Problem Set 5

CSCI 3104 Spring 2014
Cristobal Salazar
07/22
Partner: Alex Tsankov

Problem 1

See code.

Problem 2

a) σ will take $8 * 761 = 6088$ bits. 761 symbols each taking 8 bits.

b) With the entropy $H = -\sum_{i=1}^{|\Sigma|}(f_i/l) * log_2(f_1/l)$, the theoretical lower limit(and upper limit) is $H * l$. Where $| \Sigma | = 31$ is the number of symbols, f_i is the number of times a certain character is repeated, and $l = 761$ is the number of symbols. The lower bound per symbol is around 4.5, so the lower bound to encode σ is 3425 bits

c) When we run our code, the number of bits we get to encode σ is around 4000 bits, which is higher than our theoretical lower limit, but this could be because our code was not optimized.

d) To do the codebook, we need to make a table, where each ASCII character is mapped with its corresponding code word. A binary code word that is k bits long, takes k bits to write in the codebook.

Problem 3

I do not know

Problem 4

a)i) This algorithm will only create more branches, and traverse those branches, only if the n^{th} number in the sequence has not yet been defined. This means that to calculate $F(7)$, if we have already calculated the previous 6 terms, we will only need to recurse once, down to $F(7-1=6)$, and $F(7-2=5)$. Since those have already been calculated, and are in the array, we do not need to recurse any further, and $F[7]$ is then defined in the array. The array F is filled only if a term has not yet been defined.

a)ii) Since there is only one recursion per $F(n)$(worst case: calls itself at most $2*n$ times.), since all previous elements have already been defined, the algorithm has $\Theta(n)$ running time.

b) Gollum's algorithm has a running time of $\Theta(n)$ just like our MemFib algorithm, because a for-loop of n elements will still take $\Theta(n)$ time. Gollum's algorithm takes $O(1)$ amount of memory, which beats MemFib(takes $O(n)$ memory), because he is using constant operations ($O(1)$). Although, for big numbers, MemFib will beat Gollum's performance, because his alorgithm will have to recompute everything.

c) The running time for this algorithm is $O(n)$, and it takes constant $O(1)$ memory space. This is very similar to Gollum's last algorithm, except we do not need to recompute every number, we only keep the important numbers. Choosing the intermidiate numbers gives us the running time(in the long run) of MemFib, and the memory space of Gollum's previous algorithm, so it is the best of both worlds.

d) I do not know.

Problem 5

a) I do not know.

b) I do not know.