Un graphe pour résoudre 2-SAT

Philippe Gambette [1]

Résumé

On explique ci-dessous le joli algorithme d'Aspvall, Plass et Tarjan [2] résolvant le problème 2-SAT par construction d'un graphe orienté et recherche de certains chemins dans ce graphe. Plus de détails sur l'algorithme sur http://www.cl.cam.ac.uk/users/mr/qbf2slds.pdf

Le problème 2-SAT consiste à trouver une valuation de n variables $x_1, \ldots, x_n$ afin qu'une conjonction de *clauses* [3] à deux littéraux (par exemple $(x_1 \vee x_2) \wedge (x_3 \vee \neg x_4) \wedge (x_1 \vee x_4) \wedge (x_1 \vee x_3) \wedge (\neg x_2 \vee x_4)$) soit vraie.

Ce problème est polynomial ("contrairement" à *3-SAT* qui se définit similairement, avec des clauses à trois littéraux et qui est NP-complet), et on peut le résoudre en le réduisant à un problème de graphes orientés.

On considère le graphe ayant $2n$ sommets, dont n sont étiquetés par les variables, et les n autres par les négations des variables, illustré en figure 1. On représente les m clauses de la formule par $2m$ arêtes dans le graphe de la façon suivante : chaque clause de la forme $(y \vee z)$ peut être vue comme l'implication $\neg y \Rightarrow z$ ou $\neg z \Rightarrow y$, donc on ajoute pour chaque clause ainsi notée les arêtes $(\neg y, z)$ et $(\neg z, y)$.

Calculons alors les composantes fortement connexes du graphe (en temps linéaire avec un algorithme de Tarjan [4]). La formule n'est pas satisfiable ssi il existe une composante fortement connexe contenant une variable x_i et $\neg x_i$ (puisque dans ce cas on a $x_i \Leftrightarrow \neg x_i$). Si aucune composante de ce type n'est trouvée, cherchons une valuation solution.

L'algorithme de Tarjan fournit un ordre topologique sur les composantes fortement connexe (c'est à dire un ordre $\prec$ tel que s'il existe une arête de S_i vers S_j alors $S_i \prec S_j$). De plus, une composante fortement connexe $S_i = \{x_1, \ldots, x_k\}$ est telle que $\{\neg x_1, \ldots, \neg x_k\}$ est aussi une composante fortement connexe, que l'on note $\neg S_i$. On considère alors les composantes dans l'ordre inverse, et on va leur affecter des valuations : tant qu'il reste une composante non valuée S_i, affecter à tous ses noeuds la valeur VRAI et à tous les noeuds de $\neg S_i$ la valeur FAUX. Ceci assure que tous les noeuds sont finalement valués sans aucun chemin de type VRAI $\Rightarrow$ FAUX.

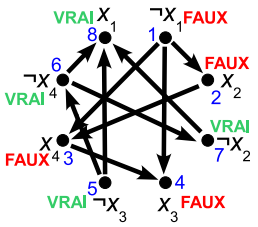

FIGURE 1 – Le graphe utilisé pour résoudre 2-SAT sur $(x_1 \vee x_2) \wedge (x_3 \vee \neg x_4) \wedge (x_1 \vee x_4) \wedge (x_1 \vee x_3) \wedge (\neg x_2 \vee x_4)$: chaque sommet est en fait une composante fortement connexe, les numéros en bleu indiquent un ordre topologique sur les composantes trouvées par l'algorithme de Tarjan. Comme il n'existe aucune composante fortement connexe contenant x_i et $\neg x_i$, on peut effectuer les valuations, dans l'ordre topologique inverse : VRAI à x_1, donc FAUX à $\neg x_1$, VRAI à $\neg x_2$ donc FAUX à x_2, VRAI à $\neg x_4$, donc FAUX à x_4 et VRAI à $\neg x_3$ donc FAUX à x_3.

1. Merci à Arthur Milchior de m'avoir indiqué des erreurs dans une version précédente.
2. Bengt Aspvall, Michael F. Plass, and Robert E. Tarjan. A linear-time algorithm for testing the truth of certain quantified boolean formulas. *Information Processing Letters*, p.121-123, 1979.
3. disjonction de *littéraux*, c'est à dire de variables ou de négations de variables.
4. Robert E. Tarjan. Depth first search and linear graph algorithms, *SIAM J. Computing* 1(2), p.146-160, 1972