Validation Experiences with the NS Simulator

Sally Floyd

April 19, 1999

Abstract

This short note is written as a submission to the DARPA Workshop on Validation of Large Scale Network Simulation Models. The purpose of the workshop is to identify issues critical to the validation of network models, and also to develop potential strategies for the research community to address these issues.

1 Current network simulation models or tools:

I have used simulations as a key component of my network research since I began working in this area about ten years ago. While some of the simulation work (e.g., on synchronized routing messages, and on Scalable Reliable Multicast) has been done on special-purpose simulators that I wrote to investigate particular issues, most of my simulations have been done on the NS Network Simulator and its predecessor simulators (NS-1, and before that 'tcpsim', and before that an earlier version) developed by Steve McCanne and myself at LBL. The NS simulator [NS95] evolved from a network simulator developed by Steve McCanne in 1990, based on the REAL network simulator. Steve and I were the only users of this simulator for some years. I used the simulator for research on issues such as TCP dynamics, RED queue management, explicit congestion notification, and CBQ scheduling, and was responsible for the details of those modules in the simulator. The LBL Network Simulator (written in C) evolved into the NS-1 network simulator (written in C++ and Tcl, and made publically available in 1995), and then into NS-2 (whose non-beta release was in 1997). For the last three years, the development of NS has been joint work at ISI/USC, Xerox PARC, LBNL, Xerox PARC, and UC Berkeley through the VINT project, and NS has become widely used in the network research community. In addition to using NS in my own research, I also have an interest, as a member of the VINT project, with the development of NS for more general use in the network research community.

2 Current validation techniques:

Raj Jain [Jai91] defines validation as "ensuring that the assumptions used in developing the model are reasonable in that, if correctly implemented, the model would produce results close to that observed in real systems", and defines verification as related to the correctness of the implementation. In this note I am discussing both validation and verification.

There has been an evolving set of validation tests for NS since the first generation of the LBL Network Simulator in 1990. I was interested in the underlying dynamics inherent in the TCP algorithms for end-to-end congestion control. I created a set of validation tests for my own use, to verify that the congestion control mechanisms in the simulator matched the intended model (not to verify that the congestion control mechanisms matched some particular real-world implementation of that model). In retrospect, this was the right decision for my purposes. For example, the real-world implementations of the window increase mechanism in Reno TCP included a significant bug, of an added multiplicative factor for the window increase during congestion avoidance; the intention of my research was not to explore the behavior of these existing TCP implementations, but to explore the dynamics inherent to the underlying congestion control mechanisms.

Some of my research has involved the development and analysis of protocols and mechanisms for which there are no real-world counterparts. Examples of this include RED active queue management, Explicit Congestion Notification, CBQ scheduling, and the Selective Acknowledgement option for TCP. Thus, for these modules, the validation tests served to verify that the modules in the simulator exhibited the behavior expected from the underlying models.

I have not used the NS simulator for large-scale simulations; the large-scale simulations that I have done, of Scalable Reliable Multicast, were done on a separate simulator that I wrote for that purpose, with its own set of validation tests. Thus, my own experience with validation in NS has not really been related to the particular concerns of large-scale simulations.

The validation tests that I have written for NS are not exhaustive. For example, I am sure that some of the code in the modules is not exercised or verified by any validation test, and certainly many combinations of events are not verified in any validation test.[1] Initially, each time that I added a new parameter or behavior, I would also add a validation test to verify that that parameter functioned as I intended; however, many of these validation tests never survived the many transformations of the underlying simulator (from the original LBL network simulator, to tcpsim, to NS-1, to NS-2...).

The main validation and verification of the modules in the simulator comes not from the validation tests themselves, but from the continued exploration of the protocols and mechanisms under many different simulation scenarios, and the detailed examination of the traces to understand the behavior of the simulations. When the behavior of simulations conforms to the researcher's expectations, that does not necessarily mean that the simulator is correct. But at the same time, the careful exploration of unexpected results can go a long way towards uncovering and distinguishing between bugs in the implementation, inadequacies of the underlying model, and unexpected behaviors inherent in the underlying model.

