TARA: Topology-Aware Resource Adaptation to Alleviate Congestion in Sensor Networks

Jaewon Kang
DATAMAN Lab.
Computer Science
Rutgers University
jwkang@cs.rutgers.edu

Yanyong Zhang
WINLAB
Electrical & Computer Engineering
Rutgers University
yyzhang@winlab.rutgers.edu

Badri Nath
DATAMAN Lab.
Computer Science
Rutgers University
badri@cs.rutgers.edu

ABSTRACT

Network congestion can be alleviated either by reducing demand (traffic control) or by increasing capacity (resource control). Unlike in traditional wired or other wireless counterparts, sensor network deployments provide elastic resource availability for satisfying the fidelity level required by applications. In many cases, using traffic control can violate fidelity requirements. Hence, we propose the use of resource control: increase capacity by enabling more nodes to become active during periods of congestion. However, a naive approach to increase resources without a careful consideration of the type of congestion, traffic pattern, and network topology will make the situation worse.

In this paper, we present TARA, a topology-aware resource adaptation strategy to alleviate congestion. The core of TARA is our capacity analysis model, which can be used to estimate capacity of various topologies. Detailed performance results show that TARA can achieve data delivery rate and energy consumption that is close to an ideal offline resource control algorithm.

1. INTRODUCTION

A broad range of remote sensing applications are being enabled by the rapid development of sensor networks. Typical remote sensing applications include target recognition and tracking, forest fire detection, structural and habitat monitoring, or traffic condition surveillance. Before the target events occur, sensor nodes report data at a lower rate (e.g., heart-beat messages) to save energy. As soon as these events are detected, however, a high reporting rate is necessary to generate sufficient data to accurately depict the phenomena of interest. As a result, the lifetime of a sensor network alternates between periods of low traffic volume (referred to as *dormant state*) and periods of high traffic volume (referred to as *crisis state*). For most applications, long dormant state dominates the network lifetime.

Unlike wired networks or other wireless counterparts, it is not viable to make all battery-powered nodes active in sensor networks because it bounds the network lifetime by the lifetime of a battery (which is in the order of months). Thus, a popular way of conserving energy is to keep only a minimum number of nodes active during dormant periods. This strategy, though energy-efficient, can cause a serious problem – as soon as the network enters a crisis state, congestion is likely to occur because the data rates may exceed the available capacity. Once congestion occurs, many packets that contain valuable data may be dropped. Although "traffic control" strategies that reduce the incoming traffic have been effective to alleviate congestion in traditional networks, they are unsuitable for our purposes because reducing source traffic during a crisis state is unacceptable. The data generated during crisis state are of utmost importance and loss of these data can defeat the very purpose of deploying unattended sensor nodes. To address this challenge, we propose to use the approach of "resource control" – increasing capacity by turning on more resources to accommodate high incoming traffic.

Activating additional resources in response to increased data volume is a challenging problem. The main challenge stems from the fact that resource control strategies require not only local knowledge, but also knowledge about the end-to-end topology. In this paper, we propose *TARA*, a topology-aware resource adaptation strategy, which can quickly add appropriate sensor nodes to form a new topology that has just enough capacity to handle the traffic, satisfying both fidelity (or throughput) and energy requirements at the same time. TARA's distinct ability to find an optimal topology for the fidelity and energy requirements in response to congestion is enabled by our capacity analysis model that can efficiently estimate the capacity of various network topologies using a graph theoretical approach.

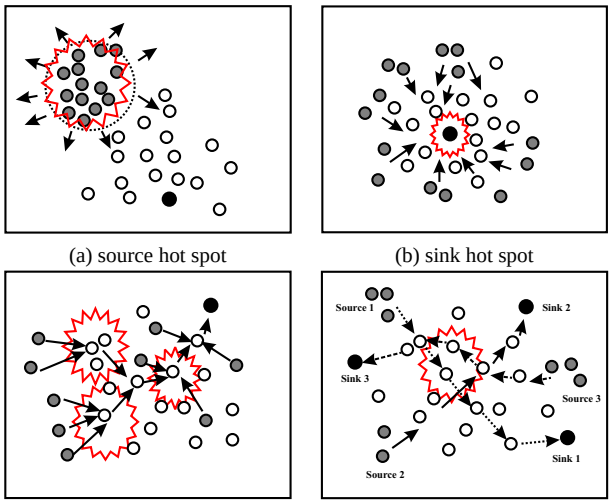
In the rest of this paper, we first identify several typical congestion scenarios in Section 2. In Section 3, we introduce both traffic control and resource control frameworks. Section 4 presents the capacity analysis model and capacities of several typical topologies. The distributed topology-aware resource control protocol is presented in detail in Section 5. Section 6 shows the performance evaluation of TARA. Related work are covered in Section 7. Finally, Section 8 provides the concluding remarks.

2. CONGESTION IN SENSOR NETWORKS

We have identified the following three typical hot spot scenarios within sensor networks:

- *Source Hot Spot*: As soon as an event takes place, it will be detected by all the nodes whose sensing ranges (with radius r) cover the event spot. These nodes will act as *sources*, reporting their observations to the sink(s). Since a node's radio range (R) is usually greater than its sensing range (for example, many realistic deployments assume that $R > 2r$ [22]), these sources are likely to be within each other's radio

.



(a) source hot spot (b) sink hot spot

(c) intersection hot spot - merging traffic (d) intersection hot spot - crossing traffic

Figure 1: Three hot spot scenarios

range as well. If all these source nodes start sending packets at a high rate to the sink, a hot spot will quickly form, and a large number of packets will be dropped around the event spot (Figure 1(a)).

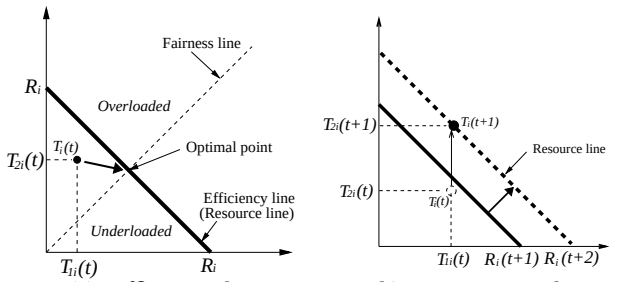
- *Sink Hot Spot*: In a crisis state, sink nodes (and the nodes around them) handle a high traffic volume (Figure 1(b)) [14]. Hence, the batteries of these nodes around sinks will be drained quickly, making the sinks unreachable from the rest of the network.

- *Intersection Hot Spot*: The presence of multiple sources and multiple sinks results in more than one flow intersecting with each other. Due to traffic merge at the intersection nodes, they can also become hot spots (See Figure 1(c) and (d)), referred to as intersection hot spots. These intersecting flows can either share segments of routing paths (illustrated in Figure 1(c)), or cross each other's routing paths (illustrated in Figure 1(d)).

Although each of the three scenarios is harmful to the operations of sensor networks, some of them can be avoided through careful planning at deployment stage. For instance, source hot spots can be eliminated by allowing only a small number of nodes to report to the sink [15, 19]. This will not degrade network services because these nodes are likely to report highly correlated observations due to their geographic proximity. At the same time, one way of alleviating sink hot spots is to deploy multiple sinks that are uniformly scattered across the sensor field, and then balance the traffic among them [14]. These types of planning are feasible because sensor applications often have good estimations of their peak load (during crisis state) and node deployment density.

On the other hand, intersection hot spots are far more challenging because it is very difficult to predict the intersection points due to the dynamic nature of sensor networks. Hence, intersection hot spots cannot be avoided at deployment time, but demand run-time countermeasures. This paper will thus focus on alleviating intersection hot spots.

