CS3230 Tutorial 3 (Brute Force Algorithms) Sample Solution

February 3, 2017

1 Question 1

1. A counter example showing that selection sort is not stable:

   `2' 2'' 1`

   After selection sort:

   `1 2'' 2'`

2. Bubble sort is stable. It follows from the fact that it swaps adjacent elements only provided $A[j+1] < A[j]$.

3. The idea is to introduce a flag. When no swapping happens in the first iteration, we simply stop performing further actions.

```
for i <- 0 to n-2 do
  swapped <- false
  for j <- 0 to n-2-i do
    if A[j+1] < A[j]
      swap A[j] and A[j+1]
      swapped <- true
    fi
  od
  if (!swapped)
    break
  fi
od
```

2 Question 2

The worst case requires $m(n-m+1)$ character comparisons. We are therefore asked to construct a worst case example.

The general worst case example is as follows:
The text of length n looks like this: `AAA...AB`.
The pattern of length m looks like this: `AAA..AC`.
Note that `C`$\neq$`B`.

3 Question 3

Note that the number of desired substrings that start with an `A` at a given position i ($0 \leq i < n-1$) in the text is equal to the number of `X`'s to the right of that position. This leads to the follwing simple algorithm:

Initialize the count of the desired substrings to 0. Scan the text left to right doing the following for every character except the last one: If an `A` is encountered, count the number of all the `X`'s following it and add this number to the count of desired substrings. After the scan ends, return the last value of the count.

For the worst case of the text composed of n `A`'s, the total number of character comparisons is $n + (n-1) + \ldots + 2 = n(n+1)/2 - 1 \in \Theta(n^2)$.

(Note: there exist $O(n)$ algorithms for this question: scanning through the array while counting the number of `A`'s to the left of each `X`. Also, the question did not explicitly ask you to count the number of distinct substrings only, therefore you should not assume so.)

4 Question 4

We will prove by induction that there will always remain at least one professor not hit by a roti prata. The basis step is easy: If $n = 3$, the two professors with the smallest pairwise distance between them throw at each other, while the third professor throws at one of them (whoever is closer). Therefore, this third professor remains "unharmed."

For the inductive step, assume that the assertion is true for odd $n \geq 3$, and consider $n + 2$ profs. Again, the two profs with the smallest pairwise distance between them (the closest pair) throw at each other. Consider two possible cases as follows. If the remaining n profs all throw at one another, at least one of them remains "unharmed" by the inductive assumption. If at least one of the remaining n profs throws at one of the closest pair, among the remaining $n-1$ profs, at most $n-1$ roti pratas are thrown at one another, and hence at least one prof must remain "unharmed" because there are not enough roti pratas to hit everybody in that group. This completes the proof.

5 Question 5

The given algorithm will select the minimal, available element in each row. This strategy will not lead to an optimal solution. Consider the following counter example. Let the cost matrix be

	J1	J2	J3	J4
P1	9	2	7	8
P2	6	4	3	7
P3	5	8	1	8
P4	7	6	9	4

The permutation 2134 gives the smallest assignment cost $2+6+1+4 = 13$. The proposed algorithm will select 2314 with a cost of $2+3+5+4 = 14$.

6 Question 6

Additional assumption: Different letters represent distinct digits.

Since the letter-digit correspondence must be one-to-one and there are only ten distinct decimal digits, the exhaustive search needs to check $P(10,k) = 10!/(10-k)!$ possible substitutions, where k is the number of distinct letters in the input. (The requirement that the first letter of a word cannot represent 0 can be used to reduce this number further.) In the example $k = 8$, hence $P(10,8) = 1,814,400$. Thus a program should run in a quite reasonable amount of time on today's computers. Note that the program should check this equality: $10^3 \times (S+M) + 10^2 \times (E+O) + 10 \times (N+R) + (D+E) = 10^4 \times M + 10^3 \times O + 10^2 \times N + 10 \times E + Y$.

A possible answer is:

```
 9567
+1085
-----
10652
```

7 Question 7

- (Exhaustive search) Number positions in an 3-by-3 matrix from 1 through 9. Generate a permutation of the numbers 1 through 9, put them in the corresponding positions of the matrix, and check the magic-square equality (sum to 15) for every row, every column, and each of the two main diagonals of the matrix. The search space of this algorithm is 9!.

- (Better method) Not hard to prove that the number in the middle of the square must be 5. We can develop an algorithm that requires two `for` loops only, which gives a search space of less than 9×8.

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