Northwestern University
Electrical Engineering and Computer Science
EECS 356: Intro. to Formal Spec. and Verification
Prof. Hai Zhou

Feb. 5, 2013
Handout #4

Project Guidelines and Suggestions

1 Guidelines

The purpose of the project is to help you extend what you learned from the course, practice problem solving methods, and get an early start on research. Try to be as creative as possible. The evaluation of the project is not based on how few errors you make but how innovative your approach is.

2 Some suggestions on research projects

Here are some suggested project ideas. You are very encouraged to come up with your own ideas, however, you must be able to justify what you propose. I will be happy to discuss them with you or discuss any ideas you have.

1. **Specification and Verification of Security** Security is the robustness of a system under attacks. In this project, we are looking at different systems and applications and study how to formal specify them and their environment so that security can be formally understood and verified.

   Specifically, we want to do a couple of case studies to see how TLA+ and TLC are effective in protocol specification and verification. One starting point is Chang and Shmatikov's "Formal Analysis of Authentication in Bluetooth Device Pairing" (`http://www.cs.utexas.edu/~shmat/shmat_fcs07.pdf`). They used a process calculus to specify the protocol and a resolution-based verifier called ProVerif to verify it. Your job here is to use TLA+ and TLC and to compare the results with them.

   You can do the same on other protocols too.

2. **Probabilistic System Specification and Analysis** Many systems exhibit random or probabilistic behaviours. Many system properties, such as anonymity, are enforced via random behaviours. TLA+ and TLC cannot be used in these tasks. However, there are languages and tools developed for them. PRISM (`http://www.prismmodelchecker.org`) is one of such tools. In this project, you are suggested to first study PRISM, and then extend TLA+ to specify probabilistic systems. You also need to have a tool to verify the specifications. There are two options: you may modify TLC to handle probability or you may translate your modified TLA+ specs into PRISM.

3. **Specification and Verification of Dijkstra's Self-Stabilization Algorithm** Dijkstra designed a self-stabilization algorithm for a distributed system. It has the property that no matter what state the system is at, it will eventually converge to a legal state where exact one of the machines is privileged. The original paper can be found at `http://www.cs.utexas.edu/users/EWD/ewd03xx/EWD391.PDF`. You need to specify the algorithm and its correctness conditions by TLA, and then prove its correctness. You are also encouraged to modify the algorithm while still maintain its correctness.

4. **Weak binary semaphore** is a binary semaphore with weak fairness. Thus, if you have two processes using a single binary semaphore to protect a critical region, it is possible for one process to never enter the critical region: it could infinitely often find itself in contention with the other process, and the other process could always win the contention. An example of a weak binary semaphore is one implemented with "test and set".

   Implement n process mutual exclusion using only weak binary semaphores. The number of processes n is a parameter of the system. Show your solution is correct using TLA+.

5. **Barrier synchronization** is a concurrency construct that allows a (fixed) set of threads to block until all threads have arrived at the barrier. If *barrier(n)* is the construct, then a thread blocks when it executes *barrier(n)* until all n threads have executed *barrier(n)*. Then, each blocked thread unblocks, but blocks again if it again executes *barrier(n)*.

   First, come up with a lower bound on the number of semaphores that are required to implement barrier synchronization. Your argument can be informal. Then, devise a solution that uses as few semaphores as you can - ideally, the number you showed to be a lower bound. Use TLA+ to show that your solution does indeed implement barrier synchronization.

6. **Consensus** Consider four processes that communicate by sending messages to each other. You can assume that if a process p sends a message to process q at time t, then q will receive the message by time $t + D$ for some known constant D. Each process i has a value $i.input$. Your goal is to design a protocol such that each correct process i determines a value $i.output$ where

   - For any two nonfaulty processes i and j, $i.output = j.output$.
   - If, for all processes i, $i.input = v$, then each nonfaulty process j sets $j.output$ to v.

   No more than one process can be faulty. A faulty process can send any message it wishes. For example, it can tell one correct process that its input value was 1 and tell another process that its input value was 17. It can also fail to send some messages, or simply stop running. Show that your protocol is correct using TLA+.

3 Submissions and deadlines

Feb 7: Reach a decision on which project you want to do. Submit the title, a brief description of the ideas and approaches.

Feb 21: A midterm progress report. Have you achieved substantial progress?

Mar 14: A complete report with all theories, results, and conclusions.