

Projet INF242

Stéphane Devismes & Pascal Lafourcade

23 janvier 2012

Pour ce projet les étudiants doivent former des groupes de 3 ou 4 et choisir un sujet ou bien en proposer un à leur responsable de TD.

1 Planning.

- Distribution du projet au premier cours.
- A la fin de la deuxième semaine de TD, constitution des groupes et choix des sujets.
- Livraison par email à votre enseignant de TD au plus tard le jeudi 1er avril 2010 à minuit du pré-rapport décrivant la partie modélisation du projet (5 pages maximum).
- Livraison le lundi 10 mai 2010 à minuit par email du rapport final.
- Dans la semaine du 10 mai 2010, soutenance de 20 minutes par groupe, questions incluses (démonstration du travail réalisé en groupe).

2 Travail à réaliser.

Partie 1. Lors de la première partie du projet il vous est demandé de rédiger un pré-rapport dans lequel vous :

1. Présentez le problème choisi en français (règles, exemples, contraintes)
2. Traduisez le sujet en logique du premier ordre
3. Modélisez le problème choisi en forme normale conjonctive (produit de clauses).
4. Définissez le format des fichiers d'entrée pour une instance du problème. Vous devez également coder un programme permettant de récupérer ces données afin de pouvoir les manipuler pour être capable dans la seconde phase de générer des formules en forme normale conjonctive au format DIMACS.

Partie 2. Dans la seconde phase du projet, une fois le problème modélisé et validé par votre chargé de TD, nous attendons que chaque groupe choisisse un SAT-solveur, par exemple : SATzilla, precosat, MXC, clasp, SApperlot, TNM, gNovelty2+, Rsat, Picosat, Minisat, Zchaff, Jerusat, Satzoo, Limmat, Berkmin, OKSolver, ManySAT 1.1, ... (<http://www.satcompetition.org/>). Vous devrez alors produire :

1. Un programme dans le langage de votre choix qui générera pour une instance du problème un fichier au format DIMACS permettant à un SAT-solveur de résoudre le problème.
2. Un programme qui, à partir de la trace produite par le SAT-solveur choisi, affichera la solution du problème de manière compréhensible.
3. Un rapport où
 - Vous détaillerez l’implémentation de vos programmes.
 - Vous illustrerez également l’utilisation de vos programmes en analysant des exemples pertinents d’exécution.
 - Vous fournirez aussi le code source du projet.
4. Une soutenance où vous exécuterez sur une machine personnelle ou une machine de l’université votre projet. Votre programme devra au minimum :
 - Permettre à l’utilisateur de rentrer un fichier décrivant une instance du problème choisi.
 - Vérifier que le fichier a le bon format.
 - Afficher le fichier DIMACS correspondant au fichier rentré.
 - Afficher de manière compréhensible la solution trouvée par le SAT-solveur choisi.

Note : Le contenu du pré-rapport et le rapport final seront notés (autant être précis, clair et concis). La soutenance est OBLIGATOIRE (sauf pour les dispenses officielles), toute absence conduira le jury à délivrer la note de 0 au projet pour le candidat absent. Chaque membre du projet devra intervenir pendant la soutenance. Il faudra présenter le sujet du projet, mais aussi de donner une démonstration de vos programmes sur machine.

Le format DIMACS : Le format des fichiers d’entrées des SAT-solveurs est un standard international pour la représentation de formules en forme normale conjonctive. Un fichier en format DIMACS commence par une ligne qui spécifie qu’il s’agit d’une forme normale conjonctive, qui précise combien de variables la formule contient, et de combien de clauses disjonctives elle est constituée. Par exemple : `p cnf 5 3` indique que le fichier contient une formule en forme normale conjonctive, avec 5 variables et 3 clauses. Ensuite, le

fichier est composé de plusieurs lignes, une par clause. Chaque ligne contient des entiers positifs et/ou négatifs, et se termine par 0. Un entier positif i indique que la i -ème variable apparaît avec polarité positive dans la clause. Un entier négatif $-i$ indique que la i -ème variable apparaît avec polarité négative dans la clause. Par exemple, le fichier suivant au format DIMACS représente la formule :

$$(x_1 \vee \neg x_5 \vee x_4) \wedge (\neg x_1 \vee x_5 \vee x_3 \vee x_4) \wedge (\neg x_3 \vee \neg x_4)$$

```

c
c start with comments
c
c
p cnf 5 3
1 -5 4 0
-1 5 3 4 0
-3 -4 0

