Métodos Formais Algébricos para Geração de Invariantes

Rachid Rebiha[1], **Arnaldo Vieira Moura**[1]

[1]Instituto de Computação – Universidade Estadual de Campinas UNICAMP, SP – Brazil

{rachid,arnaldo}@unicamp.br

Abstract. *It is well-known that the automation and effectiveness of formal verification of software, embedded or hybrid systems depends to the ease with which precise invariants can be automatically generated from source code or specifications. In this thesis, we first reduce the invariant generation problem to linear algebraic problem in order to present new computational methods that can automate the discovery of non linear invariants, thereby circumventing difficulties met by other recent techniques. We provide the first methods which generate bases of non linear invariants, while dealing with complex programs and non linear hybrid embedded systems similar to those present today in many critical systems. These methods give rise to more efficient algorithms, with much lower computational complexities than the mathematical foundations of previous approaches know today. To the best of our knowledge, we present the first verifications methods that automatically generates bases of invariant expressed by transcendental functions. Finally, we extend the domain of applications for invariant generation methods to encompass security problems. More precisely, we provide an extensible invariant-based platform for analysis and detection of malware and the most virulent intrusions attacks.*

Resumo. *É bem sabido que a eficácia de métodos de verificação formal de sistemas embarcados, ou sistemas híbridos, depende da geração automática e eficiente de invariantes precisas. Nesta tese propusemos , primordialmente, a redução de problemas de geração de invariantes para problemas algébricos lineares. Logrando esta contribução, conseguimos ultrapassar as deficiências dos mais modernos métodos de geração de invariante hoje conhecidos, permitindo, assim, a geração eficiente de invariantes para software, programs complexos, sistemas híbridos e embarcados não lineares. Os algoritmos algébricos que propusemos apresentam complexidades significativamente inferiores aqueles que suportam as demais abordagens usadas hoje. Conseguimos fundamentar técnicas inéditas e eficientes que venham a formar o núcleo de novos algoritmos para a automação de processos dedutivos, e de novos algoritmos para verificação de modelos complexos que vão além dos limites alcançados hoje por outros métodos. Em particular, desenvolvemos métodos pioneiros que automaticamente gerem bases de invariantes expressas por séries de potências multivariáveis e por funções transcendentais. Finalmente, estendemos o domínio de aplicações, acessíveis através de métodos de geração de invariantes, para a área de segurança fornecendo uma plataforma extensível baseada em invariantes pré-computadas para análise e deteção dos ataques de intrusões mais virulentos.*

1. Introdução

Métodos Formais visam descobrir técnicas matemáticas, bem como desenvolver seus algoritmos, para verificar a correção de software, de hardware, de sistemas concorrentes, e de sistemas embarcados ou híbridos. Ou seja, visa provar que os sistemas considerados

são fiéis às suas especificações. Métodos de *análise estática* atacam a parte central do problema, qual seja, a geração de propriedades invariantes do sistema sob análise (sem executar o sistema). Invariantes são afirmações que devem ser verdadeiras, em um local específico e a cada possível execução do sistema sob consideração. Na maioria dos métodos para verificação formal hoje em uso, o problema de estabelecer a desejada propriedade pode ser reduzido à procura de invariantes que fortaleçam o que queremos provar. Na verdade, o problema da verificação de propriedades de tipo seguras ("safety")–tais como a não diferenciação de apontador nulo, o extravazamento de "buffers", vazamento de memória, ou índices de vetores fora de escopo—pode ser reduzido ao problema da *geração de invariantes*. Invariantes são também essenciais para provar e estabelecer propriedades de tipo permanentes ("liveness")–tais como progresso ou terminação. Além disso, as técnicas padrão para a verificação de programas usam assertivas invariantes para provar propriedades do programa em foco. Portanto, métodos e técnicas para geração automática de invariantes possibilitarão a automação da análise estática e, como consequencia, permitirão também automatizar a geração de coberturas para casos de teste, que é outro problema muito importante para a construção de sistemas complexos confiáveis.

