15-398, Fall 2002
Assignment 2

Due October 7, 2002

1 Coloring a Graph with k-colors

The goal of this homework is to gain familiarity with SAT. We will encode an interesting graph problem into a SAT problem and use modern SAT solvers like GRASP and Chaff to solve them.

The k-coloring problem is defined as follows:

> Given an undirected graph $G(V, E)$ and a natural number k, is there an assignment $color : V \to \{1, 2, \ldots, k\}$ such that $\forall(u, v) \in E.color(u) \neq color(v)$? Informally, is it possible to assign one of k colors to each node such that no two adjacent nodes are assigned the same color? If the answer is positive, we say that the graph is k-colorable.

As an aside, the map coloring problem, where you have to color each pair of adjacent countries with separate colors, is just a special instance of this problem. All the maps that you can draw on a piece of paper make a *planar* graph. The famous four-color theorem says that all planar graphs can be colored with four colors.

With regards to this problem, answer the following questions.

1. Show a formulation of the k-coloring problem as a satisfiability problem, i.e., for a given graph $G(V, E)$ and k, derive a CNF formula φ such that φ is satisfiable if and only if G is k-colorable. How many CNF variables and clauses does your CNF formula have in terms of the number of nodes $|V|$ and the number of edges $|E|$?

2. Write a program that accepts a textual representation of a graph $G(V, E)$ and the value k, and converts it to a CNF formula. The

formats of the input files, CNF files and output files are described below.

3. Download from the course web page the 5 sample graphs. For each one of them, generate a SAT problem and check whether it is satisfiable with the help of a modern SAT solver (e.g. GRASP, Chaff, etc.). If the formula is satisfiable, write a table showing the color assignment in a text file, whose format is described ahead. **Beware:** SAT solvers might take a long time to run, and in some cases, blow out of memory. If your SAT solver doesn't come up with a satisfying assignment within 3 hours, stop the process and report it as a time out.

Input File Format

The format of the input file describing k-colorability problem should be as follows. Each line should begin with a character that signifies the content of the line.

- **Comment line.** Each comment line begins with the letter `c`. They are to be ignored.
- **Problem line.** There is only one problem line in the file, which occurs in the beginning of the file. Problem lines begin with the letter `p`. They follow the following format.

  ```
  p <# of colors, k> <# of nodes> <# of edges>
  ```

  Subsequently, the nodes of the graphs are identified by the numbers $1, 2, \ldots, |V|$ and the edges are identified by the numbers $1, 2, \ldots, |E|$.
- **Edge descriptor line.** These lines begin with the letter `e` and describe an edge (u, v) as follows:

  ```
  e <# vertex u> <# vertex v>
  ```

This format is very similar to the standard DIMACS format, except for a slightly different `'p'` line. For the sample graph of the Figure 1, the file looks like as follows:

```
c This is a 3-colorable test graph.
p 3 6 8
e 1 2
e 2 3
e 3 1
e 1 4
e 4 5
e 5 6
e 5 3
e 6 1
```

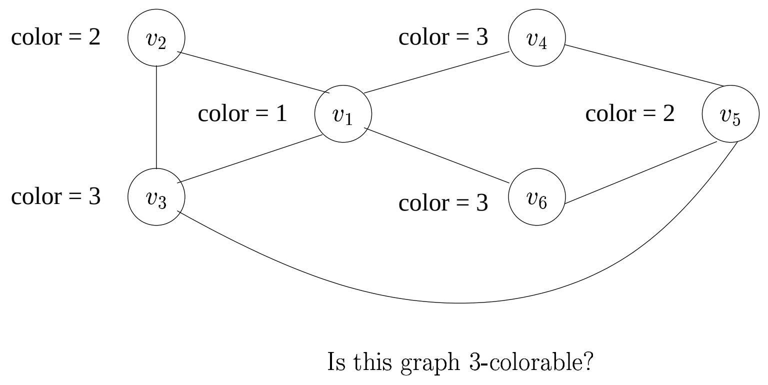

Figure 1: A sample graph with a 3-coloring.

CNF File Format

Please follow this format for CNF files, as most SAT solvers expect the CNF formula in this format.

```
p cnf <# vars> <# clauses>
<list of clauses>
```

Each clause is described by a list of numbers ending with a 0. Each number represents a variable and the variables are numbered 1, 2, ...A negative number means that the variable is negated in that particular clause. Here is an example of a CNF formula and the corresponding CNF file.

$$(x_1 \vee x_2 \vee \neg x_3) \wedge (x_2 \vee \neg x_1) \wedge (x_4 \vee \neg x_2 \vee \neg x_1)$$

```
p cnf 4 3
1 2 -3 0
2 -1 0
4 -2 -1 0
```

Output File Format

The output of the program should simply be a list of node-value pairs if the answer is yes. Each line should contain two integers, the first one being the node number and the second one being the color number. If the graph is not k-colorable, the first line of the file should be -1. For our graph, the output should be as follows.

```
1 1
2 2
3 3
4 3
6 3
5 2
```

Note that our graph is not 2-colorable.

Submission

Statically linked binaries of the GRASP sat solver for Linux and Solaris are available at

`http://www.cs.cmu.edu/afs/cs/usr/emc/www/15-398/resources/GRASP/sat-grasp.st.{linux,solaris}`.

A local copy of the man page for GRASP is located at

`http://www.cs.cmu.edu/afs/cs/usr/emc/www/15-398/resources/GRASP/grasp-man.txt`.

The easiest way to use GRASP is as follows.

```
sat-grasp.st.linux +V1 graph1.cnf
```

The `+V1` option will print out the satisfying assignment if there are any.

The graph files are located on the course web page at `http://www.cs.cmu.edu/afs/cs/usr/emc/www/15-398/assignments/graph{1-5}`. Submit the following items.

1. Source code of your program in your favorite language.
2. Binary of your program compiled for Linux. Also include the binary for any SAT solver other than GRASP that you might have used.
3. Instructions on how to compile and use your program.
4. For each graph instance, output of your program and the file containing CNF formula. For each graph file, the CNF file should have an extension `.cnf` and the solution file should have an extension `.sol`. For example, for `graph1`, the CNF file should be `graph1.cnf` and the solutions should be in the file `graph1.sol`.

Package these items in a compressed tarball and send them in an email to the TA before the deadline (the beginning of the class on 16th October). In your documentation, make a table showing for each graph, the # of CNF variables, # of clauses, running time of the SAT solver and any other statistic that you might find interesting.