CSCI 561 – Spring 2005 – Homework #2 – Due 03/01/2005

Question 1 (20%):

Search algorithms are commonly evaluated according to the following four criteria: completeness, time complexity, space complexity and optimality.

1) Evaluate three heuristic algorithms: Genetic Algorithm, Simulated Annealing, and Hill Climbing, by the above four criteria. In some cases, there may be no proper answer, in which case you should explain why.

2) Compare these heuristic algorithms with previous search algorithms (BFS, DFS, Uniform-cost) in terms of evaluation function.

3) Which is most proper algorithm for the N-Queen problem? (Explain why)

Question 2 (50% code + 30% experiments):

Let’s implement a Crossword Solver using simulated annealing. You are given a crossword grid and a list of words (dictionary), and your task is to find a way to completely fill the grid with words, without leaving any blank space in it. You will implement a program that achieves this using simulated annealing.

Example:

Given this grid…
and given a list of words, like:

ARTIFICIAL
INTELLIGENCE
COOL
SIMULATED
ANNEALING
ROCKS
ILOVETHISHOMEWORK
FINALLYSOMECHALLENGE
561BESTCLASSEVER
ABOARD
ALASKAN
ALDO
ALLAYING
AREARUG
ATE
AVISOS
…[[more words not listed here]]

you should find a way to fill the grid with the words given to you, for example obtaining:

![Crossword Grid](image)

You have to adhere to the standard crossword rules:

- words must be readable either from left to right or from top to bottom (so placing a word COOL in the grid as LOOC is incorrect);
- You cannot put several short words in a long row of blank squares (so writing COOLCOOL in a row of 8 eight squares is a mistake; you need to find a single 8-letter word from your list of given words to write in that row instead);
- Words in the vocabulary list could be any alphanumeric string of our choice, always all-caps. If we provide goofy vocabulary “words” like “ILOVETHISHOMEWORK” shown in the example list above, then it will be our responsibility to not provide components like I, LOVE, THIS, and HOMEWORK, so that there is no possible ambiguity as to whether you used the single word ILOVETHISHOMEWORK or its four components in the grid;
- Expect vocabulary lists to be long, typically several hundred or several thousand words. There will be many words in the lists that you may never use;
- You cannot start a word just anywhere you like in the grid! In the example above, the 105 legal start locations are indicated by small numbers in the corners of the corresponding blank squares. HINT: one of your first tasks will be to figure out how these squares are special.

**Input:** You will be given a text file called `input.txt` in the current directory. The first line contains two integers w and h specifying the width and height of the grid. Then the blank grid will be given, with a dot (.) for each blank square and a hash (#) for each black forbidden square. The next line will contain the size N of the list of words you can use, followed by N lines each containing a word. So, for the above example, you should expect an input.txt as follows:

```
19 19
......#. ...............
        .#       .......
        ..      .####....#
        .       #.####.##
        .       #      #
        .#  #    .#    #
        .## # .#. # .#
        .#  .#### #.  .#
        .#  .#    .#    .#
        .      #      #
        .      #      #
        .#  #    .#    #
        .## # .#. # ..#
        .      #      #
        .#  #    .#    #
        .## # .#. # .#
        .#  .#### #.  .#
        .      #      #
        .      #      #
        .#  #    .#    #
        .## # .#. # .#
        .      #      #
        .      #      #
        .#  #    .#    #
        .## # .#. # .#
        .#  .#### #.  .#
        .      #      #
        .      #      #
        .#  #    .#    #
        .## # .#. # .#
        .      #      #
        .      #      #
        .#  #    .#    #
        .## # .#. # .#
```

```
127
ARTIFICIAL
INTELLIGENCE
COOL
SIMULATED
ANNEALING
ROCKS
ILOVETHISHOMEWORK
```
FINALLY SOME CHALLENGE
561 BEST CLASS EVER
ABOARD
ALASKAN
[[and so on… 127 words in the list in total.]]

Have a look at the sample text files provided with this HW for the complete files.

Output: All output should be written to a file called output.txt in the current directory. Your output.txt should start with a single word ‘yes’ or ‘no’ indicating whether you found a solution. Then show us your solution, by writing out the entire grid with all the (.) in the original replaced by some of the words from the list. For example:

yes
MAITAI#NAMESAKE#ZAG
ABOARD#OVERPAID#ICE
CALIFORNIA GIRLS#LING
ETAL#TIFS#EEL#ALDO
###CHUGA#WAG###SLIDE
FAROES#THE BEACH BOYS
AREARUG#IDOLISE#NEE
IGETAROUND###DART###
SID###PARTIBLE#PORTS
AVISOS#GENRE#SAVEUP
LETON#REDGIANT###NEE
###TRIO###BARBARA ANN
TRA#ALASKAN#CRUSTED
HELP MER HONDA#VITALS
ESTOP##EDD#STENO###
STEW#ILL###ESA## UTAH
WAR#WOULDN'T IT B ENICE
ATE#ALLAYING#ELDERS
NED#SAUCEPAN#TOSSES

In addition to providing a nice, clean, and working piece of code, please answer the following questions:

a) [5%] Why is simulated annealing a good algorithm for this problem? (think about complexity and other considerations)

b) [10%] What would be reasonable temperature decrease schedules to use for this problem? To answer this question you will need to do some search on the web to find which schedules other people have typically been using for various problems solved using simulated annealing. Tell us about at least three different possible schedules.

c) [10%] Experiment with the different temperature schedules you found and conduct a comparative analysis, running your algorithm on at least 25 different problems of various sizes each time (that is, create 25 input.txt problems of your choice, e.g., by downloading them from the web; then run your program on those input.txt samples but each time using a different temperature decrease schedule. It
is okay if your program has to be recompiled to select which schedule to use. When you send us your code, just have the best schedule as your default). Motivate your answer with a table showing some measure of performance on your 25 test problems (for example, execution time) for your different schedules. Like in b), you should test at least 3 different schedules.

d) [5%] Could you have solved the problem using BFS? Up to roughly what size would BFS be reasonably expected to be able to find a solution?

CAUTION: please follow these simple rules
- do not expect any command-line argument.
- make sure your code compiles on aludra.usc.edu using g++
- provide a Makefile or indicate in a README how to compile your code.
- your program should read one text file named “input.txt” which will contain a problem definition, and should write one text file named “output.txt” which will contain your solution.
- Assume unix text files (single LF at end of line, no CR).