UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
INSTITUTO DE INFORMÁTICA
PROGRAMA DE PÓS-GRADUAÇÃO EM COMPUTAÇÃO

RAMON GOMES MEDRADO

Formalização de uma linguagem visual para descrição de sistemas biológicos

Dissertação apresentada como requisito parcial
para a obtenção do grau de
Mestre em Ciência da Computação

Prof[a]. Dra. Leila Ribeiro
Orientadora

Porto Alegre, novembro de 2009

CIP – CATALOGAÇÃO NA PUBLICAÇÃO

Medrado, Ramon Gomes

Formalização de uma linguagem visual para descrição de sistemas biológicos / Ramon Gomes Medrado. – Porto Alegre: PPGC da UFRGS, 2009.

78 f.: il.

Dissertação (mestrado) – Universidade Federal do Rio Grande do Sul. Programa de Pós-Graduação em Computação, Porto Alegre, BR–RS, 2009. Orientadora: Leila Ribeiro.

1. Linguagem visual. 2. Gramática de grafos. 3. Sistemas biológicos. 4. Diagrama de processos biológicos. 5. Semântica. 6. GSPN. I. Ribeiro, Leila. II. Título.

“You can’t connect the dots looking forward; you can only connect them looking backwards. So you have to trust that the dots will somehow connect in your future. You have to trust in something - your gut, destiny, life, karma, whatever. Because believing that the dots will connect down the road, will give you the confidence to follow your heart, even when it will lead you off the well-worn path, and that will make all the difference..“

—STEVE JOBS

AGRADECIMENTOS

Agradeço a Deus pela sua onipresença e por ser a razão de minha existência.

Agradeço a minha orientadora, Leila Ribeiro, pelo conhecimento, paciência e confiança com os quais me conduziram até a conclusão desta dissertação.

Agradeço aos professores Daltro José Nunes, Paulo Blauth Menezes, Roberto da Silva, Luís Lamb, Markus Ritt e Luciana Buriol pelos conhecimentos transmitidos em aula que foram úteis à minha pesquisa. Aos demais professores e funcionários que fazem do PPGC e do Instituto de Informática um centro de excelência em pesquisa, e ao CNPq, pelo suporte financeiro ao longo deste projeto.

Aos meus colegas da casa de estudante da UFRGS (CEFAV), pelos conselhos e solidariedade nos momentos difíceis desta trajetória.

Aos meus colegas de laboratório Rodrigo Machado, Willian Alves, Luciana Foss, Leonardo Couto, Cláudio Fuzitaki, Lincoln Rabelo e Germano Caumo pelos sábios conselhos e pelas críticas construtivas.

Ao povo gaúcho pela hospitalidade e cortesia.

À minha família que mesmo estando a uma distância continental de 2600 km sempre me apoiou na concretização de meus objetivos.

SUMÁRIO

LISTA DE ABREVIATURAS E SIGLAS

GSPN	Generalized Stochastic Petri Nets
VL	Visual Language
SBML	Systems Biology Markup Language
SBGN	Systems Biology Graphical Notation
BioPNML	Biology Petri Net Markup Language

LISTA DE FIGURAS

RESUMO

Vias biológicas representam interações entre entidades químicas complexas (proteínas, substratos, metabólitos etc.) que ocorrem no nível molecular das células. A representação e compreensão do comportamento destas vias é o principal alvo de estudos da Biologia Sistêmica. Esta área de estudos envolve a construção de modelos matemáticos que possam simular *in silico* (computacionalmente) o comportamento destes sistemas biológicos verificados *in vivo* (experimentalmente).

Do ponto de vista computacional é evidente que tais sistemas são complexos para abordar e descrever de modo intuitivo. São necessários modelos com valor preditivo, isto é, que permitam descrever os comportamentos do sistema que são experimentalmente verificáveis. Algumas notações gráficas foram propostas para descrever vias biológicas. Entre elas, os diagramas de processos tem sido amplamente utilizados. Um diagrama de processos é essencialmente um grafo no qual vértices e arestas representam componentes biológicos, e há uma notação gráfica associada com cada elemento.

Nesta dissertação propomos uma fundamentação formal para a linguagem dos diagramas de processos definindo a sintaxe usando gramática de grafos. Nós definimos primeiramente um grafo chamado BioProc, descrevendo o meta-modelo dos diagramas de processos. Instâncias do grafo BioProc são portanto diagramas de processos modelando vias biológicas. Para descrever a semântica foi proposta uma tradução algébrica dos grafos BioProc para redes de Petri estocásticas generalizadas (GSPNs) já amplamente utilizadas na modelagem de processos biológicos. O uso de gramática de grafos como formalismo intermediário na tradução habilita a verificação sintática da via com a checagem dos tipos válidos que podem ser definidos para cada reação antes da simulação na rede de Petri e usá-las posteriormente para explorar propriedades estruturais e estocásticas do modelo. Além disso serve como base para a evolução do modelo proposto. Isto é relevante já que modelos frequentemente são construídos incrementalmente para se adaptar a novos requisitos e/ou incluir novas características.

