Logic in CS Autumn 2024

Problem Sheet 4

S. Krishna

The DPLL algorithm, used in this tutorial, consists of the following steps, which are done before each decision step.

1. Unit propogation (UP): If a clause contains only a single unassigned literal (unit clause), the variable is assigned the necessary value to make this literal true.
2. Pure literal elimination (UP): If a propositional variable occurs with only one polarity in the formula, it is called pure. The variable is assigned the necessary value to make all the clauses containing this literal true.

1. If DPLL always chooses to assign 0 to a decision variable before assigning 1 (if needed) to a variable, then show that DPLL will never need to backtrack when given a Horn formula encoded in CNF as input.

   Solution

   In HornSAT algorithm, we mark the occurences of a variable v on the RHS of a Horn clause C (written as an implication, $C = (p_1 \wedge \cdots \wedge p_n \rightarrow v)$) true, when all the variables on the LHS is marked true. This is equivalent to a unit propogation step in the DPLL algorithm, when the unit clause is the positive literal v.

   Since, the HornSAT algorithm just tells to mark occurrences of variables true, as mentioned above, until the step cannot be done anymore, and mark all the other variables false to get a satisfying assignment. Else, you would get a clause in which all the variables in the LHS are marked true and the RHS is $\bot$, making the formula unsatisfiable.

   Hence, by the correctness of HornSAT algorithm, we can conclude that on applying only unit propogation steps on a Horn formula H until no more can be done, H is unsatisfiable $\iff$ The partial assignment reduced one of the clauses to $\bot$. Therefore, after the initial unit propogation steps, assigning 0 to any decision variable will reduce it to an equisatisfiable Horn formula (since assigning all the variables false will give a satisfying assignment, by HornSAT).

2. Let's try to solve the HornSAT problem using DPLL. We can convert each Horn clause into a CNF clause by simply rewriting $(a \rightarrow b)$ as $(\neg a \vee b)$, which preserves the semantics of the formula. The resultant formula is in CNF.

   Suppose our version of DPLL always assigns 1 to a decision variable before assigning 0 (if needed). How many backtrackings are needed if we run this version of DPLL on the (CNF-

ised version of) the following Horn formulae, assuming DPLL always chooses the unassigned variable with the smallest subscript when choosing a decision variable?

(a)
$$\left(\left(\bigwedge_{i=0}^{n-1} x_i\right) \to x_n\right) \wedge \bigwedge_{i=0}^{n-1}(x_n \to x_i) \wedge \left(\left(\bigwedge_{i=0}^{n} x_i\right) \to \bot\right)$$

(b)
$$\bigwedge_{i=0}^{n-1}((x_i \to x_{n+i}) \wedge (x_{n+i} \to x_i) \wedge (x_i \wedge x_{n+i} \to \bot))$$

Solution

In part (a), no unit clauses or pure literals are obtained until all of x_0 through x_{n-1} are assigned the value 1, one at a time. Once x_{n-1} is assigned 1, UP leads to a conflict—x_n must be assigned both 1 and 0 by UP. This causes a backtrack, which ends up setting x_{n-1} to 0. Once this happens, UP assigns the value 0 to x_n, resulting in the partial assignment $x_{n-1} = x_n = 0$, which satisfies all clauses. Therefore, there is exactly one backtrack.

In part (b), every time a variable x_i for $0 \leq i \leq n-1$ is assigned 1, UP causes x_{n+i} to be in conflict (must be assigned both 0 and 1). This induces a backtrack that sets x_i to 0, followed by UP setting x_{n+i} to 0. The DPLL algorithm will then choose x_{i+1} as the next decision variable, assign it 1, and the above process repeats. So, in this case, DPLL will incur n backtracks, one for each of $x_0, \ldots, x_{n-1}$.

3. Let P, Q, R be propositional variables. Convert the formula
$$\neg(P \vee (\neg Q \wedge R)) \to (\neg P \wedge (Q \vee \neg R))$$
to an equisatisfiable Conjunctive Normal Form (CNF) formula using Tseitin encoding.

Solution

Consider the parse of the given formula in Figure 1. We introduce 6 fresh variables $t_1, \ldots, t_6$ to replace for the sub-formulae of the given formula.
Then, the final formula after Tseitin encoding will be:
$$\begin{aligned}(t_1 \leftrightarrow (\neg Q \wedge R)) \wedge (t_2 \leftrightarrow P \vee t_1) \wedge (t_3 \leftrightarrow \neg t_2) \\ \wedge (t_4 \leftrightarrow (Q \vee \neg R)) \wedge (t_5 \leftrightarrow (\neg P \wedge t_4) \\ \wedge (t_6 \leftrightarrow (t_3 \to t_5)) \wedge (t_6)\end{aligned}$$

Homework: Convert the above formula to a CNF using common rules of semantic equivalences.

