Paraconsistência em informática e inteligência artificial

NEWTON C.A. DA COSTA e JAIR MINORO ABE

DIZ-SE QUE UMA teoria dedutiva é *consistente* se não possuir teoremas contraditórios, um dos quais, a negação do outro. Caso contrário, a teoria diz-se *inconsistente* (ou *contraditória*). Uma teoria chama-se *trivial* se todas as fórmulas (ou sentenças) de sua linguagem forem nela demonstráveis; em hipótese contrária, diz-se *não-trivial*.

Analogamente, a definição aplica-se a sistemas de proposições, sistemas de informações etc. (levando-se em conta, naturalmente, o conjunto das suas conseqüências).

Se a lógica subjacente a uma teoria T é a lógica clássica ou alguma de suas extensões, T é inconsistente se e somente se for trivial. Logo, se quisermos erigir teorias ou sistemas de informação inconsistentes mas não-triviais, temos de recorrer a um tipo novo de lógica.

Lógica *paraconsistente* é uma lógica que pode servir de base para teorias inconsistentes e não-triviais. Encontrou várias aplicações em Inteligência Artificial (IA), programação lógica etc., mostrando-se de significado básico para a ciência da computação.

Neste artigo tratamos de algumas aplicações significativas obtidas recentemente em ciência da computação e IA: ParaLog – uma linguagem de programação paraconsistente, sistemas multiagentes, representação de conhecimento (*frames*), uma nova arquitetura para ciência de computação baseada em lógica paraconsistente anotada e implementação de circuitos eletrônicos paraconsistentes.

Programação lógica paraconsistente

Inconsistências surgem naturalmente na descrição do mundo real. Isto ocorre em vários contextos. Não obstante, seres humanos são capazes de raciocinar adequadamente. A automatização de tais raciocínios requer o desenvolvimento de teorias formais apropriadas.

O emprego de sistemas lógicos que permitem a manipulação de informações inconsistentes é uma área de importância crescente em ciência de

computação, teoria de bancos de dados e IA. Por exemplo, se um engenheiro de conhecimento está projetando uma base de conhecimento *BC*, relacionado a um certo domínio *D*, ele pode consultar *n* especialistas desse domínio. Para cada especialista consultado e_j, $1 \leq j \leq n$, ele obterá alguma informação e a representará em certa lógica como um conjunto de sentenças BC_j, para $1 \leq j \leq n$. Um modo simples de se combinar o conhecimento resultante de todos os peritos em um único sistemas de conhecimento, *BC*, é unir os conjuntos obtidos BC_j:

$$BC = \bigcup_{j=1}^{n} BC_j$$

Certas bases BC_{j_1} e BC_{j_2}, porém, podem conter conseqüências contraditórias - p e $\neg p$ (negação de p). Então, *BC* é inconsistente. A base de conhecimentos *BC*, entretanto, não é um conjunto inútil de informações. Com efeito, certos subconjuntos de BC_{j_i} podem ser consistentes e expressar informações importantes. Tais informações não podem ser desconsideradas.

A discordância entre especialistas em um determinado domínio pode ser significativa. Por exemplo, se o médico M_1 conclui que o paciente *X* sofre de um câncer fatal, enquanto o médico M_2 conclui que aquele mesmo paciente sofre de tumor, mas benigno, provavelmente o paciente quererá saber mais sobre tal discordância. Essa discordância é fundamental porque pode conduzir o paciente *X* a tomar decisões apropriadas: obter a opinião de um terceiro médico, por exemplo.

Este último caso evidencia que não é aconselhável achar modos para excluir determinadas fórmulas como causando inconsistências em *BC*, pois muitas vezes podem ser removidas informações importantes. Em tais casos, a existência de inconsistências é relevante.

Embora a inconsistência seja um fenômeno de importância crescente em programação em certos ambientes – especialmente nos que possuem certo grau de distribuição – ela não pode ser manipulada, pelo menos diretamente, por meio da lógica clássica na qual estão baseadas a maioria das linguagens de programação. Assim, temos de recorrer a lógicas alternativas da clássica; torna-se então necessária a procura de linguagens de programação baseadas em tais lógicas.

