Sujet de stage L3 ENS
année 2013 - 2014

Didier Galmiche - Dominique Larchey-Wendling
Équipe TYPES, LORIA UMR 7503

Titre : Recherche de preuves compilée et certifiée

Thème. La logique intuitionniste (IL) est l'une des logiques non-classiques les plus importantes en philosophie, en logique et en informatique. On peut l'obtenir en supprimant l'axiome du tiers exclu $A \vee \neg A$ à la logique classique (CL). La logique intuitionniste IL fonde les mathématiques constructives. Elle offre un cadre naturel à l'isomorphisme de Curry-Howard qui fait correspondre preuves et programmes, formules et types de données. Elle constitue la logique de base de l'outil de preuve Coq.[1]

La recherche automatisée de preuves en logique intuitionniste propositionnelle (ILP) est un thème de recherche mature. Nous nous intéressons ici aux techniques fondées sur les calculs à base de séquents. Il existe toutefois d'autres techniques comme par exemple la méthode des tableaux sémantiques ou la méthode de connexions.

Contrairement au cas de la logique classique propositionnelle CL, le calcul des séquents de Gentzen LJ n'est pas terminant pour IL : une application naïve des règles peut conduire à des boucles dans la recherche de preuves. Toutefois, des versions raffinées de ce calcul ont été introduites dans les années 90 par Hudelmair [4] et Dyckhoff [1]. Le système LJT est un calcul des séquents dans lequel les règles vérifient la propriété de simplification : les prémisses sont strictement moins complexes que la conclusion (pour une certaine mesure de la complexité). Ainsi, n'importe quelle recherche de preuves termine, quelle que soit l'ordre choisi pour appliquer les règles du calcul. Toutefois, contrairement à LJ, LJT ne satisfait pas la propriété de la sous-formule : toute formule qui engendrée au cours de la recherche de preuves est une sous-formule de la formule initiale. Grâce à la propriété de la sous-formule, l'indexation permet d'optimiser fortement la recherche de preuves.

STRIP[2] est un prouveur automatisé qui a été développé dans l'équipe TYPES [3, 5]. Basé sur une variante de LJT qui permet le partage par une structure de données adaptée, il offre la possibilité de décider si une formule logique de IL est valide ou non, et sur demande, de produire soit une preuve de cette formule, soit un contre-modèle sous la forme d'un arbre de Kripke. STRIP intègre une indexation ce qui le rend relativement efficace mais avec l'inconvénient relatif de manipuler des structures de données sophistiquées, ce qui rend sa certification formelle dissuasive.

Des travaux très récents [2] ont conduit à un calcul des séquents pour IL appelé LSJ qui possède à la fois la propriété de la sous-formule et la propriété de simplification. Nous pensons que ce système peut conduire à une amélioration de la recherche de preuves automatisée dans IL : en effet les règles de ce calcul peuvent s'implanter efficacement sans utiliser de structures de données plus sophistiquées que des listes ou des ensembles vus comme des tableaux de Booléens.

Nous avons prouvé en Coq la correction formelle de plusieurs systèmes pour la recherche de preuve en logique intuitionniste, entre autres les systèmes LJT et LSJ mais aussi LJ et NJ (déduction naturelle). Toutefois, prouver la correction d'un système formel n'est pas la même chose que de prouver la correction d'une implantation (si possible efficace) de ce système.

[1] `http://coq.inria.fr/`
[2] `http://www.loria.fr/~larchey/STRIP/`

Sujet. Afin d'aborder ce problème de la correction des implantations efficaces de la recherche de preuves, nous proposons d'abstraire la manipulation des structures de données rencontrées dans la recherche de preuves au sein d'une machine abstraite spécifique dont on définira formellement la sémantique opérationnelle. Pour fixer les idées, on pourra commencer par le cas simple de la logique classique propositionnelle avant de passer aux calculs des séquents plus complexes de la logique intuitionniste.

Puis nous aborderons la question de la compilation de la recherche de preuve, c'est-à-dire, transformer une formule donnée en un programme (en langage fonctionnel spécifique), compiler ce programme pour une exécution sur la machine abstraite préalablement définie puis exécution de ce programme, cette dernière reproduisant le processus de recherche de preuves. L'objectif ultime étant de certifier formellement (en Coq par exemple) la correction totale du compilateur : l'exécution du code correspondant à une formule donnée termine toujours sur un résultat Booléen $\{\texttt{true}, \texttt{false}\}$ et ce résultat vaut `true` quand la formule est prouvable alors qu'il vaut `false` quand elle ne l'est pas.

Renseignements. Le stage aura lieu au sein de l'équipe TYPES du LORIA à Nancy. Pour tout renseignement complémentaire sur ce sujet, contacter D. Larchey-Wendling ou D. Galmiche (e-mail : larchey@loria.fr, galmiche@loria.fr). Une description plus précise des activités scientifiques et des résultats de l'équipe est accessible via `http://www.loria.fr/~galmiche` et `http://www.loria.fr/~larchey`).

Références

[1] Roy Dyckhoff. Contraction-free Sequent Calculi for Intuitionistic Logic. *Journal of Symbolic Logic*, 57(3) :795–807, 1992.

[2] Mauro Ferrari, Camillo Fiorentini, and Guido Fiorino. Linear Depth Sequent Calculi for Intuitionistic Propositional Logic with the Subformula Property and Minimal Depth Counter-Models. Submitted 2011.

[3] Didier Galmiche and Dominique Larchey-Wendling. Structural sharing and efficient proof-search in propositional intuitionistic logic. In P. S. Thiagarajan and Roland H. C. Yap, editors, *ASIAN*, volume 1742 of *Lecture Notes in Computer Science*, pages 101–112. Springer, 1999.

[4] Jörg Hudelmaier. An $\mathcal{O}(n \log n)$-space decision procedure for Intuitionistic Propositional Logic. *Journal of Logic and Computation*, 3(1) :63–75, 1993.

[5] Dominique Larchey-Wendling, Dominique Méry, and Didier Galmiche. Strip : Structural sharing for efficient proof-search. In Rajeev Goré, Alexander Leitsch, and Tobias Nipkow, editors, *IJCAR*, volume 2083 of *Lecture Notes in Computer Science*, pages 696–700. Springer, 2001.