PCS2056 – Linguagens e Compiladores

Assunto: 5a. parte do projeto – definição das rotinas semânticas.
Entregar no dia 27/11/2008

Assunto: Formas intermediárias.

Objetivo: Apresentar formas intermediárias de programas como alternativas para a geração de código, e seu uso potencial em atividades de otimização.

Palavras-chave:

formas intermediárias
notação infixa
notação polonesa prefixa
notação polonesa reversa
ênuplas
triplas
quádruplas
árvores de sintaxe
sintaxe concreta e abstrata
código alinhavado
máquinas abstratas
p-code

Conceitos

formas intermediárias: linguagens de saída de um passo de compilação, e entrada do passo seguinte.
notação polonesa prefixa ou funcional operação operando1 operando2 ... operando n
notação polonesa reversa (operandos, operação)
ênuplas (operador, operando1, operando2, ... operando n)
triplas (operador, operando1, operando2)
quádruplas (operador, operando1, operando2, operando3)
árvores de sintaxe – similares às árvores de derivação, porém em geral codificadas
sintaxe concreta – inclui todos os símbolos usados na codificação do texto do programa
sintaxe abstrata – elimina do texto-fonte os símbolos que não contribuem para a análise/compilação
código alinhavado – corresponde a uma lista ligada de ponteiros para procedimentos
máquinas abstratas – hardware virtual com conjunto de instruções aderente à linguagem de alto nível
p-code – código intermediário para a linguagem Pascal.

Exercícios:

1) Partindo de um reconhecedor sintático de expressões aritméticas, já construído anteriormente, acrescente rotinas semânticas que gerem como saídas as expressões equivalentes em formato de notação polonesa prefixa.

2) Repita o exercício anterior para notação polonesa reversa.

3) Usando como linguagem de saída a forma intermediária de triplas, compile expressões aritméticas para essa forma. Lembre-se de que as triplas, por convenção, associam o resultado da operação ao número seqüencial da tripla dentro da seqüência de triplas que compõem o código gerado.

4) Repita para quádruplas. Lembre-se de que o terceiro operando da quádrupla é o nome do local em que a quádrupla deposita o resultado da operação.

5) Codifique uma expressão relativamente longa usando as formas intermediárias citadas. Represente-a em forma de árvore. Compare a árvore com as demais notações e estabeleça uma correspondência entre essas formas alternativas de representação.

6) Procure na literatura ou na Internet descrições de máquinas abstratas importantes, como a máquina virtual da linguagem Java, a máquina P (cuja linguagem é o P-code), etc. Estude uma delas, e identifique as vantagens de sua utilização. Por que não se costuma usar algum hardware real que a interprete?

PCS2056 – Linguagens e Compiladores

Assunto: Exercício – Projeto de um meta-analisador sintático a partir de gramáticas de Wirth.

Objetivo: Exercitar a criação de autômatos diretamente a partir do conhecimento de uma linguagem, sem passar por todas as etapas formais de sua obtenção.
Automatizar o processo de obtenção de reconhecedores sintáticos a partir de gramáticas denotadas na notação tradicional de Wirth.

Palavras-chave:

autômato de pilha estruturado
notação de Wirth modificada
auto-recursões à direita/esquerda e centrais
grafo de dependência dos não-terminais
não-terminais essenciais e não-essenciais
expressões de Wirth modificadas
atribuição de estados às expressões na notação de Wirth modificada
expressões de Wirth modificadas numeradas
mapeamento em transições do autômato
eliminação de não-determinismos
minimização do autômato

Questões:

1) Desenhar um autômato de pilha estruturado que reconheça gramáticas na notação de Wirth tradicional.
2) Acrescentar ao autômato da questão anterior rotinas semânticas que o transformem em um transdutor capaz de gerar como saída uma gramática com as expressões numeradas conforme o algoritmo de atribuição de estados estudado nas aulas anteriores.
3) Desenhar outro autômato de pilha estruturado que reconheça as gramáticas com expressões numeradas, obtidas como saídas na questão anterior.
4) Acrescentar-lhe rotinas semânticas que gerem como saídas as transições de estados do autômato correspondente à gramática de Wirth fornecida.
5) Construir um interpretador para o conjunto de regras de transição de estados gerado na questão anterior.
6) Testar o conjunto de programas construído partindo de uma gramática de Wirth, e gerando o autômato correspondente, e alimentando esse autômato com sentenças da linguagem definida pela gramática. Se for o caso, use, como parte do interpretador das transições do autômato, o analisador léxico construído anteriormente.

Referências:

J. J. Neto, C. B. Pariente and F. Leonardi, **Compiler Construction: A Pedagogical Approach**. *Proceedings of the ICIE*. Buenos Aires. 1999. [disponível para download em www.pcs.usp.br/~lta]