Staff Solutions to Problem Set 3

Reading:
- Chapter 6. State Machines
- Chapter 7. Recursive Data

Problem 1.
“Token replacing” is a single player game using a set of tokens, each colored black or white. Except for color, the tokens are indistinguishable. In each move, a player can replace one black token with two white tokens, or replace one white token with two black tokens.

We can model this game as a state machine whose states are pairs \((n_b, n_w)\) where \(n_b \geq 0\) equals the number of black tokens, and \(n_w \geq 0\) equals the number of white tokens.

Let us assume that the game starts with one black token, that is, state \((1, 0)\).

(a) Define the predicate \(T(n_b, n_w)\) by the rule:

\[ T(n_b, n_w) ::= \text{remainder}(n_w - n_b, 3) = 2. \]

Prove that \(T\) is true for all reachable states.

COMMENTS:
- FP_token_state_machine_conflict
- minor variant of FP_token_state_machine
- F15.mid2conflict
- author: Zoran Dzunic, edited ARM 10/15/15

keywords = [ state_machines derived_variable induction congruences ]

Solution. \(T\) is true for the start state since \(n_w - n_b = 0 - 1 = -1\) and \(\text{remainder}(-1, 3) = 2.\)

Next we show that \(T\) is a preserved invariant.

So suppose \(T(n_b, n_w)\) holds. The only two possible transitions are into a state \((n'_b, n'_w)\) equal to \((n_b - 1, n_w + 2)\) or \((n_b + 2, n_w - 1)\). But since

\[ n'_w - n'_b = n_w - n_b \pm 3 \]

has the same remainder on division by three as \(n_w - n_b\), we conclude that \(T(n'_b, n'_w)\), which proves that \(T\) is invariant.

Since \(T\) is a preserved invariant that is true for the start state, it is true for all reachable states by the Invariant Principle.
(b) Does the same hold for the predicate
\[
\text{rem}(n_b - n_w, 3) = 2?
\]

Explain.

Solution. No. This predicate is not true for the start state.

(e) We will now prove the following:

Claim. If \( T(n_b, n_w) \), then state \((n_b, n_w)\) is reachable.

Note that this claim is entirely different from the result in part (a) that \( T \) is a preserved invariant. A preserved invariant satisfied by the start state tells you what can’t be reached, namely, any state that does not satisfy the invariant. It does not tell you what can be reached.

The proof of the Claim will be by induction in \( n \) using induction hypothesis

\[
\forall(n_b, n_w). [(n_b + n_w = n) \text{ AND } T(n_b, n_w)] \text{ IMPLIES } (n_b, n_w) \text{ is reachable.}
\]

The base cases will be when \( n \leq 2 \).

- Assuming that the base cases have been verified, complete the Inductive Step.

Solution. Proof. Assume that the induction hypothesis holds for some \( n \geq 2 \). Suppose \( n_b + n_w = n + 1 \) and \( T(n_b, n_w) \) holds. We want to show that \((n_b, n_w)\) is reachable.

Since \( n + 1 \geq 3 \), either \( n_b \geq 2 \) or \( n_w \geq 2 \).

In the case that \( n_b \geq 2 \), we have \( n_b - 2 \geq 0 \), so \((n_b - 2, n_w + 1)\) is a state. Also, \( T(n_b - 2, n_w + 1) \) holds because
\[
(n_b - 2) - (n_w + 1) = n_b - n_w - 3,
\]

which has the same remainder on division by three as \( n_b - n_w \).

Since
\[
(n_b - 2) + (n_w + 1) = n_b + n_w - 1 = n,
\]

we conclude by induction hypothesis \( P(n) \) that \((n_b - 2, n_w + 1)\) is reachable. But \((n_b - 2, n_w + 1)\) transitions in one step to \((n_b, n_w)\), which proves that \((n_b, n_w)\) is reachable.

The same argument applies in the case that \( n_w \geq 2 \).

We conclude that in any case \((n_b, n_w)\) is reachable, which completes the induction step.

- Now verify the Base Cases: \( P(n) \) for \( n \leq 2 \).

Solution. There are only six states with \( n \leq 2 \):

\[
(0, 0), (1, 0), (0, 1), (1, 1), (0, 2), (2, 0).
\]

Of these, only \( T(1, 0) \) and \( T(0, 2) \) hold. The state \((1, 0)\) is reachable since it is the start, and \((0, 2)\) is reachable in one step from the start. So all the states with \( n \leq 2 \) and satisfying property \( T \) are reachable.

Problem 2.
In a stable matching between an equal number of boys and girls produced by the Mating Ritual, call a person lucky if they are matched up with someone in the top half of their preference list. Prove that there must be at least one lucky person.

Hint: The average number of times a boy gets rejected by girls.

COMMENTS:
Solution. Let $a$ be the average number of times a boy gets rejected by girls. It’s not possible for all the boys to be rejected an above average number of times, so some boy gets rejected $\leq a$ times.

