

Juliana Carpes Imperial

**Técnicas para o Uso do  
Cálculo de Hoare em PCC**

**DISSERTAÇÃO DE MESTRADO**

**DEPARTAMENTO DE INFORMÁTICA**  
Programa de Pós-Graduação em Informática

Rio de Janeiro  
Agosto de 2003



**Juliana Carpes Imperial**

## **Técnicas para o Uso do Cálculo de Hoare em PCC**

### **Dissertação de Mestrado**

Dissertação apresentada como requisito parcial para obtenção do título de Mestre pelo Programa de Pós-Graduação em Informática da PUC-Rio.

Orientador: Prof. Edward Hermann Haeusler

Rio de Janeiro, agosto de 2003



**Juliana Carpes Imperial**

## **Técnicas para o Uso do Cálculo de Hoare em PCC**

Dissertação apresentada como requisito parcial para obtenção do título de Mestre pelo Programa de Pós-Graduação em Informática da PUC-Rio. Aprovada pela Comissão Examinadora abaixo assinada.

**Prof. Edward Hermann Haeusler**  
Orientador  
PUC-Rio

**Prof. Paulo Augusto Silva Veloso**  
COPPE/UFRJ

**Prof. Christiano de Oliveira Braga**  
UFF

**Prof. Mario Roberto F Benevides**  
COPPE/UFRJ

**Prof. Ney Dumont**  
Coordenador Setorial do Centro Técnico Científico - PUC-Rio

Rio de Janeiro, 14 de agosto de 2003

Todos os direitos reservados. É proibida a reprodução total ou parcial do trabalho sem autorização da universidade, da autora e do orientador.

### **Juliana Carpes Imperial**

Graduou-se em Engenharia de Computação na PUC-Rio (Pontifícia Universidade Católica do Rio de Janeiro) em julho de 2001. Sua área de pesquisa é a de Teoria da Computação, tendo participado de alguns congressos e seminários em sua área de atuação.

### Ficha Catalográfica

Imperial, Juliana Carpes

Técnicas para o uso do cálculo de Hoare em PCC / Juliana Carpes Imperial; orientador: Edward Hermann Haeusler. – Rio de Janeiro: PUC-Rio, Departamento de Informática, 2003.

137 f. : il. ; 30 cm

Dissertação (mestrado) – Pontifícia Universidade Católica do Rio de Janeiro, Departamento de Informática

1. Informática – Teses 2. Cálculo de Hoare. 3. Programação (Computadores). 4. Programas de computador – Medidas de segurança. 5. Lógica computacional. 6. Invariantes de Loops. I. Haeusler, Edward Hermann. II. Pontifícia Universidade Católica do Rio de Janeiro. Departamento de Informática. III. Título.

DDD: 004

## **Agradecimentos**

A Deus por ter me permitido concluir esse trabalho e ter me dado a chance de possuir todo o conhecimento que tenho hoje.

Aos meus pais, por durante muitos anos terem investido em mim para que eu pudesse estudar nas melhores instituições de ensino e, desta forma, adquirisse as ferramentas necessárias para facilitar meu aprendizado.

Ao Hermann, meu orientador, por sempre ter sido atencioso comigo tanto na elaboração deste trabalho quanto durante as aulas nas quais ele foi meu professor, sempre disposto a tirar minhas dúvidas e por ter me ensinado boa parte do que sei em minha área de pesquisa.

A todos os professores do departamento de Informática, por tudo o que me ensinaram e por toda atenção que sempre me dedicaram.

A todos os funcionários do departamento de Informática, por sempre estarem dispostos a ajudar e pela simpatia.

À CAPES, à FAPERJ e à PUC-Rio, pelos auxílios concedidos, sem os quais este trabalho não poderia ter sido realizado.

Aos professores que participaram da Comissão Examinadora.

Aos meus colegas e amigos do TECMF que sempre me apoiaram e ajudaram quando eu precisava.

A todos os meus colegas e amigos de pós-graduação que conviveram comigo, em especial para Carlos de Salles, Claudio de Biasi, Elton Silva, Fabio Marcos, Renato Maia, Ronaldo Luiz e Sergio de Biasi.

Aos meus gatos Tigresa, Billy e Nicky que sempre me fizeram companhia durante os meus estudos e na elaboração desta dissertação, especialmente a primeira, que muitas vezes ficava no meu colo durante tais tarefas pedindo carinho.

## Resumo

Imperial, Juliana Carpes. **Técnicas para o Uso do Cálculo de Hoare em PCC**. Rio de Janeiro, 2003. 135p. Dissertação de Mestrado - Departamento de Informática, Pontifícia Universidade Católica do Rio de Janeiro.