Em Ávila & Abe (1999b) e Ávila *et al.* (1997b e c; 1998a e b) foi proposta uma variação da linguagem de programação Prolog, a saber,

ParaLog, que permite tratar da inconsistência diretamente. Esta implementação foi feita independentemente dos resultados de Subrahmanian e colaboradores (Blair & Subrahmanian, 1988; Subrahmanian, 1987).

Assim, a lógica paraconsistente, apesar de ter sido desenvolvida inicialmente de um ponto de vista puramente teórico, encontrou em recentes anos aplicações extremamente férteis em ciência de computação, evidenciando sua fecundidade dos prismas prático e tecnológico.

Sistema multimodal paraconsistente

Sistema multiagente constitui um tópico importante em IA. O uso de sistemas modais para modelar conhecimento e crença foi amplamente considerado em IA. Parece-nos que o primeiro a considerar conhecimento e crença com relação a máquinas foi McCarthy (1979). Subseqüentemente, Rosenchein (1985), Parikh & Ramamujam (1985), Rosenschein & Kaelbling (1986), Fischer & Inmerman (1986), Halpern & Fagin (1989), Halpern & Moses (1990), entre outros, abordaram o conhecimento em sistemas multiagentes, além de outros enfoques.

As idéias essenciais que subjazem aos sistemas propostos por Halpern & Fagin (1989), Halpern & Moses (1990) e colaboradores podem ser resumidos como se segue:

- $\Box_i A$ pode ser lido: o agente i sabe A, $i = 1, ..., n$.

Também se definem *conhecimento comum* e *conhecimento distribuído* em termos de operadores modais adicionais:

- $\Box_G$ (todos do grupo G sabem),
- C_G (é conhecimento comum entre agentes em G), e
- D_G (é conhecimento distribuído entre agentes em G) para todo subconjunto não-vazio G de $\{1, ..., n\}$.

Então, tais operadores são estudados formalmente.

Não obstante, a maioria das propostas existentes usam extensões da lógica clássica ou pelo menos parte dela, mantendo tanto quanto possível características fundamentais dessa lógica. Quando se levam em conta questões de omnisciência lógica, um conceito pertinente que aparece é o de contradição. Alguns autores consideraram este problema em detalhe, entre eles Cresswell (1970, 1972, 1973).

Outros autores têm mostrado como propriedades diferentes de conhecimento podem ser capturadas impondo-se certas condições às semânticas que permitem tais contradições (veja-se Wansing, 1990; Lipman, 1992a, b e 1994).

A vantagem de se admitir paraconsistência e paracompleteza no sistema fica evidente se observamos que alguns agentes podem efetivamente mentir de fato ou podem ignorar certas proposições: um agente pode declarar A e $\neg A$ ou que A e $\neg A$ não ocorrem.

Abe *et al.* (1998) e Abe & Akama (1999) apresentaram uma classe de lógicas paraconsistentes multimodais que também são, em geral, paracompletas e não-aléticas. Tais sistemas podem se constituir, por exemplo, em candidatos para se modelar conhecimento paraconsistente (consultar também Abe, 1994a).

Arquitetura paraconsistente – Paranet

Em Prado (1996), Prado *et al.* (1997) e Prado & Abe (1995) foram descritos especificação e protótipo de uma arquitetura paraconsistente embasada nas lógicas paraconsistentes anotadas que integra vários sistemas de computação – planejador, bancos de dados, sistemas de visão etc. de uma célula de manufaturação. Ao longo deste tópico, tais sistemas serão chamados agentes.

Em domínios de aplicação, como controle de robôs e células de manufaturação flexíveis, a complexidade da tarefa de controle aumenta proporcionalmente ao aumento e à variedade de informações, estímulos que vêm do mundo externo ao sistema.

Para se lidar com tal complexidade e para que o método usado seja adequado a essas novas situações dentro do tempo imposto pelo domínio de aplicação, a tarefa de controle não deveria ser centralizada. Tal descentralização, porém, não é fácil de se implementar: paradoxalmente, pode conduzir a um aumento no tempo para a exigência de se resolver o problema, pois pode interferir com a coerência do processo de resolução. Para evitar esse fenômeno, a arquitetura especifica:

- como cada agente vai usar seu conhecimento, planos, metas e habilidades no processo de resolução;
- como cada agente vai se comportar quando se enfrentar com informações imprecisas e incompatíveis (inconsistentes);

- como, e quando, cada agente vai passar para os outros agentes seus planos, metas, habilidades e crenças;
- como cada agente vai representar a informação recebida dos outros agentes e suas crenças nas informações.

