

Optimizing Parallel Execution of Detailed Wireless Network Simulation*

Zhengrong Ji, Junlan Zhou, Mineo Takai, Jay Martin, Rajive Bagrodia

University of California Los Angeles

Parallel Computing Laboratory

3809 Boelter Hall, Los Angeles, CA 90095-1596

{jizr,zjl,mineo,jmartin,rajive}@cs.ucla.edu

Abstract

With Parallel and Discrete Event Simulation (PDES) techniques, the runtime performance of detailed wireless network simulation can be improved significantly without compromising fidelity of the simulation results. However, modelling characteristics of wireless communications such as signal propagation and interference may severely hinder the potential speedup yielded by PDES. This paper proposes various optimization techniques to address three major concerns in achieving efficient parallel execution of wireless network simulation: i.e., (1) reducing communication and computation overhead of simulating signal propagation across multiple logical processes; (2) reducing synchronization overhead among logical processes; (3) minimizing event scheduling overhead within individual logical processes. These techniques have been implemented in a parallel version of GloMoSim and QualNet. The experimental results with mobile ad hoc networking scenarios demonstrate that the proposed optimization techniques can improve the performance of parallel wireless network simulation by up to an order of magnitude.

1. Introduction

Wireless communication is playing an increasingly important role in the design of contemporary digital communication systems. Rapid growth in both complexity and deployments of such wireless networks has created a need for simulation tools that can accurately predict the performance of next generation wireless network devices, protocols and deployed services. However, high fidelity simulation of wireless networks is computationally intensive, as it requires proper modeling of signal propagation and interference that can increase the number of events in the simulation as many as $O(N^2)$ where N is the number of network nodes. Its direct consequence is computational impracticality of high fidelity discrete-event simulation for even medium size wireless networks.

With PDES techniques, the execution time of wireless network simulation can be reduced significantly without compromising fidelity of the simulation results. However, characteristics of wireless communications such as signal propagation and interference may severely hinder the potential speedup yielded by PDES. For instance, simulation of radio signal propagation requires scheduling of an event to every potential receiver, resulting in creation of a large number of signal arrival events. Although parallel execution can benefit from many of these events, it requires frequent communications and tight synchronization among logical processes (LP) in order to maintain causality of sequences of events. As a result, the runtime performance improvement by PDES is typically much less than the number of processors used for the execution.

To improve the runtime performance of parallel wireless network simulation, the following three concerns need to be addressed: (1) reduction of communication and computation overhead of simulating signal propagation across multiple LPs, (2) reduction of synchronization overhead among multiple LPs, and (3) minimization of event scheduling overhead within each LP.

This paper discusses each of the above three issues and proposes optimization techniques to substantially improve the runtime performance of wireless network simulation. These techniques have been implemented in a parallel version of GloMoSim and its commercial version QualNet [1]. The experimental results demonstrate that the proposed optimization techniques can improve the performance of parallel wireless network simulation by up to an order of magnitude.

The remainder of this paper is organized as follows: Section 2 describes related work on parallel simulation of wireless networks. Section 3 discusses simulation of signal propagation across multiple LPs and proposes effective techniques to reduce the associated communication overheads. Sections 4 and 5 respectively present techniques to extract greater lookahead to exploit additional parallelism among LPs, and reduce event scheduling overhead within each LP. Section 6 concludes the paper with future research directions.

*This work is supported by the US Department of Defense/DARPA under Contract N66001-00-1-8937 "Maya: Next Generation Performance Prediction Tools for Global Networks".

2. Related Work

There have been several studies to parallelize the execution of network simulation [9, 10]. PDNS [9] is a parallelized version of ns-2, a commonly used simulator. PDNS can execute multiple instances of ns-2 via the Georgia Tech RTIKIT. Genesis [10] is another parallel simulator based on ns-2 and runs multiple instances of ns-2 in an iterative manner such that the network performance metrics collected in each instance converge. These simulators do not require modifications in the networking models for the parallel execution, but they can execute only wired network models in parallel; to our best knowledge, they have not been used for simulation of large wireless networks. SWAN [7, 8] is a parallel simulator for wireless mobile networks that employs a conservative synchronization algorithm. The study [7] shows that it can achieve good parallel performance by abstracting out parts of physical layer models and simplifying MAC state transitions.

GloMoSim [12] is a simulation tool designed for large scale mobile ad hoc networks with highly detailed physical layer propagation models. Built on top of PARSEC [3], it supports parallel execution of wireless network simulation. This paper uses GloMoSim and QualNet, the next generation of GloMoSim, as the base wireless network simulators because their model libraries include highly detailed physical layer models. The modeling differences of radio physical layer in ns-2 and GloMoSim, as well as the impact of such differences on the simulation results can be found in [11].

