COMP 610 Project Description

MAX2SAT is a computational problem with an input of k boolean clauses each consisting of 2 literals or-ed together (eg $x_3 \vee \overline{x_5}$). The number of variables used will be called n. The goal is to find a truth assignment which maximizes the number of clauses that can be simultaneously satisfied.

EXAMPLE: I = $\{(x_1 \vee x_2), (x_1 \vee \overline{x_2}), (\overline{x_1} \vee x_2), (\overline{x_1} \vee \overline{x_2})\}$
The solution for instance I is $x_1 = True, x_2 = True$ which satisfies 3 of the 4 clauses.

TASK: For your project you will write a program which solves any given instance of MAX2SAT. There is no known method to solve this problem in polynomial time. So solutions with exponential time performance in the worst case are acceptable. However, it is expected that you will put a serious effort into beating the obvious brute force solution. In other words, writing a program that just performs a brute force search for the solution is a good place to start, but should not be where you end. In other words, you will not get full credit for only attempting a straightforward brute force search. Exactly what techniques you use are up to you. EG you might find a decent, but non-optimal solution using a greedy heuristic, improve it using local search, and then perform a branch and bound search.

Your program should read the instance from a file called instance.txt which is formatted as 2 integers per line seperated by whitespace. Each line corresponds to a single clause. The absolute values of the variables are the two variables in the clause. If the first/second integer is negative it indicates that the first/second variable is negated. Using the example above the file instance.txt would contain

```
1 2
1 -2
-1 2
-1 -2
```

The last line outputted by your program must be formatted as a string with lenght n where every character being either T or F, a space, and then an integer. The string of T/F should give the truth assignment and the integer should be the number of clauses satisfied. Again using the example the output should be

```
TT 3
```

You may want your program to output other information prior to the final solution (perhaps printing a line every time a better solution is found or giving an update as to how much of the search space has been completed.

I would prefer that you utilize Java for your programming language. However, C++ is permitted if it can be compiled under gcc or g++ on a standard mac running OSX or Debian linux.

SUBMISSION: This section may be updated to address questions from the class.

You will submit:

- Your program file(s)
- A pdf document that describes the ideas/algorithms you used (especially anything interesting you did to speed up finding the solution).
- An ascii file containing directions about how to use your program. In many (most?) cases this will be trivial (eg compile with javac *.java and run with java Driver), but if using C++ often certain flags will need to be set. Additionally. there is often a student who manages to create something bizarre (eg save all files in a directory called .!woohõo,,, compile by using the command make -qRsT fingerSandwitches, repeat the compilation process, open the resulting file bingoDingo.js in your browser, click on the prompts as appropriate). This is discouraged.

To get a poor grade (approx 50%) you just need to have your program work (eg you could use a very naive brute force algorithm). For full credit, I'll need to see smarter things attempted.

CHALLENGES: Depending upon time and interest, I may post particular instances in moodle. The person who submits the best valid solution (ties being broken by earliest posted time) will receive a point toward their project. While you cannot exceed 100%, this would allow a non-perfect score to be raised.