ECE/CS 541: COMPUTER SYSTEM ANALYSIS

PROJECT #2

November 7, 2014

Due: Paper due Thursday December 4, at 5 p.m. Presentation must be made on Saturday December 6, at ECE/CS 541 Workshop. Precise starting time and place will be announced.

This project is intended to be open-ended, and allow you to explore an area of the course material that you want to know more about that we did not have enough time to cover in class or homeworks. They may be done individually, or in groups, with optimal group size one or two.

The work for the project should be original, i.e. if you are doing a project for me or another professor in a similar area, you should not pick this as a subject to work on, but instead choose another topic. Related projects are OK, but you should make it clear to me how this project is different than other project you are doing. The following are only suggested topics; if you prefer to work on a different topic, please contact me soon and we can discuss it.

1. Pick a system that you would like to model (as a stochastic activity network or using another formalism), and build a SIMULATION model of it. This system can come from another paper that analyzed the system, or from work you are doing. A good place to find such systems is in IEEE journal articles, or the QEST or Petri Nets and Performance Models Conference proceedings. Then, do one of the following:
   - Reproduce some or all of the results (acceptable).
   - Modify the assumptions of the paper and solve the new model (better).
   - Improve on the results of the paper (best).

   You can find example of system that are already modeled as SANs in papers at my group's web site: http://www.crhc.uiuc.edu/PERFORM. Don't pick these systems to model, but you can use them as example of how to model things as SANs. Many of these papers started out as class projects.

2. Implement the standard uniformization, adaptive, and combined adaptive uniformization methods. The references for these methods are on my web page http://www.crhc.uiuc.edu/PERFORM. Compare the methods in terms of their computation and memory cost. Devise a good test (Markov) model to compare the methods.

3. Read and implement G. Horton and S. Leuteneger, "A multi-level solution method for steady-state Markov chains," Proc. 1994 ACM SIGMETRICS Conf. On Measurement and Modeling of Computer Systems, pp. 191-200, May 1994. Implement a simple Gauss-Siedel or SOR solver for comparison. Compare the new method in terms of space and time required for solution.

4. Implement the mean-value analysis method by Reiser and Lavenberg for product form queueing networks. In doing so, come up with a simple textual method for specifying the queueing formalism. Your solver should be able to handle as general queuing networks as possible (e.g. multiple classes of customers). Test your solver with several queuing networks you develop to show the space and time complexity of your solution.

5. Implement a Fault Tree or Reliability Block Diagram solver. The solver should support textual input of either reliability block diagrams or fault trees, of arbitrary complexity. It should be implemented using space and time efficient data structures and able to handle very large models. Look for recent results on solving extended fault trees and implement some of these new methods if possible.

6. Implement, compare, and analyze several different modern uniform[0,1] random number generators. There are now random number generators that have extremely long period, much longer than the range of an unsigned long. In addition to implementing the multiple random number generators themselves, you will need to implement the statistical tests that are used to compare them. A significant portion of the project will be the statistical analysis. References for these generators can be found in ACM/IEEE Transactions on Modeling and Simulation.

7. Implement a Matrix-geometric solver or Spectral expansion solver. These solvers are useful solving infinite state queues, which have more complicated structures than we have studied in class, and have been used in many practical settings.

For the projects that involve building a Markov process solver, you create your own textual format for entering the chains, or you can use as input the output format for Markov processes in Mobius. I can provide you details of that format.

For all projects, you should prepare a professionally written paper detailing what you have done, and presenting the results. You should also provide me with a tar file of your developed code, along with instructions so that I can run your test cases. Your code should be written in a manner that it is both space and time efficient, e.g. use sparse data structures for matrices. The paper should be limited to 12 pages, 2-column IEEE conference format (like the first project.)

Feel free to extend the scope of the project beyond what I have asked for, incorporating original creative thought. This project is a significant part of your grade for the course, and should represent your (high) level of understanding of its material.

Please come to office hours and tell me what you plan to do for the project. All groups should see me by November 14 (The sooner the better!) At this meeting, I would like you to bring one page sheet outlining the project you would like to do, and the initial reading and work you have done on the project. If you want to propose a project not on this list, please email me as soon as possible, so that we can agree on a topic before soon. If you are doing the first project, bring a copy of the paper you will base your model on to the meeting. For all projects, describe more specifically what you intend to do, using my short description as a starting point.