3. TRAFFIC CONTROL VS.RESOURCE CONTROL



(a) traffic control (b) resource control

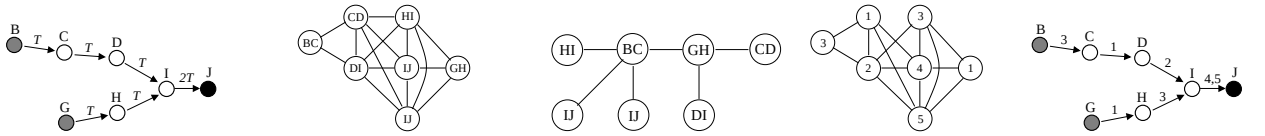
Figure 2: Frameworks of traffic control and resource control

In this section, we present the rationale behind traffic control and resource control schemes, and highlight the necessity of topology awareness for resource control schemes.

Traffic Control: The approach of controlling traffic for congestion avoidance has been extensively studied, mostly through the active queue management (AQM) in wired networks [4]. Two main criteria for traffic control policies are *resource utilization* and *fairness*. We next illustrate how these two criteria are met using the traffic control algorithm between two flows (presented in [2]). Figure 2(a) depicts the offered traffic of both flows at node i (T_{1i} on the x-axis and T_{2i} on the y-axis). Suppose the total resource provisioning at node i is R_i which is constant over time. Then, the efficiency line, also referred to as *resource line*, denotes the scenarios where the available resource is equal to the total traffic volume, i.e. $T_{1i}(t) + T_{2i}(t) = R_i$. The area above the resource line, i.e. $T_{1i}(t) + T_{2i}(t) > R_i$, corresponds to *overloaded* scenarios; the area below the resource line denotes *underloaded* scenarios. The dotted line in Figure 2(a), with $T_{1i}(t) = T_{2i}(t)$, denotes a fair resource allocation between the two flows. The resource line and fairness line intersect at $< \frac{R_i}{2}, \frac{R_i}{2} >$, which corresponds to the optimal scenario, wherein the two flows can fully and equally utilize the available resources. The goal of traffic control is to tune the offered load of all flows to approach the optimal point to ensure that the resource is fully utilized while each flow is allocated a fair share of resource.

Resource Control: The rationale behind resource control strategies is distinctively different from that of traffic control strategies. Traffic control schemes assume a fixed resource provisioning, R_i at node i. However, this constraint does not hold in sensor networks. The available resource (or capacity) at node i is elastic due to its dynamic duty cycle, interference, etc., thereby represented as a time-varying function $R_i(t)$. As a result, resource control schemes seek to satisfy the fidelity level requirement of each flow even during congestion, by assigning additional resources to the flow that has higher fidelity level requirement without taking resources away from other flows as illustrated in Figure 2(b).

Topology-aware Resource Control: The main challenge of resource control stems from the fact that resource control strategies require not only local knowledge, but also knowledge about the end-to-end topology. For example, a naive strategy is to activate all the nodes and create multiple paths to enable parallel forwarding. Nonetheless, this approach can substantially shorten the lifetime of the network. Another strategy would be to activate a random number of nodes that are outside the congested area to detour packets, but there is no guarantee that the new topology can offer larger capacity than the existing configuration, let alone satisfy the applica-



(a) topology (b) spatial interference graph (c) spatial reuse graph (d) colored graph (e) time frame assignment

Figure 3: Illustration of the capacity analysis model

tion fidelity requirement. As a result, an efficient resource control scheme should consider traffic rate, congestion level, and most important of all, network topologies.

4. CAPACITY ANALYSIS MODEL

The core of TARA involves a capacity analysis model, which can efficiently estimate the end-to-end throughput of different topologies. The capacity analysis model is formulated as a graph coloring problem, and we have shown that the model results agree with simulation results and actual experimentation results.

4.1 Model Description

The capacity of a given topology is defined by the maximum throughput (i.e. packet delivery rate) that can be observed by the sink(s). If there were no interferences between links, the capacity of a topology would be the same as the maximum throughput achievable by unlimited unidirectional transmissions in an 1-hop topology. The interference between links, however, makes the overall throughput much smaller than the 1-hop capacity. The objective of TARA's capacity analysis model is to capture the degree of interference of a given topology.

The main idea behind TARA's capacity analysis model is to map the problem of capacity estimation into a suitable graph coloring problem[20]. Figures 3(a)-(e) illustrate this process. Figure 3(a) presents a simple network topology with 7 nodes. Suppose the traffic volume from each source, B and G, is T packets/second, then the traffic for each link is as marked above the link. To produce the maximum throughput, suppose that there exist an optimal schedule that can avoid collisions within the entire topology. Under this optimal schedule, suppose every node in the network has packets in its outgoing buffer, and that the sink receives m packets every n time frames[1]. Then the capacity of this topology is calculated as $\frac{m}{n}C^{max}$, where $\frac{m}{n}$ is called *capacity fraction*, and C^{max} denotes the 1-hop capacity. The capacity fraction of a 1-hop topology is 1 because every time frame one packet is received by the sink. In order to calculate the capacity fraction using graph theoretic methods, we construct the *spatial interference graph* as depicted in Figure 3(b), where the vertexes denote all the wireless links and the edges between two vertexes indicate that these two links are within each other's interference range. That is, any two wireless links with an edge connecting them cannot transmit concurrently under the optimal schedule. In the spatial interference graph, it is necessary to replicate some wireless links (such as IJ in this example) because their traffic loads are multiple of T. In particular, the link whose traffic is mT should be repeated m times. Calculating the capacity fraction now boils down to assigning a coloring to the spatial interference graph, where each color corresponds to the particular time frame in which each link transmits. In Figure 3(d), 5 colors are needed to color the vertexes, and the sink J receives 2 packets every 5 time frames. Thus, the capacity fraction of this topology is $\frac{2}{5}$.

[1]One time frame corresponds to the time interval for a node to transmit one packet to its immediate neighbor.

It is well known that the problem of finding the minimum number of colors for a graph is NP-complete[20]. However, using the following theorems we can obtain an upper bound for colorability: (1) *If G is a simple graph with the largest vertex degree d, then G is $(d+1)$ colorable, and (2) If G is a simple connected graph and not a complete graph, and if the largest vertex degree of G is d ($\geq$ 3), then G is d-colorable*. It should be noted that a spatial interference graph is always connected because there will be no link that does not interfere with any other link. In addition, it is very likely that the graph is not complete due to the limited radio range, and that the largest vertex degree will be greater than or equal to 3 especially in the congested area. As a result, these theorems, especially the second one, give a quick estimation on the number of time frames needed for a certain topology. Applying these methods are meant to serve as guidelines for analyzing the best-case capacity capabilities of the sensor network. In reality, they provide a bound and one can find a better coloring than what these theorems promise.

In addition to theorems that give an upper bound, many heuristic solutions have been proposed to color a certain graph using the minimum number of colors for different applications. To solve our problem of estimating capacity fraction, we also propose the following heuristic solution:

1. Construct the spatial reuse graph, which is the complement of the spatial interference graph, where two vertexes are connected by an edge when they do not interfere with each other, as shown in Figure 3(c).

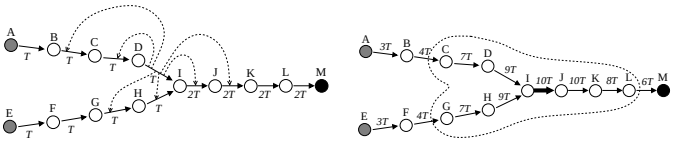
2. Sort all the vertexes in the ascending order of their degrees in the spatial reuse graph. For example, after sorting the vertexes in Figure 3(c), we have: CD, DI, HI, IJ, IJ, GH, BC, where the tie is broken alphabetically.

