

AGRADECIMENTOS

Em especial, a duas pessoas que nos deixaram antes da conclusão deste trabalho: o professor Sérgio Carvalho, que estimulou meu trabalho com transformações, colaborando com a tese da Mariela e com o artigo sobre especificação formal do tratamento de exceções na linguagem DDL; e meu avô, Fernando, que sempre estimulou meu interesse pela ciência e de quem eu herdei o meu gosto pelo raciocínio matemático.

Ao professor Hermann, que está sempre disposto a ajudar seus alunos; sempre incentivando e transmitindo entusiasmo pelo nosso trabalho.

Aos professores da PUC-Rio, por sua extrema competência na arte de instruir os alunos. Em particular ao Hermann, Veloso, Rangel e Luiz Carlos, se pudesse passaria mais alguns anos assistindo aos seus cursos.

Ao meu amigo Luiz Carlos, que soube exatamente como me atrair para o doutorado: me sugeriu que assistisse a um curso do Hermann; e também pela criação de diversos exemplos de tradução que foram usados nesta tese.

Aos meus colegas do Instituto Militar de Engenharia que deram apoio integral para a minha dedicação ao curso.

Aos meus alunos, com os quais eu aprendo continuamente. Em especial àqueles que trabalharam com transformações, Mariela, Massayoshi e Gilberto, que produziram *feedback* interessante para esta tese.

Ao professor Armando, por sua coragem de unir a indústria à academia, que possibilitou os trabalhos com transformações que deram origem a este trabalho.

Ao professor Emmanuel, pelo mesmo motivo, e também pela orientação no mestrado e pela ajuda a ingressar no doutorado.

Aos professores Armando e Tom Maibaum pelas idéias e incentivos durante seminário no Imperial College.

A Fernando, Bazilio, Patricia, Geiza, Isabel, Bel, Regina, Marcelo Felix, Marcelo Frias, Marcelo Correia, Renata, Maria da Paz, Rodrigo, Toacy, Sylvia, Alexandre, Ivan, Christian, Cristina, Fernanda, Luiz Carlos, Juan e demais colegas da PUC e do LMF, com os quais eu aprendi muito, e que foram amigos dedicados, prestando total apoio nos momentos de dificuldade.

A todos aqueles que estimularam meu interesse científico ao longo da minha vida, meus pais, meus parentes, meus professores.

À minha esposa Pat, por sua compreensão e apoio nos muitos momentos que tive que privá-la da minha companhia e atenção.

À PUC-Rio por ter oferecido ótimas condições para a realização do trabalho e por ter concedido bolsa "VRAC" durante a maior parte do mesmo.

Resumo

Esta tese propõe um modelo categórico para tradução entre linguagens de programação. Este modelo é utilizado para se estabelecer uma condição formal que uma tradução deve satisfazer para preservar a manutenibilidade dos programas traduzidos. Para tanto, partimos de um modelo categórico de R.F.C. Walters para linguagens livre de contexto. A fim de se estabelecer o critério para a preservação da manutenibilidade, fez-se necessário a introdução de semântica para os modelos categóricos de linguagens livres de contexto. Mostramos através de exemplos que a semântica dos morfismos (que são modelos de trechos de derivação) está relacionada com a preservação da manutenibilidade dos programas. Acreditamos que a técnica desenvolvida, além de se aplicar ao problema estudado nesta tese, traz um maior entendimento da semântica de linguagens de programação e da forma com a qual a intenção do programador está relacionada com a estrutura sintática de um programa.

Abstract

This thesis proposes a categorical model for translating programming languages. This model is used to establish a formal condition that a translation should satisfy in order to preserve program maintainability throughout the translation process. For this purpose, we start from a categorical model proposed by R.F.C. Walters for context-free languages. In order to establish the criterion for preserving maintainability, it was necessary to introduce semantics for the categorical models of context-free languages. The examples shown illustrate that the arrows semantics (arrows are models of derivation pieces) is related to the preservation of program maintainability. We believe that the technique we developed, not only applies to the problem studied in the thesis, but also brings an important insight regarding the semantics of programming languages and the way in which the programmer's intention is related to the syntactical structure of a program.

