OTIMIZAÇÕES DE TÁTICAS PARA PROVA DE TEOREMAS DE GRAMÁTICA DE GRAFOS

SIMONE DRAWANZ RUTZ[1]; NATÁLIA DARLEY[2]; LUCIANA FOSS[3]; SIMONE ANDRÉ DA COSTA CAVALEHIRO[4]

[1]*Universidade Federal de Pelotas - sdrutz@inf.ufpel.edu.br*
[2]*Universidade Federal de Pelotas - ntdarly@inf.ufpel.edu.br*
[3]*Universidade Federal de Pelotas - lfoss@inf.ufpel.edu.br*
[4]*Universidade Federal de Pelotas - simone.costa@inf.ufpel.edu.br*

1. INTRODUÇÃO

Gramática de Grafos (GG) (EHRIG et al., 1997) é uma linguagem de especificação formal na qual um sistema é descrito através de grafos e morfismos entre grafos (regras). Devido ao seu caráter visual, o emprego de GG em uma especificação torna a compreensão do sistema mais intuitiva, além de permitir o uso de técnicas de verificação formal a fim de garantir propriedades específicas do sistema (RIBEIRO, 2000).

Foi proposto em (DA COSTA 2010) uma tradução de GG em Event-B (DEPLOY 2013) que é a linguagem de entrada da plataforma Rodin (ABRIAL et al. 2010). Tal tradução tem por objetivo possibilitar o uso da técnica de Prova de Teoremas (ROBINSON AND VORONKOV, 2001) para a verificação formal de GG. Ao se especificar uma GG na plataforma Rodin, obrigações de prova são geradas para garantir determinadas propriedades. Algumas destas propriedades a ferramenta é capaz de provar automaticamente, levando em consideração a própria especificação do sistema, já outras necessitam de interação com o usuário. Ou seja, em geral, utilizando tal abordagem o processo de prova é semi-automático. Para ajudar na resolução de tais obrigações de provas algumas táticas já foram propostas em (LEMOS JUNIOR. et al. 2014). Estas táticas propõem uma sequência de passos necessários para a realização de provas para determinados tipos de propriedades. Para tais táticas pretende-se analisar e propor otimizações, a fim de tornar o processo de prova interativo ainda mais simples e intuitivo. Otimizações são possíveis porque nas táticas propostas são consideradas hipóteses selecionadas automaticamente pela ferramenta, sendo algumas delas desnecessárias no desenvolvimento da prova.

A importância da proposição de táticas e de suas otimizações vem auxiliar o uso de tal abordagem (Prova de Teoremas para GG) para a verificação formal de sistemas, tornando o seu uso mais simples no processo desenvolvimento de sistemas de alta complexidade.

2. METODOLOGIA

Em Gramática de Grafos um sistema é descrito através de grafos e de regras entre grafos e seu comportamento é determinado pela aplicação destas regras ao estado atual do sistema (descrito também por um grafo).

Uma GG é composta por um grafo tipo , um grafo inicial e um conjunto de regras. O grafo tipo indica todos os possíveis tipos de vértices e arcos no sistema. O grafo inicial define o estado inicial do sistema. As regras (morfismos entre grafos) definem as possíveis mudanças que podem ocorrer no sistema. As regras são compostas de: um grafo esquerdo (L), um grafo direito (R) e um morfismo entre grafos, o qual mapeia L em R. O lado esquerdo determina os componentes que devem estar no grafo estado do sistema para que a regra seja aplicada. Uma

regra só pode ser aplicada quando existe um match, ou seja, um morfismo total do grafo esquerdo com o grafo estado. O morfismo e o lado direito especificam as modificações que devem ocorrer no grafo estado após a aplicação da regra.

A especificação em GG permite a verificação formal do sistema descrito, o que possibilita investigar a validade de propriedades do mesmo. Uma das técnicas de verificação utilizadas é a Prova de Teoremas. Seguindo a abordagem proposta para especificação de GG, que possibilita a tradução para a linguagem Event-B, é possível o uso da ferramenta Rodin para a verificação de propriedades. A especificação de uma GG em Event-B na plataforma Rodin consiste em duas partes: contexto, onde são definidos o grafo tipo e o conjunto de regras; e máquina, onde são especificadas o grafo estado e os eventos que representam aplicações das regras.

