Massachusetts Institute of Technology
6.042J/18.062J, Fall '02: Mathematics for Computer Science
Professor Albert Meyer and Dr. Radhika Nagpal

Solutions to Problem Set 5

Problem 1. You are given two buckets, A and B, a water hose, a receptacle, and a drain. The buckets and receptacle are initially empty. The buckets are labeled with their respectively capacities, positive integers a and b. 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,
5. pour from A to B until either A is empty or B is full, whichever happens first,
6. pour from B to A until either B is empty or A is full, whichever happens first.

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

Solution. The states are triples (x, y, z) which give the current amount of water in the bucket A, bucket B, and the receptacle, respectively. The initial state is $(0, 0, 0)$.

The moves make the following transitions:

1. fill a bucket from the hose

$$(x, y, z) \to \begin{cases} (a, y, z) \text{ if filling A from the hose} \\ (x, b, z) \text{ if filling B from the hose} \end{cases}$$

2. pour from the receptacle to a bucket until the bucket is full or the receptacle is empty, whichever happens first

$$(x, y, z) \to \begin{cases} (a, y, z-(a-x)) \text{ if pouring to A and } z \geq (a-x) \\ (x+z, y, 0) \text{ if pouring to A and } z < (a-x) \\ (x, b, z-(b-y)) \text{ if pouring to B and } z \geq (b-y) \\ (x, y+z, 0) \text{ if pouring to B and } z < (b-y) \end{cases}$$

3. empty a bucket to the drain

$$(x, y, z) \rightarrow \begin{cases} (0, y, z) \text{ if emptying A to the drain} \\ (x, 0, z) \text{ if emptying B to the drain} \end{cases}$$

4. empty a bucket to the receptacle

$$(x, y, z) \rightarrow \begin{cases} (0, y, z+x) \text{ if emptying A to the receptacle} \\ (x, 0, z+y) \text{ if emptying B to the receptacle} \end{cases}$$

5. pour from A to B until either A is empty or B is full, whichever happens first

$$(x, y, z) \rightarrow \begin{cases} (0, y+x, z) \text{ if } x < (b-y) \\ (x-(b-y), b, z) \text{ if } x \geq (b-y) \end{cases}$$

6. pour from B to A until either B is empty or A is full, whichever happens first

$$(x, y, z) \rightarrow \begin{cases} (x+y, 0, z) \text{ if } y < (a-x) \\ (a, y-(a-x), z) \text{ if } y \geq (a-x) \end{cases}$$

■

(b) Prove that we can put $k \in \mathbb{N}$ gallons of water into the receptacle using the above operations if and only if $\gcd(a, b) \mid k$. *Hint:* Use the fact that if a, b are positive integers then there exist integers s, t such that $\gcd(a, b) = sa + tb$ (proven Week 5 Notes, §5.4).

Solution. We need to prove two facts (the "iff" statement has two directions):

1. If $\gcd(a, b) \mid k$ then we can put k gallons of water into the receptacle.

   Proof. Since there exist integers s, t s.t. $\gcd(a, b) = sa + tb$, then if $\gcd(a, b) \mid k$ then there exists an integer n s.t. $n \gcd(a, b) = k$, and hence $n(sa + tb) = k$. Assume without loss of generality that $sa \geq tb$ (otherwise exchange the buckets in the following argument). Then we can fill the receptacle with k gallons: First, we repeat ns times moves 1 and 4, filling the A bucket and pouring its content into the receptacle (note that if $sa \geq tb$ then $s \geq 0$). With this series of moves we will get from state $(0, 0, 0)$ to $(0, 0, nsa)$. Then, if $t = 0$, we are already done since $k = nsa$. If $t > 0$, we repeat nt times moves 1 and 4 but now using the bucket B. This gets us from $(0, 0, nsa)$ to $(0, 0, nsa + ntb) = (0, 0, k)$. If $t < 0$ we repeat $n\,|t|$ times moves 2 and 3 using the bucket B. At the end of this series we will be in state $(0, 0, nsa - n(-t)b) = (0, 0, nsa + ntb) = (0, 0, k)$ □

2. If we can put k gallons of water into the receptacle then $\gcd(a, b) \mid k$.

   Proof. We show that it is an invariant of our state machine that $\gcd(a, b)$ divides x, y and z. Thus, in particular z is always a multiple of $\gcd(a, b)$.
   Let's denote $\gcd(a, b)$ by c. The invariant is true in the initial state $(0, 0, 0)$.[1]. Each move preserves this invariant, because in each move the new values of x, y or z are always integer