Atualmente, a maioria dos programas para computadores é obtida através da WEB. Como muitas vezes a procedência são fontes desconhecidas, é preciso se certificar de que o código se comporta como o esperado. A solução ideal seria verificar o código contra uma especificação de políticas de segurança, contudo, isso pode consumir muito tempo. Uma outra alternativa é fazer com que o próprio código prove ser seguro. O conceito de *proof-carrying code* (PCC) é baseado nessa idéia: um programa carrega consigo uma prova de sua conformidade com certas políticas de segurança. Ou seja, ele carrega uma prova a respeito de propriedades do próprio código. Portanto, os mesmos métodos formais usados para a verificação de programas podem ser utilizados para esta tecnologia. Considerando este fato, neste trabalho é estudado como o cálculo de Hoare, um método formal para realizar a verificação de programas, aplicado a códigos-fonte escritos em uma linguagem de programação imperativa, pode ser útil à técnica de PCC. Conseqüentemente, são pesquisados métodos para a geração de provas de correção de programas utilizando o método citado, para tornar possível a geração de provas de segurança para PCC utilizando o cálculo de Hoare.

## Palavras-chave

*Proof-Carrying code*; Cálculo de Hoare; Correção de Programas; Segurança; Lógica de Primeira Ordem; Invariantes de Loops; Prova de Teoremas; Prolog

## Abstract

Imperial, Juliana Carpes. **Techniques for the Use of Hoare Logic in PCC.** Rio de Janeiro, 2003. 135p. MSc. Dissertation - Departamento de Informática, Pontifícia Universidade Católica do Rio de Janeiro.

Nowadays most computer programs are obtained from the WEB. Since their source is usually unknown, it is necessary to be sure that the code of the program behaves as expected. The ideal solution would be verify the code against a specification of safety policies. However, this can take too much time. Another approach is making the code itself prove that it is safe. The concept of proof-carrying code (PCC) is based on this idea: a program carries a proof of its conformity with certain safety policies. That is, it carries a proof concerning properties related to the code itself. Therefore, the same formal methods employed in formal verification of programs can be used in this technology. Due to this fact, in this work it is studied how Hoare logic applied to source codes written in an imperative programming language, which is a formal method to verify programs, can be useful to PCC. Consequently, methods are researched to generate proofs of program correctness using the method explained, so that it can be possible to generate PCC safety programs with Hoare logic.

## Keywords

Proof-Carrying code; Hoare Logic; Program Verification; Safety; First Order Logic; Loops' Invariants; Theorem Proving; Prolog

# Sumário

1	Introdução	12
1.1	Motivação	12
1.2	Objetivos	12
1.3	Outras Abordagens ao Problema da Segurança	13
1.4	Trabalhos Relacionados	14
1.5	Organização do Texto	17
2	PCC	19
2.1	O que é PCC?	19
2.2	Vantagens do Uso de PCC	22
2.3	Aplicações de PCC	25
2.3.1	Código Móvel	25
2.3.2	Sistemas Operacionais Extensíveis	25
2.3.3	Extensões a Linguagens de Programação de Alto-Nível	25
2.4	Problemas Potenciais de PCC	26
2.4.1	O Problema do Tamanho das Provas	26
2.4.2	Outros Problemas	28
2.5	Implementando a Validação de Provas	29
2.6	PCC no Mundo Real	29
3	Cálculo de Hoare	31
3.1	Semântica Axiomática de Linguagens de Programação	31
3.2	Correção de Programas	33
3.3	Regras do Cálculo de Hoare	35
3.4	Invariantes de <i>Loops</i>	41
3.5	Sub-Rotinas e Lemas	42
3.6	PCC Utilizando o Cálculo de Hoare	42
3.7	Provando Segurança Utilizando o Cálculo de Hoare	44
4	Heurísticas para a Aplicação do Cálculo de Hoare Automaticamente	46
4.1	Introdução	46
4.2	Fortalecimentos e Enfraquecimentos Somente para os Axiomas	48



4.3 Fortalecer a Pré-Condição antes de Enfraquecer a Pós-Condição	51
4.4 Fazer a Pós-Condição de uma Atribuição ser a Mais Forte Possível	52
4.4.1 $\{ P \} x := E \{ Q \}$	53
4.4.2 $\{ P \} x := E(x) \{ Q \}$	54
4.4.3 $\{ P(x) \} x := E \{ Q \}$	56
4.4.4 $\{ P(x) \} x := E(x) \{ Q \}$	58
4.5 Começar de Trás para Frente	61
4.6 Correção do Comando <b>if</b> Quando a Pós-Condição é Desconhecida	62
4.7 Problemas com Vetores	63
4.8 Diminuição do Tamanho das Sentenças e Asserções	65
5 Implementação do Corretor de Programas	67
5.1 Introdução	67
5.2 A Sintaxe da Linguagem e o Formato das Asserções	67
5.3 Implementação das Regras do Cálculo de Hoare	69
5.3.1 Comando <b>skip</b>	69
5.3.2 Comando de atribuição	70
5.3.3 Comandos <b>if-then</b> e <b>if-then-else</b>	75
5.3.4 Comando <b>while</b>	76
5.3.5 Seqüência de Comandos	77
5.3.6 Fortalecimento da Pré-Condição e Enfraquecimento da Pós-Condição	78
5.4 Substituição de Variáveis	78
6 Outras Abordagens para a Verificação Formal	81
6.1 Introdução	81
6.2 Álgebra de Kleene com Testes	81
6.2.1 Definição	81
6.2.2 AKT e Cálculo de Hoare Proposicional	82
6.2.3 Comparação Entre as Duas Abordagens	84
6.3 Lógica Dinâmica	84
6.3.1 Definição	84
6.3.2 Comparação Entre o Cálculo de Hoare e a Lógica Dinâmica	85
7 Conclusões	87
7.1 Decisões de Projeto	88
7.2 Trabalhos Futuros	89