O grafo estado é definido pelas seguintes variáveis: vertG e edgeG, que representam respectivamente conjuntos de vértices e arcos; sourceG e targetG que denotam respectivamente funções de origem e destino; tG_V e tG_E, que especificam funções de tipagem. A declaração de invariantes define o tipo de cada uma das variáveis (veja a coluna Invariante da Tabela 1). Invariantes são propriedades que devem ser preservadas por todos os estados alcançáveis do sistema. Para isso, sempre que uma variável é inicializada ou modificada (por exemplo, pela adição ou remoção de vértices e/ou arcos definida por uma regra) obrigações de prova são geradas automaticamente pela ferramenta de forma a garantir a preservação das invariantes (isto é, preservação de tipos). Obrigações de prova são geradas para o estado inicial e para cada uma das regras do sistema (seguindo a técnica da indução matemática).

Como o processo de prova é semi–automático e exige interação com o usuário, LEMOS JUNIOR. et al.(2014) propõe uma série de passos para a conclusão de algumas das provas. Para as táticas propostas, as hipóteses consideradas são as selecionadas pela ferramenta por default. Neste trabalho, propõe-se a otimização destas táticas, indicando quais as hipóteses são realmente necessárias para a conclusão das provas.

3. RESULTADOS E DISCUSSÃO

Neste trabalho, são consideradas as obrigações de provas que garantem a preservação de tipo das variáveis que definem o grafo estado. A seleção das hipóteses necessárias para cada obrigação de prova foi feita através da análise da aplicação das táticas propostas (LEMOS JUNIOR. et al. 2014) a especificações previamente definidas. Observou-se que para as obrigações geradas para o evento de inicialização não há hipóteses que sejam necessárias para o desenvolvimento das provas. Para as demais obrigações, geradas pelas especificações das modificações das variáveis nas aplicações das regras, as hipóteses necessárias são apresentadas na Tabela 1. A coluna Invariante traz as invariantes consideradas; na coluna Hipóteses Necessárias, são detalhadas as hipóteses necessárias para a prova da respectiva invariante através dos provadores indicados na coluna Provadores.. Estes provadores (CLEARSY, 2014), são parte do processo de prova e são adicionados a ferramenta em forma de pluging.

Para as invariantes consideradas obrigações de provas são geradas sempre que as variáveis envolvidas são modificadas pelas regras.

Para as invariantes $vertG \in \mathbb{P}(\mathbb{N})$ e $edgeG \in \mathbb{P}(\mathbb{N})$ são necessárias manter as seguintes hipóteses: aquelas que especificam o tipo do conjunto de

vértices/arcos antes da aplicação da regra; aquelas que informam que cada vértice/arco adicionado preserva o tipo e é diferente dos já existentes

Invariante	Hipóteses Necessárias	Provadores
vertG ∈ ℙ(ℕ)	vertG ∈ ℙ(ℕ) ∀ v adicionado · v ∈ ℕ \ vertG	P0, P1, ML
edgeG ∈ ℙ(ℕ)	edgeG ∈ ℙ(ℕ) ∀ e adicionado · e ∈ ℕ \ edgeG	P0, P1, ML
sourceG ∈ edgeG → vertG	sourceG ∈ edgeG → vertG mV ∈ vertLi → vertG ∀ e adicionado · e ∈ ℕ \ edgeG Seja {e1,...,en} o conjunto de arcos adicionados, se n >1 , ∀ e ∈ {e1,... en} . ei ≠ej	P0, ML, NewPP, P1
targetG ∈ edgeG → vertG	targetG ∈ edgeG → vertG mV ∈ vertLi → vertG ∀ e adicionado · e ∈ ℕ \ edgeG Seja {e1,...,en} o conjunto de arcos adicionados, se n >1 , ∀ e ∈ {e1,... en} . ei ≠ej	P0, ML, NewPP, P1
tG_V ∈ vertG → vertT	tG_V ∈ vertG → vertT ∀ v adicionado · v ∈ ∈ ℕ \ vertG Seja {v1,...,vn} o conjunto de vertices adicionados, se n >1 , ∀ v ∈ {v1,... vn} . vi ≠vj	P0, ML, NewPP, P1
tG_E ∈ edgeG → edgeT	tG_E ∈ edgeG → edgeT ∀ e adicionado · e ∈ ℕ \ edgeG Seja {e1,...,en} o conjunto de arcos adicionados, se n >1 , ∀ e ∈ {e1,... en} . ei ≠ej	P0, ML, NewPP, P1

Tabela 1. Otimizações para táticas de prova.

