A Reliable Data Transport Protocol for Partitioned Actors in Wireless Sensor and Actor Networks

Nikhil Handigol
Department of Computer Science, Stanford University, California, nikhilh@cs.stanford.edu
Ka. Selvaradjou
Department of Computer Science and Engineering, Pondicherry Engineering College, India, selvaraj@pec.edu
C. Siva Ram Murthy
Department of Computer Science and Engineering, Indian Institute of Technology Madras, Chennai, India, murthy@iitm.ac.in

Abstract—**In Wireless Sensor and Actor Networks (WSANs), effective Actor-Actor Communication (AAC) is an important requirement for the timely responses to events reported by the sensors. However, due to scattered nature of events, mobility of actor nodes, and low density of actor nodes, the network of actor nodes tends to get partitioned frequently. To provide effective AAC in such situations, the energy-constrained sensor nodes located between the partitioned actor nodes need to be utilized. This solution for healing the actor network partitions should involve minimal use of the sensor nodes so that the network lifetime is maximized. In this work, we propose an energy-efficient Actor-Actor Reliable Transport Protocol (A^2RT) for WSANs with actor nodes equipped with directional antennas and dual radio interfaces. Our proposed transport protocol consists of a transport wrapper and a dynamic priority scheduler. Using simulations, we show that our transport wrapper achieves high reliability with minimum retransmissions both under static and dynamic network topology conditions. The results also show that the traffic scheduler of our protocol helps to achieve the goals of real-time delivery by maximizing the number of packets that meet the delay constraints.**

Index Terms—**WSANs, Actor-Actor Communication, Energy-Efficiency, Reliable Transport.**

I. INTRODUCTION

Wireless Sensor and Actor Networks (WSANs) enable a closed loop control and monitoring system that holds a huge promise for future applications [1]. These networks can be an integral part of systems such as battlefield surveillance and attack detection. WSANs consist of sensor and actor nodes. Sensor nodes are low-cost, low power, and tiny devices with limited sensing, computation, and wireless communication capabilities. They sense the surrounding phenomena and communicate the information to the actor nodes. The wireless enabled actor nodes are capable of acting on the environment such as putting off fire and pumping gas on intruders. These nodes are typically resource rich and equipped with better processing capabilities, higher transmission power, and longer/renewable battery life. In a typical WSAN, a large number of sensor nodes are statically deployed and a relatively fewer number of mobile actor nodes are placed randomly. Providing effective sensor-to-actor communication (SAC) and actor-to-actor communication (AAC) are two important requirements in WSANs [1] [2]. When sensor nodes report events to one or more actor nodes, AAC is needed to enable a coordinated action between the actor nodes to meet certain real-time deadlines. AAC is relatively easy if the deployment of actor nodes is such that the actor network always remains connected. However, real-world actor networks are sparse and dynamic due to low-density deployment and mobility of actor nodes. Under such situations, achieving effective AAC becomes a hard problem. As we show in Section II, when the actor network gets partitioned, the actor nodes within one partition can no longer directly communicate with other actor nodes belonging to other partitions. To heal the partitions in actor network, we proposed an architecture [3] for WSANs that uses directional antennas with dual radio interfaces on actor nodes. We showed how the intermediate sensor nodes could be used as communication bridges between partitioned actor nodes. Henceforth, all references to our architecture imply this particular one, unless stated otherwise. In general, AAC in our WSAN architecture has three major issues: 1) Energy constraint - The sensor nodes forming the bridges have limited energy. 2) Reliable transport - AAC in WSANs requires data to be transferred with high reliability, and 3) Real-time delivery - AAC has strict delay constraints as actions must be performed immediately in response to the reported events. In another of our previous work, we had presented a novel Energy Efficient Directional Routing protocol (EEDR) [4] for our WSAN architecture. In this paper, we propose a transport protocol that ensures reliable transfer of data between actor nodes under partition. Traditional transport protocols such as basic TCP [5] [6] [7], Vegas [7], SACK [8], and Reno [9] [10] achieve reliable data transfer using end-to-end retransmissions. While such protocols help to achieve the functionality required for AAC and impose low memory requirements on the intermediate nodes, they adversely affect the network lifetime. In WSANs, wireless losses and path breaks caused by death of sensor nodes and mobility of actor nodes result in frequent packet drops. As a result, a transport protocol that achieves reliability using end-to-end retransmissions would require a greater number of packets transmitted on the network. This would, in turn, drain out the limited energy of the sensor nodes forming the bridges resulting in quick death of the WSAN. At the other extreme is a transport protocol that achieves reliability using hop-by-hop

