Projet de Syntaxe et Sémantique. (bac-3-hd, ap2c-hd)

Xavier Devroey Alain Solheid Michaël Marcozzi

2011 - 2012

Table des matières

---

1 Présentation générale

L'énoncé du travail se compose de deux documents. Celui que vous tenez dans les mains qui va tenter de poser de façon précise la réalisation qui vous est demandée et le document intitulé "LSD^{12} : Et ça repart" qui spécifie le langage que vous allez considérer.

Votre tâche consiste à réaliser un compilateur LSD^{12}pour la P-machine. (disponible à l'adresse `http://www.info.fundp.ac.be/~gpm/`).

La suite du présent énoncé a pour objectif de vous proposer quelques balises d'organisation de votre travail et de vous éclairer sur les critères d'évaluation.

Remarque importante. Les deux documents mentionnés décrivent l'entièreté du projet. **Ne vous effrayez pas s'ils vous semblent inabordables en première lecture !** Certains éléments de la description ne prendront sens pour vous que lorsqu'ils auront été abordés au cours théorique ou en séance de tp. N'attendez pas d'avoir tout compris pour vous y mettre. Pour la façon de gérer votre travail rendez-vous à la section "organisation et évaluation"

2 La démarche

Etant donné un programme, l'action du compilateur peut être divisée en trois étapes :
- analyse lexicale
- analyse syntaxique
- génération de code

La troisième étape n'étant effectuée que si le programme s'est révélé syntaxiquement correct.

2.1 L'analyse lexicale

Il s'agit, à l'aide de flex, d'écrire une spécification qui
- reconnait tous les mots clés du langage,
- reconnait les ponctuations utilisées dans le langage,
- reconnait tout les type de constantes,
- reconnait les identificateurs,....

2.2 L'analyse syntaxique

Pour le langage LSD^{12}, nous vous demandons d'écrire une spécification bison qui reconnait les programmes du langage tout en créant l'arbre de parsing correspondant. De plus, il gèrera la table des symboles associant à chacun son type.Le code nécessaire à cet effet sera écrit en C-ANSI.

L'analyse lexicale se fera au moyen de l'analyseur lexical décrit plus haut (flex).

L'écriture des spécifications bison peut vous demander de résoudre des conflits shift/reduce ou reduce/reduce.

Etant donné un programme à l'entrée standard, votre analyseur syntaxique fournira "OK\n" à l'erreur standard si et seulement si le programme est conforme aux spécifications de LSD^{12}(y compris la sémantique statique : vérification de type, doubles déclarations,...). Si le programme est non conforme aux spécifications de LSD^{12}, votre analyseur syntaxique fournira un message commençant par "KO\n" à l'erreur standard

Petits tuyaux.
- L'écriture de "OK" sur l'erreur standard s'écrit à l'aide de la commande : `fprintf(stderr,''OK\n'')`
- Si le programme n'est pas correct, vous pouvez faire suivre le "KO\n" à l'erreur standard par tous les messages d'erreurs de votre choix.

2.3 La génération de code

Dans le cas où votre analyseur syntaxique a accepté le programme en entrée, vous devez ensuite générer, sur la sortie standard, le P-code correspondant au programme.

3 Organisation et évaluation

3.1 Evaluation

Le travail, à réaliser par groupe de trois, sera évalué sur une quadruple base :
- votre respect des exigences fonctionnelles, évaluée via la réussite de vos soumissions à *automate*,
- votre respect des exigences non-fonctionnelles,
- un rapport dont le contenu est détaillé ci-dessous,
- une interrogation orale individuelle portant sur l'ensemble du travail.

3.2 Les tests fonctionnels

Deux séries de jeux de tests ont été fournies à votre (futur-)nouvel ami `http://concours.info.fundp.ac.be/automate`. La première permet de tester l'analyse lexico-syntaxique (y compris vérification de type) sur des fragments progressifs du langage (série L). La seconde teste le P-code produit (série P). Chacune de ces séries est composées d'une suite croissante de tests destinés à évaluer votre couverture des exigences fonctionnelles.

Les fragments composant cette série croissante sont définis comme suit :

0. déclarations d'entiers et booléens, affectations, lectures et affichages, opérations sur les entiers et les booléens
1. précédents + if, while
2. précédents + fonctions sans paramètres, pas de fonctions récursives ni imbriquées
3. précédents + fonctions imbriquées avec paramètres mais sans récursivité
4. précédents + fonctions récursives et imbriquées avec paramètres entiers et booléens
5. prédécents + surcharge de fonctions et déclaration forward

6. précédents + ensembles avec les instructions et opérations simples (ajout d'un élément, retrait d'un élément, min, max, nombre d'éléments, test de la présence d'un élément dans un ensemble)

Les jeux de tests portent les numéros suivants :

fragment	lex-synt	P-code
0	L0	P0
1	L1	P1
2	L2	P2
3	L3	P3
4	L4	P4
5	L5	P5
6*	L6	P6

Dans le cadre du cursus à horaire décalé, **le fragment 6 est facultatif** ; il est uniquement présent afin de servir de dépassement à ceux qui, entraînés par leur enthousiasme à réaliser un compilateur, souhaitent gérer un langage proposant plus que les simples entiers et booléens.

Remarques

- Les jeux de tests L0 et P0 ont été créés afin de vous permettre de vous familiariser à la procédure de soumission.
- Au niveau des booléens, le jeu P0 ne pourra valider que la syntaxe et la sémantique statique. Les instructions dont nous disposons alors ne sont pas suffisantes pour réaliser la validation de la sémantique des expressions booléennes. Celle-ci sera réalisée lors du jeu P1.

