Synchronization Effects in Mobile Ad Hoc Networks

Harpreet Arora
Lloyd Greenwald
Harish Sethu
and
John Novatnack

Technical Report DU-CS-04-03
Department of Computer Science
Drexel University
Philadelphia, PA 19104
October 2004

Synchronization Effects in Mobile Ad Hoc Networks

Harpreet Arora[a] **Lloyd Greenwald**[a] **Harish Sethu**[b] **John Novatnack**[a]
[a]Department of Computer Science
[b]Department of Electrical and Computer Engineering
Drexel University, Philadelphia, PA 19104

***Abstract*— This paper illustrates how networking protocols can inadvertently exacerbate obstacles to providing real-time guarantees for distributed problem solving in wireless mobile and sensor networks. We analyze the effects of control packet timing on providing quality of service guarantees. Inappropriate timing of control packets gives rise to synchronizations that result in sharp increases and decreases in throughput with small changes in node speed. Such synchronizations can seriously jeopardize network performance with direct effect on real-time guarantees. This paper introduces these synchronizations, analyzes them and suggests ways to modify the control packet timing to overcome them. These analyses include investigating the role of buffering at the network layer and its impact on network throughput. We analyze these effects and evaluate our protocol enhancements through simulation studies.**

I. INTRODUCTION

Any discussion of a Quality of Service (QoS) solution for mobile ad hoc networks (MANETs) [1] must address two predominant features of these networks: mobility and wireless communications. One thing mobility and wireless communications have in common is that both cause link breakage. Link breakage causes throughput to drop [2] while link layer and network layer protocols combine to re-connect the network through breakage detection, link repair, and/or re-routing. As has been previously pointed out [3], in order to provide QoS, networking protocols must trade-off rapid response to link breakage with bandwidth consumed by protocol control messages. These trade-offs directly influence achievable throughput and latency.

While comparing the behavior of various routing protocols we oftentimes observed sharp drops in throughput from nearly 100 percent to nearly 0 percent with only small increases in node mobility. Our analyses reveal synchronization effects that cause these phenomena. The synchronization effects are caused due to frequent link breakages and the routing protocol's attempt to heal them to restore the link. Such phenomena can result in inefficient use of resources, jeopardizing QoS guarantees during critical times. We investigate the causes of these phenomena in detail and suggest modifications to networking protocols to overcome these problems. Our resulting simulations show significant improvements in removing such drastic throughput variations.

We initially observed synchronization effects in large scale random networks. However, the effects can be obscured or averaged out when summarizing the results of multiple sets of large scale simulations. Additionally, due to random movement and the large number of the nodes, these effects are difficult to analyze. To truly understand and explain these effects we study a simple network and mobility pattern where the effects can be carefully manipulated. Solving the problems of this particular network and mobility pattern is not our intention. Our intention is to use this scenario as the basis of a focussed study that illuminates the interaction between link breakages and routing protocols. To enhance the effect, we initially show and analyze it by considering periodic movement of nodes. However, we later show that such a phenomenon is as likely to occur in real world situations where the movement of the nodes is stochastic rather than deterministic. Our simulations prove that even when the movement of the nodes is random, such an effect can cause unexpected drops in the throughput. We consider a real life scenario and show the occurence of this effect influencing the throughput of flows within the scenario. Our study extends to other forms of ad hoc network such as sensor networks [4], [5] where rapid link breakages may be caused by mechanical vibrations, interfering radio signals, or fluctuations in battery power.

This paper is organized as follows: Section II describes the simulation environment and introduces the simulation results. In Section III, we examine the functioning of the nodes during link breaks. We also present analytical expressions for the throughput of the network. Section IV explains the abrupt changes in throughput in terms of the expressions presented earlier. In Section V, we consider scenarios with stochastic movement of nodes and show the presence of synchronization effects in these scenarios. In Section VI, we suggest routing protocol modifications and provide simulation results to evaluate these modifications. We conclude with some discussion in Section VII.

II. RISES AND DROPS IN THROUGHPUT

In the first part of the paper, we focus on a single-path network with one source and one destination. Breaks in this scenario are caused by the periodic movement of one of the nodes. All other nodes remain immobile. We simulate this network and study the behavior of the DSR (Dynamic Source Routing) protocol [6], one of the more popular and mature routing protocols available to MANETs. We study DSR's effectiveness in healing the broken link and its inefficiencies. The results we obtain, however, are applicable to any rout-

ing protocol that employs periodic dissemination of routing information.

Later, we look at complex scenarios with random formation of nodes where nodes move with random speeds and hence gives rise to stochastic breaks rather than periodic. Our simulations show the presence of synchronization effects in these scenarios affecting the throughput of the flows.

