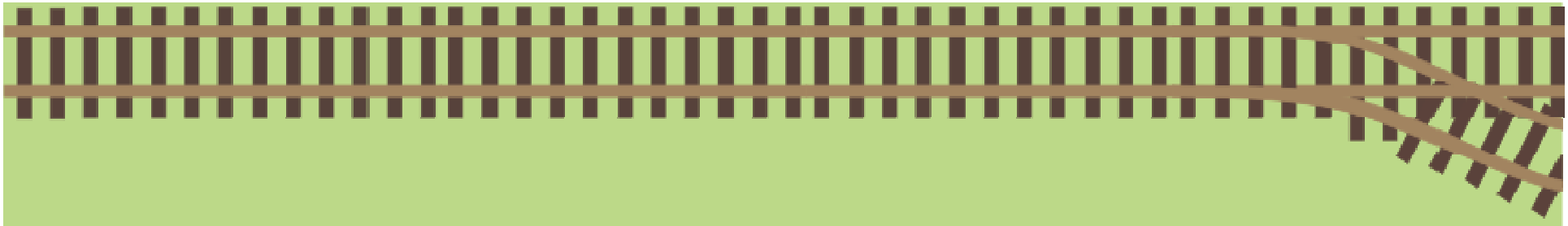


Verificador de Propriedades em Gramática de Grafos

Acadêmica: Fernanda Gums

Orientadora: Joyce Martins

Roteiro

- Introdução
- Objetivos do trabalho
- Fundamentação teórica
- Desenvolvimento do Java *Verification Graph Grammar* (JVGG)
- Conclusão
- Extensões

Introdução

- Desenvolvimento de sistemas
- Métodos formais
- Gramática de grafos
- Verificação de propriedades

Objetivos do trabalho

Desenvolver uma ferramenta para verificar propriedades em gramática de grafos.

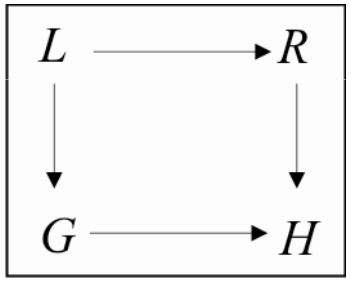
- Criação da gramática
  - Rótulos
  - Grafo inicial
  - Regras
- Verificação de propriedades
  - Alcançabilidade
  - Aplicabilidade
  - Conflito potencial

Fundamentação teórica

- Conceitos
  - Gramática de grafos
  - Propriedades da gramática de grafos
- Trabalhos correlatos
  - *Attributed Graph Grammar* (AGG) – *TU Berlin*
  - *Graph Rewrite GENerator* (GrGen) – *Opensource*
  - *Graphs for Object-Oriented Verification* (GROOVE) – *University of Twente*

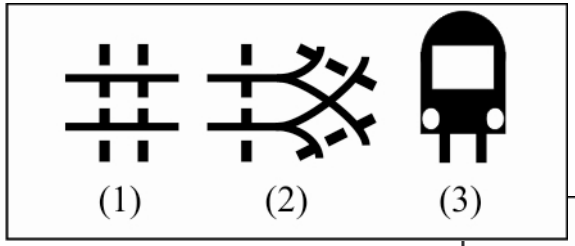
Gramática de grafos

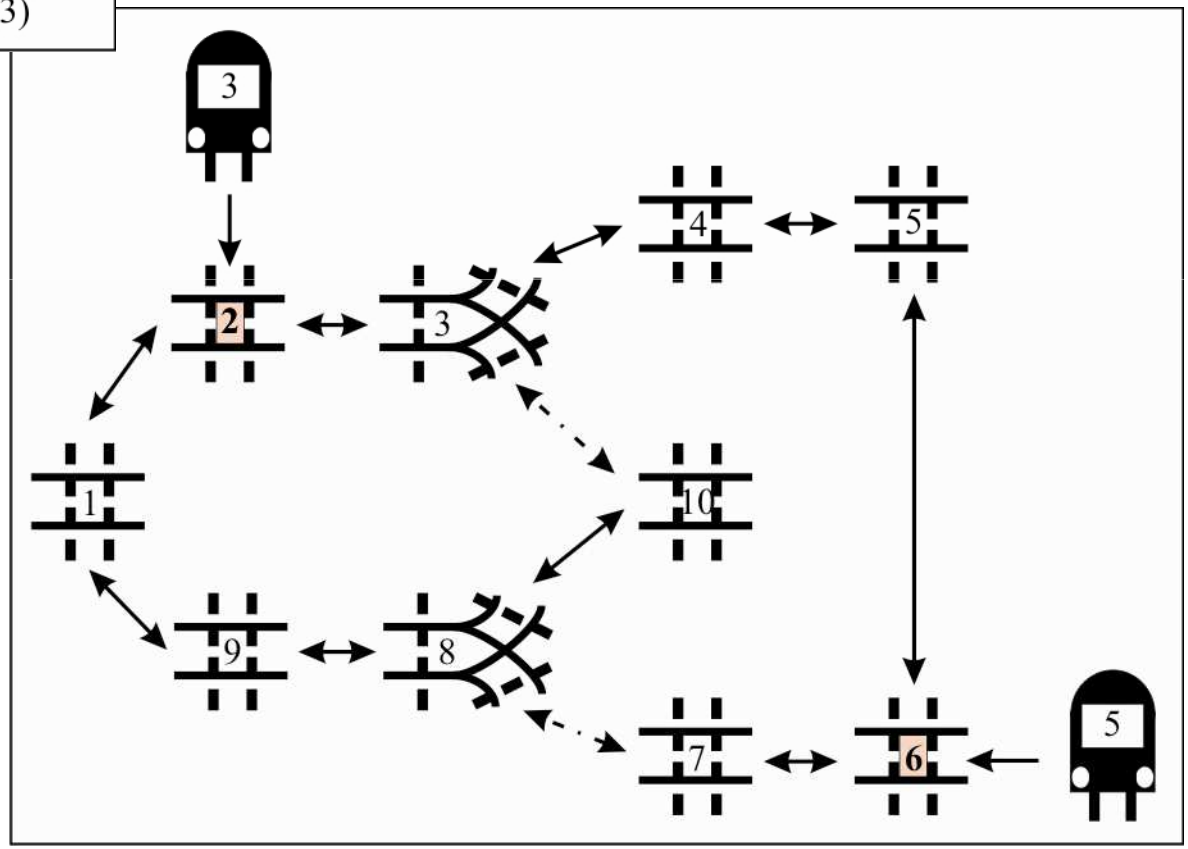
- Rótulos
- Grafo inicial
- Conjunto de regras
- Abordagem *Single PushOut* (SPO)



