

# Revisiting Effects of Detail in Ethernet Simulations \*

November 7, 2005

## Abstract

Virtually all simulation tools use some level of abstraction when creating models of various physical phenomena. In simulations of computer networks, the details of how a packet is encoded and transmitted on a copper or fiber communication medium are generally ignored. Rather, the simulation model of the packet transmission simply computes the arrival time of the packet at the other end of the medium, based on the speed of light delay on the wire, the data transmission rate of the interface, and the size of the packet being transmitted. The model may include a small probability that the packet is corrupted in-flight to capture the effects of noise on the medium. In this particular case, it is well known and accepted that the missing details of the physical encoding and data transmission methodology have little to no effect on the overall performance of the network being modeled. Similar abstractions are often used when modeling a broadcast-based CSMA/CD LAN, such as the ubiquitous Ethernet technology. The simplest abstraction of such a network ignores speed-of-light delays, and assumes that every station has instantaneous and perfect knowledge of the state of the medium. In this simple model, each station will transmit packets in the order that the interfaces received the transmit events, with no collisions or backoff. A slightly more detailed model accounts for the effect of transmission backoff when the medium is busy, and gives slightly more realistic results. This model again ignores speed-of-light delays to give a more computationally efficient model. Finally, the model could account for speed-of-light delays, which would result in a multiplicative increase in the number of simulation events needed for each packet transmission, but would result in a more

realistic model of the physical phenomena on the network. We study the effects of each of these abstractions, in terms of both simulation accuracy and simulation efficiency. We show that the simplest model has sufficient accuracy for most cases with realistic topologies and reasonable traffic intensity on the LAN. However, when the data demand on the LAN increases, the simpler models fail to capture the actual behavior of the network.

## 1 Introduction

Computer-based simulation has become the method of choice for virtually all research in the behavior of network protocols and design. For the most part, it is simply impractical to deploy new protocols or networking methods on a sufficient scale to demonstrate the features and correctness of the new protocols or methods. A number of good simulation tools exist, including ns-2 [11], SSFNet [2], JavaSim [21], IPTNE [19], and GTNetS [18], just to name a few.

All of these tools, by necessity, have abstractions for the behavior of the various network elements that compose the simulated network. For example, none of these tools have detailed models of the bit-by-bit data transmission of a packet on a point-to-point link. Such level of detail could be included, but with tremendous cost in terms of simulation execution time, and little benefit in terms of increased accuracy. In this particular case, it is generally accepted that the more abstract model, which simply schedules a single packet receipt event at the receiver at a suitable time in the future is sufficiently accurate and efficient for virtually all network simulation needs.

However, the choice of the level of detail in the models and the effects of those choices in other areas of network simulation are less clear. Indeed, the level of detail required for accurate models of wireless communications is still the subject of much

---

\*This work is supported in part by NSF under contract numbers ANI-9977544, ANI-0136969, ANI-0240477, ECS-0225417

research [22, 3, 8, 10]. Even the relatively well-understood Ethernet *Carrier-Sense Multiple-Access with Collision Detection* (CSMA/CD) [6] network medium can be modeled with several levels of abstraction, leading to various levels of accuracy. The effect of these differing abstraction levels both in terms of simulation accuracy and simulator efficiency is the focus of this work.

We have identified three different abstraction levels found one or more of the popular network simulation tools. We give an overview of the these models here, with more details later. The names given three models are ours, and not necessarily the same as those used by others.