A. Simulation Scenario

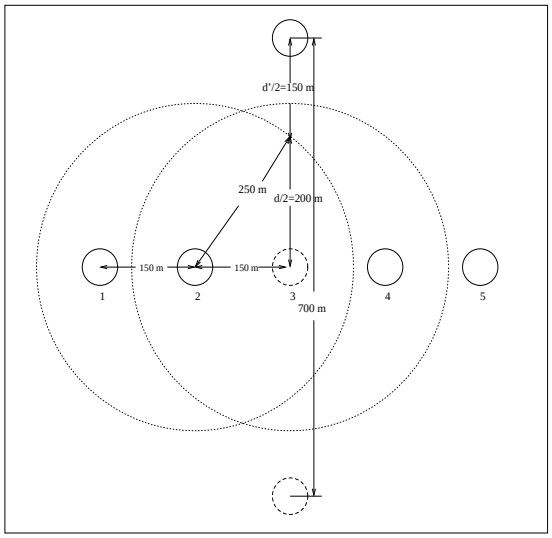

Fig. 1. Initial Simulation Scenario. Node 2 moves periodically making and breaking links across its two sides

Our simulation environment consists of the NS-2 simulator with the CMU Monarch wireless extensions [7]. The target scenario consists of 5 nodes placed as shown in Figure 1. The radio range of each node is fixed at 250 meters. The nodes are placed 150 meters apart such that they are only within the range of their immediate neighbors. The middle node oscillates up and down; at the extreme points it is out of range of the other nodes. This node pauses at the extreme point for some time and then begins moving toward the other extreme. The total distance between the two extremes is 700 meters. The node is within the range of its neighboring nodes for the center 400 meters of this traversal. The first node (node 1) acts as a constant bit rate (CBR) source, sending packets at a constant rate to the destination node (node 5) at the other end. Packets of size 256 bytes are sent at a constant rate of 3 packets/second. DSR is used as the routing protocol. Packets can flow from the source to the destination only when the middle moving node is within the range of its immediate neighbors. We vary the speed of the middle node and its pause time at the extreme points.

B. Simulation Results

Figure 2 shows the throughput of the constant bit rate flow as measured at the destination, for various speeds of the middle node and for two different pause times at the extreme points. The pause time represents the duration of time the node

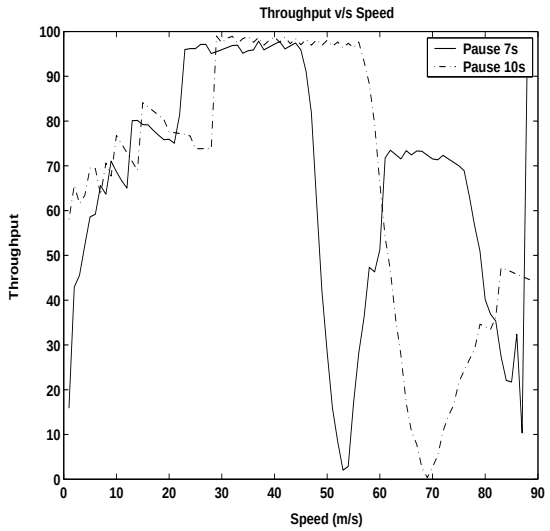

Fig. 2. Throughput v/s Speed for pause times of 7 seconds and 10 seconds. The plots show unexpected rises and drops at certain speed changes.

stops at each extreme point of its traversal before reversing its direction. Figure 2 shows unexpected rises and drops in throughput at certain speed changes. In the initial part of the graph, for low speeds, we see abrupt rises in the throughput. At one particular speed increment we see throughput rise sharply to a very high value. At a later speed increment, the throughput drops sharply to nearly 0 percent. This steep drop is a result of a degenerate synchronization effect. We study these effects, their causes and resolutions in subsequent sections.

III. THROUGHPUT AND LINK DETECTION LATENCY

A. Detecting and Healing a Broken Link

To understand the abrupt rises in throughput, we first study the functioning of both the network layer and the link layer when a node detects a broken link. IEEE 802.11 protocol is implemented at the MAC layer. For each packet transmission, the IEEE 802.11 protocol at each node follows an RTS-CTS-data-ACK sequence. In our scenario, when the MAC layer at node 2 is unable to get a response for seven continuous RTSs, it concludes that the link is broken and reports the same to the DSR agent in its network layer. This interaction between the link layer and network layer is termed *link layer notification* [8]. Such interaction is an effective mechanism for reactive protocols like DSR and AODV [9] to quickly detect broken links. Many proactive protocols like OLSR [10], do not by default support link layer notification. These proactive protocols rely completely on the network layer to detect and heal a broken link.

On being notified of the broken link, the DSR agent at node 2 sends an RERR (route error) packet back to the source node (node 1) to report about the link. When the source node receives the RERR packet, the DSR agent at its network layer stops relaying the data packets that originate at its application

agent to its MAC layer. The DSR agent at the source node then starts to search for an alternate path. It also starts buffering the packets arriving from the CBR source at its application layer. The router layer buffer in our simulations has a capacity to hold 64 packets. To search for a new path, the DSR agent at the source node broadcasts RREQ (route request) packets using an expanding ring search. The first packet is broadcast with a TTL (time to live) of 1 so that it only reaches its immediate neighbors. When no response is received within the timeout period (approximately 50 ms), the next RREQ packet is sent with a TTL equal to 16, which is the maximum number of hops from any source to any destination. The node then waits for a longer period of time before sending the next RREQ packet, again with a TTL of 1. The long timeout period is initially set to 2 seconds, then 8 seconds and subsequently 10 seconds. Henceforth, RREQ packets are sent in sets of 2, first with a TTL of 1 and upon its timeout, a second one is sent with a TTL of 16. A long timeout period of 10 seconds is maintained to keep the routing overhead low.

In our simulation scenario there is only one path from source to destination. A link in this path breaks when node 3 moves out of the radio range of its neighbors. Since there is no alternate path to discover, the link is healed and path re-detected when node 3 moves back into radio range of its neighbors and receives an RREQ packet. Since these packets are sent by the source node (node 1) approximately every 10 seconds beyond the first 3 attempts, there can be a delay in the detecting and healing of the broken link after node 3 moves back in range.

