

Um Provisor de Teoremas Multi-Estratégia
A Multi-Strategy Theorem Prover

Adolfo Gustavo Serra Seca Neto
DAINF - UTFPR

<http://www.dainf.ct.utfpr.edu.br/~adolfo>

Data desta versão: 14 de novembro de 2008

Date of this version: November 14th, 2008

Sobre

Este é um capítulo da minha tese de Doutorado intitulada “Um Provador de Teoremas Multi-Estratégia”. Esta tese, na área de Ciência da Computação, foi defendida em 30 de janeiro de 2007 no Instituto de Matemática e Estatística (IME) da Universidade de São Paulo (USP). Meu orientador foi o Prof. Dr. Marcelo Finger. O texto completo desta tese está disponível em <http://www.teses.usp.br/teses/disponiveis/45/45134/tde-04052007-175943/>

About

This is a chapter of my Ph.D. thesis entitled “A Multi-Strategy Theorem Prover”. This Computer Science thesis was defended on January 30th, 2007 at the Institute of Mathematics and Statistics (IME) of the University of São Paulo (USP). My advisor was Prof. Dr. Marcelo Finger. Thesis full text is available at <http://www.teses.usp.br/teses/disponiveis/45/45134/tde-04052007-175943/>. Only the first chapter was written in Portuguese. All the following appendices were written in English.

Capítulo 1

Um Provedor de Teoremas

Multi-Estratégia

1.1 Motivação

A Dedução Automática tem sido uma área de pesquisa ativa desde os anos 50 [69]. Os esforços iniciais na área tiveram um efeito profundo no domínio da Inteligência Artificial (IA) e em toda a Ciência da Computação [70]. A Prova Automática de Teoremas (PAT) lida com o desenvolvimento de programas de computador que demonstram que alguma sentença (a conjectura) é uma consequência lógica de um conjunto de sentenças (os axiomas e as hipóteses). Sistemas de PAT são utilizados em uma grande variedade de domínios [110], como matemática, inteligência artificial, geração e verificação de software, verificação de protocolos de segurança e verificação de hardware.

A maior parte dos provedores automáticos de teoremas hoje em dia é baseada ou no princípio da resolução [100] ou no procedimento de Davis-Logemann-Loveland¹ [32]. Porém, outros métodos também podem ser utilizados. Os métodos baseados em tablôs são particularmente interessantes para a PAT por existirem em diferentes variedades e para várias lógicas [52]. Além disso, estes métodos não exigem a conversão dos problemas para a forma clausal. Tablôs podem ser utilizados para desenvolver procedimentos de prova para lógica clássica assim como para vários tipos de lógicas não clássicas, como

¹Também conhecido como procedimento de Davis-Putnam e que é uma forma restrita de resolução.

Lógica Nebulosa [68], *Residuated Logic* [71], lógicas modais e de descrição [46], lógicas sub-estruturais [30], lógicas multi-valoradas [13], e Lógicas de Inconsistência Formal [18].

As regras de inferência dos sistemas de dedução automática, em geral, e dos provadores baseados em métodos de tablôs, em particular, são tipicamente não-determinísticas. Elas dizem o que *pode* ser feito, não o que *deve* ser feito [52]. Logo, para obter um procedimento automatizável, as regras de inferência precisam ser complementadas por um outro componente, geralmente chamado *estratégia* ou *plano de busca*, que fica responsável pelo *controle* da aplicação das regras de inferência [6]. Isto é, as regras de inferência de um método de prova definem um algoritmo não determinístico para encontrar uma prova; uma estratégia é um algoritmo determinístico para encontrar provas neste método. Para cada método de prova, muitas estratégias podem ser definidas. O tamanho das provas assim como o tempo gasto pelo procedimento de prova pode variar imensamente quando são usadas estratégias diferentes.

Algoritmos não determinísticos são utilizados em diversas áreas da ciência da computação como PAT [94], sistemas de reescrita de termos [114], especificação de protocolos [59], especificação formal [81], otimização [10], reconhecimento de padrões [61], e tomada de decisões [84]. Um algoritmo é uma seqüência de passos computacionais que recebe um valor (ou conjunto de valores) como entrada e produz um valor (ou conjunto de valores) como saída [26]. Um algoritmo não determinístico é um algoritmo com um ou mais *pontos de escolha* onde várias continuações diferentes são possíveis, sem qualquer especificação de qual será escolhida. Uma execução particular de um tal algoritmo escolhe uma das continuações sempre que chega a um ponto de escolha. Portanto, diferentes caminhos de execução do algoritmo aparecem quando ele é aplicado à mesma entrada, e estes caminhos, quando terminam, geralmente produzem diferentes saídas [115].

Algoritmos não determinísticos computam a mesma classe de funções que os algoritmos determinísticos, mas sua complexidade pode ser menor. Qualquer algoritmo não determinístico (AND) pode ser transformado num algoritmo determinístico (AD), possivelmente com uma redução de eficiência exponencial em tempo. Isto é, um AD que percorre todos os caminhos de execução possíveis de um AND polinomial pode ter com-

plexidade de tempo exponencial. Um dos mais importantes problemas em aberto na pesquisa em computação atualmente é a questão “ $P=NP?$ ” [20, 21]. Informalmente, a resposta a esta questão corresponde a saber se problemas de decisão que podem ser resolvidos em tempo polinomial por um AND podem também ser resolvidos em tempo polinomial por um AD.

O problema da satisfatibilidade (SAT) para lógica proposicional clássica foi o primeiro problema declarado como NP-completo. Um problema de decisão é *NP-completo* se (1) ele está em NP, e (2) qualquer problema em NP pode ser reduzido em tempo polinomial a ele. SAT pode ser descrito como “dada uma fórmula proposicional, decida se ela é ou não satisfatível”. Muitos outros problemas de decisão, como problemas de coloração de grafos, problemas de planejamento e problemas de agendamento podem ser codificados em instâncias de SAT.

