Minesweeper and Propositional Logics

AR 2004: Lab session, February 6

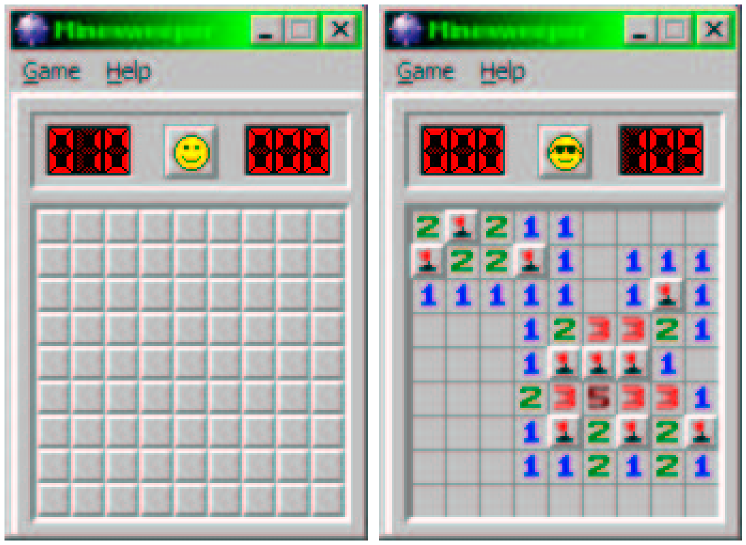
1 The task in this week's lab-session

Minesweeper is a popular computer game which comes for free both with Windows as well with Linux systems. Given a screen of $n \times n$ squares it can be checked automatically whether a configuration is consistent or not, i.e. whether a particular set of values for the number of mines surrounding a square can actually happen or not. This problem can be reduced to propositional satisfiability checking, more precisely there is a mapping from sets of number of mines surrounding a squares to propositional clause normal form.

To get familiar with propositional logic, clause normal form, satisfiability checking and the ZChaff tool, you should, in this lab, write a set of rules to transform an input file with number-of-mines values into CNF, which can the be fead into ZChaff. It is up to you to simple do the transformations by hand, or to write a set of scripts to do the job.

2 Minesweeper

The game screen consists of a rectangular field of squares. Each square can be cleared, or uncovered, by clicking on it. If a square that contained a mine is clicked upon, the game is over. If the square did not contain a mine, one of two things can happen. If a number appears, it means that that number of adjacent (including diagonally-adjacent) squares contain mines. If no number appears, then the game automatically clears those squares adjacent to the empty square (since they could not contain mines). The game is won when all squares that do not contain a mine are cleared.



The two pictures show what a Minesweeper game looks like at the beginning and at the end of a completed game.

3 The Minesweeper Decision Problem

The MD-problem: *Given a particular minesweeper configuration determine whether it is consistent, which is whether it can have arisen from some patterns of mines.*

This approach has been used as a strategy to solve a minesweeper problem: To determine whether a background square is free we can mark it with a mine. If the resulting configuration is inconsistent the square is safe to uncover. In this lab session, we are just interested in solving the MD-problem, however.

4 Solving the MD-Problem using SAT

The task for today's lab session is the following:

- Given a minesweeper configuration, i.e. a position P such as the following:

```
 __________
|2|_|_|_|3|
|_|_|_|_|_|
|_|1|_|3|_|
|_|1|_|_|_|
|_|_|_|2|1|
```

- Decide whether this situation is actually possible, i.e. whether there you can place mines in the free squares, so that the number of adjacent mines is correct. You should do this by transforming the configuration to a propositional formula in clause normal form.
- Hint: Use propositional variables M_{ij} to denote that there is a mine one square (i, j).
- Produce the formulas in Clause Normal Form solving the following two configurations:

```
 _____    _____
|3|_|2|  |3|_|2|
|_|_|_|  |_|_|_|
|_|1|_|  |_|2|_|
```

  both in propositional logic and in ZChaff input format, and check with ZChaff which of the two configurations is consistent, and which is not.

Remark and Question: It has been shown that solving the MD-Problem is an NP-complete problem. As we know that solving the SAT problem is NP-complete as well, does it mean that we have a theoretically optimal decision procedure for the MD-problem?

5 Writing Scripts

Ideally, you don't do the representations by hand, but write scripts for the transformations. In this case you can read in more difficult configurations, such as the one given above. A configuration is represented as a file, where each line contains the x-value, the y-value and the number of mines. Position P would, for example, be represented as follows:

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

To get you going I've put a template `ms2pl.pl` in Perl, of how to read in a file and get the variables etc. to the directory ∼`schlobac/linux/Minesweeper/`.

6 ZChaff

You can run ZChaff from your machine by typing the command ~`schlobac/linux/zchaff myfilename.cnf`. ZChaff input format was shown in yesterday's lecture, you have lines with comments starting with "c", relevant information about the number of clauses and variables in a line starting with "p", and then a list of clauses represented as lists of variables (positive numbers for positive literals, negative numbers for negated variables). Here's the example seen in the course for a formula $\Sigma = (P \vee \neg Q)$ & $(Q \vee R)$ & $(\neg R \vee \neg P)$ in CNF:

```
c a diplomatic problem
c created: 06/01/03
p cnf 3 3 (#variable #clauses)
1 -2 0
2 3 0
-1 -3 0
```

Note that the variables need to be numbered starting from 1, i.e. if you have 4 propositional variables, you have to use the names 1,2,3 and 4.

I will provide some test files for minesweeper configuration files in the directory ~`schlobac/linux/Minesweeper/` on `gene`.

Remark and Question: How many mines do you need to make the SAT reasoning maximally hard?

If you have questions, ask!

(and apologies for the flimsy pictures of the configurations).