JiTS: Just-in-Time Scheduling for Real-Time Sensor Data Dissemination[*]

Ke Liu, Nael Abu-Ghazaleh and Kyoung-Don Kang
Computer Science Dept., SUNY Binghamton
{kliu,nael,kang}@cs.binghamton.edu

Abstract

Most existing real-time protocols for sensor data dissemination use packet scheduling schemes to prioritize packets according to their deadlines. However, packet prioritization by itself cannot completely support real-time data dissemination requirements. In this paper, we propose new Just-in-Time Scheduling (JiTS) algorithms that take advantage of the available slack, if any, to reduce contentions and improve real-time performance by judiciously delaying packets as long as their deadlines are not missed. Specifically, we explore several policies for allocating the slack among multiple hops, including a non-linear policy where packets are non-uniformly delayed at intermediate nodes to account for expected higher contentions as packets get closer to the sink(s). Notably, our JiTS policies require neither lower layer support nor synchronization among sensor nodes making for an easy deployment. In our simulation study, JiTS significantly improves the deadline miss ratio and packet drop ratio compared to existing approaches in various situations. It is also shown that the Geographic Forwarding often used for real-time data dissemination substantially underperforms the Shortest Path routing especially when the load is high.

1 Introduction

Existing solutions for real-time data dissemination in sensor networks [1] prioritize packets at the MAC layer according to their deadlines and distances to the sink. These work have several limitations including: (1) While packets are prioritized, they are not delayed. When traffic is bursty, high contention may result increasing transmission and queuing delays; and (2) MAC level solutions cannot account for the queuing delay in the routing layer (occurring above the MAC layer) that has a significant impact on end-to-end delay especially under high loads. In addition, the role of the routing protocol in the success of real-time data dissemination is not sufficiently examined. Geographic Forwarding, used in most real-time data dissemination studies [1, 2, 3, 4], does not always use the shortest path possibly increasing the deadline miss ratio.

The primary contribution of this paper is a **Just-in-Time Scheduling (JiTS)** approach for real-time data dissemination in sensor networks. JiTS delays packets at every hop for a duration of time which is a function of the number of hops to the sink and the deadline. Unlike existing solutions, JiTS incorporates a full estimate of the delay including queueing delays at the network layer. Further, it distributes the available slack time to allow the network to tolerate transient periods of high contentions.

We compare our approach primarily to the RAP real-time architecture [1] that uses a Velocity Monotonic Scheduling (VMS) algorithm to prioritize packets. VMS computes packet "velocity", which serves as its priority, as the ratio of end-to-end distance to the time until the deadline. In the Static VMS, velocity is computed once at the source, while it is recomputed at each node in the Dynamic VMS. A packet falling in one of three velocity ranges is queued into the corresponding queue in a FIFO manner, while fixed priority is enforced among the three queues. RAP also modifies the MAC layer back-off scheme to schedule packets according to their priority.

The SPEED framework [2, 3] proposes a routing based approach to real-time transmission. To provide soft real-time guarantees, SPEED uses a MAC layer estimate of the one-hop transmission delay to select the next hop to forward the data packet to. However, SPEED does not prioritize, delay or reorder packets.

Just-in-Time scheduling was used in Mobicast[5]−a spatio-temporal multicast scheme in sensor networks. Their goal is reliable message delivery to mobile delivery zones on top of a random network topology. Just-in-Time Delivery in mobicast is only used to minimize the number of data packet copies at different nodes in the multicast zone. As a result, the network-wide storage-time footprint can be saved. However, Mobicast does not consider packet prioritization or routing to meet real-time deadlines.

The design of JiTS is discussed in detail in Section 2.

[*]This paper was partially supported by AFRL grant FA8750-05-1-0130, and NSF grant CNS-0454298.

In Section 3, the performance of JiTS is compared to the RAP [1] under different deadline constraints and routing protocols. JiTS outperforms RAP in terms of both deadline miss ratio and packet drop ratio. Especially, the nonlinear version of JiTS is able to achieve the best performance among the tested approaches. These results hold for regular and random topologies under different network traffic patterns. In particular, when the traffic is generated in a bursty manner, JiTS significantly outperforms RAP. Finally, in Section 4, we conclude the paper and discuss future work.