Um dentre os muitos métodos lógicos que podem ser utilizados para resolver o problema da satisfatibilidade é o sistema **KE**. É um método de tablôs originalmente desenvolvido para lógica clássica por Marco Mondadori e Marcello D’Agostino [31], mas que foi estendido para outros sistemas lógicos. O sistema **KE** foi apresentado como uma melhoria, no aspecto computacional, em relação ao sistema de tablôs analíticos [106]. Apesar de parecido com o sistema de tablôs analíticos, o sistema **KE** é um sistema refutacional que não é afetado pelas anomalias dos sistemas livres de corte [29].

Nós projetamos e implementamos **KEMS**, um provador de teoremas multi-estratégia baseado no método **KE** para lógicas proposicionais clássicas e não-clássicas. Um provador de teoremas multi-estratégia é um provador de teoremas onde podemos variar as estratégias utilizadas sem modificar o núcleo da implementação. Um provador de teoremas multi-estratégia pode ser usado com três objetivos: educacional, exploratório e adaptativo. Com fins educacionais, pode ser utilizado para ilustrar como a escolha de uma estratégia pode afetar a performance de um provador de teoremas. Como uma ferramenta exploratória, um provador de teoremas multi-estratégia pode ser usado para testar novas estratégias e compará-las com outras já existentes. Por fim, podemos ainda imaginar um provador de teoremas multi-estratégia adaptativo, que modifica a estratégia utilizada de

acordo com as características do problema que é submetido ao provador.

A versão atual do **KEMS** implementa estratégias para três sistemas lógicos: lógica clássica proposicional, **mbC** e **mCi**. **mbC** e **mCi** são lógicas paraconsistentes. As lógicas paraconsistentes podem ser usadas para representar teorias inconsistentes porém não triviais [42]. Essas duas lógicas são de uma classe especial de lógicas paraconsistentes, as Lógicas de Inconsistência Formal [18], uma família de lógicas paraconsistentes que internalizam as noções de consistência e inconsistência no nível da linguagem-objeto. Esta família de lógicas tem algumas propriedades interessantes em sua teoria de prova e foi utilizada em algumas aplicações em ciência da computação, como na integração de informação inconsistente em bases de dados [34].

1.1.1 Um Exemplo de Aplicação de Lógicas de Inconsistência Formal

Vamos exibir agora uma plausível aplicação prática (ainda não implementada nem completamente especificada) de lógicas de inconsistência formal². Em primeiro lugar apresentaremos o problema e depois a solução proposta. O problema que queremos ajudar a resolver está relacionado ao atendimento ao parto. De acordo com [93, 39], a cesariana é um tipo de parto que só deve acontecer quando há uma indicação médica correta. Porém, o que se observa na realidade é que algumas vezes ela acontece sem que haja indicação médica, seja porque é mais cômodo para o médico obstetra (que não precisa esperar por um longo trabalho de parto), seja por escolha da própria parturiente (que tem medo da dor). O problema é que nestes casos, quando realizada sem justificativa médica, a cesárea traz mais riscos do que benefícios tanto para o bebê quanto para a parturiente.

Estamos preocupados aqui apenas com o primeiro caso, que ocorre quando o médico obstetra indica uma cesariana sem que haja uma justificativa médica correta. Quando isto acontece, o médico apresenta como justificativa para a realização da operação uma série de motivos que na verdade não são suficientes para justificar a realização de uma cesariana. E algumas parturientes, geralmente por desinformação, aceitam esta indicação incorreta.

²Outras aplicações de lógicas paraconsistentes podem ser encontradas em [16]

Nestes casos, dizemos que aconteceu uma ‘cesárea desnecessária’. Diversos casos reais de situações como a descrita acima podem ser encontrados em listas de discussão sobre parto na Internet [41, 45, 35].

Para evitar que isto aconteça é necessário que as mulheres informem-se melhor sobre quais são as condições que justificam ou não a realização de uma cesárea. Os livros e listas de discussão citados acima são excelentes fontes de consulta sobre este assunto. Além destes, [44] traz uma abordagem baseada em evidências científicas sobre esta e outras questões relacionadas à gravidez e ao nascimento.

Nossa proposta aqui é um sistema para auxiliar a parturiente a tomar uma decisão informada sobre a finalização do próprio parto. Um sistema que a ajude compreender melhor o diagnóstico médico e, se for o caso, a procurar uma segunda opinião.

Suponha a seguinte situação: a parturiente recebe, antes de entrar em trabalho de parto, o diagnóstico (informal) do médico de que uma cesárea é necessária. Neste diagnóstico o médico elenca uma ou mais justificativas para a realização da cesariana. E algumas vezes (conforme relatado em [41, 45, 35]) estas justificativas não são suficientes para que uma cesárea seja realizada.

O sistema que estamos propondo atuaria da seguinte forma: a parturiente acessaria o sistema, informaria ao sistema as justificativas oferecidas pelo médico e o sistema responderia se a decisão de fazer cesárea é (a) realmente necessária, (b) se é uma decisão questionável ou (c) se a cesárea naquele caso é desnecessária.

A base de regras para o sistema seria algo como:

Bebê em posição pélvica não é indicação absoluta de cesárea.

Bebê pesando mais de 5kg e em posição pélvica é indicação absoluta de cesárea.

Bebê em posição transversa é indicação absoluta de cesárea.

Placenta prévia é indicação absoluta de cesárea.

