Projet 2 L3IF — Deuxième rendu

à rendre pour le **22/03/2016 à 23h59**

envoyez une archive *qui compile* à `olga.kupriianova@ens-lyon.fr`, `antoine.plet@ens-lyon.fr` et `daniel.hirschkoff@ens-lyon.fr`

1 Améliorer les imperfections du rendu 1

Corrigez les limitations et bugs de votre rendu 1 (vous ne saviez pas traiter les entrées avec des commentaires, votre solveur boucle si le nombre de clauses est premier, etc.), et signalez dans le README les corrections que vous avez faites.

2 Binômes non avancés

2.1 Reprise du code

Comme indiqué ci-dessus, il y a dans le rendu 1 des bugs et limitations dont vous êtes conscients, qu'il faudra corriger pour le rendu 2.

Outre cela, vous aurez éventuellement, dans votre retour sur le rendu 1, des consignes pour reprendre certaines parties de votre programme (organisation des fichiers, structuration du code, etc.). Celles-ci font partie du rendu 2.

2.2 Tseitin

Traitez la partie "Tseitin" qui était destinée aux avancés pour le rendu 1. Vous devrez pour ce faire vous référer à la fois au sujet du rendu 1 et au sujet du TP 1 *(NB : le sujet du TP 1 a été mis à jour, suite à une remarque judicieuse de l'un de vous, en lien avec des questions de priorité pour la conjonction lorsque celle-ci est exprimée par un retour à la ligne).*

Pour ceux qui ont fait un "refus d'obstacle" sur lex&yacc, là il n'y a plus le choix : intéressez vous à la question, demandez si nécessaire de l'aide à vos encadrants.

De même, demandez de l'aide (cela est tout à fait possible par mail) pour savoir comment faire un exécutable `resol` qui puisse être appelé en passant une option (`-tseitin` en l'occurrence).

2.3 Si votre rendu 1 ne faisait pas les déductions de DPLL

Ajoutez les déductions :

1. *clause unitaire* (tous les littéraux d'une clause sauf un sont à faux);
2. *polarité unique* (une variable n'apparaît qu'avec une seule polarité parmi les clauses vivantes);
3. *pré-traitements* : avant de démarrer DPLL, détectez les déductions qui peuvent être faites sur la formule de départ, et modifiez-la en conséquence.

Comme toujours, pensez à viser des versions intermédiaires de votre solveur (d'abord juste les déductions "clause unitaire", etc.).

2.4 Si votre rendu 1 faisait des déductions de DPLL

(et si votre rendu 1 n'était pas trop parsemé de bugs) Traitez les parties 3.2 et 3.4.

3 Binômes avancés

3.1 Littéraux surveillés

Ajoutez une option `-wl` à votre solveur, permettant de mettre en œuvre la technique des littéraux surveillés (*watched literals*). À noter qu'en présence des littéraux surveillés, on désactive la propagation "par polarité unique" (mais on garde bien sûr la propagation par clause unitaire). Pensez à expliquer dans le README comment est gérée l'activation/désactivation de la propagation par polarité unique, en lien avec l'option `-wl`.

L'expérience prouve qu'il est difficile d'obtenir une version de `-wl` sans bugs : prévoyez un temps conséquent pour tester et corriger.

3.2 Heuristiques pour les paris

On ajoute ici d'autres options possibles, en déclinant plusieurs manières de faire le pari dans les cas où on ne peut plus faire de déductions mais il reste des littéraux inconnus.

1. Pas d'option particulière : on parie sur *la prochaine* variable inconnue (expliquez dans le README ce que signifie "la prochaine" dans le cas de votre code — ça peut typiquement être par indice croissant, ou alors l'ordre est déterminé par l'ordre d'apparition dans la formule donnée en entrée, etc.).

2. RAND

   On tire au hasard la prochaine variable sur laquelle on parie, ainsi que le pari que l'on fait (variante: compter combien de fois x et $\overline{x}$ apparaissent dans des clauses *vivantes*, et parier sur le plus fréquent).

   À noter au passage qu'il est demandé que RAND soit dynamique*: il ne s'agit pas de permuter aléatoirement l'entrée avant de faire tourner le programme qui parie sur la prochaine variable non assignée, mais bien de parier au hasard "en cours de route".*

   option : `-rand`

3. MOMS (*Maximum Occurrences in clauses of Minimum Size*)

   On parie sur le littéral qui apparaît le plus dans les clauses de taille minimale, parmi les clauses vivantes (intuition: c'est l'endroit "le plus contraint", où la propagation va agir davantage)

   option : `-moms`

4. DLIS (*Dynamic Largest Individual Sum*)

   On parie sur le littéral le plus efficace, i.e., qui rend le plus de clauses satisfaites parmi les clauses vivantes (variante: score$(\alpha) = \sum_{\alpha \in C, C \in \phi} 2^{-|C|}$, choisir α qui maximise le score).

   option : `-dlis`

Bien entendu, toutes les versions fournies du solveur devront être testées et renvoyer des résultats corrects.

3.3 Prétraitements

Si votre programme pour le rendu 1 ne le faisait pas, ajoutez une étape de prétraitement, agissant sur la formule entrée par l'utilisateur :

- détecter les clauses unitaires, les clauses tautologiques (contenant $x_i \vee \overline{x_i}$);
- collecter des informations diverses sur les clauses intervenant dans la formule de départ;
- éventuellement, renuméroter les variables afin d'avoir la bonne taille pour le problème (si on annonce 10 variables mais qu'on ne parle que de x_1 et de x_{10}, on a en fait 2 variables), et de privilégier les variables qui apparaissent beaucoup dans la formule de départ.

3.4 Observations expérimentales

Comparez les performances des différentes versions que vous avez de votre solveur.

Vous trouverez sur la page www du cours un pointeur vers des exemples de problèmes SAT compliqués. Vous serez sans doute également amenés à engendrer vous-mêmes vos fichiers de tests, afin de mettre à l'épreuve des hypothèses au sujet du comportement de telle ou telle heuristique.

Votre rendu devra comporter les éléments suivants :

1. De quoi reproduire vos expériences.

   Faites-vous de petits scripts ou de petits programmes pour lancer les tests, et expliquez dans le fichier README comment se servir de tout cela.

2. Une analyse des résultats obtenus.

   Donnez vos conclusions sur les performances de votre programme, en fonction des diverses options. Soyez raisonnablement synthétiques.

   Si possible, présentez les résultats de vos expériences sur des courbes, en tenant compte du fait que vous avez le droit de présenter au maximum six courbes (là aussi, faites un travail de synthèse).

Faites signe aux encadrants si vous souhaitez avoir de l'aide sur les aspects techniques mis en jeu (vous trouverez aussi un petit exemple indicatif sur la page www du cours).

4 Et à la fin...

Les rendus c'est comme les contes, la fin ne varie pas[1] : référez-vous à la dernière partie du rendu 1 ("check list") pour vous assurer du fait que vous rendez ce qu'il faut, comme il faut.

[1] *Ils vécurent heureux et eurent beaucoup d'enfants.*