Traffic-Aware Dynamic Routing to Alleviate Congestion in Wireless Sensor Networks

Fengyuan Ren, *Member*, *IEEE*, Tao He,
Sajal K. Das, *Senior Member*, *IEEE*, and Chuang Lin, *Senior Member*, *IEEE*

Abstract—The congestion problem in Wireless Sensor Networks (WSNs) is quite different from that in traditional networks. Most current congestion control algorithms try to alleviate the congestion by reducing the rate at which the source nodes inject packets into the network. However, this traffic control scheme always decreases the throughput so as to violate fidelity level required by the applications. In this paper, we present a solution that sufficiently exerts the idle or underloaded nodes to alleviate congestion and improve the overall throughput in WSNs. To achieve this goal, a traffic-aware dynamic routing (TADR) algorithm is proposed to route packets around the congestion areas and scatter the excessive packets along multiple paths consisting of idle and underloaded nodes. Utilizing the concept of potential in classical physics, our TADR algorithm is designed through constructing a hybrid virtual potential field using depth and normalized queue length to force the packets to steer clear of obstacles created by congestion and eventually move toward the sink. The simulation results show that the proposed solution improves the overall throughput by around 370 percent as compared to MintRoute, which is one of benchmark routing protocols. Furthermore, TADR scheme has low overhead suitable for large-scale, dense sensor networks.

Index Terms—Wireless sensor networks, traffic-aware, routing, potential field, gradient.

1 Introduction

Congestion in Wireless Sensor Networks (WSNs) has negative impact on the performance, namely, decreased throughput and increased per-packet energy consumption [21]. The congestion problem in WSNs is quite different from traditional networks. As the main task of WSNs is to gather information from the physical world, all the data flows go toward a common sink, while in the traditional network, they are always irregular because the destinations are random. Due to the centralized traffic pattern in WSNs, just bypassing the hot spots is ineffective to eliminate the congestion because it will reappear near the sink. The challenge in congestion control in WSNs also stems from some additional requirements. For example, the data generated during a crisis state are of utmost importance, and loss of such data can violate the purpose of deploying an unattended sensor network. In other words, the congestion control in WSNs must not only be based on the network capacity, but also on the fidelity required by the applications.

Most of the prior works on congestion control in WSNs have only focused on the traffic control (including end-to-end and hop-by-hop). In other words, they basically try to throttle the incoming traffic into the network once congestion is detected. Although traffic control strategies are effective to alleviate congestion in traditional networks, and are also suggested in some wireless sensor network scenarios [21], [5], [19], they are restricted or even unsuitable for special purposes for the following two reasons: first, reducing source traffic during a crisis state is undesirable since it will significantly violate fidelity requirements. It may be a better option to increase capacity by turning on more resources to accommodate excessive incoming traffic during the crisis state. Fortunately, the wireless sensor network can provide elastic resource availability because of its dense deployment, unlike its wired or other wireless counterparts. This distinct advantage enables WSNs to employ adaptive capacity planning schemes to avoid possible congestion and satisfy fidelity requirements at the same time. Second, it is highly likely that the congestion caused by burst traffic is often transient by nature. For example, the sensor nodes will generate transient bursts of traffic when the abnormal events occur. It could be inefficient to cope with these transient congestions by the traffic control based on feedback because they may be alleviated through quickly adjusting the network resource provisioning.

There are various common congestion control schemes, such as capacity planning, end-to-end or hop-by-hop traffic control, connection admission control, and buffering. It has been pointed out in [6] that the selection of congestion control schemes should depend upon the characteristic of congestion.

In this paper, we propose a traffic-aware dynamic routing (TADR) algorithm to route packets around the congestion areas and scatter the excessive packets along multiple paths where the idle or underloaded nodes are sufficiently utilized in response to congestion under the fidelity requirements. The cornerstone of the TADR algorithm is to construct two independent potential fields using depth[1] and queue length, respectively. Normally, the depth field would find the shortest paths for packets. Once the

- *F. Ren, T. He, and C. Lin are with the Department of Computer Science and Technology, Tsinghua University, Beijing 100084, P.R. China. E-mail: {renfy, chlin}@tsinghua.edu.cn, hetao@csnet1.cs.tsinghua.edu.cn.*
- *S.K. Das is with the Department of Computer Science and Engineering, University of Texas at Arlington, PO Box 19015, Arlington, TX 76019. E-mail: das@uta.edu.*

Manuscript received 24 Nov. 2009; revised 22 May 2010; accepted 23 June 2010; published online 5 Jan. 2011.
Recommended for acceptance by S. Papavassiliou.
For information on obtaining reprints of this article, please send e-mail to: tpds@computer.org, and reference IEEECS Log Number TPDS-2009-11-0579.
Digital Object Identifier no. 10.1109/TPDS.2011.24.

1. In this work, depth is defined as the least number of hops that a node is away from the sink.

queue length grows over a certain threshold, which always means congestion, the packets would flow along other suboptimal paths or just be cached in areas with more buffers. Thus, the queue length potential field endows our traffic-aware solution, and the depth field provides the basic routing backbone to direct the packets to the sink. These two fields will be combined into a hybrid potential field to dynamically make routing decisions.

The rest of this paper is organized as follows: in Section 2, we introduce the related work about congestion control in WSNs and the motivation that drives us to design a potential-based traffic-aware dynamic routing algorithm. Section 3 introduces the basic ideas underlying our solution and builds up the potential fields used by our TADR. Section 4 presents some properties and other considerations about our algorithm. In Section 5, we describe the details of the implementation. This is followed by Section 6 where the performance of TADR is evaluated through simulation experiments on an arbitrary deployed network, and the impact of parameters on the performance is also examined. We conclude this paper in Section 7.

2 Related Works and Motivation

Congestion control is an important issue in WSNs. In [3], the congestion in WSNs is classified into two categories. The first one is node-level congestion caused by buffer overflow in the node and can result in packet loss, and queuing delay increasing. Another is link-level congestion that is related to the wireless channels shared by several nodes using the competitive MAC protocol, such as Carrier Sense and Multiple Access with Collision Avoidance (CSMA/CA). In this case, collisions could occur when multiple active sensor nodes try to seize the channel at the same time. Most of investigations mainly cope with node-level congestion and leave link-level congestion to proper MAC protocols.

The traffic control is widely employed in studies on congestion control in WSNs. CODA [21] presents the first detailed investigation on congestion detection and avoidance in WSNs, where congestion is detected by sampling the wireless medium and by monitoring the queue occupancy. As soon as a node detects congestion, it broadcasts a backpressure message upstream, and then the upstream nodes will throttle the traffic volume to alleviate congestion. In addition, CODA also employs the closed-loop source regulation, where long-term end-to-end constant feedback from the sink to the source nodes is required to adjust the sending rate through employing additive increase and multiplicative decrease (AIMD) scheme. FUSION [5] introduces three congestion control techniques: hop-by-hop flow control, limiting source rate, and a prioritized medium access control. Basically, it mitigates congestion by throttling the transmissions of the upstream nodes and the source nodes. However, in its rate limiting mechanism, nodes need to continuously watch their parents' sending actions to determine when to generate tokens. This continuous monitoring is too costly and consumes more energy. ESRT [19] is an event-to-sink reliable transport protocol, which can serve as a congestion control protocol. In ESRT, the sink is required to periodically configure the source sending rate to avoid congestion. Upon detecting congestion, all the data flows are throttled to a lower rate. Similarly, Interference-Aware Fair Rate Control (IFRC) [18] employs static queue thresholds to determine congestion level and exercises congestion control by adjusting outgoing rate on each link based on AIMD scheme. Its feature is to use a tree rooted at each sink to route all data. When congestion occurs, the rates of the flows on the interfering trees are throttled. In [24], a priority-based rate control mechanism is proposed to adjust the source traffic rates based on current congestion in the upstream nodes and the priority of each traffic source for congestion control and service differentiation in wireless multimedia sensor networks. Zawodniok and Jagannathan designed a decentralized predictive congestion control (DPCC) [25] scheme which detects the onset of congestion using queue utilization and the embedded channel estimator for predicting the channel quality. In DPCC, the adaptive flow control algorithm selects suitable rate enforced by the adaptive back-off interval selection scheme. RCRT [12] is a reliable transport protocol for wireless sensor networks. It uses end-to-end explicit loss recovery and places its congestion control functionality at the sink, whose perspective into the network enables better aggregate control of traffic, and affords flexibility in rate allocation. In [2], Chen and Yang propose a congestion avoidance scheme based on lightweight buffer management, which follows the basic idea of the credit-based hop-by-hop flow control in ATM networks, and employs a $1/k$ buffer scheme to prevent hidden terminals from causing congestion. Although traffic control can effectively alleviate congestion, it could impose negative impact on fidelity. Thus, the special nature of sensor networks calls for a new approach to alleviate congestion that can satisfy the application fidelity requirements.