Gravidez com mais de 40 semanas não é indicação absoluta de cesárea.

Circular de cordão não é indicação absoluta de cesárea.

⋮

Esta base de regras (que chamaremos de **BR**) pode ser representada por fórmulas

lógicas como as abaixo:

$$\begin{aligned} & \text{BEBE_PELVICO} \rightarrow \neg \text{FAZER_CESAREA} \\ & (\text{BEBE_GRANDE} \wedge \text{BEBE_PELVICO}) \rightarrow \text{FAZER_CESAREA} \end{aligned}$$

Se usarmos **BR** e as justificativas fornecidas pela parturiente ao sistema como premissas, podemos verificar a que conclusões é possível chegar. Vejamos alguns exemplos:

$$\begin{aligned} & \mathbf{BR}, \text{BEBE_PELVICO} \vdash \neg \text{FAZER_CESAREA} \\ & \mathbf{BR}, \text{PLACENTA_PREVIA} \vdash \text{FAZER_CESAREA} \\ & \mathbf{BR}, \text{BEBE_GRANDE}, \text{BEBE_PELVICO} \vdash \text{FAZER_CESAREA} \wedge \neg \\ & \quad \text{FAZER_CESAREA}. \end{aligned}$$

Em lógica clássica, a última dedução acima levaria à seguinte conclusão:

$$\mathbf{BR}, \text{BEBE_GRANDE}, \text{BEBE_PELVICO} \vdash X$$

para qualquer fórmula X , pois para quaisquer fórmulas C e X , $(C \wedge \neg C) \vdash X$ em lógica clássica. Esta característica é chamada de *explosividade* (*explosiveness*) [18]. Isto é, a partir de uma contradição ‘ A e $\neg A$ ’ tudo é derivável. Em outras palavras, as teorias contraditórias são triviais. Em lógicas de inconsistência formal (e em lógicas paraconsistentes em geral) esta característica é controlada, ou seja, nem toda teoria contraditória é trivial.

Portanto, quando se utiliza lógica clássica para a construção de uma base de conhecimento [99] como esta, se ela contiver uma contradição, todas as fórmulas seguem trivialmente desta base, o que torna extremamente difícil a construção da base. Porém, se utilizarmos um sistema lógico que tolera contradições para descrever uma base de conhecimento, esta pode produzir resultados úteis mesmo que contenha contradições. No sistema em questão, naturalmente podem aparecer contradições entre as diferentes fontes que podemos consultar para criar a base de regras do sistema. Portanto, lógicas de inconsistência formal seriam bastante adequadas para a construção desta base.

Utilizando uma lógica de inconsistência formal, um base de regras como a descrita acima não iria tornar-se inútil na ocorrência de uma contradição. Um sistema baseado em **BR** pode indicar que encontrou uma contradição, e exibir o que o levou a encontrar

esta contradição. E não deixaria de chegar a outras conclusões a partir das justificativas. Deste modo, este sistema orientaria a parturiente sobre como informar-se mais sobre a sua situação, e ela (a parturiente) teria como discutir melhor esta situação ao procurar uma segunda opinião.

Este é apenas um exemplo de um possível uso de lógicas de inconsistência formal. Pode-se pensar em outros. Não conhecemos, porém, nenhuma aplicação de lógicas de inconsistência formal em uso na prática.

1.2 Apresentação e Resumo dos Apêndices

Esta tese contém este capítulo escrito em português e vários apêndices escritos em inglês. Abaixo descrevemos cada um dos apêndices, exceto o Apêndice A que é apenas uma introdução em inglês para os outros apêndices.

1.2.1 Tablôs para Lógica Clássica e Lógicas Paraconsistentes

O Apêndice B apresenta as lógicas com as quais iremos trabalhar: lógica clássica proposicional (em inglês, *classical propositional logic* — **CPL**), **mbC** e **mCi**. As duas últimas são lógicas paraconsistentes, da família de lógicas de inconsistência formal. Em seguida, exibimos alguns sistemas de tablôs relevantes para a nossa análise: o sistema de tablôs analíticos para **CPL**, o sistema de tablôs **KE** para **CPL** e os sistemas de tablôs **KE** que desenvolvemos para **mbC** e **mCi**, entre outros. Por fim, discutimos brevemente a questão da complexidade computacional de alguns sistemas de prova.

1.2.2 Projeto e Implementação do KEMS

No Apêndice C apresentamos o projeto e a implementação do **KEMS**. Primeiramente discutimos provadores de teoremas baseados em tablôs e as idéias que estão por trás do desenvolvimento do **KEMS**. Depois exibimos algumas extensões dos métodos **KE** discutidos no Apêndice B, extensões estas cujo desenvolvimento foi motivado por questões de implementação. Em seguida apresentamos uma breve descrição do sistema, mostrando

sua arquitetura e alguns diagramas de classe. Por fim, cada uma das estratégias implementadas é discutida.

1.2.3 Avaliação do KEMS

Provedores de teoremas são geralmente comparados usando-se *benchmarks* [112]. SATLIB [102] and TPTP [111] são dois exemplos de sítios na Internet que contêm problemas para avaliar provedores de teoremas. Nós avaliamos o **KEMS** para **CPL** usando como benchmarks algumas famílias de problemas difíceis encontradas na literatura [12, 95]. Além disso, desenvolvemos algumas novas famílias especialmente para avaliar o nosso provedor em suas estratégias para lógicas paraconsistentes, para as quais não encontramos famílias de problemas na literatura. Apresentamos no Apêndice D todas estas famílias além dos resultados da avaliação.

1.2.4 Conclusão

Desenvolvemos um provedor de teoremas multi-estratégia onde podemos variar a estratégia sem modificar o núcleo da implementação. **KEMS** permite descrever várias estratégias de prova para o mesmo sistema lógico, além de permitir implementar diferentes sistemas lógicos.