```

3 Sujets.

- Reines : comment un SAT-solveur peut-il placer k reines sur un échiquier de telle sorte qu'aucune d'entre elles ne soit prenable en un coup par une autre ?
- Foot : comment un SAT-solveur peut-il planifier un championnat pour x équipes avec des matchs aller-retour, sachant qu'à chaque journée toutes les équipes jouent et qu'une équipe reçoit une et une seule fois toutes les autres ?
- Sudoku : comment un SAT-solveur peut-il résoudre une grille de sudoku ?
- Tomographie : une image en noir et blanc est une matrice remplie de cases vides ou noires. Comment un SAT-solveur peut-il reconstituer une image à partir du nombre de cases noires de chaque ligne et de chaque colonne ?
- Master Mot : comment un SAT-solveur peut-il résoudre un Master Mot ?
- Squaro : comment un SAT-solveur peut-il résoudre une grille de Squaro ?
- Autres

4 Exemple pour des pigeons.

Nous illustrons par un exemple simple le minimum que nous attendons des étudiants pour le projet de INF242.

4.1 Problème

Un colombophile possède n nids et p pigeons. Il souhaite que :

- chaque pigeon soit dans un nid
- il y ait au plus un pigeon par nid.

Comment la logique peut-il l'aider ?

4.2 Modélisation en logique du premier ordre

Le prédicat $P(i, j)$ représente le fait que le pigeon i est dans le nid j . Les contraintes du problème se modélisent comme suit :

- Chaque pigeon est dans un nid : $\forall i, \exists j, P(i, j)$.
- Il y a au plus un pigeon par nid : $\forall i, \forall k, i \neq k \implies \forall j, \overline{P(i, j)} \vee \overline{P(k, j)}$.

4.3 Modélisation en forme normale conjonctive

La variable booléenne $x_{i,j}$ représente le fait que le pigeon i est dans le nid j . Par exemple, pour un ensemble de pigeons $\{a, b, c\}$ et un ensemble de nids $\{1, 2, 3, 4\}$, nous devons donner au SAT-solveur les contraintes suivantes :

$$\begin{aligned} & (x_{a,1} \vee x_{a,2} \vee x_{a,3} \vee x_{a,4}) \\ \wedge & (x_{b,1} \vee x_{b,2} \vee x_{b,3} \vee x_{b,4}) \\ \wedge & (x_{c,1} \vee x_{c,2} \vee x_{c,3} \vee x_{c,4}) \\ \wedge & (\overline{x_{a,1}} \vee \overline{x_{b,1}}) \wedge (\overline{x_{a,1}} \vee \overline{x_{c,1}}) \wedge (\overline{x_{b,1}} \vee \overline{x_{c,1}}) \\ \wedge & (\overline{x_{a,2}} \vee \overline{x_{b,2}}) \wedge (\overline{x_{a,2}} \vee \overline{x_{c,2}}) \wedge (\overline{x_{b,2}} \vee \overline{x_{c,2}}) \\ \wedge & (\overline{x_{a,3}} \vee \overline{x_{b,3}}) \wedge (\overline{x_{a,3}} \vee \overline{x_{c,3}}) \wedge (\overline{x_{b,3}} \vee \overline{x_{c,3}}) \\ \wedge & (\overline{x_{a,4}} \vee \overline{x_{b,4}}) \wedge (\overline{x_{a,4}} \vee \overline{x_{c,4}}) \wedge (\overline{x_{b,4}} \vee \overline{x_{c,4}}) \end{aligned}$$

4.4 Résolution par un SAT-solveur

Après avoir traduit la forme normale conjonctive en un fichier DIMACS, un SAT-solveur donnera une affectation à chaque variable satisfaisant les contraintes du problème. Ainsi nous saurons comment placer les pigeons. Nous remarquons qu'il nous est facile de trouver une solution à ce problème alors que l'ordinateur doit résoudre un nombre de contraintes exponentielles.