Except for the schemes based on traffic control, there have been some attempts to explore other mechanisms for congestion avoidance in WSNs. SPEED [4] 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, leading to congestion. Siphon [22] introduces some virtual sinks (VS) with a longer range multiradio within the sensor network. When the versus finds the redirection bit enabled, it routes the packets using its own long range communication network toward the physical sink, bypassing the underlying sensor network routing protocols to avoid potential congestion. In [3], Ee and Bajcsy proposed a distributed congestion control scheme based on hop-by-hop automatic repeat request in many-to-one routing scenarios where the Congestion Control and Fairness (CCF) routing scheme uses packet service time at the node as an indicator of congestion. However, the service time alone may be misleading when the incoming rate is equal to or lower than the outgoing rate. The limitation of CCF is that it requires the network topology to be static or near static. Wang et al. [23] propose a hop-by-hop node priority-based upstream congestion control protocol (PCCP), which refutes providing equal fairness (e.g., CCF) to each sensor node in a multihop WSN by attaching a weighted fairness to each sensor node. PCCP infers the degree of congestion through packet interarrival time and packet service time and then imposes hop-by-hop congestion control depending on the measured congestion degree and the priority index. Congestion-Aware Routing (CAR) [10] enforces a differentiated routing approach to discover the congested zone of the network that exists between high-priority data sources and the data sink, and to dedicate this portion of the network to forward primarily high-priority traffic. Kang et al. [9] suggest increasing the network resource (referred to as "resource control") to alleviate congestion and improve the throughput. It incipiently checked the influence of multiple paths on the end-to-end channel capacity and provided some guidelines to design the "resource control" algorithms. Two practical "resource control" schemes are indeed proposed in [7] and [8]. One is topology-aware resource adaptation (TARA) strategy [7], which activates appropriate sensor nodes whose radio is off to form a new topology that has just enough capacity to handle the increased traffic. To efficiently estimate the capacity using a graphic-theoretic approach, TARA requires not only local knowledge, but also knowledge about the end-to-end topology. This overhead is too high to allow the network to scale up to a large number of nodes. Another scheme proposed in [8] just finds multiple paths to alleviate the congestion by bypassing the hotspots, ignoring the characteristics of the centralized traffic pattern of WSNs. In fact, alleviating congestion with multipath routing has been proposed by several other works. In [16], Pham and Perreau proposed splitting the traffic from the source into multiple paths to achieve load balance and increase throughput. An Interference-Minimized Multipath Routing (I2MR) protocol is developed to increase throughput by discovering zone-disjoint paths for load balancing, requiring minimal localization support [20]. To avoid the additional costs of multipath routing when the network is not congested, Popa et al. [17] proposed Biased Geographical Routing (BGR) protocol to reactively split traffic when congestion is detected. The "bias" determines how far the trajectory of splitting traffic will deviate from a greedy route (which is always the shortest path). Because the bias is randomly chosen, BGR likely makes the congestion worse under some situations, as explained in the next section. In addition, BGR needs location information of nodes provided by either GPS or other coordinate system, thus incurring in additional overhead.

Although BGR has the above-mentioned drawbacks, the basic idea is interesting and suitable for congestion control in WSNs because it essentially employs the dynamic routing technique to alleviate congestion so as to decrease the additional cost of static multipath routing. If its drawbacks, such as the blindness during scattering packets and the restrictions on coordinate systems, can be properly overcome, we may obtain a more effective and practicable congestion control mechanism for WSNs. Based on this understanding, in this paper, we will follow the dynamic capacity planning philosophy to alleviate congestion and meet the fidelity requirement through designing a traffic-aware dynamic routing protocol. Fig. 1 intuitively illustrates our motivations (only one small part of wireless sensor network is presented). If nodes A and B send or relay packets on their shortest paths, respectively, node 1 will easily be congested as shown in Fig. 1a. If node B blindly scatters its excessive packets on a random detour path to alleviate the congestion on its shortest path as in the BGR algorithm, for example, forwarding packets to node 2, congestion may arise or become worsen on other nodes as shown in Fig. 1b. If the forwarding node takes the load status on its neighbors into account to scatter the excessive packets, an appropriate detour path consisting of idle or underloaded nodes can be found purposely, but not blindly, such as the detour path $A \rightarrow 4 \rightarrow 5 \rightarrow 6 \rightarrow Sink$ for the packets from node A illustrated in Fig. 1c. Under this traffic-aware dynamic routing paradigm, traffic will be spatially spread in more even patterns and the more resources can be sufficiently utilized so as to reduce occurrence of congestion while improving overall throughput. Obviously, the packets on the detour path will experience a relatively large end-to-end delay. Undoubtedly, it is a negative impact in traditional networks; however, this temporal spreading will benefit avoiding congestions appearing readily around the sink because of traffic centrality. Anyway, the objective of our traffic-aware dynamic routing is to alleviate congestion and improve throughput by distributing packets in both time and space. Considering the features of large-scale dense sensor networks and the practicability of protocol, our routing algorithm needs to keep low computational overhead to allow timely execution on slow processors, maintain the minimal amount of state information, and dispense with the global knowledge and extra overhead, such as node position information. To achieve this goal, we will borrow the concept of potential in classical physics to design a traffic-aware dynamic routing scheme for WSNs.

3 Basic Routing Algorithm

3.1 Description of Proposed TADR Scheme

Basu et al. [1] utilize the steepest gradient search method to propose a potential-based routing paradigm in the context of traditional network. However, it does not attract widespread attention because of its huge management overhead. It is indeed expensive to build an exclusive virtual field for each destination in traditional networks where numerous destinations may be distributed arbitrarily. On the contrary, the centralized traffic pattern in WSNs with a single (at most several) sink(s) is very beneficial for drastic decrease of management cost to implement a practical potential-based routing algorithm. Therefore, our TADR scheme is also designed based on the concept of potential field.

Before starting to design the concrete algorithm, we first show how it works. Intuitively, the potential field used by the TADR packets can be viewed as a "bowl" illustrated in Fig. 2. The sink resides at the bottom, and all data packets flow down along the surface just like water. In lightly loaded networks, the surface of the "bowl" is smooth, and hence our algorithm acts just like the shortest path routing. But in heavily loaded cases (e.g., burst of data packets caused by detection of a monitoring event), the congestion will form some bulges on the bowl surface, which will block the packets to flow directly down to the bottom along the shortest path, namely, bypass the congestion areas. The

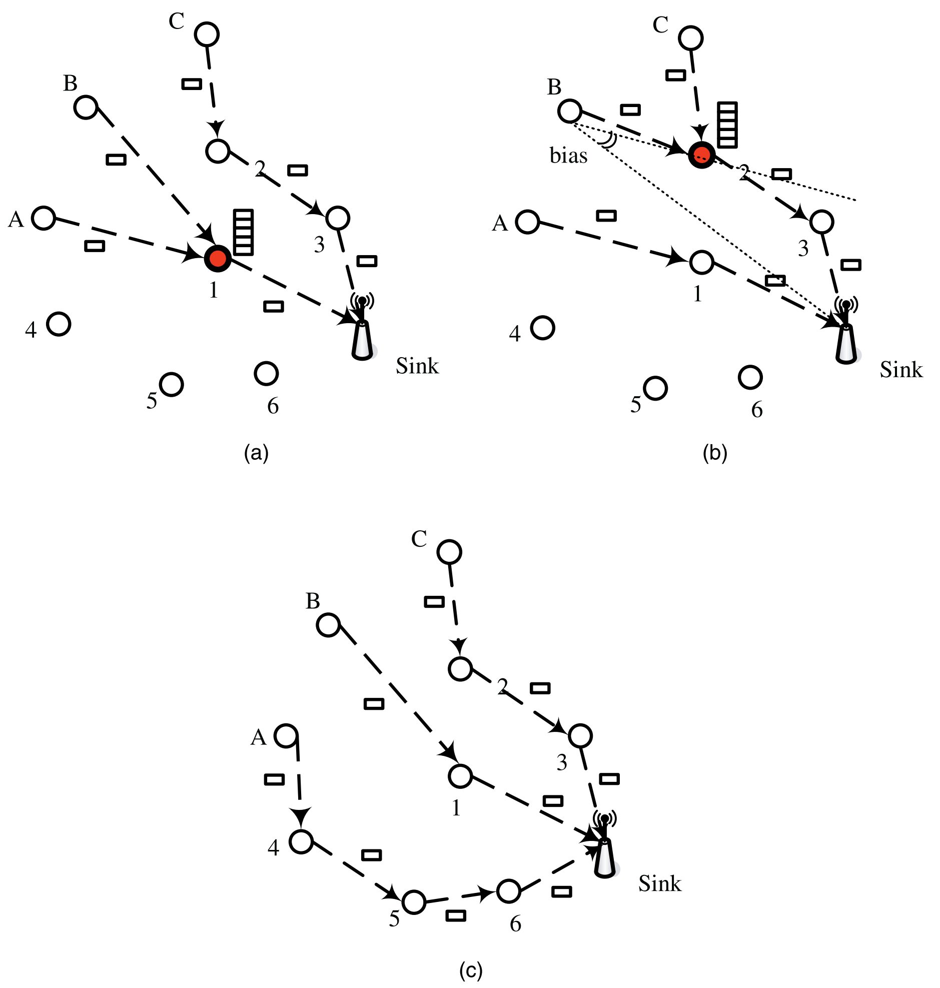


Fig. 1. Motivation example. Circles represent nodes, rectangles beside the circles represent the queue at nodes, and dashed lines denote the links in routing paths.

excessive packets will be driven by the potential field to find the appropriate detour path without obstacles, i.e., idle or underloaded nodes. When the congestion disappears, the bowl surface becomes smooth, and the packets continue to move along the shortest path. Essentially, through spreading the packet transmissions spatially and temporarily, our TADR scheme can alleviate congestion, while improving the throughput at the same time.

In the next section, we will describe how to construct the routing potential fields using depth and queue length, respectively, and then how to integrate them together to make dynamic routing decision.

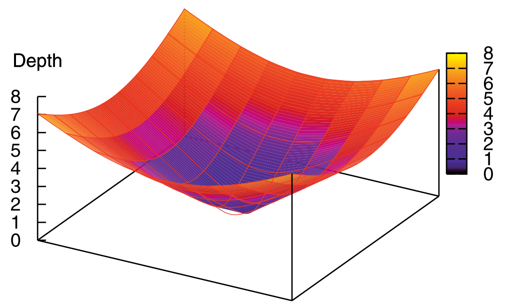


Fig. 2. The smooth "bowl" of depth potential field.

3.2 Design of Potential Fields

3.2.1 *Potential Field Model*

In the "bowl" model shown in Fig. 2, we can view the whole network as a gravitational field. A packet can also be viewed as a drop of water, moving down to the bottom along the surface of the bowl. The trajectory of this packet is determined by the "force" from the gravitational potential field. Otherwise, there is an imaginary "hole" which is analogous with the sink node at the bottom.