2 Just-in-Time Scheduling Framework

In this section, our Just-in-Time-Scheduling (JiTS) algorithms for real-time data dissemination, in which sensor data are forwarded to and gathered at the sink, are described. In JiTS, each node decides how long to delay a packet, while leaving sufficient time to meet the deadline. For packet forwarding, the JiTS scheduler uses a single priority queue based on the computed target transmission time. The packet with the earliest target transmission time is inserted to the head of the queue. The first packet in the queue is transmitted when its target transmission time is reached. When the queue is full, the JiTS scheduler immediately pass the packet at the head of the queue to the MAC layer instead of dropping packets in the queue.

The end-to-end delay can be divided into two parts: *lower layer transmission delay*, called End-to-End Transmission Delay (EETD), which is the estimated end-to-end delay due to channel contentions and packet transmissions occurring below the network layer and *queuing delay* taken in the network-layer queues at intermediate nodes. To estimate the EETD, we periodically measure the one-hop Estimated Transmission Delay (ETD) by measuring the time between packet transmission and acknowledgment. This estimate is extrapolated linearly using the ratio of the remaining end-to-end distance to the current hop distance. Since the queuing delay generally dominates the transmission delay in a heavy traffic environment, a precise EETD is not necessary. Further, scheduling cannot directly affect it, since this time is spent below the network layer. The second component is the queuing delay, i.e., the delay for which the packet is queued at the network layer before being handed to the MAC layer for transmission. This delay more critical under overload can be directly controlled by the JiTS scheduler.

In the basic JiTS algorithm, the target transmission time is set to be equal at all intermediate hops as follows:

$$Target\ Delay = \frac{Deadline - EETD}{Distance(X, sink)} \cdot \alpha \qquad (1)$$

where $Deadline$ is the end-to-end deadline between a source and sink, and α is a constant "safety" factor used to ensure that the real-time deadline would be met. When $\alpha = 0.7$, for example, the target delay of the packet will be set to 0.7 times the available slack time, leaving the remaining time as a safety margin.

Different JiTS scheduling policies can be developed by varying the approach to allocating the available slack time among the intermediate nodes. In this paper, we consider the three variations in the following.

1. Static JiTS (JiTS-S): In JiTS-S, the target delay is set at the data source. The end-to-end deadline in Equation 1 is fixed at the data source. Further, we set X in Equation 1 to be the data source. However, we estimate the EETD using the ETD of the current forwarding node and the distance from the source to the sink. Thus, even though we call it static, different ETD's at intermediate forwarding nodes would make the target delays at those nodes different.

2. Dynamic JiTS (JiTS-D): In JiTS-D, the target delay is reset at each forwarding node with local values. The end-to-end deadline (Equation 1) of a packet at some forwarding node is set to the *remaining slack time*, i.e., $Deadline - Elapsed\ Time$. The EETD is decided by the one-hop ETD of the forwarding node and the distance from it to the sink instead of the distance from the source to sink. Hence, the dynamic JiTS is able to continuously readjust the priority of data packets, if necessary, to meet their deadlines.

3. Non-linear JiTS (JiTS-NL): In this approach, we non-uniformly allocate the available slack time among the intermediate hops along the path to the sink. Specifically, we provide the packets with additional slack time as they go closer to the sink, since the contention is usually higher as a packet moves closer to the sink in a data gathering application. Thus, the target delay at a forwarding node is allocated as follows.

$$Target\ Delay = \frac{Deadline - EETD}{2^{R/O}} \cdot \alpha \qquad (2)$$

where R and O are the remaining distance to the sink and average one hop distance. More generally, we may want to allocate the slack time proportionately to the degree of contention along the path. Such a heuristic may be developed by passing the contention information along with the routing advertisement and allocating the available slack time accordingly. A thorough investigation is reserved as future work.

JiTS can be adapted to work with virtually any underlying routing protocol. However, the JiTS algorithm may need to be adapted to consider the cost metric used by the routing algorithm. For example, in a system based on the shortest path routing (SP), the distance is measured in the number of hops.

