Université
catholique de
Louvain

Syllabus du cours

LINGI1101: Logique et Structures Discrètes

Titulaire :
Peter Van Roy

2014

Table des matières

Remerciements

Je tiens à remercier les étudiants de LINGI1101 pour avoir pris des notes pendant mon cours, ce qui faisait la base de ce syllabus.

Introduction

Ce document est le syllabus du cours LINGI1101 “Logique et Structures Discrètes” donné par Peter Van Roy.

1 Introduction à la programmation logique

1.1 Introduction à la programmation logique

Prolog est l'un des principaux langages de programmation logique. Il est à la base de beaucoup de chose.

La programmation logique fait de la déduction sur les axiomes. On utilise la logique comme un langage de programmation : on va adapter l'algorithme de réfutation qu'on a vu précédemment

Le programme (ressemble à une théorie) :
- Axiomes en logique des prédicats
- Une requête, un but (=goal) -> le but du système est de prouver quelque chose
- Un prouveur de théorème -> attention : il faut des conditions sur le prouveur car on doit être capable de prévoir le temps et l'espace utilisé par le programme.

Exécuter un programme = faire des déductions en essayant de prouver le but. Mais est-ce que cette idée peut donner un système de programmation pratique ?

Il y a une tension entre expressivité et efficacité : Si c'est trop expressif, ça devient moins efficace, par contre si c'est trop peu expressif, on ne peut rien programmer, ça ne sert à rien non plus. Il faut donc être expressif tout en restant efficace. Le Prolog offre un bon mélange entre expressivité et efficacité.

Mais pour arriver à cela, il y a quelques problèmes à surmonter :

1. un prouveur est limité vérité = $p \models q$ (= q est vrai dans tous les modèles de p)
   preuve = $p \vdash q$
   $p \models q \Rightarrow p \vdash q$ (= Si c'est vrai dans tous les modèles, on peut trouver une preuve)
   Si $p \models q$ alors l'algorithme se terminera. Cependant, on ne peut pas trouver les preuves pour des choses vraies dans tous les modèles. (Comme c'est impossible on ne prend qu'une partie des modèles. Ceci est une limitation du programme).
2. b) Même si on peut trouver une preuve, le prouveur est peut-être inefficace (utilise trop de temps ou de mémoire) ou imprévisible. → On ne peut pas raisonner sur l'efficacité du prouveur.
3. c) La déduction faite par le prouveur doit être constructive

Si le prouveur dit : $(\exists X)P(X)$ alors le prouveur doit donner une valeur de x (c'est quoi x).

Conclusion

Pour conclure, avec LaTeX on obtient un rendu impeccable mais il faut s'investir pour le prendre en main.

Références

[Nis] Nimal Nissanke. *Introductory Logic and Sets for Computer Scientists.*

[LPP] David Easley and Jon Kleinberg. *Networks, Crowds, and Markets : Reasoning About a Highly Connected World.*