Many past studies on parallelizing wireless network simulation described above use abstract PHY layer models and neglect effects of signal interference. In those studies, radio signals weaker than the sensing threshold are not delivered to a node. These simplifications significantly reduce the number of events generated to simulate wireless communication. However, as demonstrated in [11] and [5], omitting such details could result in misleading experiment results and a simulator with such simplifications does not serve as a good analysis tool for wireless networking studies. In the studies presented in this paper, we take into consideration effects of signal interferences and carefully adopt a propagation limit, which is much larger than the sensing threshold used by previous studies. This propagation limit guarantees the aggregated interference power neglected at a receiver will not exceed the thermal noise level inside a receiver. Although it preserves the fidelity of wireless network simulation, this propagation limit produces limited performance gains and adds complexities to improving parallel runtime performance of the simulator. If this propagation limit is applied, performance improvements that can be achieved by the optimization techniques proposed in the past studies diminish. This motivates us to propose a number of novel optimization techniques, which can substantially improve

the parallel performance of such detailed wireless network simulation without compromising fidelity of the simulation.

3. Reducing costs of simulating signal propagation across multiple LPs

3.1. Propagation limit

Due to the broadcast nature of radio transmissions, a radio signal transmitted towards a specific node can interfere with communications of many nearby network nodes and contribute to their noise level. To model interference, for each signal transmitted, two events (signal arrival and end events) will be scheduled to each of the nodes in a wireless network. Propagation limit is a known technique to reduce the complexity of modelling signal interference. With propagation limit, signal arrival or end events will be scheduled to a node only when the received power of the corresponding signal is above a certain threshold. However, although the power of each interference signal is not high enough to affect the node's behaviors, accumulation of such signals could cause a node to sense the channel to be busy thereby forcing the node to backoff from transmitting, or failing to receive a signal of interest due to poor SNR (Signal to Noise Ratio). Simulation ignoring this effect could produce unreliable results [11]. Thus it is important to gauge the impact of interference signals and apply a propagation limit with negligible impacts on accuracy of simulation results. In this study, we use a propagation limit of -111.0dBm, with which the aggregated power of the neglected interference signals is guaranteed to be less than the average thermal noise level of the receiver [5]. With this propagation limit, a radio signal can propagate up to 2.11km away from the transmitter with the two ray pathloss model.

3.2. Simulation of signal propagation across multiple LPs

In addition to reducing the cost of modelling signal interference, the propagation limit can also reduce the number of nodes to be scanned when used in combination with partitioning. A common approach to parallelize wireless network simulation is to divide the target network into multiple geographical areas of equal size, each represented by an LP. Each LP simulates the complete network stack of all nodes contained within its partition. To simulate signal propagation, when a node in a partition transmits a radio signal, the corresponding LP will send an event to each remote LP. Upon receiving the event, the remote LP will scan the list of nodes within its partition, and schedule signal arrival and end events for nodes at which the received signal power is above the propagation limit. It results in a large number of pathloss calculations which are computationally expensive and incurs high scanning cost. Since the terrain is geographically partitioned into grids, when a signal is transmitted by

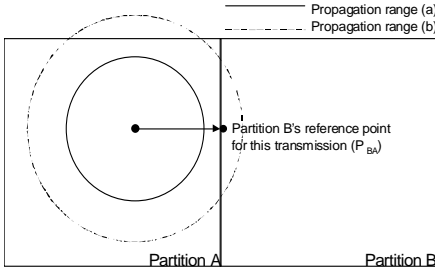


Figure 1: Effect of multiple partitions.

a node within a partition, the corresponding LP can compute path loss values for the reference points of every remote partition. The reference point of a remote partition is chosen so that it yields the minimum path loss value for all locations in the remote partition. If the power of a signal received at the reference point in a remote partition is lower than the propagation limit, the remote partition will not receive a message of the signal transmission. For instance, consider a case with two partitions (A and B) where a node in Partition A transmits a signal as depicted in Figure 1. Depending on the radio power at the reference point in Partition B (P_{BA}), Partition A may or may not schedule an inter-LP event for the signal transmitted. Assuming that each partition has equal number of nodes, the number of nodes scanned for the signal is reduced by half compared to the single partition case, if the signal is not reachable to P_{BA}.

