Computer Science 81, Spring 2013
Assignment 10
Due Monday April 29
Reading: Sipser, chapters 3 and first part of 4

1. **[20 points] Using JFLAP, program a Turing machine that accepts the following language:** $\{ww^R \mid w \in \{0, 1\}^* \wedge \#_0(w) = \#_1(w)\}$. Include a screenshot of the state diagram and one of the machine test on the following 24 inputs (using the run-multiple feature).

ε	0101	100110011001
0	1100	010101101010
1	110011	0100110110110010
00	11000011	0110110000110110
11	00111100	110100110100001011001011
010	01011010	001011001011110100110100
0110	10100101	000001111100111110011111100000
1001	11000011	1010101010101010010101010101010101

Below, all proofs are informal and intended to develop intuition and expository style, so please provide convincing write-ups of each.

2. **[10 points] Sipser book, Problem 3.15 *c (star) only*:**

3.15 Show that the collection of decidable languages is closed under the operation of

- **A a.** union.
- **b.** concatenation.
- **c.** star.
- **d.** complementation.
- **e.** intersection.

---

3. **[10 points] Sipser book, Problem 3.16 *b (concatenation) only*:**

3.16 Show that the collection of Turing-recognizable languages is closed under the operation of

- **A a.** union.
- **b.** concatenation.
- **c.** star.
- **d.** intersection.

---

4. **[20 points] Sipser book, Problem 3.18:** Show that a language is decidable iff some enumerator enumerates the language in standard string order.

5. [10 points] Sipser book, Exercise 4.2:

4.2 Consider the problem of determining whether a DFA and a regular expression are equivalent. Express this problem as a language and show that it is decidable.

---

6. [10 points] Sipser book, Problem 4.11:

Let $INFINITE_{\mathrm{PDA}} = \{\langle M \rangle \mid M \text{ is a PDA and } L(M) \text{ is an infinite language}\}$. Show that $INFINITE_{\mathrm{PDA}}$ is decidable.

---

7. [20 points] Construct a proof of Church's Theorem: There is no algorithm for determining whether a first-order predicate calculus formula is valid.

Do this by showing how to translate a Turing machine with initial and halting state into a set of clauses such that the set of clauses is unsatisfiable iff the Turing machine halts on the given initial state. (Create a clause for each TM rule and a few extra clauses for situations where the end of the tape is reached.) We show in class that the latter problem is not decidable.

Extra credit [up to 30 points]: Submit a demonstration using Prover9 or Prolog.