Fall 2013 **CMSC 423: MidTerm1** H. Corrada Bravo
Time: 1 Hour, 15 Minutes

WAIT FOR INSTRUCTIONS BEFORE BEGINNING

HONOR PLEDGE: "I pledge on my honor that I have not given or received any unauthorized assistance on this examination."

Signature and UID: _____

Print name: _____

- ***Write your answers with enough detail about your approach and concepts used, so that the grader will be able to understand it easily.***
- ***The sum of the grades is 105, but your grades would be out of 100 (thus you get 5 bonus points by solving all the problems).***
- ***Select the best choice for the first 10 problems and mark it by X in the table below.***

Problem	1	2	3	4	5	6	7	8	9	10
A	X				X		X			
B			X						X	
C		X				X				
D								X		X
E				X						

DO NOT WRITE BELOW THIS LINE

Problems 1-10:		Problem 17:	
Problem 11-15:		Problem 18:	
Problem 16:		**Total:**	

Multiple-choice Problems (Answer THE BEST CHOICE in the Table of the First Page and NOT HERE):

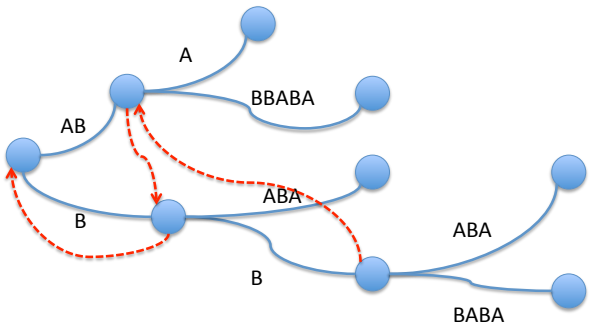
Problem 1. **(5 points)** Assuming Z-values $Z_2, \ldots, Z_8$ are already computed, how many character comparisons are required to compute Z_9 for string S=ACCACTACCAG by the linear time Z-algorithm discussed in class?

a) 0 b) 3 c) 1 d) 9 e) 4

Problem 2. **(3 points)** In which of the following situations would you use an algorithm like KMP or Boyer-Moore instead of a suffix tree?

a) There is no situation in which a suffix tree is not preferred
b) Matching many short patterns to a single long target
c) Matching many short patterns to many long distinct targets
d) (b) and (c) e) None of the above

Problem 3. **(5 points)** Which nodes in the following suffix tree for string S=ABBBABA have their suffix links incorrectly specified?



a) AB b) BB c) B
d) (a) and (c) e) None of the above

Problem 4. **(2 points)** Which of these best represents the relationship between genotype and phenotype?

a) there is no relationship between genotype and phenotype
b) an individual's phenotype completely determines their genotype
c) an individual's genotype completely determines their phenotype
d) an individual's phenotype partially determines their genotype
e) none of the above

Problem 5. **(5 points)** How many nodes in the keyword tree for set D={DRUG,RUM,DRUM} would have failure links *not* pointing to the root of the tree?

a) 3 b) 8 c) 2
d) 0 e) 4

Problem 6. **(2 points)** Which of the following terms is *not* associated with epigenetic mechanisms of gene regulation?

a) Chromatin conformation b) DNA Methylation c) Single Nucleotide Polymorphism (SNP)
d) Histone Modification e) None of the above

Problem 7. **(2 points)** Which of the following resources *does not* contain high-throughput sequencing data from population experiments:

a) KEGG Database b) 1000 genomes project

c) Short Read Archive d) (a) and (b) e) all of the above

Problem 8. **(2 points)** What is an *open reading frame* (ORF)?

a) any translatable sequence of nucleotides

b) any sequence of codons

c) a long enough sequence of aminoacids

d) a long enough sequence of codons without an intervening stop codon

e) None of the above

Problem 9. **(2 points)** Which of the following terms is *not* associated with chromatin.

a) nucleosome b) tRNA c) histone proteins

d) (a) and (b) e) None of the above

Problem 10. **(2 points)** Which of the following are examples of sequence mutations?

a) Single Nucleotide Polymorphism (SNP) b) insertion c) inversion

d) (a) and (b) e) None of the above

Short Questions (show all derivations as appropriate for full credit):

Problem 11. (6 points) (You can refer to the genetic code figure below). Consider nucleotide sequence S=…CGCATATGAACAAGA...

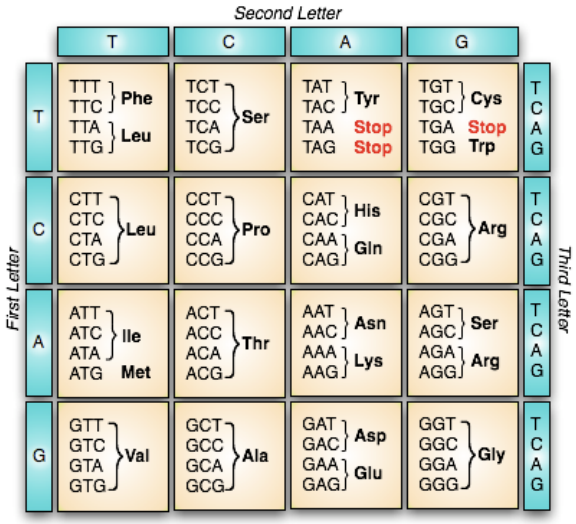
(a) How many completely *open reading frames* (ORFs) are there in this sequence (only considering forward strand)

(b) Write down the aminoacid sequence resulting from translation of one ORF (specify which).

(c) Specify a *synonymous* nucleotide substitution in this ORF, i.e., does not change aminoacid sequence.

