

Fabio Tirelo

Uma Ferramenta Para Execução de um Sistema Dinâmico Discreto Baseado em Álgebras Evolutivas

Dissertação apresentada ao Curso de Pós-Graduação em Ciência da Computação da Universidade Federal de Minas Gerais, como requisito parcial para a obtenção do grau de Mestre em Ciência da Computação.

Belo Horizonte

22 de agosto de 2000

Fabio Tirelo

Uma Ferramenta Para Execução de um Sistema Dinâmico Discreto Baseado em Álgebras Evolutivas

Dissertação apresentada ao Curso de Pós-Graduação em Ciência da Computação da Universidade Federal de Minas Gerais, como requisito parcial para a obtenção do grau de Mestre em Ciência da Computação.

Belo Horizonte

22 de agosto de 2000



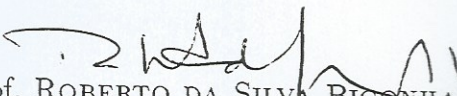
UNIVERSIDADE FEDERAL DE MINAS GERAIS

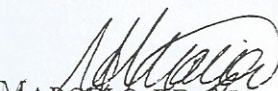
FOLHA DE APROVAÇÃO

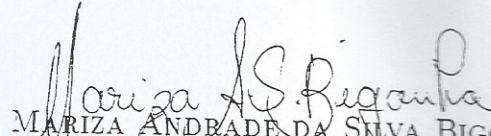
Uma Ferramenta para Execução de um Sistema Dinâmico Discreto Baseado em Álgebras Evolutivas

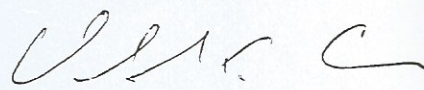
FABIO TIRELO

Dissertação defendida e aprovada pela banca examinadora constituída pelos Senhores:


Prof. ROBERTO DA SILVA BIGONHA - Orientador
Departamento de Ciência da Computação - ICEX - UFMG


Prof. MARCELO DE ALMEIDA MAIA
Departamento de Computação - UFOP


Profa. MARIZA ANDRADE DA SILVA BIGONHA
Departamento de Ciência da Computação - ICEX - UFMG


Prof. OSVALDO SÉRGIO FARHAT DE CARVALHO
Departamento de Ciência da Computação - ICEX - UFMG

Belo Horizonte, 22 de agosto de 2000.

Resumo

Máquinas de Estado Abstratas (ASM, do inglês *Abstract State Machines*) são um formalismo cujo objetivo é fornecer semântica operacional para algoritmos em qualquer nível de detalhamento. Uma grande vantagem do método é a possibilidade de se executar a especificação, o que pode facilitar a sua verificação. Definimos uma linguagem de nome *Machina* baseada no formalismo ASM. Além de implementar todos os recursos fundamentais do formalismo, *Machina* possui também mecanismos de controle de visibilidade e é fortemente tipada. Implementamos também um compilador de *Machina* para linguagem C. Descrevemos algumas técnicas de otimização que poderão trazer grandes benefícios no tempo de execução do programa compilado.

Abstract

Abstract State Machines (ASM) are a formal method designed to provide operational semantics for algorithms in different levels. The main advantage of this method is the possibility of executing a specification, which may ease program verification. We have designed and implemented a language named *Machina*, based on ASM formalism. *Machina* implements all ASM features, as well as visibility control mechanisms and a strongly typed system. The *Machina* compiler generates C code. In addition, this thesis describes optimization techniques which may significantly improve the execution time of the compiled *Machina* program.

Agradecimentos

Os criadores de ferramentas foram recriados por suas próprias ferramentas. Isso porque, ao usarem porretes e pedras, suas mãos haviam desenvolvido uma destreza única no reino animal, permitindo-lhes fabricar ferramentas ainda melhores que, por sua vez, desenvolveram ainda mais suas mãos e seus cérebros. Foi um processo acelerado e cumulativo, estando no fim o Homem.

Arthur Clarke, 2001 – *Odisséia Espacial*

Gostaria de agradecer a todos que ajudaram, direta ou indiretamente, na conclusão deste trabalho.

Agradeço primeiramente a Deus por estar presente em todos os momentos, dando forças para persistir até quando a conclusão deste trabalho parecia impossível.

Agradeço a Leonor, minha mãe, e a Angelina, minha avó, pelo apoio constante, desde antes da graduação até o fim do Mestrado.

Agradeço também ao mestre e amigo Vladimir Oliveira Di Iorio, que ajudou de diversas maneiras em vários momentos do Mestrado, desde antes do seu início, até a sua conclusão.

Agradeço também a todos os colegas da Pós-Graduação, sem os quais a conclusão deste trabalho seria muito mais penosa.

Agradeço ao professor Roberto da Silva Bigonha pela sugestão do projeto, pela orientação precisa e por estar sempre disponível para discutir os aspectos mais complicados do trabalho.

Agradeço ao Curso de Pós-Graduação em Ciência da Computação, pela estrutura que forneceu para o bom andamento deste trabalho, aos professores Mariza Andrade da Silva Bigonha, Marcelo de Almeida Maia e Osvaldo Sérgio Farhat de Carvalho, por aceitarem participar da banca, e ao CNPq pelo apoio financeiro.