We define link detection latency, denoted by L, as the amount of time taken by the routing protocol to detect the link after the node is back in range. L is obviously a characteristic of the routing protocol and is an indicator of the efficiency of the routing protocol. A good routing protocol should have a low value of L while maintaining low control overhead. We show in Section IV that link detection latency is one of the major causes of observed abrupt changes in throughput. Before that, in the following subsection, we develop expressions for link detection latency and throughput for the given scenario.

B. Analytical Expressions

The expressions derived in this section are used in the subsequent sections to explain rises and drops in throughput. We define throughput as the ratio of the number of packets reaching the destination to the total number of packets sent by the application agent. In our simulations, the middle node is moving periodically. Hence we can divide the simulation time into cycles. A cycle consists of the middle node starting from an extreme point, traversing through the reception range of its neighboring nodes, reaching the other extreme and pausing there, until it is ready to go back. With respect to a cycle, throughput can be found by measuring the number of packets arriving at the destination in a cycle and dividing that by the total number of packets that are sent by the source node in that cycle. If we define d as the distance over which the middle node is in range of its neighboring nodes (see Figure 1) and s as the speed of the node, then the total cycle time that the middle node spends in the connectivity of its neighboring nodes is

$$T_{connectivity} = \frac{d}{s} - L \tag{1}$$

where L is the link detection latency as defined in the previous subsection. L is a function of the speed of the node and the time interval between two RREQ packets. To analyze the dependency of throughput on speed, we need to analyze the value of L.

We note that after the source node is informed of the link breakage, it sends RREQ packets initially at intervals of 0 seconds, 2 seconds, and 8 seconds, and subsequently in 10 second intervals. We define I as the maximum interval between two RREQ packets. The link is detected when the moving node receives one of the RREQ packets after the node is back in range. Thus the sum of the time the node is out of range and the time the node is in range before it receives the RREQ packet is an integral multiple of I. We define d' as twice the distance from the point the middle node goes out of range to the extreme point of its trajectory (see Figure 1) and p as its pause time at the extreme point. The link detection latency can then be expressed as,

$$(\frac{d'}{s} + p) + L = Ik$$

or, equivalently,

$$L = Ik - \frac{d'}{s} - p \tag{2}$$

where k is a positive integer chosen such that

$$0 \leq L \leq I \tag{3}$$

that is,

$$|I(k - 0.5) - \frac{d'}{s} - p| \leq 0.5I \tag{4}$$

As soon as a connection is re-established (i.e. path re-detected), the packets that accumulated in the network layer buffer of the source node while the middle node was out of range are transferred to the destination. If there are b packets in the buffer and these packets are transferred at a rate r' (including transmission time, negligible propagation and processing times, and no further queueing delays), then the time available to transfer these packets is,

$$\tau = min(\frac{d}{s} - L, \frac{b}{r'}) \tag{5}$$

The packets in the buffer are transferred utilizing the entire bandwidth of the channel. The number of packets in the buffer b when the middle node just enters communication range depends on the time the node stays out of range, which is inversely proportional to the speed of the node. When the speed is such that

$$\frac{d}{s} - L \geq \frac{b}{r'} \tag{6}$$

the node stays in connectivity for sufficient time to empty the contents of the buffer. When there are no more packets in the

buffer, a steady connection is established between the source and the destination and packets are transferred at a rate r (this rate now includes the packet generation rate, CBR). In this case, the time over which a steady connection is established is given by

$$T_{steady} = \frac{d}{s} - L - \tau \tag{7}$$

Thus the total number of packets transferred while the middle node is in connectivity is given by

$$n_{received} = (\frac{d}{s} - L - \tau)r + \tau r' \tag{8}$$

When the node is moving fast such that,

$$\frac{d}{s} - L < \frac{b}{r'} \tag{9}$$

the contents of the buffer cannot be completely emptied. Hence the packets are transferred only for a time τ. Number of packets transferred during this time is,

$$n'_{received} = \tau r' \tag{10}$$

In general, the number of packets transferred in a cycle is given by combining (8) and (10) as

$$N_{received} = max(0, \frac{d}{s} - L - \tau)r + \tau r' \tag{11}$$

where τ is given by equation (5). The total number of packets sent by the source during a cycle can be divided into the number of packets sent while the middle node was in range and the number of packets sent while the middle node was out of range. The number of packets sent in a complete cycle is thus given by

$$N_{sent} = (\frac{d}{s} + (\frac{d'}{s} + p))r \tag{12}$$

The throughput of the network is calculated from equations (11) and (12) as,

$$throughput = \frac{N_{received}}{N_{sent}} = \frac{max(0, \frac{d}{s} - L - \tau)r + \tau r'}{(\frac{(d'+d)}{s} + p)r} \tag{13}$$

Thus throughput is a function of speed of the middle node,link detection latency, rate at which packets are sent by the source, rate at which the packets are removed from the buffer, size of the source buffer, in-range and out-of-range distances, and pause time of the node at the extreme points. In our simulations, we keep the value of r and r' constant. We evaluate the dependence of throughput on L in the subsequent sections. The phenomena of abrupt changes in the throughput as captured by these equations is also explained in the next section.