3.3. Performance evaluation

Figure 2 shows the reduction in the total number of events by applying the propagation limit. In this set of experiments, one LP is used to simulate the entire target network. Note that the number of events executed remains invariant with the number of LPs. Nodes in the target network are randomly placed on a flat terrain, at a density of $50,000m^2$ per node. This placement results in terrain sizes of $2.23 \times 2.23km^2$, $7.07 \times 7.07km^2$ and $22.36 \times 22.36km^2$ for 100, 1,000, and 10,000 node cases respectively. CBR (Constant Bit Rate) traffic at a rate of 20 packets per second is given to 15% of network nodes chosen randomly in the networks. LAR (Location Aided Routing protocol) scheme 1 [6] and IEEE 802.11 DCF are used for all the nodes in the network. As each signal needs to be propagated up to 2.11km with the propagation limit of -111dB and the two ray pathloss model, the effect of the propagation limit is very small for the 100-node case, while 94% of events are eliminated for the 10,000-node case. The percentage of reduction in execution time for single partition is almost identical to that of reduction in the total number of events executed. This result proves that although each radio signal needs to be propagated to a wide area, limiting the propagation of each signal has a significant impact

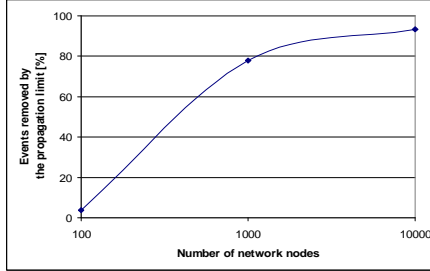


Figure 2: Effect of the propagation limit on the number of events.

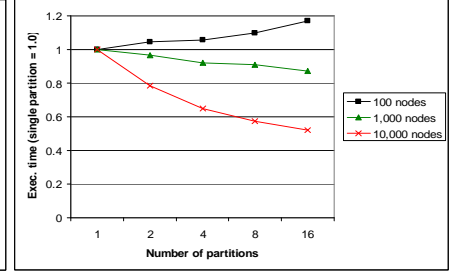


Figure 3: Execution time changes with multiple partitions.

on the runtime performance of simulation, particularly for large-scale networks.

Figure 3 shows the speedup of simulation execution with various numbers of partitions. To better quantify the total amount of reduction in computation and communication by the propagation limit, all partitions are executed on a single processor in this experiment, in which all event processing and node scanning are carried out. In the 100 node cases, the simulated terrain size ($2.23 \times 2.23km^2$) is too small compared to the propagation range of the signal (2.11 km) and most signals can still travel across the entire terrain. Due to the overhead of context switching between LPs, simulation execution slows down a little as more partitions are created. In the larger network cases, the propagation limit significantly reduces the number of inter-LP events and nodes scanned. Compared to the sequential simulation execution without the propagation limit, speed-up of up to 6 and 20 times can be achieved (using 16 partitions) with the propagation limit for the 1000 and 10,000 node cases respectively.

To preserve the modeling fidelity, a conservative propagation limit is used in our studies. A large number of events are still generated at MAC and PHY layer, and simulation preserving such details at MAC and PHY layer has limited parallelism. In order to optimize parallel execution of such detailed wireless network simulation, there exist two other challenges as described in Section 1, i.e. reducing synchronization overhead among multiple LPs, and minimization of event scheduling overhead within each LP. This motivates us to propose a number of novel optimization techniques to be discussed in the next two sections.

4. Extracting greater lookahead

4.1. Synchronization protocol

Parallel simulation requires the use of a synchronization protocol among LPs to guarantee that causality errors are not introduced in the simulation results. An excellent overview of the problem of parallel discrete-event simulation and a comprehensive discussion of commonly used synchronization protocols can be found in [4]. In this paper, we use the synchronous conservative protocol. Each

LP defines its lookahead as the minimum duration (measured using the simulation clock) for which it will not send any messages on its outgoing links. Periodically, a global minimum of all LPs' simulation time plus their lookahead values will be computed, referred to as the ceiling of the next time window. The LPs can process all events within this time window safely without the need of additional synchronization.

When using this synchronous conservative protocol, as long as there is one LP with poor lookahead, other LPs can process only few events per global synchronization and cannot exploit great parallelism. Therefore, the ability of each LP to "look ahead" is important to the performance of parallel simulation. However, in wireless network simulation with detailed PHY and MAC layer models, nodes could transit from state to state very frequently due to fluctuation of channel conditions. On one hand, such state changes must be considered when extracting lookahead at a node so that no causality errors could occur. It adds complexity to extracting lookahead in wireless network simulation with detailed PHY and MAC layer models. On the other hand, exploiting characteristics of state changes of nodes being simulated with detailed PHY and MAC layer models could yield additional sources of lookahead that are absent in simulation with abstract PHY and MAC layer models. The following subsections discuss techniques of extracting lookahead in wireless network simulation with detailed PHY and MAC layer models, given that all nodes in a wireless network follow IEEE 802.11 standard [2], and present the performance improvement obtained by exploiting these sources.