I wrote a short note discussing some of the TCP validation tests in 1994 (before the simulator itself was made available) [Flo95], in response to a question on a mailing list about how other researchers verified their implementations of TCP in their own simulators. The note describes basic tests verifying that the Tahoe TCP Slow-Start, Congestion Avoidance, and Fast Retransmit mechanisms, the Reno TCP Fast Recovery mechanisms, and the SACK TCP mechanisms work in the simulator as intended. Other papers describe the validation tests for the scheduling mechanism CBQ [Flo97b], the active queue management mechanism RED [Flo97c], the Explicit Congestion Notification mechanisms [Flo98], two-way TCP [FFH97], and the TCP congestion control mechanisms [FF96]. I have also written several validation tests for NS (e.g., for larger TCP initial windows, and for TCP retransmit timers) that have no accompanying document other that a short text file distributed with the simulator.

3 A self-assessment of the effectiveness of the validation techniques:

There are two key issues in defining the model to be used for Internet research. One is that the Internet is inherently difficult to characterize, due to the heterogeneity and rapid change of Internet topology, link characteristics, protocols, router mechanisms, applications, and traffic mixes [PF97, Flo97a].

A second key issue is that, as network researchers proposing or analyzing protocols or mechanisms for the future Internet, the reason to conduct simulations is often to understand the possible relevance of these proposals to the future evolution of the Internet. That is, we might be concerned not with the behavior of these protocols in the current Internet, but with their behavior in the Internet as it might be some years in the future.

This is one reason to sometimes use models that do not include those details of the current Internet that are likely to heavily skew simulation results and are not inherent to the proper functioning of the network. A related problem with an overly-specified model is that fine-grained optimization of some performance parameter in the model might not necessarily transfer to fine-grained optimization in the more heterogeneous and changeable evolving Internet. The appropriateness of a particular model depends heavily on the ultimate purposes of the research that uses that model.

One illustration of the inappropriate use of models is the wealth of papers proposing a major change to the Internet infrastructure to deal with the problems caused by Reno TCP's lack of resilience to multiple packets dropped from a window of data. There is a first-order fix to this problem of Reno TCP, and that is to make a small change to the congestion control algorithms (e.g., NewReno or SACK TCP). These new TCP implementations are in the process of being deployed in the global Internet. Another change currently in progress that minimizes Reno TCP's problem with multiple packet drops is the deployment of active queue management (e.g., RED).

I have reasonable confidence in the effectiveness of the validation techniques for my own research. My own research has not been to use NS to give numerical comparisons between competing mechanisms, but to explore and illustrate mechanisms proposed for the future Internet. Behavior illustrated and explored with simulations, but also explained with straightforward analysis, include TCP phase effects (the sensitive dependence of TCP throughput on small changes in simulation parameters) [FJ92]; the relationship between TCP throughput and the roundtrip time or the number of congested gateways; and the degradation of performance of TCP over ATM in heavy congestion in the absence of modified cell-discard policies. For research projects such as these, while I do not claim that the simulations are without bugs, I have confidence that my research results are sound, and are not simply based on broken simulations.

I have less confidence in the effectiveness of the existing validation techniques for an arbitrary user of NS. While there is an extensive set of validation tests, these validation tests are not complete, and different users might have need of different underlying models. The following warning is at the top of the web page for NS:

We emphasize that ns is not a polished and finished product, but is the result of an on-going effort of research and development. In particular, significant bugs in the software are still being discovered and corrected. Users of ns are responsible for verifying for themselves that their simulations are not invalidated by bugs. We are working to help the user with this by significantly expanding and automating the validation tests and demos.

[1] I have contributed to validation tests that have been considerably more complete, in particular, validation tests in the mid-70's of software used as part of the underlying safety system for the Bay Area Rapid Transit (BART) subway system.

Similarly, users are responsible for verifying for themselves that their simulations are not invalidated because the model implemented in the simulator is not the model that they were expecting. The ongoing ns Notes and Documentation should help in this process.

This warning is intended quite seriously, and not just as a formal abnegation of responsibility on the part of the NS developers. The validation tests are intended not only as a tool for use by the developers to minimize the introduction of undetected errors, but also as a tool for users to understand for themselves the model implemented in the simulator, and to identify which aspects of the model are covered by existing validation tests. To help the user in this task, the model and implementation are described in the NS documentation [FV99]. Unfortunately, the underlying model is not formally or completely specified, and in fact can change over time.

