Analysis of Algorithms - Assignment 2

Colby Rush

March 13, 2014

1 Section 3.1

1.1 No. 1

1. An algorithm based on a min-max principle to determine what item to choose in a tree
2. A problem that will run forever without a specific input, aka a deciding problem

1.2 No. 3

1. All of them are brute force.

1.3 No. 4

1.

2 Section 3.2

2.1 No. 4

1. It would do 43 comparisons, looking for G until it found it, then checking what the next character was, then continuing over again until reaching the end.

2.2 No. 5

1. Four successful and then one unsuccessful, shifting, getting 3 success and then another unsuccessful, shifting, etc., so it would be 5 * 996 = 4980 tries.
2. Similar to above, making one unsuccessful comparison each try and then shifting, yielding 1*996 = 996 tries.
3. Again, similar, with one unsuccessful and one successful, yielding 2*996 = 1992 tries.

2.3 No. 6

1. The worst case scenario would be along the line of a pattern of n-1 characters, where n is the length of the pattern to be searched, which matches all but one of the pattern to be searched. Identical to problem 5's a. question, however long you wish.

3 Section 3.3

3.1 No. 2

1. The merge sort algorithm is more efficienct, by splitting the map into two and finding the closest pair in each half and then checking those two.

3.2 No. 3

1. The village that is halfway is always the best choice for the post office. However, for even villages, it is literally the halfway point, as in halfway between the farthest pair, while for odd it is the other halfway point, where the number of villages on the left is equal to those on the right.

2. Find the village closest to the middle point, then find the farthest pair from that village, and place it halfway between those two villages.

3.3 No. 5

1.

3.4 No. 9

1. Find the smallest x point, then the largest x point. If there are two same points, find the corresponding largest/smallest y point.

4 Section 3.4

4.1 No. 1

1. $\Theta(n!)$

2. 15, 16, 18, and 20, respectively

4.2 No. 5

1. $\begin{pmatrix} 1 & 2 \\ 2 & 9 \end{pmatrix}$

4.3 No. 7

1. Make a subset of k vertices and check each pair in the subset for a connecting edge, repeat until true

5 Section 3.5

5.1 No.1

1. $$\begin{pmatrix} 0 & a & b & c & d & e & f & g \\ a & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ b & 1 & 0 & 0 & 1 & 0 & 1 & 0 \\ c & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ d & 1 & 1 & 0 & 0 & 0 & 1 & 0 \\ e & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ f & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ g & 0 & 0 & 1 & 0 & 1 & 0 & 0 \end{pmatrix}$$

2. Reached by a,b,d,f (fail) a,c,g,e. Popped by f,d,b,e,g,c,a

5.2 No. 4

1. Order of a,b (fail), a,c,d,e,f,g

6 Section 4.1

6.1 No. 4

1. Make the power set $P_(n-1)$ of n elements, for each set it contains, add both that set and the sets of that element to P_n. Output P_n

6.2 No. 7

1. E, X, A, M, P, L, E
2. A, E, X, M, P, L, E
3. A, E, E, X, M, P, L
4. A, E, E, L, X, M, P
5. A, E, E, L, M, X, P
6. A, E, E, L, M, P, X

7 Section 4.4

7.1 No. 3

1. 4 comparisons
2. 3, 27, 31, 42, 70, 81, 85, 98
3. $log_2 4 = 2$ avg
4. $log_2 15 =$ approx. 4

7.2 No. 4

1. Avg of $log_2 1000000$ = approx. 20 for binary, whereas sequetial is $(n+1)/2$ = approx. 500000. So binary is 25000 times faster.

7.3 No. 8

1. Divide and conquer searches
2. T(n) = $\begin{cases} 1 & \text{if } n = 1 \\ T(n/2) + T(2n/3) & \text{if } n \text{ greater than } 1 \end{cases}$

7.4 No. 10

1.