(d) Specify a *non-synonymous* nucleotide substitution in this ORF.

(e) Specify a substitution that closes this ORF; write down the resulting aminoacid sequence.



(a) 2, …CGC ATA TGA ACA AGA has a stop codon
(b) …C GCA TAT GAA CAA GA… Ala Tyr Glu Gln
(c) …C GCA TA**C** GAA CAA GA…
(d) …C GCA TAT GA**C** CAA GA… Ala Tyr Asp Gln
(e) …C GCA TA**A** GAA CAA GA… Ala stp

Problem 12. (6 points) Provide a definition of *reproducible data analysis.* Discuss its importance in experimental computational biology. Mention computational tools that can help ensure data analyses are reproducible.

The ability of an independent analyst to obtain the same results starting from the same raw data as a published data analysis. Use of existing libraries, scripting languages (instead of excel), literate programming.

Problem 13. (6 points) Consider pattern P=AGTCGA and target T= AGCAGTCGAGTC. Show how the *Z*-algorithm would be used to find *all* occurrences of P in T in linear time. Calculate all *Z*-scores required. List those positions in T that need any explicit character comparisons to compute their *Z*-scores.

```
                AGTCGA$AGCAGTCGAGTC
Z               x000010200600004000
# comparisons   x111110201600003000
```

Problem 14. (6 points) Consider pattern P= ATGCATG and target T= ATGCATATGCATGCT. (1) Calculate all spm_i values for P; (2) Trace *all* steps used by the KMP algorithm to find all occurrences of pattern P in target T.

```
P: ATGCATG
   0000123
scan
t
ATGCATATGCATGCT
ATGCATG

      t
ATGCATATGCATGCT
ATGCATG
shift
      t
ATGCATATGCATGCT
    ATGCATG
scan
      t
ATGCATATGCATGCT
    ATGCATG
shift
      t
ATGCATATGCATGCT
      ATGCATG
scan
            t
ATGCATATGCATGCT
      ATGCATG
```

Problem 15. (6 points). We showed in class that the worst-case space complexity of suffix *tries* is $O(n^2)$, where *n* is the size of the string. Discuss the two tricks used to turn a suffix trie to a suffix tree so that only linear space is used in the worst case. Sketch a proof that only linear space is required.

Collapse all paths where no nodes split and label resulting edge with start and stop indices of substring. Each edge label takes O(1) space. Since every node now has at least a binary split, #nodes $= O(n)$.

Long Questions (you should always PROVE THE CORRECTNESS of your solutions)

Problem 16. (10 points) Prefix-Suffix matching. Give an algorithm that takes two strings α and β, and finds the longest suffix of α that matches a prefix of β. The algorithm should run in time O(|α| + |β|).

Construct string $S = \beta\$\alpha$ and compute Z-values for S. Identify positions *i* in α such that $Z_i = |S| - i + 1$ which corresponds to suffixes of α that matches a prefix of β. Find the smallest *i* satisfying this condition.

Problem 17 (15 points) Use a suffix trie to solve the following problem in linear time: Find the longest common substring of strings S and T. Show that your algorithm runs in linear time.

Construct a suffix trie for string S. Now follow the path determined by string T in the suffix trie for S until a dead end is found. Record the depth of the dead end node and follow it's suffix link. Now continue following the corresponding path determined by T recursively recording the depth of any dead end node (including leaves). The deepest dead end node corresponds to the longest common substring.

The first path followed by this algorithm corresponds to finding the longest common substring between S and T[1,…]. After following the suffix link on the dead-end, the next path corresponds to finding the longest common substring between S and T[2,…]. Eventually all suffixes of T are considered by the algorithm. The depth of the node corresponds to the length of the common substring, therefore returning the deepest node returns the longest common substring.

Problem 18. (20 points) Suppose you are given a *dictionary D* of strings $s_1, \ldots, s_k$. Define language L as the set of strings that can be generated by concatenating two or more strings in D. We are interested in determining if D is a *minimal*, that is, no subset of strings $s_1, \ldots, s_k$ completely generate language L. You can use keyword trees to address this question

(1). Write an algorithm using a keyword tree to determine if dictionary *D* is minimal. *Hint: use a keyword tree to determine if a string in D can be generated by concatenating two or more other strings in D.*

(2). Write an algorithm using keyword trees to find a *minimal* subset of *D.*

(3). (Extra Credit 5 points) Answer the two questions above using linear time algorithms. You can assume there exists a linear time algorithm to construct a keyword tree. You are free to extend the data structure as long as your extensions can be constructed along with a failure link.

Consider example {auto,tune,autotune}, where the third string can be removed. First of all, notice that in a keyword tree some of the interior nodes correspond to a string in the dictionary (we'll call these output node). As we construct the tree we annotate each node as output node or note if they correspond to a string in the dictionary. Let v->nv be the failure link for leaf v in the keyword tree, labeled L(v) in D. By definition, L(v)=alpha L(nv) for prefix alpha, then L(v) can be removed from the dictionary if alpha and L(nv) are output nodes. We can follow the failure link to check if L(nv) is an output node. We also know alpha corresponds to a node u in the path L(v) exactly |L(nv)| nodes up from v. We can recursively use the algorithm to check if alpha can be constructed from shorter strings in the dictionary. So, to find if D is minimal, traverse the keyword tree to find an output node v that satisfies this condition.

BONUS: to make it linear, we need to store on each node if it can be created from concatenating shorter strings in D. You can show that failure links strictly point to shorter strings in the keyword tree, and that this information can be propagated once the failure link is found.