In many cases, there are a range of options in terms of the underlying model: the one-way TCP implementations, for example, reflect a simplified underlying model of one-way data transfer of fixed-size packets, with no special functionality for SYN or FIN packets. A richer two-way TCP implementation allows two-way data transfer, a range of data packet sizes within a single connection, and connection set-up and tear-down with SYN and FIN packets. It is critical that users understand the model underlying their own simulations.

4 Key difficulties when attempting to validate the model:

A key difficulty has been determining which behaviors need to be added to the validation tests. Some reported problems in NS were never checked in any validation tests. An example of this is a reported problem in the TCP ECN behavior with delayed-ACK receivers; this particular problem was reported by a user of NS, and was due to the incompleteness of the underlying model.

Other reported probems have been for behaviors that once worked correctly in the simulator, but where a bug was introduced in one of the transformations of the simulator (from the early LBL simulator, to tcpsim, to NS-1, and finally to NS-2). The two transformations from the early LBL simulator to tcpsim, and from tcpsim to NS-1, both required that the validation tests be completely rewritted in a new format, and many validation tests did not survive all of these transformations. The transformation from NS-1 to NS-2 included as a condition that NS-2 be able to run the validation scripts from NS-1 in backward compatibility mode, without having to rewrite all of the validation scripts in the new format.

An example of a bug introduced in the transformation from NS-1 to NS-2, and for which the original validation test was not retained, is a bug with TCP retransmit timers. In the original beta version of NS-2, the TCP retransmit timers backed-off correctly in response to successive losses, but once backed-off, they never returned to their original default value. This bug was reported by an early user of NS-2.

Identifying key behaviors to add to the validation tests to illustrate and verify the underlying model is particularly critical when the model is itself being clarified and redefined by the researcher in the process of the validation.

References

[FF96] K. Fall and S. Floyd. Simulation-based comparisons of tahoe, reno, and sack tcp. *ACM Computer Communication Review*, Jul. 1996. URL http://www-nrg.ee.lbl.gov/nrg-papers.html.

[FFH97] K. Fall, S. Floyd, and T. Henderson. Ns simulator tests for reno fulltcp. URL http://www.aciri.org/floyd/papers/fulltcpsims.ps, Jul. 1997.

[FJ92] S. Floyd and V. Jacobson. On traffic phase effects in packet-switched gateways. *Internetworking: Research and Experience*, 3(3):115–156, Sep. 1992.

[Flo95] S. Floyd. Simulator tests. URL http://www-nrg.ee.lbl.gov/nrg-papers.html, Jul. 1995.

[Flo97a] S. Floyd. Internet simulations: Issues in defining the model. URL http://www.aciri.org/floyd/talks/sf-dimacs-97.ps.Z, Oct. 1997.

[Flo97b] S. Floyd. Ns simulator tests for class-based queueing. *Unpublished draft*, Apr. 1997. URL ftp://ftp.ee.lbl.gov/papers/cbqsims.ps.Z.

[Flo97c] S. Floyd. Ns simulator tests for random early detection (red) queue management. URL http://www.aciri.org/floyd/papers/redsims.ps.Z, Apr. 1997.

[Flo98] S. Floyd. Ecn implementations in the ns simulator. URL http://www.aciri.org/floyd/papers/ecnsims.ps, Dec. 1998.

[FV99] K. Fall and K. Varadhan. Ns notes and documentation. Technical report, 1999. URL http://www-mash.cs.berkeley.edu/ns/ns-documentation.html.

[Jai91] R. Jain. *The Art of Computer Systems Performance Analysis*. John Wiley and Sons, Inc., 1991.

[NS95] Ns (network simulator), 1995. URL http://www-mash.cs.berkeley.edu/ns/.

[PF97] V. Paxson and S. Floyd. Why we don't know how to simulate the internet. *Proceedings of the 1997 Winter Simulation Conference*, Dec. 1997. URL http://www.aciri.org/floyd/papers/wsc97.ps.