Homework 3

Lei Zhang

February 23, 2016

1 Exercises 3.1 - 5

Algorithm 1: Identifying topologies

//Input: An boolean matrix $A[0..n-1, 0..n-1]]$, where $n > 3$;
//Output: 0 denotes the topology is a ring, 1 denotes the topology is a star, 2 denotes the topology is a fully connected mesh, 3 denotes the topology is none of the three choices;

for $i \leftarrow 0$ to $n-1$ **do**
    $sumOfLine[i] \leftarrow 0$
end for
for $i \leftarrow 0$ to $n-1$ **do**
    for $j \leftarrow 0$ to $n-1$ **do**
        $sumOfLine[i] \leftarrow sumOfLine[i] + A[i,j]$
    end for
end for
for $i \leftarrow 0$ to $n-1$ **do**
    if $sumOfLine[i] == 2$ **then**
        $countOf_2 += 1$
    end if
end for
for $i \leftarrow 0$ to $n-1$ **do**
    if $sumOfLine[i] == n-1$ **then**
        $countOf_{nminus1} += 1$
    end if
end for
if $countOf_2 == n-1$ **then**
    return 0
end if
if $countOf_2 == n-2$ and $countOf_{n-1} == 1$ **then**
    return 1
end if
if $countOf_{nminus1} == n-1$ **then**
    return 2
end if

return 3

Time efficiency is $\Theta(n)$

2 Exercises 3.1 - 7

3 Exercises 3.1 - 8

- outer loop 1: $min = 2$. The list is A, X, E, M, P, L, E
- outer loop 2: $min = 2$. The list is A, E, X, M, P, L, E
- outer loop 3: $min = 6$. The list is A, E, E, M, P, L, X
- outer loop 4: $min = 5$. The list is A, E, E, L, P, M, X
- outer loop 5: $min = 5$. The list is A, E, E, L, M, P, X
- outer loop 6: $min = 5$. The list is A, E, E, L, M, P, X

4 Exercises 3.1 - 11

- outer loop 1: The list is E, A, M, P, L, E, X
- outer loop 2: The list is A, E, M, L, E, P, X
- outer loop 3: The list is A, E, L, E, M, P, X
- outer loop 4: The list is A, E, E, L, M, P, X
- outer loop 5: The list is A, E, E, L, M, P, X
- outer loop 6: The list is A, E, E, L, M, P, X

5 Exercises 3.1 - 12

5.1 a.

Assume the list was not sorted before a outer loop, at least one element must be greater than it's former element. If so, at least one exchange must be made. Thus, if bubble sort makes no exchages on its pass through a list, the list must be sorted.

5.2 b.

Algorithm 2: BubbleSort

```
//Input: An array A[0..n − 1] of orderable elements
//Output: Array A[0..n − 1] sorted in nondecreasing order
  numOfSwap ← 0
  for i ← 0 to n − 2 do
      for j ← 0 to n − 2 − i do
          if A[j + 1] < A[j] then
              swapA[j]andA[j + 1]
              numOfSwap+ = 1
          end if
      end for
      if numOfSwap == 0 then
          Break
      end if
  end for
```

5.3 c.

It is known that the standard bubble sort has quadratic performance in the worst case. The worst case is to sort a array in increasing order. For this situation, bubble sort will exchage elements in every outer loop. The improvement won't get a change to take affect. Thus, worst-case efficiency of the improved version is still quadratic.

6 Exercises 3.4 - 10

6.1 a.

$$\sum_{i=0}^{n^2} i = \frac{n^2(n^2+1)}{2}$$

6.2 b.

Step 1: Generate all permutations of 1 to n^2.

Step 2: Fill in the numbers of permutations to matrices.

Step 3: Test all the matrices if it's a magic square. It's a magic square only if each row, each column, and each main diagonal of the matrix has the same sum.

Step 4: Output all the magic squares.

6.3 c.

6.4 d.

7 Exercises 3.5 - 1

7.1 a.

Adjacency matrix:

	A	B	C	D	E	F	G
A	0	1	1	1	1	0	0
B	1	0	0	1	0	1	0
C	1	0	0	0	0	0	1
D	1	1	0	0	0	1	0
E	1	0	0	0	0	0	1
F	0	1	0	1	0	0	0
G	0	0	1	0	1	0	0

Adjacency lists:

$a \to b \to c \to d \to e$
$b \to a \to d \to f$
$c \to a \to g$
$d \to a \to b \to f$
$e \to a \to g$
$f \to b \to d$
$g \to c \to e$

8 Exercises 3.5 - 4