Avaliamos o **KEMS** com várias instâncias de famílias de problemas (Apêndice D) e nenhuma configuração do **KEMS** obteve resultados incorretos. Algumas destas instâncias, como as de PHP e ST (veja Apêndice D) são reconhecidamente bastante difíceis de provar. Para várias destas instâncias, mesmo algumas de tamanho bem pequeno, o procedimento de busca de prova não terminou no tempo limite estabelecido para nenhum par estratégia-ordenador utilizado nos testes. Algumas instâncias de outras famílias foram difíceis apenas para alguns dos pares estratégia-ordenador.

Executamos os testes com problemas para os três sistemas lógicos implementados. Para lógica clássica proposicional, *Memory Saver Strategy* foi a melhor estratégia para a maior parte das famílias. Porém, nenhum ordenador se destacou como tendo desempenho consistentemente melhor do que os outros. E para as lógicas de inconsistência formal

(**mbC** e **mCi**) nenhuma estratégia ou ordenador se destacou. É importante observar que os resultados obtidos pelo **KEMS** para as famílias de problemas para lógicas de inconsistência formal são os primeiros resultados de avaliação para estas famílias.

Todos os resultados usados nesta tese estão disponíveis em [92]. Estas e outras conclusões, além de uma revisão das contribuições desta tese e algumas sugestões de trabalhos futuros são apresentados no Apêndice E.

1.2.5 Manual do Usuário Simplificado

No Apêndice F descrevemos o procedimento de instalação do **KEMS** (disponível em [92]), além de alguns cenários para sua utilização. Na apresentação dos cenários, as funcionalidades básicas do sistema ficam claras.

1.3 Contribuições

Os seguintes resultados foram obtidos e estão sendo apresentados nesta tese:

- desenvolvemos sistemas **KE** para duas lógicas paraconsistentes: **mbC** e **mCi**. Provamos que os dois são corretos e completos. Além disso, demonstramos que o primeiro é analítico (ver Apêndice B);
- implementamos um provador de teoremas multi-estratégia, chamado **KEMS**, de código aberto e dispendo de uma interface gráfica que permite a visualização de provas. O provador tem 10 estratégias implementadas (seis para **CPL**, duas para **mbC** e duas para **mCi**) além de 13 ordenadores de fórmulas (ver Apêndice C);
- comparamos as estratégias para lógica clássica proposicional observando os resultados da avaliação do **KEMS** com várias famílias de problemas;
- desenvolvemos sete famílias de problemas para avaliar provadores de teoremas paraconsistentes (ver Seção D.1.2) e obtivemos os primeiros benchmarks para sete destas famílias.

1.3.1 Publicações e Submissões

Resultados parciais obtidos durante a elaboração desta tese foram divulgados nas seguintes publicações:

- *Effective Prover for Minimal Inconsistency Logic* [91] – artigo sobre a implementação das estratégias do **KEMS** para **mbC** e sobre os problemas para avaliar provadores para **mbC** e outras lógicas de inconsistência formal;
- *Implementing a Multi-Strategy Theorem Prover* [89] – artigo descrevendo uma versão inicial do **KEMS**;
- *Using Aspect-Oriented Programming in the Development of a Multi-Strategy Theorem Prover* [90] – artigo contendo considerações preliminares sobre o uso de programação orientada a aspectos no desenvolvimento de provadores de teorema multi-estratégia;
- duas versões do artigo *A Multi-Strategy Tableau Prover* [87, 88], artigo este que mostra uma visão geral do **KEMS**.

E os seguintes artigos estão sendo preparados para em breve serem submetidos a conferências e/ou revistas:

- *A KE Tableau for a Logic of Formal Inconsistency* – artigo sobre a implementação das estratégias do **KEMS** para **mCi** e sobre os problemas desenvolvidos para avaliar provadores para **mCi**;
- *Implementing Backjumping in a Multi-Strategy Tableau Prover* – artigo sobre a implementação de uma estratégia com a técnica chamada ‘retrossalto’ (*backjumping*);
- *Implementing Learning in a Multi-Strategy Tableau Prover* – artigo sobre a implementação de duas estratégias com a técnica chamada ‘aprendizado’ (*learning*);
- *A KE-based Multi-Strategy Tableau Prover* – um artigo mais longo contendo os resultados desta tese.

Referências Bibliográficas

- [1] Michael Alekhnovich, Jan Johannsen, Toniann Pitassi, and Alasdair Urquhart. An exponential separation between regular and general resolution. In *STOC '02: Proceedings of the thirty-fourth annual ACM symposium on Theory of computing*, pages 448–456, New York, NY, USA, 2002. ACM Press.
- [2] Noriko Arai, Toniann Pitassi, and Alasdair Urquhart. The complexity of analytic tableaux. In *STOC '01: Proceedings of the thirty-third annual ACM symposium on Theory of computing*, pages 356–363. ACM Press, 2001.
- [3] Bernhard Beckert, Richard Bubel, Elmar Habermalz, and Andreas Roth. jTAP - a Tableau Prover in Java, February 1999. Universitat Karlsruhe. Available at <http://i12www.ira.uka.de/~aroth/jTAP/>. Last accessed, November 2006.
- [4] Bernhard Beckert and Joachim Posegga. leanTAP: Lean tableau-based deduction. *Journal of Automated Reasoning*, 15(3):339–358, 1995.
- [5] Evert W. Beth. *The Foundations of Mathematics*. North-Holland Publishing Company, Amsterdam, 1959.
- [6] Maria Paola Bonacina and Thierry Boy de la Tour. Fifth Workshop on Strategies in Automated Deduction - Workshop Programme, 2004. <http://tinyurl.com/y8dkbj>. Last accessed, November 2006.
- [7] Maria Luisa Bonet and Nicola Galesi. A study of proof search algorithms for resolution and polynomial calculus. In *FOCS '99: Proceedings of the 40th Annual*