978-1-4244-8520-8/10/$26.00 ©2010 IEEE

retransmissions. Such a protocol would minimize the number of packets transmitted on the network and thus, save sensor node energy. However, such a protocol would be infeasible to implement. First, it would require large memory on each node along the route but the routes used for AAC in our architecture consist of sensor bridges with small memory. Second, it would not be able to handle the frequent path breaks in WSANs. Thus, a good transport protocol for AAC should have low memory requirements and minimal retransmission at the sensor bridges. In our architecture, the sensor nodes route two types of data packets - i) sensor data packets, and ii) actor data packets. Both sensor data and actor data are delay intolerant. If the data does not reach the destination in time, the required functionality of the WSAN can not be achieved. Therefore, scheduling the transmission of the two types of data packets at the sensor nodes should achieve two important goals: i) maximization of the number of packets meeting their deadline. ii) fairness between sensor and actor data packets. Static scheduling schemes that give high priority to either the sensor data packets or the actor data packets does not achieve the desired fairness. Even a partially dynamic solution such as First Come First Serve (FCFS) does not guarantee a low Deadline Miss Ratio (DMR) and high fairness. Therefore, a completely dynamic traffic scheduling scheme that addresses these issues is required. Thus, we require a transport protocol that provides reliable transport without affecting the network lifetime adversely and also performs traffic scheduling to maximize the number of packets meeting the deadline. In this paper, we mathematically formulate the problems of reliable transport and traffic scheduling, and propose a novel protocol, Actor-Actor Reliable Transport (A^2RT), that ensures the real-time and reliable delivery of data between partitioned actor nodes. To the best of our knowledge, AAC in WSANs has not been studied from this perspective before. Though Real-Time and Reliable Transport ($(RT)^2$) [11] is designed for AAC in WSANs, it assumes that the actor network is always connected. On the contrary, A^2RT specifically considers the partitioning of the actor network and provides reliable transport in an energy efficient way.

The rest of the paper is organized as follows. In Section II, we highlight the difficulties in providing AAC in a sparse topology of actor nodes and describe the motivation behind our work. In Section III, we study the problems of reliable transport and real-time data delivery and describe the proposed A^2RT protocol. In section IV, we evaluate the performance of A^2RT protocol using simulations. In Section V, we conclude our work with discussions on future work.

II. MOTIVATION

Figure 1 shows an overview of a WSAN architecture. The network shown in the figure consists of a few actor nodes (triangles) and numerous sensor nodes (circles). Actor nodes communicate with each other over a long-range actor-channel (shown as zig-zag lines) without interfering with the short-range sensor-channel (shown as directed arrows). As the terrain size is larger and relatively a few number of actor nodes are deployed, the topology of actor network tends to be disconnected and results in several partitions. Even if the initial deployment of actors is such that it forms a single connected network, the connectivity can not be preserved throughout the lifetime of the network as the actor nodes need to move towards the events. As shown in the figure, the actor nodes in partition *P1* can not directly communicate to any of the actor nodes in other partitions (*P2, P3, or P4*) and vice versa. In [4], we showed by simulations that the partition probability increases when the deployed actors are smaller in quantity in a given terrain. Under such partitioned situations, the sensor nodes, being numerous in the field, can be used to form bridges that heal these partitions. But the sensor nodes, being constrained in terms of energy, memory, and computational power, must be intelligently utilized to maximize the time for which AAC across the partitions can continue.

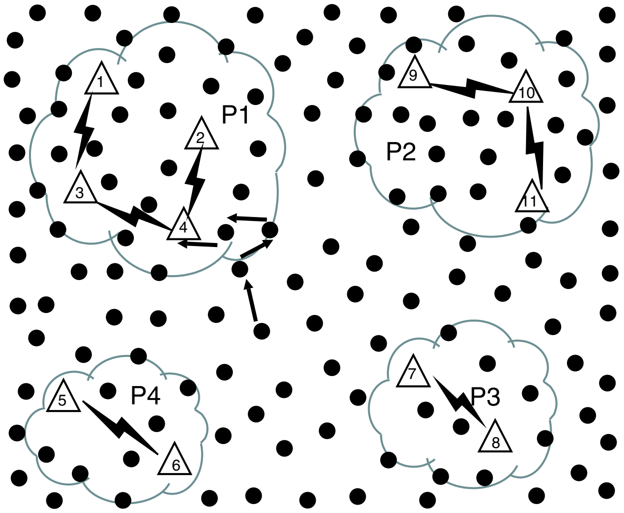

Fig. 1. WSAN architecture and network partition.

When communication between two actor nodes belonging to different partitions need to be established, actor nodes can switch to *short-range sensor-channel* and thus heal the partition by means of intermediate sensor nodes. But, such a solution would result in enormous amount of delay in AAC that can not be tolerated in real-time applications. Alternatively, the partitioned actor nodes can switch to sensor-channel but with a long-range transmission such that only a minimal number of intermediate sensor nodes act as forwarders. Though, this approach reduces the end-to-end delay in AAC, a large amount of sensor nodes would suffer severe energy drain as the collision domain is large. The use of directional antenna on actor nodes in our architecture, can significantly reduce the energy consumption of sensor nodes as the collision domain is reduced.

III. ACTOR-ACTOR RELIABLE TRANSPORT PROTOCOL

AAC in general has two major requirements - reliable transport and real-time delivery of data. In order to meet these two needs, we require a transport protocol that provides reliable transport without affecting the network lifetime negatively and also performs traffic scheduling to achieve real-time data delivery. In this section, we mathematically formulate the problems of reliable transport and traffic scheduling, and then describe our Actor-Actor Reliable Transport (A^{2}RT) protocol that addresses these two problems.