3 Experimental Study

We have implemented the Static, Dynamic and Non-Linear JiTS with both Shortest Path (SP) routing and Geographic Forwarding (GF) in the Network Simulator (NS2, version 2.27) [6]. We have also implemented the RAP Velocity Monotonic Scheduling (VMS) with GF, including the specialized MAC support following the original specification [1]. Since GF has significantly outperformed traditional ad hoc routing protocols, such as DSR [7], we restrict the routing comparisons to GF and SP.

Unless otherwise indicated, we use scenarios with 100 sensors and investigate both 10×10 grid and random deployments. We use IEEE 802.11 with a bandwidth of 2Mbps and transmission range of 250 meters. Sensors report data at a rate of 2 packets/sec. A packet is 32 bytes long. In the grid scenarios, the sink is placed on the northwest corner of the network. In random deployment, the 100 nodes are randomly placed in the simulated area, while the sink is placed roughly at the center.

We compare JiTS with VMS using the same routing protocol (GF) that was originally used in the RAP scheme [1]. Later, we also show that SP significantly outperforms GF for JiTS. Since JiTS does not require any MAC layer modification, we use the original IEEE 802.11 as our MAC layer protocol, while we use the modified MAC layer for RAP as done in [1].

Experimentally, we have observed that a safety margin parameter of 0.7 works well across different deadlines. Thus, 30% of the slack time is set aside to account for unexpected variations in transmission or queuing delays.

3.1 Evaluation of JiTS and VMS

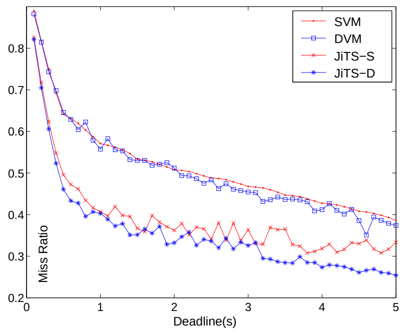

Figure 1. JiTS(GF) vs VMS Miss Ratio

The first experiment compares the performance of JiTS to RAP. Figures 1 and 2 show that, for different deadline requirements, the miss ratios of Static and Dynamic JiTS are much lower than those of DVM and SVM across all the tested deadlines. (Dynamic JiTS outperforms static JiTS in terms of miss ratio.) The same observation holds for the drop ratios.

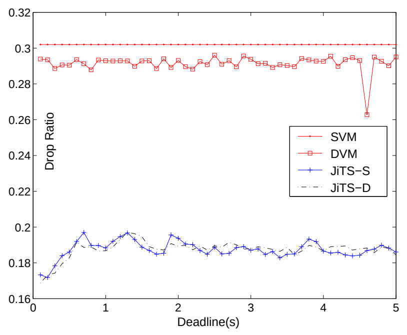

Figure 2. Drop Ratio: JiTS(GF) vs VMS

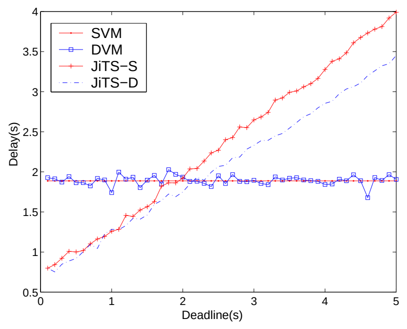

Figure 3. Average Delay: JiTS (GF) vs VMS

Figure 3 shows the average delay of JiTS and RAP to illustrate the difference between the two scheduling approaches. The average delay of JiTS grows linearly as the deadline increases, because the intermediate nodes delay packets proportionately to the deadline. In this way, we can take advantage of the slack under overload. Note that dynamic JiTS manages to keep the average delay around the value of $0.7 \cdot deadline$, while the static JiTS has slightly higher average delay. Since RAP does not delay packets, its delay only depends on the data generation pattern but does not change with the deadline.

Since the VMS uses multiple FIFO queues as its priority queue, packet starvation commonly happens. As a result, the maximum packet delay suffered by RAP is longer than that of JiTS by a factor of 2 to 3. (Due to space limitations, we do not include the results here.) We will only compare our JiTS policies with SVM in the remainder of this paper, because the original authors [1] observed SVM to be superior to DVM.