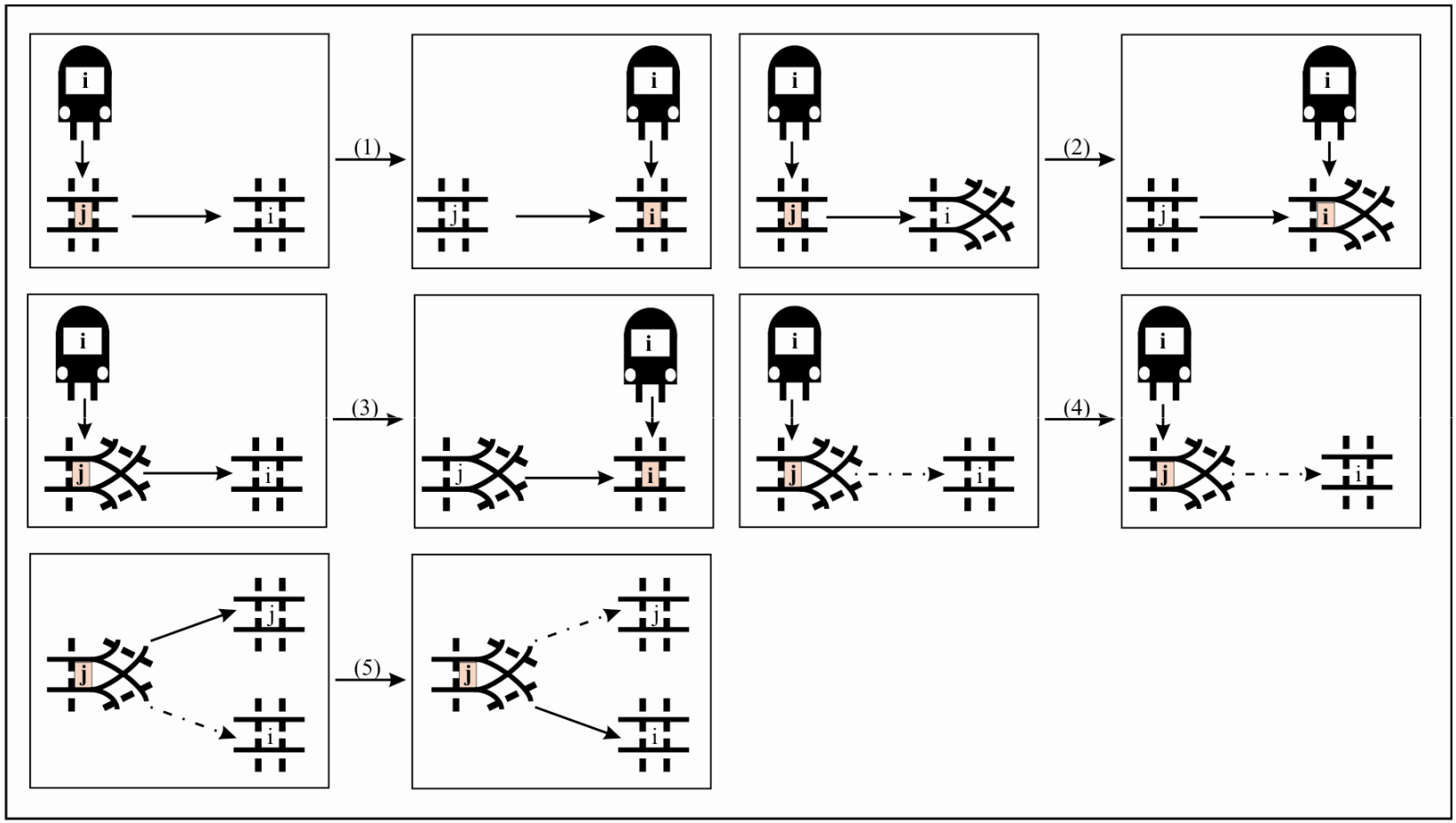
- Exemplo da Ferrovia (Déharbe)