1. The *completely abstract* model ignores speed-of-light delays, and assumes each station is immediately aware when any other station is transmitting. Any station wishing to transmit on a busy medium asks for a notification when the medium is free, and transmits immediately when notified. With this model, each packet transmission can be modeled with exactly two simulation events, one for the packet arrival, and one for the link-free event which causes the next packet to be transmitted.<sup>1</sup>
2. The *partially abstract* again ignores speed-of-light delay, but does account for the exponential backoff technique specified by IEEE 802.3[6]. In this model, a station wishing to transmit on a busy medium will schedule a *backoff* event for some random interval, and again attempt to transmit when the backoff event is processes. Here, packets can be transmitted with no less than two events (the same two above), and perhaps several more depending on how many times the station schedules backoff events.
3. The *fully detailed* model completely captures the effect of speed-of-light delay on the LAN, and the resulting possibility of *collision*. As above, stations schedule backoff events when attempting transmission on a busy medium, as

---

<sup>1</sup>An even more abstract model could transmit packets with only one event, using the “packet receipt” event as the “link free” action as well. Since by design stations on an Ethernet are reasonably close together, the two events are close together in time, differing only by the speed-of-light delay between two stations. We are not aware of any network simulation tool utilizing this abstraction.

well as when a collision is detected. Further, stations must wait for a specified time interval after sensing a non-busy medium and before starting a transmission. This model requires no less than  $2(N - 1)$  events, where  $N$  is the number of stations on the LAN.

Clearly, the three different abstraction models will have differing levels of accuracy and simulator performance. These differences are the focus of this work.

## 2 Related Work

There is a substantial body of work studying the various levels of detail in network simulation models [1, 4, 14, 16, 15]. The most relevant to our work is a recent study by Hussain [5]. In that work, the authors used the ubiquitous ns-2 [11] network simulator to perform a study similar to ours. However, their approach did not consider the fully detailed model, but rather compared the completely abstract model to the partially abstract model. Further, their study focused primarily on measuring web browser performance where the browsers and servers were on the same LAN. We use the more general on-off data source models, which gives a finer grained control of the load on the LAN.

Several researchers have discussed abstractions that ignore details of the behavior of the end-system protocols, using the so-called *fluid models* of networks. Riley and Jaafar [17] show that the reduced accuracy fluid model approach can be used to increase simulation performance with slightly reduced accuracy when modeling web browser performance in the presence of competing traffic. Kiddle [7] reports similar results using fluid abstractions in the IP-TNE simulator[19]. Melamed[12] discusses the *HDCF* simulation environment that uses *only* fluid abstractions to efficiently model buffer occupancy and loss rate in feed-forward networks. Liu et. al [9] give a detailed discussion of the efficiency and difficulties found when using the fluid modeling methods. Nicol[13] shows that one can abstract away details from not only a single protocol instance, but an entire set of similar protocol instances. With this type of abstraction, speedups of three orders of magnitude are possible, with only moderate reduction in accuracy.

In the field of wireless simulations, there is an even wider body of existing work analyzing the effects of details in physical layer modeling. Heidemann et. al [3] analyze five different levels of detail for physical layer modeling and show significantly different results in several cases, depending on which level of detail is used. Bagrodia et. al [22] show results from using an extremely detailed bit-level model of the OFDM encoding method for wireless physical layer modeling. That work showed a significant difference in the number of re-transmitted packets (almost a factor of 10) between the detailed OFDM model and a simpler path-loss model. However, the detailed model resulted in a reduced simulation efficiency of more than a factor of 1000. Kotz et. al [8] perform a detailed comparison of measured wireless performance in the field as compared to simulation-based analysis of the same network. Here, the simulation models contained three different levels of detail, and were shown to differ significantly in two of the three models, with only moderate agreement in the third. Liu et. al [10] extend the above work to observe the effects of physical layer details on various ad-hoc routing protocols. That work shows significant variation due to the path loss exponent, with less effects due to the log-normal shadowing parameters.

### 3 Detailed Ethernet Modeling

In this section we present the details of simulation models for each of the three levels of abstraction of Ethernet local area networks. We have implemented two of these in the “anonymous network simulator” which was used for all experiments discussed later.