Palavras-chave: Linguagem visual, gramática de grafos, sistemas biológicos, diagrama de processos biológicos, Semântica, GSPN.

Formalization of a visual language to specify biological pathways

RESUMO

Biological pathways represent interactions between complex chemical entities (proteins, substrates, metabolites, etc.) that occur at the molecular level of cells. The representation and comprehension of biological pathways behavior is the main target of research in the field of Systems Biology. This area investigates the construction of mathematical models that can simulate *in silico* (computationally) the behavior of biological systems checked *in vivo* (experimentally).

From a computational view point it is clear that such systems are too complex to analyze and describe in an intuitive way. Models with predictive value are needed, describing the behaviors that are experimentally verifiable. There are some graphical notations to describe biological pathways. Among them, process diagrams have been widely used. A process diagram is essentially a graph in which vertices and edges represent biological components, and there is a graphical notation associated with each element.

In this master thesis we give a formal foundation for biological process diagrams, by defining their (concrete and abstract) syntax and semantics using a formalism called graph grammars. We first build a graph called BioProc Graph, describing the meta-model of process diagrams. Instances of this BioProc graph are concrete process diagrams modeling biological pathways. To describe the semantics we proposed a translation of BioProc diagrams to generalized stochastic Petri networks (GSPNs) already widely used in modeling biological processes. The use of graph grammar formalism as a basis for translation enables the syntatic verification to check the valid types that can be defined for each reaction after the simulation of Petri net and before that to explore structural and stochastic properties of the model. In addition it serves as the basis for model evolution proposed. This is relevant because models are often built incrementally to adapt to new requirements and/or include new features.

Palavras-chave: visual language, graph grammar, biological pathways, biological process diagrams.

1 INTRODUÇÃO

Um dos tópicos de investigação da biologia sistêmica é a representação de interações entre entidades químicas complexas (proteínas, substratos, metabólitos etc.) que ocorrem no nível molecular das células. Tais interações são determinantes nas funções vitais de qualquer organismo e são denominadas vias ou sistemas biológicos.

Do ponto de vista computacional é evidente que tais sistemas são complexos para abordar e descrever de modo intuitivo. São necessárias linguagens de descrição que gerem modelos com valor preditivo, isto é que habilitem métodos de análise para verificar se o modelo corresponde ao sistema real, bem como predizer seu comportamento, ou como um sistema isolado ou como parte de uma rede complexa. Tais modelos aumentariam a compreensão dos sistemas vivos, relacionando o comportamento básico das moléculas a comportamentos complexos.

Vias biológicas possuem um comportamento que pode ser relacionado às descrições de sistemas reativos. As células traduzem estímulos externos (hormônios, fatores de crescimento, entre outras substâncias) em adequadas respostas biológicas (PINTO et al., 2007). As descrições destas vias como modelos computacionais (*in silico*) tem uma forma geral (KITANO, 2007) composta por entidades envolvidas, os substratos, os produtos e uma taxa associada.

É possível construir modelos matemáticos (como sistemas de equações diferenciais) para descrever o comportamento a partir dos dados observados do sistema. Entretanto usar linguagem matemática diretamente torna a compreensão do modelo proposto mais complexa, especialmente para biólogos. Um modo natural de superar este problema é usar uma linguagem de descrição adequada tal que os "programas" escritos com ela possam ser traduzidos em modelos matemáticos automaticamente. Se a linguagem tem uma semântica formal é possível raciocinar sobre o modelo matemático inerente a cada programa em diferentes modos (simulação, verificação, análise estática, etc.).

Algumas linguagens para descrição foram propostas como SBML (HUCKA et al., 2003), BioPNML (CHEN et al., 2002), BioPAX (LUCIANO; STEVENS, 2007). Estas são linguagens textuais baseadas na notação XML, apropriadas para o armazenamento em bancos de dados. Manipular tais especificações diretamente neste formato demanda ferramentas de apoio à verificação sintática com algum formalismo que defina as regras de transformação das especificações e associe os diagramas especificados por biólogos a sintaxe abstrata do XML.

