

## REO7 – Atividade 2

Pedro Antônio de Souza – 201810557

A geração do código intermediário é um processo não é obrigatório e, quando é implementado, deve ser executado logo após a análise semântica. Embora não seja obrigatória, a geração do código intermediário é realizada na maior parte dos compiladores visando otimizar o código-fonte recebido do usuário antes de se iniciar o processo de síntese para linguagem alvo. Dentre outras otimizações, podemos apontar a eliminação de partes não alcançáveis do código (como código logo após o retorno dentro de uma função) e remoção de redundâncias (operações que são efetuadas várias vezes).

Os produtos do analisador sintático que não contêm erros verificados pelo analisador semântico são utilizados para gerar o código intermediário. Esse novo código com instruções simplificadas é escrito em uma linguagem interposta entre o alto nível e o baixo nível, o que faz todo sentido, já que essa etapa da compilação está entre o *front-end* e o *back-end* do compilador.

A Tradução Dirigida pela Sintaxe (TDS) é a principal método para a geração do código intermediário. Essa representação intermediária possui várias representações, como Árvore Sintática Abstrata (AST), forma posfixa e Código de Três Endereços (TAC), sendo essa última a mais comum. Basicamente, a AST altera a árvore sintática gerada pelo analisador sintático omitindo informações irrelevantes para as etapas posteriores. Já a forma posfixa modifica a notação para representar operações aritméticas. Nessa nova notação, os operadores são colocados após os operandos. Assim, explicita-se a prioridade de operações. Por ser mais popular, o TAC será abordado mais profundamente a seguir.

Utilizar o códigos de três endereços simplifica e padroniza instruções. Em uma linguagem imperativa, sempre pode haver uma variável que armazena o resultado de uma operação entre outras duas variáveis. Os três endereços são referentes justamente às três variáveis citadas anteriormente. Sabendo que no código de três endereços pode haver no máximo um operador do lado direito de uma instrução, é comum a criação de variáveis temporárias no código intermediário. Deste modo, uma instrução  $x + y * z$  presente no código fonte deve ser simplificada para:

$$t1 = y * z$$
$$t2 = x + t1$$

onde **t1** e **t2** são variáveis temporárias criadas na etapa de geração de código intermediário. Com o código simplificado, a tarefa de encontrar redundâncias, e outras otimizações, é facilitada radicalmente. O código de três endereços, nada mais é do que a representação linear de uma árvore sintática. Assim, infere-se que a geração desse código pode ser paralelizada à etapa de análise sintática, otimizando ainda mais o compilador.