1 Introduction et méthode

Dans ce rapport, nous allons rapporter les performances de différentes versions des solveurs *SAT4J*, *MiniSAT* et *CryptoMiniSAT*. Pour ce faire, nous utiliserons la base de donnée de formules booléennes trouvable à cet url : `https://github.com/diverse-project/samplingfm/tree/master/Benchmarks`. Plus précisément, nous utiliserons uniquement les fichiers .cnf qui sont dans le répertoire "Benchmarks" de cette base de données.

Les solveurs testés sont :

- SAT4J - version 2.3.1 - JAR exécutable - paramètres par défaut
- ————————————
- MiniSAT - version 1.14.0 - binaire - paramètres par défaut
- MiniSAT - version 2.2.0 - binaire - -rnd-freq=0 (Défaut)
- MiniSAT - version 2.2.0 - binaire - -rnd-freq=0.5
- MiniSAT - version 2.2.0 - binaire - -rnd-freq=0.9
- ————————————
- CryptoMiniSAT - version 2.4.0 - binaire - paramètres par défaut
- CryptoMiniSAT - version 3.1.0 - binaire - paramètres par défaut
- CryptoMiniSAT - version 4.5.3 - binaire - paramètres par défaut
- CryptoMiniSAT - version 5.6.8 - binaire - –freq=0 (Défaut)
- CryptoMiniSAT - version 5.6.8 - binaire - –freq=0.5
- CryptoMiniSAT - version 5.6.8 - binaire - –freq=0.9

Afin de comparer ces solveurs, nous récupérons leur résultat et leur temps d'exécution sur toutes les formules du dossier "Benchmarks", avec un timeout à 10 minutes. Afin d'obtenir des résultats les moins bruités possible, nous effectuons ces calculs plusieurs fois et utilisons la moyenne.

Les tests sont lancés depuis un projet XText. Pour reproduire les données présentées dans ce rapport, il suffit d'exéctuer le fichier "org.xtext.example.msat.tests/src/org/xtext/example/msat/theos/Mein.xtend" avec JUnit. Les résultats seront stockés dans les fichiers "result_*i*.csv".

En tout les benchmarks ont durés 20 heures soit cinq itérations par tests.

2 Résultats

Les données brutes peuvent être trouvées dans le dossier "results", avec "results_*i*.csv" les données de chaque exécution de la base de données.

2.1 Aspect fonctionnel

Tous les solveurs ont renvoyé la même réponse pour toutes les formules présentes dans "Benchmarks".

A noter que dans les cas où les solveurs ont timeout il se pourrait qu'il existe en réalité un bug mais que les conditions de tests ne nous aient pas permis de le découvrir. Par exemple, pour la formule *listReverse.sk_11_43*, le solveur Sat4J a timeout systématiquement.

2.2 Efficacité

Afin de comparer l'efficacité, nous ne parlerons ici que du temps d'exéction, sans compter d'autres facteurs tels que l'espace utilisé.

Informations cool : - nombre de formules résolues - nombre de timeout (simple) - Temps moyen pour les formules (timeout = 15m) (?) - nombre de formules pour lequelles Cryptominisat a été strictement plus rapide que les deux autres. - nombre de formules pour lesquelles MiniSAT a été strictement plus rapide que les deux autres. - nombre de formules pour lesquelles SAT4J a été strictment plus rapide que les deux autres. - pour chaque solveur/version/parametres / combien de fois il a été le plus rapide.

3 Questions

3.1 Le "meilleurs" solveur ?

Voici le temps moyen de résolution pour toutes les formules :

On remarque par ailleurs que CryptoMiniSAT 4 n'a été battu que sur ??? formules. Et seulement de ???.

Les SAT-solveurs mordernes utilisent des heuristiques qui peuvent légérements varier et, pour des instances particulières, mener à des écars de performances plus ou moins importants. De fait, la question du "meilleurs" solveur n'a pas tant de sens sans ses guillemets. Cependant dans notre cas, au vu des information à notre disposition, la réponse n'appel pas particulièrement à la nuance :

Sur cet ensemble de tests, CryptoMiniSAT 4 domine largement le reste des solveurs. Il est d'ailleurs éttonant de constater que la version 4 a de bien meilleurs résultats que la version 5. Une explication possible est que la version 5 a été optimisé pour de très grandes instances et que notre timeout de 10 minutes (ainsi que notre ensembles de tests limité) ne permet pas de mettre en valeur ces cas de figures.

3.2 L'existance de bugs

Are there functional bugs in some solvers? An immediate approach is to control whether the solvers return the same result for the same formula (we hope so!). You can use the benchmark or generate random formulae Are there performance deviations of some solvers? Significant deviations may suggest performance-related bugs in some solvers. Is it the case?

Pour toutes les exécutions n'ayant pas timeout, les résultats des solveurs étaient cohérents entre eux.

Il est cependant probable que les erreurs auraient tendances à apparaitre pour les formules "difficiles". Il serait donc intéressant de refaire les tests avec un timeout plus grand.

3.3 La difficulté des formules

Are there harder SAT formulae?

3.4 De meilleurs benchmark ?

Can we reduce a benchmark and only consider a subset? (open question) Numerous formulae are used, but only a few lead to performance variations. Can't we only use a subse

3.5 Lol lol lol

Write a report that addresses the questions above: your answers should be supported by data (visualizations, statistics, tables, etc.). You should also include a technical description of your experiments (what solvers you use, what formulae, the measurements conditions, etc.), explain how to reproduce your results, and point out the data you have relied on. Push the report and all material in a subfolder (with a unique name) in the "reports" folder: repo.