Gramática de grafos: exemplo ferrovia



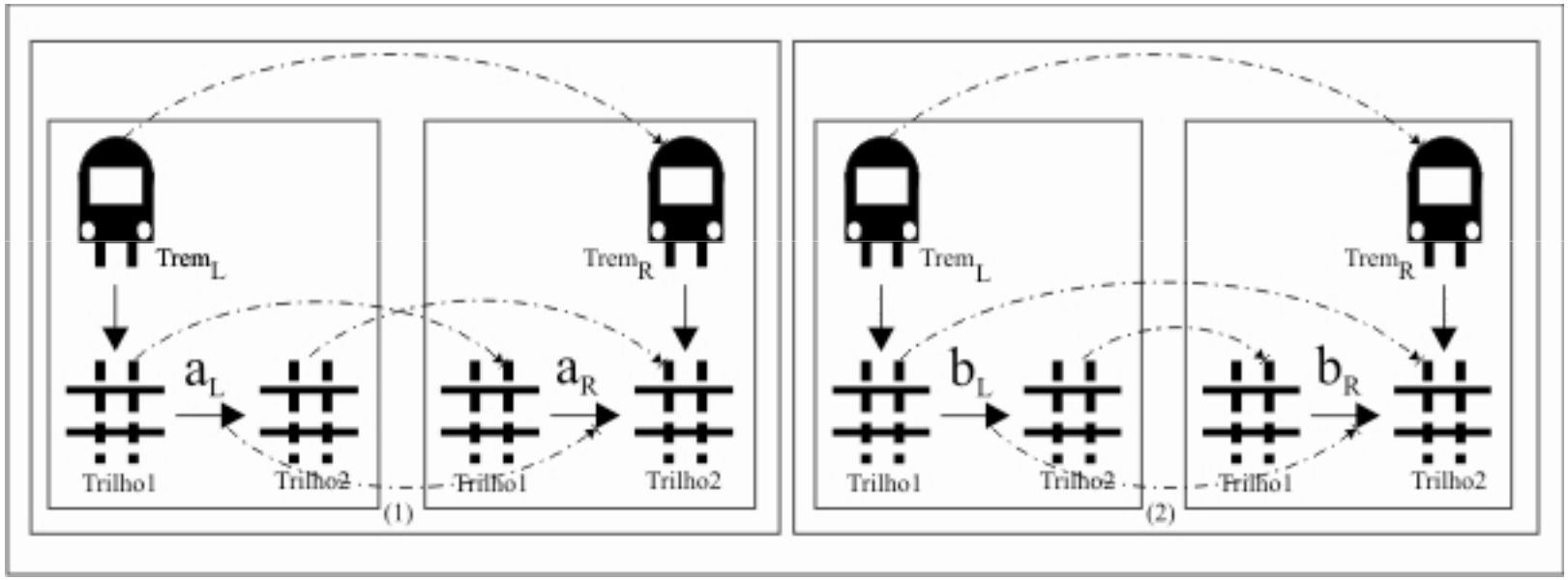


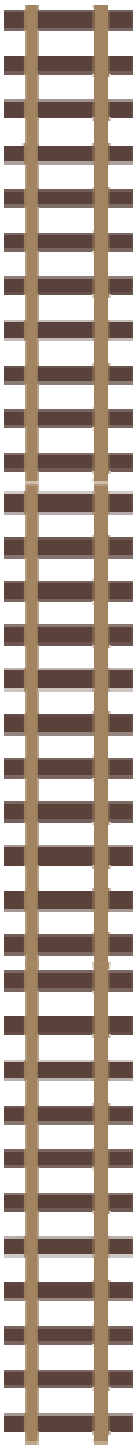
Gramática de grafos: exemplo ferrovia



Gramática de grafos: sintaxe

Morfismo e não morfismo





Gramática de grafos: sintaxe

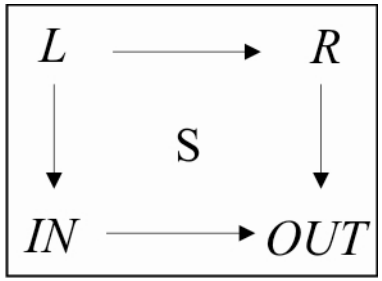
Especifica a mudança de estado do sistema:

- Itens em L devem estar presentes no estado atual para que a regra possa ser aplicada
- Itens em L que não têm uma imagem em R são removidos
- Itens em L que são mapeados para R são preservados
- Itens em R que não tem uma pré-imagem em L são criados.