Finalmente, a arquitetura proposta tanto engloba os sistemas de computação existentes, quanto estende tais sistemas aos mecanismos de cooperação, coordenação e manipulação de inconsistência, reduzindo o esforço necessário para se integrar os mesmos.

Unindo conceitos e técnicas de IA distribuída e lógica paraconsistente anotada, a arquitetura proposta Paranet também permite a agentes trabalharem em cooperação, ainda que na presença de dados e resultados incompatíveis, para alcançar propósito comum ou metas interativas distintas.

Em sistemas de IA distribuída, os agentes são os componentes de uma rede, e cada um deles possui apenas a sua própria percepção local do problema a ser resolvido. Em um processamento distribuído tradicional, uma intensa troca de mensagem entre os nós da cadeia é necessária, para provê-los com a informação imprescindível para o processo e controle local de cada nó. Essa intensa comunicação resulta no desempenho do sistema inteiro, e em alto nível de sincronismo no processo dos agentes.

Uma possível maneira para se reduzir a taxa de comunicação e sincronização entre agentes é lhes permitir a produção de resultados parciais, incompletos ou incorretos. Ou, até mesmo, resultados inconsistentes e/ou paracompletos, em comparação com os resultados parciais produzidos por outros agentes.

Esse tipo de processamento requer solução de arquitetura, que permite a cooperação entre agentes de tal modo que os resultados parciais de cada um deles podem ser revisados e relacionados com as informações obtidas durante a interação com os outros agentes.

Durante as últimas duas décadas, algumas arquiteturas de IA distribuída foram propostas nos campos mais diversos, variando de sinais de integração a aplicações industriais. Tais arquiteturas, porém, não tratam do conceito de inconsistência. Na maioria delas, somente os dados mais recentes são considerados durante o processo de resolução. Os dados mais antigos (indiferentemente de sua origem) que podem conduzir a inconsistências não são levados em conta. Apesar de sua importância, a idéia de inconsistência é tema ao qual a IA distribuída não deu a atenção devida.

A inconsistência e/ou a paracompleteza não podem ser tratadas diretamente por meio da lógica clássica. Então, para se manipular inconsistências e paracompletezas diretamente, deve-se empregar lógicas distintas da clássica. O Paranet foi edificado com base na lógica paraconsistente anotada com a finalidade de lidar com as inconsistências e paracompletezas nos sistemas de planejamento.

Para tornar exeqüível o uso de tal lógica em domínios de aplicação complexos (intensa informação introduz tempo crítico de resposta do agente), como na manufaturação de células, tornou-se necessário estender e refinar as técnicas e conceitos da programação lógica paraconsistente e de base de conhecimento de amálgama.

Representação de conhecimento paraconsistente por *frames*

Boas soluções para determinados problemas em ciência da computação muitas vezes dependem de uma boa representação. A escolha da representação de conhecimento é, para a maioria das aplicações em IA, até mesmo a mais difícil, além de o critério para essa escolha não ser ainda claro.

Embora não exista consenso geral algum do que seja representação de conhecimento, muitos esquemas foram propostos para representar e armazenar conhecimento. Muitos de tais esquemas têm sido utilizados de modo profícuo como fundamento para a implementação de alguns sistemas existentes. Todavia, há várias características do conhecimento que não são ainda bem entendidas, como paracompletezas e inconsistências. Até que haja melhor compreensão de tais características, a representação de conhecimento permanecerá como um campo aberto de estudo em IA.

Há vários esquemas para se representar conhecimento. Dois deles, que capturam melhor o conhecimento relativo a objetos e as suas propriedades, são as cadeias semânticas e *frames*.

