# Solutions — Sample Midterm 1

## Problem 1:

The transition table is given below. The beginning state is $\{A\}$, the accepting states are the ones containing $A$, that means $\{A\}, \{A,C\}, \{A,B\}, \{A,B,C\}$.

| State | 0 | 1 |
|---|---|---|
| {A} | {B} | {B,C} |
| {B} | {A} | {A,C} |
| {B,C} | {A,B} | {A,B,C} |
| {A,C} | {B} | {B,C} |
| {A,B} | {A,B} | {A,B,C} |
| {A,B,C} | {A,B} | {A,B,C} |

## Problem 2:

$T(n) = 4T(n/2) + c$: Compare with $n^{\log_2(4)} = n^2$. Since $c = o(n^{2-\epsilon})$ for any $\epsilon \in (0,2)$, we are in Case 1 of the MT and therefore $T(n) = (n^2)$.

$T(n) = 3T(n/2) + c$: Compare with $n^{\log_2(3)}$. Since $c = o(n^{\log_2(3)-\epsilon}$ for any $\epsilon \in (0, \log_2(3))$, we are still in Case 1 of the MT and therefore $T(n) = n^{\log_2 3}$.

## Problem 3:

(a) The recurrence is not in the form for the MT.

(b) Compare $\sqrt{n}$ with $n^{\log_{25}(5)} = n^{1/2} = \sqrt{n}$. Clearly, they are equal and we are in Case 2 of the MT. Therefore $T(n) = \log(n)\sqrt{n}$.

(c) Compare $\log(n)n$ with $n^{\log_2(2)} = n$. Because of $\lim_{n\to\infty} \frac{n \log n}{n} = \infty$, we are not in Case 2 of the MT (if you use the book, wikipedia has a more extensive version of the MT). Because for any $\epsilon > 0$, we have

$$\lim_{n\to\infty} \frac{n \log n}{n^{1+\epsilon}} = \lim_{n\to\infty} \frac{\log n}{n^{\epsilon}} = \lim_{n\to\infty} \frac{1/n}{\epsilon n^{-1+\epsilon}} = (1/\epsilon) \lim_{n\to\infty} n^{-1+1-\epsilon} = (1/\epsilon) \lim_{n\to\infty} n^{-\epsilon} = 0.$$

This implies $n \log(n) \notin \Omega(n^{1+\epsilon})$ for any $\epsilon > 0$. We are therefore not in Case 3 of the MT.

## Problem 4:

Let $n$ be the difference between hi and lo. The algorithm does some things, then has a loop with $n$ iterations, during which it calls the function with an input of $n-1$. Therefore the recurrence is $T(n) = nT(n-1)$. Obviously, we can get rid of the loop and get a much better algorithm.

**Problem 5:**

We use an induction argument. Before the first run of the loop, the loop invariant has something to say about an empty array and is therefore vacuously true. This establishes the base case.

Assume that the loop invariant is true before iteration $j$. We want to show that the invariant is true after iteration $j$. Assume that there is a violation of the order in $A[0..j]$. If the offending elements belonged to $A[0..j-1]$, then they must have been out of order before, because the loop potentially moves them one to the right, but does not change their order. So, this cannot have been the case. Therefore, one of the offenders is $A[j]$. What about the other one? There are two cases: the other offender is smaller than $A[j]$ or the other offender is larger than $A[j]$.

If the other offender is smaller than $A[j]$, but now is located to the right of $A[j]$, then if was looked at in the loop, in which case we would have gotten out of the while loop before changing its position. Therefore, it is not to the right of $A[j]$ and this case cannot have happened.

If the other offender is larger than $A[j]$, then we did not move $A[j]$ by it. For this to have happened, we needed to have stopped at an element $A[k]$ of $A[0..j-1]$ because it was smaller than $A[j]$. But this cannot have happened either: $A[k]$ was smaller than $A[j]$, $A[k]$ was located to the right of the offender before the loop, and the other offender is larger than $A[j]$, which implies that the array $A[0..j-1]$ was out of order.