Gramática de grafos: semântica

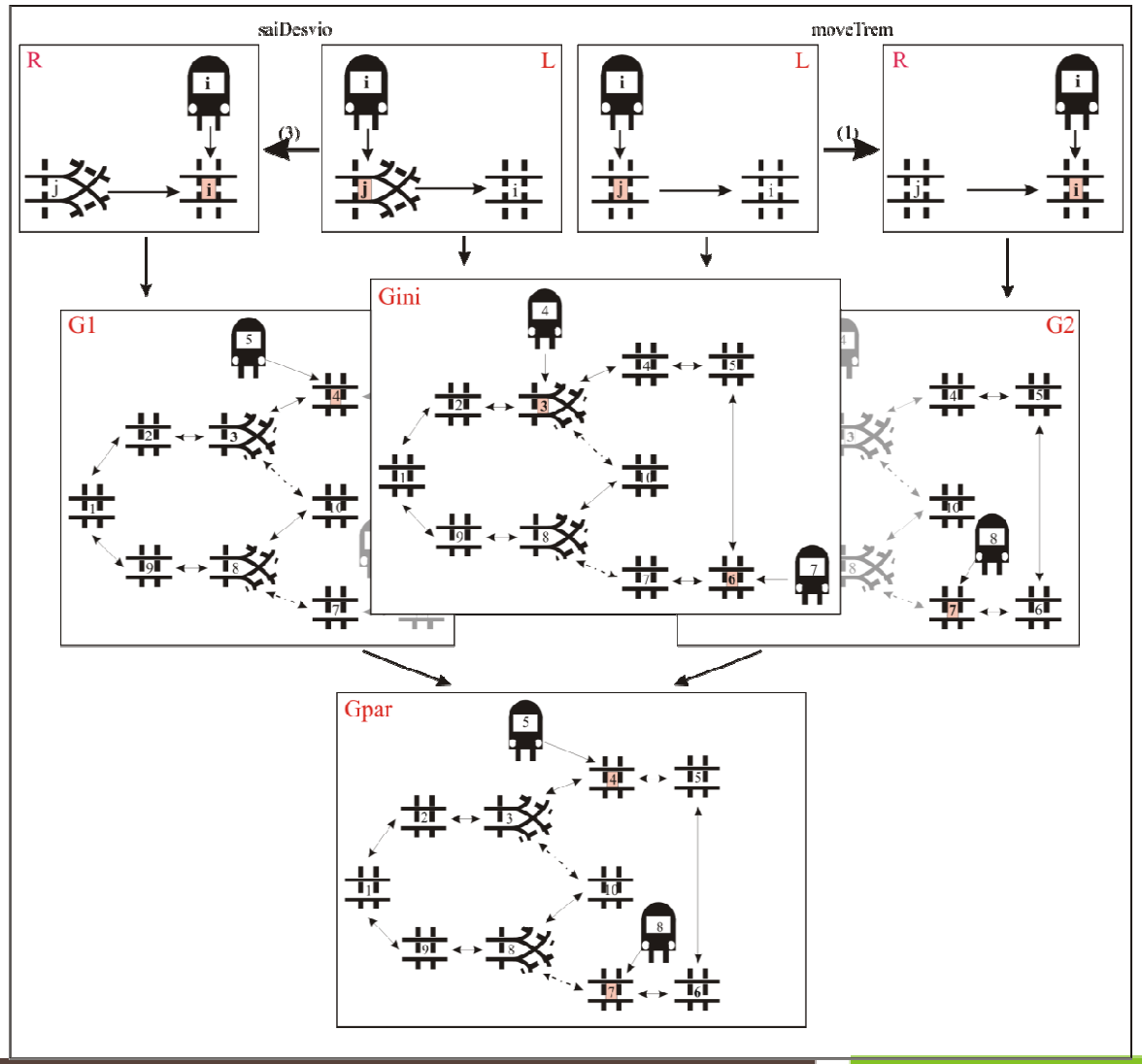
Indica o comportamento da gramática

- Adicionar ao grafo inicial tudo que foi criado pela regra
- Remover do grafo resultante do passo acima tudo que deve ser eliminado pela regra
- Remover arestas pendentes



