Proposition de TER L3

Une interface graphique pour réaliser des preuves en logique classique

Sujet

En logique, pour valider une formule, il suffit d'utiliser la sémantique de la logique pour voir si la formule est toujours vraie. En logique propositionnelle (sans quantificateurs) par exemple, cela revient à faire la table de vérité correspondant à la formule et à vérifier que pour toutes les combinaisons de valeurs pour les variables propositionnelles, la formule rend vrai. Cette méthode, en plus d'être peu efficace (surtout si on l'implante de manière naïve), est plutôt complexe, surtout lorsqu'on passe à la logique du premier ordre (avec quantificateurs). Une méthode alternative pour valider une formule est de faire une preuve.

Une preuve est un moyen purement syntaxique pour vérifier la validité d'une formule. Ainsi, lors d'une preuve, on n'effectue plus aucune référence à la sémantique et seule la forme de la formule est importante pour savoir quelle règle de preuve on peut appliquer. Pour construire une preuve, on applique des règles de preuve sur des formules en construisant un arbre dont les feuilles sont des règles axiomatiques (terminales). Si un tel arbre peut être construit, alors il constitue une preuve de la formule initiale, qui est donc valide. Par exemple, si on veut démontrer que $A \Rightarrow B \Rightarrow A \wedge B$, la preuve dans le calcul des séquents classique (appelé système LK) est la suivante :

$$\dfrac{\dfrac{\dfrac{\dfrac{}{A, B \vdash A}\,\mathsf{ax} \quad \dfrac{}{A, B \vdash B}\,\mathsf{ax}}{A, B \vdash A \wedge B}\,\wedge_{\mathsf{right}}}{A \vdash B \Rightarrow A \wedge B}\,\Rightarrow_{\mathsf{right}}}{\vdash A \Rightarrow B \Rightarrow A \wedge B}\,\Rightarrow_{\mathsf{right}}$$

Où l'on voit en particulier deux branches apparaître.

Le principal souci de cette représentation graphique des preuves est que même pour des preuves relativement modestes, les arbres de preuve peuvent être de grande taille et deviennent donc irréalisables manuellement sur papier. L'objectif de ce TER est de développer un outil permettant de réaliser des preuves dans cette représentation graphique (dans le système LK en particulier).

Dans ce TER, on ne partira pas de rien, le moteur de preuve est déjà partiellement écrit (en Java) pour la logique propositionnelle et une interface graphique, partielle également, a été écrite (en Java aussi) pour ce moteur. Dans un premier temps, il faudra donc compléter le code pour qu'il fonctionne pour la logique propositionnelle. Ensuite, dans un deuxième temps, il faudra étendre le code à la logique du premier ordre.

Dans ce projet, il faudra se concentrer non seulement sur la partie interface graphique, mais aussi sur la partie moteur de preuve, pour lequel il faudra sans doute prévoir des fonctionnalités supplémentaires. Parmi les fonctionnalités souhaitées, on voudra pouvoir construire l'arbre de preuve incrémentalement en proposant à l'utilisateur les règles de preuve applicables selon la formule sélectionnée. Lorsqu'un arbre de preuve est complet, on voudra aussi pouvoir le sauvegarder dans un format à déterminer et le recharger également éventuellement plus tard. Des fonctionnalités permettant de défaire/refaire des étapes de preuve seront aussi à prévoir.

Quant au langage à utiliser pour le développement, ce sera Java, afin que ce soit homogène avec le développement déjà effectué. Par ailleurs, Java a l'avantage de proposer une bibliothèque riche pour écrire des interfaces graphiques.

Travail à réaliser

— Compléter le code existant pour réaliser une interface graphique pour représenter les preuves en logique propositionnelle dans le calcul des séquents classique ;
— Étendre le code pour représenter les preuves en logique du premier ordre dans le calcul des séquents classique.

Prérequis

— Aucun prérequis en logique n'est nécessaire. Avoir suivi l'UE « Logique 1 » de L2 peut être un plus, mais tout le bagage logique sera (ré)introduit pendant le TER.
— Les étudiants devront avoir un goût prononcé pour le développement d'interfaces graphiques et avoir vu Java, qui facilite le développement de telles interfaces.

Remarques additionnelles

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