ÍNDICE

I – Introdução	1
1.1 Posicionamento	1
1.2 Tradução entre Linguagens de Programação	3
1.2.1 Compilação	3
1.2.2 Transformação	4
1.2.3 Ferramentas de transformação	5
1.2.4 Aplicações	6
1.3 Contribuições	7
1.4 Apresentação	8
 II – Histórico	10
2.1 Introdução	10
2.2 A ferramenta TXL	11
2.2.1 O funcionamento TXL	11
2.2.2 Técnicas de programação de tradução usando-se TXL	16
2.3 A tradução de DDL para C++	21
2.3.1 Visão Geral	22
2.3.2 Dificuldades identificadas no projeto	24
2.4 Sistema de Tradução Baseado em Semântica	26
2.4.1 Visão Geral	26
2.4.2 Dificuldades identificadas no projeto	28
2.3 Tradução de CHILL para C++	29
2.3.1 Visão Geral	29
2.3.2 Dificuldades identificadas no projeto	31
 III - Modelo Categórico de Linguagens Livres de Contexto	32
3.1. Introdução	32
3.2. Noções usuais de linguagens e gramáticas	32
3.3. Noção categórica de gramáticas livres de contexto	36
3.3.1 Multigrafos	36
3.3.2 Gramáticas Livres de Contexto	37
3.4 Noção categórica de linguagens livres de contexto	40
3.4.1 Linguagens livres de contexto	40
3.4.2 Linguagem gerada por uma gramática	42
3.5 Relação entre as noções usuais e categóricas	45
3.6 Resumo	46
 IV – Semântica de Árvores de Derivação	47
4.1. Introdução	47
4.2. Semântica denotacional usual	47
4.3. Uma interpretação concreta de uma floresta de derivações	49
4.4. Semântica de árvores de derivação	50
 V - Tradução e Preservação da Manutenibilidade	54
5.1. Introdução	54
5.2. Exemplos reais de tradução	54

5.3 Outras aplicações para o conceito de semântica de árvores	58
5.3.1 Avaliar a expressividade de uma linguagem	58
5.3.2 Comparar programas	59
5.3.3 Estudar padrões de projeto	61
VI – Modelo Categórico de Traduções entre Linguagens	62
6.1. Introdução	62
6.2. Noção usual de tradução entre linguagens	62
6.3 Noção categórica de tradução entre linguagens	64
6.4 Relação entre a noção usual de tradução e a categórica	69
6.5 Modelo categórico de um ETDS	70
6.6 Tradução que preserva a semântica de árvores	73
VII – Aplicação	76
7.1. Introdução	76
7.2. Os tradutores estudados	76
7.3. Conclusões	77
7.4. Exemplos	80
VIII – Conclusão	97
8.1. Trabalhos Futuros	97
8.2. Contribuições	99
Apêndice A – Demonstração dos Teoremas	101
A.1. Adjunção entre \mathbf{Gram}^Σ e $\mathbf{Cat}_\square^\Sigma$	101
A.2. Equivalência entre o conceito usual e o categórico de LLC.	104
A.3. Adjunção entre $\mathbf{Cat}_\square^\Sigma$ e \mathbf{Cat}_+^Σ	107
A.4. Correção e Completude do Modelo de ETDS simples	110
A.5. Adjunção entre $\mathbf{Cat}_\square^\Sigma$ e \mathbf{Cat}_τ^Σ	121
A.6. Correção e Completude do Modelo de ETDS	123
A.7. Adjunção entre \mathbf{Cat}_τ^Σ e \mathbf{Cat}_+^Σ	129
Referências Bibliográficas	132

Lista de Figuras

Figura 1.1	Processo de compilação.	3
Figura 1.2	Tradução usando transformações.	5
Figura 2.1	Sintaxe de TXL – I.	12
Figura 2.2	Especificação informal das transformações da “calculadora”.	13
Figura 2.3	Sintaxe de uma regra em TXL.	14
Figura 2.4	Código TXL das transformações da “calculadora”.	14
Figura 2.5	Regra <i>mainRule</i> para a calculadora.	15
Figura 2.6	Gramática para a tradução de expressões pelo método junção de gramáticas.	16
Figura 2.7	Regras para a tradução de expressões pelo método junção de gramáticas.	17
Figura 2.8	Gramática para a tradução de expressões pelo método da gramática achatada.	18
Figura 2.9	Regras para a tradução de expressões pelo método da gramática achatada.	18
Figura 2.10	Gramática para a tradução de expressões pelo método da gramática única.	20
Figura 2.11	Regras para a tradução de expressões pelo método da gramática única.	20
Figura 2.12	Visão Geral da Tradução de DDL para C++.	24
Figura 2.13	Visão da transformação do programa em L_e em programa equivalente em C++.	26
Figura 2.14	Visão completa do sistema de tradução baseado em semântica.	28
Figura 2.15	Exemplo de tradução de CHILL para C++.	30
Figura 3.1	Tupla de árvores representando um trecho de derivação.	35
Figura 3.2	Exemplo de Multigrafo.	36
Figura 3.3	Categoria D .	37
Figura 3.4	Categoria D' com símbolo inicial.	38
Figura 3.5	Gramática G_{EXP} .	39
Figura 3.6	Categoria A .	40
Figura 3.7	Floresta associada à linguagem gerada por G_{EXP} .	41
Figura 3.8	Exemplo de $U \sqsubseteq F$.	43
Figura 3.9	Parte da Floresta $F_{EXP} = L \sqsubseteq (G_{EXP})$.	44
Figura 4.1	Especificação denotacional usual.	48
Figura 4.2	Avaliação da semântica de ' let A = 10 in A * A'.	48
Figura 4.3	Resultado de $f(A_1, \dots, A_n)$.	49
Figura 4.4	Exemplo de $I(C)$.	50
Figura 4.5	Exemplo de semântica de árvores de derivação.	51
Figura 4.6	Árvore sintática de " let A = 10 in A * A".	52
Figura 5.1	Tradução do comando with preservando a extensão.	55
Figura 5.2	Tradução do comando with preservando a intenção.	56
Figura 5.3	Tradução da inicialização de array de CHILL.	56
Figura 5.4	Tradução da inicialização de array preservando a semântica de árvores.	57