A. Reliable Data Transfer and Energy Efficient Traffic Scheduling - An Analysis

The following sections provide an analytical framework and solutions for the problems just stated. We use these solutions to design our transport protocol.

1) Reliability: We model the problem of reliability and energy-efficiency by first studying a simple k-hop path with respect to expected number of link-level transmissions and extending the results to a multi-phase reliability scheme. A multi-phase reliability scheme is one in which end-to-end reliability is achieved by dividing the path into multiple phases and transmitting packets reliably phase-by-phase.

Consider an end-to-end reliable transport on a simple multi-hop path with characteristics such as p, the probability of successful transmission, k, the number of hops, and n_f, number of failed transmissions.

Using elements of basic probability theory [12], we derive the expected number of transmissions, as follows.

The probability that n number of link-level transmissions are required to reliably transport a packet is given by

$$P(n) = \sum_{n_f=\lfloor \frac{n}{k} \rfloor}^{(n-k)} f(n_f, n, k) p^{(n-n_f)} (1-p)^{n_f} \tag{1}$$

where, $f(n_f, n, k)$ is the number of ways such that the sum of n_f integers is $(n-k)$ and $1 \leq n_f \leq k$. The expression for $f(n_f, n, k)$ is given by

$$f(n_f, n, k) = \left(\begin{array}{c} n-k-1 \\ n_f - 1 \end{array} \right) - n_f \left(\begin{array}{c} n-2k-1 \\ n_f - 1 \end{array} \right) \tag{2}$$

Therefore, the expected number of transmissions over a k-hop path is given by

$$E_k(N) = \sum_{n=k}^{\infty} nP(n) \tag{3}$$

Substituting the expression for $P(n)$ from Equation 1, we get,

$$E_k(N) = \sum_{n=k}^{\infty} np^n \left\{ \sum_{n_f=\lfloor \frac{n}{k} \rfloor}^{(n-k)} f(n_f, n, k) \left(\frac{1-p}{p} \right)^{n_f} \right\} \tag{4}$$

Now, consider a reliable transport over an L-hop path, where L is an integral multiple of k. If end-to-end retransmission is used, the expected number of transmissions required to deliver a packet can be obtained by replacing k in Equation 4 by the total number of hops (L) and is given by

$$E_{end-end}(N) = \sum_{n=L}^{\infty} np^n \left\{ \sum_{n_f=\lfloor \frac{n}{L} \rfloor}^{(n-L)} f(n_f, n, L) \left(\frac{1-p}{p} \right)^{n_f} \right\} \tag{5}$$

On the other hand, if reliable transport is achieved on the path by m phase-wise retransmissions, the expected number of transmissions required will be the sum of the expected number of transmissions required for each phase.

$$E_{m-phase}(N) = \sum_{k=1}^{m} E_k(N) \tag{6}$$

The expected number of transmissions will, therefore, be given by

$$E_{m-phase}(N) = m \sum_{n=k}^{\infty} np^n \left\{ \sum_{n_f=\lfloor \frac{n}{k} \rfloor}^{(n-k)} f(n_f, n, k) \left(\frac{1-p}{p} \right)^{n_f} \right\} \tag{7}$$

From the Equations 5 and 7, we observe that the number of transmissions required when retransmissions are split over multiple phases is lower than that required when the retransmissions happen end-to-end. Figure 2 shows the variation of the expected number of hop-by-hop transmissions for a 40-hop path with the number of phases the path is segmented into. We can observe that the number of transmissions required falls exponentially with increasing number of phases. Therefore, a multi-phase retransmission scheme is significantly more energy-efficient compared to an end-to-end retransmission scheme. The trade-off, however, is the need for additional memory by the intermediate nodes for data buffering. Because we assume actor nodes to be resource rich (in terms of energy as well as memory), they are capable of acting as data-buffering nodes.

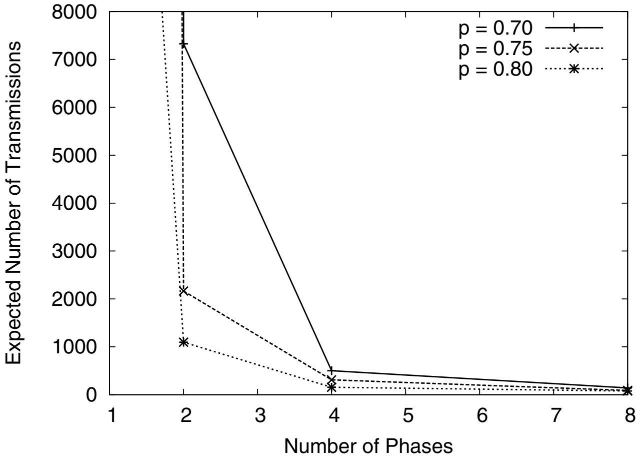

Fig. 2. Expected number of transmissions for various link-level transmission probabilities.

2) Scheduling: We model the problem of traffic scheduling at the sensor nodes as a classical queueing theory problem. We assume that the sensor and actor packets arrive as Poisson traffic at rates λ_s and λ_a, respectively, and the arriving packets are queued up in two separate queues - i) sensor traffic queue and ii) actor traffic queue. The scheduler acts as a single server serving the packets from the two queues with mean service rates μ_s and μ_a, respectively.