- Symposium on Foundations of Computer Science*, page 422, Washington, DC, USA, 1999. IEEE Computer Society.
- [8] Krysia Broda, Marcello D’Agostino, and Marco Mondadori. A Solution to a Problem of Popper. In *The Epistemology of Karl Popper*. Kluwer, 1995. <http://citeseer.nj.nec.com/broda95solution.html>. Last accessed, November 2006.
- [9] S. R. Buss. Polynomial size proofs of the propositional pigeonhole principle. *Journal of Symbolic Logic*, 52:916–927, 1987.
- [10] Liming Cai. *Nondeterminism and optimization*. PhD thesis, Texas A&M University, USA, 1994.
- [11] Carlos Caleiro, Walter Carnielli, Marcelo E. Coniglio, and Joao Marcos. Two’s company: “The humbug of many logical values”. In *Logica Universalis*, pages 169–189. Birkhäuser Verlag, Basel, Switzerland, 2005. Pre-print available at <http://tinyurl.com/yb5qbz>. Last accessed, November 2006.
- [12] Alessandra Carbone and Stephen Semmes. *Graphic Apology for Symmetry and Implicitness*. Oxford University Press, 2000.
- [13] W. A. Carnielli. Systematization of the finite many-valued logics through the method of tableaux. *The Journal of Symbolic Logic*, 52:473–493, 1987.
- [14] W.A. Carnielli and J. Marcos. Ex Contradictione Non Sequitur Quodlibet. *Bulletin of Advanced Reasoning and Knowledge*, 1:89–109, 2001.
- [15] W.A. Carnielli and J. Marcos. Tableau systems for logics of formal inconsistency. *Proceedings of the International Conference on Artificial Intelligence (IC-AI’2001)*, pages 848–852, 2001.
- [16] Walter Carnielli. How to build your own paraconsistent logic: an introduction to the Logics of Formal (In)Consistency. *Proceedings of the Workshop on Paraconsistent Logic (WoPaLo)*, 2002.

- [17] Walter Carnielli, Marcelo Coniglio, and Ricardo Bianconi. *Logic and Applications: Mathematics, Computer Science and Philosophy (in Portuguese)*. Unpublished, 2005. Preliminary version. Chapters 1 to 5.
- [18] Walter Carnielli, Marcelo E. Coniglio, and Joao Marcos. Logics of Formal Inconsistency. In *Handbook of Philosophical Logic*, volume 12. Kluwer Academic Publishers, 2005. To appear. Pre-print available at <http://tinyurl.com/ybn4yw>. Last accessed, November 2006.
- [19] Walter A. Carnielli and Richard L. Epstein. *Computabilidade – Funções Computáveis, Lógica e os Fundamentos da Matemática*. Editora da Unesp, 2006.
- [20] Stephen Cook. The P versus NP problem, 2000. <http://tinyurl.com/n5thm>. Last accessed, May 2005.
- [21] Stephen Cook. The importance of the P versus NP question. *J. ACM*, 50(1):27–29, 2003.
- [22] Stephen Cook and Robert Reckhow. On the lengths of proofs in the propositional calculus (preliminary version). In *STOC '74: Proceedings of the sixth annual ACM symposium on Theory of computing*, pages 135–148, New York, NY, USA, 1974. ACM Press.
- [23] Stephen A. Cook. The complexity of theorem-proving procedures. In *STOC '71: Proceedings of the third annual ACM symposium on Theory of computing*, pages 151–158, New York, NY, USA, 1971. ACM Press.
- [24] Stephen A. Cook. A short proof of the pigeon hole principle using extended resolution. *SIGACT News*, 8(4):28–32, 1976.
- [25] W. Cook, C. R. Coullard, and G. Turán. On the complexity of cutting-plane proofs. *Discrete Appl. Math.*, 18(1):25–38, 1987.
- [26] Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein. *Introduction to Algorithms - Second Edition*. MIT Press, 2001.

- [27] Newton C. A. da Costa, Décio Krause, and Otávio Bueno. Paraconsistent logics and paraconsistency: Technical and philosophical developments. *CLE e-prints (Section Logic)*, 4(3), 2004. Pre-print available at <http://tinyurl.com/yxhon7>. Last accessed, November 2006.
- [28] Marcello D’Agostino. Are Tableaux an Improvement on Truth-Tables? Cut-Free proofs and Bivalence, 1992. Available at <http://citeseer.nj.nec.com/140346.html>. Last accessed, May 2005.
- [29] Marcello D’Agostino. Tableau methods for classical propositional logic. In Marcello D’Agostino et al., editor, *Handbook of Tableau Methods*, chapter 1, pages 45–123. Kluwer Academic Press, 1999.
- [30] Marcello D’Agostino, Dov Gabbay, and Krysia Broda. Tableau methods for substructural logics. In Marcello D’Agostino et al., editor, *Handbook of Tableau Methods*, chapter 6, pages 397–467. Kluwer Academic Press, 1999.
- [31] Marcello D’Agostino and Marco Mondadori. The taming of the cut: Classical refutations with analytic cut. *Journal of Logic and Computation*, pages 285–319, 1994.
- [32] Martin Davis, George Logemann, and Donald Loveland. A machine program for theorem-proving. *Commun. ACM*, 5(7):394–397, 1962.
- [33] Martin Davis and Hilary Putnam. A computing procedure for quantification theory. *J. ACM*, 7(3):201–215, 1960.
- [34] Sandra de Amo, Walter Carnielli, and João Marcos. A Logical Framework for Integrating Inconsistent Information in Multiple Databases. In Thomas Eiter and Klaus-Dieter Schewe, editors, *Lecture Notes in Computer Science*, volume 2284, pages 67–84. Springer-Verlag, Berlin., 2002.
- [35] Eleonora de Moraes. *Lista de discussão gestar bem interior-sp*, 2004. <http://br.groups.yahoo.com/group/gestarbeminterior-sp>. Last accessed, December 2006.