2. Resumo da Tese

Na academia e na industria, já foi claramente estabelecido que o tratamento de sistemas não-lineares é atualmente um gargalo crítico para realizar verificação automática de programas e de sistemas híbridos críticos. Necessitamos, sem dúvida, de novas técnicas simbólicas, associadas a algoritmos numéricos rápidos para lidar com tais sistemas não-lineares. É bem sabido que a automação e a eficácia de métodos de verificação formal de softwares, sistemas embarcados ou sistemas híbridos, depende da facilidade com que invariantes precisas possam ser geradas automaticamente a partir do código fonte. Uma invariante é uma propriedade, especificada sobre um local específico do código fonte, e que sempre se verifica a cada execução de um sistema. Apesar dos progressos enormes ao longo dos anos, o problema da geração de invariantes ainda está em aberto para tanto programas não-lineares discretos, como para sistemas não-lineares híbridos. Primeiramente, apresentamos novos métodos computacionais que podem automatizar a descoberta e o fortalecimento de relações não-lineares entre as variáveis de um programa que contém laços não-lineares, ou seja, programas que exibem relações polinomiais multivariadas e manipulações fracionárias. Inicialmente, vamos examinar alguns exemplos.

Example 1 [Geração de Invariantes para Logica com Aritmetica Não-linear]. *Considere o programa simples abaixo à esquerda:*

```
1. int x, y, y_0;                Invariant Generation
2. (x=1)&&(y=y_0) //initial.     [Logic: Non-linear Arith.]
3.  While (...){                 ========================
4.    x := x*y+x;                Basis of invariant Ideal:
5.    y := y^2; }                {x^2, x*y-x, y^2-2*y+1}
```

As instruções de laço (linhas 6 e 5) induzem um sistema não linear, polinomial e discreto, o qual pode ser representado pela relação de transições no autômato. Se apenas um laço não linear deste tipo aparece no código, técnicas recentes para geração de invariantes retornariam uma resposta inconclusiva, mesmo usando técnicas que lidam com números elevados de linhas de código. Podemos calcular, em tempo polinomial, as bases de um espace vectorial descrevendo invariantes indutiva não triviais, como representado à direita no exemplo acima. Em outras palavras, para todos os polinômios G_1, G_2, $G_3 \in \mathbb{R}[x, y]$,

temos que $G_1(x,y)(x^2) + G_2(x,y)(xy - x) + G_3(x,y)(y^2 - 2y + 1) = 0$ *é uma invariante indutiva. Por exemplo, considere o passo inicial* $(y = y_0, x = 1)$*. Uma invariante possível seria* $y_0(1 - y_0)x^2 + xy - x + y^2 - 2y + 1 = 0$*.* □

Example 2 [Geração de Invariantes com Funções Não Interpretadas]. *Seja o trecho de código com chamadas de função não interpretadas, sendo que poderíamos ter quaisquer outras chamadas de sistemas ou chamadas de funções com assinaturas semelhantes. Como exemplo, podemos gerar invariantes para este código como mostradas na coluna à esquerda.*

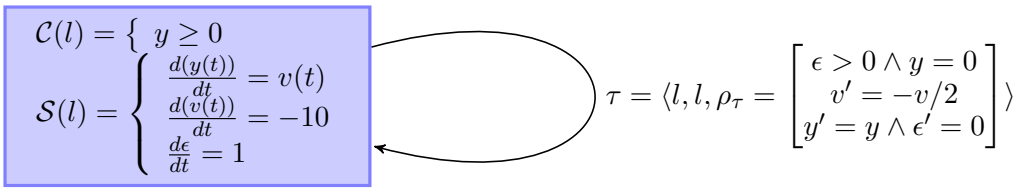
```
1  void Func_File_Descriptor(int fd_1,int fd_2){
2   int t1,t2,t3,t4,t5,a;
3    if (fd_1==fd_2) {t1=Func_F(fd_1-fd_2);
4       t2=Func_F(Func_F(0));}
5    else {t2=Func_F(t1);}
6    t3=Func_F(t1); a=fd_2-1;
7    if (fd_1==a){t4=Func_G(fd_1,2*fd_1+1);
8       t5=Func_G(fd_1,Func_G(fd_1,2*fd_2-1));}
9    else {t5 = Func_G(fd_1, t4);}}
```

```
Invariant Generation
[Logic: Uninterpreted Function]
==============================
t3 = t2
fd_2 + -1.000000*a = 1.000000
t2 = Func_F(t1)
t5 = Func_G(fd_1, t4)
```