IV. EXPLAINING SYNCHRONIZATION EFFECTS

A. Abrupt Rises in Throughput

As seen in Figure 1, the throughput curve shows abrupt rises at certain node movement speed changes. Referring to equation (13), we see that the only factor influencing throughput as a function of speed is the link detection latency (L). L is itself a function of the speed of the node and its value varies between 0 and I. This can be verified from Figures 3 and 4. The graphs of link detection latency show a saw-tooth behavior. The value of L increases gradually to nearly 10 seconds, before dropping sharply. To explain this, we refer to equations (2), (3) and (4). As s increases, L also increases gradually till it exceeds I. Then the value of k is decremented by 1, resulting in a sharp drop in L. In other words, as the speed of the node increases, it enters the region of connectivity a little earlier compared to the time it entered the region with a lower speed. Since the RREQ packets are sent at approximately the same time, the moving node is detected a little later by the routing protocol resulting in an increased value of L. Thus at some speed, the speed-pause time combination is such that the routing packets are sent just before the node is in range, resulting in a link detection latency of nearly 10 seconds. Subsequently, for slightly higher speeds, the RREQ packets are sent immediately after the node in the region of connectivity reducing L to nearly zero.

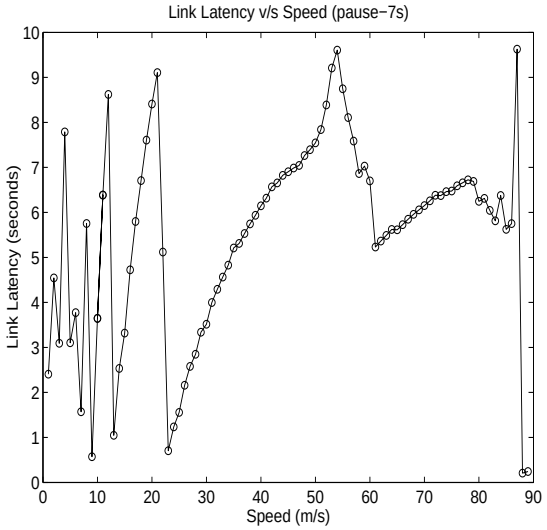

Fig. 3. Link Detection Latency v/s Speed for pause time of 7 seconds. The graph shows a saw-tooth pattern

Now referring back to equation (13), it can be seen that as L increases with speed, the throughput decreases gradually. When L drops suddenly to zero, the throughput rises sharply. The abrupt changes in the throughput are thus a result of the abrupt changes in the value of L, caused by a synchronization between the moving node and the RREQ packets. A small change in the speed can thus cause a significant change in the throughput. Since the value of L is influenced by the value of I, the maximum interval between two RREQ packets, it is important to have a right value of I to obtain an optimum throughput under all scenarios. In section 5, we change the value of I and observe its effects on throughput and link detection latency.

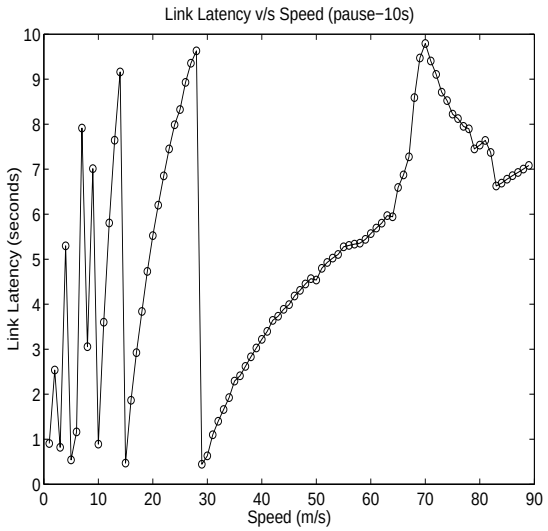

Fig. 4. Link Detection Latency v/s Speed for pause time of 10 seconds. The graph shows a saw-tooth pattern

B. The Region of Constant Throughput

The steady rise in the throughput continues until a specific speed of the middle node, beyond which it increases sharply to a very high value. Very few packets are thus dropped once the node starts to move faster than a particular speed in spite of the node staying out of range for some time (which contradicts the intuition, since high speeds should cause more link breakages and hence a drop in the throughput). The reason for this can be explained as follows. As soon as the source node detects a breakage in the link, it starts buffering the data packets in its buffer at the network layer. Such a high throughput is attained when the middle node is moving at a sufficiently high speed that it is able to come back into the range before the buffer starts to overflow. Once the node is detected and the link is formed, packets within the buffer are transferred at a much faster rate, utilizing the full bandwidth of the channel (2 Mbps), than they are produced by the application agent (approximately 7200 bps). As soon as the packets within the buffer have been transferred, a steady connection is formed between the source and the destination. If the speed of the node is constrained by equation (6), such that the contents of the buffer are emptied out before the node goes out of range, then the packets are re-filled in the buffer when the node is out of range. The amount of time the node spends out of range is given by,

$$\frac{d'}{s} + p + L$$

Since packets are transferred at a rate r during this time, if

$$(\frac{d'}{s} + p + L)r < B \tag{14}$$

then the node is back into the range before the buffer overflows. Thus the range of speeds constrained by equations (6) and (14) define the region where the throughput is constant and has a very high value. The value of buffer B is crucial in maintaining a high throughput and its value should be carefully selected. We believe that the right value of the buffer size will be influenced by the network topology and the kinds of applications supported by the network. The analysis of buffer size is beyond the scope of this paper.

C. The Sharp Drops in the Throughput

Expression (14) indicates that a high throughput is achievable beyond a certain speed because the moving node comes back into the range before the buffer at the network layer of the source node, which holds packets from the application agent, overflows. Theoretically, the throughput should remain steady at near 100 percent beyond this speed, assuming that there is an infinite bandwidth to transfer the contents of the buffer as soon as the connection is established. In an ideal world, the moving node should get detected as soon as it comes back into the region of connectivity. However, in the real world, the routing protocol has constraints. To keep the overhead low, the DSR agent broadcasts RREQ messages only once every I seconds beyond the first three attempts. This I second periodic timeout interval leads to an interesting phenomenon.