4.2. Lookahead extraction

4.2.1 Lookahead at MAC layer

According to the 802.11 standard, all network nodes must perform virtual carrier sensing in addition to physical carrier sensing. A node has to wait for at least one type of inter frame space defined in the 802.11 standard before it starts to transmit a frame. Depending on the current status of the node's MAC or physical layer, simulation can determine the type of inter frame space to be used and use it as the minimum lookahead value. In addition to inter frame spaces, the duration of the current frame being received can also be added to the node's lookahead, as the node cannot initiate a transmission until the completion of the reception of current frame. A third source of lookahead at MAC layer is the backoff timer. According to the 802.11 standard, a node has to sense the channel before starting to transmit a packet (requested by the network layer). If the channel is busy, the node will go through a backoff procedure, of which the duration is uniformly distributed between zero and CW (Contention Window).

Table 1: Terrain Size for the High and Low Density Experiments

# nodes	# CBR sessions	High Density (20000m ² /node)		Low Density (50000m ² /node)	
		Terrain Size (m ²)	Partition Element Size (m ²)	Terrain Size (m ²)	Partition Element Size (m ²)
100	15	1414x1414	354x354	2236x2236	559x559
1000	150	4472x4472	1118x1118	7071x7071	1768x1768
10000	1500	14142x14142	3536x3536	22360x22360	5590x5590

4.2.2 Lookahead at PHY layer

The simplest source of lookahead that can be extracted is the time defined for an 802.11 radio to transit from receiving to transmitting. This transition produces a delay between the time when the 802.11 MAC layer decides to transmit and the time when the transmitted signal is actually transmitted. This delay stated in the 802.11b spec is 5us, which is far too small to yield good parallel performance.

Fortunately, lookahead much larger than the radio transmitter turnaround delay can be extracted at the PHY layer using information of interference signals. As defined in the 802.11 standard, a node has to sense the channel before starting to transmit a packet. If the channel is busy, the node will back off. By looking at the current interfering signals (from far away transmissions) received at a node, simulation can predict the earliest possible time that the radio of the node will transit to "Idle State". Since the node cannot transmit when the channel is sensed busy, the duration when channel is busy can be used as a minimum lookahead of the specific node.

4.2.3 Lookahead from cross-interaction between PHY and MAC layer

An even larger lookahead can be extracted exploiting the cross interaction between the PHY and MAC layers. The backoff timer of a node pauses when the channel transits from idle to busy during the node's backoff period. The backoff timer will not resume until the channel becomes idle again and remains idle for a duration of DIFS. Before its backoff timer becomes zero, a node cannot transmit. Hence, if a node's backoff timer is paused, based on the list of interference signals received by the node, simulation can project the total duration of the channel being busy and predict the earliest possible time when the node can initiate a transmission. This duration can be used as a lookahead value of the node. This technique is illustrated in the Figure 4. The channel becomes busy at t_0 and t_2 (due to arrival of interference signals) and transits from busy to idle at t_1 and t_3 (due to end of interference signals). The backoff timer is T_{BO} at t_0 and $t_2 - t_1$ is shorter than DIFS. The backoff timer will not become zero until time $t_3 + T_{BO} + DIFS$. Therefore, the duration between the current simulation time and time $t_3 + T_{BO} + DIFS$ can be used as the lookahead value of the node. This technique and the one presented

in the previous subsection to extract lookahead can only be done with detailed PHY layer models.

4.3. Performance evaluation

Table 1 summarizes the basic parameters that were used in the experiments. The nodes in the network were placed on a flat plane partitioned into grids. Only one node is placed in each grid cell, whose location is randomly chosen within the cell. A number of CBR sessions are set up between randomly selected source and destination nodes. The source and destinations for each session are chosen randomly to be between 1 to 8 grid squares away in both horizontal and vertical directions. Each CBR session generates 10 packets per second with a packet size of 512 bytes. The propagation limit radius for transmission is 2118m (4236m diameter) and the exclusionary "sensing" radius is 670m (1340m diameter). These test cases are executed using the synchronous protocol on a Sun Ultra Enterprise 6500 running at 336Mhz using 4, 8 and 16 CPUs.

The parallel speedup for the low density cases is shown in Figure 5. These speedup results are with respect to the best sequential implementation and thus are on top of the speedup already achieved with the techniques described in Section 3. The 100 node test cases achieve the maximum speedup of approximately 3.8 times on 16 CPUs with a performance plateau beyond 8 CPUs. The plateau occurs because when running on 16 CPUs, the small 100 node case has only on average $100/16 = 6.25$ network nodes to simulate per CPU and synchronization costs begin to dominate the execution time of the model. This plateau contrasts with the observed improvements in the 1000 and 10000 node cases, where the parallel performance of the network model continues to improve from 8 to 16 CPUs, peaking at about 4.5 and 6 times speedup for the 1000 and 10,000 node models respectively.