3.2 Effect of Routing Protocol

In the second set of experiments, we compare the performance of JiTS under GF with JiTS under Shortest Path (SP)

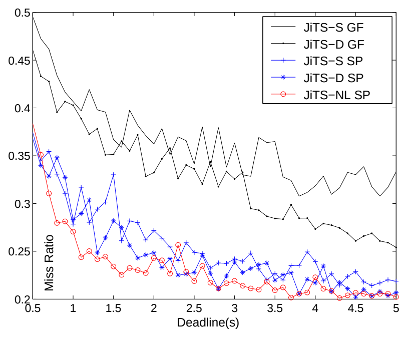

Figure 4. JiTS SP vs GF Miss Ratio

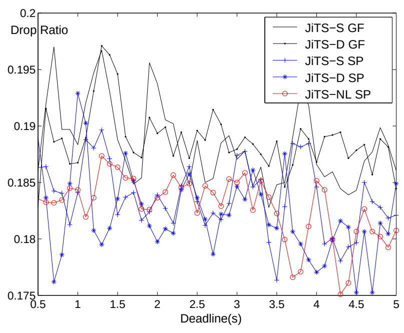

Figure 5. Drop Ratio: JiTS SP vs GF

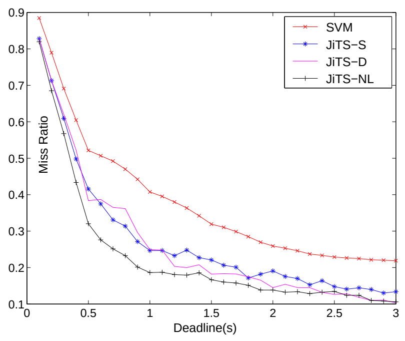

Figure 6. Miss Ratio of Bursty: JiTS vs SVM

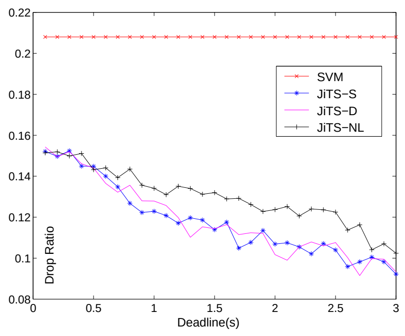

Figure 7. Drop Ratio of Bursty: JiTS vs SVM

routing. In this experiment, we also show the performance of the nonlinear JiTS scheduling algorithm (JiTS-NL). Figures 4 and 5 show the miss ratio and drop ratio. From these figures, we observe that JiTS performs considerably better with SP than GF. In general, dynamic JiTS performs better than static JiTS for both routing protocols. Further, JiTS-NL significantly improves the performance compared to static and dynamic JiTS. Especially, the improvement is most pronounced under tight deadlines, showing the applicability of JiTS-NL to real-time data dissemination. We have also observed that the maximum and average delay of JiTS with GF is higher than that of JiTS with SP for the same deadlines (results not shown), because GF may use longer paths than SP in terms of the number of hops.

3.3 Performance under Bursty Traffic

In this set of experiments, we evaluate the performance of JiTS vs. RAP under bursty traffic conditions. At every 10 seconds, we let all the nodes publish data packets with the pre-set data rate in the first 5 seconds then stop publishing for the remaining 5 seconds. Figures 6 and 7 show the miss ratio and drop ratio of JiTS and SVM under bursty traffic with end-to-end deadline increasing from 0.1 second to 3.0 seconds. In Figure 6, we can see that the miss ratio of dynamic JiTS is much lower than that of SVM under the bursty traffic, because JiTS can tolerate the traffic burst by delaying some packets and taking advantage of the idle period. On the other hand, SVM cannot make use of the traffic behavior, since it does not delay packets even if there is slack. In Figure 7, we observe that the JiTS policies achieve the lower drop ratio than SVM, delivering more packets as the deadline constraints are relaxed. In contrast, SVM suffers almost the same drop ratio even when the deadlines are relaxed.