Se interpretarmos `Func_F` *e* `Func_G`*, como sendo duas chamadas comuns,* `dup()` *and* `dup2()`*, nós imediatamente obtemos uma invariante similar, na forma* $(r3 = r2) \wedge (fd\_2 - a = 1) \wedge (r2 = dup(r1)) \wedge (r5 = dup2(fd\_1, r4))$*.* □

Sistemas híbridos exibem comportamentos discretos e contínuos. Este tipo de modelo é freqüentemente encontrado quando se lida com sistemas digitais embutidos em ambientes analógicos. A maioria dos sistemas que apresentam aspectos críticos de segurança—tais como aviões, automóveis, produtos químico, usinas de energia e sistemas biológicos—operam semanticamente como se fossem sistemas híbridos não lineares. Como tal, só podem ser adequadamente modelados por meio de logicas com aritmética não linear sobre os reais, e envolvendo polinômios multivariados e funções transcendentais.

Example 3 [Sistemas Híbridos] *O sistema dinâmico de uma bola quicando pode ser modelado pelo autômato híbrido a seguir.*



É fácil verificar que a sentença $5\epsilon^2 + v \times \epsilon - y = 0$ *é uma invariante indutiva.* □

Nesse trabalho, apresentamos poderosos métodos computacionais que são capazes de gerar bases de ideais polinomiais de invariantes não-lineares para sistemas híbridos não-lineares. Fomos capazes de tratar sistemas híbridos complexos com regras diferenciais não lineares (expressas por funções trigonométricas, polinomiais, ...). Por exemplo, fomos capazes de lidar com a verificação de modelagem de sistemas de gestão de tráfego aéreo (quando se quer mostrar que aviões sigam trajetórias elípticas com uma distância não-nula entre eles, por pelo menos algum lapso de tempo), de sistemas embarcado em veículos autônomos, do metabolismo da glicose do sangue humano (paciente diabético do tipo I), ... Considerando o estado da arte previa deste trabalho, as principais abordagens

estavam limitados a sistemas lineares e foram compostas por quatro etapas principais, onde os tres primeiros passos eram duplamente exponenciais e o ultimo passo era quasi impossivel na pratica (resolução de sistems não lineares e parametrica ...). Por outro lado, nossos algoritmos só tenho três passos de complexidade polinomiais.

Em segundo lugar, apresentamos métodos pioneiros de verificação que automaticamente gerem bases de invariantes expressas por séries de potências multi-variáveis e por funções transcendentais. Discutimos, também, a sua convergência em sistemas híbridos que exibem modelos não lineares. Verificamos que as séries de potência geradas para invariantes são, muitas vezes, compostas pela expansão de algumas funções transcendentais bem conhecidas, tais como "log" e "exp". Assim, apresentam uma forma analisável fechada que facilita o uso de invariantes na verificação de propriedades de segurança. É importante ressaltar que não há outros métodos conhecidos que sejam capazes de gerar este tipo de invariantes, ou que possam lidar com este tipo de sistema. Para cada problema de geração de invariantes estabelecemos condições suficientes, muito gerais, que garantem a existência e permitem o cálculo dos ideais polinomiais para situações que não podem ser tratadas pelas abordagens de geração invariantes hoje conhecidas.

Example 4 [Invariantes Transcendentais] *Considere as seguintes regras diferenciais* $\{\frac{d(x(t))}{dt} = ax(t)\ ;\ \frac{d(y(t))}{dt} = ay(t) + bx(t)y(t)\}$ *que podem ocorrer em um modo contínuo de um sistema híbrido. Fomos capazes de computar bases de espaços vetoriais para séries formais invariantes na forma:* { x, exp(-bx/a)y }. *Por exemplo, tomando o valor inicial* (x_0, y_0)*, a seguinte sentença* $x(e^{-bx_0/a})y_0 - x_0(e^{-bx/a})y = 0$ *é uma invariante indutiva, quaisquer que sejam as condições iniciais, ou seja, para todos os valores de* x_0 *e de* y_0*. Também, podemos gerar invariantes expressas por desigualdade.* □

Finalmente, estendemos o domínio de aplicações, acessíveis através de métodos de geração de invariantes, para a área de segurança. Mais precisamente, fornecemos uma plataforma extensível baseada em invariantes pré-computadas que seriam usadas como "assinaturas semânticas"para análise de intrusos ("malwares", trecho de código com intenções malévolas: vírus, cavalos de Tróia, worms, ...) e deteção dos ataques de intrusões mais virulentos. Seguindo a concepção de tais plataformas, propomos sistemas de detecção de intrusão, usando modelos gerados automaticamente, onde as chamadas de sistema e de funções são vigiados pela avaliação de invariantes, pré-calculadas para denunciar qualquer desvio observado durante a execução da aplicação. Considerando o estado da arte prévia deste trabalho, propusemos um sistema de detecçoes de intrusão mais precisos e somos os primeiros a introduzir os efetivos invariants de vírus como assinaturas.

