First GA Assignment

Assignment 3 CS472-S15 Due: Tue Mar 3, 2015 at 5pm PT

Points: 150

Reference Files (these should be live links):

- rand.cpp A simple fast random number generator
- rand.h

1 The Problem

Write a genetic algorithm to "solve" a k-satisfiability (k-sat) problem. That is, given a logical expression in n Boolean variables expressed in conjunctive normal form with c clauses and at most k distinct variables in each clause, find an assignment of the variables that makes the expression true. The problem may or may not be satisfiable, that is, you may or may not be able to find an assignment to the variables that makes the expression true. All logical expressions can be expressed in conjunctive normal form. Here is an example of a general problem in conjunctive normal form with $n = 4,\ k = 3,\ c = 3$:

$$(A \vee \overline{B} \vee C) \wedge (B \vee \overline{C} \vee D) \wedge (\overline{A} \vee \overline{C} \vee D)$$

This is satisfiable with $A = true, B = true$, and $D = true$.

$$(\overline{A} \vee \overline{B} \vee C) \wedge (A \vee B \vee \overline{C})$$

is not satisfiable for any set of values given to the variables. The k-sat problem for $k \geq 3$ is NP.

2 The Algorithm

Use a generational genetic algorithm using the parameters in Table 1 to solve this problem. The input to the program will be three integers being n, k, and c in that order. Then two n character binary strings per clause. For the general problem above it is:

```
4 3 3
1110 0100
0111 0010
1011 1010
```

The program will be invoked with three arguments passed as described in Table 1. You need only handle up to 64 variables, but you must handle any number of clauses and any $k\ :\ 0 < k \leq n$

Table 1: An extended table of parameters for the genetic algorithm. This includes command line parameters.

Type of algorithm	Generational GA
Num of generations	1^{st} argument on command line
Population size	2^{nd} argument on command line
Num elite	2
Selection for mating	Tournament size 3
Xover probability	3^{th} argument on command line
Mutation probability	1/16 per bit

The output from the program should be an integer followed by an n-bit binary number as a character string. The binary is the assignment of the variables to make the given expression true. The integer is the number of clauses that are satisfied. Then on the next line it should print either `Satisfiable` or `Could not find a solution`.

Your program should be called `ksat` and be built using a makefile in the usual way.

3 Submission

Homework will be submitted as an uncompressed tar file to the homework submission page linked from the main class page. You can submit as many times as you like. The LAST file you submit BEFORE the deadline will be the one graded. For all submissions you will receive email giving you some automated feedback on the unpacking and compiling and running of code and possibly some other things that can be autotested. I will read the results of the runs and the reports you submit. Have fun.