Nantes Université

PROJET

PROGRAMMATION FONCTIONNELLE

2022-2023

LICENCE 3 - INFORMATIQUE

ATALLA SALIM - 191350P
NISO ARAM - 184256D

GROUPE 684J

LAMBDA-CALCUL

Le lambda calcul, inventé par Alonzo Church en 1932, est un modèle mathématique de calcul largement utilisé en informatique théorique et en théorie des langages de programmation. Ce modèle permet de représenter des programmes et des algorithmes, ainsi que de prouver des théorèmes en utilisant des fonctions anonymes qui prennent des arguments et retournent des valeurs. Grâce à sa capacité à représenter n'importe quel calcul qu'un ordinateur peut effectuer, le lambda calcul est considéré comme un système de calcul universel.
Dans ce rapport, nous allons explorer en détail les concepts et opérations du lambda calcul en utilisant le langage de programmation OCaml pour l'implémentation.

TYPAGE:

Nous allons commencer par introduire le type lambda_expr, qui représente une expression du lambda-calcul. Ce type contient trois constructeurs de données :

- Var : qui représente une variable
- Abs : qui représente une abstraction
- App : qui représente l'application d'une expression à une autre.

EXPRESSIONS:

Nous allons définir les nombres de Church, qui sont des fonctions servant à représenter les nombres entiers. De plus, nous allons introduire les valeurs booléennes true_val et false_val ainsi que les opérations booléennes not_op, and_op et or_op dans le lambda calcul.

FONCTIONS:

Nous allons maintenant explorer plusieurs opérations fondamentales du lambda calcul. Tout d'abord, la substitution est une opération clé qui permet de remplacer toutes les occurrences d'une variable par une autre expression dans une expression donnée. La fonction substitute prend en

entrée une expression expr, une variable var et une nouvelle expression new_expr, et retourne l'expression obtenue en substituant toutes les occurrences de var dans expr par new_expr.

Ensuite, nous allons définir la fonction free_vars, qui permet de trouver toutes les variables libres présentes dans une expression donnée. En entrée, cette fonction prend une expression expr et retourne une liste contenant toutes les variables libres de l'expression.

Nous allons également introduire la notion de variable fraîche, qui est une variable qui n'a pas encore été utilisée dans l'expression. Cette notion sera utilisée pour définir la fonction alpha_conversion, qui permet de renommer une variable dans une expression et d'effectuer une conversion α. Cette fonction prend en entrée une expression expr, une variable old_var à remplacer par une variable new_var, et retourne l'expression obtenue.

Enfin, la beta réduction est l'opération centrale du lambda calcul. Elle permet de simplifier une expression en appliquant une substitution. La fonction beta_reduction prend en entrée une expression expr et effectue une réduction β.

Dans le lambda calcul, la réduction beta est l'une des deux réductions fondamentales. Elle permet de transformer une expression en remplaçant une abstraction appliquée à une expression par une nouvelle expression. Plus précisément, lorsqu'une abstraction de la forme (λx.e1) est appliquée à une expression e2, la réduction beta remplace toutes les occurrences de la variable x dans e1 par e2.

La fonction beta_reduction implémente cette réduction beta en parcourant récursivement l'expression donnée en entrée et en appliquant les règles de réduction beta lorsque cela est possible. La première règle de réduction beta stipule que si l'on a une expression de la forme (λx.e1)e2, alors toutes les occurrences de la variable x dans e1 peuvent être remplacées par e2. Pour effectuer cette substitution, la fonction substitute prend en entrée l'expression e1, la variable v à remplacer par new_expr, et renvoie la nouvelle expression obtenue.

La deuxième règle de réduction beta stipule que si l'on a une expression de la forme e1 e2, alors e1 et e2 peuvent être réduits récursivement en appliquant les règles de réduction beta. Pour cela, la fonction beta_reduction applique la réduction beta à e1 et e2 en appelant récursivement la fonction beta_reduction, puis renvoie l'application de ces deux expressions réduites.

En résumé, le lambda calcul est un formalisme mathématique utilisé pour décrire les fonctions. Il repose sur des termes, qui peuvent prendre la forme de variables, d'abstractions ou d'applications de termes. Les fonctions sont définies à l'aide d'abstractions et sont évaluées en utilisant deux réductions fondamentales : la réduction alpha, qui permet de renommer les variables liées, et la réduction beta, qui permet de réduire une abstraction appliquée à une expression en une nouvelle expression. Les fonctions dans l'exemple fourni utilisent ces réductions pour définir des valeurs de base du lambda calcul, telles que les nombres de Church et les valeurs booléennes, ainsi que pour effectuer des opérations sur ces valeurs. En somme, le lambda calcul est un outil précieux pour décrire et manipuler les fonctions dans le domaine mathématique.

———————————————————————————