Para as invariantes *sourceG ∈ edgeG → vertG* e *targetG ∈ edgeG → vertG* são necessárias as seguintes hipóteses: a hipótese de indução, a qual indica que sourceG e targetG preservam o tipo antes da aplicação da regra (ou seja são funções totais que mapeiam arcos com o seu vértice origem e destino, respectivamente); as hipóteses que indicam que todos arcos adicionados pela regra preservam tipos e são diferentes entre si; e, se um arco com origem ou destino num vértice do grafo estado é adicionado, deve-se também selecionar a hipótese que indica o tipo da componente do match que mapeia vértices *mV ∈ vertLi → vertG.*

Por fim, para as ultima invariantes consideradas, *tG_V ∈ vertG → vertT* e *tG_E ∈ edgeG → edgeT*, são necessárias as hipóteses de indução, a qual indica que tG_V e tG_E preservam o tipo antes da aplicação da regra (ou seja são funções totais que mapeiam respectivamente arcos e vértices com seus tipo no grafo tipo); as hipóteses que indicam que todos arcos adicionados pela regra preservam tipos e são diferentes entre si.

Tanto as táticas já propostas, como suas otimizações, são propostas de forma genérica, podendo ser aplicadas a qualquer sistema especificado dentro desta abordagem.

4. CONCLUSÕES

Neste trabalho foram propostas otimizações a táticas previamente definidas para a abordagem de prova de teoremas de GG (DA COSTA 2010). Tanto as táticas quanto as otimizações objetivam auxiliar os desenvolvedores no processo de prova quando adotada esta abordagem.

Aqui foram consideradas as obrigações geradas para garantir à consistência do sistema, particularmente, a preservação de tipos das variáveis. Sabe-se que é possível especificar em forma de invariante propriedades que se deseja verificar do sistema. LEMOS JUNIOR at al, 2014 já sugeriu propriedades e táticas para as mesmas. Como próximo passo deste trabalho pretende-se realizar a analise da otimização destas táticas para tais propriedades

5. REFERÊNCIAS BIBLIOGRÁFICAS

EHRIG, H.; HECKEL, R.; KORFF, M.; LÖWE, M.; RIBEIRO, L.; WAGNER, A.; CORRADINI, A. Algebraic approaches to graph transformation. Part II: single pushout approach and comparison with double pushout approach. **Handbook of graph grammars and computing by graph transformation: volume I. foundations**, River Edge, NJ, USA, p.247–312, 1997.

RIBEIRO, L. Métodos formais de especificação: Gramáticas de grafos. In: VIII ESCOLA DE INFORMÁTICA DA SBC-SUL, Porto Alegre, 2000. **Anais. . .** Editora da UFRGS, 2000. p.1–33.

COSTA, S. A. da. **Relational Approach of Graph Grammars**. 2010. Tese (Doutorado em Ciência da Computação) — INF, UFRGS, Brasil.

DEPLOY. **Event-B and the Rodin Platform**. http://www.event-b.org

ABRIAL, J.-R.; BUTLER, M.; HALLERSTEDE, S.; HOANG, T. S.; MEHTA, F.; VOISIN, L. Rodin: An Open Toolset for Modelling and Reasoning in Event-B. **International Journal on Software Tools for Technology Transfer (STTT)**, [S.l.], Abril 2010.

ROBINSON, J. A.; VORONKOV, A. (Ed.). **Handbook of Automated Reasoning (in 2 volumes)**. [S.l.]: Elsevier and MIT Press, 2001.

LEMOS JUNIOR, Luiz Carlos. **PROPOSTA DE TÁTICAS PARA PROVA DE TEOREMAS DE GRAMÁTICA DE GRAFOS**. 2014. 168 f. Dissertação (Mestrado em Ciência da Computação) – Programa de Pós-Graduação em Computação, Universidade Federal de Pelotas, Pelotas, 2014.

CLEARSY. **Atelier B**. Atelier B is an industrial tool that allows for the operational use of the B Method to develop defect-free proven software (formal software). http://www.atelierb.eu/. 2014.