**CEE 5290/COM S 5722/ORIE 5340**
**Heuristic Methods for Optimization Fall 2013**
# Satisfiability Testing

Let **E** be a Boolean expression consisting of n variables. A *literal* is either p or ~p, where p is any binary (Boolean) variable. A disjunction of literals (terms OR-ed together) is called a *claus*e. **E** is said to be in *conjunctive normal form* (CNF) if it can be expressed as a conjunction of clauses. The problem of determining if there exists a truth assignment of the n variables that causes a CNF expression **E** to evaluate to a "true" is referred to as the problem of **Propositional Satisfiability (SAT**). Many interesting combinatorial optimization problems, including ones involving scheduling and planning, temporal reasoning, learning, constraint satisfaction, VLSI circuit synthesis, circuit diagnosis, can be encoded for solution as CNF SAT problems. Due to this reason, finding good algorithms for Satisfiability testing is of enormous interest to researchers in Artificial Intelligence and other fields.

An exhaustive search procedure to solve the SAT problem is to try systematically all the ways to assign truth values to the n variables. The total number of such assignments is, of course, $2^n$; thus, this exhaustive search is clearly infeasible if n is even reasonably large. The SAT problem was the very first decision problem to be identified as NP-complete. A special case of SAT that is also NP-complete is 3-SAT, where each clause in the expression **E** consists of exactly 3 literals.

SAT, as explained above, is actually a decision problem -- it can be answered by a "yes" or a "no." It can also be expressed as an optimization problem - find the truth assignment that minimizes the number of unsatisfied clauses in the given expression. If the global minimum is 0, then the expression is satisfiable. Note that the problem of determining if an expression is unsatisfiable seems harder (since you may not know this until all $2^n$ assignments have been tested).

Consider the following example:
**E**= (~s1+~s3+s4)&(~s2+s3+s4)&(s1+s3+~s4)&(s1+~s3+s4)&(s1+s2+s4)&(~s1+~s3+~s4)

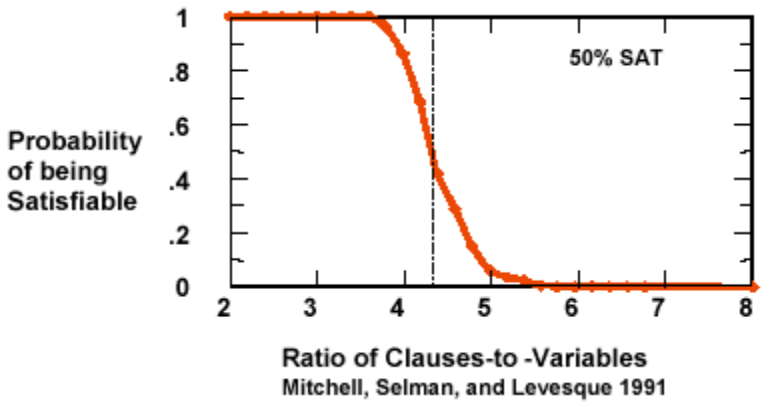
**E** is a 3-SAT CNF expression with 6 clauses and 4 variables. Let s = [s1, s2, s3, s4]. There are16 possible truth assignments to the variables, with the following cost values Cost(s):

| s1 | s2 | s3 | s4 | Cost(s) |
|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 1 |
| 0 | 0 | 0 | 1 | 1 |
| 0 | 0 | 1 | 0 | 1 |
| 0 | 0 | 1 | 1 | 0 |
| 0 | 1 | 0 | 0 | 1 |
| 0 | 1 | 0 | 1 | 1 |
| 0 | 1 | 1 | 0 | 0 |
| 0 | 1 | 1 | 1 | 0 |
| 1 | 0 | 0 | 0 | 0 |
| 1 | 0 | 0 | 1 | 0 |
| 1 | 0 | 1 | 0 | 1 |
| 1 | 0 | 1 | 1 | 1 |

| | | | | |
|---|---|---|---|---|
| 1 | 1 | 0 | 0 | 1 |
| 1 | 1 | 0 | 1 | 0 |
| 1 | 1 | 1 | 0 | 1 |
| 1 | 1 | 1 | 1 | 1 |
| 1 | 1 | 0 | 1 | 0 |
| 1 | 1 | 1 | 0 | 1 |
| 1 | 1 | 1 | 1 | 1 |

This expression is satisfiable since 6 of the 16 possible truth assignments make the expression true (i.e. have cost 0). As the number of variables *n* and the ratio of clauses to variables $\rho$ both increase, it turns out that the number of expressions that are satisfiable decrease. In fact for 3-SAT problems, when the $\rho$ crosses a threshold value of about 4.3, the probability that a randomly generated instance is satisfiable goes to zero sharply. Professor Selman along with other researchers has shown in recent years that this "phase transition" is analogous to the physical phenomenon whereby, for example, solids turn to liquids and liquids to gases. For more information on SAT problems and local search strategies for solving them, see Prof. Selman's homepage at http://www.cs.cornell.edu/selman

**For the project, you have been provided with a suite of ten 3-SAT instances from SATLIB, a repository of benchmark problems**. Each instance consists of 200 variables and 860 clauses (Note that $\rho = 4.3$). The names of the ascii files containing these instances are **uf200-01.txt to uf200-10.txt**, available in a .zip file **uf200.zip** on the course homepage. You can load .txt file in MATLAB using the *load* command. For each instance, you will then have a matrix of size 860 x 4. The last column consists of all zeros (an artifact of the formatting used in SATLIB files) and must be discarded. Each row then corresponds to a different clause, each consisting of 3 literals. For example, the 2nd row is [73, -22, -24], and is equivalent to the clause {s73, ~s22, ~s24}. **Your task is to use heuristic search algorithms to find the truth assignment (indicated by a vector of 200 binary variables) that yields the minimum number of unsatisfied clauses for each instance.**



**Specific SAT Project requirements:**

- You will need to code your own 'cost' function for this problem.
- For each algorithm you use, you are expected to perform at least 10 optimization trials per instance (e.g. for simulated annealing, at least 100 total optimization trials).
- Algorithm performance should be assessed based first on whether the algorithms find satisfying assignments (or the best value of the objective function) and then second on the number of evaluations required to find the optimal solution. In other words, finding a satisfiable solution is more important than the speed at which an algorithm reduces the number of unsatisfied clauses.
- Based on the algorithm rankings for each of the 10 instances, report the algorithm that performed best overall.
- Report how many of these 10 instances are satisfiable.
- Any general course project requirements that are not covered in above bullets.