In our scenario, the RREQ packet broadcast may just miss the node before the node enters the range for some speed-pause time combination. The node then remains undetected for nearly I seconds before the next RREQ broadcast is made. If the speed of the node is sufficiently high that it again moves out of range before the next set of route request packets are broadcast, the node may never be detected at all.

For some speed-pause time combinations, the periodically moving node in our scenario gets synchronized to the route requests such that the node is not detected during the whole course of simulation. The throughput drops to nearly 0 percent, while the control overhead goes very high as seen in Figure 10. The source node keeps sending RREQ packets every I seconds for the whole duration of simulation. We call this *a point of degenerate synchronization*. Such points of degradation caused by synchronization and poor timing of control packet dissemination can seriously jeopardize the network performance and render the QoS assurances useless.

When the degenerate synchronization occurs, the node is effectively never in the range. Thus from expression (1) we have $d/s - L = 0$ and from equation (5), τ=0. Substituting these values in expression (13) results in a 0 throughput. Since the value of L is influenced by I as seen earlier, we vary the value of I and observe its influence on degenerate synchronizations in section VI.

V. SYNCHRONIZATIONS WITH STOCHASTIC SPEEDS

The degenerate synchronization effects do not necessitate periodic movement of nodes. In this section, we show that the synchronization effect can occur in networks where the speeds of the nodes vary stochastically.

A. Old Scenario

We considered the same scenario as shown in fig 1 with a single moving node. However, the node does not oscillate

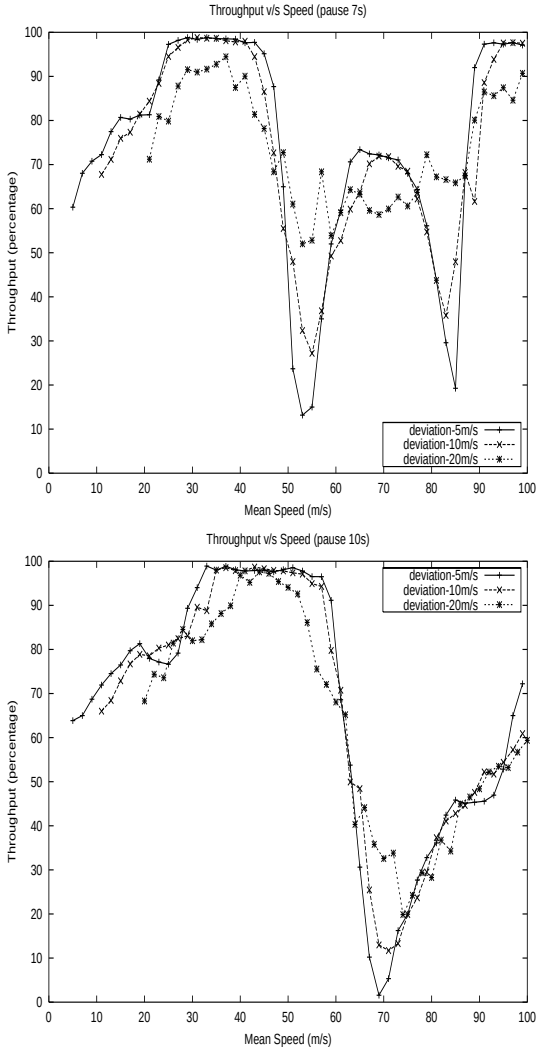

Fig. 5. Throughput v/s Speed for pauses time of 7 seconds (left) and 10 seconds (right) for deviations of 5 m/s, 10 m/s and 20 m/s around the mean speed. The degenerate synchronizations can be clearly seen

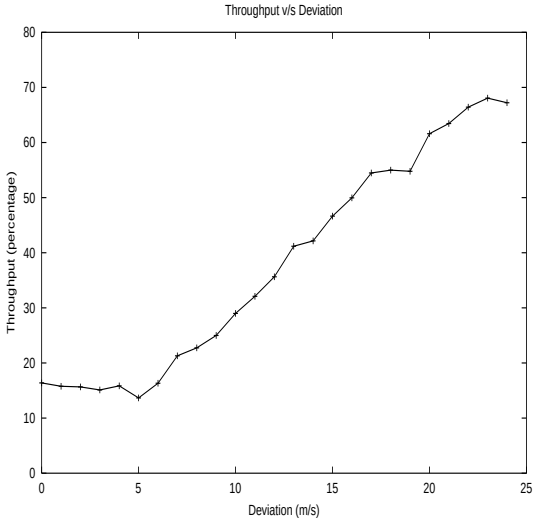

Fig. 6. Throughput v/s Deviation around the denegerate speed of 70 m/s when the node does not pause at the ends. The improvement in the throughout is linear beyond the speed of 5 m/s.