Gramática de grafos: semântica

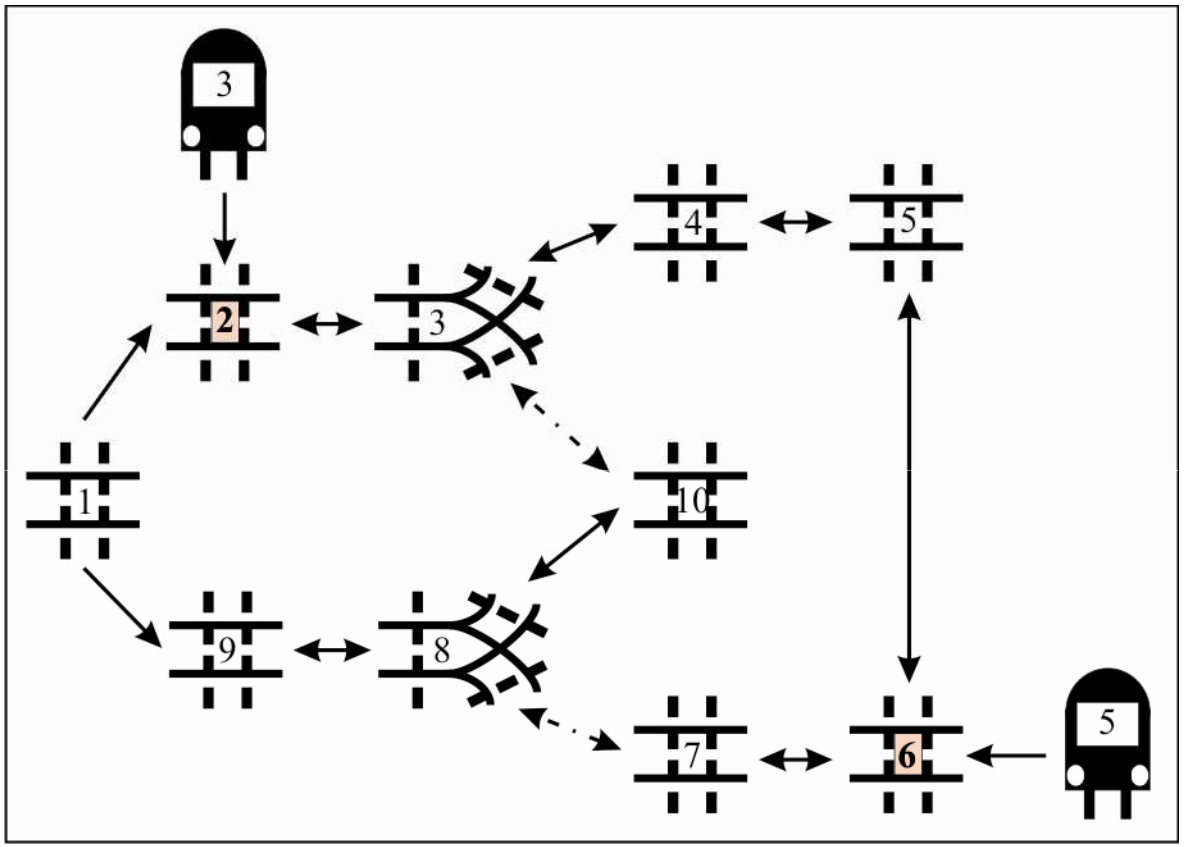
- Derivação
  - Sequencial
  - Paralela



Propriedades da gramática de grafos

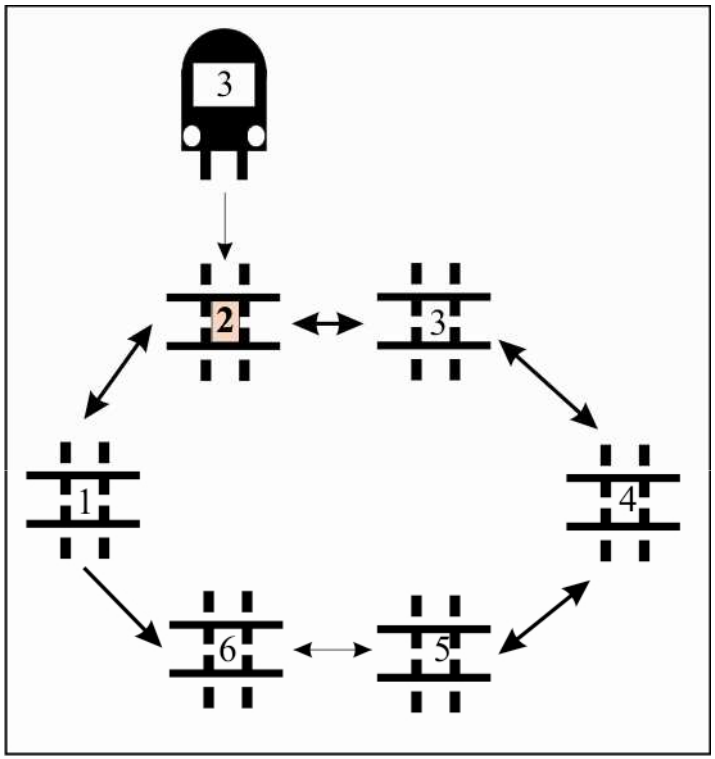
- Análise de computações da especificação
- Propriedades:
  - Alcançabilidade
  - Aplicabilidade
  - Conflito potencial

