SORBONNE UNIVERSITÉ

TD : Machines virtuelles — semaine 2

15 février 2018

Objectif(s)

- ✯ Étude de la machine de Krivine.
- ✯ Préparation au TME avec la machine universelle.

1 La machine de Krivine

La machine de Krivine est une machine pour l'évaluation paresseuse des λ-termes. Dans cet exercice, nous allons examiner son fonctionnement et l'étendre pour adjoindre au langage les déclarations `let` x `=` $expr$ `in` $expr$, et `let rec` x `=` $expr$ `in` $expr$.

La machine de Krivine est composée de trois éléments distincts (env, $terme$ et $pile$) :

— Un environnement, qui contient une liste de couples $(terme, env)$. C'est à dire des termes calculables et les valeurs de ses variables liées.
— Un segment de termes, qui contient le terme à évaluer.
— Une pile, contenant elle aussi des couples $(terme, env)$ servant à stocker des termes non encore évalués.

Évaluation d'un λ terme

Le schéma d'évaluation d'un λ-terme est le suivant :

$avant$			$apres$		
env	$terme$	$pile$	env	$terme$	$pile$
e	c	S	e	c	S
e	MN	S	e	M	$(N, e) :: S$
e	$\lambda x.M$	$s :: S$	$s_x.e$	M	S
$(M, e_2)_x.e$	x	S	e_2	M	S
$(M, e_2)_y.e$	x	S	e	x	S

Les règles sont respectivement appelées :

— le none,
— l'application d'un terme,
— l'abstraction d'un lambda terme,
— la liaison de l'argument courant avec l'indice de de Bruijn 0 (appelée zéro), et
— la recherche de cette liaison dans l'indice de de Bruijn suivant (appelée succ).

Pour distinguer les listes de la pile et de l'environnement, nous noterons $s :: S$ la première et $e.E$ la seconde. Finalement, e_x indique le nom de variable lié à un terme de l'environnement.

Exercice 1 – Évaluation de termes

1. Évaluer les λ-termes suivants en utilisant les règles ci-dessus :
   — $\lambda x.\lambda y.xy$ avec pour valeurs initiales dans la pile 1 et 2.
   — $(\lambda x.xx)(\lambda y.yy)$.
2. Écrire les règles d'évaluation pour les instructions `let` x `=` $expr$ `in` $expr$, et `let rec` x `=` $expr$ `in` $expr$.

Les instructions de la machine

La machine de Krivine dispose de 4 instructions :
- `Grab` : mettre le sommet de la pile dans l'environnement
- `Push` *`label`* : mettre un label et l'environnement dans la pile.
- `Access` *`n`* : accéder à la n^{eme} variable dans l'environnement.
- `Label` *`label`* : étiqueter un terme avec le label *label*

Exercice 2 – Ajout d'instructions

Ajouter deux nouvelles instructions à la machine de Krivine, `Let` et `Rec`, permettant de compiler les instructions `let` x `=` $expr$ `in` $expr$, et `let rec` x `=` $expr$ `in` $expr$.

On notera $[M]_\rho$ la compilation du terme M dans l'environnement ρ. Avec ces instructions on compilera un λ-terme comme suit :
- $[MN]_\rho =$ `Push l;`$[M]_\rho$`; Label l ;` $[N]_\rho$
- $[\lambda x.M]_\rho =$ `Grab;`$[M]_{\rho,x}$.
- $[x]_\rho =$ `Access n` (où n est la position de x dans l'environnement ρ)

Exercice 3 – Schéma de compilation

Décrire le schéma de compilation des nouvelles constructions `let` et `let rec` avec `Rec` et `Let`.

Schéma d'évaluation

Étant donné un λ-terme compilé en instruction pour la machine de Krivine, on obtient l'évaluation de celui ci ainsi :

$avant$			$apres$		
env	$terme$	$pile$	env	$terme$	$pile$
e	$Push\ l; C$	S	e	C	$(l, e) :: S$
e	$Grab; C$	$s :: S$	$s.e$	C	S
e	$(*)Grab; C$	$()$	Arrêt de l'execution		
$(l, e_2).e$	$Access\ 0; C; Label\ l; C_2$	S	e_2	C_2	S

Un des points important est le critère d'arrêt (troisième ligne). L'instruction Grab, correspond à une abstraction. Si la pile est vide, cette abstraction ne trouve pas d'argument et le calcul s'arrête.