When a packet is to be served, the server randomly picks a packet from the sensor or the actor traffic queue with probabilities p_s and p_a, respectively. Thus, each queue can be considered as an $M/M/1/B$ queue, with service rates $p_s\mu_s$ and $p_a\mu_a$, respectively.

Each packet has its own values of number of hops needed to reach the destination, delay threshold, and time spent in the queue. In addition, sensor packets have an additional parameter of redundancy (η) - the number of copies of the sensor packet being routed to the destination simultaneously. The notation used is enlisted in Table I. The subscript s denotes sensor node and a denotes actor node.

TABLE I
NOTATION USED IN TRAFFIC SCHEDULING ANALYSIS

Number of hops taken	h_s, h_a
Number of hops left	h'_s, h'_a
Delay threshold	T_s, T_a
Time spent	t_s, t_a
Remaining time	$\Delta t_s, \Delta t_a$
Average hop delay incurred	τ_s, τ_a
Queue size (in Packets)	B
Sensor packet redundancy	η

Using basic Queueing Theory [13], we determine the values of various metrics as described below.

The traffic intensity ρ_s is defined as

$$\rho_s = \frac{\lambda_s}{p_s\mu_s} \tag{8}$$

The loss rate λ'_s is given by

$$\lambda'_s = \lambda_s \left(\frac{1-\rho_s}{1-\rho_s^{B+1}} \right) \rho_s^B \tag{9}$$

The mean delay experienced by a packet at a node, $d(p_s)$, is given by

$$d(p_s) = \frac{1}{\lambda'_s} \left(\frac{\rho_s}{1-\rho_s} - \frac{(B+1)\rho_s^{B+1}}{1-\rho_s^{B+1}} \right) \tag{10}$$

The corresponding values for actor packets are similarly defined.

We now formulate the problem of traffic scheduling as a Mixed Integer Non-Linear Programming (MINLP) [14] problem as follows.
Minimize:

$$E_s + E_a \tag{11}$$

Subject to the following constraints:

$$E_s = \frac{\lambda'_s h_s}{\lambda_s} + \sum_i \frac{n_{s_i}}{\eta q_s}(h_{s_i} + h'_{s_i}) \tag{12}$$

$$E_a = \frac{\lambda'_a h_a}{\lambda_a} + \sum_i \frac{n_{a_i}}{q_a}(h_{a_i} + h'_{a_i}) \tag{13}$$

$$\left((d(p_s) + \tau_s) \times h'_{s_i} - \Delta t_{s_i} \right) \times n_{s_i} \geq 0, \forall i \tag{14}$$

$$\left(d(p_a) + \tau_a) \times h'_{a_i} - \Delta t_{a_i} \right) \times n_{a_i} \geq 0, \forall i \tag{15}$$

$$p_s + p_a = 1 \tag{16}$$

where $n_{s_i}, n_{a_i} \in \{0, 1\}$, $p_s, p_a \geq 0$.

Equation 11 gives the objective function to be minimized. The variables E_s and E_a refer to the unwanted *effort* put by sensor and actor nodes, respectively in transmitting the packets that miss the deadline. Here, the *effort* is proportional to energy. The Equation 12 gives the expression for the calculation of E_s. We can see that the expression consists of two parts. The first part gives the amount of energy that is expected to be wasted on packets that are dropped from the queue due to buffer overflow. The second part gives the amount of energy that is expected to be wasted on the packets that are enqueued at the node but are expected to ultimately miss the deadline. Similarly, Equation 13 gives the expression for the value of E_a. To calculate the number of packets that will miss the deadlines, we have binary variables n_{s_i} and n_{a_i} associated with each sensor packet and actor packet enqueued, respectively. The variables assume a value of 1 if the packet is expected to miss the deadline, and a value of 0 otherwise. Equations 14 and 15 are the constraints that determine the values of n_{s_i} and n_{a_i}, respectively. Equation 16 ensures that the sum of the priorities assigned to the sensor queue and the actor queue is equal to 1.

The values of p_s and p_a at the optimal solution gives the values of optimal priorities of the sensor traffic and the actor traffic, respectively.

Algorithm 1 Approximation iterative algorithm for the MINLP problem

```
Initialize $n_s = B/2, n_a = B/2, p_s = 1.0, p_{s\_old} = 0.0$
while $|p_s - p_{s\_old}| > \epsilon$ do
    $p_{s\_old} = p_s$
    Substitute the values of $n_s$ and $n_a$ in the equation of $E_s + E_a$
    Calculate $p_s$ s.t., $E_s + E_a$ is maximized and $0 \leq p_s \leq 1$
    $p_a = 1 - p_s$
    $n_s$ = number of packets s.t., $(d(p_s) + \tau_s)h'_{s_i} > \Delta t_{s_i}$
    $n_a$ = number of packets s.t., $(d(p_a) + \tau_a)h'_{a_i} > \Delta t_{a_i}$
end while
```

MINLP problems can not be solved optimally in polynomial time. Moreover, sensor nodes, that solve this problem are constrained in terms of energy and computational power. We, therefore, propose an alternative approximate iterative algorithm, similar to the Schweitzer's Approximation for Mean Value Analysis in closed queuing networks [13], as shown in Algorithm 1. We start by a rough estimation of n_s and n_a, substitute their values in Equations 12 and 13, and then find the optimal values of p_s and p_a. Based on these values, we obtain a new estimate of the values of n_s and n_a. We make multiple iterations of this estimation process till we gain reasonable amount of accuracy. Thus, exploiting the nature of our problem, we can obtain an approximate solution to the given problem efficiently.

