Homework 1 of Sequence Informatics 2008

Dae-Ki Kang

September 12, 2008

Deadline: September 30, 2008

1. Consider the following sort algorithm (Figure 1) where A is an array with n integers:

InsertionSort(A,n):
begin

(a) for $i \leftarrow 2$ to n do

(b) $\quad key \leftarrow A[i]$

(c) $\quad j \leftarrow i - 1$

(d) $\quad$ while $j > 0$ and $A[j] > key$ do

(e) $\quad\quad A[j+1] \leftarrow A[j]$

(f) $\quad\quad j \leftarrow j - 1$

(g) $\quad A[j+1] \leftarrow key$

end.

Figure 1: Pseudo-code of Insertion Sort

Give lower and upper bounds on the running time of the InsertionSort using the O (big-Oh) and Ω (big-omega) notation.

2. Consider the following recursive algorithm (Figure 2) where A is an array with n integers:

Give a tight asymptotic bound of the algorithm using the Θ notation.

3. Write a Graphviz DOT file for the following decision tree (Figure 3).

Recursive(A,n):
begin

(a) if $i > 1$ then

(b) for $i \leftarrow$ n - 1 downto 1 do

(c) $A[n] \leftarrow A[i] + A[n]$

(d) Recursive(A, $n-1$)

end.

Figure 2: Pseudo-code of Recursive Algorithm

4. Write a Graphviz DOT file for the following finite automata (Figure 4).

5. Construct the string matching automation for the pattern $P = aabab$ and illustrate its operation on the text string $T = aaababaabaababaab$.

6. Compute the KMP prefix function π for the pattern $P = ababbabbabbababbabb$ when the alphabet is $\Sigma = \{a, b\}$.

7. Briefly describe Boyer-Moore algorithm for string matching. Explain why Boyer-Moore algorithm is sometimes sublinear. For the description of the algorithm, check the following URI.

   `http://www.cs.utexas.edu/users/moore/best-ideas/string-searching`

8. Determine a longest common sequence (LCS) of 10010101 and 010110110.

9. Draw a suffix tree or a generalized suffix tree for the following strings.

   (a) "banana"

   (b) "cacao"

   (c) "bagle" and "beagle".

10. Let a circular string $\hat{s}$ which is constructed by connecting the ending character of the original linear string s to its starting character. Design a linear string matching algorithm for a circular string.

    The algorithm should be able to find a match across the boundary of the original string. For example, if the original string is "Beethoven virus", the algorithm can find "rusBe", "sBeetho", "virusBeethoven", "eethoven", "Beethoven virus", etc.

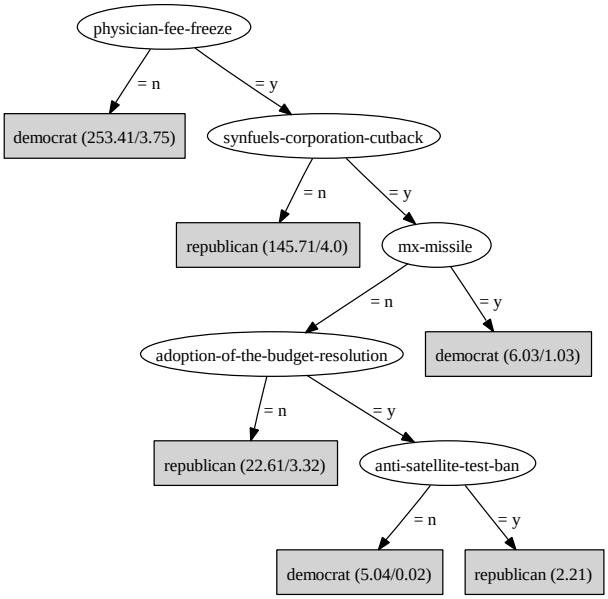

Figure 3: Decision tree of VOTE data set

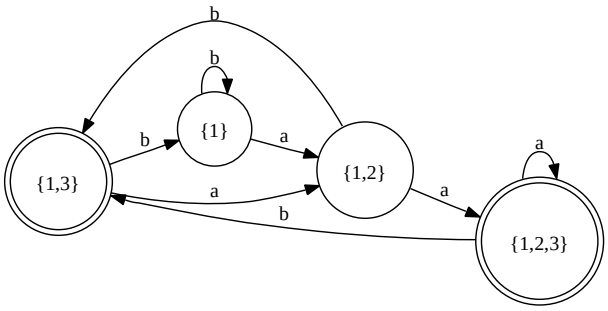

Figure 4: Finite state automata