De modo abrangente, nesta tese, propomos a redução de problemas de geração de invariantes para problemas algébricos lineares. Ao reduzir os problemas de geração de invariante não-triviais de sistemas híbridos não-lineares para problemas algébricos lineares relacionados, somos capazes de ultrapassar as deficiências dos mais modernos métodos de geração de invariante hoje conhecidos permitindo, assim, a geração automática e eficiente de invariantes para programas e sistemas híbridos não lineares complexos. Tais métodos algébricos lineares apresentam complexidades computacionais significativamente inferiores àquelas exigidas pelos os fundamentos matemáticos das abordagens usadas hoje, tais como a computação de bases de Gröbner, a eliminação de quantificadores e decomposições cilíndricas algébricas.

3. Publicações e Distinções Acadêmicas

A primeira subseção lista as publicações mais importantes do candidato, relacionadas à área de pesquisa em que se situa esta tese. A segunda subseção alinha algumas distinções acadêmicas e prêmios obtidas pelo candidato. A terceira subseção lista as palestras dadas

nos seminarios internacionais e convite academicos.

Publicações

1. Semantic Malware Resistance Using Inductive Invariants. The International Journal of Forensic Computer Science IJoFCS (2010), Volume 5, Number 1, pages 38-48, DOI: 10.5769/J201001005.
2. Transcendental Inductive Invariants Generation for Non-linear Differential and Hybrid Systems. 15th International ACM Conference in Hybrid Systems: Computation and Control (HSCC) and ACM/IEEE Cyber-Physical Systems CPSWeek'12. Bejing/Pequim China.
3. Generating Invariants for Non-linear Hybrid Systems by Linear Algebraic Methods. 17th Int. Static Analysis Symposium, SAS2010, Lecture Notes in Computer Science (LNCS). Perpignant France
4. Endomorphisms for Non-trivial Non-Linear Loop Invariant Generation, 5th Int. Conf. Theoretical Aspects of Computing ICTAC2008, Lecture Notes in Computer Science (LNCS). Istambul Turquia.
5. Endomorphism for Non-Trivial Semi-Algebraic Loop Invariant Generation, Technical Report, IC Unicamp, 2008.
6. Morphisms for Analysis of Hybrid Systems. ACM/IEEE Cyber-Physical Systems CPSWeek'09, Second International Workshop on Numerical Software Verification. NSV2009, Verification of Cyber-Physical Software Systems. San Francisco, California, USA, 2009.
7. Morphisms for Non-trivial Non-linear Invariant Generation for Algebraic Hybrid Systems, 12th Int. Conf. Hybrid Systems: Computation and Control (HSCC2009), Lecture Notes in Computer Science (LNCS), 2010. San Francisco, California, USA, 2009.
8. Morphisms for Non-trivial Non-Linear Invariant Generation for Algebraic Hybrid Systems. Technical Report, IC Unicamp, 2008.
9. Generating multivariate formal power series for hybrid systems. Technical Report, IC Unicamp, 2009.
10. An Ant Colony Verification Algorithm. 7th. IEEE. International Conference on Intelligent Systems Design and Applications.
11. Formal Methods for Forensic Computer Science: Automated Malware Invariant Generation, 6th International Conference on Forensic Computer Science, ICoFSC'2009 and 7th International Conference on Cyber Computer Science, **best paper award**, ICCYBER'2009.
12. Quasi-static binary analysis: guarded model for intrusion detection, Technical Report, USI Lugano – SRI International.
13. Transcendental Invariants Generation for Non-linear Differential and Hybrid Systems. Under submission to the Journal of Symbolic Computation.
14. Invariants Generation for Non-linear Hybrid Systems by Linear Algebraic Methods. Under submission to the Journal of ACM Transactions in Embedded Computing Systems (TECS).
15. Morphism for Non-Linear Loop Invariant Generation. Under submission to the Journal of Formal Aspect of Computing.