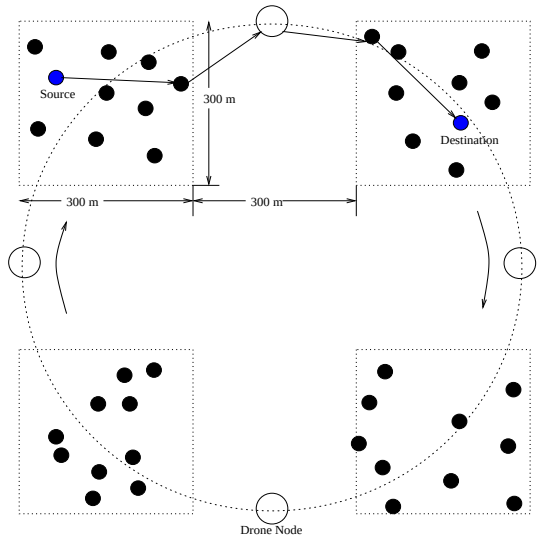

Fig. 7. Real-life simulation scenario. Inter-group communication is only possible via overhead flying drones.

periodically anymore. Instead, after reaching the end point, it chooses a new speed from a uniform random interval. Such speeds can be seen, for example, in a set of cars moving on the highway where the speeds oscillate about the speed limit. The resulting graphs are shown in fig 5. The three curves correspond to deviations of 5 m/s, 10m/s and 20m/s respectively about the mean which is indicated on the X-Axis. Even with a high deviation of 20 m/s, the throughput is not able to recover by a large extent. In real networks such a high variance in the speed is unlikely. Figure 6 shows the change in the throughput with an increase in the variance of the node speed. The graph is a result of averaging 10 random runs. As we can see, the improvement in the throughput is linear beyond a certain speed. The gradient of improvement is low and even with large variances in the speed, the improvement in the throughput is marginal.

B. Real Life Scenario

We also considered a possible battle field scenario involving group based communications. In such a scenario, soldiers move in groups and communication between groups is possible by overhead flying drones. The drones may go in and out of the range of the groups causing inter-group communication to take place intermittently. One such scenario is shown in fig 7. Four groups of ten nodes each are placed randomly within areas of 300 m X 300 m. Four drone nodes revolve with a separation of 90 degrees such that when any drone is within the range of one of the nodes from two adjacent groups, communication between the two groups takes place. We varied the speed of the overhead nodes and monitored the packet delivery ratio

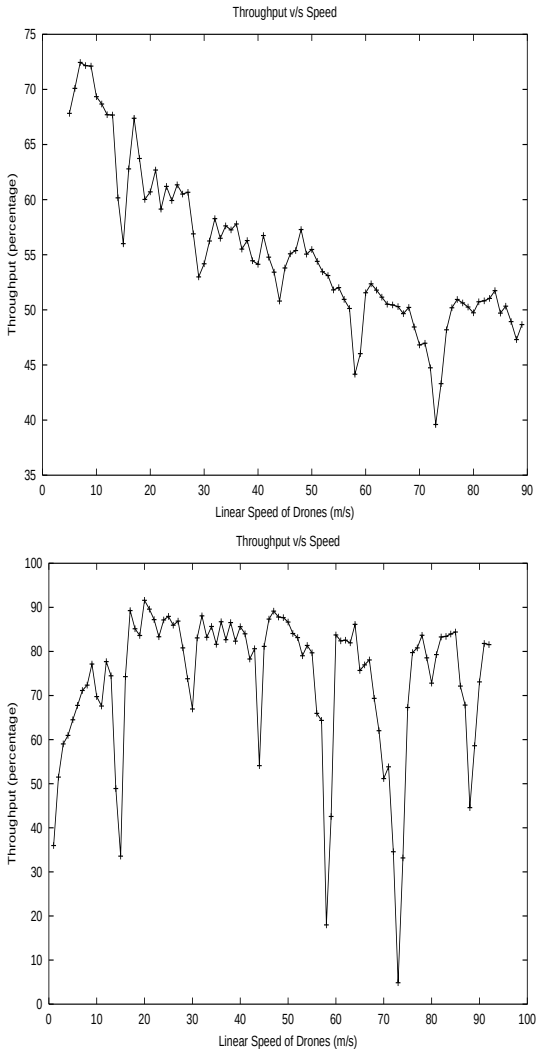

Fig. 8. Throughput v/s Speed for different linear speeds of the overhead flying drones. Left graph is an average of 10 random scenarios. Right graph shows the worst case scenario. Distinct presence of synchronizations can be seen

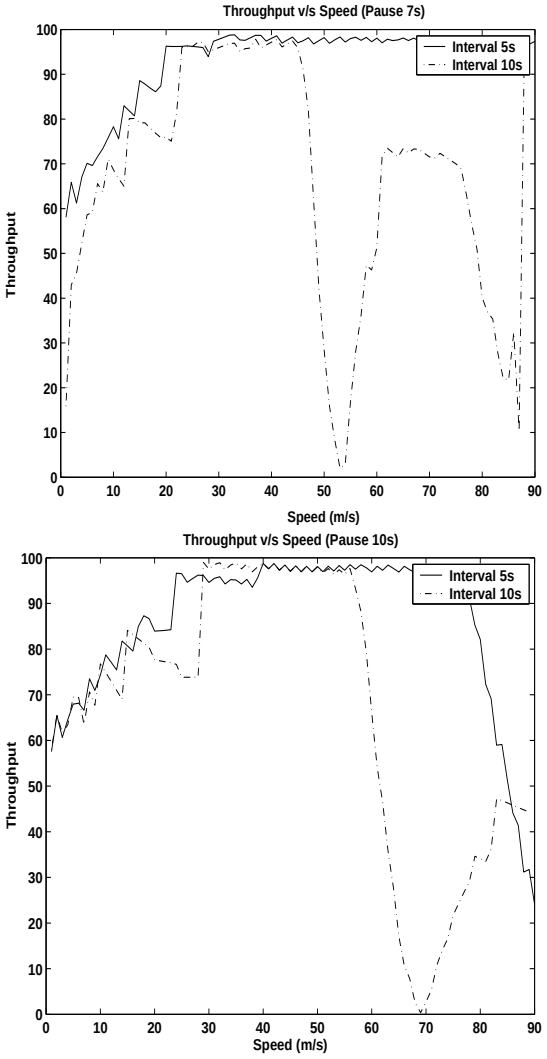

Fig. 9. Throughput v/s Speed for pause times of 7 seconds (left) and 10 seconds (right), for two control packet intervals of 5 seconds and 10 seconds. The improvement in the throughput with double frequency is clearly seen.

