distributed fashion, PRTR provides good scalability and adaptability to large-scale dynamic sensor networks simply by local information. Attributes like the node depth and queue length are fairly easy to obtain and thus the implementation is simplified.
(4) PRTR successfully satisfies the requirements of real-time routing and simultaneously avoids possible congestions that cause serious packet loss.

ACKNOWLEDGMENTS

We sincerely acknowledge the associate editor and the anonymous referees for constructive suggestions that helped us improve the quality of the manuscript significantly. We also thank Giacomo Ghidini for carefully reading an earlier draft of the manuscript.

REFERENCES

AHMED, A. A. AND FISAL, N. 2008. A real-time routing protocol with load distribution in wireless sensor networks. *Comput. Comm. 31*, 14, 3190–3203.

BASU, A., LIN, A., AND RAMANATHAN, S. 2003. Routing using potentials: A dynamic traffic-aware routing algorithm. In *Proceedings of the Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications (SIGCOMM'03)*. 37–48.

BOUDEC, J.-Y. L. AND THIRAN, P. 2001. *Network Calculus. A Theory of Deterministic Queuing Systems for the Internet*. Springer.

CHIPARA, O., HE, Z., XING, G., CHEN, Q., WANG, X., LU, C., STANKOVIC, J., AND ABDELZAHER, T. 2005. Real-time power control in wireless sensor networks. Tech. rep. WUCSE-2005-31, Washington University in St. Louis.

CHIPARA, O., HE, Z., XING, G., CHEN, Q., WANG, X., LU, C., STANKOVIC, J., AND ABDELZAHER, T. 2006. Real-time power-aware routing in sensor networks. In *Proceedings of 14th IEEE International Workshop on Quality of Service (IWQoS'06)*. 83–92.

COUTO, D. S. J. D., AGUAYO, D., BICKET, J. C., AND MORRIS, R. 2003. A high-throughput path metric for multi-hop wireless routing. In *Proceedings of the 9th Annual International Conference on Mobile Computing and Networking*. 134–146.

FELEMBAN, E., LEE, C.-G., EKICI, E., BODER, R., AND VURAL, S. 2005. Probabilistic qos guarantee in reliability and timeliness domains in wireless sensor networks. In *Proceedings of 24th Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM'05)*. Vol. 4. 2646–2657.

HE, T., REN, F., LIN, C., AND DAS, S. K. 2008. Alleviating congestion using traffic-aware dynamic routing in wireless sensor networks. In *Proceedings of the 5th Annual IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks (SECON'08)*. 233–241.

HE, T., STANKOVIC, J., LU, C., AND ABDELZAHER, T. 2003. SPEED: A stateless protocol for realtime communication in sensor networks. In *Proceedings of the 23rd International Conference on Distributed Computing Systems (ICDCS'03)*. 46–55.

HE, T., VICAIRE, P., YAN, T., LUO, L., GU, L., ZHOU, G., STOLERU, R., CAO, Q., STANKOVIC, J. A., AND ABDELZAHER, T. 2006. Achieving real-time target tracking using wireless sensor networks. In *Proceedings of the 12th IEEE Real-Time and Embedded Technology and Applications Symposium (RTAS'06)*. 37–48.

HUANG, S.-C. AND JAN. R.-H. 2004. Energy-aware, load balanced routing schemes for sensor networks. In *Proceedings of 10th International Conference on Parallel and Distributed Systems (ICPADS'04)*. 419–425.

HULL, B., JAMIESON, K., AND BALAKRISHNAN, H. 2004. Mitigating congestion in wireless sensor networks. In *Proceedings of the 2nd International Conference on Embedded Networked Sensor Systems (SenSys'04)*. 134–147.

LENZINI, L., MARTORINI, L., MINGOZZI, E., AND STEA, G. 2004. A methodology for deriving per-flow end-to-end delay bounds in sink-tree diffServ domains with fifo multiplexing. In *Proceedings of the 19th International Symposium on Computer and Information Sciences (ISCIS'04)*. Lecture Notes in Computer Science, vol. 3280, Springer, 604–614.

LENZINI, L., MARTORINI, L., MINGOZZI, E., AND STEA, G. 2006. Tight end-to-end per-flow delay bounds in fifo multiplexing sink-tree networks. *Perform. Eval. 63*, 9, 956–987.

LEVIS, P., LEE, N., WELSH, M., AND CULLER, D. 2003. TOSSIM: Accurate and scalable simulation of entire tinyos applications. In *Proceedings of the 1st International Conference on Embedded Networked Sensor Systems (SenSys'03)*. 126–137.

LU, C., BLUM, B. M., ABDELZAHER, T. F., STANKOVIC, J. A., AND HE, T. 2002. Rap: A real-time communication architecture for large-scale wireless sensor networks. In *Proceedings of the 8th IEEE Real-Time and Embedded Technology and Applications Symposium*. 55–66.

LU, C., XING, G., CHIPARA, O., FOK, C.-L., AND BHATTACHARYA, S. 2005. A spatiotemporal query service for mobile users in sensor networks. In *Proceedings of the 25th IEEE International Conference on Distributed Computing Systems (ICDCS'05)*. 381–390.

NA, J., SOROKER, D., AND KIM, C.-K. 2007. Greedy geographic routing using dynamic potential field for wireless ad hoc networks. *IEEE Comm. Lett. 11*, 3, 243–245.

POOR, R. D. 2000. Gradient routing in ad hoc networks. www.media.mit.edu/pia/Research/ESP/texts/poorieeepaper.pdf.

Pothuri, P. K., Sarangan, V., and Thomas, J. P. 2006. Delay-constrained, energy-efficient routing in wireless sensor networks through topology control. In *Proceedings of the IEEE International Conference on Networking, Sensing, and Control (ICNSC'06)*. 35–41.

Ren, F., He, T., Das, S. K., and Lin, C. 2011. Traffic-aware dynamic routing to alleviate congestion in wireless sensor networks. *IEEE Trans. Parallel Distrib. Syst. 22*, 1585–1599.

Zhao, J. and Govindan, R. 2003. Understanding packet delivery performance in dense wireless sensor networks. In *Proceedings of the 1st International Conference on Embedded Networked Sensor Systems (SenSys'03)*. 1–13.

Received February 2011; revised October 2011; accepted April 2012