B. Protocol Design

In this section, we describe our A^{2}RT protocol which is based on the solutions formulated in the previous section. As shown in Figure 4, our A^{2}RT protocol consists of the following two additional layers in the protocol stack: a) *Transport Wrapper* which is on top of the transport layer, to achieve end-to-end reliability by setting up multiple inter-partition transport sessions and b) *Traffic Scheduler* which is between the network and MAC layers, to perform scheduling based on the Algorithm 1.

1) Transport Wrapper: In Section III-A1, it has been established that end-to-end reliability obtained by using multiple phases of reliable transport results in a lower number of retransmissions compared to the one that uses end-to-end retransmissions, and hence, is more energy efficient. Our WSAN architecture and the underlying Energy Efficient Directional Routing protocol EEDR [4] allow us to exploit this result and develop a transport protocol based on this paradigm. We assume that the WSAN consists of a number of partitions of actor nodes and numerous sensor nodes bridging these partitions. While sensor nodes are constrained in terms of energy and memory, actor nodes are resource rich and have large memory to store received packets and forward at a later point of time. Therefore, reliable transport between actor nodes belonging to different partitions can be obtained by setting up multiple inter-partition reliable transport sessions, one between each pair of consecutive partitions along the routes. Figure 3 gives a brief idea of the functioning of the transport wrapper.

As shown in Figure 4, the wrapper interacts with the routing protocol and obtains route information from it. Based on the route information, it establishes multiple transport sessions between consecutive actor network partitions as shown in the Figure 3. Any reliable transport protocol, such as TCP or its variants, can be used to establish the inter-partition transport sessions. The first node of every partition along the route acts as the transport cache and stores the packets in the memory before it is reliably transferred to the next transport cache along the route. The transport cache removes the packets from memory only after it receives acknowledgments from the transport cache of the next actor network partition. At the same time, there is an end-to-end wrapper session active between the source actor node and the destination actor node. The source node maintains a “master-copy” of the packets and flushes them out only when it receives an end-to-end acknowledgment from the destination node. The routing protocol informs the wrapper in case of path breaks, and the wrapper sets up a new wrapper session along the new path. Thus, the transport wrapper with sufficient cross-layer collaboration from the routing protocol achieves end-to-end reliable transport.

2) Traffic Scheduler: The dynamic priority assignment solution obtained in Section III-A2 is implemented as a traffic scheduler in our protocol. As shown in Figure 4, the scheduler is positioned between the routing layer and the MAC layer in the protocol stack. It maintains two separate queues for packets of sensor data and actor data, respectively. It periodically calculates the priorities using Algorithm 1 given in Section III-A2 and schedules the packets accordingly. The main objective of this scheduler is to minimize the energy spent on unwanted transmissions by sensor and actor nodes.

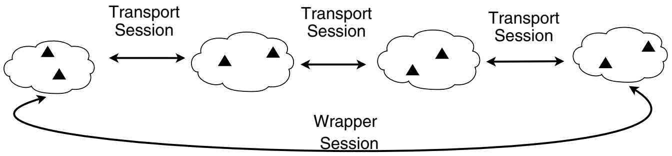

Fig. 3. End-to-end wrapper session through multiple inter-partition transport sessions.

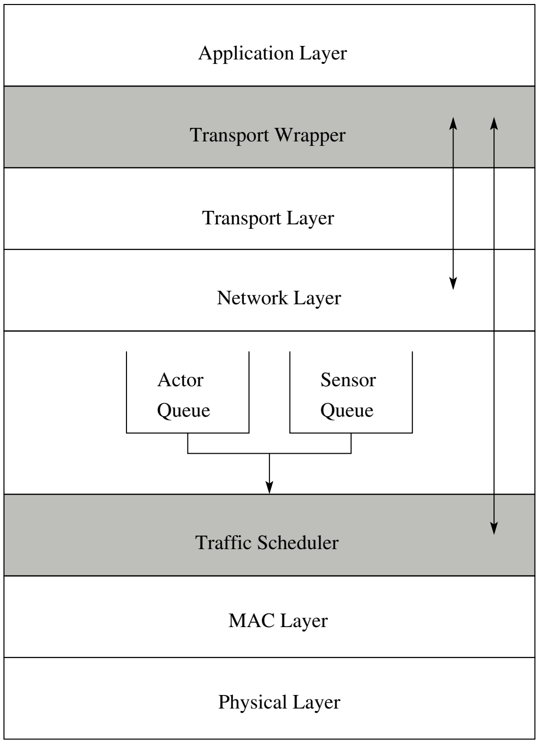

Fig. 4. A^2RT protocol stack and traffic scheduler.