of a flow between adjacent groups. Ten random scenarios were considered. Left side of fig 8 shows an average of the ten scenarios. The right figure shows the throughput of the worst case scenario. Although the effect of synchronizations is pacified by the randomness of the topology, it can still be distinctly seen. The effect can be enhanced in some cases as seen in the worst case scenario. For guaranteed QoS, it is important to consider the worst case scenarios otherwise, the unexpected drops in the throughput may cause the QoS mechanisms to fail.

In the next section, we propose some simple techniques to overcome these degenerate synchronizations.

VI. OVERCOMING DEGENERATE SYNCHRONIZATIONS

Degenerate synchronizations would not occur in an ideal world where the node would be detected as soon as it comes back into the range. This can happen if the source node sends route request packets at an infinite rate. Since it is not possible, we identify two practical ways of overcoming the degenerate synchronizations.

A. Reducing the Timeout Interval

From equation (2), we note that the value of I can have a significant effect on the throughput. In our first scheme, we reduce the value of I by half, so that the RREQ packets are sent at a maximum interval of 5 seconds instead of 10. Figure 9 shows the throughput achieved with the reduced timeout interval. The throughput is maintained close to 100 percent for much higher speeds, sufficient for all practical purposes. Obviously this improvement comes at a price. Figure 10 shows the graphs of control packet overhead with the 2 two timeout intervals. The consistently higher overhead with the reduced interval may be intolerable in a congested, bandwidth

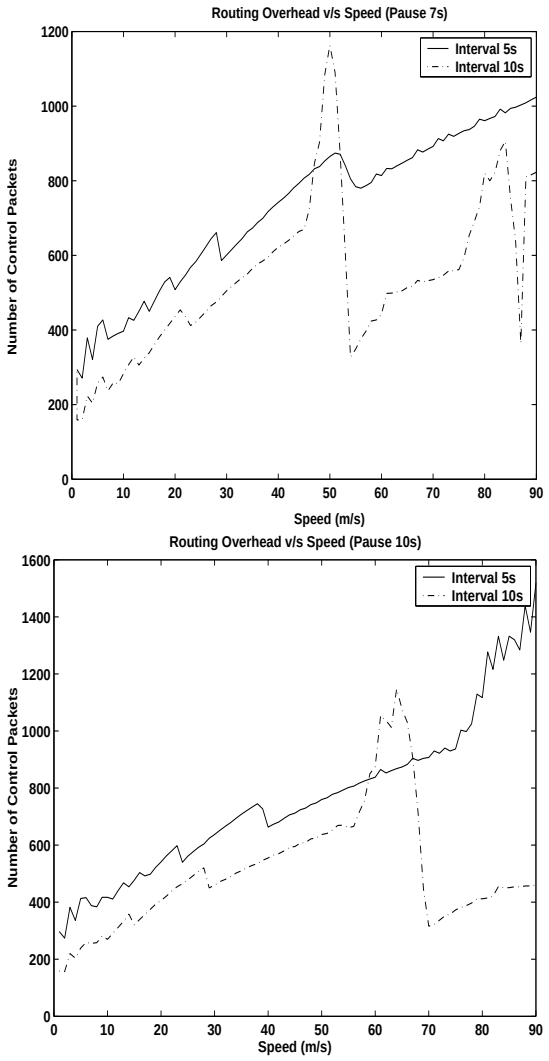

Fig. 10. Control Packet Overhead v/s Speed for pause times of 7 seconds (left) and 10 seconds (right), for two control packet intervals of 5 seconds and 10 seconds. The consistently high overhead with double frequency may not be acceptable

constrained network. The right timeout value should be chosen by taking into account factors such as mobility of the nodes, throughput requirements and amount of overhead that can be tolerated. Making the right tradeoff could be a difficult decision.

B. Randomizing the Timeout Interval

We note that synchronizations are caused because of fixed and periodic nature of DSR RREQ packets. Proactive protocols like OLSR refrain from sending periodic broadcasts of hello messages to avoid another kind of synchronization effect which has also been noted in [11]. In OLSR, or similar proactive protocols, periodic broadcasts of hello messages can lead to a synchronization where all nodes try to broadcast hello messages at the same time. This can cause collisions and since most of these messages are sent over UDP, these messages can be lost, leaving stale topology information with