Propriedade: Alcançabilidade



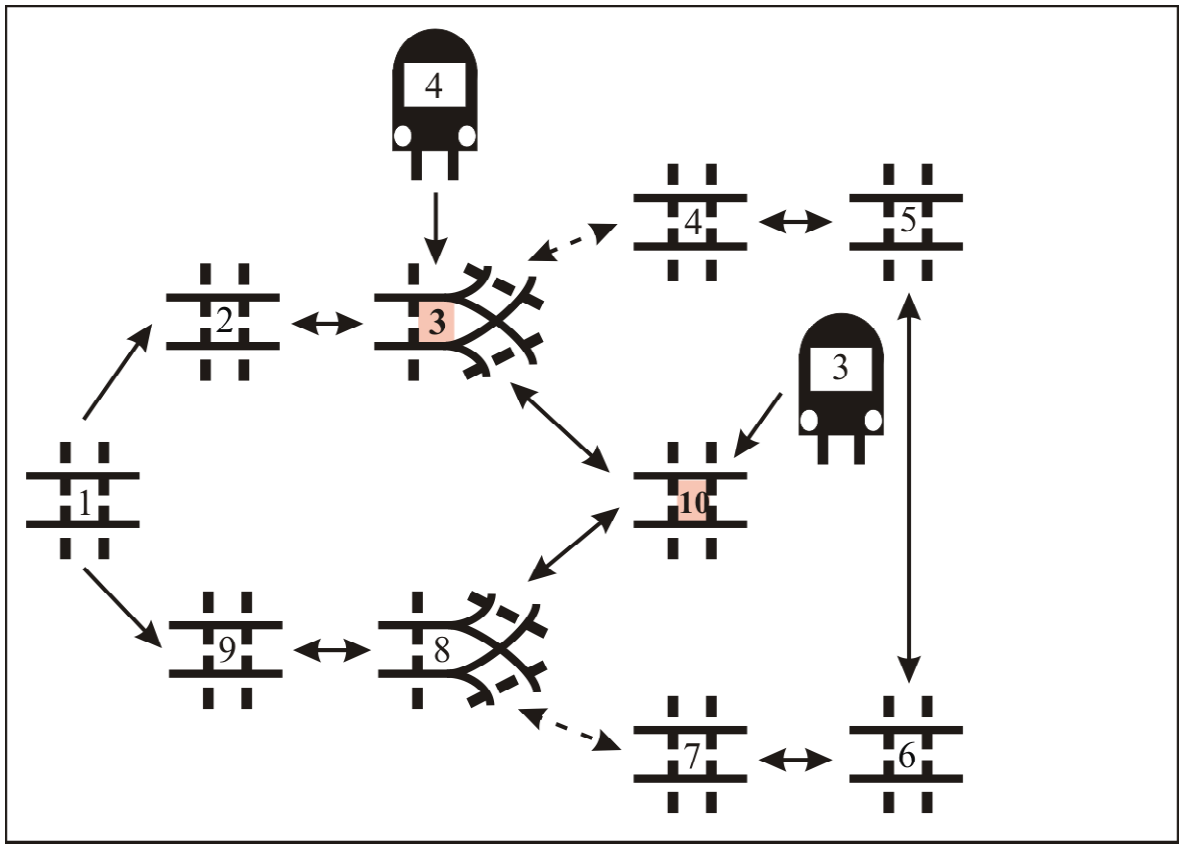
- trilho1

Propriedade: Aplicabilidade



- entraDesvio
- saiDesvio
- mudaDesvio
- aguardaDesvio

Propriedade: Conflito potencial



- entraDesvio
- mudaDesvio

Desenvolvimento do JVGG

- Levantamento dos requisitos
- Especificação do JVGG através dos diagramas de caso de uso, classes e sequência
- Implementação e testes do JVGG

Requisitos

Funcionais

- Criação de vértices e arestas
- Criação da gramática (grafo inicial e regras)
- Verificação de propriedades

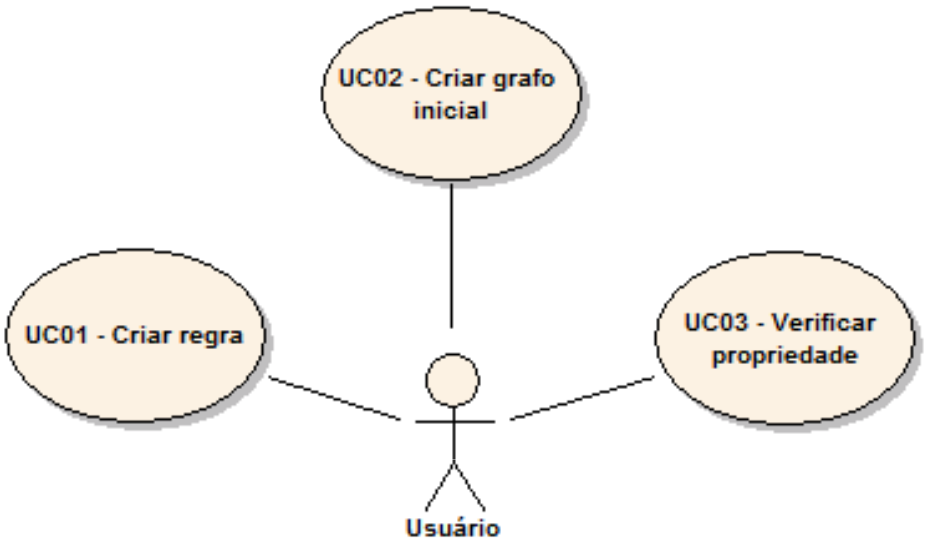
Não funcionais

- Abordagem *Single PushOut* (SPO)
- Utilizar a linguagem Java e as bibliotecas JGraph e JGraphT