Figure 5 also shows an anomalous crossing of the performance curves for the 100 and 1000 node test cases. The Figure 6 plots the probability density function (PDF) of time window size for test cases executed using 8 CPUs. For the 100 node case, the size of time windows spreads all the way up to the maximum lookahead value of around 250us, while in the 1000 node case, almost all time windows are below 75us. For the 10,000 node case, almost all windows are below 25us. There are two factors that affect the parallel speedup of simulation: the number of events executed per time window and size of time window. When the number of nodes in a target network increases, the number of events grows and size of time windows decreases as shown in Figure 6. Depending on which of these two factor dominates, the parallel speedup of simulation fluctuates as the size of target network increases.

The speedup for the high density experiments is shown in Figure 7. Compared to sequential simulation, parallel

simulation of the same network scenarios achieves the maximum parallel speedup of 4, 5.8, and 4.6 for 100, 1000, and 10000 node cases respectively. The parallel performance of 10000 node case with high node density is worse than the corresponding low node density case. Figure 8 compares the PDF of ratio of the maximum over the average number of events per processor for 10k nodes cases with high and low node density. It indicates that the high density test case has a worse instantaneous load balance. As a result, less parallel speedup can be achieved compared to the corresponding low density case.

5. Reducing Intra-LP events

Although the runtime performance of parallel wireless network simulation is improved significantly by reducing scanning costs (see Section 3) and increasing parallelism (see Section 4), it is equally important to minimize the overhead within each LP. One observation made from the existing wireless network simulators is that in order to model the fluctuation of channel physical status (busy or idle) or interference signal power, each transmitter must schedule signal arrival and end events for every other node within the propagation range. Simulation will update changes to the aggregated power of signals or physical status whenever such an event occurs. These potential changes must be simulated in order to accurately model the wireless MAC/PHY device operations, such as evaluation of packet error and a backoff procedure. However, the high fidelity comes at the cost of scheduling a great number of signal arrival and end events, which incurs significant event scheduling overhead, especially when each LP is assigned many wireless nodes.

Clearly the simulation overhead at each LP will be greatly reduced if the scheduling for most of the signal arrival and end events can be avoided. To achieve this goal, the dependency between these events and the device operation of wireless nodes is investigated first. The only way a node interacts with other wireless nodes in the network simulation is via transmitting a signal. If it is ensured that all the incoming signals are processed in orderly fashion, the behavior of a node will remain the same, which also means that this node will causally affect other nodes in correct manner subsequently. Thus, the remaining goal is to ensure that eliminating arrival and end events of incoming signals does not alter the operation of a node.

There are three cases in which an incoming signal may alter a node's operation: first, the receiver could lock on to the signal and try to receive the corresponding packet; second, the receiver could fail to receive another packet due to high aggregated power of received interference signals; finally, the receiver could sense the channel busy while performing physical carrier sensing; In the first case, a signal arrival event must be scheduled because a transmitter cannot tell for certain if the packet will be locked on by

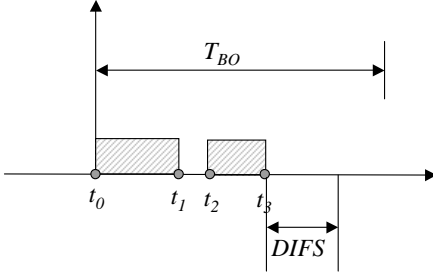


Figure 4: look extraction exploiting cross interaction PHY and MAC layer

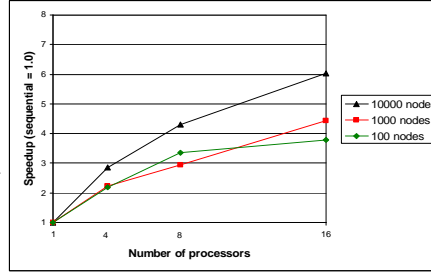


Figure 5: Lookahead extraction: low density test case.

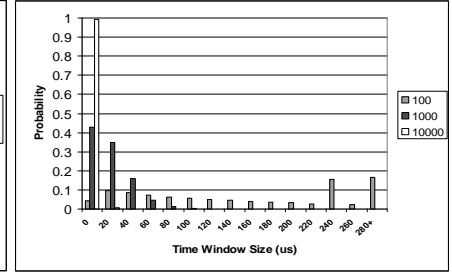


Figure 6: low density window size PDF.