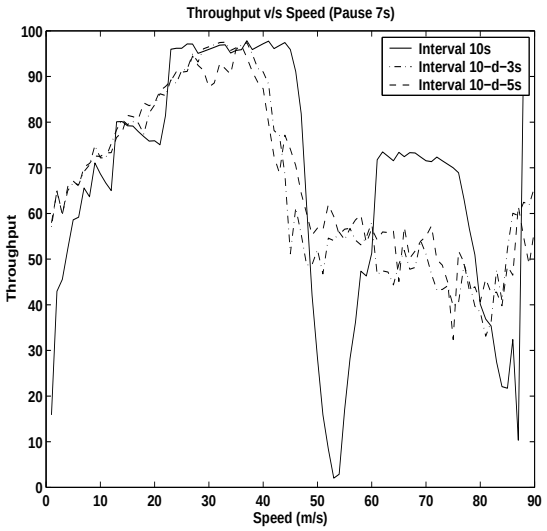

Fig. 11. Throughput v/s Speed for pause times of 10 seconds, comparing throughputs with different control packet intervals. The uniform random time intervals lead to smoothening of throughput graphs

the nodes. OLSR overcomes this problem by adding a random jitter to the hello message interval. The periodic broadcasts of DSR packets in our case is causing a different kind of synchronization which can however be dealt with in a similar way. Therefore in our second scheme, we randomize the control packet interval so that the time after which the next request packet is sent is chosen from a uniform random distribution. We experimented with 2 different intervals: 10 seconds with a deviation of 3 seconds (uniform distribution of 7 seconds to 13 seconds) and 10 seconds with a deviation of 5 (uniform distribution of 5 seconds to 15 seconds). The results are shown in Figure 11. The graphs certainly show a smoothening of the throughput, though now, the peak throughput is not sustained as consistently as for fixed control frequency. The control overhead is nearly the same as that with the fixed interval of 10 seconds. This scheme guarantees a minimum throughput under worst case scenarios, and can be used for critical applications where the failure of the network can lead to a catastrophe.

VII. DISCUSSION

In this paper we have examined the influence of timing and frequency of control packets and buffer size at the network layer on the throughput of the network. Small changes in speed of the nodes can cause large changes in the throughput due to synchronizations between movement of the nodes and control packets. The synchronizations can cause abrupt rises as well as drops in the throughput. The effect has a direct impact on the achievable QoS for mobile or sensor networks. The results produced here are especially important for time-critical applications in crucial scenarios, where overlooking of such effects can result in catastrophic failures.

We identified points of degenerate synchronization where the throughput can unexpectedly drop to 0 percent and studied some ways to overcome these points. This synchronization

effect has not been identified previously. Its appearance in random and more complex scenarios is either ignored or obscured by running multiple sets of simulations and averaging the results out. Although the effect is explained in the context of DSR protocol, it applies in general to any routing protocol, which exhibits a periodic pattern in disseminating the control information. We initially studied the problem with respect to a simple scenario with periodic motion of node, but later we showed that the problem may appear in real life scenarios. The problem is studied in context of ad hoc networks but it is equally likely to occur in sensor networks which have the property of exchanging messages periodically. The solutions to the problem are built from known solutions in the wired world, which are associated with a similar, though not the same problem as discussed in [11]. In our ongoing work, we are evaluating some adaptive schemes where the timeout interval is calculated based on the past history and other factors such as the speed of the nodes and the average time to heal a link. Such adaptive schemes, we believe, will result in substantial improvement in the throughput of the network while keeping a low overhead.

In this work, we do not consider the effect of an alternate path when the original path is lost. The availability of an alternate path may reduce the degenerations to some extent, though they might still affect the network performance. We plan to study this effect in our continuing work.

REFERENCES

[1] J. Sun, 'Mobile ad hoc networking: an essential technology for pervasive computing", *Proceedings of International Conferences on info-tech and Info-net*, vol. 3, 2001, pp. 316-321.

[2] H. Arora, H. Sethu, "Performance analysis of effects of mobility patterns on network performance in mobile ad hoc networks," *Proceedings of Applied Telecommunications Symposium*, SCS, April-2002, San Diego, CA

[3] S. Chakrabarti and A. Mishra, "QoS issues in ad hoc wireless networks",*IEEE Communications Magazine*, vol. 39, issue 2, February-2001, pp. 142-148

[4] I. Akyildiz, Weilian Su, Y. Sankarasubramaniam and E. Cayirci, " A survey on sensor networks," IEEE Communications Magazine, vol. 40, issue 8, August 2002, pp. 102 -114

[5] L. Greenwald and H. Sethu, "On Scheduling Sensor Networks", *AAAI/KDD/UAI-2002 Joint Workshop on Real-Time Decision Support and Diagnosis Systems*, Alberta, Canada, July 2002, pp. 83-84.

[6] D. Johnson, D. Maltz, Y. Hu, J. Jetcheva, "The Dynamic Source Routing protocol for mobile ad hoc network", http://www.ietf.org/internet-drafts/draft-ietf-manet-dsr-07.txt, IETF internet draft, Feb. 2002

[7] http://www.isi.edu/nsnam/ns

[8] S. Wu, S. Ni, Y. Tseng and J. Sheu, "Route maintenance in a wireless mobile ad hoc network", Proceedings of the 33rd Annual Hawaii International Conference on System Sciences, 2000 pp. 3015-3024

[9] C. Perkins, E. Belding-Royer, S. Das,"Ad hoc on Demand Distance Vector Routing", http://www.ietf.org/internet-drafts/draft-ietf-manet-aodv-11.txt, IETF internet draft, Feb. 2002

[10] T. Clausen, P Jacquet et al, "Optimized Link State Routing Protocol", http://www.ietf.org/internet-drafts/draft-ietf-manet-olsr-06.txt, IETF internet draft, Sept. 2001

[11] S. Floyd, V. Jacobson, "The synchronization of periodic routing messages,"*IEEE Transactions on Networking*, vol. 2, issue 2, pp. 122-136