#### 3.1 Completely Abstract Model

This model is the simplest of the three models, both in terms of simulator efficiency and ease of implementation. The LAN is represented by a single C++ object called `Ethernet`, that maintains the state of the broadcast medium. All stations connected to the LAN have an interface object called `EthernetInterface` that represents the network interface card (NIC) that is connected to the LAN. Each of the `EthernetInterface` objects has a pointer to the single `Ethernet` object. When an `EthernetInterface` object receives a data request (a packet to transmit) from the higher layer, it

first checks the busy flag in the shared `Ethernet` object. If the medium is not busy, the transmitting interface sets the busy flag in the `Ethernet` object, and schedules two future events. The first event indicates the packet is received at the intended recipient, and the second to indicate that the transmission is complete. If the medium is busy, the interface queues the packet in a queue for later transmission, and adds itself to a first-in-first-out queue in the `Ethernet` object known as the *Pending Transmit* queue. This queue is a list of all interfaces that have a packet ready to transmit on the medium. When an interface processes the `LinkFree` event, it will remove the head of the pending transmit queue (if any), and notify that interface it can now transmit on the medium. It is easy to see that this model uses exactly two events per packet transmission, and is relatively efficient. However, it does result in a strict FIFO ordering of packet transmissions on the medium, regardless of the station from which the packet is transmitted. This is clearly not an accurate representation of the actual behavior of an Ethernet LAN.

#### 3.2 Partially Abstract Model

The partially abstract Ethernet model takes into account the exponential backoff mechanism specified in IEEE 802.3. The `Ethernet` object representing the LAN and the `EthernetInterface` objects are identical to those described above. However, when attempting to transmit a packet, the interface will react differently to a busy medium. When the medium is busy, this model will select a random backoff timeout within a specified interval and schedules a *TransmitRetry* event after the backoff timeout expires. If the medium is still busy, the width of the backoff interval is doubled, and another backoff timeout is selected and scheduled. This process repeats for a maximum of 16 retries, after which the packet is discarded. This approach is a more faithful representation of actual Ethernet LAN behavior, and allows for packet re-ordering on the LAN as do actual networks. There is only a slight decrease in simulation efficiency due to one or more extra events (the *TransmitRetry* events) in the case where a busy medium is detected. However, this approach still does not capture the effects of Ethernet *collisions*, which occur in actual networks due to speed-of-light delays on the medium. Further, this approach ne-

<i>Parameter</i>	<i>Value</i>
Jam time	32 bit times
Slot size	512 bit times
Inter-frame gap	96 bit times
Initial backoff window	1 slot time
Maximum backoff	1024 slot times
Maximum retries	16
Max frame size	1518 bytes
Min frame size	64 bytes

Table 1: CSMA/CD Configuration parameters

<i>Event</i>	<i>Description</i>
FBE	First bit event
ClrE	Clear event
RtrE	Retransmit event

Table 2: Additional Events in “anonymous”

glects the carrier sensing delay specified by the IEEE 802.3 document.

### 3.3 Fully Detailed Model

This model is the most detailed of the three, and is an almost completely accurate representation of the behavior and performance of an actual LAN. Table 1 lists the parameters used in the model in accordance with the IEEE 802.3 document.

The CSMA/CD model was implemented in “anonymous”<sup>2</sup> using its event driven capabilities to accommodate the CSMA/CD behavior. Table 2 lists the newly introduced events into “anonymous” to accommodate the CSMA/CD behavior. The “FBE” (first bit event) is used by a host to inform other hosts on the wire that it is transmitting a packet. The time the FBE reaches each node depends on the distance from the sender, the FBE carries the size of the packet being transmitted so that each node would know how much time the Ethernet will be busy. The “ClrE” (clear event) is used by a host to inform the rest of the hosts on the Ethernet that it is aborting its transmission (due to collision). The “RtrE” (retransmit event) is used by a host to reschedule the transmit of its packet(s).

<sup>2</sup>The name of our simulator is omitted for reason of anonymity

The details of the CSMA/CD Model is represented by the finite state machine shown in Figure 1.