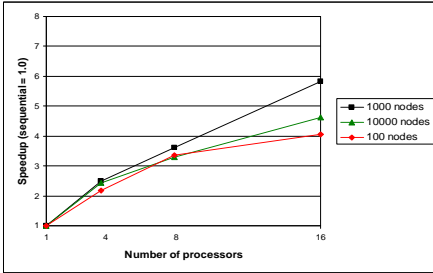


Figure 7: Lookahead extraction: high density test case.

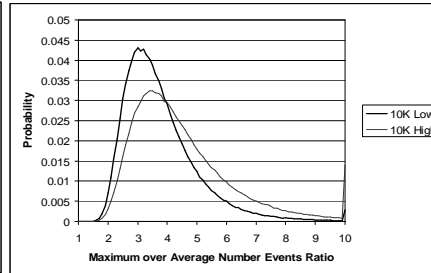


Figure 8: 10K Case: Maximum number events over the average per processor ratio PDF.

```

BOOL IsPacketDefinitelyNonReceivable(Packet p, Node s)
{
  IF (reception_power(p, s) < RECEIVING_THRESHOLD)
    Return TRUE;
  ELSE IF (CurrentStatus(s) == TRANSMITTING
    && TxEndTime(s) > ArrivalTime(p, s))
    Return TRUE;
  ELSE IF (CurrentStatus(s) == RECEIVING
    && RxEndTime(s) > ArrivalTime(p, s))
    Return TRUE;
  ELSE
    Return FALSE;
}

```

Figure 9: Routine to decide if a packet p is non-receivable at node s .

another node for reception until that actually happens. In the remainder of this section, we will propose solutions for the latter two situations to eliminate signal arrival and end events whose absences do not violate the causality constraints of the simulation, and we use the causal relationships among the events to illustrate how causality is preserved in those situations.

5.1. Eliminating events for non-receivable signals

A signal (or its corresponding packet) transmitted by a node is non-receivable by a neighbor if its arrival will NOT trigger the neighbor to lock on and receive the corresponding packet. A signal can be non-receivable to a neighbor in the following cases. First, the signal strength is below the neighbor's receiving energy threshold and cannot be detected by the neighbor's radio. Second, the neighbor is either transmitting its own packet or receiving another packet when the signal arrives. Figure 9 illustrates how a transmitter decides whether the signal is non-receivable to other nodes. Note that non-receivable signals can only affect the neighbor's successful reception of another packet or its channel status (idle or busy) (the latter two of the three ways mentioned earlier). We argue that the signal arrival and end events for non-receivable signals can be completely eliminated without affecting the correctness of the simulation by using the following approach. The decision of whether a packet is definitely non-receivable at each node is made at the beginning of its transmission. The signal will

be appended to the signal history of every node regardless of whether it generates signal arrival and end events for that node. Each signal recorded in signal history contains information such as signal arrival and end time, received signal power etc. The next two subsections describe this approach in more detail.

5.2. Non-receivable signals vs. evaluation of packet reception

Figure 10 illustrates the happen-before causal relationships among the signal transmission beginning (end) incident, and the signal arrival (end) incident (corresponding to a non-receivable packet), as well as the end of reception for some other packet at the receiver. As one can see, a packet transmission happens strictly before its signal arrives at the receiver, due to nonzero propagation delay. The other signals that affect the packet reception must arrive before the end of the packet reception period, which means that their corresponding packet transmission incidents must have happened before the end of packet reception at the receiver.

It is intuitive that as long as the history of other interference signals overlapping with the packet reception is known (i.e. all causal events happened before are known) at the end of the packet reception, the success (or failure) evaluation to the packet reception can be bulk-processed at the end of the packet reception without violating the causality constraints. This is because that one can simply reproduce those processings according to the order in which the sig-

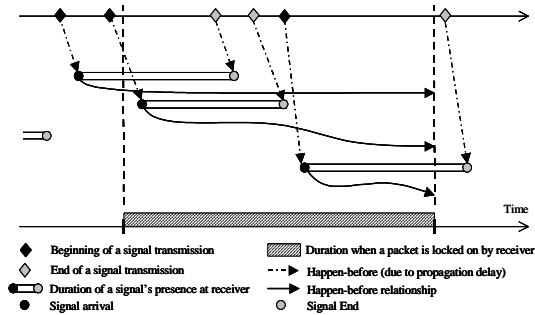


Figure 10: Causal orders of signal arrival/end events with reference to the end of packet reception.

nals arrive and end. Note that signals ended before the end of the packet reception can be removed from the signal history.