Exercice 4 – Schéma d'évaluation

Donner le schéma d'évaluation des nouvelles instructions.

Exercice 5 – Constantes entières et booléennes

Étendre le lambda calcul traité par la machine en y rajoutant la gestion des constantes. Rajouter l'instruction `Pushenv` à la machine de Krivine permettant d'affecter une variable dans l'environnement. Donner les schémas de compilations afférents.

2 La machine universelle

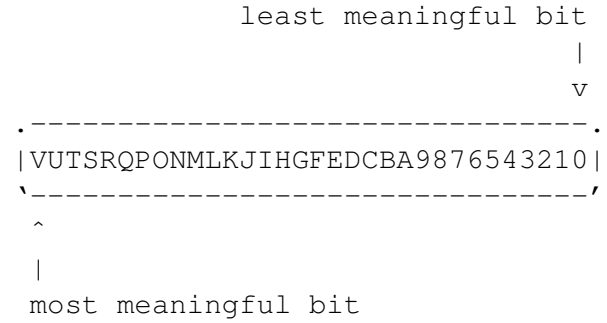
Introduction

Cette partie est inspirée d'un sujet de la conférence *ACM International Conference on Functional Programming* (ICFP) de 2006. La version originale du sujet peut se trouver via le lien : `http://www.boundvariable.org/task.shtml`.

La machine

On nous demande de créer la machine universelle qui est composée de :

— Une quantité infinie de plateaux de sable avec des cases pour 32 marques (en langage plus usuel : 32 *bits*). Un plateau ressemble à :



Chaque bit peut prendre les valeurs 0 ou 1. En utilisant le système de numération 32 bits, ces marques peuvent aussi encoder des entiers.

— 8 registres a usage général capables chacun de contenir un plateau.

— Une collection de tableaux de plateaux, chacun étant référencé par un identificateur 32 bits. On distinguera le tableau 0 des autres, il contiendra le programme de la machine. Ce tableau sera référencé comme étant le tableau '0'.

— Une console de 1 caractère capable d'afficher les glyphes du code ASCII et faisant l'input et l'output de `unsigned 8-bit characters`.

Comportement

La machine sera initialisée avec un tableau '0' dont le contenu sera lu depuis le parchemin de programme. Tous les registres seront initialisés à '0'. L'indice d'exécution pointera sur le premier plateau de sable qui devra avoir l'indice zéro.

Quand on lit des programmes depuis des parchemins codés sur 8 bits, une série de 4 bytes (A,B,C,D) devra être interprétée comme suit :

— A sera le byte de poids fort.

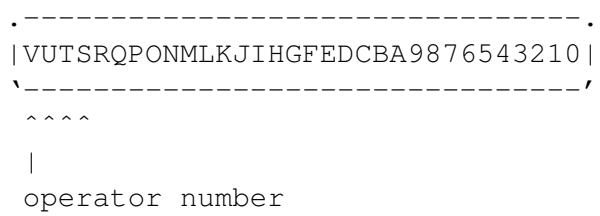
— D le byte de poids faible.

— B et C seront intercalés dans cet ordre.

Une fois initialisée, la machine débute son cycle. À chaque cycle de la machine universelle, un opérateur devra être lu depuis le plateau indiqué par l'indice d'exécution. Les sections ci-après décriront les opérateurs que l'on peut récupérer. Avant que cet opérateur ne soit déchargé, l'indice d'exécution devra être avancé jusqu'au plateau suivant (s'il existe).

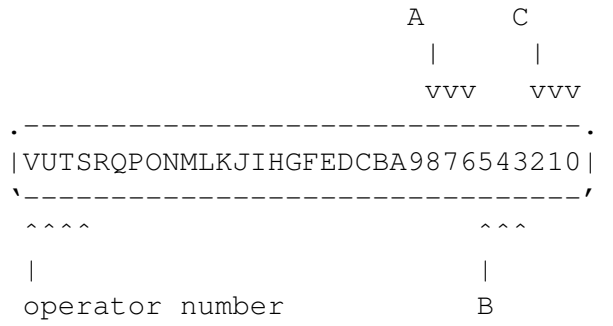
Opérateurs

La machine universelle dispose de 14 opérateurs. Le numéro des opérateurs est décrit par les 4 bit de poids fort du plateau d'instructions.