- [36] Rina Dechter. *Constraint Processing*. Morgan Kaufmann, 2003.
- [37] Luis Fariñas del Cerro, David Fauthoux, Olivier Gasquet, Andreas Herzig, Dominique Longin, and Fabio Massacci. Lotrec: The generic tableau prover for modal and description logics. In *IJCAR '01: Proceedings of the First International Joint Conference on Automated Reasoning*, pages 453–458. Springer-Verlag, 2001.
- [38] Wagner Dias. Tableaux implementation for approximate reasoning (in portuguese). Master's thesis, Computer Science Department, Institute of Mathematics and Statistics, University of São Paulo, 2002.
- [39] Simone Grilo Diniz and Ana Cristina Duarte. *Parto normal ou cesárea? O que toda mulher deve saber (e todo homem também)*. Editora da Unesp, São Paulo, 2004.
- [40] Heidi Dixon. *Automating Pseudo-Boolean Inference Within a DPLL Framework*. PhD thesis, University of Oregon, USA, Dec 2004. Available at <http://www.cirl.uoregon.edu/dixon/dixonDissertation.pdf>. Last accessed, August 2005.
- [41] Amigas do Parto. *Lista de discussão Parto Nosso*, 2003. br.groups.yahoo.com/group/partonosso. Last accessed, December 2006.
- [42] Itala M. Loffredo D'Ottaviano and Milton Augustinis de Castro. Analytical Tableaux for da Costa's Hierarchy of Paraconsistent Logics $C_n, 1 \leq n \leq \omega$. *Journal of Applied Non-Classical Logics*, 15(1):69–103, 2005.
- [43] Tzilla Elrad, Robert E. Filman, and Atef Bader. Aspect-Oriented Programming. *Communications of the ACM*, 44, 2001.
- [44] M. Enkin, M. Skiers, J. Nelson, C. Crowder, L. Duly, and E. Hodnett. *A guide to effective care during pregnancy and childbirth*. Oxford, UK: Oxford University Press, 2000.
- [45] Fadyinha. *Lista de discussão partonatural*, 1999. br.groups.yahoo.com/group/partonatural. Last accessed, December 2006.

- [46] Luis Fariñas del Cerro, David Fauthoux, Olivier Gasquet, Andreas Herzig, Dominique Longin, and Fabio Massacci. Lotrec: the generic tableau prover for modal and description logics. In *International Joint Conference on Automated Reasoning*, LNCS, page 6. Springer Verlag, 18-23 juin 2001.
- [47] M. Finger and R. Wassermann. Approximate Reasoning and Paraconsistency-Preliminary Report. *Proceedings of the Eighth Workshop on Logic, Language, Information and Communication (WoLLIC), Brasilia, Brazil, 2001*.
- [48] M. Finger and R. Wassermann. The universe of approximations. *Electronic Notes in Theoretical Computer Science*, 84:1–14, 2003.
- [49] M. Finger and R. Wassermann. Anytime Approximations of Classical Logic from Above. *Journal of Logic and Computation*, 2006.
- [50] M. Finger and R. Wassermann. The universe of propositional approximations. *Theoretical Computer Science*, 355(2):153–166, 2006.
- [51] Melvin Fitting. *First-order logic and automated theorem proving (2nd ed.)*. Springer-Verlag New York, Inc., 1996.
- [52] Melvin Fitting. Introduction. In Marcello D’Agostino et al., editor, *Handbook of Tableau Methods*, chapter 1, pages 1–43. Kluwer Academic Press, 1999.
- [53] Zhaohui Fu, Yogesh Mahajan, and Sharad Malik. New Features of the SAT’04 versions of zChaff, 2004. <http://www.princeton.edu/~chaff/zchaff/sat04.pdf>. Last accessed, September 2005.
- [54] Dov M. Gabbay. *Labelled Deductive Systems, Volume 1*. Oxford University Press, Oxford, 1996.
- [55] Erich Gamma, Richard Helm, Ralph Johnson, and John Vlissides. *Design Patterns: Elements of Reusable Object-Oriented Software*. Addison-Wesley, 1994.

- [56] Gerhard Gentzen. Investigations into logical deductions, 1935. In M. E. Szabo, editor, *The Collected Works of Gerhard Gentzen*, pages 68–131. North-Holland, Amsterdam, 1969.
- [57] J. Gosling, B. Joy, and G. Steele. *The Java Programming Language*. Addison-Wesley, Reading, MA, 1996.
- [58] Armin Haken. The intractability of resolution. *Theor. Comput. Sci.*, 39:297–308, 1985.
- [59] Klaus Havelund and Natarajan Shankar. Experiments in theorem proving and model checking for protocol verification. In *FME '96: Proceedings of the Third International Symposium of Formal Methods Europe on Industrial Benefit and Advances in Formal Methods*, pages 662–681. Springer-Verlag, 1996.
- [60] J. Hintikka. Form and content in quantification theory. *Acta Philosophica Fennica*, 8:7–55, 1955.
- [61] Jan Holub and Borivoj Melichar. Implementation of nondeterministic finite automata for approximate pattern matching. In *WIA '98: Revised Papers from the Third International Workshop on Automata Implementation*, pages 92–99. Springer-Verlag, 1999.
- [62] Scott E. Hudson, Frank Flannery, C. Scott Ananian, Dan Wang, and Andrew Appel. *CUP Parser Generator for Java*, 1999. <http://www2.cs.tum.edu/projects/cup>. Last accessed, November 2006.
- [63] Ullrich Hustadt and Renate A. Schmidt. Simplification and backjumping in modal tableau. In *Lecture Notes in Computer Science*, volume 1397 of *Lecture Notes in Computer Science*, pages 187–201, 1998.
- [64] Gregor Kiczales, Erik Hilsdale, Jim Hugunin, Mik Kersten, Jeffrey Palm, and William G. Griswold. Getting Started with AspectJ. *Communications of the ACM*, 44:59–65, 2001.