Since the total of number rejections is the same for the boys and the girls, $a$ is also the average number of times a girl rejects boys. It’s not possible for all the girls to reject boys a below average number of times, so some girl rejects $\geq a$ boys.

Let $n$ be the number of boys. If $a \leq n/2$, then some boy gets rejected at most $\lceil n/2 \rceil$ times, and therefore he is lucky. Likewise, if $a \geq n/2$, then some girl rejects at least $\lfloor n/2 \rfloor$ boys, and therefore she is lucky. 

Problem 3.
Let $P$ be a propositional variable.
(a) Show how to express $\neg P$ using $P$ and a selection from among the constant True, and the connectives XOR and AND.

Solution.

$$\neg P \equiv P \text{ XOR True}.$$ 

The use of the constant True above is essential. To prove this, we begin with a recursive definition of XOR-AND formulas that do not use True, called the PXA formulas.

Definition. Base case: The propositional variable $P$ is a PXA formula.

Constructor cases If $R, S \in \text{PXA}$, then

- $R \text{ XOR } S$,
- $R \text{ AND } S$

are PXA’s.

For example,

$$(((P \text{ XOR } P) \text{ AND } P) \text{ XOR } (P \text{ AND } P)) \text{ XOR } (P \text{ XOR } P)$$

is a PXA.
(b) Prove by structural induction on the definition of PXA that every PXA formula \( A \) is equivalent to \( P \) or to \( \text{False} \).

**Solution.** **Proof.** **Base case:** \((A \equiv P)\). \( A \) is equivalent to \( P \) since it equals \( P \).

**Constructor case:** \((A \equiv [R \text{ AND } S])\). Each of \( R \) and \( S \) are equivalent either to \( P \) or to \( \text{False} \) by structural induction hypothesis. If either of them is equivalent to \( \text{False} \), then \( A \) is equivalent to \( \text{False} \). If both are equivalent to \( P \), then \( A \) is equivalent to \( P \). It follows that \( A \) satisfies the induction hypothesis.

**Constructor case:** \((A \equiv [R \text{ XOR } S])\). Each of \( R \) and \( S \) are equivalent either to \( P \) or to \( \text{False} \) by structural induction hypothesis. If \( R \) is equivalent to \( \text{False} \), then \( A \) is equivalent to \( S \), and therefore \( A \) satisfies the induction hypothesis; likewise if \( S \) is equivalent to \( \text{False} \), then \( A \) is equivalent to \( R \). If both \( R \) and \( S \) are equivalent to \( P \), then \( A \) is equivalent to \( P \text{ XOR } P \) which is equivalent to \( \text{False} \) and therefore satisfies the induction hypothesis. It follows in any case that \( A \) satisfies the induction hypothesis.

---

**Problem 4.**  
The set \( \text{SupSym} \) of “super-symmetric strings” is defined recursively as follows:  
**Base Case:** The 26 lower case letters of the Roman alphabet, \( a, b, \ldots, z \), are in \( \text{SupSym} \).  
**Constructor Case:** If \( \alpha \) and \( \beta \) are strings in \( \text{SupSym} \), then the string \( \alpha \beta \alpha \) is in \( \text{SupSym} \).

(a) Which of the following are super-symmetric strings? Briefly explain your answers.

(i) \( a \)  
**Solution.** Yes, by the Base Case.

(ii) \( aaaba \)  
**Solution.** No. This string is not of the form \( \alpha \beta \alpha \).

(iii) \( abcbacabcb \)  
**Solution.** Yes. Let \( \beta = \text{aca}, \alpha = \text{bcb} \). Then we have a string of the form \( a\alpha\beta\alpha \).

(iv) \( \lambda \), the empty string  
**Solution.** No. An immediate structural induction implies that all super-symmetric strings have positive length.

(v) \( abaabcbaaba \)  
**Solution.** Yes. Similar reasoning to case (iii) shows that the string \( \text{bcb} \) is in the middle of the super-symmetric string, with the string \( a \) wrapped around it, and with the string \( \text{aba} \) wrapped around that.
(b) Prove by structural induction that in any super-symmetric string, exactly one letter appears an odd number of times.

**Solution.** *Proof.* The induction hypothesis will be:

\[ P(\alpha) ::= [\alpha \text{ has exactly one letter that appears an odd number of times}] \]

**Base case:** \( \alpha \) is a single letter. Then \( \alpha \) itself is the one letter which appears an odd number of times—namely, once.

**Inductive Step.** \( \alpha = \beta \gamma \beta \) where \( \beta, \gamma \in \text{SupSym} \). The total number of occurrences of any given symbol in \( \alpha \) is twice the number of occurrences of that symbol in \( \beta \) plus the number of occurrences in \( \gamma \). So the total number of occurrences of each symbol that appears in \( \alpha \) is an even number plus the number of times it appears in \( \gamma \). But \( \gamma \in \text{SupSym} \), so only one symbol occurs an odd number of times in \( \gamma \), and this is the single symbol that occurs an odd number of times in \( \alpha \). This proves \( P(\alpha) \), as required. \( \blacksquare \)