Massachusetts Institute of Technology
6.042J/18.062J, Fall '17: Mathematics for Computer Science
Prof. Albert R Meyer, Dr. Zachary Abel

September 27
revised Thursday 28th September, 2017, 01:05

Problem Set 3

Due: October 6

Reading:

- Chapter **??** *Induction*
- Chapter **??***State Machines*

Problem 1.
Token replacing-1-2 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.

(a) List the numbers of the following predicates that are preserved invariants.

$$n_b + n_w \operatorname{rem}(n_b + n_w,\ 3) \neq 2 \tag{1}$$
$$n_w - n_b \operatorname{rem}(n_w - n_b,\ 3) = 2 \tag{2}$$
$$n_b - n_w \operatorname{rem}(n_b - n_2,\ 3) = 2 \tag{3}$$
$$n_b + n_w > 5 \tag{4}$$
$$n_b + n_w < 5 \tag{5}$$

Now assume the game starts with a single black token, that is, the start state is $(1, 0)$.

(b) List the numbers of the predicates above are true for all reachable states:

(c) Define the predicate $T(n_b, n_w)$ by the rule:

$$T(n_b, n_w) ::= \operatorname{rem}(n_w - n_b,\ 3) = 2.$$

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 different from the claim that T is a preserved invariant.

The proof of the Claim will be by induction in n using induction hypothesis $P(n)::=$

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

The base cases will be when $n \leq 2$.

- Assuming that the base cases have been verified, complete the **Inductive Step**.
- Now verify the **Base Cases**: $P(n)$ for $n \leq 2$.

Problem 2.
Let's extend the jug filling scenario of Section **??** to three jugs and a receptacle. Suppose the jugs can hold a, b and c gallons of water, respectively.

The receptacle can be used to store an unlimited amount of water, but has no measurement markings. Excess water can be dumped into the drain. Among the possible moves are:

1. fill a bucket from the hose,
2. pour from the receptacle to a bucket until the bucket is full or the receptacle is empty, whichever happens first,
3. empty a bucket to the drain,
4. empty a bucket to the receptacle, and
5. pour from one bucket to another until either the first is empty or the second is full.

(a) Model this scenario with a state machine. (What are the states? How does a state change in response to a move?)

(b) Prove that Bruce can get $k \in \mathbb{N}$ gallons of water into the receptacle using the above operations if k is a nonnegative linear combination of a, b, c (coefficients themselves are allowed to be negative).

Problem 3. (a) Prove by induction that a $2^n \times 2^n$ courtyard with a 1×1 statue of Bill in *any position* can be covered with L-shaped tiles.

(b) *(Discussion Question)* In part (a) we saw that it can be easier to prove a stronger theorem. Does this surprise you? How would you explain this phenomenon?

Problem 4.
The preferences among 4 boys and 4 girls are partially specified in the following table:

B1:	G1	G2	–	–
B2:	G2	G1	–	–
B3:	–	–	G4	G3
B4:	–	–	G3	G4
G1:	B2	B1	–	–
G2:	B1	B2	–	–
G3:	–	–	B3	B4
G4:	–	–	B4	B3

(a) Verify that

$$(B1, G1), (B2, G2), (B3, G3), (B4, G4)$$

will be a stable matching whatever the unspecified preferences may be.

(b) Explain why the stable matching above is neither boy-optimal nor boy-pessimal and so will not be an outcome of the Mating Ritual.

(c) Describe how to define a set of marriage preferences among n boys and n girls which have at least $2^{n/2}$ stable assignments.

Hint: Arrange the boys into a list of $n/2$ pairs, and likewise arrange the girls into a list of $n/2$ pairs of girls. Choose preferences so that the kth pair of boys ranks the kth pair of girls just below the previous pairs of girls, and likewise for the kth pair of girls. Within the kth pairs, make sure each boy's first choice girl in the pair prefers the other boy in the pair.

Problem 5.

For any binary string α let $\text{num}(\alpha)$ be the nonnegative integer it represents in binary notation. For example, $\text{num}(\texttt{10}) = 2$, and $\text{num}(\texttt{0101}) = 5$.

An $n+1$-*bit adder* adds two $n+1$-bit binary numbers. More precisely, an $n+1$-bit adder takes two length $n+1$ binary strings

$$\alpha_n ::= a_n \dots a_1 a_0,$$
$$\beta_n ::= b_n \dots b_1 b_0,$$

and a binary digit c_0 as inputs, and produces a length-$(n+1)$ binary string

$$\sigma_n ::= s_n \dots s_1 s_0,$$

and a binary digit c_{n+1} as outputs, and satisfies the specification:

$$\text{num}(\alpha_n) + \text{num}(\beta_n) + c_0 = 2^{n+1} c_{n+1} + \text{num}(\sigma_n). \tag{6}$$

There is a straighforward way to implement an $n+1$-bit adder as a digital circuit: an $n+1$-bit *ripple-carry circuit* has $1 + 2(n+1)$ binary inputs

$$a_n, \dots, a_1, a_0, b_n, \dots, b_1, b_0, c_0,$$

and $n+2$ binary outputs,

$$c_{n+1}, s_n, \dots, s_1, s_0.$$

As in Problem **??**, the ripple-carry circuit is specified by the following formulas:

$$s_i ::= a_i \ \text{XOR} \ b_i \ \text{XOR} \ c_i \tag{7}$$
$$c_{i+1} ::= (a_i \ \text{AND} \ b_i) \ \text{OR} \ (a_i \ \text{AND} \ c_i) \ \text{OR} \ (b_i \ \text{AND} \ c_i),. \tag{8}$$

for $0 \leq i \leq n$.

(a) Verify that definitions (7) and (8) imply that

$$a_n + b_n + c_n = 2c_{n+1} + s_n. \tag{9}$$

for all $n \in \mathbb{N}$.

(b) Prove by induction on n that an $n+1$-bit ripple-carry circuit really is an $n+1$-bit adder, that is, its outputs satisfy (6).

Hint: You may assume that, by definition of binary representation of integers,

$$\text{num}(\alpha_{n+1}) = a_{n+1} 2^{n+1} + \text{num}(\alpha_n). \tag{10}$$

Problem 6.

Suppose we want to assign pairs of "buddies," who may be of the sex, where each person has a preference rank for who they would like to be buddies with. For the preference ranking given in Figure 1, show that there is no stable buddy assignment. In this figure Mergatroid's preferences aren't shown because they don't even matter.

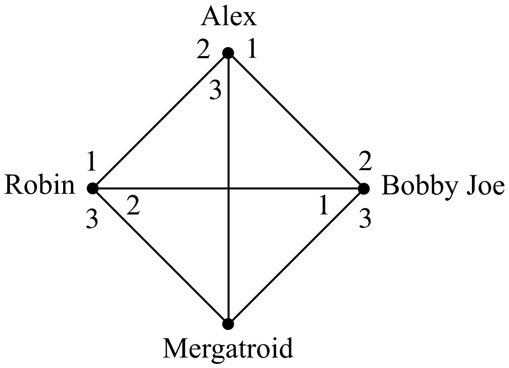

Figure 1 Some preferences with no stable buddy matching.