3. Start from the first vertex in the list, and find the largest complete subgraph that contains this vertex. All the vertexes in this subgraph comprise a *concurrent transmission set*, which includes all the links that can transmit within the same time frame. Then remove all these vertexes and the corresponding edges from the graph, and repeat the process until the graph is empty. In this example, we have the following 5 concurrent transmission sets: {CD, GH}, {DI}, {HI, BC}, {IJ}, {IJ}. The reason why we start forming concurrent sets from the vertexes with least number of edges is because those nodes are the most constrained.

The number of concurrent sets are the minimum number of time frames needed to deliver a packet from each source to the sink. In this example, the result of our heuristic matches the result from the second theorem. One side product of using this heuristic approach is a packet schedule based on the time frame assignment shown in Figure 3(e).

4.2 Estimating Capacity for a Large-Scale Network

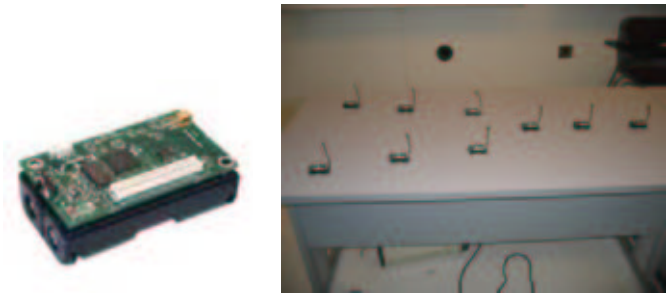
The time complexity of our model is greatly impacted by the number of nodes in the topology. As a result, it may be hard to



(a) two merging flows, each with traffic load T.

(b) a bottleneck link IJ and its bottleneck zone.

Figure 4: Finding a bottleneck zone



(a) Mica2 (MPR400CB)

(b) Experimental sensor field

Figure 5: Mote experimentation

apply our model over a large topology. To deal with this, we take the viewpoint that *the throughput of a topology is limited by the throughput of the bottleneck link(s)*. Therefore, one only needs to calculate the capacity fraction for the portion of the network topology that contains the bottleneck link(s), referred to as bottleneck zone. Extracting the bottleneck zone from a topology involves two steps: identifying the bottleneck link, and then identifying all the links that interfere with it.

Usually, a bottleneck link handles higher traffic loads, and/or experiences higher degree of interference. To capture these two factors, we introduce a term - *congestion sum* of a link - which is the sum of that link's traffic volume and the traffic volumes of all the links that cannot transmit concurrently with this link. The bottleneck link, therefore, has the largest congestion sum. Figure 4(a) shows a simple topology in which the two source nodes, A and E, each generate a traffic of T packets/second. The traffic load of each subsequent link is shown in the figure. To calculate the congestion sum, for each link, we need to identify its interfering links. For example, if B and D transmit at the same time, then C will not receive correctly. For another example, G's data packet may collide with I's ACK packet (to D) at node H. As a result, even under the assumption that the radio interference range equals the communication range, link DI interferes with the following links: BC, CD, GH, HI, IJ, and JK, which are also highlighted in the figure. Adding up the traffic volumes of all the interfering links, one can calculate the congestion sum of a link. Figure 4(b) shows the congestion sum of all the links, and the bottleneck links are IJ and JK. Between these two, we focus on the one closer to the source to infer the capacity fraction of the entire topology. Therefore, we pick IJ and isolate its bottleneck zone, which is shown in Figure 4(b).

After the above step, we can focus on the appropriate bottleneck zone, which is much smaller than the entire topology, and apply our model to calculate the capacity for the bottleneck zone, which is also the capacity of the entire topology. Finally, we would like to point out that we do not need to calculate every link's congestion sum to find the bottleneck link; instead, we can just focus on those links connected to the nodes with a large number of links. This way, the overhead involved in finding bottleneck links will be kept low.

4.3 Capacity of Typical Hot Spot Topologies

The model can be used to estimate capacities for arbitrary topologies, under different system assumptions. Next, we validate our model results against both simulation and actual experimental results, by studying how to increase capacities for string, merging, and crossing topologies. Through these studies, we have learned four valuable lessons, which can guide the design of TARA.

For each topology scenario, we have measured the capacity using the following three methods:

- *Mote experimentation*: The Mica2 motes (MPR400CB) [19, 21] shown in Figure 5(a) are used, running TinyOS v1.0. The motes are carefully placed so that only adjacent ones can communicate with each other as shown in Figure 5(b). The interference range is irregular and may span multiple hops. The raw data rate of Mica2 radio is 38.4Kbps. The motes adopt a simple MAC protocol that samples the channel activity several times before sending a packet.

- *NS-2 simulation*: We configure the NS-2 simulator using the same parameters as in Mica2 motes (i.e. raw radio date rate and packet size). We set up the network in such a way that only adjacent nodes can communicate each other. The interference range is within 2 hops. A simple CSMA MAC protocol, with RTS/CTS, is used. ACKs are sent to acknowledge the reception of packets.

- *Our model*: The capacity from our model has the format of εC^{max}, where ε is the capacity fraction and C^{max} is the 1-hop channel capacity obtained from simulations. In our model, we assume ACKs are used.

We would like to emphasize that our focus is not to match the absolute capacity figures from different methods, which is impractical given the extreme dynamic nature of a wireless experiment. However, we would like to examine whether the trends from different methods agree with each other.

4.3.1 String Topology

In this set of experiments, we study how the capacity of a string topology varies with different hop counts between the source and the sink, i.e. path length l. The results are presented in Figure 6. For the string topology, our analytical model has the following capacity fractions:

$$\varepsilon = \begin{cases} 1 & if \quad l = 1 \\ 1/2 & if \quad l = 2 \\ 1/3 & if \quad l \geq 3. \end{cases} \tag{1}$$

C^{max} is set to the 1-hop throughput obtained using the simulations.

Among the three methods, mote experiments have lower capacities due to the simple MAC protocol and larger degree of interference. However, all three methods demonstrate the similar trend: as l increases from 1 to 3, the capacity quickly drops, but stabilizes as l further increases. Similar observations have also been found in [12], where the authors have pointed out that the capacity fraction of a string topology is constant if local communications dominate and path lengths do not grow arbitrarily with the network dimensions (which is different from the assumption in [7]). Similarly, we have an implicit assumption that path lengths in sensor networks stay more or less constant regardless of the network size, which can be achieved by deploying more sinks as the network size grows.

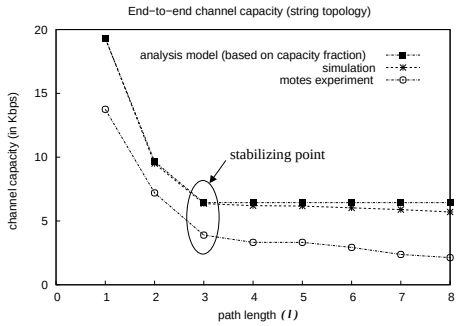


Figure 6: Channel capacity of a string topology.