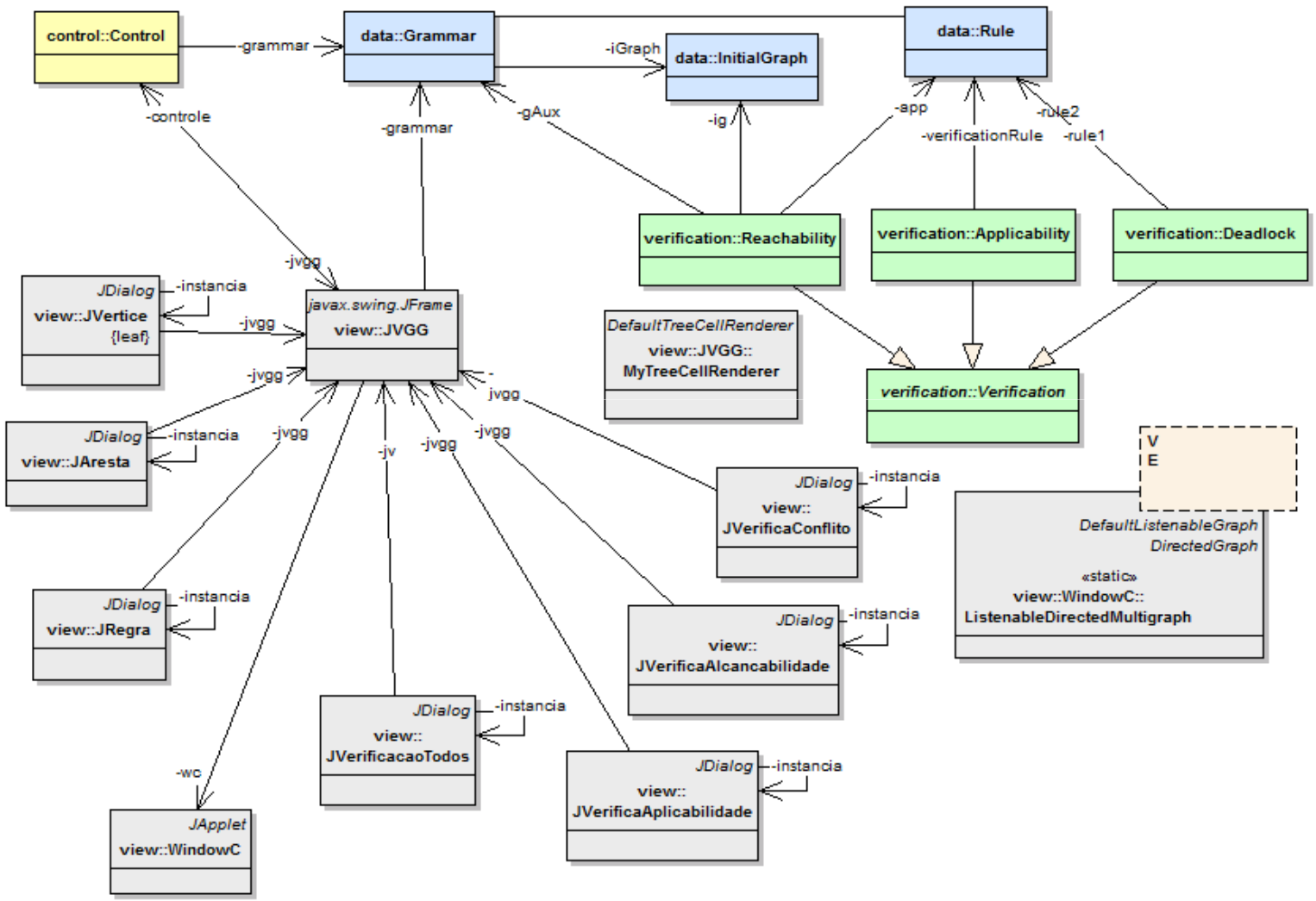
Especificação do JVGG

- Orientação a objetos
- UML / Entreprise Architect
  - Diagrama de casos de uso
  - Diagrama de classes
  - Diagrama de sequência

Especificação: Casos de uso



Especificação: Diagrama de classes



Implementação do JVGG

- Java (Eclipse + NetBeans)
- JGraph e JGraphT

- Etapas
  - Criação da gramática
  - Verificação de propriedades

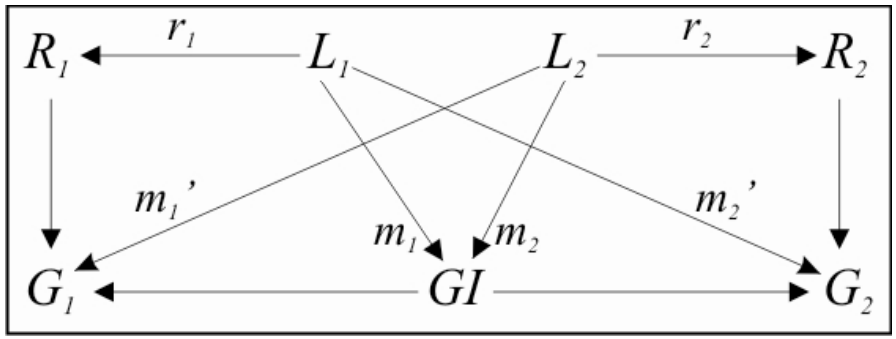
Criação da gramática

- `addVertex(String vertex) : **Grammar**`
- `addVertex(String vertex) : **InitialGraph**`
- `addEdge(String vertexSource, String vertexTarget) : **InitialGraph**`
- `Rule(String ruleName, boolean isGeneric) : **Rule**`
- `addVertex(String vertex) : **Graph**`
- `addEdge(String vertexSource, String vertexTarget) : **Graph**`