- **IDLE:** The interface starts in this state and remains in it until the application layer sends data to be transmitted on the wire.
- **DATA\_READY:** If the number of retries for the pending packet exceeds the retry limit the interface drops the packet and moves to the IDLE state, otherwise the interface senses the channel for a period equal to the inter-frame gap. If a FBE is received in that period it would move to the BUSY state otherwise it would move to the TRANSMITTING state.
- **BUSY:** The interface holds back its transmissions for a period equal to the transmission time of the packet size specified in the FBE. If another FBE is received in this state the interface moves to the EXTERNAL COLLISION state, otherwise it would move to the DATA\_READY state (by scheduling a RtrE).
- **EXTERNAL COLLISION:** The interface waits until a ClrE is received from all colliding nodes, afterwards it moves to the IDLE state.
- **TRANSMITTING:** The interface transmits the pending packet and sends FBE to every other node on the wire along-with the size of the packet being transmitted. If an FBE is received while transmitting the interface moves to the COLLISION state, otherwise it moves to the CHAN\_ACQ state.
- **COLLISION:** The interface sends ClrE to all other nodes on the wire, doubles the value of the maximum backoff and chooses a random time in the backoff window, after which it would increment the number of retries for that packet and move to the DATA\_READY state (by scheduling a RtrE).
- **CHAN\_ACQ:** The interface resets the value of its maximum backoff to its default value.

## 4 Experiments

In this section the efficiency and accuracy of the fully abstract and the detailed models are being studied

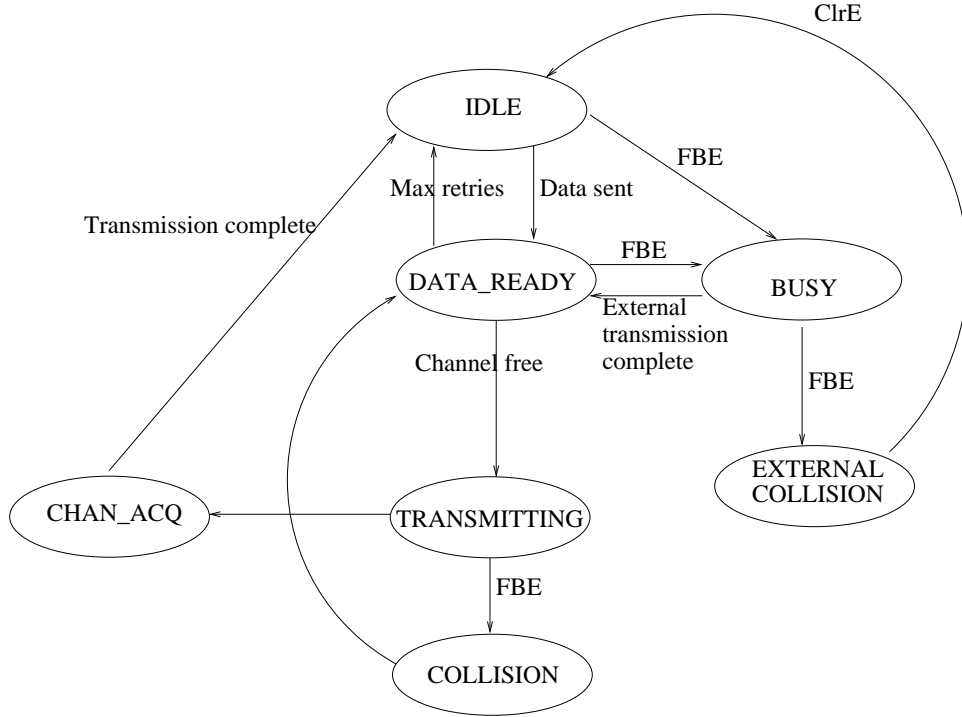


Figure 1: CSMA/CD implementation state transition diagram

through simulation experiments we did not include the partially abstract model in our study because it was already studied in [5]. The topology is an Ethernet with 31 hosts attached to it as shown in Figure 2, the distance between each node and the next one on the wire is one meter. There is one server node and the number of clients is varied as well as the traffic intensity. Each client runs an on-off application which is on for a random period of time and off for another random period of time. These random periods are chosen using exponential random variable generators with the same mean. During the on period the applications send data at a constant rate which is set according to the required traffic intensity. The constant rate is calculated using the following formula :

$$R = \beta * \rho / (N * d)$$

Where  $\beta$  is the bandwidth of the link (set to 10Mbps),  $\rho$  is the required traffic intensity or average load,  $N$  is the number of active clients, and  $d$  is the percentage of time the application is in the on state. Since the on and off random variables have the same mean, the duty cycle  $d$  is 0.5.