[1] Remember that $(x \mid y) \longleftrightarrow (\exists_{b \in \mathbb{Z}}\ y = xb)$

linear combinations of the previous values x, y, z or a, b, i.e., they are expressed by a formula $n_1x + n_2y + n_3z + n_4a + n_5b$ for some $(n_1, n_2, n_3, n_4, n_5) \in \mathbb{Z}^5$. For example in move 2, case 1, the new value of z is expressed by the above formula with $(n_1, n_2, n_3, n_4, n_5) = (1, 0, 1, -1, 0)$. Since c divides a and b, if c divides x, y, z before the move, then it divides every such linear formula, and hence it divides the values of x, y and z after each move. □

■

Problem 2. The following algorithm, called the *Russian peasants algorithm*, can be used to multiply any two natural numbers x and y using only addition, left bit-shift (i.e., multiply by 2) and right bit-shift (*i.e.,* divide by 2 and drop any remainder) operations. The answer is accumulated in variable a; variables r and s are for temporary storage.

```
r := x;
s := y;
a := 0;
do until s = 0:
    if s is even then
        r := 2r;
        s := s/2;
    else
        a := a + r;
        r := 2r;
        s := (s − 1)/2;
```

The answer xy is the value left in the accumulator a when the procedure terminates.

(a) Model the algorithm as a state machine. That is, define the states Q, the start states Q_0, and the transitions.

Solution.

$$\begin{aligned} Q &::= \{(r, s, a) \mid r, s, a, \in \mathbb{N}\} \\ Q_0 &::= \{(x, y, 0)\} \\ \text{transitions} &::= (r, s, a) \longrightarrow (r', s', a'), \end{aligned}$$

where

$$\begin{array}{ll} r' ::= 2r, & \\ s' ::= s/2,\ a ::= a' & \text{for } s \text{ even} \\ s' ::= (s-1)/2,\ a' ::= a + r & \text{for } s \text{ odd.} \end{array}$$

■

(b) List the sequence of states that appears in an execution of the algorithm for inputs $x = 5$ and $y = 9$.

Solution. $(5, 9, 0) \rightarrow (10, 4, 5) \rightarrow (20, 2, 5) \rightarrow (40, 1, 5) \rightarrow (80, 0, 45)$ ■

(c) Find an invariant that implies the algorithm is partially correct, that is, that $s = 0$ implies $a = xy$.

Solution. We prove the following predicate is invariant:

$$P(r, s, a) ::= [xy = rs + a].$$

We need to prove that that for any states (r, s, a) and (r', s', a') such that $P(r, s, a,)$ holds and for which there exists a transition $(r, s, a) \rightarrow (r', s', a')$ that $P(r', s', a')$ also holds. The proof is by cases based on the type of action, *even* or *odd*.

***even*:** Doubling r while halving s doesn't change the product rs, and a doesn't change. So $r's' + a' = rs + a = xy$.

***odd*:** From the transition relation we know that $r' = 2r,\ s' = s - \frac{1}{2}$, and $a' = a + r$. So $r's' + a' = 2r(s - \frac{1}{2}) + a + r$. Reorganizing terms yields $r's' + a' = rs + a = xy$.

The invariant is proved.

Notice that $P(r, 0, a) ::= xy = (0s) + a = a$, as needed. ■

(d) Prove that the algorithm terminates.

Solution. We first notice that $s \geq 0$ in all states since we defined s to be a natural number. Furthermore, for $s > 0$, all transitions decrease s by at least 1. Thus, s is a decreasing, natural-number-valued variable of the state machine and, thus, the program terminates in a finite number of steps. ■

Problem 3. Four VI-A students would like a position at a company. There are four companies, each with one position for a VI-A student. Below are the students' rankings of the companies and the companies' rankings of the students.

Student	Ranking of Companies
Nikos	Lotus, IBM, Compaq, Akamai
George	Compaq, Lotus, Akamai, IBM
Radhika	Compaq, Lotus, IBM, Akamai
Tina	Akamai, Compaq, Lotus, IBM

Company	Ranking of Students
Compaq	George, Tina, Nikos, Radhika
Lotus	George, Radhika, Nikos, Tina
IBM	Radhika, George, Nikos, Tina
Akamai	Tina, George, Radhika, Nikos

Based on the rankings, the VI-A office will assign one student to each company so the assignment forms a stable marriage set.