- [65] Gregor Kiczales, Erik Hilsdale, Jim Hugunin, Mik Kersten, Jeffrey Palm, and William G. Griswold. An overview of AspectJ. *Lecture Notes in Computer Science*, 2072:327–355, 2001.
- [66] Gerwin Klein. *JFlex: the fast lexical analyzer generator for Java*, 1998. <http://jflex.de>. Last accessed, November 2006.
- [67] S. Kundu and J. Chen. Fuzzy logic or Lukasiewicz logic: A clarification. *Fuzzy Sets and Systems*, 95(3):369–379, 1998.
- [68] L. Di Lascio. Analytic fuzzy tableaux. *Soft Computing - A Fusion of Foundations, Methodologies and Applications*, 5(6):434–439, Dec 2001.
- [69] D. W. Loveland. Automated deduction: Some achievements and future directions. Technical report, National Science Foundation, 1997. Available at <http://tinyurl.com/y7mb6p>. Last accessed, May 2005.
- [70] D. W. Loveland. Automated deduction: achievements and future directions. *Commun. ACM*, 43(11es):10, 2000.
- [71] Wendy MacCaull. Tableau method for residuated logic. *Fuzzy Sets Syst.*, 80(3):327–337, 1996.
- [72] Alistair Manning, Andrew Ireland, and Alan Bundy. Increasing the Versatility of Heuristic Based Theorem Provers. In *LPAR'93*, 1993.
- [73] Heiko Mantel and Jens Otten. lintap: A tableau prover for linear logic. In *TABLEAUX '99: Proceedings of the International Conference on Automated Reasoning with Analytic Tableaux and Related Methods*, pages 217–231, London, UK, 1999. Springer-Verlag.
- [74] João Marcos. Personal communication by email, October 2006.
- [75] Fabio Massacci. Simplification: A general constraint propagation technique for propositional and modal tableaux. In *TABLEAUX '98: Proceedings of the Inter-*

- national Conference on Automated Reasoning with Analytic Tableaux and Related Methods*, pages 217–231. Springer-Verlag, 1998.
- [76] Fabio Massacci. The proof complexity of analytic and clausal tableaux. *Theor. Comput. Sci.*, 243(1-2):477–487, 2000.
- [77] William McCune and Larry Wos. Otter - The CADE-13 Competition Incarnations. *J. Autom. Reason.*, 18(2):211–220, 1997.
- [78] Elliott Mendelson. *Introduction to Mathematical Logic*. Chapman & Hall, London, UK, fourth edition, 1997.
- [79] Paulo Blauth Menezes. *Linguagens Formais e Autômatos*. Instituto de Informática da UFRGS : Editora Sagra Luzzatto, Porto Alegre, 232p., 2005.
- [80] Sun Microsystems. Java Runtime Environment (JRE) 5.0 Installation Notes, 2006. <http://java.sun.com/j2se/1.5.0/jre/install.html>. Last accessed, November 2006.
- [81] S. H. Mirian and M. Mousavi. Nondeterminism in set-theoretic specifications (in persian). In *Proceedings of Iranian Computer Society Annual Conference (CSICC'02)*, feb 2002.
- [82] David G. Mitchell, Bart Selman, and Hector J. Levesque. Hard and easy distributions for SAT problems. In Paul Rosenbloom and Peter Szolovits, editors, *Proceedings of the Tenth National Conference on Artificial Intelligence*, pages 459–465, Menlo Park, California, 1992. AAAI Press.
- [83] M. Moskewicz, C. Madigan, Y. Zhao, L. Zhang, and S. Malik. Chaff: Engineering an Efficient SAT Solver. In *Proceedings of the 38th Design Automation Conference (DAC'01)*, June 2001.
- [84] John K. Myers. An introduction to planning and meta-decision-making with uncertain nondeterministic action using 2nd-order probabilities. In *Proceedings of the first*