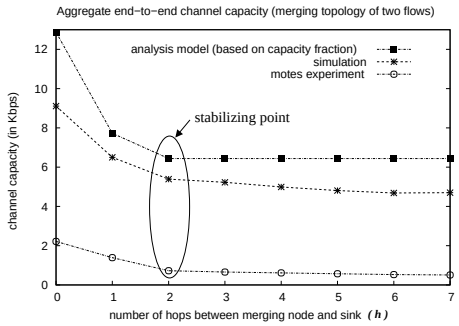


Figure 7: Channel capacity of a 2-flow merging topology

Using the model, we can quickly draw the following conclusion:

Lesson 1 Shortening a string topology does not increase the capacity if the resulting capacity fraction remains the same. □

The results also indicate that a string topology can provide a certain level of channel capacity (denoted as C^{min}) since the capacity faction is stabilized. Therefore, we have the following approach of alleviating congestion:

Lesson 2 If the node whose incoming traffic volume is less than C^{min} experiences congestion due to interference with other flows, the congestion can be eliminated by re-routing the incoming traffic onto the *non-interfered* path. □

4.3.2 Merging Topology

A merging topology is more complicated than a string topology (illustrated in Figure 4), and an n-flow merging case (assuming they merge at the same point) can be characterized by the following $n+1$ parameters: l_1, ..., l_n, and h, where l_i is the path length for the ith flow, while h is the number of hops between the merging point and the sink. Since the merging point is the bottleneck of the topology, h plays a more important role than individual path lengths, especially when each individual flow length is reasonably long.

Our analysis model derives the following capacity fractions:

$$\varepsilon = \begin{cases} n/(n+1) & if \quad h = 0 \\ n/(2n+1) & if \quad h = 1 \\ n/3n = 1/3 & if \quad h \geq 2. \end{cases} \qquad (2)$$

Figure 7 presents the results from the model, simulations and mote experiments for a two-flow merging case. In this set of experiments, we have $l_1 = l_2 = 8$, and we vary h. When two flows merge at the sink, i.e. $h = 0$, we have maximum capacity. Among the three, capacity from our model is the best, and capacity from the mote experiments is the worst. This is because actual experiments have more interference than simulations, which in turn have more complicated interference than the model. Though the absolute numbers are different, the trends from all three methods are the same. When h increases from 0 to 2, the resulting capacity quickly decreases, and it stays constant after h exceeds 2.

From these observation, we have the following lesson to boost the capacity for a merging topology:

Lesson 3 The capacity of a merging topology can be increased by moving the merging point within a small number of hops from the sink. □

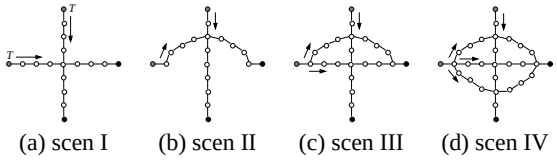


Figure 8: Four scenarios of two crossing flows

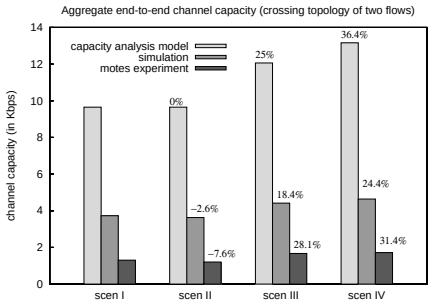


Figure 9: Channel capacity of four scenarios shown in Figure 8

4.3.3 Crossing Topology

A crossing topology consists of multiple flows that have distinct sinks. A two-flow example is shown in Figure 8(a). Unlike the merging case, moving the crossing point, as shown in Figure 8(b), will not increase the capacity. Instead, one may want to have multiple paths for either of the flows, and split the traffic of that flow onto these paths, as shown in Figures 8(c) and (d). These points are further validated by the calculated capacity fractions of these four scenario: $\frac{2}{5}$, $\frac{2}{5}$, $\frac{1}{2}$, and $\frac{6}{11}$. We have measured the capacities of these four scenarios using all three methods, and presented them in Figure 9. The percentage of capacity increase in scenarios II, III, and IV with respect to scenario I are shown on top of each bar. From the results, we observe that the capacity of the crossing topology can be increased by having multiple paths in either flow. This is because the resulting incoming traffic volume at each crossing node is reduced, which are $2T$, $2T$, $\frac{3}{2}T$, and $\frac{4}{3}T$ in the above four scenarios, when each source generates a traffic volume of T. Therefore, we have the following conclusion:

Lesson 4 To increase the capacity of a crossing topology, at least one flow should have multiple paths, and split its traffic onto these paths. □

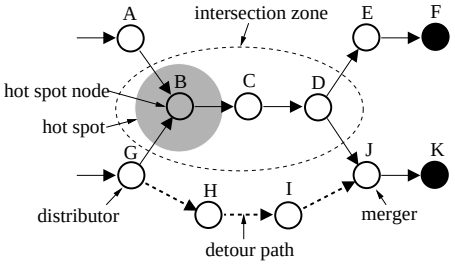


Figure 10: Illustration of TARA

5. TOPOLOGY-AWARE RESOURCE ADAPTATION SCHEME

Building upon the capacity analysis tool, we develop TARA, a topology-aware resource adaptation protocol, which can adapt network resources based on the congestion level. Hence, it can serve the dual purposes of alleviating congestion during crisis states and conserving energy during dormant states.

To prevent "thrashing" effect due to frequent resource adaptations, TARA uses watermarks to control the adaptation frequency: it starts to increase resources when congestion level hits the specified upper watermark, and shrink resources when congestion level hits the lower watermark.

5.1 Increasing Resources During Crisis States

The framework of TARA is illustrated in Figure 10, where the hot spot is around node B. As soon as the hot spot node detects its congestion level is above the upper watermark, it needs to quickly locate two important nodes: the distributor (G) and the merger (J). Then a detour path can be established, starting at the distributor and ending at the merger. As suggested by their names, the distributor distributes the incoming traffic between the original path and the detour path, while the merger merges these two flows. The choice of these two nodes, the establishment of the detour path, and the distribution of loads between the two paths, are challenging research topics.

5.1.1 Congestion Detection and Hot spot Node

Congestion detection in sensor networks has been studied in [9, 19]. As pointed out in both papers, the actual congestion level around a node cannot be accurately reflected by the buffer occupancy alone, which is a popular metric in traditional wired networks [1, 4, 5]. Therefore, TARA measures not only the buffer occupancy but also the channel loading [9, 19]. A node's congestion level is an aggregation of these two metrics. As soon as the congestion level hits the upper watermark, it declares congestion, and becomes a *hot spot node*.

5.1.2 Traffic Distributor

In order to select the traffic distributor, every sensor node keeps track of the incoming traffic volume from each of its neighbors. This can be achieved by adding one more column in the neighbor table, a data structure that is already maintained by each node in most sensor networks. Using this information, the hot spot node can select the upstream neighbor that injects the most packets, and send an *upstream control packet* to that neighbor. If that neighbor node is also congested, it repeats the process until it reaches a node with a low congestion level. That node will then become the traffic distributor.

5.1.3 Traffic Merger

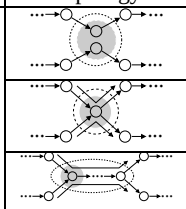
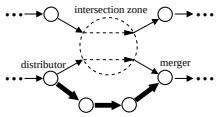
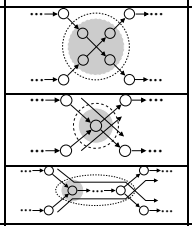
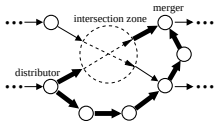
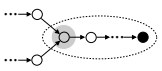
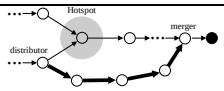
Intersection Zone	Congestion Topology	Traffic Distribution	Related Lessons
Braided			2
Crossing			4
Merging			1,3

Table 1: Three types of intersection zones

Starting from the distributor, we trace downstream by sending a *downstream control packet* to find a suitable traffic merger. Among all the sinks that are connected to the distributor, we choose the one that absorbs the most traffic. Then the merger should be located on the routing path from the distributor towards this sink, with a low congestion level. We would like to point out that congestion level should not be the sole criteria for choosing a merger, but a node's location is also important. For instance, in the example shown in Figure 10, nodes C, D, and J may all have low congestion levels as many packets get dropped at node B. However, in this particular example, only J is qualified to be a merger: building a detour path connecting G and C (or G and D) does not help distribute the traffic, as C handles exactly the same packets as B does, and will become the new hot spot node. Therefore, the choice of the merger is dependent on the topology of the *intersection zone*, which includes all the nodes the two intersecting flows have in common on their routing paths. Table 1 gives a detailed summary of possible topologies for interference zones for two flows (the second column), and illustrates the possible detour path topologies and traffic distribution (the third column) and the related lessons which corresponding resource controlling measures are based on (the fourth column).

