Comp 555: BioAlgorithms -- Fall 2013

Problem Set #1

Issued: 9/3/2013 Due: In class 9/26/2013

SOLUTIONS

Homework Information: Some of the problems are probably too long to attempt the night before the due date, so plan accordingly. No late homework will accepted. However, your lowest homework will be dropped. Feel free to work with others, but the work you hand in should be your own.

Question 1. What is the expected length of shortest subsequence that appears nowhere else in the human chromosome of 1 which is 249 million bp long? What are the assumptions of your answer, if any? How many times would one expect to see a given 12-base subsequence appear in the same chromosome, assuming all bases are independent and equally likely?
Answer: Assuming a genome of length N where nucleotides are equally likely and their co-occurrences independent, the expected number of appearances of any k-mer would be $N/4^k$. Our objective is to make this expectation less than 1. Thus, $249 \times 10^6/4^k < 1.0$. Solving for k suggests a length of at least 14. Using the same equation, one would expect a given 12-mer to appear ,$249 \times 10^6/4^{12} \approx 14.8$ times.

Question 2. Calculate the entropy of all possible dinucleotides in the sequence in this file. What are the most and least frequent dinucleotides? Ungraded optional question:What might be the reason for this imbalance?

Answer: For a dinucleotide of frequency p entropy is -log2(p)

Dinucleotide	Count	Freq	Entropy (base 2)
AA	242	0.201835	2.308752706
AC	98	0.081735	3.612906099
AG	112	0.093411	3.420261021
AT	97	0.080901	3.627703101
CA	128	0.106756	3.227615943
CC	66	0.055046	4.183221824
CG	13	0.010842	6.527176225
CT	45	0.037531	4.735762847
GA	106	0.088407	3.499695489
GC	38	0.031693	4.97968843
GG	41	0.034195	4.870063939
GT	16	0.013344	6.227615943
TA	73	0.060884	4.037791385
TC	50	0.041701	4.583759754
TG	35	0.029191	5.098332926
TT	39	0.032527	4.942213725

Question 3. Design an algorithm for computing n-th Fibonacci number that requires less than O(n) time. Hint: you probably want use the close form expression but note that computing it naively still requires O(n) time since each multiplication is a single operation.

Answer: Closed form expression:

$$F_n = \frac{\varphi^n - \psi^n}{\varphi - \psi} = \frac{\varphi^n - \psi^n}{\sqrt{5}}$$

Where

$$\varphi = \frac{1+\sqrt{5}}{2} \approx 1.61803\,39887\cdots$$ and

$$\psi = \frac{1-\sqrt{5}}{2} = 1 - \varphi = -\frac{1}{\varphi} \approx -0.61803\,39887\cdots$$

We can compute phi(n) in a recursive way.

```
FUNCTION Phi(n)
IF n==1
RETURN (1+sqrt(5))/2
ELSE IF n is even
RETURN Phi(n/2)*Phi(n/2)
ELSE
RETURN Phi(n/2)*Phi((n+1)/2)
```

Other function (Ksi) can be computed similarly

So Fibonacci function becomes

```
FUNCTION Fib(n)
  RETURN (Phi(n)-Ksi(n) )  / sqrt(5)
```