IV. PERFORMANCE EVALUATION

In this section, we present the performance evaluation of our proposed A^2RT protocol by simulations. We used the ns-2.29 [15] network simulator to evaluate the performance of the proposed protocol.

A. Simulation Metrics and Parameters

The simulation parameters are summarized in Table II.

TABLE II
SIMULATION PARAMETERS

Simulation Area	1000 m x 1000 m
Number of sensor nodes	1000
Number of actor nodes	50
Number of directional antenna sectors	6
Number of actor flows	1, 2, 4, 8
Sensor node transmission range	30 m
Actor node transmission range	150 m
Packet size	512 Bytes
Sensor traffic packet inter-arrival time	0.1 to 1 s
Actor traffic application packet size	1 KB to 1 MB
Routing protocol	EEDR
Queue length	50 packets
Mean deadline	250 ms
Mean sensor packet inter-arrival time	0.5 s
Underlying transport protocol	TCP NewReno, Tahoe
Mean event inter-arrival time	10, 20, 30, 40, 50 s
Actor node speed	2 m/s

We consider network lifetime, throughput, Deadline Miss Ratio (DMR) and DMR fairness as metrics to evaluate the performance of A^2RT. All metrics are evaluated as a function of the actor traffic for various number of actor flows. The death of the network is defined as a point of time at which there exist at least two actor nodes that cannot communicate with each other. Accordingly, the time between the beginning of the functioning of the network to the death of the network is considered to be the network lifetime. We measure throughput as the ratio between successful packets delivered to destination actor nodes and the overall packets generated by the source actor nodes in the network. Both SAC and AAC have strong delay constraints. To measure the performance of A^{2}RT with respect to delay constraint, we associate a deadline for each packet generated, before which it has to reach its destination. We define the DMR as the fraction of packets that miss the deadline. A good transport protocol for WSANs should minimize DMR. Suppose DMR of the sensor traffic is χ_s and that of actor traffic is χ_a. Using Jain's fairness index [13], we define DMR fairness as follows:

$$DMR\ fairness = \frac{(\chi_s + \chi_a)^2}{2 \times (\chi_s^2 + \chi_a^2)} \tag{17}$$

We measure the performance of our transport wrapper for various values of the parameters and compare the corresponding results with underlying transport protocols as TCP NewReno [6] and Tahoe [16]. We would like to emphasize at this point that we have found no appropriate parallel solution scheme in the literature for the proposed architecture for healing network partitions. For the lack of existing schemes, we compare the performance of our protocol with that of the corresponding transport protocols without the wrapper.

To simulate real-life scenarios, the AAC data is generated as files of random sizes picked from a uniform distribution between 1 KB to 1 MB. The deadlines for the packets are generated by picking a value randomly between ±20% of the pre-determined mean deadline.

B. Simple Topology

In this section, we describe a set of simulations that we conducted on a simple topology. The idea behind these simulations is to study the performance of various underlying transport protocols with our transport wrapper and compare it with that of the corresponding transport protocols without the wrapper in a simple controlled environment.

Topology Description: In this simulation, we deployed 10 actor nodes randomly in a terrain such that it results in 4 partitions. The deployment included 100 sensor nodes uniformly deployed in the terrain. Metrics such as throughput and DMR are studied by varying the average number of simultaneous AAC flows and average deadlines, respectively. The flows are established between actors belonging to isolated partitions. The results presented are averaged over 10 runs for each set of parameters.

In Figure 5, we compare the total AAC throughput of two transport protocols namely, TCP NewReno and TCP Tahoe, with and without our transport wrapper. From the figure, we observe that the performance increases drastically when the transport protocols are used along with the transport wrapper. The transport wrapper enables reliable data transmission between every two partitions, enabling an end-to-end reliable transfer of data with fewer retransmissions. The throughput increases by as much as 75% with the usage of the wrapper.

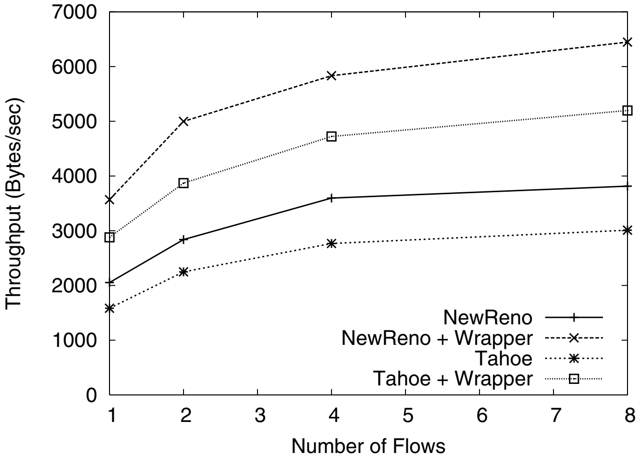

Fig. 5. Throughput (simple topology) comparison with and without the transport wrapper.