Selecting a merger for a braided intersection zone is similar to selecting a merger for a crossing intersection zone in that the merger has to be located after the two flows split. As a result, those nodes that forward packets to both sinks are not qualified to be a merger, and they should pass the control packet further downstream.

In the merging intersection zone, the above-mentioned problem is not an issue any more since every node on the routing path only serves one sink, but the location of the merger is still of critical importance because it bears a great impact on the resulting capacity based on lesson 3. In order to apply this lesson, each node on the routing path keeps track of the hop count between itself and the sink, which is readily available from the routing table. Therefore, as the downstream control packet reaches a node, which contains the desired capacity level based on the traffic rate at the distributor, the node can decide whether to become a merger based on its distance to the sink. For instance, if the desired capacity exceeds the

resulting capacity by this node becoming the merger, then it should just forward the control packet to the next downstream node on the routing path which is closer to the sink.

In parallel with the effort of tracing the merger, we also need to notify the distributor's ID to the merger so that the merger can establish a detour path. In the presence of heavy congestion, the distributor needs to embed its own ID in every packet that it sends to ensure higher reliability.

5.1.4 Detour Path

The merger tries to establish the detour path by locally flooding a REQ packet including the time-to-live (TTL) field towards the distributor. A node may receive multiple REQ packets from a merger due to the nature of flooding. In order to limit the flooding overhead, we have adopted the following optimizations:

- A node discards all the REQ messages if its local congestion level is above a threshold.
- A node discards all the REQ messages if it is already on the original routing path. This information is usually made available from the routing table.
- A node decrements the TTL value of the packet before forwarding, and discards the packet if the TTL value falls below 0.
- A node keeps track of the largest TTL value it has seen. It drops the REQ messages whose TTL values are lower because these messages have traveled a longer path to reach this node.

When the REQ message reaches the distributor, a candidate detour path is established. Usually, the distributor will receive more than one candidate detour path, and it chooses the one whose REQ message has the largest TTL value, which corresponds to the smallest path length. To break the tie, the REQ message also records the path congestion level, which is the highest congestion level among all the nodes on the candidate detour path, and the distributor chooses a path with a lower congestion level by sending an ACK message towards the merger through the selected path.

5.1.5 Traffic Distribution

To alleviate congestion, the distributor should split the outgoing traffic between the original path and the detour path. It should be careful in this process because some packets should not be routed via the detour path. In the example illustrated in Figure 10, the distributor, G, may need to forward packets to both sinks, and packets that are destined to sink F should not be assigned to the detour path because it is costly for the merger (J) to connect to F. To address this challenge, we introduce the concept of *streams*. As far as an intermediate forwarding node is concerned, all the flows that are destined to the same sink belong to the same stream. After introducing this concept, we propose a stream-based traffic distribution strategy. This strategy requires the distributor checks each packet's destination sink before routing it. Each detour path has a corresponding sink, and if a packet's destination sink does not match the detour path sink, the distributor will only send that packet to the original path.

Among the packets that have the matching sink, we adopt the *weighted fair-share scheduling* to split traffic between two routes. Suppose the original path's congestion level is l_1, and the detour path's congestion level is l_2, with $0 < l_1 < 1$, and $0 < l_2 < 1$. Then the traffic rates for these two paths, t_1 and t_2, should follow $\frac{t_1}{t_2} = \frac{l_2}{l_1}$. To further understand this algorithm, let us look at the detour topologies shown in Table 1. Among the three types of intersection zones, the detour path in the crossing case has higher congestion level than the detour path in the other two cases, and as a result, the detour path in the crossing case takes a smaller share of traffic compared to the detour paths in the other two types of zones.

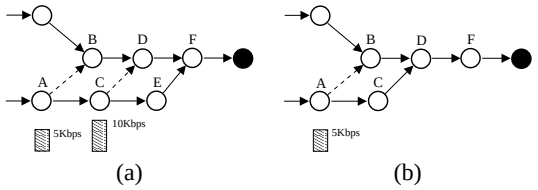


Figure 11: Shrinking resource provisioning using lower watermark

5.2 Shrinking Resources During Dormant States

During crisis states, TARA tries to quickly increase resources to alleviate congestion. Similarly, after the crisis state is over, a sensor network must go to a low resource level to conserve energy. This seamless adaptation can be ensured by TARA through closely watching the congestion level of the network. Many sensor applications have comparable data rates during dormant periods, and hence, for the purpose of decreasing resources, we can simply let the network stop using the detour path and go back to the original path. As a result, each distributor needs to remember which neighbor is on the original path, no matter how many detour paths it has created.

Figures 11 (a) and (b) illustrate an example scenario and how TARA deals with the traffic change. The original path connecting A and F is (A-B-D-F). As the network enters a crisis state, the data rate at node A becomes as high as 5Kbps, resulting in congestion around node B. As a result, a detour path (A-C-D) is constructed to ease congestion. However, as the underlying events keep evolving, the data rate continues to go up to 10Kbps, and D becomes the new hot spot node. To alleviate the new congestion, C builds a detour path (C-E-F), and congestion subsides. In this process, both A and C keep track of their neighbors on the original paths. If the traffic rate gets below 10 Kbps after some time, C observes that its congestion level is below the lower watermark. Then, it stops distributing the traffic, but sends all the packets to the previous path (to D), as shown in Figure 11 (b). The detour path (C-E-F) is destroyed since it is based on "soft state" – it will expire after a certain period of inactivity.

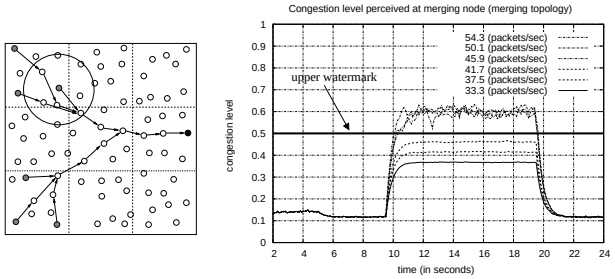
6. PERFORMANCE EVALUATION

In this paper, we have conducted a detailed simulation-based study to compare TARA and several other strategies that cope with congestion in sensor networks. The results show that TARA clearly outperforms others in satisfying application fidelity requirements while spending less energy for each packet that is delivered from source to sink.

6.1 Performance Metrics

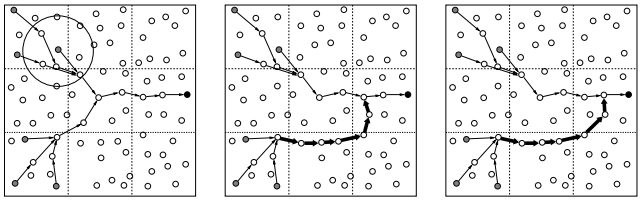
In this study, we propose to measure the effectiveness of TARA using the following metrics:

- *Fidelity Index*: The primary goal of TARA is to satisfy application fidelity requirements. If the application intends to receive F^o packets per time unit, and a congestion control strategy actually delivers F packets per time unit to the applications (across all the sinks to which this application is



(a) sensor field (b) traffic model

Figure 12: Simulation setup



(a) (b) (c)

Figure 13: Topologies for various congestion control strategies in the case of merging intersection: (a) no congestion control and traffic control; (b) topology-unaware rc; and (c) ideal rc and TARA.