We assign a single-valued potential, $V(v)$, to every node v on the bowl surface to form a scalar potential field.

Let us consider a packet p at node v. To reach the sink, p must be forwarded to one of the neighbors of v (in the rest of this paper, we refer the neighbor set of node v as $nbr(v)$). To determine this "next hop" neighbor, we define a "force" acting on the packet p at node v based on the potential difference between node v and each one of its neighbors. Thus, for a neighbor $w \in nbr(v)$, we define this "force" as

$$F(v,w) = \frac{V(v) - V(w)}{c_{vw}}. \quad (1)$$

Here, c_{vw} is the *cost* of the radio link from node v to w, and will be described in detail later in this section.

The packet p is forwarded to the neighbor for which the "force" $F(v,x)$ is maximum, namely, the neighbor in the direction of the steepest gradient. If the surface is smooth, this water drop will fall straight to the bottom; but if it is somewhat rough, the water drop will move along an irregular curve which is formed by a series of "valleys."

Our TADR scheme constructs this smooth "bowl" using the depth. At the same time, the queue length on the nodes will bring some bulges on its surface to make the algorithm traffic aware.

3.2.2 Depth Potential Field

To provide the basic routing function (which the smooth "bowl" does), namely, to make each packet flow toward the sink, TADR defines the depth potential field $V_d(v)$ as

$$V_d(v) = Depth(v), \tag{2}$$

where $Depth(v)$ is the depth of node v. Thus, the depth field force from node v to its neighbor $w \in nbr(v)$ is given by

$$F_d(v,w) = \frac{V_d(v) - V_d(w)}{c_{vw}}. \tag{3}$$

$Depth(v)$ is quite similar to the length of the shortest path since they both represent the distance from the destination. If the shortest path algorithm chooses the radio hops as its routing metric, $Depth(v)$ will actually become the length. Due to the centralized traffic pattern in WSNs, $Depth(v)$ has some special properties. The depth difference between node v and its neighboring node $w \in nbr(v)$ can only be one of -1, 0, and 1, since the nodes two hops away from a node cannot become its neighbors. Hence, the depth field force $F_d(v,w)$ should also be one of 0, $\frac{1}{c_{vw}}$, and $-\frac{1}{c_{vw}}$.

In a word, the depth potential field encourages packets to flow directly to the sink.

3.2.3 Queue Length Field

In our "bowl" model, packets move from a node to a neighbor with lower potential. To avoid a hotspot which is identified by a large queue, the potential at this node should be raised. Now, we define the queue length potential field at node v as

$$V_q(v) = Q(v). \tag{4}$$

Here, the function $Q(v)$ denotes the normalized queue length at node v and defined by

$$Q(v) = \frac{Number\ of\ packets\ in\ the\ queue}{Buffer\ Size\ at\ node\ v}.$$

Then, we define the queue length potential force $F_q(v,w)$ from node v to $w \in nbr(v)$ as follows:

$$F_q(v,w) = \frac{V_q(v) - V_q(w)}{c_{vw}}. \tag{5}$$

The range of $Q(v)$ is $[0,1]$, and hence $F_q(v,w) \in [\frac{-1}{c_{vw}}, \frac{1}{c_{vw}}]$.

Driven by this potential field, packets will always be forwarded toward the underloaded areas, bypassing the hotspots. However, this field is not loop free since it changes dynamically. We have noticed these routing loops in our simulation experiments, and explain later why some of these loops are reasonable.

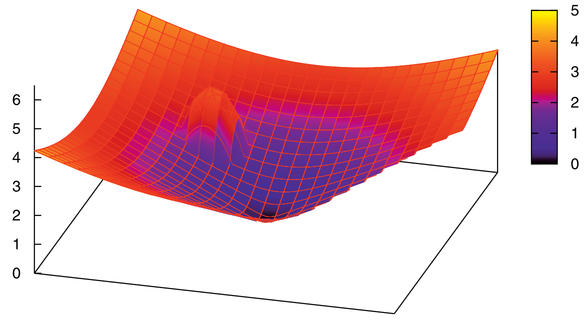


Fig. 3. An example of hybrid potential field.

3.2.4 Superposition of Potential Fields

We have defined two different potential fields. The queue length field makes the routing algorithm traffic aware, which is the main attribute of our TADR scheme. How the queue length field performs on the routing decision depends on the way that the above two independent potential fields are combined together. Naturally, there are various combination patterns, including linear and nonlinear. It is difficult to find a perfect method to make an optimal combination. Intuitively, whichever combination pattern is adopted, the final routing decision depends on three factors, i.e., the values of depth field and queue length field, the combination expression, and coefficients of various items in the combination expression. Actually, the latter two factors determine the degree of influence of different potential fields on decision making together. For simplicity and tractability, we linearly combine them as follows:

$$V_m(v) = (1-\alpha)V_d(v) + \alpha V_q(v), \tag{6}$$

where $V_m(v)$ is the potential of this combined field at node v, and $0 \le \alpha \le 1$. The adjustable α independently controls the degree of influence of two fields on routing decision. Then, the combined force from node v to $w \in nbr(v)$ is

$$F_m(v,w) = \frac{V_m(v) - V_m(w)}{c_{vw}}. \tag{7}$$

This equation can be rewritten as

$$F_m(v,w) = (1-\alpha)F_d(v,w) + \alpha F_q(v,w). \tag{8}$$

The radio link cost c_{vw} in the proposed TADR scheme takes the distance into account for the sake of convenience, namely:

$$c_{vw} = dist_{v\to w}, \tag{9}$$

where $dist_{v\to w}$ denotes the distance from node v to node w. If we use the radio wave range as unit to measure the distance between the nodes, then

$$\begin{cases} 0 < dist_{v\to w} \le 1, & \text{if } w \in nbr(v), \\ dist_{v\to w} > 1, & \text{if } w \notin nbr(v). \end{cases}$$

Fig. 3 depicts an example of the hybrid potential field. The queue occupancy only appears on the small part of nodes, where the hybrid potential field bulges.

3.2.5 Select the Next Hop Node

In the aforementioned bowl model, the water drops will flow along the direction of the maximum force, which is also the direction of the steepest gradient. TADR selects the next hop node just following this rule. Note that, in a lightly loaded network or a lightly loaded period, the steepest gradient method will choose the shortest paths. Otherwise, it will dynamically choose multiple paths, not only the shortest ones, since the queue length is dynamically changing. That opens the way for the underloaded areas to cache or relay the overflowing packets.

4 Properties of TADR and Other Considerations

4.1 Choosing Multiple Paths

The parameter α reflects the weight of the queue length factor, and thus naturally decides when TADR begins to drive packets out of the congested path and scatter them in other underloaded areas. It is determined by the following theorem:

Theorem 4.1. *For node v in depth d, let $S = \{x|Depth(x) = d, x \in nbr(v)\} \cup \{v\}$ and $L = \{x|Depth(x) = d-1, x \in nbr(v)\}$. If $\exists s \in S$ and s can be chosen as the next hop node of v, the parameter α should satisfy*

$$\alpha > \frac{1}{1+\left(\frac{c_{vl}}{c_{vs}}-1\right)Q(v)+\left(Q(l)-\frac{c_{vl}}{c_{vs}}Q(s)\right)}, \tag{10}$$

where c_{vl} and c_{vs} are the link cost from v to l and s, respectively.

Proof. For node v in depth d, let $l \in L$ be the node with the minimal queue length among all the lower depth neighbors, and let $s \in S$. Note that, if v does not choose l as its parent, it will not choose any other nodes in L since the queue length of l has already been the minimal. Now, the potential values at each of nodes are

$$V_m(v) = (1-\alpha)d + \alpha Q(v), \tag{11}$$
$$V_m(l) = (1-\alpha)(d-1) + \alpha Q(l), \tag{12}$$
$$V_m(s) = (1-\alpha)d + \alpha Q(s). \tag{13}$$

Then, we get the force values in node v as

$$F_m(v,l) = \frac{1-\alpha+\alpha(Q(v)-Q(l))}{c_{vl}}, \tag{14}$$
$$F_m(v,s) = \frac{\alpha(Q(v)-Q(s))}{c_{vs}}. \tag{15}$$

If node v chooses s rather than l as its parent, there must hold

$$F_m(v,s) > F_m(v,l). \tag{16}$$

Substituting (14) and (15) into (16), we can get

$$\alpha > \frac{1}{1+\left(\frac{c_{vl}}{c_{vs}}-1\right)Q(v)+\left(Q(l)-\frac{c_{vl}}{c_{vs}}Q(s)\right)}.$$

□

Specially, when $c_{vl} = c_{vs}$, we get

$$\alpha > \frac{1}{1+\Delta Q}, \tag{18}$$

where $\Delta Q = Q(l) - Q(s)$.

Remarks. Theorem 4.1 grants that a node can choose its parent from the same depth neighbors. A node can also choose itself as its own parent, which means that all packets will stay in this node rather than being sent out. Intuitively, in our bowl model, only if the potential of nodes in the lower depth grows higher than that of nodes in the same depth, the packets can be forwarded to the same depth neighbors. For any node, the potential can increase by α at most (when the queue length is 1). Therefore, if and only if $\alpha > 1-\alpha$ where $(1-\alpha)$ is the inherent depth potential difference, this node will choose a neighbor in the same depth as its parent. Hence, we get $\alpha > \frac{1}{2}$. This is simply the special case of expression (18) with $\Delta Q = 1$.

Note that ΔQ is an important value since it is the threshold for allowing the same depth neighbors to join in the competition of being chosen as the parent: if the difference of queue length between the lower depth neighbors and the same depth ones exceeds ΔQ, nodes will send their packets out of the shortest path. Given $\Delta Q \in [0,1]$, we have $\alpha > \frac{1}{2}$. Since ΔQ has definite physical meaning, i.e., the difference in queue length, we assign

$$\alpha = \frac{1}{1+\Delta Q} \tag{19}$$

and then control the weight of the queue length factor by adjusting ΔQ.

