UNIVERSIDADE FEDERAL DO CEARÁ
CAMPUS DE CRATEÚS
CURSO: CIÊNCIA DA COMPUTAÇÃO
DISCIPLINA: COMPILADORES

MARCUS VINÍCIUS MARTINS MELO

SELEÇÃO DE INSTRUÇÃO

Trabalho apresentado à Disciplina de Compiladores, ministrada pelo Prof. Roberto Cabral Rabelo Filho, para obtenção parcial de créditos e título de Bacharel em Ciência da Computação.

JUNHO / 2018
CRATEÚS-CE

Sumário

1 INTRODUÇÃO

Seleção de instrução é o estagio do back-end de um compilador que transforma sua representação intermediaria(IR) em uma representação de baixo nível, onde cada operação corresponde exatamente a uma instrução na maquina de origem, depende da maquina, ou seja ela transforma a arvore em um código correspondente, pode-se gerar o código correspondente utilizando duas estrategias, a primeira usa a estrategia gulosa que consiste em tentar encontrar a solução ótima local em cada fase com a esperança de encontrar uma solução ótima global. O algoritmo utilizado é o MAXIMAL MUNCH, bastante simples, a partir da raiz (TOP DOWN) ele tenta encontrar o maior padrão que se encaixa nesse nó, a cada padrão gerado uma expressão é gerada. O segundo é o algoritmo dinâmico(optimum), ao contrario do guloso este usa BOTTOM-UP, de baixo para cima, monta uma solução ótima baseada em soluções ótimas já encontradas, a cada nó é dado um peso ou custo, as duas estrategias tem custo linear, na verdade a fase de seleção de instrução é bem rápida.

2 OBJETIVO DO TRABALHO

A proposta deste trabalho se objetiva em construir uma estrutura que de suporte a fase seleção de instrução de um compilador com base na arquitetura Jouette, imprimir e construir as arvores com base na IR, além de gerar o código resultante baseado nas estrategias gulosa e dinâmica.

3 METODOLOGIA

Para a elaboração deste trabalho, as atividades necessárias foram divididas em cinco fases, a primeira fase de planejamento, pois não se pode iniciar implementação sem antes saber o que vai ser implementado e como vai ser implementado, após isso, segunda parte montar a estrutura, a classe Node, representando o no, e a representação das expressões.

A terceira parte foi responder a primeira questão do trabalho, ou seja, a partir de uma sequencia de instruções imprimir uma arvore correspondente, a ideia principal é cortar sempre a string aonde tem virgula, que separa o filho direito e esquerdo da instrução, para isso foi usado uma pilha, aonde quando eu leio abre parêntese eu empilho e fecha parêntese eu desempilho, quando eu ler uma string e a pilha estiver vazia, bazinga eu encontrei aonde separa os filhos e dai é só chamar pra recursão.

A quarta parte foi implementar os metodos guloso e dinâmico, o método guloso foi implementado da seguinte forma, a partir do no raiz eu verifico se eu consigo encaixar a maior instrução naquele no, se eu conseguir bazinga, se não eu tento a segunda maior e assim por diante ate encontrar uma que se encaixe, conforme a instrução se encaixa, eu chamo recursivo para os nos que ainda faltam e o procedimento se repete, o dinamico deu muito trabalho, tanto que ficou incompleto, vou nem falar muito dele.

A quinta e ultima parte foi receber o conjunto de padrões e exibir o código correspondente, para essa foi utilizada a questão 2 como parâmetro de resolução, baseado nas duas estrategias dinâmica e gulosa foi impresso o código equivalente.

4 DIFICULDADES ENCONTRADAS

Durante a elaboração do trabalho, a principal dificuldade encontrada foi a manipulação de arvores em python, dificuldade que logo foi solucionada, por incrível que pareça o python guarda referencias dos objetos mesmo que não sejam feitas explicitamente, no meu caso do objeto Node, que representa o no da arvore, esse problema foi resolvido pois mudei de estrategia de implementação, caso contrario teria que criar novas instancias cada vez que modificasse o objeto senão modificaria o original, houve um pouco de dificuldade também durante a implementação do algoritmo dinâmico.

5 CONCLUSÃO

A partir do trabalho, foi possível perceber, de início, que a fase de seleção de instrução tem sua importância no processo de compilação, foi visto também que a estrategia gulosa, embora seja uma estrategia muito simples comparada ao dinâmico, nos trás resultados bem significativos e portanto que resolvem meu problema, detalhe não necessariamente é a melhor resolução possível, mas visto isso e portanto é a mais utilizada, por fim foi adquirido muito conhecimento sobre o tema em questão, que obviamente era o objetivo desse trabalho.

6 REFERÊNCIAS

APPE A. ; **Modern Compiler Implementation in Java**. Second Edition. October 2002;

Instruction Selection; Disponível em :http://cs.au.dk/~eernst/dOvsE13/materials/45a-instrsel.pdf; Acessado em 7/06/2018;