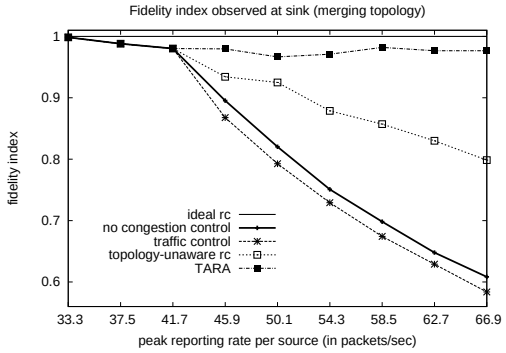


Figure 14: Fidelity index during the crisis period in the merging congestion scenario.

connected), then we define the fidelity index of this strategy as the ratio of F/F^o.

- *Total Energy Consumption*: The total energy consumption, E^{total}, is represented as follows:

$$E^{total} = \sum_{i=1}^{n} (p^{xmit} \times t_i^{xmit} + p^{recv} \times t_i^{recv} + p^{idle} \times t_i^{idle}), \quad (3)$$

where p^{xmit}, p^{recv}, and p^{idle} are the power consumption in transmission, reception and idle modes respectively.

- *Bit Energy Consumption*: This metric is the ratio of the total energy consumption with respect to the total number of bits successfully delivered to the sink(s). It measures how effective is the energy usage from the end application's point of view.

6.2 Simulation Environment

First, we introduce the network model in our simulation studies. A sensor network has two main functionalities: sensing (data collection) and networking (data delivery), and our network model addresses both of these. We have 81 sensor nodes that are uniformly randomly distributed over a 160x160m^2 field. The radio has a communication range of 30m and an interference range of 50m. On average, each node has about 9 neighbors, resulting in a rather dense network with a reasonable degree of redundancy. As far as sensing is concerned, we partition the entire field into 9 grids, as shown in Figure 12(a), and all the sensors that belong to the same grid can detect any event within that grid.

We use the NS-2 simulator in our studies, and we have tuned many parameters according to the parameters of an actual Mica2 radio [23]. Each packet is 100 byte long, and each node can hold at most 10 packets in its outgoing buffer. A node consumes 13.5mW, 13.5mW, and 24.75mW in the idle listening, receiving, and transmitting mode, respectively. The MAC protocol is 2Mbps 802.11 DCF. Upon collision, a packet will have up to 7 retransmissions. To exclude the impact of a routing protocol, the initial routing topology is hard-wired.

In our simulations, we focus on two types of intersection hot spots: merging and crossing, as shown in Figures 1(c) and (d). In particular, we have considered the following traffic pattern to simulate the lifetime of a sensor network. At the beginning of the lifetime, the network is in its dormant period, and the offered load is one packet per second per source. After a random amount of time, the network enters the crisis state, and the crisis lasts for 10 seconds. In our studies, we vary the traffic rate during the crisis state from 33.3 to 66.9 packets/second to result in congestion with different severity levels. Figure 12(b) plots the resulting congestion levels at the intersection node with increasing traffic rate. In our simulations, the upper watermark is set to 0.5 unless otherwise noted, and therefore, Figure 12(b) shows that TARA will be triggered when traffic rate reaches around 45.9 packets/second in the merging topology.

In this study, we compare the following five strategies that can be adopted during a crisis period:

- *No congestion control*. This represents the baseline scenario in which no congestion control is performed.
- *Traffic control*. In this case, when a node detects congestion, it sends a back pressure message to the upstream nodes on the routing path so that they can reduce the traffic load. In the simulations, we have carefully tuned the parameters, such as the frequency of back-pressure messages, to ensure it delivers the best possible fidelity index.
- *Ideal rc*. This corresponds to an optimal offline resource control algorithm. Given a certain traffic load, this algorithm always finds the minimum topology for it. Though it can not be implemented in a real system, we include this algorithm to investigate the gap between TARA and an optimal algorithm.
- *Topology-unaware rc*. This corresponds to an ad-hoc resource control algorithm. Specifically, it chooses the first downstream node with a low congestion level as the merger to form the detour path, and blindly routes all the packets to the detour path.
- *TARA*. TARA is a topology-aware resource control algorithm.

6.3 Simulation Results for the Merging Intersection

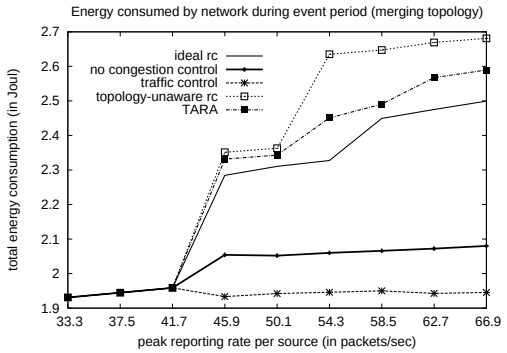


Figure 15: Total energy consumption by network during the crisis period in the merging congestion scenario.

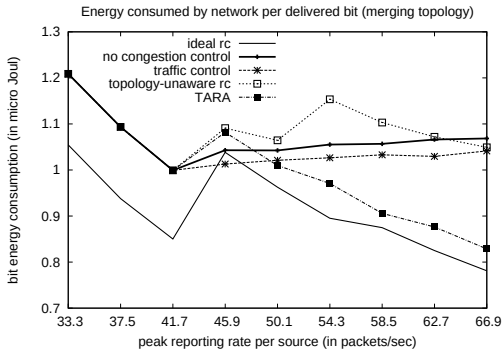


Figure 16: Bit energy consumption during the crisis period in the merging congestion scenario.

In the merging scenario, six sources report to one sink, three of them in the upper left grid, and the other three in lower left grid. The topology is illustrated in Figure 13(a). In the simulation, topology-unaware rc selects the first downstream node whose congestion level is below the threshold as the merger (the resulting topology is shown in Figure 13(b)), and routes all the packets to the detour path. On the other hand, TARA can construct the same detour path as ideal rc, which is shown in Figure 13(c). As shown in Lesson 3, 1 hop here can make a big difference on the capacity faction.

Figure 14 plots the fidelity index under different offered load for each strategy. When the load is lower than the network capacity, the fidelity index is closer to 1. After the load exceeds the capacity, the fidelity index for no congestion control degrades significantly. Traffic control shows the similar trend, but the fidelity index is even worse because it incurs additional overhead to control the traffic volume. On the other hand, the three resource control schemes produce much better fidelity indexes. Ideal rc has a fidelity index of 1 at all load ranges. Under the topology-unaware rc scheme, since the construction of the detour path ignores the network topology, the resulting capacity is not enough to satisfy high traffic loads. As a result, though it performs much better than no congestion control and traffic control, the gap between this scheme and ideal rc is still large. However, as shown in the figure, TARA, due to its topology awareness, results in a fidelity index of 0.97 on the average regardless of the offered load level, close to ideal rc. The gap between TARA and the ideal case is caused by the latency for TARA to establish a new topology, though both strategies end up having the same detour paths.

