Büchi Automata

- Understand how BA intersection of Labeled Transition Systems (LTS) works. Use the constructions illustrated in Figure 9.5 and 9.6 of the CGP book (Clarke, Grumberg, and Peled).

Your task: What is the Omega-regular language of the automata of Figure 9.6?

- Suppose Figure 9.5 (LHS) has only final states; then perform the intersection operation according to the equation on top of Page 127 of CGP book.

Your task: Draw the resulting Omega-regular automaton.

- Treat the Kripke structure in Figure 23.1 of my book as a “system automaton.” Draw an Omega-regular automaton for expressing the temporal property $\square\lozenge(c_1 \lor c_2)$. Then negate the property $\square\lozenge(c_1 \lor c_2)$ using the duality between $\square$ and $\lozenge$. Draw an Omega-regular automaton for the negated property.

Then perform the intersection algorithm between the system automaton and the negated property automaton. For simplicity, consider only the path $N_1, N_2, T_1, N_2, C_1, N_2$, and back to $N_1, N_2$.

See whether the language is empty and explain. The intersection algorithm between two Kripke structures follows a similar logic as that followed between two LTSs, except: (i) we start the system automaton (SA) and property automaton (PA) at their initial states, (ii) we march the PA and SA in alternation, (iii) we watch for the final state in $Q_1 \times F_2$ to recur (Büchi acceptance).

Your task: The above will be solved by me; you may take notes.

- Take the property $\square\lozenge c_1$, negate it, and perform the intersection. Explain whether you get an empty language or not. Hence, explain whether the original property is violated or not. For simplicity, consider only the path $N_1, N_2, N_1, T_2, N_1, C_2$, and back to $N_1, N_2$. 
• Practice nested DFS on the above example violation.
• Your task: Do the above. (Study my book’s material around Figure 23.11 and 23.12.)

• Will go through the many Promela examples, finally!

**Floyd/Hoare logic, Weakest Precondition, Strongest post-condition, and Path Forcing**

• Discuss Strongest Post-condition briefly.
• Will take any questions you might have from Asg 2 and Asg 3, and work them out. I will also introduce you to Yices.
• Do a WP inference through Binary search to print $L^*$
• Your task: Do the above

• Confirm that the loop invariant works along the other path of Tournament min (only consider traversals within the loop).
  Your task: Do the above

**Take-home questions**

(50% of the points, due Thursday 2/24 midnight).

• (25%) Create a Dining Philosophers model that avoids deadlocks and livelocks (ensures the *No Lockout* property of Herlihy/Shavit—*individual progress* as I put it). Verify using SPIN. Study how the number of states grow with the number of philosophers for 2 through 5 philosophers. See if partial order reduction obtains any benefits.

• (25%) Encode the N-processor Mutual Exclusion protocol of Peterson as a Promela model.
  – Check for mutual exclusion.
  – State and check two interesting liveness properties. It is highly recommended that you use the LTL translator to generate the *never* automata.