3.3 Les tests non-fonctionnels

A coté de l'évaluation fonctionnelle vérifiant votre respect de la spécification, plusieurs critères "de bonne conduite" seront aussi évalué (cette liste est non-exhaustive) :

- Propreté du code (par ex. : nom de variables significatifs, conventions de nommage cohérentes, respect des indentations, ...)
- Spécification et abstraction du code (montrant, par exemple, que vous avez réfléchi à la structure de l'implémentation au lieu de foncer tête baissée, ...)
- Accessibilité pour l'utilisateur (par ex. : messages d'erreur clairs et compréhensibles, ...)
- Gestion d'équipe, du fonctionnement en groupe.

3.4 Planning

Vous êtes libres d'organiser le travail de votre groupe selon la perspective de votre choix : par phases (lexicale, syntaxique, code) ou par fragments. L'évaluation se basera uniquement sur le résultat de vos soumissions lors de l'**échéance du 20 avril**.

À titre d'information, et afin de vous permettre de planifier au mieux votre effort, voici le planning imposé aux étudiants du cours du jour. Libre à vous de vous en inspirer afin de réaliser votre plan de travail.

date	tests numéro
10/02	L0, L1
24/02	P0, L2
09/03	P1, P2
23/03	L3, L4, L5
06/04	P3, P4, P5
20/04	L6, P6

Conseil : la réalisation d'un compilateur est un travail de longue haleine qui demande un investissement conséquent ; ne tardez donc pas à démarrer vos travaux.

3.5 Rapport

Le rapport **imprimé** sera remis le **26 avril** au plus tard à Xavier Devroey (bureau 334 [1]). Le corps du rapport comptera **environ 10 pages**. Le rapport contiendra :

1. En cas d'absence, glissez le dans la boite aux lettres à gauche de la porte

1. le nom et l'adresse email de chacun des membres du groupe,
2. une table des matières,
3. les choix faits pour lever les ambiguïtés de la grammaire et leur justification,
4. une présentation de vos principales structures de données,
5. la liste et description de vos fichiers sources,
6. les spécifications des principales fonctions (en-tête, objets utilisés de l'environnement, pré, post)
7. tout commentaire que vous jugerez pertinent [2]
8. en annexe, une version imprimée **avec numérotation des lignes** de chacun des fichiers source.

Attention.
- Mentionner le nom des auteurs dans chacun des fichiers du code source.
- **Veillez à la lisibilité** de votre code... Choisissez des noms de variables et de fonctions pertinents, n'hésitez pas à commenter votre code, indentez,...

3.6 Utilisation de l'automate

Vous trouverez l'automate à l'adresse : `http://concours.info.fundp.ac.be/automate` [3]. De même, une aide en ligne détaillée est disponible sur `http://concours.info.fundp.ac.be/automate/doc`.

Avant de pouvoir commencer à soumettre auprès de l'automate, vous devrez disposer d'un login et d'un mot de passe. Ceux-ci vous seront envoyés une fois les groupes constitués.

Remarque : les outils utilisés par *automate* sont flex, bison et gcc tous trois sous Linux (dans la dernière version fournie par la distribution Debian en branche stable) . Les commandes utilisées sont, dans l'ordre :

```
bison -dty *.y *.Y
flex *.l *.L
gcc *.c *.C
```

3.6.1 Accès à l'automate depuis l'extérieur des FUNDP

Pour atteindre l'interface de l'automate depuis l'extérieur du réseau des FUNDP, il vous faudra créer un tunnel ssh vers le port 80 de `concours.info.fundp.ac.be` puis utiliser l'adresse `http://localhost:XXXX/automate` (en remplacant XXXX par le numéro de port local que vous aurez choisi).

Une explication détaillée de la marche à suivre se trouve dans la documentation de l'automate à l'adresse : `http://concours.info.fundp.ac.be/automate/doc/index.php/Site/HowToTunnel`

3.7 Questions / problèmes techniques de l'automate / ...

Toutes les questions sur la réussite ou non-réussite des différents tests ou bien sur des éclaircissements via-à-vis de l'énoncé sont à adresser, de préférence , par mail aux **deux** encadrants (`aso@info.fundp.ac.be` et `xde@info.fundp.ac.be`. **Tous les mails devront comporter "[S&S]" dans leur sujet afin d'être correctement filtrés**, sans quoi le délai de réponse peut se voir considérablement allongé.

Soyez clairs dans vos demandes d'assistance : indiquez l'addresse de soumission utilisée ainsi que tous les éléments de réponse que vous possédez déjà. De même, joignez les éventuels codes utiles à la reproduction du comportement incriminé.

Il en va de même pour les éventuels problèmes techniques de l'automate (non-réponse dans un délai raisonnable, erreur de compilation incompréhensible, ...).

2. En particulier, tous les commentaires constructifs sur l'usage de l'automate sont les bienvenus.
3. Comme dit plus haut, cette interface est uniquement accessible depuis l'intérieur du réseau des FUNDP.

3.8 L'interrogation

L'interrogation orale aura lieu dans le courant de la semaine du 1 mai. Chaque étudiant doit être capable d'expliquer l'entièreté du travail de son groupe. Chacun se présentera à l'interrogation avec une version papier complète du rapport et des fichiers sources.

Bon travail !