The amount of traffic that successfully reaches the server is added up to measure the overall throughput which is used to determine the model accuracy. We

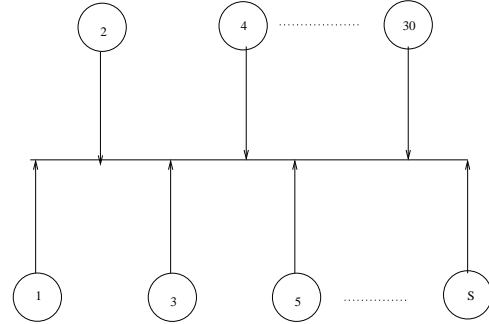


Figure 2: The Ethernet Simulation Topology

run the applications for 100 seconds and then let the simulator run for enough time to drain all the queues and the real execution time is recorded for the two models. Table 3 lists the parameters for the different experiments.

## 5 Results

In this section the results of the different experiments are discussed. Figure 3 shows the Ethernet performance in the case of 30 active clients sending data to the same server for varying traffic intensity for both the simple and detailed models. It is

<i>Parameter Description</i>	<i>Range</i>
Number of hosts	31
Number of servers	1
Number of clients	2-30
Bandwidth (Mbps)	10
Packet size (bytes)	512
Traffic intensity	0.1-1.0
Simulation time (seconds)	100

Table 3: Parameter definitions for simulation experiments

clear that for low traffic intensity the results of the simple and detailed models are very close and as the traffic intensity increases the difference widens. The reason is that for low loads the number of collisions/retransmissions is minimal and therefore the difference between the two models is negligible. The results of the detailed model are more accurate as they show that the maximum achievable CSMA/CD utilization is 60-70% as mentioned in [20]. Figure 4 shows the running time of the simulator for the previous experiments. The running time for the simple model grows linearly with increasing the average load because as the load increases the number of packets transmitted grows linearly and so the number of events needed to transmit those packets grows linearly. However in the detailed model we notice that the growth is exponential with increasing the load, this is due to the increased number of events needed to notify all neighbors of each packet being transmitted as well as transmission abortion. Assuming that we have  $N + 1$  nodes on the wire then for each packet that transmits the excess number of events needed to notify the neighbours is  $N$  (FBE for each neighbour), if that packet collides then  $N$  more events are needed to notify the rest of nodes that we are aborting transmission as well as one retransmit event needed to re-send that packet. Therefore for low average loads, when the amount of collisions is negligible the excess number of events would be linear and depending only on  $N$ . In case of high average load the number of collisions will grow and therefore the number of retransmissions would grow and for each retransmission we would need  $2N + 1$  more events on average to simulate the detailed model which counts for the exponential growth of the simulation running time.

Figure 5 shows the Ethernet performance for vary-

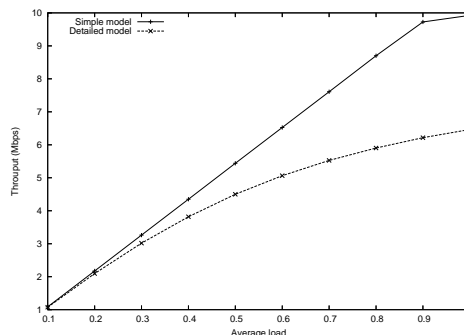


Figure 3: Effect of traffic intensity on the throughput for 30 active clients

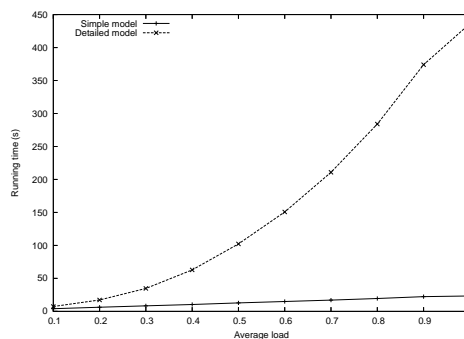


Figure 4: Effect of traffic intensity on the simulator running time for 30 active clients

