# Automated Reasoning 2012 (KB)
# PROBLEMS 1 (Various Propositional Techniques)

**NOTE**: *The problem sheets are an important part of the course and will contain a variety of problem types including technical exercises for the methods discussed in class as well as more open-ended problems to encourage you to pursue further the ideas covered; occasionally problems require a simple proof. There will also be a few quick questions - very suitable for a short bus or tube journey!*

QUICKIES

(1, 2 and 3 are about Davis Putnam and 4 and 5 are about Model Generation and 6 is about OBDDs)

***1. In DP, if a branch runs out of clauses then the given clauses have a model - Yes or No?***

***2. How many pure literals are there in the set of clauses: $A \vee B \vee \neg D$, $D \vee B \vee A$, $C \vee E \vee \neg A$, $\neg E \vee F \vee \neg A$, $F \vee \neg C \vee E$?***

***3. In clauses of Q2 if the next atom chosen to split on is A, how many clauses are left in the A/¬A branches, after applying subsumption? Repeat, but for atom D.***

***4. In MG, if no given clause has a positive literal then the given clauses have a model - Yes or No?***

***5. Which of the following clauses can be selected if the branch already contains the atoms A, B, C:***
***$\neg B \vee D$, $D \vee E$, $\neg A \vee \neg C \vee, F$, $\neg D \vee \neg B$, $\neg C \vee \neg E \vee D$, G***

***6. What is the OBDD of the formula $\neg(\neg A \vee B) \vee B$? Use the OBDD to state an equivalent formula.***

LONGER QUESTIONS

1. (a) *Use the Davis Putnam (DP) procedure to show the unsatisfiability of*

(i) $\neg A \vee \neg B \vee C$, $D \vee E$, $\neg C \vee \neg E \vee F$, $D \vee \neg F$, $A$, $B$, $\neg D$ (i.e. 7 clauses)

(ii) $A \vee B \vee C$, $A \vee \neg B \vee C$, $\neg A \vee \neg B \vee \neg C$, $A \vee B \vee \neg C$,
$\neg A \vee B \vee \neg C$, $\neg A \vee \neg B \vee C$, $A \vee \neg B \vee \neg C$, $\neg A \vee B \vee C$

(b) Remove the last clause in part a(ii) and use DP to find a model for the remaining clauses.

2. (a) *Use the model generation (MG) procedure*:

(i) to show that there are no models for the clauses in Q1a(i) (which rule applies to a positive fact?)

(ii) to find a model for the clauses of Q1a(ii), first without the second clause and then (instead) without the first clause.

(b) Explain in general how, in the MG procedure, a model is found from a branch to which no steps are applicable (i.e. case 4 of the MG procedure).

3. The OBDD method can be used to show equivalences between formulas.

(a) Show $\neg(A \Leftrightarrow B)$ is equivalent to $\neg A \Leftrightarrow B$ by considering OBDD($\neg(A \Leftrightarrow B) \Leftrightarrow (\neg A \Leftrightarrow B)$).

(b) Show $A \Leftrightarrow (B \Leftrightarrow C) \models (A \Leftrightarrow B) \Leftrightarrow C$ by finding

$$\text{OBDD}( (A \Leftrightarrow (B \Leftrightarrow C)) \wedge \neg ((A \Leftrightarrow B) \Leftrightarrow C) )$$

Compare your answer with DP and/or MG for the clausal form of the problem, which is the clauses in Q1(aii). (Next week, you'll see a systematic way to convert to clauses.)

4. A top-down recursive algorithm to form an OBDD uses REDUCE from the slides and the following step, for any atom A; it terminates when all tests are atomic.

OBDD(W) = REDUCE(if(A, REDUCE(OBDD(W[A==t])), REDUCE(OBDD(W[A==f])))

Compare its use with the bottom-up algorithm.

5. Write a program that implements (one of) the various procedure for propositional clauses. ..... *PTO*

There are many systems for checking satisfiability and unsatisfiability (called Sat Solvers), most based on the Davis Putnam procedure. Often they make use of clever ideas for saving work and various provers have been entered into competition with one another at conferences such as CADE and more recently IJCAR. Two programs you can try out for fun are available on our system. They are called GRASP and ZCHAFF. You can have a look in Google for various papers related to these provers and some links I found useful are listed below. So that you don't have to make up problems Rob Craven (an RA in this department) has written a short perl script that generates data files. Of course, you can use your own files and/or generating program as well. The file format is very simple, and the given program could be improved by avoiding subsumed clauses and permutations of clauses and so on. It doesn't generate tautologies or clauses with duplicate literals, but that's all it avoids.

To use *zchaff* or *grasp* is very easy. Unfortunately, neither program comes with a good manual, but the suggested options seem to work for the examples I have tried.

The data file (let's call it test) should look something like:

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

where p means propositional (I guess) cnf means the data are clauses, 3 means there is a maximum of 3 atoms (called 1, 2, 3) and 4 means there are 4 clauses. The data lines each end in 0. If we use q, r, s for the 3 atoms, the first clause represents q ∨ ¬r ∨ ¬s. There's no need to suppose that the literals are in a particular order. If you put c at the beginning of a line it is treated as a comment line. Running

```
grasp +O +sa +S200 test
```

where you get output that gives you various models (even the inventor told me he couldn't remember what the parameters meant(!)). In particular: {¬q, ¬r} (meaning q is false and r is true), {¬q, r, ¬s} and {q, ¬r, s}, which do indeed cover the 4 models of these clauses.

If you run zchaff instead (again finding details of parameters is not easy) using

```
zchaff test
```

you get one model: {¬q, ¬r, ¬s}

Running

```
perl /vol/lab/satsolvers/makesat.pl -na 10 -nc 40 -minlc 3 -maxlc 7 > outputfile
```

will generate a file with between 3 and 7 literals per clause, using 10 atoms and having 40 clauses. There are several options for makesat which are listed if you type

```
perl /vol/lab/satsolvers/makesat.pl makesat.pl -h
```

Enjoy! In a few more weeks we'll look at a first order prover.
Some useful grasp/chaff related websites:
www.cs.cmu.edu/~mtschant/15414-f07/lectures/**grasp-ex.pdf**
www.cis.upenn.edu/~alur/CIS670/**malik.ppt**
www.cse.nd.edu/Reports/2005/TR-2005-04.pdf