(a) There are VI-A student, S, and a company, C, that would form a rogue pair in *any* assignment in which S was not assigned to C. Give an example of such a pair (S, C), and briefly explain why your example has this property.

Solution. Tina likes Akamai best and Akamai likes Tina best. So these two would be sure to prefer each other over their spouses if they were not married, *i.e.*, they would be a rogue pair if they were not married. George and Compaq are another such pair. ■

(b) Verify that in this case, the Mating Algorithm yields the same assignment whether Students are treated as Boys or as Girls.

Solution. Either way, the result is the following assignment:

Student	Companies
Nikos	IBM
George	Compaq
Radhika	Lotus
Tina	Akamai

■

(c) Explain why this implies that there is only one possible stable assignment (even including possible stable assignments that may not be produced by the Gale/Shapley Mating Algorithm).

Solution. In the student-optimal (pessimal) assignment, a student S, is assigned to their most (least) preferred company, C, among all possible companies they could be assigned to in a stable assignment. So in any stable assignment, S would be to be assigned to a company they like no worse than C nor more than C, so they have to be assigned to C no matter what. ■

Problem 4. The Stable Buddy Problem is a variant of the Stable Marriage Problem without the constraint that matched pairs must be Boy with Girl.

In contrast to the Boy-Girl Marriage Problem, there are buddy preferences where all buddy assignments are unstable. Give an example of four people and their preference rankings for which there is no stable way to assign buddies. Explain why.

Solution. Let the buddy preferences of four men A, B, C, D be

$$\begin{array}{rc} A: & B, C, D \\ B: & C, A, D \\ C: & A, B, D \\ D: & \text{any order} \end{array}$$

Let Y and Z be any two among A, B, and C, and let X be the remaining one. Notice that the first choices of A, B, C are arranged in a cycle (A prefers B, B prefers C, and C prefers A), so that X must be the first choice of either Y or Z.

Now for any matching into pairs, let Y and Z be the two among A, B, C who are paired, and X be the one paired with D. So one of Y and Z, say Y, has X as his first choice. But since A, B, and C each rank D last, X prefers Y to his own buddy, D. So X, Y are a rogue couple. Notice that the ranking of A, B, and C by D doesn't matter. ■

Problem 5. Here is the generalization of the "choose-a-pair" game from Week 5, Friday Class Problems to "choose-a-triple." The rules are:

Player 1 chooses any triple in $\mathbb{N}^3$. Then, starting with Player 2, the players alternate moves, choosing as a move any triple, $\mathbf{t} \in \mathbb{N}^3$, such that no previous move is $\preceq_c \mathbf{t}$. A player wins when the other player chooses the origin (0,0,0).

For example, Player 1 might choose the 3-tuple (8, 9, 10). Possible subsequent choices might then be

$$(7, 8, 9), (0, 1, 67), (83, 0, 0), (1, 0, 0), (0, 0, 1)(0, 1, 0)$$

This finally leaves Player 2 with only the move (0,0,0), and the game now ends with his loss.

Prove that there is a winning strategy for the choose-a-triple game.

Solution. By the Fundamental Theorem for terminating games of perfect information, one of the players will have a winning strategy providing no game can continue for a infinite number of moves.

To prove that the game always terminates, choose any well-founded total order, $\preceq$ on the set $\mathbb{N}^3$ of possible moves. Lexicographic order, for example, will do nicely.

At any point in the game after the first move, let $\mathbf{x}$ be the least triple in the well-ordering such that the x-coordinate of $\mathbf{x}$ is the minimum, x_m, of the x coordinates of all the previous moves. Likewise for $\mathbf{y}$ and $\mathbf{z}$. Then $x_m + y_m + z_m$ is a *weakly* decreasing natural-number valued variable because none of the minimum values x_m, y_m and z_m can increase as more moves are made, so neither can their sum.

Now the only possible moves that do not decrease $x_m + y_m + z_m$ must occur in the six-sided trapezoidal polyhedron with corners at $\mathbf{x}$, $\mathbf{y}$, $\mathbf{z}$. Define k to be the number of possible moves left in the polyhedron. In other words, k is the number of points (x, y, z) where $x \geq x_m, y \geq y_m, z \geq z_m$, and (x, y, z) has not already been eliminated by a previous move.

Define the *size* of a game position to be $(x_m + y_m + z_m, k)$. Now size is a decreasing variable under the lexicographic ordering on $\mathbb{N}^2$. Since this lexicographic order is well-founded, the size of the states reached in any game must have a minimal value. Such a value must occur at a state in which the game is terminated. ■