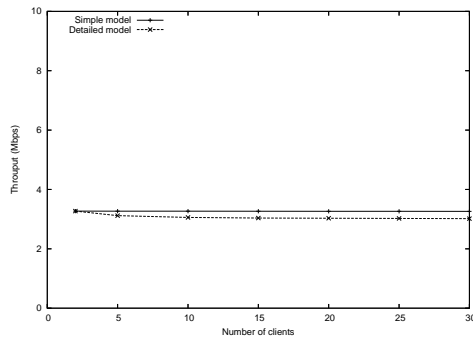


Figure 5: Effect of varying number of active clients for 30% average load on the throughput

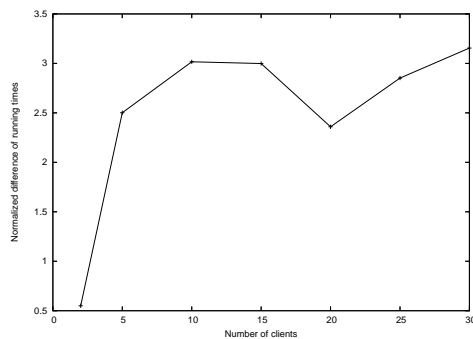


Figure 6: normalized Difference of Running Times for 30% Average Load between simple and detailed models

ing number of clients with 30% average load. We chose the 30% due to the fact that the difference in throughput is negligible as shown in Figure 3. It is clear from this figure that there is little difference in accuracy between the simple and detailed model, while Figure 6 shows the normalized difference in running times, which is calculated by subtracting the running time for the simple model from the running time of the detailed model and dividing by the running time of the simple model. The figure shows that for this low utilization and when the number of clients is low the difference between the simple and the detailed models is small (low number of collisions), however as the number of clients increase the difference tends to fluctuate around a constant number. The figure shows that the detailed model is about 3 times slower than the simple model, which agrees with Figures 3 and 4. As the average load increases, the overhead would also increase.

## 6 Conclusion

This paper revisited the study of effect of detail on ethernet simulations with the introduction of a more detailed simulation model that captures the fine details of the CSMA/CD protocol. We presented simulation experiments with the more general on-off data source models, which gives a finer grained control of the load on the LAN. Our results show that for moderate average load (30%) the simple model performance is acceptable and more efficient than the detailed model, which runs 3 times slower. While for high average load (over 70%) the simple model overestimates the performance of the LAN by about 50%. It should be noted that low average loads on the LANs is the normal case in real world with the abundance of high speed LANs which are rarely used to the full capacity.

## References

- [1] G. T. Almes and E. D. lazowska. The behavior of ethernet-like computer communication networks. In *7th ACM Symposium on Operating Systems Principles (SOSP)*, pages 66–81, 1979.
- [2] J. Cowie, A. Ogielski, and D. Nicol. The SSFNet network simulator. Software on-line: <http://www.ssfnet.org/homePage.html>, 2002. Renesys Corporation.
- [3] J. Heidemann, N. Bulusu, J. Elson, C. Intanagonwiwat, K. chan Lan, Y. Xu, W. Ye, D. Estrin, and R. Govindan. Effects of detail in wireless network simulation. In *Proceedings of SCS Communication Networks and Distributed Systems Modeling and Simulation Conference*, January 2001.
- [4] H. Hughes and L. Li. Simulation model of an ethernet. *Computer Performance*, 3(4):210–217, December 1982.
- [5] A. Hussain, A. Kapoor, and J. Heidemann. The effect of detail on ethernet simulation. In *18th Workshop on Parallel and Distributed Simulation*, May 2004.
- [6] IEEE. Ieee standard 802-3 carrier sense multiple access with collision detection(CSMA/CD)