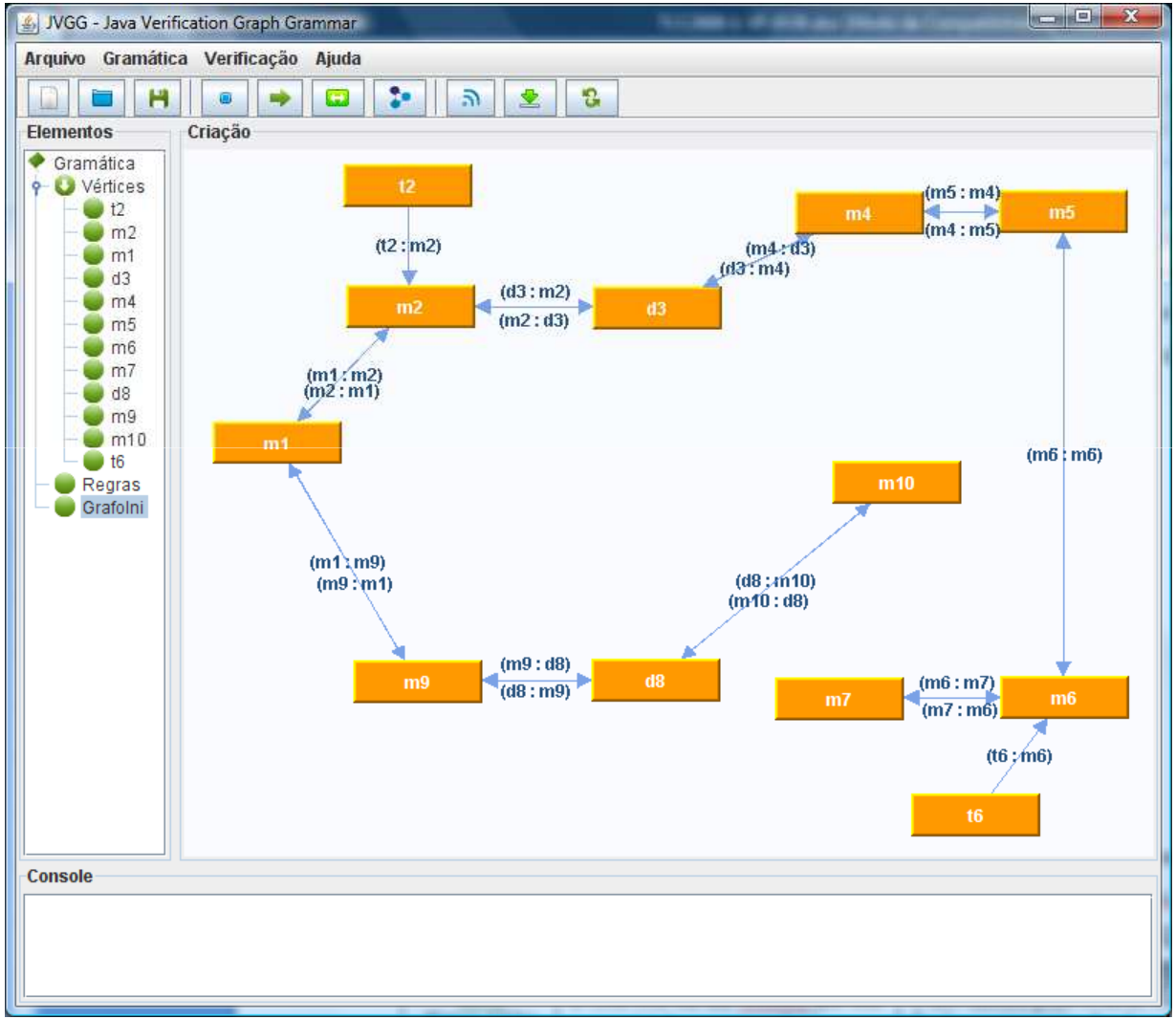
Verificação das propriedades

```
public void verification(Grammar g)
```

- Alcançabilidade
  - Entrada: Vértice origem + vértice destino
  - Dijkstra
- Aplicabilidade
  - Entrada: Regra
  - Específica ou genérica?
  - Subgrafo de L
  - Todas situações
- Conflito potencial
  - Entrada: Duas regras
  - G1 = M1 e G2 = M2

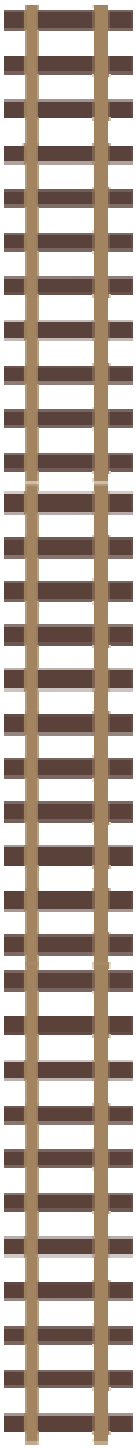


Operacionalidade



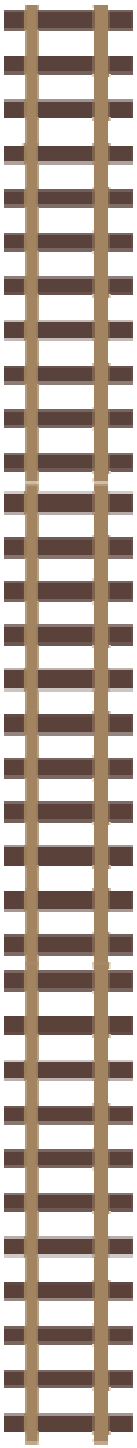
Resultados e discussão

	AGG	GrGen	GROOVE	JVGG
criação / edição de regras	✓	✓	✓	✓
criação / edição do grafo inicial	✓	✓	✓	✓
criação / edição de vértices / arestas rotulados	×	×	✓	✓
criação / edição de vértices / arestas tipados	✓	✓	×	×
criação / edição de vértices / arestas com atributos	✓	✓	✓	×
verificação de conflito	✓	×	×	✓
verificação de aplicabilidade	✓	×	×	✓
verificação de alcançabilidade	×	×	×	✓
abordagem	SPO	SPO	SPO	SPO
inicio do projeto	1997	2003	2002	2009



Conclusões

- Objetivos alcançados
- Escolha da abordagem SPO
- Verificação das propriedades
- Uso da JGraph e JGraphT



Extensões

- Transformações em tempo real
- Verificação de novas propriedades
- Gramáticas de grafos orientadas a objetos
- Grafos tipados e/ou grafos com atributos
- Criação de regras com condições negativas

Referências

DÉHARBE, David et al. Introdução a métodos formais: especificação, semântica e verificação de sistemas concorrentes. **Revista de Informática Teórica e Aplicada**, Porto Alegre, v. vii, n. 1, p. 7-48, set. 2000.

Perguntas...

Obrigada!

Fernanda Gums
ruanita@gmail.com