

Réseaux booléens: algorithmes de graphes et polynômes booléens

Vincent Delecroix

Bruno Grenet

Damien Jamet

EJCIM 2018

Dans cette feuille de travail on s'intéresse à des algorithmes de graphes et de fonctions booléennes en rapport avec les réseaux booléens étudiés dans le Chapitre 4 du livre "Informatique Mathématique. Une image en 2018".

1 Recouvrement et empilement

On conseille aux lecteurs de jeter un oeil sur la feuille de travail "sudoku" ainsi que les sections 5.5.1 et 5.5.2 du livre avant de s'attaquer à cette partie.

On cherche à résoudre le problème suivant.

Recouvrement

Entrée : A_1, A_2, \dots, A_n sous-ensembles d'un ensemble fini E

Sortie : un sous-ensemble $C \subset E$ de taille minimale tel que $C \cap A_i \neq \emptyset$ pour tout $i \in \{1, 2, \dots, n\}$

Ce problème de recouvrement est une façon de calculer $\tau(G)$ et $\tau^+(G)$ de la Section 4.3.2 "borne du feedback positif" du livre (sous couvert qu'on ait calculé l'ensemble des cycles ou des cycles positifs).

1. En utilisant des variables binaires x_i pour $i \in E$ écrire un problème linéaire (en nombre entiers) pour répondre au problème de recouvrement.
2. Refaire l'exercice en utilisant un solveur SAT.

On s'intéresse maintenant au problème suivant.

Empilement

Entrée : A_1, A_2, \dots, A_n sous-ensembles d'un ensemble fini E

Sortie : un sous-ensemble $I \subset \{1, 2, \dots, n\}$ de taille maximale tel $A_i \cap A_j = \emptyset$ pour toute paire d'éléments distincts i et j de I

Ce problème d'empilement est une façon de calculer $\nu(G)$ et $\nu^+(G)$ de la section 4.3.4 "Influence des cycles disjoints dans le cas monotone".

3. En utilisant des variables binaires x_i pour $i \in E$ écrire un problème linéaire (en nombre entiers) pour répondre au problème d'empilement.
4. Refaire l'exercice en utilisant un solveur SAT.

2 Graphe d'interaction

Pour encoder un graphe d'interaction on va utiliser les graphes orientés de Sage en mettant des étiquettes sur les arêtes : +1 pour une interaction positive, -1 pour une interaction négative et 0 pour les interactions à la fois positives et négatives.

Pour construire le graphe d'interaction G en haut de la page 162 on peut faire

```
sage: G = DiGraph(multiedges=False, loops=True)
sage: G.add_edge(1, 1, -1) # arête négative 1 -> 1
sage: G.add_edge(1, 2, +1) # arête positive 1 -> 2
```

```

sage: G.add_edge(1, 3, +1) # arête positive 1 -> 3
sage: G.add_edge(2, 1, -1) # arête négative 2 -> 1
sage: G.add_edge(2, 3, +1)
sage: G.add_edge(3, 1, +1)
sage: G.add_edge(3, 2, +1)
sage: G.add_edge(3, 3, +1)

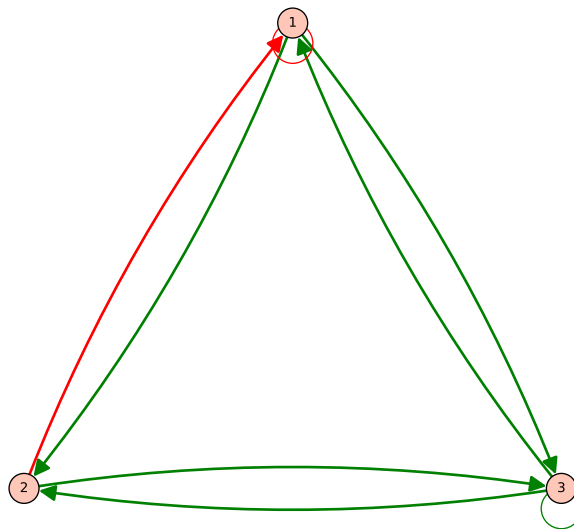
```