- access method with physical layer specifications. *Institute of Electrical and Electronic Engineers*, 2000.
- [7] C. Kiddle, R. Simmonds, C. Williamson, and B. Unger. Hybrid packet/fluid flow network simulation. In *17th Workshop on Parallel and Distributed Simulation*, June 2003.
- [8] D. Kotz, C. Newport, R. S. Gray, J. Liu, Y. Yuan, and C. Elliott. Experimental evaluation of wireless simulation assumptions. In *MSWiM '04: Proceedings of the 7th ACM international symposium on Modeling, analysis and simulation of wireless and mobile systems*, pages 78–82. ACM Press, 2004.
- [9] B. Liu, D. R. Figueiredo, Y. Guo, J. Kurose, and D. Towsley. A study of networks simulation efficiency: Fluid simulation vs. packet-level simulation. In *Proceedings of the 20th Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM'01)*, April 2001.
- [10] J. Liu, Y. Yuan, D. M. Nicol, R. S. Gray, C. C. Newport, D. Kotz, and L. F. Perrone. empirical validation of wireless models in simulations of ad-hoc routing protocols. *Simulation: Transactions of the Society for Modeling and Simulation International*, 81(4):307–323, 2005.
- [11] S. McCanne and S. Floyd. The LBNL network simulator. Software on-line: <http://www.isi.edu/nsnam>, 1997. Lawrence Berkeley Laboratory.
- [12] B. Melamed, S. Pan, and Y. Wardi. Hybrid discrete-continuous fluid-flow simulation. In *Proc. of the SPIE International Symposium on Information Technologies and Communications (ITCOM 01)*, Aug 2001.
- [13] D. M. Nicol and G. Yan. Simulation of network traffic at coarse time-scales. In *19th Workshop on Parallel and Distributed Simulation*, June 2005.
- [14] P. J. P. O'Reilly and J. L. H. Jr. An efficient simulation technique for performance studies of csma/cd local networks. *IEEE Journal on Selected Areas in Communications*, SAC-2(1):238–249, January 1984.
- [15] K. Prasad and R. Patel. Simulation of ethernet performance based on single server and single queue model. In *12th Conference on Local Computer Networks*, pages 74–85, October 1987.
- [16] K. Prasad and R. Patel. Performance analysis of ETHERNET based on an event driven simulation algorithm. In *13th Conference on Local Computer Networks*, pages 253–267, 1988.
- [17] G. Riley, T. Jaafar, and R. Fujimoto. Integrated fluid and packet network simulations. In *Proceedings of Tenth International Symposium on Modeling, Analysis and Simulation of Computer and Telecommunication Systems (MASCOTS'02)*, Oct 2002.
- [18] G. F. Riley. The Georgia Tech Network Simulator. In *Proceedings of the ACM SIGCOMM workshop on Models, methods and tools for reproducible network research*, pages 5–12. ACM Press, 2003.
- [19] R. Simmonds, R. Bradford, and B. Unger. Applying parallel discrete event simulation to network emulation. In *Proceedings of the 14th Workshop on Parallel and Distributed Simulation*, May 2000.
- [20] H. Takagi and L. Kleinrock. Throughput analysis for persistent CSMA systems. *IEEE Transactions on Computers*, COM-33(7):627–638, July 1985.
- [21] J.-Y. Tyan and C.-J. Hou. Javasil: A component-based compositional network simulation environment. In *Proceedings of the Western Simulation Multiconference, Communication Networks And Distributed Systems Modeling And Simulation*, Jan 2001.
- [22] G. Yeung, M. Takai, R. Bagrodia, A. Mehrnia, and B. Daneshrad. Detailed ofdm modeling in network simulation of mobile ad hoc networks. In *18th Workshop on Parallel and Distributed Simulation*, May 2004.