5.3. Non-receivable signals vs. timer operation

Besides changing the fate of a packet being received at a node, non-receivable signals can also affect the physical status (idle or busy) at the node, which may in turn affect when the node can transmit its own packet. Using the 802.11 DCF as an example, the following describes how the DIFS/EIFS and Backoff (BO) timers are implemented in existing wireless network simulators. When a node has a packet to transmit and the channel is determined to be idle at the same time, a timeout event will be scheduled to occur after the DIFS/EIFS duration. If the channel becomes busy (triggered by signal arrival event) before the DIFS/EIFS duration expires, the timer will be cancelled by deleting the timeout event. Otherwise, when the timeout event is processed, the node knows that the channel is idle for a DIFS/EIFS period and randomly picks a BO value to continue backoff similarly. When the BO timer is suspended before it expires, the old timeout event is deleted. A new timeout event will be scheduled with remaining BO duration plus additional DIFS/EIFS when backoff is resumed. When a BO timeout event is processed, the BO counter finally reaches zero, and the node is allowed to transmit the packet. Backoff is also scheduled when a node successfully receives an ACK for its data packet.

Based on the above description, we can see that the incident, in which the DIFS/EIFS or BO timer reaches zero, will potentially trigger the node to transmit a packet. Thus, it is important to study the causality among arrival/end events of non-receivable signals and actual incident of packet transmission at a node. Figure 11 illustrates the causal relationships with an example that a node will backoff after it senses the channel to be busy (The example would be equivalent to no backoff if BO is 0). Though it is non-decidable when exactly backoff finishes, we can find a deterministic bound on the earliest time that the backoff will end by assuming that

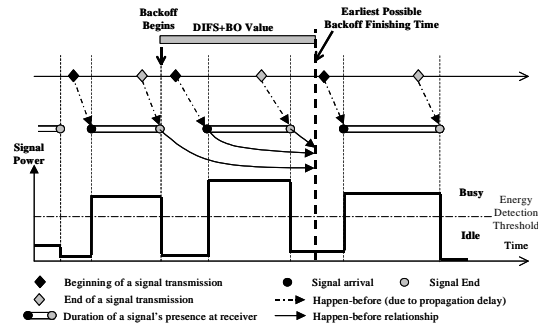


Figure 11: Causal orders of signal arrival/end events with reference to the deterministic lower bound of backoff finishing time.

the channel will always be idle from the moment the backoff procedure starts. An event can be scheduled for that time. When the event is processed, similar to the previous subsection, all the signals that arrived previously will be available in the signal history. Thus, retroactively, simulation can decide exactly how much the BO value should have been decreased. Since there is always an idle DIFS/EIFS period before BO value starts to decrement, it is easier for the simulation to restart the backoff process from the most recent busy to idle transition point with the readjusted BO value. It is as if we are inserting additional checkpoints between the time when backoff first starts and the time when the BO value finally reaches zero. As the causal effects of non-receivable signals on backoff process are still maintained by checkpointing at the next deterministic backoff timeout bound, the correctness of the simulation is unaffected. Note that during the backoff process, it is possible that a node receives packets from other nodes. The beginning and end of the packet reception are two special checkpoints for the above timer operation simulation as well, having priority over the ordinary checkpoint described above. Old signals will also be removed from the signal history at the checkpoints. Due to space constraints, a formal correctness proof of the above event reduction method will not be given here. However, it is important to know that simulation results are exactly the same with or without this optimization.

5.4. Performance evaluation

Experiments have been conducted to evaluate the impacts of the above optimization technique on the runtime performance of the parallel simulation. Note that the baseline simulator for comparison here has incorporated optimizations in both Section 3 and 4. In the first set of experiments, the node density is fixed to one node per 200x200 square meters, while the total number of nodes in the modeled wireless network is varied from 400 up to 3200. Nodes in the network use LAR as their routing protocol. 15% randomly chosen nodes have a CBR session of 20 packets per second and 512 bytes per packet to another randomly se-

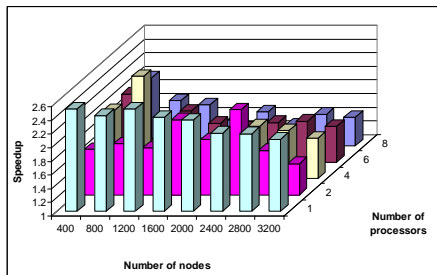


Figure 12:
Reducing
intra-LP
events: varied
network size.

lected receiver approximately within two hops of distance. This setting isolates scalability problem of LAR in large-scale network scenarios, which is not of interest to this paper. Figure 12 shows the relative speedup achieved with the optimization technique described earlier in this section. For networks with 400 to 3200 nodes, the achieved speed-up ranges from 1.3 to 2.6. In the second set of experiments, the number of nodes in the network is fixed at 3200 while node density is varied from one node per $140 \times 140m^2$ to one node per $220 \times 220m^2$. As the number of partitions (processors) increases, the additional speedup achieved via reducing event scheduling overhead drops slightly. It suggests that as the number of partitions increases, other overhead, such as synchronization among LPs becomes dominant in overall simulation execution time. As a result, effect of reducing event scheduling overhead becomes less significant.

