SAT, Clique y Idset

Alberto Benavides
19 de febrero de 2020

1. SAT

En álgebra booleana, una fórmula está dada en *forma normal conjuntiva* cuando consiste de una conjunción de disyunciones que, a su vez, no incluyen variables con su propio complemento [3]. Por ejemplo, dadas las variables a, b, c las siguientes son fórmulas en forma normal conjuntiva:

$$(a \vee b),$$

$$\neg a,$$

$$a \wedge (b \vee \neg c),$$

$$(a \vee b) \wedge (\neg c \vee b).$$

El problema de satisfacibilidad booleana (SAT por su abreviatura) consiste en determinar si las variables de una fórmula dada en la forma normal conjuntiva pueden tomar valores verdaderos $\top$ o falsos $\bot$ de modo que la fórmula en sí sea verdadera $\top$ [4]. Si ninguna combinación en los valores de las variables da por resultado que la expresión sea verdadera, se dice que que el problema es *no satisfacible*.

Una manera exhaustiva de determinar esto consiste en realizar una *búsqueda de profundidad*, representable en un árbol binario mostrado en la figura 1.1 (p. 2), donde en cada nodo a partir de la raíz se define el valor de la variable de modo que al tener valores verdaderos para cada variable se evalúa si determinada fórmula en forma normal conjuntiva es verdadera. En caso de no serlo, se utiliza un algoritmo de *vuelta atrás* (*backtracking* en inglés) [1] para probar con otra combinación de valores para las variables y se continúa este proceso hasta que se halle una combinación satisfacible o se terminen las combinaciones.

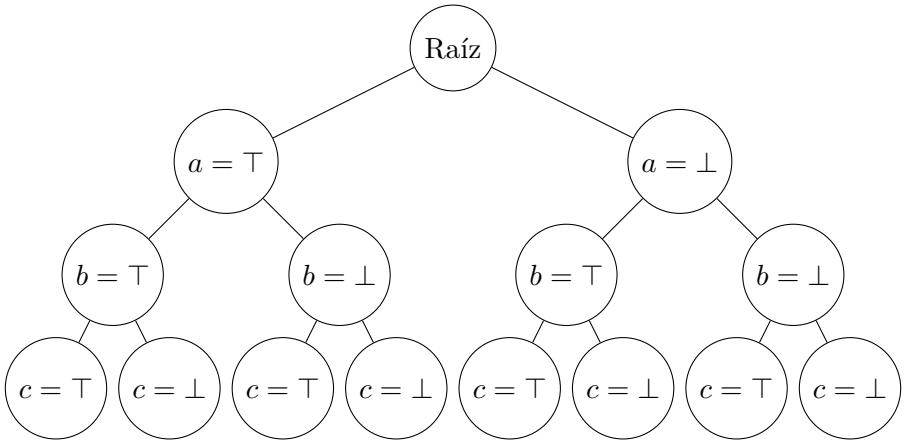

Figura 1.1: Grafo de búsqueda de profundidad para algoritmo de satisfacibilidad de tres variables booleanas.

2. Clique

Un conjunto C es un clique de un grafo G si $C \subseteq G$ y los vértices $u, v \in G \land u \neq v \implies u, v \in E(G)$ [2]. De esta manera, puede existir más de un clique en un grafo si su número de vértices (conocido como orden) es el mismo.

Para encontrar un clique se puede seguir el algoritmo 1 (p. 3).

3. Conclusiones

Para resolver estos problemas, en el peor de los casos, se deben recorrer todas las variables o nodos y compararlos entre sí, lo cual crece exponencialmente al incrementarse el número de variables o nodos en cada caso. Esto implica un gran esfuerzo computacional cuando se trata de fórmulas o grafos muy grandes (millones de variables o nodos).

Existen esfuerzos por simplificar estas búsquedas, como algoritmo de vuelta atrás mencionado, que consisten en detener una búsqueda cuando se encuentran contradicciones en el caso del problema de satisfacibilidad.

Referencias

[1] Wolfram Research. Backtracking, . URL `http://mathworld.wolfram.com/Backtracking.html`.

[2] Wolfram Research. Clique, . URL `http://mathworld.wolfram.com/Clique.html`.

[3] Wolfram Research. Conjunctive normal form, . URL `http://mathworld.wolfram.com/ConjunctiveNormalForm.html`.

```
i = 0;
para cada u ∈ G hacer
    C_i = {u};
    para cada v ∈ N(u) hacer
        para cada t ∈ C_i hacer
            si t ∈ N(v) entonces
                C_i+ = {v};
            fin
            en otro caso
                C_i− = {v};
                Romper;
            fin
        fin
    fin
    i = i + 1;
fin
devolver máx(|C_i|)
```

Algoritmo 1: Algoritmo para encontrar clique.

[4] Wolfram Research. Satisfiability problem, . URL `http://mathworld.wolfram.com/SatisfiabilityProblem.html`.