O primeiro desses esquemas, as cadeias semânticas, originou-se na psicologia como resultado da *modelagem* de sistemas para a memória associativa humana. Mais recentemente, vários investigadores de ciência da computação estenderam o conceito original de cadeias semânticas para facilitar a manipulação de objetos mais complexos e suas relações. Basicamente, uma cadeia semântica é um *grafo* no qual os nós representam objetos (ou uma classe), e os vínculos mostram uma relação, geralmente binária, entre objetos ou classes conectadas pelo vínculo. Os nós podem ser de dois tipos: individuais ou genéricos. Os primeiros representam descrições ou afirmações relativas a uma instância individual de um objeto, enquanto os segun-

dos são relacionados a uma classe ou categoria de objetos. As classes são pré-ordenadas em uma taxonomia, e há vínculos que representam relações binárias especiais como *isa* – é um (do inglês *is a*) – e *ako* – um tipo de (do inglês *a kind of*). O primeiro tipo de vínculo conecta um nó individual a um nó genérico e identifica um indivíduo como pertencendo a certa classe. O segundo une dois nós genéricos entre eles e mostra que determinada classe é subdivisão de outra classe.

A segunda representação de conhecimento – *frames* – ficou popular nos anos 70 devido ao aparecimento da teoria dos *frames*, que surgiu inicialmente como resultado de um artigo escrito por M. Minsky. Um sistema de *frames*, como formulado por Minsky, consiste em coleção de *frames* articulada em uma cadeia semântica. Na ocasião, o uso de *frames* foi recomendado como básico para se entender a percepção visual, os diálogos em linguagem natural e outros conceitos complexos. O desenvolvimento de linguagens por *frames* era em parte destinada à implementação de sistemas de IA baseados em estruturas de *frames*.

Sistemas baseados em cadeias semânticas e sistemas baseados em *frames* podem ser considerados semelhantes com respeito às suas estruturas, mas diferem no que representam. Quer dizer, enquanto cadeias semânticas representam objetos simples, um sistema de *frames* pode representar objetos complexos.

Não obstante, há grandes lacunas entre o conhecimento representado via sistema baseado em *frames* e o conhecimento do mundo real. Como dissemos, a maioria desses sistemas não trata adequadamente conceitos como exceção e inconsistência.

Frame é uma representação de um objeto complexo. Ele é identificado por um nome e consiste em conjunto de *slots*. Cada *frame* possui ao menos um *frame* hierarquicamente superior e, portanto, constitui uma base com mecanismo de herança. Um *frame* especial é a raiz desta hierarquia de herança.

A hierarquia de herança é conseqüência da noção clássica de hierarquia taxonômica, modo de se organizar conhecimento. A hierarquia taxonômica é justamente o começo do raciocínio por herança. Investigadores em IA juntaram ferramentas para representar propriedades de classe, exceções para propriedades herdadas, superclasses múltiplas e conceitos estruturados com relações específicas sobre os elementos estruturais. Mais que isso, o raciocínio por herança conduz naturalmente a raciocínio *default* e raciocínio não-monotônico e pode ser usado para se raciocinar sobre protótipos e instâncias típicas de classes de sistemas de herança.

Os dois tipos principais de sistemas de herança existentes são os que não admitem exceções a propriedades herdadas, e os que admitem exceções a propriedades herdadas. É fácil descrever a semântica do primeiro tipo de herança em lógica de primeira ordem, na qual os *frames* podem ser interpretados como predicados unários, e os *slots*, como predicados binários. A descrição da semântica do segundo tipo de sistema de herança em lógica de primeira ordem é muito mais difícil, porquanto exceções introduzem não-monotonicidade.

Desde finais da década de 70, surgiram vários formalismos não-monotônicos. Entre os amplamente divulgados estão os seguintes: predicado de conclusão de Clark, a lógica do *default* de Reiter, a lógica não-monotônica de Doyle & McDermott, a circunscrição de McCarty, a lógica não-monotônica de McDermott e a lógica autoepistêmica de Moore. Porém, nenhum destes formalismos trata adequadamente os conceitos de inconsistência e paracompleteza.

Para se poder estudar as inconsistências diretamente, como já apontamos, necessitamos recorrer a lógicas alternativas, isto é, novas linguagens de programação baseadas em tais lógicas.

A variação da linguagem de programação Prolog baseada nas lógicas anotadas foi edificada e permite lidar diretamente com inconsistência e paracompleteza. A linguagem de programação proposta denominou-se paraconsistente – ParaLog.

Para se implementar sistemas de *frames* que lidam com inconsistência, teve de ser levada em conta a dificuldade causada pela falta de uma semântica formal para sistemas de *frames* paraconsistentes e para raciocínios por herança que lidam com inconsistências e sistemas de *frames* de múltipla herança.