Figura 5.5	Tradução do tratador de exceções a nível de classe em DDL	57
Figura 5.6	Tradução do comando while usando if e goto .	59
Figura 5.7	Tradução do comando finally .	59
Figura 5.8	Exemplo de comparação entre programas devido a Dijkstra.	60
Figura 6.1	ETDS que mapeia expressões infixas em pósfixas.	63
Figura 6.2	Categoria A .	65
Figura 6.3	DAG correspondente a uma derivação na qual as ocorrências de ID são igualadas.	66
Figura 6.4	Construção do morfismo correspondente ao DAG.	67
Figura 6.5	Condição para τ_+ mapear árvores não-ordenadas em árvores não-ordenadas.	74
Figura 6.6	Condição para τ_+ preservar a semântica das árvores traduzidas.	74
Figura 7.1	Tradução do comando with no P2C.	80
Figura 7.2	Tradução de funções aninhadas no P2C.	81
Figura 7.3	Tradução de strings no P2C.	82
Figura 7.4	Tradução de constantes no P2C.	83
Figura 7.5	Tradução de conjuntos no P2C.	85
Figura 7.6	Tradução de strings no P2C.	86
Figura 7.7	Tradução de funções-comando pelo F2C.	87
Figura 7.8	Tradução de funções-comando pelo ForC.	87
Figura 7.9	Tradução de <i>arrays</i> com dimensões ajustáveis no F2C.	88
Figura 7.10	Tradução de <i>arrays</i> com dimensões ajustáveis no ForC.	89
Figura 7.11	Tradução do “comando” entry no F2C.	90
Figura 7.12	Tradução do “comando” entry no ForC.	91
Figura 7.13	Tradução do go to atribuído no F2C.	92
Figura 7.14	Tradução do go to atribuído no ForC.	92
Figura 7.15	Tradução do equivalence no F2C.	93
Figura 7.16	Tradução do equivalence no ForC.	94
Figura 7.17	Tradução de equivalence junto com common no F2C.	95
Figura 7.18	Tradução de equivalence junto com common no ForC.	96
Figura 8.1	Tradução de <i>records</i> variantes no P2C.	98
Figura A.1	Diagrama da adjunção $(\eta_{\square}, L_{\square}, U_{\square})$.	103
Figura A.2	Construção do morfismo m do passo indutivo.	105
Figura A.3	Diagrama da adjunção $(\eta_{+/\square}, L_{+/\square}, U_{+/\square})$.	109
Figura A.4	Exemplo de substituição de regra para um STDS simples.	111
Figura A.5	$\tau_{\square} = \eta$, morfismo de $\mathbf{Cat}_{\square}^{\Sigma}$ com $\eta(*)$ não injetiva.	114
Figura A.6	$\tau_{\square} = \eta$, morfismo de $\mathbf{Cat}_{\square}^{\Sigma}$ com $\eta(*)$ injetiva.	114
Figura A.7	Construção do morfismo m_1 do passo indutivo.	117
Figura A.8	Construção do morfismo m_1 do passo indutivo.	120
Figura A.9	Diagrama da adjunção $(\eta_{T/\square}, L_{T/\square}, U_{T/\square})$.	122
Figura A.10	Construção do morfismo m_1 do passo indutivo.	125
Figura A.11	Construção do morfismo m_1 do passo indutivo.	129
Figura A.12	Diagrama da adjunção $(\eta_{+/T}, L_{+/T}, U_{+/T})$.	131

