MPRI 2-27-1 Automne 2007

Gérard Huet & Philippe de Groote

Examen du 22-11-07

1 Machines d'état fini

Les notes de cours expliquent la notion de machine d'Eilenberg, et comment de telles machines peuvent servir à reconnaître les langages rationnels. Rappeler brièvement la définition de machine d'Eilenberg, et expliquer pourquoi un langage rationnel peut être reconnu par certaines de ces machines.

Ces machines peuvent aussi reconnaître des langages non réguliers, comme les langages de Dick. Ces langages engendrent tous les langages algébriques ou hors-contexte par homomorphisme. Rappeler le théorème exact.

Par exemple, on peut résoudre le problème suivant. On veut vérifier qu'un texte comportant des parenthèses ouvrantes '(' et des parenthèses fermantes ')' est bien parenthésé. A part ces caractères, toutes les lettres d'un alphabet Sigma sont autorisées. Décrire la machine d'Eilenberg comme un automate à compteur qui reconnaît les mots bien parenthésés. Vérifier que c'est une machine d'Eilenberg finie, en explicitant toutes les conditions.

Transformer le compteur en pile d'entiers pour permettre des paires de parenthèses () [] {} etc. arbitraires. Au vu de la construction de machines d'Eilenberg modulaires, et en considérant le théorème pré-cité, on doit en conclure que ces machines à piles d'entier doivent suffire à reconnaître les langages algébriques (context-free).

On revient à la machine comptant une race de parenthèses. On aimerait intégrer cette machine dans un traitement de textes, Word ou Emacs ou autre. Mais plutôt que d'avoir un booléen permettant de signaler "Erreur de parenthèsage", on aimerait un mécanisme plus malin, qui non seulement s'arrête à la première parenthèse fautive - ou à la fin s'arrête sans fermeture d'une parenthèse ouvrante la plus englobante - donne cette information à l'utilisateur, ou mieux le prépare à entrer dans une phase d'édition structurée de son buffer pour corriger cette anomalie. Tout utilisateur de Word sait que ce logiciel a précisément cette fonction, notamment pour éditer sous forme arborescente un document structuré de manière hiérarchique. Expliquer comment utiliser la structure de zipper pour effectuer commodément ces opérations par coopération de machines d'Eilenberg appropriées.

Maintenant que vous êtes convaincus que les langages algébriques sont reconnaissables par des machines d'Eilenberg finies, expliquer comment écrire un analyseur non déterministe de ces langages comme machine d'Eilenberg finie. On tâchera de faire le lien avec le cours sur les structures syntaxiques, notamment les algorithmes CKY et Earley.

Donner une algèbre relationnelle définissant les opérations de base d'une machine de Turing vue comme calculant sur une donnée qui est la valeur de sa bande au point de focus de sa tête de lecture. Cette donnée peut être représentée par un zipper de liste de caractères. Le programme de la machine de Turing est le graphe de transition de la machine d'Eilenberg correspondante. Essayer de réduire le nombre d'opérations permises tout en restant complet au sens de Turing. Cette machine d'Eilenberg est-elle finie ?

Rappeler la définition de machine de Turing universelle. Esquisser la construction d'une telle machine en utilisant l'étude précédente.

2 Logique dynamique

Soit le λ-calcul simplement typé construit sur une signature comprenant les connecteurs et quantificateurs de la logique classique du premier ordre et des opérations de mise à jour et d'accès aux contextes:

$$\begin{aligned} \iota, o, \gamma &: \text{type};\\ \top &: o;\\ \neg &: o \to o;\\ \wedge &: o \to (o \to o);\\ \exists &: (\iota \to o) \to o;\\ \forall &: (\iota \to o) \to o;\\ - :: - &: \iota \to (\gamma \to \gamma);\\ \mathsf{sel} &: \gamma \to \iota. \end{aligned}$$

Le type des propositions dynamiques est défini comme suit:

$$\Omega = \gamma \to ((\gamma \to o) \to o)$$

On définit alors une négation et des quantificateurs dynamiques:

$$\begin{aligned} \sim A &= \lambda e \phi.\, \neg\, (A\, e\, (\lambda e.\, \top)) \wedge \phi\, e,\\ \Sigma x.\, P\, x &= \lambda e \phi.\, \exists x.\, P\, x\, (x{::}e)\, \phi,\\ \Pi x.\, P\, x &= \lambda e \phi.\, (\forall x.\, P\, x\, (x{::}e)\, (\lambda e.\, \top)) \wedge \phi\, e \end{aligned}$$

Etablir l'équivalence suivante:

$$\Pi x.\, P\, x \;\equiv \sim (\Sigma x.\; \sim (P\, x))$$