

Trabalho Prático

Compilador da linguagem `Arith` para MIPS

1 Introdução

Este trabalho é parte integrante da avaliação prática da Unidade Curricular de código 11567 designada por Processamento de Linguagens, na sua edição de 2018-2019. O trabalho visa o exercício dos conceitos e das técnicas básicas de desenho de linguagens de programação e dos seus processadores e compiladores expostos nas aulas da presente UC. Como tal este trabalho é estruturado em várias fases. Estas diferentes fases visam ilustrar a utilização de técnicas (e ferramentas computacionais associadas) de processamento de linguagens. Em particular, as técnicas de análises léxica, sintática e semântica e produção de código máquina.

2 Descrição geral do trabalho

O objetivo geral deste trabalho é a construção de um compilador para a linguagem `Arith` (ver TD1 - link), tendo por alvo a arquitetura MIPS. Para além dos objetivos especificados, serão também valorizadas contribuições adicionais.

Pretende-se que neste trabalho sejam executadas as diferentes etapas de processamento de linguagens (análises léxica, sintática e semântica) sobre a linguagem `Arith`, terminando com a produção de código máquina correspondente para a arquitetura MIPS. Desta forma, o trabalho está estruturado em três fases.

A primeira fase consiste em entender a linguagem de base `Arith` e formalizar a sua sintaxe abstrata, semântica (operacional) e sistema de tipos. **Esta fase dará lugar à primeira entrega dia 19 de novembro de 2018.**

A segunda fase consiste em desenhar e implementar o *lexer*, *parser* e mecanismo de tipagem da linguagem `Arith`. **Esta fase dará lugar à segunda entrega dia 3 de dezembro de 2018.**

A terceira fase consiste em completar o código da entrega anterior com a parte de geração de código máquina MIPS.

O trabalho completo, concluído com a execução da terceira fase, **deverá ser entregue no dia 7 de janeiro de 2019**, e as defesas terão lugar nos primeiros dias da semana seguinte em calendário, por definir e afixar na página a UC. A entrega deverá respeitar as modalidades de entrega descritas na secção 5.

3 Arquitectura MIPS

Informação adicional sobre a arquitetura MIPS, alvo do seguinte trabalho, pode ser encontrado aqui:

- um curto descritivo do assembly MIPS ([link](#));
- uma aula de produção de código tendo o assembly MIPS como alvo ([link](#));
- um módulo OCaml para a produção de código MIPS ([link](#));
- um simulador online de MIPS com visualização dos registos ([link](#));
- um simulador MARS ([link](#)) (aconselhado);
- um simulador SPIM ([link](#));
- a documentação do SPIM ([link](#)) descreve o conjunto de instruções.

4 Trabalho requerido

- Como entregável para a primeira fase do trabalho, requer-se um documento que apresenta a linguagem `Arith` do ponto de vista da sua sintaxe abstrata, o seu sistema de tipos e a sua semântica.
- Como entregável para a segunda fase, o código `OCaml/menhir` com a implementação do *lexer*, *parser*, árvore de sintaxe abstrata e sistema de tipagem para a linguagem `Arith`.
- Como entregável para a terceira fase, um compilador completo escrito em `OCaml` com `menhir` para a linguagem `Arith`. Imperativamente, a arquitetura alvo considerada é a **arquitetura MIPS**.

O grupo de trabalho deverá considerar extensões que achar apropriadas e documentadas no relatório final. Estas extensões deverão ser discutidas com o regente da UC. Em jeito de sugestão, propõem-se as seguintes extensões:

- novos tipos primitivos (*boolean*, *float*, *character*, *string*, etc.);
- mecanismos de controlo de fluxo e ciclos (`if / else`, `while / for`);
- mecanismos de definição de variáveis mutáveis;
- mecanismos de definição e chamada de funções;

- mecanismos de definição de tipos eventualmente polimórficos predefinidos (listas, coleções, pilhas, filas, etc.) ou definidos pelo programador;
- mecanismos de declaração de tipos definidos pelo programador;
- mecanismos de declaração e utilização de apontadores/memória dinâmica;
- etc.

É esperado igualmente que o grupo reporte atempadamente à equipa docente os eventuais entregáveis preliminares (a conclusão incremental das fases de geração de código, como introduzido nas aulas) mas igualmente as dificuldades ou questões encontradas.

5 Entrega do trabalho

O trabalho deve ser entregue num arquivo `tar` comprimido (`nome.tgz`) em que `nome` é o identificador do grupo. Este arquivo deve conter todos os ficheiros fonte necessários à compilação, assim como um `Makefile` completo (as entradas `all` e `clean` devem estar presentes). Este arquivo deverá igualmente conter o relatório que descreve o trabalho feito, as escolhas (de desenho, etc.) tomadas, a documentação do código e o manual do utilizador. É igualmente esperada que seja preparada uma apresentação para a respetiva defesa.