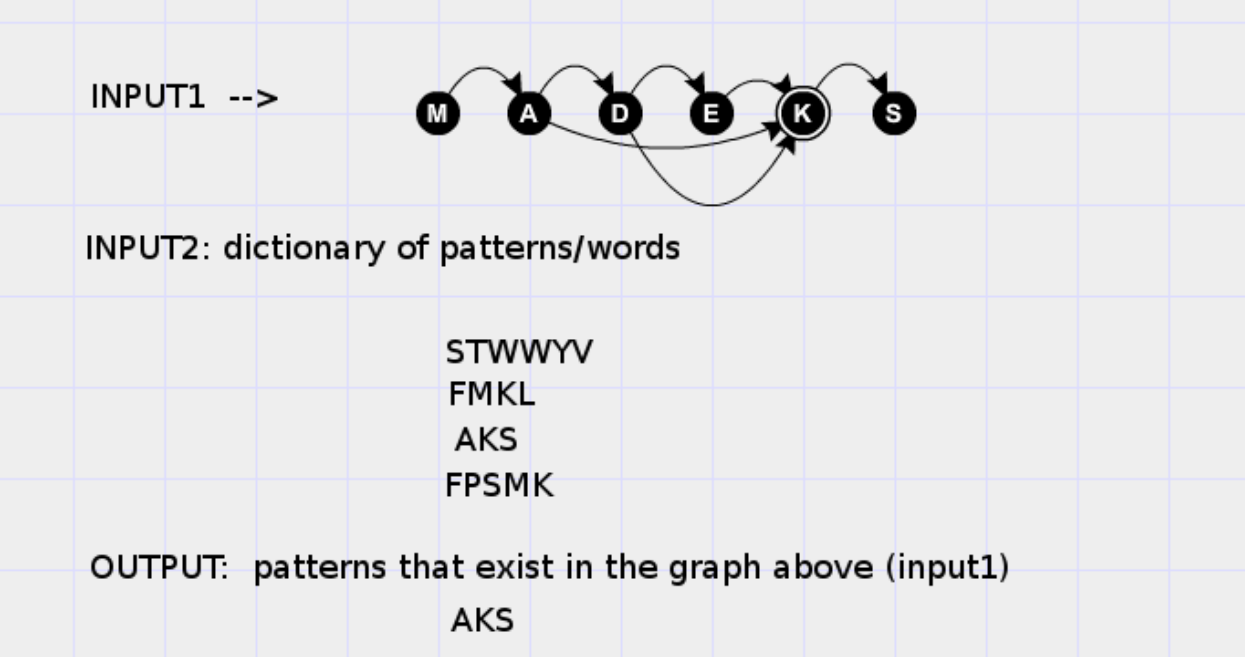
Question 4. Reconsider the incorrect "Change Problem" algorithm from Lecture 3. In how many of the 99 possible inputs for M does it given an incorrect answer for the denominations c = (35, 25, 10, 5, 1)? Suppose that you are desigining coinage for a new nation, but you are limited to minting only 5 denominations. How do you choose them such that the average number of coins is minimized for all values of change from 1 to 99? Will the greedy algorithm work with your proposed coinage?
Answer: incorrect solutions are 50,51,52,53,54,85-89
Best coinage [33, 23, 16, 5, 1] 3.32323232323. Many solutions have 3.35353535354 averages. A brute force algorithm can be implemented to try all possible combinations.

Question 5. Design a brute force algorithm for Probed Partial Digest Problem described below and suggest a branch-and-bound approach to improve its performance
Probed Partial Digest Problem: In this method DNA is partially registered with restriction enzyme thus generating a collection of DNA fragments between every cutting sites. After this a labelled probe that attaches to the DNA between two cutting sites is hybridized to partially digested DNA, and the size of fragments to which the probe hybridized are measured. In contract to the Partial Digest Problem where the input consist of all partial fragments the input for the PPDP consists of all partial fragments that contain a given point (This point corresponds to the position of the labeled probe) The problem is to reconstruct the position of the sites from the multiset of measured lengths. In the PPDP we assume that the labeled probe is hybridized a position 0 and that A is the set of restriction sites with negative coordinates while B is the set of restriction sites with positive coordinates. The probed partial digest From the book, so problem statement should be clear.

Answer:

Programming Problem. Proteins are sequence of aminoacids. Not only all consecutive aminoacids are interacting with each other but also non-consecutive aminoacids might be interacting with each other. Suppose that we are given all pairs of aminoacids that interact with each other in a spesific protein. We

are also given list of patterns which are chains of interacting aminoacids. Since these patterns give us clues about the protein function, we want to find all patterns that exists in a protein. Design and implement anefficient algorithm for this problem. Input file 1:(sample file) Aminoacid sequence followed by 1-based position of interacting pairs of aminoacids. i'th line starts with number i-1 denoting aminocid i-1. Other numbers in the same line denotes the positions of aminoacids that this aminoacid (i-1) interacts with. Input file 2(sample file): List of patterns. One pattern in each line Output: List of patterns that exist in each protein. One pattern in each line Visual representation of the problem:



Solution:
1. construct a graph using input1
2. Start from first aminoacid/node in the graph and walk through the graph in a depth-first fashion. At each step the path you have constitutes a word. Check if this word is in the dictionary of patterns (input2) (see note for performance issues)
3. repeat 2 by starting from other nodes
Note: In order to check if a word is in a dictionary a search tree or trie can be used. This way search is pruned if prefix doesn't exist in the dictionary.