ASSIGNMENT 1: STRING MAPPING

Goal: The goal of this assignment is to take a complex new problem and formulate and solve it as search. Formulation as search is an integral skill of AI that will come in handy whenever you are faced with a new problem. Heuristic search will allow you to find optimal solutions. Local search may not find the optimal solution, but is usually able to find good solutions for really large problems. So, both are very useful to learn about.

Scenario: You are a Genetics researcher working on phylogeny data – you have gene sequences of various organisms. You want to prove that some organisms are more related than others. You decide to map the strings onto each other and calculate mapping scores between each pair of organisms. The organisms with lower map scores probably are more related. Similarly, later you may have multiple organisms and you want to prove that they are all cumulatively related. The computational task is to compute an overall mapping score for a group of strings.

Problem Statement: There are K strings X_i from the vocabulary V. Each string X_i has length N_i. Your goal is to map the strings to each other. An easy way to do this is to think of this in two steps – conversion and matching. Conversion is a function F that takes in a string and returns another string. All $F(X_i)$s have the same length N. N is greater than equal to all N_is. The function F is only allowed to make one change to the original string – it can introduce dashes. It can introduce any number of dashes at any position. The conversion cost of X to F(X) is CC*number of dashes, CC being a constant. Once all strings have been converted the matching step just matches characters at each position. The matching cost between two characters is given by a symmetric function MC(c1, c2) where c1 and c2 are two characters ε V U {-}. Matching cost of two strings is the sum of matching costs of their conversions at each position. Finally, the matching cost of K strings is the sum of pairwise matching costs between each pair.

Example: Let K =3. Let vocabulary V= {A,C,T,G}. Suppose the three strings X_is are:

X_1: ACTGTGA

X_2: TACTGC

X_3: ACTGA

So, for this problem N_1, N_2, N_3 are 7, 6 and 5 respectively. Let all costs be as follows: CC = 3, MC(x,y) = 2 if x, y ε V and x != y; MC(x, -) = 1; MC(x, x) = 0. We may define our conversions as follows:

$F(X_1)$: -ACTGTGA

$F(X_2)$: TACT--GC

$F(X_3)$: -ACTG--A

With these conversions N = 8. The conversion costs are respectively 3, 6, and 9. The matching cost between $F(X_1)$ and $F(X_2)$ is 1+0+0+0+1+1+0+2 = 5. Similarly between $F(X_2)$ and $F(X_3)$ is 1+0+0+0+1+1+1+2=6 and between $F(X_1)$ and $F(X_3)$ is 0+0+0+0+0+1+1+0=2. Hence the total matching cost of this conversion is 5+6+2=13.

The final output cost of this mapping problem is sum of conversion and matching costs = 3+6+9+13=31.

Your goal is to find a conversion with the lowest final cost. (The current solution is not the optimal solution for this problem).

Algorithm 1: Implement an optimal algorithm for this problem. You can choose any algorithm of your choice such as uniform cost search, A* (with your admissible heuristic function), IDA*, depth first search branch and bound, etc. If you are implementing non-dfs solutions, do pay attention to duplicate detection or you may exhaust memory soon. Submit your best algorithm.

Algorithm 2: Implement a non-optimal but highly scalable algorithm. Our recommendation is to try a local search algorithm of your choice. Define a state space, neighborhood function, and starting state. Definitely try greedy hill climbing with random restarts. You may try other methods to improve further. Submit your best algorithm.

Input format:

Time (in mins)

|V|

V

K

X1

X2

…

CC

MC

#

Here is the input format for the example

5.5

4

A, C, T, G

3

ACTGTGA

TACTGC

ACTGA

3

0 2 2 2 1

2 0 2 2 1

2 2 0 2 1

2 2 2 0 1

1 1 1 1 0

#

Here is the format of MC is that it is representing a matrix of |V|+1 rows and columns. The last row and column is for dash. All diagonal entries will be zero. Any other entry represents the MC between the appropriate characters in V (based on the order they appear in V in the input line number 3). # represents end of the file.

For this problem you are given 5.5 mins of wallclock time to solve the problem.

Output format:

F(X1)

F(X2)

…

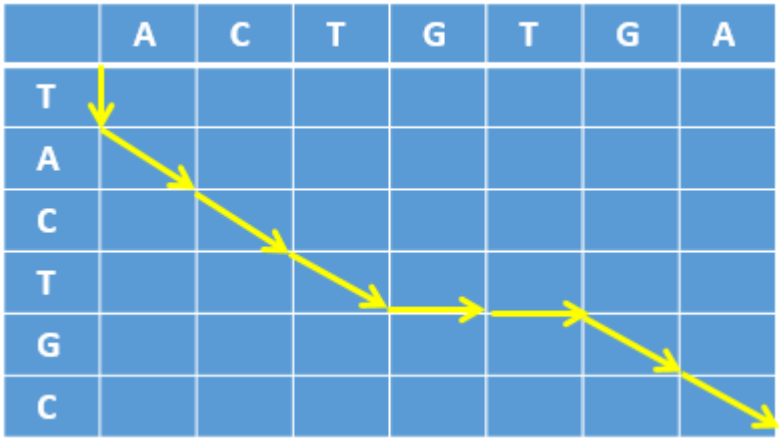
In our example

-ACTGTGA

TACT--GC

-ACTG—A

Hint (State Space Formulation): As a hint (you may of course choose to ignore it), here is a way to formulate the pairwise mapping problem as a shortest path problem in a matrix where the row is one string and the column is the other. For example, ACTGTGA and TACTGC are mapped in our running example using the following path below:



One can similarly generalize the path to K dimensions for K string mappings.

Hint (Dynamic Programming): An astute reader may observe that there is a dynamic programming formulation to this problem. We won't stop you from implementing that, however, dynamic programming does not scale very well for bigger K. Heuristic search or branch & bound might do much better for K=5 or more. We will definitely test on problems with large enough K to dissuade you from submitting a dynamic programming solution (the goal of the course is AI techniques). That said, dynamic programming might be a subcomponent of the final good solution.

Code: Your code must compile and run on a **machine named 'todi' or any machine with similar configuration present in GCL.** Please supply a compile.sh script for compilation. Also supply two shell scripts runopt.sh and runsubopt.sh. Executing the command ./runopt.sh inputfile outputfile should run your code taking in the input from inputfile and outputting the optimal solution to outputfile. Similarly for ./runsubopt.sh. Your codes must finish in the allocated time or else you will not get credit.

What is being provided? We are providing you with a format checker. The format checker outputs "False" if the output format is not according to the specifications: the output strings are not of the same size, or have out-of-vocabulary characters, or do not match with corresponding input strings. If the format is correct, it also gives the cost of the output.

There can be multiple input examples right after each other in the input file; with their corresponding outputs appended in the output file. The code returns the cost of each example in this case, and returns False if any of the outputs is not according to the format.

Usage: format_checker.py input.txt output.txt

Code verification before submission: Your submission will be auto-graded. This means that it is absolutely essential to make sure that your code follows the input/output specifications of the assignment. Failure to follow **any** instruction will incur significant penalty.

From this year onwards, we are running a pilot project where we shall be generating a log report for every submission within 12 hours of submission. This log will let you know if your submission followed the assignment instructions (format checker, scripts for compilation & execution, file naming conventions etc.). Hence, you will get an opportunity to resubmit the assignment within half a day of making an inappropriate submission. However, please note that the late penalty as specified on the course web page will still apply for resubmissions beyond the due date. Exact details of log report generation will be notified on Piazza soon.

*Also, note that the log report is an **additional** utility in an experimental stage. In case the log report is not generated, or the sample cases fail to check for some other specification of the assignment, appropriate penalty for not adhering to the input/output specifications of the assignment will still apply at the time of evaluation on real test cases.*

What to submit?

1. Submit your code **in one file named in the format ENTRYNUMBER.zip or ENTRYNUMBER1_ENTRYNUMBER2.zip based on whether you have one person or two people in the team.**
2. Submit at-most 1 page writeup (10 pt font) called Writeup.pdf (should be present in the main .zip directory) mentioning your names, and describing your choices and rationale for Algorithm 1 and Algorithm 2. This is not graded but failure to submit a satisfactory writeup will incur negative penalty of 20% of total score. Your writeup will help us identify any common misconceptions and particularly good ideas for discussion in the class. Moreover, your writeup should mention the names of all the students you discussed the assignment with (see Freedom of Information rule on course webpage).

Evaluation Criteria

1. Final competitions on a set of similar benchmark problems. The points awarded will be your normalized performance relative to other groups in the class. Both optimal and suboptimal solutions will be tested separately and will carry equal weight.
2. **If your code1 outputs a suboptimal solution for any benchmark problem (identified in case another code produces a better solution than yours), you will lose ALL points for the optimal solution.**
3. Extra credit may be awarded to standout performers.

What is allowed? What is not?

1. You may work in teams of two or by yourself. If you work in a team of two then make sure you mention the team details in the write-up.
2. You are required to work in C++ for fair comparison amongst all teams.
3. Your code must be your own. You are not to take guidance from any general purpose AI code or problem specific code meant to solve this or related problem.
4. It is preferable to develop your algorithm using your own efforts. However, we will not stop you from google searching.
5. You must not discuss this assignment with anyone outside the class. **Make sure you mention the names in your write-up in case you discuss with anyone from within the class outside your team.** Please read academic integrity guidelines on the course home page and follow them carefully.
6. Your submitted code will be automatically evaluated against another set of benchmark problems. You get a heavy penalty if your output is not automatically parsable.
7. We will run plagiarism detection software. Any team found guilty will be awarded a tough penalty as per IIT rules.