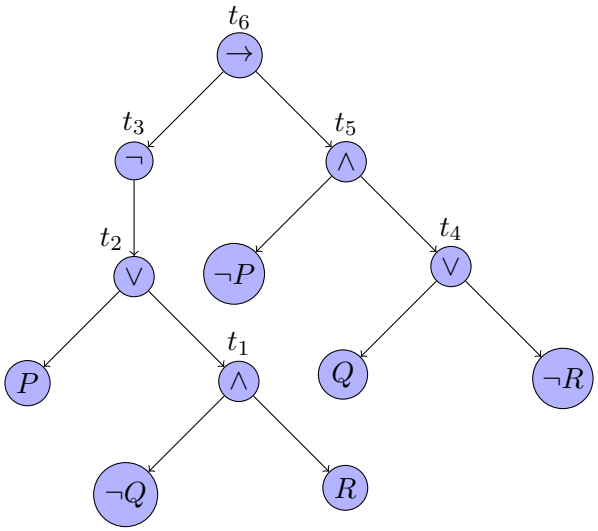

Figure 1: Tseitin Encoding

4. Consider the following CNF formula:

$$\begin{aligned}(\neg p_1 \vee p_2 \vee p_3) &\wedge (\neg p_1 \vee p_3 \vee p_9) \\ &\wedge (\neg p_2 \vee \neg p_3 \vee p_4) \\ &\wedge (\neg p_4 \vee p_5) \\ &\wedge (\neg p_4 \vee p_6 \vee \neg p_8) \\ &\wedge (\neg p_5 \vee \neg p_6) \\ &\wedge (p_7 \vee p_1 \vee \neg p_{10}) \\ &\wedge (p_1 \vee p_8) \\ &\wedge (\neg p_7 \vee \neg p_8)\end{aligned}$$

(a) Check if this CNF formula is satisfiable using the DPLL method.

(b) Assume $p_9 = \bot$ and $p_{10} = \top$. Now, check if we can find a solution for the above formula using the DPLL method.

Solution

(a) Notice that there is no unit clause (clause with a single literal) in the given CNF formula. But p_9 and $\neg p_{10}$ are pure literals (literals which appear either positive or negative throughout). So we begin with Pure Literal Elimination (PLE) and proceed as follows:

Step No.	Variable Assigned	Value Assigned	Reason
1	p_9	1	PLE
2	p_{10}	0	PLE
3	p_7	0	PLE
4	p_1	0	Decision
5	p_8	1	UP on $(p_1 \vee p_8)$
6	p_2	0	PLE
7	p_4	0	PLE
8	p_5	0	PLE

(b) If $p_9 = \bot$ and $p_{10} = \top$, the formula after simplification becomes:

$$\begin{aligned}(\neg p_1 \vee p_2 \vee p_3) &\wedge (\neg p_1 \vee p_3) \\ &\wedge (\neg p_2 \vee \neg p_3 \vee p_4) \\ &\wedge (\neg p_4 \vee p_5) \\ &\wedge (\neg p_4 \vee p_6 \vee \neg p_8) \\ &\wedge (\neg p_5 \vee \neg p_6) \\ &\wedge (p_7 \vee p_1) \\ &\wedge (p_1 \vee p_8) \\ &\wedge (\neg p_7 \vee \neg p_8)\end{aligned}$$

Step No.	Variable Assigned	Value Assigned	Reason
1	p_1	1	Decision
2	p_7	0	PLE
3	p_8	0	PLE
4	p_3	1	UP on $(\neg p_1 \vee p_3)$
5	p_2	0	PLE
6	p_4	0	PLE
7	p_5	0	PLE

5. Discuss the best-case and worst-case time complexities of the DPLL algorithm. Provide a representative example for each scenario.

Solution

The best case time complexity of DPLL is $\mathcal{O}(n)$, because the algorithm needs to read through the formula atleast once to check for a pure literal. An example of a formula on which DPLL terminates within $\mathcal{O}(n)$ time is $p_1 \wedge p_2 \wedge \cdots \wedge p_n$.

The worst case time complexity of $\mathcal{O}(2^n)$ and it is attained by any unsatisfiable formula which does not result in a conflict during initial steps of Unit propagation and Pure literal elimination (*Why?*).