We now check the condition that allows packets to go backward to the higher depth neighbors.

Theorem 4.2. *For node v in depth d, let $H = \{x|Depth(x) = d+1,\ x \in nbr(v)\}$ and $L = \{x|Depth(x) = d-1, x \in nbr(v)\}$. If $\exists h \in H$ and h can be chosen as the next hop node of v, the parameter α needs to satisfy*

$$\alpha > \frac{1+\frac{c_{vl}}{c_{vh}}}{1+\frac{c_{vl}}{c_{vh}}+\left(\frac{c_{vl}}{c_{vh}}-1\right)Q(v)+\left(Q(l)-\frac{c_{vl}}{c_{vh}}Q(h)\right)}, \tag{20}$$

where c_{vl} and c_{vh} are the link metric from v to l and h, respectively.

Proof. The proof of Theorem 4.2 is similar to that of Theorem 4.1 □

Also, when $c_{vl} = c_{vh}$, we get

$$\alpha > \frac{1}{1+\frac{\Delta Q'}{2}}, \tag{21}$$

where $\Delta Q' = Q(l) - Q(h)$.

Remarks. $\Delta Q'$ is the threshold that allows the higher depth neighbors to join in the competition of being chosen as the parent, similar to ΔQ. According to expression (19), the relationship between $\Delta Q'$ and ΔQ is

$$\Delta Q' > 2\Delta Q. \tag{22}$$

The meaning of this expression is that, for $\alpha = \frac{1}{1+\Delta Q}$, if the difference of queue length between the lower depth neighbors (which are in the shortest paths) and the same depth ones exceeds ΔQ, the proposed TADR may choose its parent from the same depth neighbors; and if

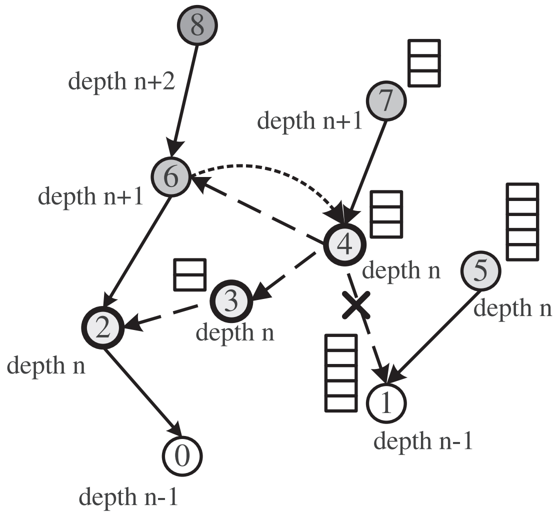


Fig. 4. Illustration of TADR procedure.

the difference of queue length between the lower depth neighbors and the higher depth ones exceeds $2\Delta Q$, then TADR may choose the parent from the higher depth neighbors, namely, to forward the packets backward.

4.2 Caching and Spreading over Time

Due to the centrality of traffic in WSNs, simply bypassing the intermediate local hotspots is hard to completely avoid congestion because the hotspots would reappear near the sink if most of the scattered packets approach the sink from different directions simultaneously.

A straightforward solution for this problem is just to cache these packets in the idle or underloaded nodes, waiting for some time to approach the sink. In other words, besides spreading spatially the excessive packets along the multipaths, it is also necessary to spread temporally the packets for further improving the overall throughput. As an enhanced mechanism, the TADR scheme will use the following rule to achieve this goal:

Rule 1. If $Q(w) = 1,\ w \in nbr(v), w$ should not be selected as the parent in any case.

Rule 1 avoids dropping packets at the hotspots—it is more reasonable for the packets to be cached in the local node rather than to be dropped at the parent. Analogously, in our bowl model, if the bottom is full of water, the newcomers will stop at the surface to wait for the water to drain through the "hole" at the bottom.

Additionally, *Rule 1* implies implicit source rate control. If all the paths between a source node and the sink are full of cached packets, this source node will be compelled to slow down. *Rule 1* may also benefit energy efficiency since it pushes the dropping of packets backward to the source nodes, which reduces the energy wastage.

However, nodes will sometimes stop transmitting packets under *Rule 1*, and thus the movement of these packets is slowed down, which in turn increases the end-to-end delay.

As a summary, Fig. 4 can clearly explain the meaning of above theorems and RULE. Because of the burst traffic from node 5, node 1 becomes a hotspot. Therefore, when node 4 chooses its parent, node 1 is not an appropriate candidate. If the difference of the normalized queue length between node 3 and node1 exceeds ΔQ, TADR may choose node 3 as its parent, namely, the packets in node 4 are moved to the same depth neighbor. If the difference of the normalized queue length between node 6 and node 1 exceeds $2\Delta Q$, the packet is likely sent backward. This choice can be purposely prohibited, but it also implies that the routing protocol gives up the chance that the packet is moved to node 0 with the lower depth through the detour path $4 \rightarrow 6 \rightarrow 2 \rightarrow 1$. Otherwise, the packet in both node 6 and node 3 could also be sent back to node 4 when the transient congestion disappears and the queue on node 4 is drained away after some time, which forms a routing loop. Most of these routing loops are beneficial to alleviating congestion and improving throughout because both node 3 and node 6 can be regarded as a caching pool under this situation. The memory resources on node 4 will be released to accommodate the packets from its upstream nodes, such as node 7. In addition, RULE 1 defines another caching scheme. For example, if the normalized queue length in all direct neighbors of node 4 equals to 1, the packet will stay in node 4.

5 Design of Distributed Routing Algorithm

TADR needs the state information from neighbor nodes, such as queue length and depth, to construct the potential field. The excessive update messages will introduce overhead. To exchange message among direct neighbors and retain this overhead, we design a signaling and some auxiliary mechanisms.

5.1 Update Message

There are only two bytes in the update message used in the TADR scheme. An 8-bit field for depth and another 8-bit field for queue length. We assume that all nodes in the network are homogenous and have the same buffer size, thus TADR can get the normalized queue length. The reason for not sending the potential V_m directly is that it will cost more space to store a floating point number than two integers.

5.2 Depth

The function of TADR significantly depends on the depth potential field, which forms the smooth surface of the bowl. We will discuss how to construct this field in this section.

In the beginning, the depth of all nodes is initialized to $0xff$, except for the sink whose default depth is 0. The sink first sends the update message, and nodes one hop away from the sink get their own depth by adding 1 to the depth value in the update message. Then, the other nodes also obtain their own depth by receiving update messages from their neighbors which already have a depth in the same way as the nodes one hop away. Otherwise, TADR will recalculate the depth when it detects the topological changes. The simplest way is to add 1 to the minimum effective depth value remaining in the routing table.

5.3 Distance

The distance between two neighbors can be easily obtained by several techniques, such as signal attenuation evaluation or estimation depending on Received Signal Strength Indicator (RSSI) [15]. It is noted that the distance used in TADR may be approximate since it is enough to distinguish relatively far or near from the local node.

```
TADR Algorithm – Time to Update
If one of these Events Occurred:
1. Time Out of the Most Updating Interval
2. Depth Changed
3. The variation of Queue Length exceeds Q_update_threshold

1: if (Time Out of the Least Update Interval) then
2:         sendUpdateMsg();
3: else
4:         UpdateMsgPending = TRUE;
5: end if
```

Fig. 5. Pseudocode of TADR algorithm: Time to update.

5.4 Time to Update

Generally, the depth field is quite stable and can be updated in a long-term fashion; the queue length field varies just when there is traffic passing by, so it can be updated in a triggered fashion. In our implementation, to keep the pace of updating, TADR defines a *maximum updating interval* (MUI) and a *least updating interval* (LUI) between two successive update messages. The update messages should be delivered between an LUI and an MUI since the last one. The LUI prevents sending too many update messages and the MUI is used to keep the connectivity of the network. If there are more than two MUIs since receiving the last message from a neighbor, this neighbor is considered dead, and TADR will recalculate the local depth and the routing table. TADR will send an update message when any one of the following events occur:

- *Event 1.* MUI timer expires. If the elapsed time since sending the last update message exceeds an MUI, the node will send a new one immediately, no matter whether the depth or the queue length has changed. This is always used to maintain the depth potential field and keep the connectivity of the network since MUI is relatively large.
- *Event 2.* Depth changes. If the depth of a node has changed, and the elapsed time also exceeds an LUI since the last successful update message, the node will also send a new one.
- *Event 3.* The variation of the queue length exceeds a certain threshold, denoted $Q_{update\_threshold}$. If the queue length on a node has changed by $Q_{update\_threshold}$, such as 0.1, and the elapsed time also exceeds an LUI since the last update message, the node will send a new one. The queue length potential field is maintained by this event.

By setting a large MUI and updating the queue length field in a triggered fashion, TADR can efficiently reduce the overhead. Additionally, TADR just exchanges routing messages with its direct neighbors, which further decreases the cost. The pseudocode about how to update in TADR is listed in Fig. 5.