Prêmios Acadêmicos

1. Stanford Research International Fellow Award.
2. Joint Doctoral Shool Univ. Lugano – IC/Unicamp Univ. Campinas Award.
3. Prêmio de melhor artigo na conferência Internacional de segurança ICoFSC'2009 e ICCYBER'2009.

Esse trabalho foi premiada pelo Departamento de Polícia Federal do Brasil (DPF), pelo Federal Bureau of Investigation dos EUA (FBI), e pela Associação Brasileira de Especialistas em Alta Tecnologia (ABEAT).

Seminários e Palestras Convidadas

• ”Métodos Formais Algébricos para Geração de Invariantes”Phd defense at IC/Unicamp University of Campinas Brasil.
• ”Algebraic Formal Methods for Invariant Generation”. Phd dissertation presentation, University of Lugano Switzerland and Stanford Research Institute. 2011.
• Transcendental Inductive Invariants Generation for Non-linear Differential and Hybrid Systems. 15th International ACM Conference in Hybrid Systems: Computation and Control (HSCC) and ACM/IEEE Cyber-Physical Systems CPSWeek'12.
• “Generating Invariants for Non-linear Hybrid Systems by Linear Algebraic Methods”, 17th Int. Static Analysis Symposium, 2010.
• “Morphisms for Analysis of Hybrid Systems”. ACM/IEEE Cyber-Physical Systems CPSWeek'09, Second International Workshop on Numerical Software Verification. NSV2009, Verification of Cyber-Physical Software Systems.
• “Morphisms for Non-trivial Non-linear Invariant Generation for Algebraic Hybrid Systems”, 12th Int. Conf. Hybrid Systems: Computation and Control, 2009.
• ”Automated Malware Invariant Generation”, 6th International Conference on Forensic Computer Science, ICoFSC'2009 and 7th International Conference on Cyber Computer Science, ICCYBER'2009.
• 5th Int. Conf. Theoretical Aspects of Computing ICTAC2008. ”Endomorphisms for non-trivial non-linear loop invariant generation”.
• Institute of Computing UNICAMP, University of Campinas, São Paulo Brazil. ”Algebraic Formal Methods I : Invariant Generation for Program Verification and Security”.
• ”Invariant Generation for Host-Based Intrusion Detection”. USI Faculty of Informatics, University of Lugano, Switzerland, 2007.
• ”Quasi-Static Binary analysis: Guarded Model for Host-Based Intrusion Detection”. Stanford Research Institute, 2007.
• ”Program and Memory Heap Verification by Infinite Model Checking: Segmentation Fault and Memory Leak Checking with Recursive Data Structures”. Stanford Research Institute, 2007.
• ”An Extensible LTL Model Checking Library and Transition-based Generalized Büchi Automata”, Stanford Research Institute, 2007.
• ”Automatic Memory Heap Verification”, ETH, University of Zurich, Switzerland, 2007.
• ”Formal Verification with CTL* and (Propositional-) Fixed point Theory”, USI Faculty of Informatics, University of Lugano, Switzerland, 2007.

4. Programa de Dulpa Diplomação e Collaborações Internacionais

O primeiro autor iniciou o seu doutoramento como um ”Stanford Research Institute International Ph.D. Fellow”, no Stanford Research Institute (SRI), na California trabalhando no Laboratorio de Segurança e no Laboratorio de Ciencia da Computatçao, sob a supervisão do Prof. Dr. H. Saidi e do Prof. S. Dawson. Até recentemente foi doutorando do programa de ”Dupla Diplomação”, um programa conjunto operado através de convenio entre o Instituto de Computação da UNICAMP, Universidade Estadual de Campinas, no Brasil e a Faculdade de Informatica da USI, Universidade de Lugano, na Suiça. O Prof. Dr. Arnaldo Vieira Moura foi o principal orientador de doutorado junto ao Instituto de Computação, e o Prof. Dr. Fernando Pedone foi seu co-orientador, junto a Faculdade de Informatica da Universidade de Lugano, na Suiça. Durante a estada no SRI, o Prof. Dr. H. Saidi foi o co-orientador. Collaboraçoes e visitas ao Prof. Dr. N. Matringe, da Universidade de Paris VII-Denis Diderot - ”Institue de Mathématiques”, na França, foram importantes para o desenvolvimento de algumas parte das concepções teoricas dos trabalhos.