Langages recommandés : Java ou Python

1 Prise en main de `minisat`

Dans tout le TP, nous allons utiliser le programme `minisat`, qui est un solveur de satisfaisabilité de formules propositionnelles (SAT).

Voilà un exemple d'une formule en CNF :

$$(x_1 \vee x_2 \vee x_3) \wedge (\neg x_1 \vee \neg x_2 \vee x_3) \wedge (x_2 \vee \neg x_3) \wedge (x_1 \vee \neg x_2)$$

et la même formule sous le format de base de `minisat` (**dimacs cnf**) :

```
p cnf 3 4
1 2 3 0
-1 -2 -3 0
2 -3 0
1 -2 0
```

- La ligne `p cnf 3 4` indique qu'il s'agit d'une CNF avec 3 variables et 4 clauses.
- Chaque ligne suivante décrit une clause.
- Un nombre positif indique une variable.
- Un nombre négatif indique la négation d'une variable.
- `0` termine la ligne.

Pour utiliser le programme, il suffit de taper la commande

```
> minisat fichier.cnf fichier.out
```

où `fichier.cnf` contient l'exemple ci-dessus. Le résultat est affiché comme ceci :

```
...
Memory used           : 21.00 MB
CPU time              : 0 s

SATISFIABLE
```

Dans le fichier `fichier.out` vous trouvez une affectation satisfaisante :

```
SAT
1 -2 -3 0
```

Ce fichier indique que la formule est saitsfaisable (`SAT`) et qu'une affectation satisfaisante est donnée par $(x_1, x_2, x_3) = (1, 0, 0)$.

Notez que ce n'est pas obligatoire d'entrer les bonnes valeurs pour le nombre de variables et le nombre de clauses. On peut mettre `p cnf 0 0` et dans ce cas `minisat` va donner un avertissement et lui même calculer les bonnes valeurs.

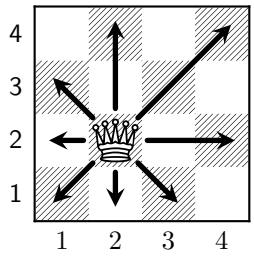
Une petite introduction au format dimacs cnf et l'utilisation de `minisat` se trouve à l'adresse

http://www.dwheeler.com/essays/minisat-user-guide.html

► **Question 1** ◄ Vérifiez que vous trouvez le coupable dans l'exemple "Mystère" du cours en utilisant `minisat`.

2 4 Reines

► **Question 2** ◄ Écrire un programme qui génère un fichier au format **cnf** avec des clauses qui déterminent une solution au problème "4 Reines" vu au TD et au cours.

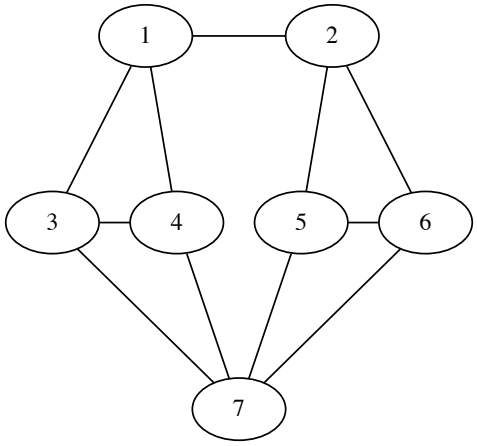


Est-ce qu'il est possible de placer les 4 reines de manière à ce qu'aucune d'entre elles ne soit en prise ?

► **Question 3** ◄ Lancer le solveur `minisat` avec le fichier de l'exercice précédent. Quelle solution obtenez-vous ?

3 Coloration de graphe

► **Question 4** ◄ Écrire un programme qui produit un fichier au format **cnf** avec des clauses qui déterminent une coloration du graphe ci-dessous avec 3 couleurs.



► **Question 5** ◄ Lancer le solveur `minisat` avec le fichier de l'exercice précédent. Quelle solution obtenez-vous ?

► **Question 6** ◄ Écrire un programme qui étant donné un fichier décrivant un graphe G et un entier K, produit un fichier au format **cnf** qui détermine si G a une coloration avec K couleurs.

Le fichier d'entrée est constitué de trois types de ligne :

- `c` marque une ligne qui est un commentaire ;
- `p edge N M` commence un graphe non-orienté avec N sommets (nommés $1, \ldots, N$) et M arêtes ;
- `e X Y` indique une arête entre les sommets X et Y.

L'exemple ci-dessous définit le graphe de la question précédente :

```
c
c un graphe sur 7 sommets
c
p edge 7 10
e 1 2
e 1 3
e 1 4
e 2 5
e 2 6
e 3 4
e 3 7
e 4 7
e 5 6
e 5 7
e 6 7
```

► **Question 7** ◄ Utiliser le programme de la question précédente pour trouver le nombre minimum de couleurs d'une coloration des graphes dans le répertoire :

http://monge.univ-mlv.fr/~thapper/teaching/mmpo/graphes/

► **Question 8** ◄ Pour certains graphes de la question précédente, le solveur est très lent et ne termine pas. Une des causes de ce problème est qu'il y a beaucoup de *symétries* dans l'espace de solutions. Par exemple, pour un triangle, il y a 6 colorations différentes avec 3 couleurs.

On peut essayer de "casser" ces symétries. On fait d'abord un choix de $K-1$ sommets $v_1, v_2, \ldots, v_{K-1}$. Ensuite, on ajoute des contraintes qui disent "le sommet v_1 prend la couleur 1", "le sommet v_2 prend soit la couleur 1 soit la couleur 2", etc., jusqu'à "le sommet v_{K-1} prend une des couleurs 1, 2, ..., $K-1$". Noter que si le graphe a une coloration avec K couleurs, alors il y a toujours une coloration avec ces contraintes.

Les clauses suivantes permettent d'ajouter les contraintes décrites ci-dessus :

Pour chaque sommet $i = 1, \ldots, K-1$, ajouter la clause $x_{i1} \vee x_{i2} \vee \cdots \vee x_{ii}$

Conseil : Cette méthode est plus efficace si vous pouvez trouver un ensemble de sommets $v_1, v_2, \ldots, v_{K-1}$ ayant beaucoup d'arêtes entre eux.