Opérateurs standards

Chaque opérateur standard effectue un calcul en utilisant trois registres nommés : A,B et C. Chaque registre est décrit par un segment de 3 bits du plateau d'instruction. Le registre C est décrit par les 3 bits de poids faibles du plateau, le registre B par les 3 suivant et le registre A se trouve à la suite de B.



Les opérateurs standards sont les suivants :

Opérateur 0 Mouvement conditionnel
: Le registre A reçoit la valeur du registre B sauf si le registre C contient 0.

Opérateur 1 Indice de tableau
: Le registre A reçoit la valeur à l'offset contenu dans le registre C dans le tableau identifié par registre B.

Opérateur 2 Modification de tableau
: Le tableau identifié par le registre A est modifié à l'offset B par la valeur contenu dans le registre C.

Opérateur 3 Addition
: Le registre A reçoit la valeur contenu dans le registre B plus la valeur contenu dans le registre C modulo 2^{32}.

Opérateur 4 Multiplication
: Le registre A reçoit la valeur du registre B multipliée part la valeur du registre C modulo 2^{32}.

Opérateur 5 Division
: Le registre A reçoit la valeur du registre B divisée par la valeur du registre C.

Opérateur 6 Not-And
: Le registre A reçoit le `NAND` des registres B et C.

Autres opérateurs

D'autre opérateurs sont disponibles :

Opérateur 7 Stop
: Arrête la machine universelle.

Opérateur 8 Allocation
: Un nouveau tableau est créé avec une capacité égale à la valeur du registre C. Ce nouveau tableau est initialisé entièrement avec des plateaux à 0. Le registre B reçoit l'identificateur du nouveau tableau.

Opérateur 9 Abandon
: Le tableau identifié par le registre C est abandonné et une future allocation peut réutiliser son identificateur.

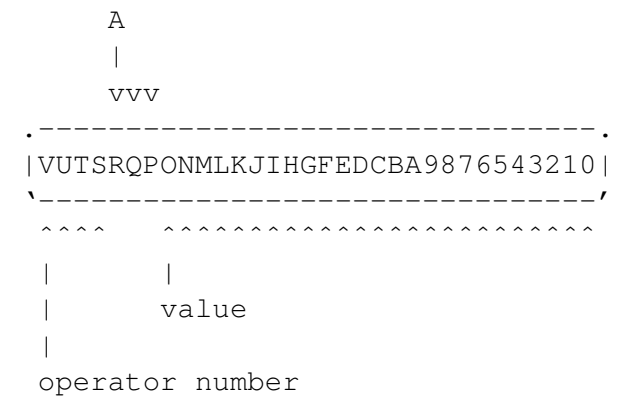
Opérateur 10 Sortie
: Écrit sur la console la valeur du registre C, seules des valeurs comprises entre 0 et 255 sont tolérées.

Opérateur 11 Entrée
: La machine universelle attend une entrée sur la console. Quand une donnée arrive, le registre C est chargé avec :
  — Tous ses bits positionnés à 1, si la fin de l'entrée est signalée.
  — La valeur de cette entrée sinon.

Opérateur 12 Chargement de programme
: Le tableau dont l'identificateur est B est dupliqué et sa copie remplace le tableau '0', quelle que soit sa taille. L'indice d'exécution doit être placé sur le plateau dont l'indice est contenu par le registre C.

Opérateurs spéciaux

Pour les opérateurs spéciaux, les registres utilisés ne sont pas décrits de la même façon. Les trois bits immédiatement après ceux décrivant l'opérateur donnent le numéro du registre à utiliser. Les 25 bits restant donnent une valeur qui devra être chargée dans le registre A.



Opérateur 13 Orthographe
La valeur indiquée doit être chargée dans le registre A.

Mesures de réduction de couts

Tout comportement non décrit par le schéma précédent pourra mettre la machine en panne.