Em Ávila (1996), Ávila & Abe (1999a) e Ávila *et al.* (1997a) encontram-se as características principais de um raciocinador de herança paraconsistente que permite lidar diretamente com exceções e inconsistências em sistemas de *frames* de múltipla herança. O raciocinador de herança paraconsistente representa conhecimento por meio de *frames* paraconsistentes e deduz com base no grau de inconsistência e/ou indeterminação. Este raciocinador, de grande amplitude de aplicação, também permite englobar heranças menos complexas. Além disso, sua característica principal é não eliminar contradições *ab initio*.

Lógica paraconsistente e inferência não-monotônica

Há vários sistemas inteligentes, incluindo os sistemas não-monotônicos. Cada sistema possui semânticas convenientes e diversas em geral. Usualmente, mais de dois raciocínios não-monotônicos são necessários em sistemas complexos inteligentes. É desejável ter-se uma semântica comum para tais raciocínios não-monotônicos. Elaborou-se uma semântica comum para os raciocínios não-monotônicos através das lógicas anotadas e programação lógica anotada, por exemplo em Nakamatsu *et al.* (1998, 1999a e b; Nakamatsu & Abe, 1999).

Circuitos eletrônicos paraconsistentes

Em Abe & Silva Filho (1998), Silva Filho (1997) e Silva Filho *et al.* (1997) descrevem-se circuitos elétricos digitais (portas lógicas complemento, conjunção e disjunção) inspirados em uma classe de lógicas paraconsistentes anotadas $P\tau$. Esses circuitos permitem *sinais incompatíveis* de maneira não-trivial em sua estrutura.

Os circuitos propostos consistem de seis estados; devido à existência de operadores literais para cada um deles, a lógica subjacente é funcionalmente completa; é multivalorada e paraconsistente (pelo menos *semanticamente*). As simulações foram feitas usando o *software Aimspice* 15.a numa freqüência típica de 50 MHz. Os *layouts* das portas foram implementados para um processo de fabricação *ES2* de 1.2 μm e, também, projetado um módulo de analisador paraconsistente (MAP) (Silva Filho, 1997) combinando vários circuitos paraconsistentes, que têm como atrativo especial tratar sinais incompatíveis e dar-lhes tratamento não-trivial.

Ao que nos consta, tais resultados parecem pioneiros na área de circuitos elétricos e, por intermédio do conceito de paraconsistência, estão se abrindo inumeráveis rotas de pesquisa na teoria de circuitos eletrônicos. Os estudos iniciais parecem ser de grande envergadura: ampliam o âmbito de aplicações nas quais sinais contraditórios são comuns, como em circuitos de sensores em robótica, circuitos de automação em indústria e em muitos outros campos. Em Silva Filho & Abe (1999a e b) estuda-se o controlador lógico Para-Fuzzy, que une características da *lógica* fuzzy e da lógica anotada paraconsistente, ou seja o Para-Fuzzy faz tratamento de conceitos difusos ou contraditórios e paracompletos, prenunciando aplicações extremamente fecundas em robótica. Um controlador lógico que trata de inconsistências é descrito em Silva Filho & Abe (1999c).

Conclusão

Como decorre da exposição precedente, as aplicações dos sistemas paraconsistentes estão sendo assaz frutíferas em muitos aspectos. Hoje, a paraconsistência converteu-se em uma área de pesquisa das mais interessantes em ciência da computação, robótica, IA e informática em geral, abrindo-se uma nova era no tocante às aplicações de lógicas não-clássicas nas ciências aplicadas.

Referências bibliográficas

ABE, J.M. *Fundamentos da lógica anotada*. São Paulo, 1992. Tese (doutoramento). Faculdade de Filosofia, Letras e Ciências Humanas da Universidade de São Paulo.

_____. On annotated modal logics. *Mathematica Japonica*, v. 40, n. 3, p. 553-560, 1994a.

_____. Annotated logics, reduced direct products, ultraproducts and ultrapowers. *Workshop on Logic, Language, Information and Computation*, Wollic 94. Recife (PE), 1994b.