5.5 Processing of Update Message

When a node receives an update message from one of its neighbors, it will refresh its routing table and reselect a next hop node according to the previous algorithm. The pseudocode for processing the update message is listed in Fig. 6.

```
TADR Algorithm – Update Message Processing
If Received an Update Message (u_Msg) from a
Neighboring Node (neighbor_ID)

1: insertToRoutingTable(neighbor_ID, u_Msg)

2: for each entry in routing table
3:        w = ID of the neighbor;
4:        c = cost of radio link to w;
5:        d = depth of w;
6:        q = queue length of w;
7:        $F_d(w)$ = (Local_Depth – d) / c;
8:        $F_q(w)$ = (Local_QueueLength – q) / c;
9:        $F_m(w) = (1 - \alpha)F_d(w) + \alpha F_q(w)$;
10: end for

# recalculate the depth
11: select the Lowest Depth from the routing table as LD
12: setLocalDepth(LD + 1);

# Choose the next hop node
13: selects from the entries with QL<1 //RULE 1
                  according to max-$F_m$, max-$V_m$, min-depth,
                           min-cost, Random in turn
```

Fig. 6. Pseudocode of TADR algorithm: Processing the update message and choosing a parent. Function *inserToRoutingTable()* updates the depth and queue length values in the routing table for the corresponding neighbor; *setLocalDepth()* sets the depth value of the local node.

TADR uses the steepest gradient method to choose the parent. More precisely, if there are more than one neighbor which has the same maximum force F_m, TADR will choose the next hop node according to maximum potential V_m, minimum depth of neighbors and minimum cost of links in turn. After doing that, if TADR still cannot determine the candidate parent, it will choose one randomly.

6 Performance Evaluations

In this section, we evaluate the performance of TADR using simulation experiments conducted on the TOSSIM platform [11] built in TinyOS. The benchmark protocol is TinyOS's standard routing algorithm MintRoute, which is correctly implemented in TinyOS. MintRoute employs hops from the sink and the quality of radio links to find the next hop node. In an overloaded network, MintRoute also chooses multiple paths to route packets since it uses the number of dropped packets at the local node to identify the link quality. We will also use TADR ($\Delta Q = \infty$) which degenerates to a kind of shortest path algorithm enhanced by *Rule 1* as a comparative algorithm. Finally, we show the impact of key parameters, such as ΔQ and *Rule 1*, respectively.

To evaluate the performance of routing protocols separately, it is feasible to orthogonalize the network layer and the MAC layer. Therefore, we assume a perfect MAC protocol that can efficiently provide a stable radio link and implement it in the TOSSIM simulator.

6.1 Performance Metrics

For a comprehensive performance evaluation, let us define several quantitative metrics.

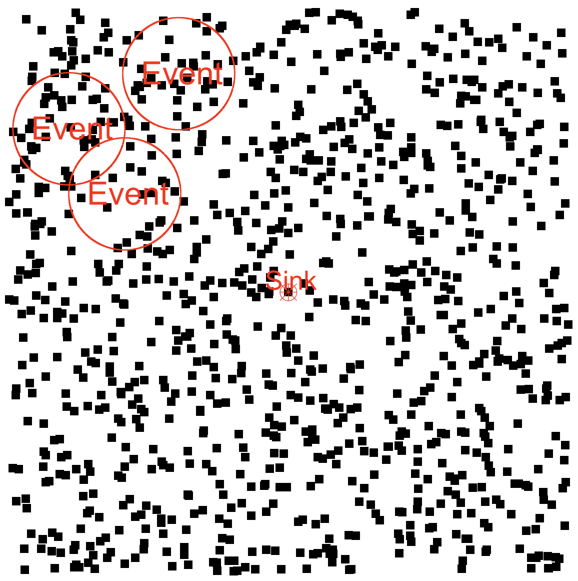


Fig. 7. Random deployment network.

1. **Receiving packets rate** (RPR). It is defined as the rate at which the sink receives packets. It is the appropriate metric that reflects the effect of "caching and spreading over the time and space," because RPR will keep a nonzero value for a relatively long period if the excessive packet is spread spatially and temporarily.
2. **Throughput ratio** (TR). It is defined as

$$TR = \frac{Number\ of\ Packets\ Received\ by\ the\ Sink}{Number\ of\ Packets\ Sent\ by\ Source\ Nodes}.$$

3. **Energy consumption per received packet** (ECRP). The average consumed energy per packet received by the sink reflects the energy efficiency of the protocols.

TABLE 1
Configuration of Parameters: Random Deployment

Deployment	Area Size	100m × 100m
	Deployment Type	Randomly
	Network Architecture	Homogeneous, Flat
	Number of Nodes	999
	Largest Depth	36
	Average Node Degree	10
	Sink Coordinate	(50‹ 50)m
	Radio Range	6m
	Link Layer Transmission Rate	8 Kbps
Task	Application Type	Event-driven
	Packet Size	25 Byte
TADR	Buffer Size	31pkts
	ΔQ	0▹4 (α = 0▹71)
	MUI	10 seconds
	LUI	0▹1 seconds
	$Q_{update\_threshold}$	10%
Simulation	Time	400 seconds

TABLE 2
Events and Bursts Description

	Event 1	Event 2	Event 3
burst 1	$110s \sim 140s$	$120s \sim 150s$	$130s \sim 160s$
burst 2	$210s \sim 240s$	$220s \sim 250s$	$230s \sim 260s$

It is ratio of the total energy consumption to the number of packets received by the sink successfully. The lower the energy consumed per packet, the higher the energy efficiency. In our simulation, we assume 1 unit of energy (1 unit = 60 mJ according to the measurements in Telos platform in [13]) is consumed for receiving packet. Since sending a packet always needs more energy than receiving one [14], assume that 1.5 units of energy is consumed for sending packet with a constant size.

6.2 Simulation Setup

Fig. 7 shows a randomly deployed network and three event areas. All 999 nodes spreading over a $100\ \mathrm{m} \times 100\ \mathrm{m}$ square form a flat multihop network. There is only one sink residing at the center, and the radio wave range is 6 m. The detailed deployment configuration is summarized in Table 1, and the occurrence of the three events is described in Table 2. Every event triggers 40 packets every second, and there are two bursts for each event. The line with cross marker in Fig. 8 describes the average number of sending packets triggered by the three events every second.

6.3 Comparative Analysis

Receiving packets rate. Fig. 8 also shows the RPR value of different schemes, which is an average over periods of 10 s. Obviously, there is a spread of transmission over the time with TADR ($\Delta Q = 0.4$). A lot of packets, which are likely dropped by MintRoute, are cached for a short time and eventually reach the sink. Thus, TADR ($\Delta Q = 0.4$) successfully smooths the bursts and improves the throughput. Relatively, MintRoute drops most of the burst packets, while TADR ($\Delta Q = \infty$ without *Rule 1*) performs much better than MintRoute. Out of the bursting time ($160-210$ s, $260-310$ s), both of them receive few packets at the sink, but TADR ($\Delta Q = 0.4$) continues to receive more

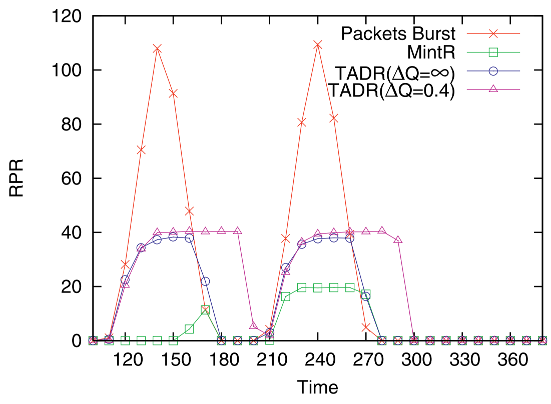
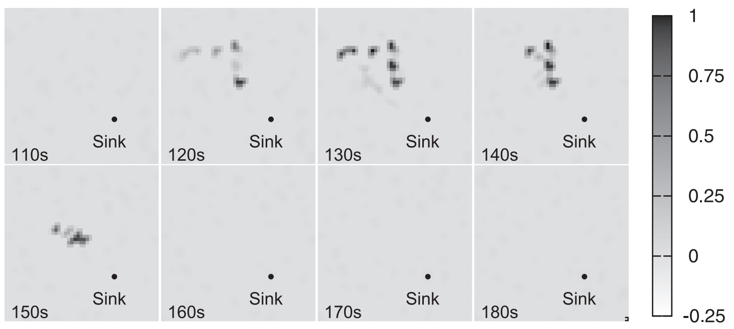
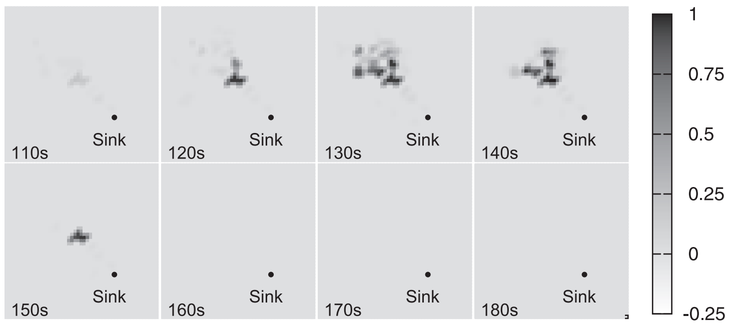


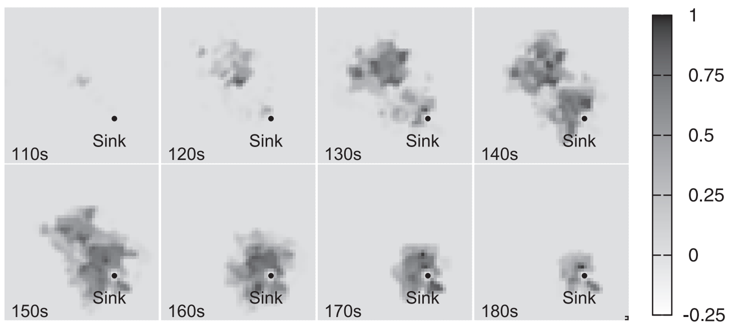
Fig. 8. The distribution of transmission over the time. The vertical axis denotes the rate of receiving packet rate on the sink in unit of pkts/s.



