ECE1387 CAD for Digital Circuit Synthesis and Layout Assignment #3 A Branch-and-Bound Solver for MAX-SAT and Weighted MAX-SAT

J. Anderson

Assignment date: October 30, 2021
Assignment due: November 19, 2021 (by electronic submission)
Marks available: 20 (-2 per day late)

1 Introduction

MAX-SAT refers to the *maximum satisfiability problem*, where the objective is to find the maximum number of satisfiable clauses in a Boolean formula expressed in conjunctive-normal form (CNF). A Boolean formula in CNF is a conjunction (AND) of clauses, where each clause is a disjunction (OR) of Boolean literals. For example:

$$F = (x_1 \vee -x_2) \wedge (-x_1 \vee x_2) \wedge (x_2) \wedge (-x_1 \vee x_3) \wedge (-x_1 \vee -x_3) \qquad (1)$$

where $\vee$ means OR; $\wedge$ means AND; and $-x_1$ means "not x_1". Each x_i is a Boolean variable that can assume the value of TRUE or FALSE. The CNF formula F has five clauses and 3 variables. By setting x_1, x_2, x_3 to FALSE, the first, second, fourth and fifth clauses are satisfied (evaluate to TRUE). Closer inspection will reveal there is no assignment of TRUE/FALSE values to variables that allows all five clauses to evaluate to TRUE. Hence, the *maximum* number of clauses that can be satisfied is four. A variety of different CAD (and other) problems can be formulated as MAX-SAT problem instances. For this assignment, you are to build a MAX-SAT solver using branch-and-bound optimization.

The input file format is shown below. The first line contains two integers, representing the number of variables n_v and number of clauses n_c, respectively. This is followed by n_c lines, each representing a clause in the CNF. Each of these lines has a list of integers, then a 0, then a weight. Considering first the integers

before the 0, a positive integer represents a variable in true (non-complemented) form; a negative integer represents a variable in negated (complemented) form. For example, the line: `1 -2 0 1`, corresponds to the clause $(x_1 \vee -x_2)$. The example file below represents the Boolean function F shown above.

```
3 5
1 -2 0 1
-1 2 0 1.5
2 0 2
-1 3 0 1
-1 -3 0 1
```

For the B&B decision tree for a MAX-SAT solver, the following design is typically used: each level of the tree should correspond to one Boolean variable. As in the partitioning example described in class, each node of the tree will have two edges descending from it, corresponding to that variable being assigned TRUE or FALSE, respectively. Each decision tree node thus represents a partial assignment of TRUE/FALSE values to Boolean variables. Leaf nodes represent an assignment of TRUE/FALSE values to all Boolean variables.

1.1 Part A – Standard (Vanilla) MAX-SAT

In this part, your solution should ignore the weights in the input file. For each internal node, your bounding function must compute a lower bound on the number of unsatisfiable clauses given the node's partial assignment of TRUE/FALSE values. If the computed lower bound equals or exceeds the count of unsatisfied clauses for the current "best" solution, then clearly, the node can be pruned. However, with a bit of thinking, you can likely come up with a bounding function that does better than this bare minimum. Innovate to devise a clever bounding function for a given partial solution.

Your solver should find the maximum number of clauses that can be satisfied, and also print out the TRUE/FALSE assignments to variables producing the maximum. The progress of your MAX-SAT optimization (the decision tree) should be displayed with computer graphics, as in Assignments #1 and #2, using the package provided on the Piazza site.

1.2 Part B – Weighted MAX-SAT

In this part, your solution should consider the clause weights in the input file and choose TRUE/FALSE assignments to variables to maximize the following objective function:

$$TotalWeight = \sum_{c \in F} satisfied(c) \cdot weight(c) \tag{2}$$

where for clause c, $satisfied(c)$ is 1 if the clause is satisfied for the variable assignment (evaluates to TRUE), otherwise $satisfied(c)$ is 0. $weight(c)$ is the

weight of clause c in the input file. That is, your goal is to find *weighted* maximum of satisfied clauses. This problem is referred to as *weighted MAX-SAT*.

Your solver should print out $TotalWeight$, and also print out the TRUE/FALSE assignments to variables producing the maximum $TotalWeight$. The progress of your weighted MAX-SAT optimization (the decision tree) should be displayed with computer graphics.

2 What To Do and What to Hand In?

Your solver should run on the ECF network. Instructions for electronic submission will be posted on Piazza.

1. A report with a short description of the flow of your program assuming that I have a basic knowledge of branch-and-bound optimization. Please be sure to describe:
   - For both parts A and B: How did you determine the initial "best" solution?
   - For both parts A and B: The bounding function that you used and others that you experimented with.
2. For both parts A and B, and for each test case on the Piazza site, provide the following:
   - A plot of the decision tree for the test case.
   - A count of the number of tree nodes visited. There will be a **prize** for the smallest number of nodes traversed on the third test case (`3.cnf`). **NOTE:** Tuning the algorithm to the exact problem is not allowed. Regarding the definition of "visited", if you are calling your bounding function for a node, that should count as "visiting" the node.
   - The run-time for each of the test cases on `remote.ecf.utoronto.ca`
   - A table showing maximum number of clauses satisfied for each test case and the $TotalWeight$ of the satisfied clauses.
3. Summarize and discuss your results.

Aside: You may be interested to know that MAX-SAT solving is an area of active research and that MAX-SAT has many applications in CAD for integrated-circuit design. In fact, there is a competition held yearly where MAX-SAT researchers "bake off" their solvers against one another:
`https://maxsat-evaluations.github.io/`.