CS3230 Tutorial 1 (Introduction) Sample Solutions

January 28, 2017

1 Question 1

Let us first prove two assertions that we need later on. First, if d divides two integers u and v, it also divides both $u+v$ and $u-v$. By definition of division, there exist integers s and t such that $u = sd$ and $v = td$. Therefore $u + v = sd + td = (s + t)d, u - v = sd - td = (s - t)d$, i.e., d divides both $u + v$ and $u - v$.

Second, note that if d divides u, it also divides any integer multiple ku of u. Indeed, since d divides u, $u = sd$. Hence $ku = k(sd) = (ks)d$, i.e., d divides ku.

Now we can prove the assertion in question. Let us divide the proof into two statements. Here $r = m\%n = m - qn$:

- (i) If $d|m$ and $d|n$ then $d|n$ and $d|r$.
- (ii) If $d'|n$ and $d'|r$ then $d'|m$ and $d'|n$.

Start with statement (i). For any pair of positive integers m and n, if $d = gcd(m, n)$, then d divides both m and n, and d also divides both n and $r = m\%n = m - qn$. $d|r$ because $d|(m - qn)$ according to the first two rules we developed (i.e., $d|m$ and $d|(qn)$, and furthermore $d|(m - qn)$). Therefore d is a common divisor of n and r.

Similarly for statement (ii), $d' = gcd(n, r)$ divides both n and $r = m\%n = m - qn$; it also divides both $m = r + qn$ and n. Thus, the two pairs (m, n) and (n, r) have the same finite nonempty set of common divisors, including the largest element in the set, i.e., $gcd(m, n) = gcd(n, r)$.

2 Question 2

Let us denote the set of numbers that can eventually be written as S. We are really interested in the parity of $|S| - 2$ (the parity reflects whether the number of elements in the set $|S| - 2$ is even or odd), which is also the parity of $|S|$. If the parity is odd, one should choose to go first; if it is even, one should choose to go second.

Let us look at an example, where the two starting numbers are 18 and 15. Then the sequence of numbers written to the board could be:

15 18 3 12 9 6

Firstly we prove that the number $d = gcd(m, n)$ can be eventually written to the set due to the subtraction version of Euclid's algorithm: $gcd(m, n) = gcd(n, m - n)$. In our example $d = 3$. Note that a slightly modified proof of Question 1 can be used here. We know that if d divides both u and v it also divides $u - v$, i.e., $u - v = xd$. This means any differences that are generated from u, v, $u - v$, etc., are also multiples of d. Since the differences cannot get bigger, only smaller, we eventually will generate the number xd, with $x = 1$.

Next we prove that all the numbers in S must be divisible by d (by mathematical induction). Now we can count that the total number of elements (numbers) in S is $\max(m, n)/gcd(m, n)$. Consequently, if $\max(m, n)/gcd(m, n)$ is odd, one should choose to go first; if it is even, one should choose to go second.

3 Question 3

Let's rewrite the Euclid's algorithm in a recursive way:

```
Procedure Euclid(m,n)
if m%n=0
    return n
else
    d <-Euclid(n, m%n)
    return d
```

By modifying the above algorithm, we have the extended Euclid's algorithm:

```
Procedure Euclid_ext(m,n)
if m%n=0
    return <0,1>
else
    <p',q'> <-Euclid_ext(n, m%n)
    r <-m%n
    t <-(m-r)/n
    p <-q'
    q <-p'-q'*t
    return <p,q>
```

4 Question 4

There exist two solutions for this question. Let 1, 2, 5, 10 be labels representing the men of the problem, and the number in the parenthesis be the total amount of time elapsed. The following sequence of moves solves the problem:

```
1,2,5,10 ========= (0)
    5,10 ========= 1,2 (2)
  1,5,10 ========= 2 (3)
       1 ========= 2,5,10 (13)
     1,2 ========= 5,10 (15)
         ========= 1,2,5,10 (17)
```

An alternative solution:

```
1,2,5,10 ========= (0)
    5,10 ========= 1,2 (2)
  2,5,10 ========= 1 (4)
       2 ========= 1,5,10 (14)
     1,2 ========= 5,10 (15)
         ========= 1,2,5,10 (17)
```

Mathematical solution:
If n persons $\{1 \dots n\}$ with walking times `t(1) <= <t(2) <= ... <= t(n)` have to cross the bridge, then the optimal schedule is:

```
 n=1:
         crossing       time
         {1}      ->    t(1)

         total time:    t(1)

 n=2:
         crossing       time
         {1, 2}   ->    t(2)
```

```
        total time:    t(2)

n=3:
        crossing       time
        {1, 2}   ->    t(2)
        {1}      <-    t(1)
        {1, 3}   ->    t(3)

        total time:    t(1) + t(2) + t(3)

n>3:
     Induction step: n -> n-2

     case: I                          case: II

        crossing       time              crossing       time
        {2}       <-   t(2)              {1}       <-   t(1)
        {n-1, n}  ->   t(n)              {1, n-1}  ->   t(n-1)
        {1}       <-   t(1)              {1}       <-   t(1)
        {1, 2}    ->   t(2)              {1, n}    ->   t(n)

     Total time of this part:  t(1) + t(n) + Min( 2*t(2), t(1) + t(n-1) )
     (For t1=1, t2=2, t3=5, and t4=10 we have: 1+10+Min(2*2,1+5) = 15)
```

So the slow ones walk in pairs; the faster ones go with the fastest.
Other walking times in use:

```
  t : (1,2,3,4) = (5,10,20,25)   min ct = 60   // European version
  t : (1,2,3,4) = (1,2,5,10)     min ct = 17   // American version
  t : (1,2,3,4) = (2,3,5,8)      min ct = 19   // Torsten Sillke
  t : (1,2,3,4) = (2,2,3,3)      min ct = 11   // Torsten Sillke
  t : (1..5)    = (1,3,6,8,12)   min ct = 29   // Plastelina
  t : (1..6)    = (1,3,4,6,8,9)  min ct = 31   // Dick Hess
```

(min ct is the minimal crossing time.)

5 Question 5

It is not difficult to observe that if the total number of flips is odd, the bits will be 1, otherwise 0. A bit at position i ($1 \leq i \leq n$) gets flipped at j-th iteration ($1 \leq j \leq n$) if and only if there exits a certain k such that $k \times j = i$. In other words, j is a divisor of i. Hence the number of flips of the i-th bit is exactly the number of divisors of j.

Note that if j divides i, i.e., $i = jk$, then k divides i too. Hence all the divisors of i can be paired (e.g., for $i = 12$, such pairs are 1 and 12, 2 and 6, 3 and 4) unless i is a perfect square (e.g., for $i = 16$, 4 does not have another divisor to be matched with). This implies that i has an odd number of divisors if and only if it is a perfect square, i.e., $i = j^2$. Hence bits that are in the positions that are perfect squares and only such bits will be zero after the last pass. The total number of such positions not exceeding n is equal to $\lfloor \sqrt{n} \rfloor$: these numbers are the squares of the positive integers between 1 and $\lfloor \sqrt{n} \rfloor$ inclusively.

Any bugs and typos, please report to Roger Zimmermann (rogerz@comp.nus.edu.sg).