Figure 13 shows the relative speedup achieved with the optimization technique described in this section. The achieved speedup decreases as the node density increases because more nodes fall in the reception range of a signal transmitted. Although the event scheduling overhead is reduced, the complexity of the bulk-processing is increased at both packet reception evaluation and timer checkpoints for high node density cases. In the ranges of the node density and the number of partitions used by the experiment, additional speedup from 1.09 up to 2.2 is achieved.

6. Conclusion

Explosive growth of wireless communication technology spurs a growing demand for detailed simulation tools that accurately predicate performance of wireless devices, protocols, and services. However, detailed simulation of wireless networks is computationally expensive. Speed-up achieved with parallel execution of wireless network simulation might be hindered by complexities of wireless networks. This paper has presented a number of optimization techniques to improve the parallel performance of detailed wireless network simulation. First, a propagation limit is introduced to reduce the cost of simulating signal propagation across multiple LPs. Experimental results show that the maximum speed-up of 20 times can be achieved with this optimization technique for the 10,000 node cases. Second, various sources of lookahead are identified to reduce

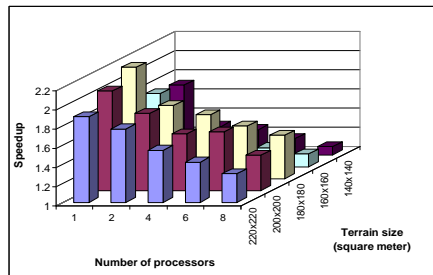


Figure 13:
Reducing
intra-LP
events: varied
node density
(3200 nodes).

synchronization overhead among LPs. With the above optimization, parallel simulation can achieve a speedup of up to 6 times with respect to sequential simulation for the 10,000 node cases. The last optimization technique eliminates events scheduled for non-receivable signals within each LP. An average speedup of 1.8 times can be achieved with this optimization technique for a network of 3200 nodes. Unlike many past studies, the optimization techniques take into account the details of the PHY and MAC layers for fidelity of wireless network simulation while exploiting the characteristics of such details to improve parallel performance of the simulation.

References

- [1] Qualnet. *www.qualnet.com*.
- [2] International Standard ISO/IEC 8802-11. *ANSI/IEEE Standard 802.11*, 1999.
- [3] R. Bagrodia, R. Meyer, and et al. Parsec: Parallel simulation environment for complex systems. *IEEE Computer*, 31(10):77 – 85, Oct 1998.
- [4] R. Fujimoto. *Parallel and Distributed Simulation Systems*. Wiley-Interscience, Dec, 1999.
- [5] Z. Ji, J. Zhou, M. Takai, and R. Bagrodia. Scalable simulation of large-scale wireless networks with bounded inaccuracies. In submission.
- [6] Y. B. Ko and N. H. Vaidya. Location-aided routing (LAR) in mobile ad hoc networks. In *Proceedings of ACM/MOBICOM'98*, pages 66–75, Oct 1998.
- [7] J. Liu and D. Nicol. Lookahead revisited in wireless network simulations. In *Proceedings of PADS'02*, May 2002.
- [8] J. Liu, L. Perrone, D. Nicol, M. Liljenstam, C. Elliott, and D. Pearson. Simulation modeling of large-scale ad-hoc sensor networks. In *European Simulation Interoperability Workshop 2001*, 2001.
- [9] G. F. Riley, R. M. Fujimoto, and M. H. Ammar. A generic framework for parallelization of network simulations. In *Proceedings of the Seventh International Symposium on Modelling, Analysis, and Simulation of Computer and Telecommunication Systems (MASCOTS)*, Oct 1999.
- [10] B. K. Szymanski, A. Saifee, and et al. Genesis: A system for large-scale parallel network simulation. In *Proceedings of PADS'02*, May 2002.
- [11] M. Takai, J. Martin, and R. Bagrodia. Effects of wireless physical layer modeling in mobile ad hoc networks. In *Proceedings of ACM/MOBIHOC'01*, pages 87–94, Oct 2001.
- [12] X. Zeng, R. Bagrodia, and M. Gerla. Glomosim: a library for parallel simulation of large-scale wireless networks. In *Proceedings of PADS'98*, May 1998.