On peut alors le dessiner en utilisant

```

sage: P = [e for e in G.edges() if e[2] == +1]
sage: N = [e for e in G.edges() if e[2] == -1]
sage: G.set_pos({1: (0,sqrt(3.)), 2: (-1,0), 3: (1,0)})
sage: G.plot(edge_colors={'green': P, 'red': N}, edge_labels=False)

```



5. Écrire une fonction qui étant donné un graphe d'interaction G calcule $\tau(G)$ et $\tau^+(G)$ (indice : vous pouvez utiliser la méthode `all_simple_cycles` des graphes).
6. Écrire une fonction qui étant donné un graphe d'interaction G calcule $\nu(G)$ et $\nu^+(G)$.

On se propose maintenant une approche dynamique pour le calcul de τ . Plutôt que de précalculer l'ensemble des cycles comme on vient de le faire on va faire un calcul dynamique en ajoutant des contraintes au solveur MILP au fur et à mesure. Le pseudo-code est le suivant.

```

M = MixedIntegerLinearProgram(maximization=False)
x = M.new_variable(binary=True)
M.set_objective(sum(x[i] for i in G))
M.solve()
S = ensemble des i tels que x[i] == 1
tant qu'il y a un cycle c dans G \ S
    ajouter à M la contrainte que c doit être couvert
    M.solve()
    S = ensemble des i tels que x[i] == 1
renvoyer S

```

7. Implanter cette nouvelle version pour le calcul de $\tau(G)$.
8. Faire des comparaisons de temps de calcul entre les deux méthodes.

Afin de calculer $\tau^+(G)$ il faudrait un algorithme pour trouver un cycle positif dans un graphe... qui n'a pas l'air un problème algorithmique simple !

9. (**) Développer un algorithme pour trouver un cycle positif dans un graphe.

3 Graphe d'influence d'une fonction booléenne

Une manière d'encoder les fonctions booléennes sont les polynômes booléens, c'est-à-dire les polynômes sur le corps $\mathbb{F}_2 = \{0, 1\}$ quotienté par les relations $x^2 = x$ pour toute variable x .

```
sage: R.<a,b,c,d> = BooleanPolynomialRing()
sage: (a + b) * (a + c + 1)
a*b + a*c + b*c + b
```

10. Justifier la bijection entre fonctions booléennes et polynômes booléens.
11. Les opérations algébriques de base disponibles sur les polynômes sont la multiplication $*$ et l'addition $+$. Déterminer comment effectuer les opérations logiques standards : négation, ou, et.
12. Écrire une fonction `is_growing(p)` qui détermine si le polynôme booléen `p` est croissant.
13. Vérifier que les polynômes booléens en les variables a, b et c suivants sont croissants

$$0, 1, a, ab, abc + bc + a, abc + ab + ac + bc + a + b + c.$$

14. Vérifier que les polynômes suivants ne sont pas croissants

$$a + 1, ab + bc, ab + ac + a + b + c, abc + ab + ac + b.$$

15. Écrire une fonction `interaction(p)` qui étant donné un polynôme booléen `p` retourne la paire (P, N) des variables qui influencent `p` de manière positive et négative respectivement. (indice : utiliser pour chaque variable l'unique décomposition $p = x_i f + g$ où f et g ne contiennent pas x_i et analyser les implications entre f et g)
16. Combien y a-t-il de fonctions booléennes croissantes de 3 variables ?
17. Et pour 4 variables ?
18. Déterminer l'entrée de l'encyclopédie des séquences d'entiers <http://oeis.org/> qui correspond aux fonctions booléennes croissantes (indice : on pourra utiliser la fonction `oeis` de SageMath).