(a)



(b)



(c)

Fig. 9. The spatial distribution of transmission. The higher the gray-scale bar, the larger the normalized queue length on nodes. (a) MintRoute. (b) TADR ($\Delta Q = \infty$). (c) TADR ($\Delta Q = 0.4$).

packets. Fig. 8 only exhibits that TADR can enforce the temporal spreading during packet transmission, in order to vividly demonstrate the spatial spreading, we take a series of snapshots of the normalized queue length, and present them in Fig. 9. Apparently, comparing with MintRoute and the enhanced shortest path routing (TADR ($\Delta Q = \infty$)), TADR ($\Delta Q = 0.4$) spreads traffic over the network resulting in sufficient utilization of more resources.

Throughput ratio. The statistics of the throughput are listed in Table 3. There are 371.3 and 25.0 percent improvements compared with MintRoute and TADR ($\Delta Q = \infty$ with *Rule 1*), respectively. When comparing with TADR ($\Delta Q = \infty$ without *Rule 1*), there is a 54.5 percent improvement.

Energy efficiency. Table 3 also presents the energy consumption per received packet. The energy efficiency of our TADR with $\Delta Q = 0.4$ is a little worse than the enhanced shortest path algorithm TADR ($\Delta Q = \infty$ with *Rule 1*) but better than MintRoute.

Obviously, MintRoute cannot properly avoid congestion and results in excessive packet droppings. These worthless energy wastages make its energy consumed per received packet (ECRP) be higher than any one TADR.

TABLE 3
Statistics of Performance Metrics

		TADR (Δ $Q = \infty$)		
	MintRoute	no *RULE 1*	*RULE 1*	TADR (Δ $Q = 0.4$)
TR	17.8%	54.2%	67.1%	83.9%
ECRP	151.1	118.6	132.8	143.5

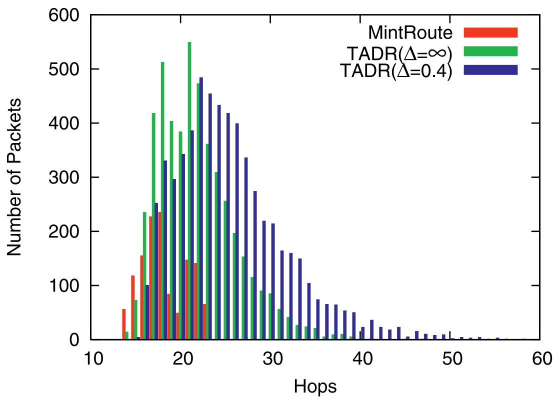


Fig. 10. The distribution of hops.

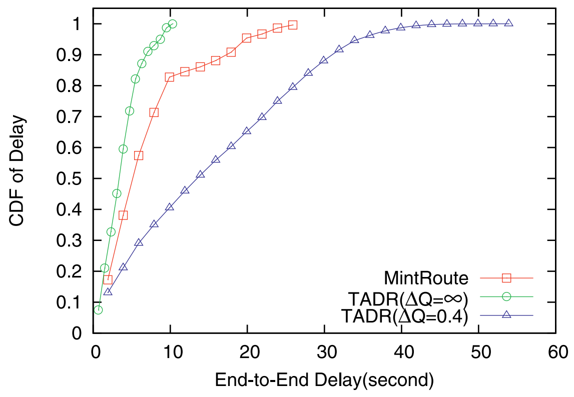


Fig. 11. Cumulative distribution function of end-to-end delay.

The reason that TADR with $\Delta Q = 0.4$ has relatively higher ECRP than TADR with $\Delta Q = \infty$ includes twofold aspects. On the one hand, compared with the enhanced shortest path routing algorithm TADR ($\Delta Q = \infty$), the packets in TADR with $\Delta Q = 0.4$ are scattered to multiple paths, and most of the cached packets travel for more hops before reaching the sink. Fig. 10 shows the distribution of received packets with respect to the hops experienced by packets. The statistical results confirm this fact. Naturally, the energy consumption in TADR with $\Delta Q = 0.4$ is higher than one in TADR with $\Delta Q = \infty$. This cost needs to be paid for achieving high throughput. On the other hand, when the long-traveled packets compete for the limited bandwidth of the last hop, which always is the bottleneck of the network, it is hard to eliminate packet droppings due to occasional buffer overflow. The transmission of packets traveling many hops before being dropped will deteriorate energy efficiency. The longer the packets travel, the more obvious the negative impact on energy efficiency is. This extra energy wastage is not beneficial for improving throughput, but causes ECRP increase. *Rule 1* can only restrain this wastage, but hardly eliminate it completely.

Compared with the shortest path routing algorithm TADR ($\Delta Q = \infty$) and MintRoute, TADR with $\Delta Q = 0.4$ makes a detour, which implies that the end-to-end delay increases. The cumulative distribution function of end-to-end delay presented in Fig. 11 confirms this prediction.

6.4 Different Traffic Pattern

In the previous experiment, to examine the ability of TADR to alleviate congestion and improve throughput, the events are elaborated to generate burst and heavy traffic. Next, the light load is configured to verify its adaptability to various load. Three events inject traffic with rate 10 packets/s during 0-200 s, 100-300 s, and 200-400 s, respectively. The other parameters keep unchangeable. The statistics are listed in Table 4. Combining with the RPR curve shown in Fig. 12 and the cumulative distribution function of end-to-end delay depicted in Fig. 13, we can readily find that our TADR degenerates a shortest path routing algorithm when the traffic is relatively light.

TABLE 4
Statistics of Performance Metrics under Light Load

	MintRoute	TADR ($\Delta Q = \infty$)	TADR ($\Delta Q = 0.4$)
TR	73.6%	100%	100%
ECRP	56.10	44.47	44.47

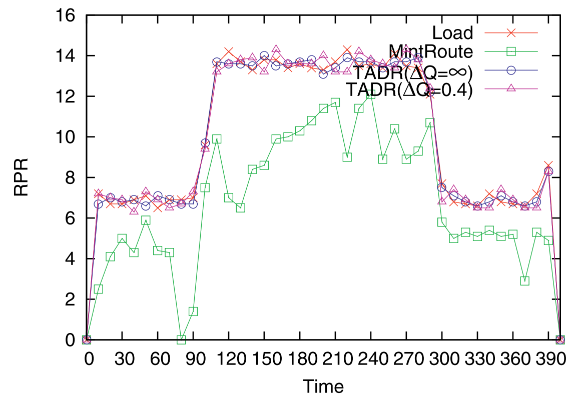


Fig. 12. Receiving packet rate under light load.

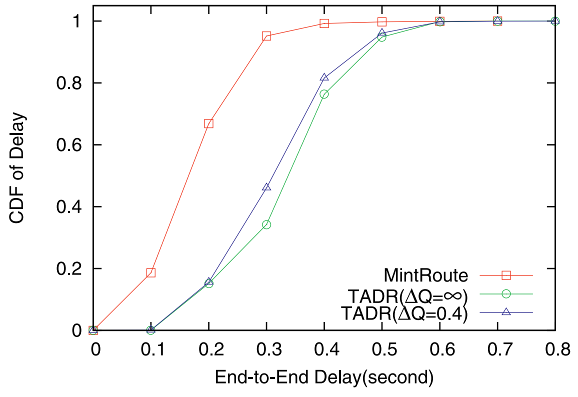


Fig. 13. CDF of delay under light load.

6.5 Impact of ΔQ

Throughput ratio. As explained in Section 4, ΔQ represents the threshold at which TADR begins to route the packets into multiple paths. The smaller ΔQ, the more packets will be scattered to the multipath or be cached for subsequent transmission, which finally brings a higher throughput. Fig. 14 confirms this conclusion.

Additional overhead. Since the cost of detecting the topology changes is unavoidable for all the routing algorithms, we just check the additional overhead caused by the variation of the dynamic queue length potential field. Fig. 15 shows the ratio of the total number of bytes transmitted in the

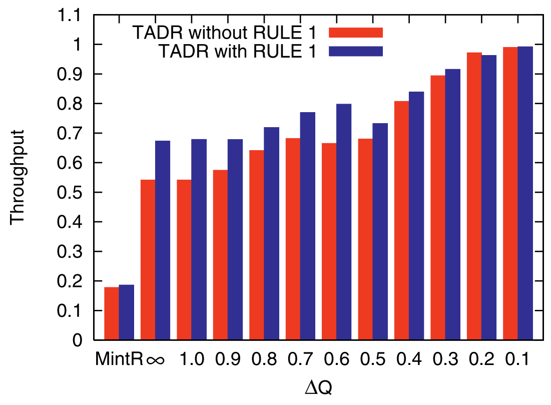


Fig. 14. Throughput under different parameters.

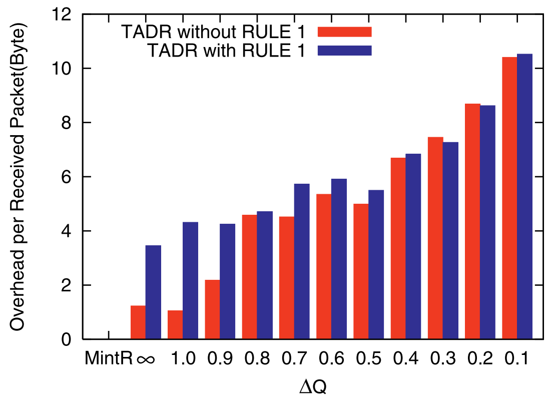


Fig. 15. Additional overhead caused by queue variations.