Figure 15 shows the total energy consumption of the network. When the offered load is below network capacity, the total energy consumption increases slowly with the load. To explain the sub-linear increase, let us look at the total energy consumption E, which can be calculated as the sum of the energy consumption in transmission, reception and idling states, i.e. $E = p^{xmit} \times t^{xmit} + p^{recv} \times t^{recv} + p^{idle} \times t^{idle}$. When the load is low, a sensor node spends more time idling than transmitting; given the fact that transmission power and idling power are close to each other, idling energy actually dominates the total energy consumption. As a result, as the traffic load increases, the increase in total energy consumption is rather modest. After the load is higher than the capacity, however, the total energy consumption increases dramatically. As expected, the traffic control scheme consumes the least amount of energy because fewer packets are transmitted. In the case of no congestion control, the energy consumption increases considerably as congestion first occurs because many packets now need multiple transmissions, but after some time, the total energy consumption stabilizes because the system has reached the maximum energy budget of the current topology. Resource control schemes consume more energy because more nodes that previously were turned off become actively involved in packet transmissions and many more packets are delivered to the sink, as shown earlier. However, the amount of extra energy budget by resource control strategies is rather modest compared to the number of additional packets they can handle: at the traffic load of 66.9 packets/second, TARA can deliver (398-234)/234 = 70% more packets than traffic control schemes, by consuming (2.589-1.945)/1.945 = 33% more energy. Among the three resource control schemes, ideal rc consumes least energy. Topology-unaware rc spends the most energy because congestion is not fully eliminated so that many packets still incur multiple transmissions (in the figure, the total energy consumption significantly increases when a new detour path is created). However, we proudly point out that TARA's energy consumption is only marginally higher than that of ideal rc, i.e. 4% most of the times.

Figure 16 shows the bit energy consumption when varying the traffic load. Before congestion occurs, the bit energy consumption decreases with the traffic rate. This is because within this range, the total energy consumption only increases marginally with the traffic load. During light congestion (with a traffic load around 45.9 packets/sec), the bit energy consumption of three resource control schemes are larger than the other schemes because the created detour path is not yet fully utilized. The reason why this happens even in ideal rc is that the network capacity is not always a continuous function. As traffic rate further increases, the bit energy consumption in TARA and ideal rc significantly decreases because the detour path gets better utilized. Again, the bit energy consumption of TARA is only slightly more than that of ideal rc. Whether to enforce the resource controlling during light congestion is dependent on the application requirement. To balance the bit energy consumption and fidelity index, one can carefully decide when to enable resource controlling methods by configuring the watermark values appropriately.

6.4 Simulation Results for Crossing Intersection

In this section, we study the performance of different congestion control strategies in a crossing scenario. In our experiments, we have 6 sources and two sinks, each sink connected to three sources, as shown in Figure 17(a). These two streams cross each other in

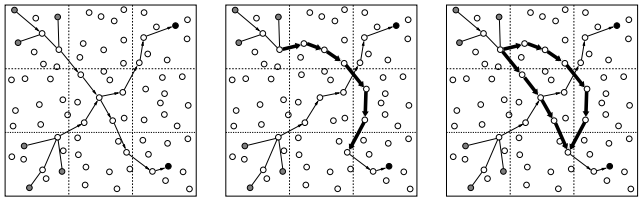


Figure 17: Topologies for various congestion control strategies in the case of crossing intersection: (a) no congestion control and traffic control; (b) topology-unaware rc; and (c) ideal rc and TARA.

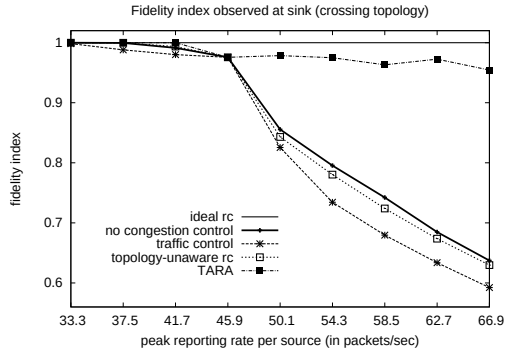


Figure 18: Fidelity index during the crisis period in the crossing congestion scenario.

the center of the sensor field. In this case, all three resource control strategies will find the same merger and establish the same detour path, but the topology-unaware rc blindly forwards all the packets to the detour path as shown in Figure 17(b), while the other two split them between the original path and the detour path as shown in Figure 17(c).

Figure 18 shows the fidelity indexes for all five strategies. The trend is similar to what we have observed in the merging case, with the exception that topology-unaware rc now performs the worst. This is because the topology-unaware rc simply forwards all the packets to the detour path, which cannot alleviate congestion but only shift the hot spot. What makes matters worse is that it will result in the "ping-pong" effect in which a new detour path continues to be created. On the other hand, TARA effectively accommodate the incoming traffic by splitting them onto the two paths at the same time.

Figures 19 and 20 show the total energy consumption and bit energy consumption for the five strategies. The trend in the crossing case is similar to that in the merging case, except that the topology-unaware rc strategy performs much worse here. The bit energy consumption for topology-unaware rc is significantly higher because it does not alleviate congestion at all though a large amount of resources are provided.

6.5 Discussion: Resilient to Transient Congestion

Setting the appropriate value for the upper watermark is a tricky issue. Ideally, we would like it to be small enough so that TARA can quickly respond to congestion. A small upper watermark, however, will make the network unstable: a short "glitch" may trigger the algorithm, and the glitch may disappear even before the detour path is established, causing much energy to be wasted. In order

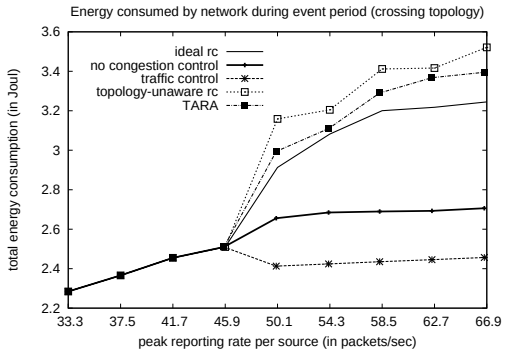


Figure 19: Total energy consumption by network during the crisis period in the crossing congestion scenario.

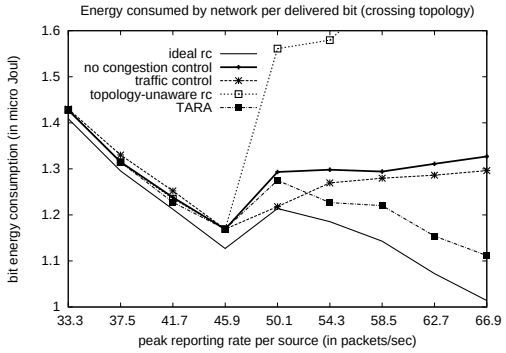


Figure 20: Bit energy consumption during the crisis period in the crossing congestion scenario.

to allow quick recovery from real congestion without responding to glitches, we introduce an application-specific parameter, *transient threshold*, to filter out glitches. Using this threshold, resource increasing methods will only take effect when the instantaneous congestion level has been above the upper watermark for a certain period of time. As a result, we can make the upper watermark small without undermining the stability of the network. We next conduct an experiment to illustrate our point. Figure 21 shows the bit energy consumption with different upper watermark values. In this experiment, we considered a merging intersection case with 6 sources. The sources first generate a transient traffic surge which lasts 0.9 second, and then 3 seconds later, a crisis period that lasts 10 seconds. The average congestion level during the transient traffic surge is 0.32. If the transient threshold is not used, then the upper watermark of 0.35 results in the least bit energy consumption since it can ignore the transient congestion and react to the persistent congestion early. However, when the transient threshold of 1 second is used, the transient surge is filtered out and the bit energy consumption is lowest when the upper watermark is 0.3. Clearly, the introduction of the transient threshold can lead to a smaller upper watermark, and more importantly, the bit energy consumption is lower than that of no transient threshold case.