8 Referências Bibliográficas	91
Apêndice: Código Fonte do Corretor de Programas	98
lexica.pl	98
sintatica.pl	104
condicoes.pl	112
expressoes.pl	120
erro.pl	123
hoare.pl	124
principal.pl	129
html.pl	130

## Lista de figuras

Figura 1 - <i>proof-carrying code</i>	22
Figura 2 - Estrutura da regra do <b>while</b>	38
Figura 3 - Redução da regra do <b>if-then</b> a do <b>if-then-else</b>	39
Figura 4 - Verificação de propriedades de segurança	45
Figura 5 - Forma dos ramos em uma prova normal	46
Figura 6 - Fortalecimento da pré-condição para uma seqüência de comandos	49
Figura 7 - Enfraquecimento da pós-condição para uma seqüência de comandos	49
Figura 8 - Fortalecimento da pré-condição para um comando <b>if-then-else</b>	49
Figura 9 - Enfraquecimento da pós-condição para um comando <b>if-then-else</b>	50
Figura 10 - Colocando o invariante na pré-condição e na pós-condição	50
Figura 11 - Fortalecer a pré-condição antes de enfraquecer a pós-condição	51
Figura 12 - Equivalência de provas do comando <b>skip</b>	52
Figura 13 - Prova de $\{ P \} x := E \{ P \dot{\cup} x = E \}$	53
Figura 14 - Prova de $\{ y = 0 \} x := a \{ y = 0 \text{ and } x = a \}$	53
Figura 15 - Prova de $\{ \text{true} \} x := E \{ x = E \}$	54
Figura 16 - Prova de $\{ P \} x := E(x) \{ P \}$	54
Figura 17 - Prova de $\{ \text{true} \} x := x * x \{ x \cong 0 \}$	55
Figura 18 - Prova de $\{ P \} x := E(x) \{ P \dot{\cup} \{ \exists y ( x = E(y) ) \}$	55
Figura 19 - Prova de $\{ \text{true} \} x := E(x) \{ \exists y ( x = E(y) ) \}$	56
Figura 20 - Prova de $\{ \text{true} \} x := x * x \{ \exists y ( x = y * y ) \}$	56
Figura 21 - Prova de $\{ P(x) \} x := E \{ P(a) \dot{\cup} x = E \}$	57
Figura 22 - Prova de $\{ x = b \dot{\cup} y = x \} x := a \{ y = b \dot{\cup} x = a \}$	57
Figura 23 - Prova de $\{ P(x) \} x := E \{ x = E \}$	57
Figura 24 - Prova de $\{ P(x) \} x := E \{ \exists a P(a) \dot{\cup} x = E \}$	58
Figura 25 - Prova de $\{ x = b \} x := a \{ \exists c ( c = b ) \dot{\cup} ( x = a ) \}$	58
Figura 26 - Prova de $\{ P(x) \} x := E(x) \{ P(a) \dot{\cup} x = E(a) \}$	59
Figura 27 - Provas de $\{ x = b \dot{\cup} y = x \} x := a + x \{ y = b \dot{\cup} x = a + y / b \}$	59
Figura 28 - Prova de $\{ P(x) \} x := E(x) \{ x = E(a) \}$	59
Figura 29 - Prova de $\{ P(x) \} x := E(x) \{ \exists a ( P(a) \dot{\cup} x = E(a) ) \}$	60
Figura 30 - Prova de $\{ x = b \} x := a + x \{ \exists c ( c = b \dot{\cup} a + x = a + c ) \}$	60
Figura 31 - Corrigir os ramos do <b>if</b> de trás para frente	62

Figura 32 - Correção do <b>if-then-else</b> para determinar sua pós-condição	63
Figura 33 - Correção do <b>if-then</b> para determinar sua pós-condição	63
Figura 34 - Prova que não pode ter a correção iniciada de trás para frente	64