_____. Curry algebras $N_{1.}$ *Atti Acc. Lincei Rend. Fis.*, s.9, v. 7, p. 125-128, 1996.

_____. A logical system for reasoning with inconsistency. 5th *Annual Meeting of SBPN'97*, Ciência e Cultura na Globalização - Novos Paradigmas, 8-10 ago. 1997. Águas de Lindóia (SP), p. 196-201, 1997a.

_____. Lógica paraconsistente e IA. *Coleção Cadernos de Estudos e Pesquisas - UNIP*, Série: Estudos e Pesquisas, n. 1-004/97. Universidade Paulista, 28p, 1997b.

_____. Some aspects of paraconsistent systems and applications. *Logique et Analyse*, 157, p. 83-96, 1997c.

_____. Uma algebrização dos sistemas anotados. Atas da 6ª *Reunião Anual da Sociedade Brasileira de Pesquisadores Nikkeis* – SBPN, p. 54-260, 1998.

ABE, J.M.; ÁVILA, B.C. & PRADO, J.P.A. Multi-agents and inconsistency. *International Conference on Computational Intelligence and Multimedia Applications 1998*, ISBN 981-023352-3, H. Selvaraj & B. Verma (eds.), World Scientific, (*ICCIMA*' 98, Proceedings of the 2nd International Conference), p. 137-142, 1998.

ABE, J.M. & AKAMA, S. A logical system for reasoning with fuzziness and

inconsistencies in distributed systems. IASTED *International Conference on Artitficial Intelligence and Soft Computing*, Honolulu, Hawaii (USA), 1999.

ABE, J.M. & SILVA FILHO, J.I. da. Inconsistency and electronic circuits. *Proceedings of the International ICSC Symposium on Engineering of Intelligent Systems* (EIS'98), v. 3, Artificial Intelligence.E. Alpaydin (ed.), ICSC Academic Press International Computer Science Conventions Canada/Switzerland, ISBN 3-906454-12-6, p. 191-197, 1998.

AKAMA, S. & ABE, J.M. Many-valued and annotated modal logics. IEEE 1998 *International Symposium on Multiple-Valued Logic* (ISMVL'98). Proceedings, p. 114-119 Fukuoka, (Japão), 1998a.

_____. Natural deduction and general annotated logics. Atas do *The First International Workshop on Labelled Deduction* (LD'98), Freiburg (Alemanha), p. 1-14, 1998b.

ÁVILA, B.C. *Uma abordagem paraconsistente baseada em lógica evidencial para T tratar exceções em sistemas de frames com múltipla herança*. São Paulo, 1996. Tese (doutoramento). Universidade de São Paulo.

ÁVILA, B.C. & ABE, J.M. Inconsistencies, exceptions, and *frame* systems (a ser publicado), 1999a.

_____. Handling inconsistencies in logic programming (a ser publicado), 1999b.

ÁVILA, B.C.; ABE, J.M. & PRADO, J.P.A. Reasoning in paraconsistent frame systems, *The Second International Workshop on CSCW in Design*, P. Siriruchatapong Z. Lin & J.P. Barthes (eds). International Academic Publishers, ISBN: 7-80003-412-7/TP.19, Bangkok (Thailand), p. 239-244, 1997a.

_____. Uma extensão da linguagem Prolog para suportar programação lógica evidencial. *Coleção Documentos, Série Lógica e Teoria da ciência*, IEA-USP, n. 29, 43p., 1997b.

_____. ParaLog-e: a paraconsistent evidential logic programming language. *XVII International Conference of the Chilean Computer Science Society, IEEE Computer Society Press,* p. 2-8, Valparaíso (Chile), Novembro, 1997c.

_____. ParaLog-e: a paraconsistent logic programming language. *International Conference on Computational Intelligence and Multimedia Applications 1998*, ISBN 981-023352-3, H. Selvaraj & B. Verma (ed.) World Scientific, *ICCIMA*' 98, Proceedings of the 2nd International Conference, p. 143-148, 1998a.

_____. A paraconsistent logic programming language. *Proceedings of the International ICSC Symposium on Engineering of Intelligent Systems* (EIS'98),

v 3, Artificial Intelligence. E. Alpaydin (ed.), ICSC Academic Press International Computer Science Conventions Canada/Switzerland, ISBN 3-906454-12-6, p.281-287, 1998b.

BLAIR, H.A. & SUBRAHMANIAN, V.S. Paraconsistent foundations for logic programming. *Journal of Non-Classical Logic*, v. 5, n. 2, p. 45-73, 1988.