3.4 Performance with Different Deadlines

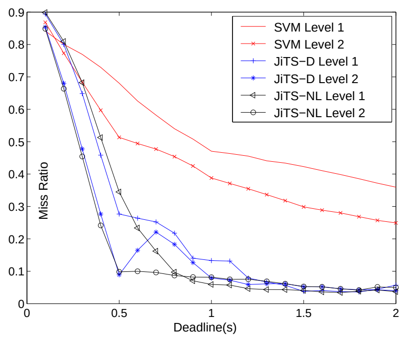

Figure 8. Two Level Deadlines: Miss Ratio

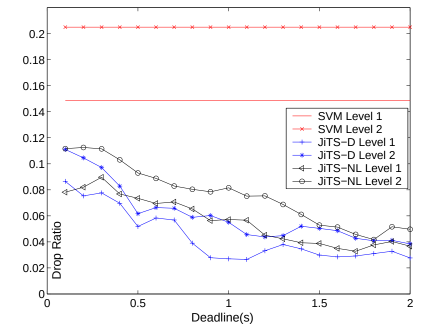

Figure 9. Two Level Deadlines: Drop Ratio

In this set of experiments, we have simulated bursty generations of two types of data packets with different deadlines where the level 1 deadline is a half of the level 2 deadline. Ideally, the scheduling algorithm would allow both data types to meet their deadlines. Figures 8 and 9 show the miss ratio and drop ratio of SVM, JiTS-D and JiTS-NL. Given short deadlines, the level 2 traffic shows better performance, because the network becomes often unable to satisfy the aggressive level 1 deadlines. Once the deadlines increase beyond a certain level, JiTS is able to provide similar performance for the two traffic types with different timing requirements. However, for SVM, the level 2 traffic continues to show better performance than the level 1 traffic.

4 Conclusions and Future Work

Real-time data dissemination is a service of great interest to many sensor network applications. We proposed and evaluated a *Just-in-Time* scheduling protocol that offers significant advantages over existing real-time sensor data dissemination schemes. JiTS delays packets by a fraction of their slack time at each hop. As a result, it can better tolerate bursts than schemes that simply prioritize packet transmission. We also explored the effect of routing in meeting timing constraints and showed that Geographic Forwarding can lead to suboptimal operation. JiTS outperforms RAP in both the miss ratio and overall delay. We explored several approaches to allocating the available slack time among intermediate nodes and showed that nonlinear distribution of the slack time, which assigns more time to hops closer to the sink, results in better performance than linear distribution of the slack time. JiTS is a routing layer solution that requires no modification of lower level protocols. Thus, it can be deployed independent of the underlying MAC layer and hardware capabilities. From the simulation study, we found the drop ratio is the lower bound of the miss ratio of real-time communication. Given a reasonable end-to-end deadline, if the frequency of packet dropping mainly due to congestion is decreased, the miss ratio is also decreased. In the future, we will further investigate JiTS in the context of wireless sensor networks. Further, we will investigate other related issues such as data aggregation in JiTS and congestion control for real-time data transmission in sensor networks.

References

[1] Chenyang Lu, Brian M. Blum, Tarek F. Abdelzaher, John A. Stankovic, and Tian He. RAP: A real-time communication architecture for large-scale wireless sensor networks. *Real-Time and Embedded Technology and Applications Symposium, 2002*, 24-27 Spet. 2002.

[2] Tian He, John A Stankovic, Chenyang Lu, and Tarek Abdelzaher. SPEED: A stateless protocol for real-time communication in sensor networks. *International Conference on Distributed Computing Systems(ICDCS 2003)*, May 2003.

[3] Tian He, John A. Stankovic, Chenyang Lu, and Tarek F. Abdelzaher. A spatiotemporal protocol for wireless sensor network. *IEEE Transactions on Parallel and Distributed Systems*, 2005.

[4] Emad Felemban, Chang-Gun Lee, Eylem Ekici, Ryan Boder, and Serdar Vural. Probabilistic QoS guarantee in reliability and timeliness domains in wireless sensor networks. *IEEE INFOCOM*, 2005.

[5] Qingfeng Huang, Chenyang Lu, and Gruia-Catalin Roman. Mobicast: Just-in-time multicast for sensor networks under spatiotemporal constraints. *International Workshop on Information Processing in Sensor Networks (IPSN'03)*, April 2003.

[6] Network simulator. http://www.isi.edu/nsnam/ns/, 2002.

[7] David B. Johnson, David A. Maltz, and Josh Broch. DSR: The dynamic source routing protocol for multi-hop wireless ad hoc networks. In Charles E. Perkins, editor, *Ad Hoc Networking*, pages 139–172. Addison-Wesley, 2001.