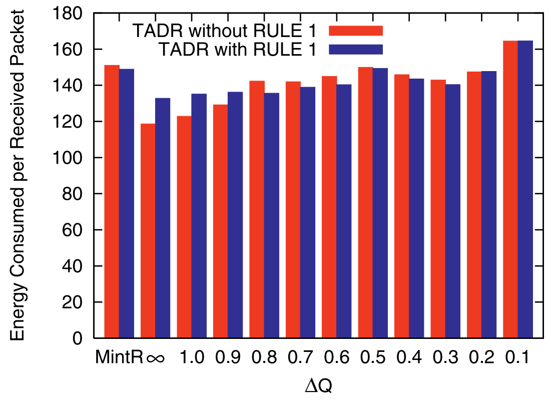


Fig. 16. Average energy consumed per received packet.

update message to the number of received packets at the sink. For a clear explanation, we take TADR with $\Delta Q = 0.4$ for example: for a successfully received packet, the whole additional overhead is to broadcast less than 7 bytes over just one hop, since no update message is ever relayed. A smaller ΔQ increases the weight of the queue length potential field (see (19)), and hence aggravates the dynamics of the hybrid field, thus resulting in a higher overhead.

We have also checked the influence of ΔQ on the energy efficiency, as shown in Fig. 16. ECRPs of TADR are quite similar for all ΔQs and most of them are below that of MintRoute, which shows a good energy efficiency.

6.6 Impact of RULE 1

As mentioned in Section 4, *Rule 1* ensures the excessive packets be effectively cached and also provides an implicit source rate control. In this section, we will compare the

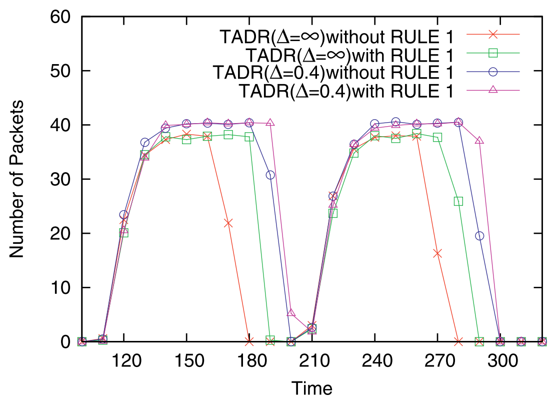


Fig. 17. Spreading the transmission over the time.

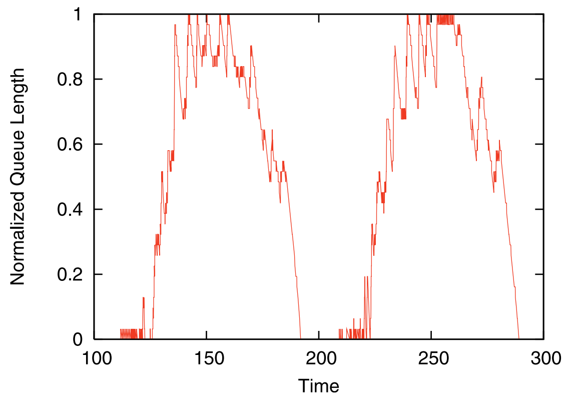


Fig. 18. The queue length at node 298.

performance of TADR under this rule. The RPR curves in Fig. 17 confirm that *Rule 1* makes more excessive packets be well cached and thus also improves the throughput (also see Fig. 14).

Additionally, it seems that *Rule 1* works well only when there are many packets dropped. That is because if the queues on nodes rarely overflow, *Rule 1* will never be activated. We can see from the simulation results in Fig. 14 that there is an obvious throughput improvement for high ΔQs under which the queues are prone to overflow because TADR fails to find enough idle nodes to cache all the dropped packets.

6.7 Explanations to Routing Loops

In [1], Basu et al. proved that the time-invariant potential field is loop free. Unfortunately, from our observation, the queue length potential field varies with time in TADR scheme. We have also noticed routing loops in our simulation results. Fig. 18 shows the variation of the length of the queue at node 298 which resides near the sink (depth 1) and on the shortest path of the event areas and Fig. 19 presents how many times the routing loops occur under different ΔQ. However, we think that most of these routing loops may be reasonable, or even indispensable for congestion avoidance and throughput improvement.

A typical routing loop is caused by a local minimal potential, which is a hollow in our bowl model. At the beginning, nodes around this minimal potential node may send their packets to it, so this hollow will be filled up after some time. Once the potential of this node goes higher than that of any node around it, the node will send back packets

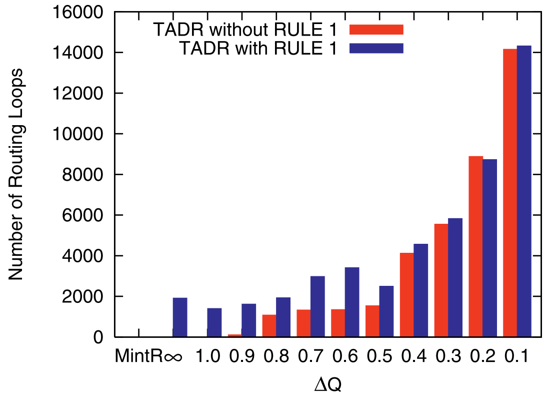


Fig. 19. ♯ of routing loops under different ΔQ.

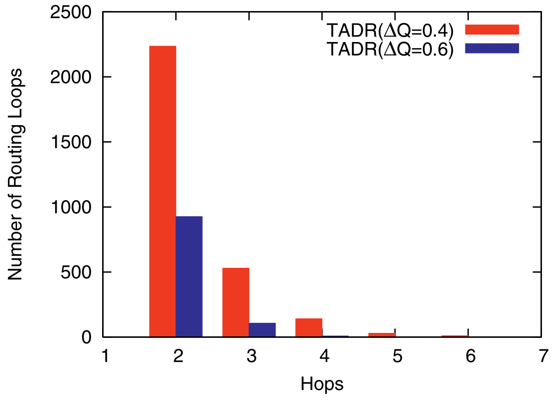


Fig. 20. Distribution of hops in routing loops.

to its neighbors which just sent packets to it, thus creating a routing loop. An example is illustrated in Fig. 4. Node 6 is a "hollow" node, and it will form a looping circle with node 4 by the dashed line after some time when its queue is filled up. We think this type of loops is reasonable because the node with a locally minimal potential acts like a packet pool, which caches packets to achieve the regular temporal spreading. Most of this kind of routing loops occur within two hops. The hop distribution of routing loops shown in Fig. 20 can verify that the loop with two hops is the majority, although there are other types of routing loops caused by other factors, such as inaccurate information about queue length caused by MAC delay when transferring the update messages. Note that all routing loops are not permanent because the time-invariant depth field dominates routing decision when a WSN operates. On the other hand, TADR is capable of making a WSN recover from congestion, namely, keeping it run normally.

7 Conclusion and Future Work

The congestion control in WSNs is different from that in tradition networks, such as Wireless LAN and ad hoc networks. The pure traffic control is able to alleviate congestion, but hard to satisfy the fidelity required by applications. In this work, we follow the philosophy of dynamic capacity planning to deal with the congestion problem in WSNs. By carefully examining the special characteristics of WSNs, we propose a potential-based traffic-aware dynamic routing algorithm, called TADR. The key idea underlying our algorithm is to define a hybrid scalar potential field, which contains a depth field and a queue length field. The depth field provides the basic routing backbone which routes the packets directly to the sink along the shortest path. The queue length field makes TADR traffic aware. When the congestion appears, the excessive packets are dynamically rerouted to multiple path consisting of idle or underloaded nodes. Therefore, the TADR scheme can effectively alleviate congestion through bypassing the hot spots, and meet the fidelity requirements through improving the overall throughput. An extra Rule is also introduced to prevent dropping of packets at the hotspots near the sink. In a word, our TADR achieves its objectives through spreading spatially and temporally packets in a reasonable pattern. In addition, TADR has other features which simplify the implementation. It does not need any accessorial support, such as coordinate system. In addition, TADR is a distributed and scalable routing algorithm because it just needs local information (from local node and its direct neighbors), and the routing state information (i.e., depth and queue length) is quite easy to obtain.

The limitation in this work is due to a lack of sufficient understanding about the dynamics of time-varying potential fields; thus, the value of the key parameter ΔQ is determined by numerous simulation experiments. If we could build an analytical model for them, we would dynamically select the optimal weight of the hybrid field to adapt to different situations. This is a complex task, but worthwhile to investigate in future work.

Moreover, some WSNs are application oriented, and different applications have different requirements. To adapt to this diversity, an open framework is needed to account for other factors. We believe that a general framework provided by the potential-based routing paradigm could be extended to optimize various other metrics through constructing other potential fields and introducing additional mechanisms for performance enhancement. For example, we can extend TADR to support delay-sensitive and high-integrity applications simultaneously through introducing an enhanced mechanism. The packets from different applications can be assigned to different weights carried in the packet header. The hybrid potential field imposes the different "force" on the different packets to drive them to move along the different paths. The high-integrity packet with light weight is well cached on the idle nodes and/or detoured the underloaded paths, and the delay-sensitive packet with heavy weight travels along shorter paths to approach the sink as soon as possible. These may be possible subjects of future work in this direction.

Acknowledgments

The authors gratefully acknowledge the anonymous reviewers for their constructive comments. This work is supported in part by the National Natural Science Foundation of China (NSFC) under Grant Nos. 60773138, 60971102, the National Grand Fundamental Research Program of China (973) under Grant Nos. 2009CB320504, 2006CB303000, and the National Science and Technology Major Project of China (NSTMP) under Grant Nos. 2009ZX03006-001-003 and 2009ZX03006-003-01.

REFERENCES

[1] A. Basu, A. Lin, and S. Ramanathan, "Routing Using Potentials: A Dynamic Traffic-Aware Routing Algorithm," *Proc. ACM SIGCOMM,* pp. 37-48, 2003.

[2] S. Chen and N. Yang, "Congestion Avoidance Based on Lightweight Buffer Management in Sensor Networks," *IEEE Trans. Parallel and Distributed Systems,* vol. 17, no. 9, pp. 934-946, Sept. 2006.

[3] C. Ee and R. Bajcsy, "Congestion Control and Fairness for Many-to-One Routing in Sensor Networks," *Proc. ACM Int'l Conf. Embedded Networked Sensor Systems (SenSys '04),* 2004.

