Documentation de Conception

Valentin Laclautre, Anthony Dard, Damien Trouche, Martin Gangand,
Basel Darwish Jzaerly

Table des matières

1 Conception Architectural Etape B

2 Conception Architectural Etape C

Tout le code specifique se situe dans le package codegen. Il est constitué de plusieurs fichiers permettant de gérer la génération de code. Toutes les classes nécessaires sont instanciées dans DecacCompiler pour pouvoir être appelées lors de l'execution dans l'arbre.

2.1 Politique de gestion de pile et de registre

A chaque variable créée, on la place dans la pile et dans sa définition, on lui donne son adresse dans la pile. Lors de l'initialisation ou de l'affectation d'une variable, ou de n'importe quelle instruction nécessitant le calcul d'une expression, celui-ci est enregistré sur le registre R1. Bien sur lors du calcul, si c'est nécessaire, d'autres registres sont utilisé mais le résultat final est sur R1.

2.2 RegisterManager

Cette classe permet de gérer les registres. Elle prends en attributs le nombres de registres utilisés (ceux données en paramètres par la commande -r ou 16 sinon). Elle possède aussi un tableau de boolean en attributs. Chaque indice de ce tableau correspond à la valeur d'un registre. La valeur du tableau à cette indice est à vrai si le registre est utilisé et faux sinon. Cette classe possède aussi des méthodes permettant de renvoyer un registre inutilisé ou d'en libérer un.

2.3 Stack

Cette classe possède un attributs donnant la hauteur de la pile (par rapport à GB). Elle possède aussi de nombreuses méthodes permettant de mettre la valeur d'un registre au sommet de la pile, ou à un endroit précis de la pile. Elle possède aussi d'autres méthodes permettant de récupérer une variable se situant à une adresse précise dans la pile.

2.4 LabelManager

Cette classe permet de créer et de renvoyer des label uniques à partir d'un nom. Elle utilise pour cela un HasMap qui a un nom de label associe un

compteur correspondant au nombre de fois que ce nom de label est utilisé. Cela permet de s'assurer que tout les labels sont uniques

2.5 Utils

Cette classe regroupe des méthodes statiques utilisées à de nombreux endroits permettant la génération de code. Elle permet entre autre de renvoyer un Immediat d'après son type. Elle permet aussi de renvoyer tout le code correspondant à la gestion d'erreur qui appelé à la fin du codeGen du programme.

2.6 Propagation du code - Les fonctions codeGen

A chaque action devant être réalisé, il existe une fonction codeGen spécifique appelant récursivement dans l'arbre d'autres fonction codeGen. La première fonction appelé est codeGenProgram qui se propage à la génération de classe et du programme principale. Il est interressant de revenir sur quelque fonction codeGen importantes qui sont réutilisés de nombreuses fois.

2.6.1 codeGenPrint