PCS2056 – Linguagens e Compiladores

Assunto: Semântica informal – comandos

Objetivo: Estudo informal da semântica dos comandos em uma linguagem imperativa.

Palavras-chave:
Expressões aritméticas e booleanas
Comandos imperativos: atribuição de valor, chamada de sub-rotina, read, print
Rótulos e comandos de desvio incondicional, condicional e múltiplo
Comandos de controle de fluxo: if, while, do-until, for, case

Questões:

1) Mostre, através de um conjunto de exemplos significativos, de que forma pode ser escrito, em uma linguagem de baixo nível (do tipo assembly para uma arquitetura baseada em registradores), um programa que calcula o valor de uma expressão aritmética dada. Considerar que tal expressão aritmética envolve apenas constantes e variáveis aritméticas simples, podendo utilizar as operações aritméticas usuais, e um número arbitrário de parênteses aninhados.
2) Repita o exercício anterior para uma arquitetura baseada em pilha.
3) Repita mais uma vez, considerando que a arquitetura disponível é uma simples máquina de calcular à sua escolha, e que a linguagem a ser usada tem como conjunto de instruções o conjunto de teclas disponíveis nessa máquina.
4) Amplie as expressões em questão inserindo, um de cada vez, os seguintes elementos: (a) variáveis indexadas – vetores ou matrizes aritméticas; (b) chamadas de funções aritméticas (atenção à forma como os parâmetros devem ser passados); (c) elementos aritméticos, selecionados de algum dos campos de uma estrutura.
5) Mostre como escrever programas, em linguagem de baixo nível, que sejam equivalentes a um comando responsável pela atribuição de um valor aritmético a: (a) alguma variável aritmética; (b) alguma posição de vetor ou de matriz aritmética; (c) algum elemento aritmético selecionado de algum campo pertencente a uma estrutura.
6) Mostre como representar, em linguagem de baixo nível, o cálculo de uma expressão booleana: (a) formada pela comparação de duas expressões aritméticas; (b) formada a partir de outras expressões booleanas mais simples, combinadas com operadores booleanos usuais e aninhamento arbitrário de parênteses; (c) combinando os casos (a) e (b); (d) permitindo ainda elementos booleanos extraídos de agregados quaisquer; (e) permitindo adicionalmente chamadas de funções booleanas.
7) Repita, para o caso booleano, o exercício 5, usando as expressões criadas no exercício 6.
8) Mostre como escrever, em linguagem de baixo nível, programas equivalentes aos comandos de desvios incondicionais, condicionais e múltiplos.
9) Como programar, na linguagem de baixo nível, um programa que realize os comandos de entrada e saída da linguagem de alto nível? Lembre-se que a entrada e saída em linguagem de alto nível não se limita a efetuar transferências de dados através das portas do computador.
10) Mostre como construir programas em linguagem de baixo nível, equivalentes aos comandos de controle de fluxo – if, while, do-until, for, case.

Uma referência sobre semântica (aulas desta semana)
Tremblay e Sorenson *The theory and practice of compiler writing*. McGraw-Hill, 1985

PCS2056 – Linguagens e Compiladores

Assunto: Especificação do ambiente de execução.
Enunciado da **parte 4** do trabalho: Especificação do ambiente de execução da linguagem de programação definida
Data de entrega: **10 de novembro de 2016**

Objetivo: Estudo do ambiente de execução de uma linguagem imperativa.

Palavras-chave:
Preparação do ambiente
Sistemas de programação
Máquina virtual
Chamadas de rotinas (call/return)
Uso de pilha
Inicialização de variáveis/constantes
Alocação de espaço para o programa: código, dados, pilha e *heap*
Definição do processo de chamada e retorno de sub-rotinas da linguagem
Rotinas da biblioteca: Entrada/saída, acesso ao SO
Programas de sistema: carga programas (loader), programa montador, relocador, editor de ligações

Questões:

1) Explique como deve ser efetuada a seqüência de chamada e retorno de uma sub-rotina:
   a) Em que momento é possível determinar o espaço necessário a ser alocado para um registro de ativação?
   b) Como calcular os endereços dos parâmetros e variáveis locais?
   c) Como criar um registro de ativação?
   d) O que deve ser armazenado para permitir o controle da chamada/retorno dentro do registro de ativação?

2) Deduza uma fórmula geral para calcular, em relação ao endereço do apontador de registro de ativação, os endereços virtuais dos parâmetros, cujos elementos possam ser bytes, palavras de 32 bits ou agrupamentos de 64 bits conforme o tipo do elemento (char, int ou double, respectivamente).

3) Deduza, de forma semelhante à efetuada no exercício anterior, uma fórmula geral para calcular os endereços virtuais das variáveis locais, cujos elementos possam ser bytes, palavras de 32 bits ou agrupamentos de 64 bits conforme o tipo do elemento (char, int ou double, respectivamente).

4) Para o caso de sub-rotinas aninhadas, explique como acessar variáveis locais do bloco aninhador dentro do bloco aninhado.

5) Deduza uma fórmula geral para calcular, em relação ao endereço do apontador de registro de ativação, os endereços virtuais das variáveis locais do bloco aninhador. Onde armazenar a informação do apontador do registro de ativação do bloco aninhador?

6) Escreva em uma linguagem do tipo *assembly*, para a máquina objeto definida, rotinas para tratamento de estrutura de pilha (push/pop/reset).

7) Complete a definição da estrutura de chamada de rotinas em *assembly*, usando a estrutura de pilha definida (estabeleça o que deve ser empilhado/desempilhado).

8) Escreva em uma linguagem do tipo *assembly*, para a máquina objeto definida, rotinas para tratamento de entrada e saída (para/de uma região de memória).

9) Complete a definição do acesso ao SO.

10) Defina a estrutura dos programas montador e carregador. Como serão executados os seus programas na máquina virtual?