COSTA, N.C.A. da. *O conhecimento científico*, São Paulo, Discurso Editorial, 1997.

COSTA, N.C.A. da; SUBRAHMANIAN, V.S. & VAGO, C. The paraconsistent logics Pt, *Zeitschr. f. Math. Logik und Grundlagen d. Math.*, 37, p. 139-148, 1991a.

COSTA, N.C.A. da; ABE, J.M. & SUBRAHMANIAN, V.S. Remarks on annotated logic. *Zeitschrift f. math. Logik und Grundlagen Math.* 37, p. 561-570, 1991b.

COSTA, N.C.A. da.; ABE, J.M.: SILVA FILHO, J.I. da; MUROLO, A.C. & LEITE, C.F.S. *Lógica paraconsistente aplicada.* São Paulo, Atlas, 1999.

CRESSWELL, M.J. Classical intensional logics. *Theoria*, 36, p. 347-372, 1970.

_____. Intensional logics and logical truth. *Journal of Philosophical Logic,* 1, p. 2-15, 1972.

_____. *Logics and languages.* London, Methuen and Co., 1973.

FISHER, M.J. & INMERMAN, N. Foundation of knowledge for distributed systems. In J.Y. Halpern (ed.), *Theoretical aspects of reasoning about knowledge. Proc. Fifth Conference*, p. 171-186. San Francisco (Cal.), Morgan Kaufmann, 1986.

HALPERN, J.Y. & FAGIN, R. Modeling knowledge and action in distributed systems, *Distributed Computing*, v. 3, n. 4, p. 159-179, 1989.

HALPERN, J.Y. & MOSES, Y. Knowledge and common knowledge in a distributed environment. *Journal of the ACM*, v. 37, n. 3, p. 549-587, 1990.

LIPMAN, B.L. Decision theory with impossible possible worlds. *Technical report Working paper*, Queen's University, 1992a.

_____. Logics for non-omniscience agents: an axiomatic approach. *Technical report Working paper 874*, Queen's University, 1992b.

_____. An axiomatic approach to the logical omniscience problem. In R. Fagin (ed.), *Theoretical aspects of reasoning about knowledge. Proc. Fifth Conference,* p. 182-196. San Francisco (Cal.), Morgan Kaufmann, 1994.

McCARTHY, J., Ascribing mental qualities to machines. *Technical report* CRL 90/10, DEC-CRL, 1979.

NAKAMATSU, K.; ABE, J.M. & SUZUKI, A. An approximate reasoning in a

framework of vector annotated logic programming, *The Vietnam-Japan Bilateral Symposium on Fuzzy Systems And Applications*, VJFUZZY' 98, Nguyen H. Phuong & Ario Ohsato (eds), HaLong Bay (Vietnam), p. 521-528, 1998.

NAKAMATSU, K. & J.M. ABE, Reasonings Based On Vector Annotated Logic Programs, atas do CIMCA'99*, International Conference on Computational Intelligence for Modelling Control and Automation*,M. Mohammadian (ed.), IOS Press – Ohmsha, ISBN 90 5199 474 5 (IOS Press), Netherlands, 396-403, 1999.

NAKAMATSU, K.; HASEGAWA, Y.; ABE, J.M. & SUZUKI, A. A framework for intelligent systems based on vector annotated logic programs. IPMM'99 *The Second International Conference on Intelligent Processing and Manufacturing of Materials*, ISBN 0-7803-5489-3, J.A. Meech, M.M. Veiga, M.H. Smith & S.R. (ed.). LeClair, IEE Catalogue Number: 99EX296, Library of Congress Number: 99-61516, Honolulu, Hawaii (USA), p. 695-702, 1999a.

NAKAMATSU, K.; ABE, J.M. & SUZUKI, A. Defeasible reasoning between conflicting agents based on VALPSN. American Association for Artificial Intelligence - AAAI'99 *Workshop on Agents' Conflicts*, ISBN 1-57735-092-8, TR WS-99-08, AAAI Press, American Association for Artificial Intelligence, Menlo Park, California (USA), p. 20-27, 1999b.

PARISH, R. & RAMAMUJAN, R. Distributed processing and the logic of knowledge. In R. Parikh (ed.), *Proc. Workshop on Logics of Programs*, p. 256-268, 1985.