Sumário

1	Visão Geral	1
1.1	Introdução	1
1.2	Trabalho Proposto	2
1.3	Motivação	2
1.3.1	Avaliação da Metodologia	2
1.3.2	Aplicação da Ferramenta Desenvolvida	5
1.4	Ambientes de Execução de ASM	6
1.5	Organização do Texto	6
2	Máquinas de Estado Abstratas	9
2.1	Introdução ao Modelo ASM de Especificação	9
2.1.1	Exemplos	12
2.2	O Modelo Formal de ASM	15
2.2.1	Definição da Máquina	15
2.2.2	Exemplo de Especificação ASM	19
2.2.3	Regras Não-Básicas	20
2.3	ASM Multiagentes	22
2.3.1	Exemplos de ASM Multiagente	23
2.4	Conclusões	26
3	A Linguagem Machina	27
3.1	Introdução	27
3.2	Visão Geral de Machina	27
3.3	Programas em Machina	28
3.4	Escopo e Visibilidade	29
3.5	Estado Inicial	30
3.5.1	Definição de Tipos	30
3.5.2	Definição de Funções	31

3.5.3	Definição de Ações	33
3.6	Regras de Transição	33
3.6.1	Regra de Atualização	34
3.6.2	Regra Bloco	34
3.6.3	Regras Condicionais	34
3.6.4	Regras <i>let</i>	36
3.6.5	Regras Não-Básicas: <i>forall</i> e <i>choose</i>	36
3.6.6	Regras de Execução de Agentes	37
3.6.7	Chamadas de Ação	38
3.7	Invariantes	39
3.8	Execução Intercalada	40
3.9	Conclusões	40
4	Implementação de Machina	43
4.1	Arquitetura do Compilador	43
4.2	Tabela de Símbolos	44
4.3	Código MIR	45
4.4	Compilação de Machina Para MIR	46
4.4.1	Análises Léxica e Sintática	46
4.4.2	Análise Semântica	47
4.4.3	Geração de Código MIR	47
4.5	Geração de Código C	47
4.5.1	Estrutura de Dados Para Geração de Código	47
4.5.2	Organização do Código Gerado	48
4.5.3	Importações	48
4.5.4	Compilação de Declarações	49
4.5.5	Listas de Atualização	54
4.5.6	Estrutura de Tabela Hash	57
4.5.7	Regra de Transição	62
4.5.8	Invariante do Módulo	71
4.5.9	Expressões	72
4.5.10	Inicialização	79
4.5.11	Compilação de Ações	81
4.6	Otimização de Código Machina	84
4.6.1	Principais Fontes de Otimização	84
4.6.2	Escalonamento de Código	85

4.6.3	Otimização de Desvios	88
4.6.4	Outras Fontes de Otimização	89
4.7	Conclusões	92
5	Avaliação do Trabalho	93
5.1	Avaliação de Machina	93
5.2	Avaliação da Implementação	93
5.3	Avaliação das Otimizações Propostas	94
5.4	Conclusões	94
6	Conclusões e Trabalhos Futuros	97
6.1	Conclusões do Trabalho	97
6.2	Principais Contribuições	98
6.3	Trabalhos Futuros	98
A	Manual de Referência da Linguagem Machina	105
A.1	Introdução	105
A.2	Comentários	105
A.3	Literais	105
A.4	Palavras Reservadas	106
A.5	Identificadores	106
A.6	Notação Para Sintaxe	107
A.7	Uma Especificação em Machina	107
A.7.1	Módulos	107
A.7.2	Definição de Máquinas	109
A.8	Agentes	110
A.9	Término da Execução	111
A.10	Mecanismos de Visibilidade	112
A.11	Declaração de Tipos	113
A.11.1	Valores Booleanos	114
A.11.2	Caracteres	115
A.11.3	Números Inteiros	115
A.11.4	Números Reais	116
A.11.5	Cadeias de Caracteres	117
A.11.6	Enumerações	117
A.11.7	Unões Disjuntas	118
A.11.8	Tipo ?	118

A.11.9 Intervalos	118
A.11.10 Tuplas	118
A.11.11 Conjuntos	119
A.11.12 Listas	119
A.11.13 Tipos Funcionais	121
A.11.14 Tipos Agentes	121
A.12 Regras de Visibilidade para Tipos	121
A.13 Regras para Dedução de Tipos	122
A.14 Compatibilidade de Tipos	123
A.15 Valor <i>Default</i> de um Tipo	124
A.16 Declaração de Funções	125
A.17 Declaração de Funções Externas	126
A.17.1 Protocolo de Comunicação de Funções Externas	126
A.18 Definição das Regras de Inicialização	128
A.19 Declaração de Abstrações de Regras	128
A.20 Seção de Definição de Regras de Transição	129
A.20.1 Regras Básicas	129
A.20.2 Regra <i>forall</i>	130
A.20.3 Regra <i>choose</i>	131
A.20.4 Regras Para Criação e Eliminação de Agentes	132
A.20.5 Regras da Parada	133
A.20.6 Regra <i>let</i>	133
A.20.7 Regra <i>case</i>	134
A.20.8 Regra <i>with</i>	135
A.20.9 Ações	136
A.21 Invariante da Execução	136
A.22 Sintaxe de Expressões	137
A.22.1 Operadores de Máquina	137
A.22.2 Aplicação de Funções	137
A.22.3 Mapeamentos	138
A.22.4 Condicionais e Let	138
A.22.5 Conversão de Tipos Condicional	139
A.22.6 Inspeção de Tipos	140
A.22.7 Expressões Especiais	140
A.22.8 Gramática das Expressões	141
A.23 Manipulação de Arquivos	143

B Exemplos de Programas em Machina	147
B.1 Pesquisa Binária	147
B.2 Ordenação por Seleção	148
B.3 Números Primos	149
B.4 Especificação da Semântica de <i>Tiny</i>	150
B.4.1 Módulo de Globais (Globals)	150
B.4.2 Módulo de Operações (Operations)	151
B.4.3 Módulo de Expressões (Expressions)	153
B.4.4 Módulo de Comandos (Comandos)	155
B.4.5 Módulo principal (Tiny)	156
B.5 Jantar dos Filósofos	157