7. RELATED WORK

Most of prior work have focused on traffic control. A guideline of congestion control in sensor networks was first given in [16]. The authors suggest that congestion control must be based on not only the network capacity, but also the application fidelity require-

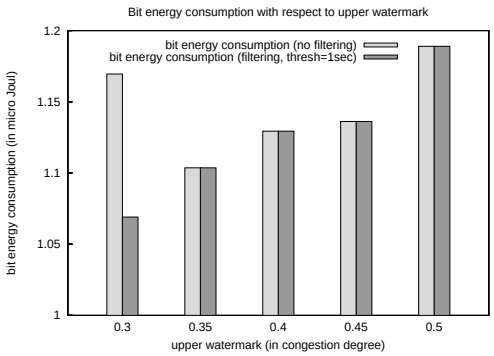


Figure 21: Bit energy consumption during the crisis period.

ments. CODA [19] presents the first detailed study on congestion detection and avoidance in sensor networks. In CODA, as soon as a node detects congestion, it broadcasts a backpressure message upstream. An upstream node will thus throttle the traffic volume. Sankarasubramaniam [15] proposed an event-to-sink reliable transport (ESRT) protocol, which can serve as a congestion control protocol. In ESRT, the sink reduces the traffic of all sources during congestion. In [8], Hull studied three congestion control techniques: hop-by-hop flow control, limiting source rate, and a prioritized medium access control (MAC). Ee and Bajcsy [3] proposed a distributed congestion control scheme based on hop-by-hop automatic repeat request (ARQ) in many-to-one routing scenario. Woo and Culler [21] proposed to alleviate congestion by assigning bandwidth proportionally. Yi and Shakkottai [25] proposed a hop-by-hop congestion control scheme that allocates bandwidth to various users in a fair manner. They show a hop-by-hop traffic control scheme push-backs and spreads congestion over space, leading to scattered small peak loads.

Resource control has received little attention. In [10, 17], the authors propose a congestion adaptive routing scheme in ad hoc networks. Due to their topology-unaware path creation, however, these schemes cannot effectively alleviate congestion in sensor networks. Several multipath routing protocols [6, 11, 13, 18, 24] are developed in the context of reliability, load balancing, failure recovery rather than congestion control.

8. CONCLUDING REMARKS

The special nature of sensor networks calls for a new approach to alleviating congestion that can satisfy the application fidelity requirements. Therefore, TARA, a topology-aware resource adaptation strategy was designed, and its performance was experimentally evaluated. TARA offers several advantages: (1) it is topology-aware, (2) it is energy-efficient, and (3) it is distributed. TARA uses a capacity analysis model to determine the needed topology. This model is formulated using a graph coloring problem. The model results are compared against simulation (NS-2) and experimental results (using Mica2 motes). Detailed simulation results have shown that TARA can absorb incoming traffic load. The results have also demonstrated TARA performs very close to an ideal offline resource control algorithm, in terms of both fidelity satisfaction and energy conservation.

9. REFERENCES

[1] S. Athuraliya, V. H. Li, Steven H. Low, and Qinghe Yin. REM: Active Queue Management. *IEEE Network*, May 2001.

[2] D.-M. Chiu and R. Jain. Analysis of the increase and decrease algorithms for congestion avoidance in computer networks. *Computer Networks and ISDN Systems 17(1):1-14*, 1989.

[3] C. T. Ee and R. Bajcsy. Congestion control and fairness for many-to-one routing in sensor networks. In *Proceedings of ACM SenSys*, Nov. 2004.

[4] S. Floyd and K. Fall. Promoting the Use of End-to-End Congestion Control in the Internet. *IEEE/ACM Transactions on Networking*, Aug. 1999.

[5] S. Floyd and V. Jacobson. Random Early Detection Gateways for Congestion Avoidance. *IEEE/ACM Transactions on Networking*, Aug. 1993.

[6] D. Ganesan, R. Govindan, S. Shenker, and D. Estrin. Highly-Resilient, Energy-Efficient Multipath Routing in Wireless Sensor Networks. *ACM Mobile Computing and Communications Review, Volume 5, Issue 4, pp. 11-25*, 2001.

[7] P. Gupta and P. R. Kumar. The Capacity of Wireless Networks. *IEEE Transactions on Information Theory, 46(2):388-404*, March 2000.

[8] B. Hull, K. Jamieson, and H. Balakrishnan. Mitigating congestion in wireless sensor networks. In *Proceedings of ACM SenSys*, Nov. 2004.

[9] J. Kang, Y. Zhang, and B. Nath. Accurate and energy-efficient congestion level measurement in ad hoc networks. In *Proceedings of IEEE WCNC*, Mar. 2005.

[10] S. J. Lee and M. Gerla. Dynamic Load-Aware Routing in Ad hoc Networks. In *Proceedings of IEEE ICC*, June 2001.

[11] S. J. Lee and M. Gerla. Split Multipath Routing with Maximally Disjoint Paths in Ad hoc Networks. In *Proceedings of IEEE ICC*, June 2001.

[12] J. Li, C. Blake, D. S. J. D. Couto, H. I. Lee, and R. Morris. Capacity of Ad Hoc Wireless Networks. In *Proceedings of ACM/IEEE MobiCom*, July 2001.

[13] M. K. Marina and S. R. Das. On-demand Multipath Distance Vector Routing in Ad Hoc Networks. In *Proceedings of International Conference for Network Protocols (ICNP)*, November 2001.

[14] S. Ratnasamy, D. Estrin, R. Govindan, B. Karp, S. Shenker, L. Yin, and F. Yu. Data-centric storage in Sensornets. In *the first ACM International Workshop on Wireless Sensor Networks and Applications (WSNA)*, 2002.

[15] Y. Sankarasubramaniam, O. Akan, and I. Akyildiz. ESRT: Event-to-Sink Reliable Transport in Wireless Sensor Networks. In *proceedings of the ACM MobiHoc Conference*, 2003.

[16] S. Tilak, N. B. Abu-Ghazaleh, and W. Heinzelman. Infrastructure tradeoffs for sensor networks. *ACM Workshop on Wireless Sensor Networks and Applications (WSNA)*, Sep. 2002.

[17] D. A. Tran and H. Raghavandra. Routing with congestion awareness and adaptivity in mobile ad hoc networks. In *Proceedings of IEEE WCNC*, Mar. 2005.

[18] A. Valera, W. K. G. Seah, and SV Rao. Cooperative Packet Caching and Shortest Multipath Routing in Mobile Ad hoc Networks. In *Proceedings of IEEE INFOCOM*, March 2003.

[19] C. Wan, S. Eisenman, and A. Campbell. CODA: Congestion Detection and Avoidance in Sensor Networks. In *proceedings of the ACM SenSys*, 2003.

[20] R. J. Wilson. *Introduction to Graph Theory, fourth edition*. Longman Group Ltd., 1996.

[21] A. Woo and D. Culler. A Transmission Control Scheme for Media Access in Sensor Networks. In *Proceedings of the Seventh Annual ACM/IEEE MobiCom*, July 2001.

[22] T. Yan, T. He, and J. A. Stankovic. Differentiated Surveillance Service for Sensor Networks. In *Proceedings of the ACM SenSys 2003*, 2003.

[23] W. Ye, J. Heidemann, and D. Estrin. An Energy-Efficient MAC Protocol for Wireless Sensor Networks. In *Proceedings of IEEE INFOCOM'02*, June 2002.

[24] Z. Ye, S. V. Krishnamurthy, and S. K. Tripathi. A Framework for Reliable Routing in Mobile Ad Hoc Networks. In *Proceedings of IEEE INFOCOM*, March 2003.

[25] Y. Yi and S. Shakkottai. A Hop-by-hop Congestion Control over a Wireless Multi-hop Network. In *Proceedings of IEEE INFOCOM'04*, March 2004.