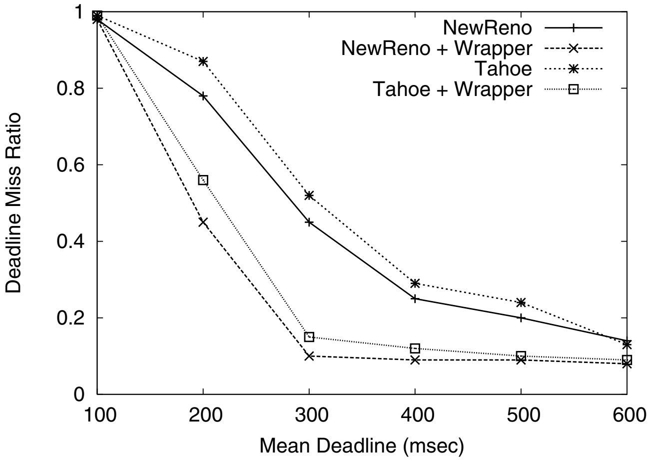

Fig. 6. DMR (simple topology) comparison with and without the transport wrapper.

Figure 6 shows the variation of DMR for various values of average deadline for the generated packets. We study the performance of TCP NewReno and TCP Tahoe with and without the transport wrapper. Since there is no background sensor traffic involved in these experiments, the existence of the traffic scheduler does not make a difference in the performance. From the figure, we observe that DMR decreases steeply with the initial increase in mean deadline and beyond a certain threshold, flattens out and shows a smaller slope. The threshold point is the average delay within which most of the packets can be expected to be delivered to the destination using that particular protocol. A lower threshold point indicates that the protocol delivers most of the packets with a lower delay and hence performs better. The transport protocols, when used with the transport wrapper consistently show lower DMR when compared to those without wrapper. The average deadline threshold for transport protocols with wrapper is much lower than that of transport protocols without the wrapper. This can be attributed to the fewer retransmissions that the packets experience when the transport wrapper is used resulting in a lower end-to-end delay.

C. Dynamic Topology

In this section, we describe a set of simulations that we ran on a relatively larger network to study the performance of A^2RT under dynamic topology conditions. We assume that the node movement is caused due to events occurring in the field. We generate the events as a Poisson process with varying arrival rate and uniform distribution in space. When an event occurs, the actor node closest to the event starts moving towards the event with a constant speed. If an actor node is required to act on more than one event, then the actor node chooses to move towards the latest event and "drops" the other events. Thus, a higher event-rate leads to more frequent actor node movements leading to a more dynamic topology. From the experiments in Section IV-B, we found that the TCP NewReno performs better than Tahoe. Henceforth, we use TCP NewReno for rest of our experiments. In addition, we introduce the background sensor traffic in these simulations. Sensor traffic is generated as CBR traffic with deadlines generated similar to those in actor traffic.

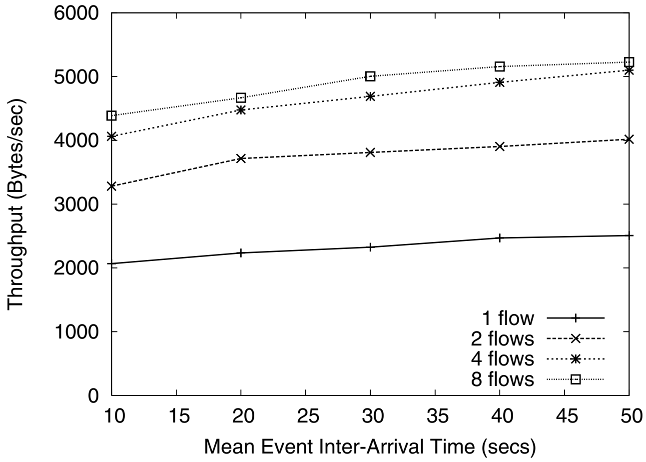

Fig. 7. Throughput (dynamic topology) comparison for various number of flows.

Figure 7 shows the variation of total AAC throughput with mean event inter-arrival time for various number of flows. We observe that A^2RT achieves good throughput even under dynamic network conditions. We also observe that the total throughput increases with increasing number of flows. The increase, however, is not proportional to the increase in the number of flows. This is mostly because of losses due to contention and the corresponding decrease in congestion window sizes. For a given number of flows, we observe that the throughput increases with increasing event inter-arrival time. This is because, a higher value of event inter-arrival time means a more stable network and less frequent path breaks, resulting in higher throughput.

Figure 8 shows the variation of the network lifetime with mean event inter-arrival time for various number of flows. We observe that the lifetime decreases with increase in number of flows. This is due to increasing contention for the same resource, namely the energy of the underlying network of sensor nodes. Also, for a given number of flows, the network lifetime increases with decreasing event-rate. This is because greater event-rate results in a more dynamic actor network topology, which in turn results in more frequent path breaks leading to more aggressive consumption of sensor node energy.