- international conference on Artificial intelligence planning systems*, pages 297–298. Morgan Kaufmann Publishers Inc., 1992.
- [85] Adolfo Neto. An Object-Oriented Implementation of a KE Tableau Prover, nov 2003. Available at <http://tinyurl.com/y3qmkx>. Last accessed, November 2006.
- [86] Adolfo Neto. Modifications on the implementation of a framework for tableau methods, jul 2003. Available at <http://tinyurl.com/yjgjnq>. Last accessed, November 2006.
- [87] Adolfo Neto and Marcelo Finger. A Multi-Strategy Tableau Prover. In *I Simpósio de Iniciação Científica e Pós-Graduação do IME-USP*. University of São Paulo, 2005. Available at <http://tinyurl.com/tbdd6>. Last accessed, November 2006.
- [88] Adolfo Neto and Marcelo Finger. A Multi-Strategy Tableau Prover. In *SeMe-2005. Workshop “Semantics and Meaning”*, IFIP International Federation for Information Processing. Unicamp. Campinas-SP., 2005. Available at <http://tinyurl.com/yzx8ve>. Last accessed, November 2006.
- [89] Adolfo Neto and Marcelo Finger. Implementing a multi-strategy theorem prover. In Ana Cristina Bicharra Garcia and Fernando Santos Osório, editors, *Proceedings of the V ENIA (Encontro Nacional de Inteligência Artificial), held in São Leopoldo-RS, Brazil, July 22-29 2005*, 2005. Available at <http://tinyurl.com/yd6n6n>. Last accessed, November 2006.
- [90] Adolfo Neto and Marcelo Finger. Using Aspect-Oriented Programming in the Development of a Multi-Strategy Theorem Prover. In *Anais da II Jornada do Conhecimento e da Tecnologia do Univem*, Marília-SP, 2005. Available at <http://www.ime.usp.br/~adolfo/trabalhos/jornada2005.pdf>. Last accessed, November 2006.
- [91] Adolfo Neto and Marcelo Finger. Effective Prover for Minimal Inconsistency Logic. In *Artificial Intelligence in Theory and Practice*, IFIP International Federation for Information Processing, pages 465–474. Springer Verlag, 2006. Available at <http://>

- [//www.springerlink.com/content/b80728w7m6885765](http://www.springerlink.com/content/b80728w7m6885765). Last accessed, November 2006.
- [92] Adolfo Neto and Marcelo Finger. *KEMS - A KE Multi-Strategy Tableau Prover*, 2006. <http://kems.iv.fapesp.br>. Last accessed, November 2006.
- [93] Michel Odent. *The Caesarean*. Free Association Books, 2004.
- [94] Lawrence C. Paulson. *Handbook of logic in computer science (vol. 2): background: computational structures*, chapter Designing a theorem prover, pages 415–475. Oxford University Press, Inc., 1992.
- [95] Francis Jeffrey Pelletier. Seventy-five problems for testing automatic theorem provers. *J. Autom. Reason.*, 2(2):191–216, 1986.
- [96] J. V. Pitt and R. J. Cunningham. Theorem proving and model building with the calculus ke. *Journal of the IGPL*, 4(1):129–150, 1996.
- [97] Awais Rashid and Lynne Blair. Editorial: Aspect-oriented Programming and Separation of Crosscutting Concerns. *The Computer Journal*, 46(5):527–528, 2003.
- [98] Alexandre Riazanov and Andrei Voronkov. Vampire 1.1 (system description). In *IJCAR '01: Proceedings of the First International Joint Conference on Automated Reasoning*, pages 376–380. Springer-Verlag, 2001.
- [99] M. Richardson and P. Domingos. Markov logic networks. *Machine Learning*, 62(1):107–136, 2006.
- [100] J. A. Robinson. A machine-oriented logic based on the resolution principle. *J. ACM*, 12(1):23–41, 1965.
- [101] Satisfiability suggested format, 1993. <http://www.satlib.org>. Last accessed, March 22, 2005.
- [102] Satisfiability library, 2003. <http://www.satlib.org>. Last accessed, March 22, 2005.

- [103] N. Scharli, S. Ducasse, O. Nierstrasz, and A.P. Black. Traits: Composable units of behaviour. *Proc. of ECOOP*, 2743:248–274, 2003.
- [104] J. Schumann. Tableau-based theorem provers: Systems and implementations. *Journal of Automated Reasoning*, 13(3):409–421, 1994. <http://www.springerlink.com/content/k182u80451306371>. Last accessed, November 4th, 2006.
- [105] Bart Selman, Hector J. Levesque, and D. Mitchell. A New Method for Solving Hard Satisfiability Problems. In Paul Rosenbloom and Peter Szolovits, editors, *Proceedings of the Tenth National Conference on Artificial Intelligence*, pages 440–446, Menlo Park, California, 1992. AAAI Press.
- [106] Raymond M. Smullyan. *First-Order Logic*. Springer-Verlag, 1968.
- [107] Sérgio Soares and Paulo Borba. AspectJ - Programação orientada a aspectos em Java. *Tutorial no SBLP 2002, 6o. Simpósio Brasileiro de Linguagens de Programação. 5 a 7 de Junho, PUC-Rio, Rio de Janeiro, Brasil*, pages 39–55, 2002.
- [108] R. Statman. Bounds for proof-search and speed-up in the predicate calculus. *Annals of Mathematical Logic*, pages 225–287, 1978.
- [109] Friedrich Steimann. The paradoxical success of aspect-oriented programming. *SIGPLAN Not.*, 41(10):481–497, 2006.
- [110] Geoff Sutcliffe. An overview of automated theorem proving, 2001. <http://www.cs.miami.edu/~tptp/OverviewOfATP.html>. Last accessed, March 2005.
- [111] Geoff Sutcliffe. Thousands of problems for theorem provers, 2001. <http://www.cs.miami.edu/~tptp>. Last accessed, March 2005.
- [112] Geoff Sutcliffe and Christian Suttner. The CADE ATP System Competition, 2003. <http://www.cs.miami.edu/~tptp/CASC>. Last accessed, March 2005.
- [113] Alasdair Urquhart. Hard examples for resolution. *J. ACM*, 34(1):209–219, 1987.

-
- [114] Marian Vittek. A compiler for nondeterministic term rewriting systems. In *RTA '96: Proceedings of the 7th International Conference on Rewriting Techniques and Applications*, pages 154–167. Springer-Verlag, 1996.
- [115] Wikipedia. *Nondeterministic algorithm*, 2007. <http://en.wikipedia.org/wiki/Nondeterministic>. Last accessed, February 2007.