PRADO, J.P.A. *Uma arquitetura em IA baseada em lógica paraconsistente*. São Paulo, 1996. Tese (doutoramento). Universidade de São Paulo.

PRADO, J.P.A.; ABE, J.M. & ÁVILA, B.C. PARANET. A paraconsistent multiagent system, *The First World Congress On Paraconsistency.* Belgium, Universiteit Gent, 1997.

PRADO, J.P.A. & ABE, J.M. Um planejador baseado em lógica paraconsistente. *2º Simpósio Brasileiro de Automação Inteligente.* Curitiba (PR), p. 177-178, 1995.

ROSENSCHEIN, S.J. Formal theories of AI in knowledge and robotics. *New Generation Computing*, 3, p. 345-357, 1985.

ROSENSCHEIN, S.J. & L.P. KAELBLING, The synthesis of digital machines with provable epistemic properties. In J.Y. Halpern (ed.), *Theoretical aspects of reasoning about knowledge: Proc. Fifth Conference*, p. 83-97. San Francisco (Cal.) Morgan Kaufmann, 1986.

SILVA FILHO, J.I. da. *Circuitos de portas lógicas primitivas implementados a partir de uma classe de lógicas paraconsistentes anotadas*. São Paulo, 1997. Dissertação (mestrado). Universidade de São Paulo.

SILVA FILHO, J.I. da & ABE, J.M. Para-fuzzy logic controller. Part I: A new method of hybrid control indicated for treatment of inconsistencies designed with the junction of the paraconsistent logic and fuzzy logic, *Proceedings of the International ICSC Congress on Computational Intelligence Methods and Applications* CIMA'99, Rochester Institute of Technology, RIT, Rochester, N.Y. (USA), ISBN 3-906454-18-5. H. Bothe, E. Oja, E. Massad & C. Haefke (eds.), ICSC Academic Press, International Computer Science Conventions, Cabada/ Switzerland, p. 113-120, 1999a.

_____. Para-fuzzy logic controller. Part II: A hybrid logical controller indicated for treatment of fuzziness and inconsistencies. *Proceedings of the International ICSC Congress on Computational Intelligence Methods and Applications* CIMA'99, Rochester Institute of Technology, RIT, Rochester, N.Y. (USA), ISBN 3-906454-18-5. H. Bothe, E. Oja, E. Massad & C. Haefke (eds.), ICSC Academic Press, International Computer Science Conventions, Cabada/ Switzerland, p. 106-112, 1999b.

_____. Para-analyser and inconsistencies in control systems. IASTED *International Conference on Artitficial Intelligence and Soft Computing*, Honolulu, Hawaii (USA), 1999c.

SILVA FILHO, J.I. da; ABE, J.M. & SANCHES, P.L. Circuitos de portas lógicas primitivas fundamentados em lógica paraconsistente anotada. *III Workshop de Ibership,* 19-21 fev. 1997*, Departamento de Ingeniería Elétrica, CINVESTAV* - IPN, México D.F. (Mexico), p. 227-237, 1997a.

_____. Circuitos de portas lógicas primitivas implementados a partir de uma classe de lógicas paraconsistentes anotadas. *Boletim Técnico da Escola Politécnica da USP*, Departamento de Engenharia Eletrônica, ISSN 1413-2206, BT/PEE/9723, 13p., 1997b.

SUBRAHMANIAN, V.S. On the semantics of quantitative logic programs, *Proc. 4th IEEE Symposium on Logic Programming.* Washington, D.C., Computer Society Press, p. 173-182, 1987.

SYLVAN, R. & ABE, J.M. On general annotated logics, with an introduction to full accounting logics. *Bulletin of Symbolic Logic*, 2, p. 118-119, 1996.

WANSING, H. A general possible worlds framework for reasoning about knowledge and belief. *Studia Logica*, v. 49, n. 4, p. 523-539, 1990.

Newton C.A. da Costa é professor do Departamento de Filosofia da Faculdade de Filosofia, Letras e Ciências Humanas da USP e da Universidade Paulista (Unip).

Jair Minoro Abe é presentemente coordenador do Grupo de Lógica e Teoria da Ciência do Instituto de Estudos Avançados da Universidade de São Paulo.