Project of Algorithms on Condition Satisfiability

Kevin Lei

July 22, 2024

1 Main Idea

In the "Condition Satisfiability Problem", we are given the following:

1. A number of n boolean variables $x_1, x_2, \ldots, x_n$, where x_i must be either true or false.
2. A set of P "lead-to" conditions $L = \{T_1, T_2, \ldots, T_P\}$, where each T_i has $k_i \geq 0$ boolean variables to the left of its $\Rightarrow$ symbol and always one boolean variable to the right. A "lead-to" condition takes the form of $(x_{i_1} \wedge x_{i_2} \wedge \ldots \wedge x_{i_k}) \Rightarrow x_j$. The $k_i + 1$ variables for each $T_i \in L$ are unique, and the degenerate case with $k = 0$ simply means that x_j is true.
3. A set of Q "false-must-exist" conditions $F = \{M_1, M_2, \ldots, M_Q\}$, where each M_i is a cumulative logical OR of $m_i > 0$ negated boolean variables. A "false-must-exist" condition takes the form of $(\neg x_{i_1} \vee \neg x_{i_2} \vee \ldots \vee \neg x_m)$. The m_i variables for each $M_i \in F$ are unique.

We want to find the truth values of $x_1, x_2, \ldots, x_n$ such that the P conditions in L and the Q conditions in F will all evaluate to true. Such an assignment of truth values is called a "satisfying solution". If there is no such assignment, then we want to output "*No satisfying solution exists*".

The main idea behind this algorithm is to initialize a set of n boolean variables to be all false. Then, we iterate through all of the conditions until no more changes are being made. We check the "lead-to" conditions first, and if any has a true left-hand side and a false right-hand side, then we set the variable on the right-hand side to true. We then check the "false-must-exist" conditions, and if any has all of its variables to be true, then we set the first variable in the statement to false. Once no more changes can be made according to our rules, we do a final check of all the conditions. If any of them fail, then we output "*No satisfying solution exists*". After we check all of the conditions and they all pass, then we output the satisfying solution.

2 Pseudocode

Algorithm 1: Condition Satisfiability

Input: n boolean variables, a set $L = \{T_1, T_2, \ldots, T_P\}$ of P "lead-to" conditions, and a set $F = \{M_1, M_2, \ldots, M_Q\}$ of Q "false-must-exist" conditions.

Output: A satisfying solution or "No satisfying solution exists".

$x_1, x_2, \ldots, x_n$ = false;
changed = true;
while *changed* **do**
    changed = false;
    for *each* $T_i \in L$ **do**
        if $(x_{i_1} \wedge x_{i_2} \wedge \ldots \wedge x_{i_k}) \wedge \neg x_j$ **then**
            x_j = true;
            changed = true;
        end
    end
    for *each* $M_i \in F$ **do**
        if $\neg M_i$ **then**
            x_{i_1} = false;
            changed = true;
        end
    end
end
for *each* $T_i \in L$ **do**
    if $(x_{i_1} \wedge x_{i_2} \wedge \ldots \wedge x_{i_k}) \wedge \neg x_j$ **then**
        return "No satisfying solution exists";
    end
end
for *each* $M_i \in F$ **do**
    if $\neg M_i$ **then**
        return "No satisfying solution exists";
    end
end
return $x_1, x_2, \ldots, x_n$;

3 Proof of Correctness

4 Runtime Complexity Analysis