# Proposition de TER L3

## Un outil pour gérer les expérimentations d'outils de déduction automatique

## Sujet

Les outils de déduction automatique sont des outils qui ont pour objectif de vérifier automatiquement la validité de formules logiques (en logique propositionnelle ou en logique du premier ordre). Lorsque l'on veut tester ces outils, on considère généralement un benchmark de problèmes, comme la bibliothèque TPTP [3] pour les outils au premier ordre ou SMT-LIB [1] pour les solveurs SMT.

Les tests de ces outils requièrent généralement une certaine paramétrisation (temps limite, limite mémoire, stratégies, etc.), qu'il faut pouvoir gérer. Par ailleurs, ces différents tests nécessitent de pouvoir les stocker et les documenter suffisamment pour se rappeler les conditions d'expérimentation de ces tests à des fins de comparaison.

Il faut également pouvoir exploiter les résultats de ces tests. En particulier, il faut pouvoir visualiser les résultats suivant un certain nombre de catégories identifiées au préalable (test réussi, échec pour dépassement de la limite de temps, échec pour dépassement de la limite mémoire, etc.). Il faut ensuite pouvoir comparer les résultats entre différents outils de déduction automatique. La comparaison peut concerner le nombre de problèmes prouvés, mais il peut aussi s'agir de critères plus complexes, comme le nombre de problèmes qu'un outil est le seul à prouver.

L'objectif de ce TER est de concevoir et de développer un outil permettant de gérer les expérimentations d'outils de déduction automatique. Dans ce travail, on peut identifier 3 étapes. La première étape consistera à réaliser une interface graphique pour la paramétrisation de l'expérimentation. Pour ce faire, une interface web pourrait parfaitement convenir. La deuxième étape consistera à lancer l'expérimentation proprement dite. Cette étape nécessite de lancer chaque outil sur tous les problèmes. On veillera à lancer les différents tests en parallèle suivant le nombre de cœurs disponibles sur la machine d'expérimentation. Enfin, la dernière étape se concentrera sur la visualisation des

résultats. Pour cette étape, une idée pourrait être de voir cela comme du monitoring du programme de test et d'utiliser un dashboard comme fourni par freeboard [2] par exemple.

Concernant les langages de programmation et les outils à utiliser pendant ce TER, rien n'est imposé. Il est à noter que les besoins en programmation sont très hétérogènes dans ce TER. Il y a de la programmation web, de la programmation concurrente, ainsi que de la manipulation de données. Il y aura donc plusieurs langages et outils différents à utiliser au cours de ce TER et il faudra utiliser ceux qui semblent les plus appropriés pour accomplir les tâches à réaliser.

## Travail à réaliser

— Réaliser une interface graphique pour paramétrer l'expérimentation consistant à lancer des outils de déduction automatique sur un benchmark de problèmes ;
— Développer un outil qui lance en parallèle les différents outils de déduction automatique considérés sur le benchmark de problèmes ;
— Implanter une interface de visualisation des résultats de l'expérimentation, qui permet de comparer les différents outils de déduction automatique considérés.

## Prérequis

— Aucun prérequis en logique n'est nécessaire. Les outils de déduction automatique seront en effet utilisés comme des boîtes noires.
— Aucun prérequis en programmation n'est nécessaire, même si avoir vu de la programmation web et/ou de la programmation concurrente pourrait être un plus.

## Remarques additionnelles

L'encadrement du TER sera réalisé par :
— David Delahaye (Université de Montpellier, LIRMM, David.Delahaye@lirmm.fr).

## Références

[1] C. Barrett, P. Fontaine, and C. Tinelli. The Satisfiability Modulo Theories Library (SMT-LIB). http://www.SMT-LIB.org, 2016. Accessed on November 25, 2019.

[2] Bug Labs, Inc. freeboard. https://freeboard.io/, 2019. Accessed on November 25, 2019.

[3] G. Sutcliffe. The TPTP Problem Library and Associated Infrastructure : The FOF and CNF Parts, v3.5.0. *Journal of Automated Reasoning (JAR)*, 43(4) :337–362, Dec. 2009.