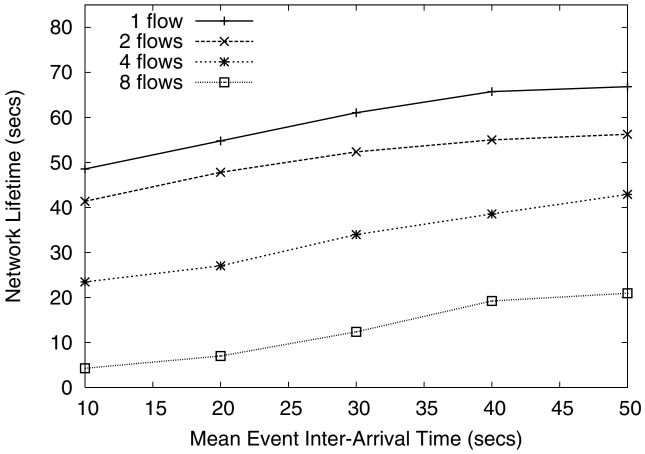

Fig. 8. Network lifetime (dynamic topology) comparison for various number of flows.

V. CONCLUSIONS AND FUTURE WORK

In this paper, we studied the problem of providing effective actor-actor communication (AAC) in WSANs. In particular, our work considered the problems that could occur in a sparse and partitioned actor network. We addressed the challenges in providing reliable and timely data delivery between partitioned actors by first developing an analytical framework and proposing solutions based on the analysis. Based on these solutions, we developed a novel transport protocol, A^2RT, which comprises of a transport wrapper and a traffic scheduler. Using simulations, we showed that the transport wrapper achieves reliability with minimum retransmissions under static and dynamic network topology conditions. We also showed that the traffic scheduler helps to achieve the goals of SAC and AAC by maximizing the number of packets that meet the delay deadlines. In summary, A^2RT maximizes the network lifetime and ensures fairness among the actor and sensor traffic while providing a reliable data transfer between separated actors in the network. As a possible direction for future work, we propose the implementation and evaluation of A^2RT in a real-world WSAN. We believe, it will provide better insight into the performance and drawbacks of the protocol.

VI. ACKNOWLEDGMENT

This work was supported by the Department of Science and Technology, New Delhi, India.

REFERENCES

[1] I. F. Akyildiz and I. Kasimoglu, "Wireless Sensor and Actor Networks: Research Challenges," *Ad Hoc Networks*, vol. 2, no. 4, pp. 351–367, 2004.

[2] T. Melodia, D. Pompili, V. C. Gungor, and I. F. Akyildiz, "Communication and Coordination in Wireless Sensor and Actor Networks," *IEEE Transactions on Mobile Computing*, vol. 6, pp. 1116–1129, October 2007.

[3] Ka. Selvaradjou, B. Goutham, and C. Siva Ram Murthy, "Exploiting Resource-rich Actors for Bridging Network Partitions in Wireless Sensor and Actor Networks," in *Proceedings of the International Conference on Distributed Computing and Networking (ICDCN), LNCS 4904*, pp. 416–427, January 2008.

[4] Ka. Selvaradjou, N. Handigol, A. Franklin, and C. S. R. Murthy, "Energy Efficient Directional Routing between Partitioned Actors in Wireless Sensor and Actor Networks," *IET Communications*, pp. 1–14, 2010 (To appear).

[5] J. Postel, "Transmission Control Protocol," *Request for Comments 793*, 1981.

[6] S. Floyd and T. Henderson, "The NewReno Modification to TCP's Fast Recovery Algorithm," *Request for Comments 2582*, 1999.

[7] L. S. Brakmo, S. W. O'Malley, and L. L. Peterson, "TCP Vegas: New Techniques for Congestion Detection and Avoidance," in *Proceedings of the ACM Conference on Communications Architectures, Protocols, and Applications (SIGCOMM)*, pp. 24–35, 1994.

[8] M. Mathis, J. Mahdavi, S. Floyd, and A. Romanow, "TCP Selective Acknowledgment Options," *Request for Comments 2018*, April 1996.

[9] W. R. Stevens, *TCP/IP Illustrated, Volume 1: The Protocols*. Addison-Wesley, 1994.

[10] J. Mo, R. La, V. Anantharam, and J. Walrand, "Analysis and Comparison of TCP Reno and Vegas," in *Proceedings of the IEEE INFOCOM*, vol. 3, pp. 1556–1563, March 1999.

[11] V. C. Gungor, Özgür B. Akan, and I. F. Akyildiz, "A Real-Time and Reliable Transport $(RT)^2$ Protocol for Wireless Sensor and Actor Networks," *IEEE/ACM Transactions on Networking*, vol. 16, no. 2, pp. 359–370, April 2008.

[12] A. Papoulis and S. U. Pillai, *Probability, Random Variables, and Stochastic Processes*. Tata McGraw-Hill, 4th ed., 2002.

[13] R. Jain, *The Art of Computer Systems Performance Analysis: Techniques for Experimental Design, Measurement, Simulation, and Modeling*. John Wiley and Sons, New York, April 1991.

[14] C. A. Floudas and E. N. Pistikopoulos, "Nonlinear and Mixed-Integer Optimization: Fundamentals and Applications," *Journal of Global Optimization*, vol. 12, no. 1, pp. 108–110, 1998.

[15] "The Network Simulator NS-2." Available: http://www.isi.edu/nsnam/ns/.

[16] K. Fall and S. Floyd, "Simulation-Based Comparisons of Tahoe, Reno, and SACK TCP," *ACM SIGCOMM Computer Communication Review*, vol. 26, no. 3, pp. 5–21, 1996.