[4] T. He, J. Stankovic, C. Lu, and T. Abdelzaher, "SPEED: A Stateless Protocol for Real-Time Communication in Sensor Networks," *Proc. IEEE Int'l Conf. Distributed Computing Systems (ICDCS '03),* 2003.

[5] B. Hull, K. Jamieson, and H. Balakrishnan, "Mitigating Congestion in Wireless Sensor Networks," *Proc. Int'l Conf. Embedded Networked Sensor Systems (SenSys '04),* pp. 134-147, 2004.

[6] R. Jain, "Congestion Control and Traffic Management in ATM Networks: Recent Advances and a Survey," *Computer Networks and ISDN Systems,* vol. 28, no. 13, pp. 1723-1738, 1995.

[7] J. Kang, Y. Zhang, and B. Nath, "TARA: Topology-Aware Resource Adaptation to Alleviate Congestion in Sensor Networks," *IEEE Trans. Parallel and Distributed Systems,* vol. 18, no. 7, pp. 919-931, July 2007.

[8] J. Kang, B. Nath, Y. Zhang, and S. Yu, "Adaptive Resource Control Scheme to Alleviate Congestion in Sensor Networks," *Proc. First Workshop Broadband Advanced Sensor Networks,* 2004.

[9] J. Kang, Y. Zhang, and B. Nathg, "End-to-End Channel Capacity Measurement for Congestion Control in Sensor Networks," *Proc. ACM/IEEE SANPA,* 2004.

[10] R. Kumar, R. Crepaldi, H. Rowaihy, A.F. Harris, G. Cao, M. Zorzi, and T.F.L. Porta, "Mitigating Performance Degradation in Congested Sensor Networks," *IEEE Trans. Mobile Computing,* vol. 7, no. 6, pp. 682-697, June 2008.

[11] P. Levis, N. Lee, M. Welsh, and D. Culler, "TOSSIM: Accurate and Scalable Simulation of Entire TinyOS Applications," *Proc. ACM Int'l Conf. Embedded Networked Sensor Systems (SenSys),* pp. 126-137, 2003.

[12] J. Paek and R. Govindan, "RCRT: Rate-Controlled Reliable Transport for Wireless Sensor Networks," *Proc. ACM Int'l Conf. Embedded Networked Sensor Systems (SenSys '07),* 2007.

[13] Y. Panthachai and P. Keeratiwintakorn, "An Energy Model for Transmission in Telos-Based Wireless Sensor Networks," *Proc. Int'l Joint Conf. Computer Science and Software Eng.,* 2007.

[14] A. Papadoulos and J.A. Mccann, "Towards the Design of an Energy-Efficient, Location-Aware Routing Protocol for Mobile, Ad-Hoc Sensor Networks," *Proc. 15th Int'l Workshop Database and Expert Systems Applications,* pp. 705-709, 2004.

[15] N. Patwari, A.O. Hero III, M. Perkins, N. Correal, and R.J. O'Dea, "Relative Location Estimation in Wireless Sensor Networks," *IEEE Trans. Signal Processing,* vol. 51, no. 8, pp. 2137-2148, Aug. 2003.

[16] P.P. Pham and S. Perreau, "Performance Analysis of Reactive Shortest Path and Multipath Routing Mechanism with Load Balance," *Proc. IEEE INFOCOM,* 2003.

[17] L. Popa, C. Raiciu, I. Stoica, and D.S. Rosenblum, "Reducing Congestion Effects in Wireless Networks by Multipath Routing," *Proc. IEEE Int'l Conf. Network Protocols (ICNP '06),* 2006.

[18] S. Rangwala, R. Gummadi, R. Govindan, and K. Psounis, "Interference-Aware Fair Rate Control in Wireless Sensor Networks," *Proc. ACM SIGCOMM,* pp. 63-74, 2006.

[19] Y. Sankarasubramaniam, O. Akan, and I. Akyildiz, "ESRT: Event-to-Sink Reliable Transport in Wireless Sensor Networks," *Proc. ACM Int'l Symp. Mobile Ad Hoc Networking and Computing (MobiHoc '03),* 2003.

[20] J. Teo, Y. Ha, and C. Tham, "Interference-Minimized Multipath Routing with Congestion Control in Wireless Sensor Network for High-Rate Streaming," *IEEE Trans. Mobile Computing,* vol. 7, no. 9, pp. 1124-1137, Sept. 2008.

[21] C.Y. Wan, S.B. Eisenman, and A.T. Campbell, "CODA: Congestion Detection and Avoidance in Sensor Networks," *Proc. ACM Int'l Conf. Embedded Networked Sensor Systems (SenSys '03),* pp. 266-279, 2003.

[22] C. Wan, S. Eisenman, A. Campbell, and J. Crowcroft, "Siphon: Over-Load Traffic Management Using Multi-Radio Virtual Sinks in Sensor Networks," *Proc. ACM Int'l Conf. Embedded Networked Sensor Systems (SenSys '05),* pp. 116-129, 2005.

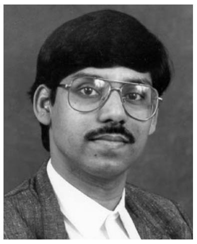
[23] C. Wang, B. Li, K. Sohraby, M. Daneshmand, and Y. Hu, "Upstream Congestion Control in Wireless Sensor Networks through Cross-Layer Optimization," *IEEE J. Selected Areas in Comm.,* vol. 25, no. 4, pp. 786-795, May 2007.

[24] M.H. Yaghmaee and D.A. Adjeroh, "Priority-Based Rate Control for Service Differentiation and Congestion Control in Wireless Multimedia Sensor Networks," *Computer Networks,* vol. 53, no. 11, pp. 1798-1811, 2009.

[25] M. Zawodniok and S. Jagannathan, "Predictive Congestion Control Protocol for Wireless Sensor Networks," *IEEE Trans. Wireless Comm.,* vol. 6, no. 11, pp. 3955-3963, Nov. 2007.

Fengyuan Ren received the BA and MSc degrees in automatic control from Northwestern Polytechnic University, China, in 1993 and 1996, respectively. In Dec. 1999, he received the PhD degree in computer science from Northwestern Polytechnic University. He is a professor in the Department of Computer Science and Technology, Tsinghua University, Beijing, China. From 2000 to 2001, he worked at the Electronic Engineering Department of Tsinghua University as a postdoctor. In Jan. 2002, he moved to the Computer Science and Technology Department of Tsinghua University. His research interests include network traffic management and control, control in/over computer network, wireless network and wireless sensor networks, etc. He authored and coauthored more than 80 international journal and conference papers. He is a member of the IEEE, and has served as a technical program committee member and local arrangement chair for various IEEE and ACM international conferences.

Tao He received the MS degree in computer science and technology from Tsinghua University of China, in 2008. He has been an assistant researcher of NEC Labs, China, since his graduation. His research interests include wireless sensor network and recommendation system.



Sajal K. Das is a university distinguished scholar professor of computer science and engineering and the founding director of the Center for Research in Wireless Mobility and Networking (CReWMaN) at the University of Texas at Arlington (UTA). He is currently a program director at the US National Science Foundation (NSF) in the Division of Computer Networks and Systems. He is also a visiting professor at the Indian Institute of Technology (IIT) Kanpur and IIT Guwahati; an honorary professor of Fudan University in Shanghai; and an international advisory professor of Beijing Jiaotong University, China. His current research interests include wireless and sensor networks, mobile and pervasive computing, smart environments and smart heath care, pervasive security, resource and mobility management in wireless networks, mobile grid computing, biological networking, applied graph theory, and game theory. He has published more than 400 papers and more than 35 invited book chapters in these areas. He holds five US patents in wireless networks and mobile Internet, and coauthored the books *Smart Environments: Technology, Protocols, and Applications* (Wiley, 2005) and *Mobile Agents in Distributed Computing and Networking* (Wiley, 2010). He is a recipient of the IEEE Computer Society Technical Achievement Award (2009) for pioneering contributions to sensor networks, IEEE Region 5 Outstanding Engineering Educator Award (2008), IEEE Engineer of the Year Award (2007), and seven Best Paper Awards including those at EWSN '08, IEEE PerCom '06, and ACM MobiCom '99. At UTA, he is a recipient of the Lockheed Martin Teaching Excellence Award (2009), UTA Academy of Distinguished Scholars Award (2006), University Award for Distinguished Record of Research (2005), and College of Engineering Research Excellence Award (2003). He serves as the founding editor-in-chief of Elsevier's *Pervasive and Mobile Computing* (PMC) journal, and also as an associate editor of *IEEE Transactions on Mobile Computing*, *ACM/Springer Wireless Networks*, *Journal of Parallel and Distributed Computing*, and *Journal of Peer-to-Peer Networking*. He is the founder of IEEE WoWMoM symposium and cofounder of IEEE PerCom conference. He has served as the general and technical program chair as well as TPC member of numerous IEEE and ACM conferences. He is a senior member of the IEEE.

Chuang Lin received the PhD degree in computer science from the Tsinghua University, Beijing, China, in 1994. He is a professor in the Department of Computer Science and Technology, Tsinghua University. He is an honorary visiting professor, University of Bradford, United Kingdom. His current research interests include computer networks, performance evaluation, network security analysis, and Petri net theory and its applications. He has published more than 300 papers in research journals and IEEE conference proceedings in these areas and has published four books. He serves as the technical program vice chair, the 10th IEEE Workshop on Future Trends of Distributed Computing Systems (FTDCS 2004); the general chair, ACM SIGCOMM Asia Workshop 2005 and the 2010 IEEE International Workshop on Quality of Service (IWQoS 2010); the associate editor, *IEEE Transactions on Vehicular Technology*; the area editor, *Journal of Computer Networks*; and the area editor, *Journal of Parallel and Distributed Computing*. He is a senior member of the IEEE and the Chinese Delegate in TC6 of IFIP.