Um verificador de modelos descritos em Redes de Autômatos Estocásticos*

Claiton M. Correa, Eli Maruani, Lucas G. Oleksinski, Fernando L. Dotti, Paulo Fernandes, Afonso Sales

Faculdade de Informática – Pontifícia Universidade Católica do Rio Grande do Sul
Av. Ipiranga, 6681, Prédio 32 – 90.619-900 – Porto Alegre – RS – Brasil

{claiton.correa, eli.maruani, lucas.oleksinski}@acad.pucrs.br,
{fernando.dotti, paulo.fernandes, afonso.sales}@pucrs.br

1. Introdução

O advento da tecnologia e seu avanço nos últimos anos trouxeram para a população inúmeras facilidades, agilidade na realização de tarefas, segurança, além do aperfeiçoamento de *software* e *hardware* responsáveis pelo controle de sistemas altamente críticos, como aqueles que controlam voos de aeronaves e usinas nucleares. A essa crescente necessidade de desenvolvimento de sistemas computacionais cada vez mais complexos e robustos, faz-se necessária uma abordagem capaz de verificar a corretude dos mesmos.

A abordagem de verificação de modelos conhecida como *Model Checking* é uma técnica para verificação de sistemas concorrentes de estados finitos. Apesar desta restrição, a técnica apresenta o benefício de que toda a verificação pode ser realizada automaticamente. A mesma tem grandes vantagens sobre outras abordagens tradicionais e é baseada em simulação, testes e raciocínio dedutivo [3].

Para a técnica de verificação de modelos, é necessário que as realidades sejam descritas em um formalismo estruturado de estados finitos e as propriedades a serem verificadas sejam definidas. Redes de Autômatos Estocásticos (SAN – *Stochastic Automata Networks*) é um formalismo para modelagem de sistemas com grande espaço de estados [2]. O formalismo SAN busca prover uma forma compacta de descrever realidades complexas, modelando sistemas como um conjunto de subsistemas quase independentes que interagem ocasionalmente. Cada subsistema é composto de autômatos, onde cada autômato é composto de um ou mais estados e transições entre estes estados, que podem ser realizadas através de eventos locais e/ou sincronizantes.

A definição das propriedades a serem verificadas do modelo pode ser descrita fazendo uso de lógicas temporais, as quais permitem a descrição de sequências de transições entre estados em um sistema reativo. Neste tipo de lógica, o tempo não é mencionado explicitamente. Ao invés disto, uma fórmula da lógica temporal pode especificar que eventualmente um estado foi alcançado ou que um estado de erro nunca o foi [3].

A CTL (*Computation Tree Logic*) é uma lógica temporal baseada na lógica proposicional com noção discreta de tempo. Em outras palavras, a semântica deste tipo de lógica temporal não se baseia em uma noção linear de tempo. É definida em termos de uma árvore infinita de estados onde o futuro não é determinado [1].

O objetivo desse trabalho é o desenvolvimento de um verificador de modelos para aferir possíveis falhas de um sistema em tempo de modelagem.

*Projeto financiado pelo CNPq e FAPERGS.

2. Verificador baseado em CTL para modelos em SAN

A implementação do verificador é baseada no algoritmo de satisfação de fórmulas CTL descrita em [1] (página 348). Este algoritmo trabalha com fórmulas CTL em Forma Normal Existencial (ENF), um subconjunto de fórmulas CTL quantificadas apenas com o operador Existencial (∃).

O verificador tem como entradas um modelo descrito em SAN e propriedades descritas em fórmulas CTL. Como a entrada pode ser qualquer fórmula CTL válida, foi necessário desenvolver um *parser* para converter as fórmulas para ENF. Esta conversão utiliza regras de equivalência descritas em [1], manipulando a árvore sintática que representa a fórmula computacionalmente.

O modelo SAN de entrada é compilado, gerando a Cadeia de Markov subjacente, que contém todos os estados possíveis do mesmo. Esses estados são representados através de Diagramas de Decisão (DD), os quais permitem o armazenamento e a manipulação de funções discretas de maneira eficiente [4].

Como saída o verificador retorna um DD que mapeia o conjunto de estados do modelo satisfeitos pela propriedade verificada, podendo se extrair dele uma *testemunha*, isto é, uma execução do modelo que mostra a satisfação da fórmula. Caso contrário, deve ser gerado um *contra-exemplo* que apresenta uma sequência de computações possíveis no modelo que provam a falsidade da fórmula de entrada, sendo esta informação uma ótima fonte para depuração do sistema [3].

No estágio atual, a ferramenta já realiza a satisfação de fórmulas CTL para modelos descritos em SAN. O próximo passo é a geração de contra-exemplos e testemunhas para as fórmulas verificadas contra o modelo de entrada. Em paralelo, estão sendo realizados estudos de técnicas de otimização da ferramenta, tais como: emprego de técnicas de saturação de Diagramas de Decisão e implementação de versões paralelas do verificador.

References

[1] C. Baier and J.-P. Katoen. Principles of Model Checking. MIT Press, 2008.

[2] L. Brenner, P. Fernandes, and A. Sales. Avaliação de Desempenho de Sistemas Paralelos. In 4a. Escola Regional de Alto Desempenho (ERAD 2004), 97-120, A.C. Yamin and J.L.V. Barbosa (editors), Pelotas, RS, Brasil, 2004.

[3] E. M. Clarke, O. Grumberg, and A. D. Peled. Model Checking. MIT Press, Cambridge, Massachusetts, 1999.

[4] H. Hermmans, M. Kayser, and M. Siegle. Multi-Terminal Binary Decision Diagrams to Represent and Analyse Continuous-Time Markov Chains.