Existem editores como BioUML (KOLPAKOV, 2004), Pavesy (LüDEMANN et al., 2004), Celldesigner (FUNAHASHI et al., 2008) que realizam o *parsing* destas especificações em algum tipo de notação gráfica compreensível por biólogos como diagramas de processos. Entretanto a relação entre os dados gerados e a notação gráfica não é apropriadamente formalizada. Estas ferramentas de editoração de especificações utilizam como

modelo semântico de simulações equações diferenciais que, apesar de ser um formalismo matemático altamente detalhado, não é adequado para a modelagem de grandes vias biológicas (já que o número de variáveis e equações levam a modelos intratáveis).

1.1 Objetivos

Neste trabalho formalizaremos a representação visual amplamente usada por biólogos para descrever vias biológicas proposta por Kitano (KITANO et al., 2005), definindo uma linguagem formal com sintaxe concreta e abstrata bem como semântica para tais diagramas.

Como o objetivo desta linguagem é a geração de diagramas, propomos o modelo conceitual de gramática de hipergrafos tipada com atributos (EHRIG et al., 2006) para a representação da sintaxe abstrata.

Tratamos também da escolha de um modelo semântico que represente as simulações de vias biológicas, de forma a contemplar as características descritas nos bancos de dados atuais. Propomos uma tradução formal de diagramas de processos biológicos para GSPNs (KARTSON et al., 1994) uma definição híbrida de redes de Petri que possibilita a exploração de propriedades estruturais e estocásticas do modelo usando lógica estocástica contínua (CEROTTI et al., 2006), já que é um modelo isomórfico a cadeias de Markov. O modelo de diagrama de processos ainda não possui uma versão definitiva, podendo adquirir novas características e propriedades que não haviam sido previstas na sua origem num processo evolutivo da especificação. Gramáticas de grafos são uma ferramenta formal bem sucedida na área de refatoração e evolução de modelos, permitindo a especificação de regras para descrever como o metamodelo evolui.

1.2 Estrutura da dissertação

Os próximos capítulos da dissertação estão organizados como segue:

- No capítulo 2 são apresentadas a definição e classificação de vias biológicas, além da motivação para este trabalho.
- No capítulo 3 são apresentados trabalhos relacionados a especificação e simulação de sistemas biológicos.
- No capítulo 4 é apresentada a definição formal da gramática de grafos da linguagem.
- No capítulo 5 é apresentada a tradução semântica da linguagem para Redes de Petri Estocásticas Generalizadas, a especificação de dois exemplos utilizando a linguagem visual proposta e a verificação de propriedades.
- No capítulo 6 são apresentadas conclusões e perspectivas de trabalhos futuros.
- No apêndice são apresentadas todas as regras que definem a sintaxe da linguagem.

2 VIAS BIOLÓGICAS

Neste capítulo será apresentada uma explicação geral da área de aplicação deste trabalho. Conforme citado no capítulo anterior, vias biológicas são sistemas de interações ou cadeias sequenciais de reações químicas. Estes sistemas podem ser classificados segundo (LUCIANO; STEVENS, 2007) em 4 tipos:

- Vias metabólicas - são formadas por um conjunto de reações químicas que, após etapas sucessivas, transformam uma molécula em outra(s) substância(s). Durante este processo, elementos auxiliares como enzimas e cofatores como metais, vitaminas, etc. podem ser necessários para acionar apropriadamente cada etapa da reação, funcionando como catalisadores. O exemplo mais conhecido de via metabólica é a glicólise, processo no qual a molécula de glicose é transformada em ATP (molécula responsável pelo armazenamento de energia na célula).

- Vias de sinalização - são vias que alteram o funcionamento da célula, a partir de estímulos do ambiente. A via da insulina se enquadra nesta classificação, onde a entrada de insulina na célula promove o aumento da absorção de glicose livre no sangue para o interior da célula.

- Vias regulatórias - são sistemas de interação compostos de genes e proteínas responsáveis pelo controle das quantidades necessárias de substâncias para desenvolvimento de um determinado organismo.

- Vias de interações moleculares - neste grupo enquadram-se as interações do tipo proteína-proteína ou entre uma proteína com moléculas de outro tipo (exemplo enzima-substrato).

A reconstrução destas vias em modelos computacionais (modelos *in silico*) devem obedecer um conjuntos de regras gerais. As interações (transformações químicas) entre os componentes da via, são baseados em princípios da química básica. Estas reações químicas segundo (SHAPIRO; LEVCHENKO; MJOLSNESS, 2001) possuem uma forma geral mostrada na fórmula 2.1:

$$\sum_{X_i \in S' \subset S} X_i \xrightarrow{k} \sum_{Y_i \in S'' \subset S} Y_i \tag{2.1}$$

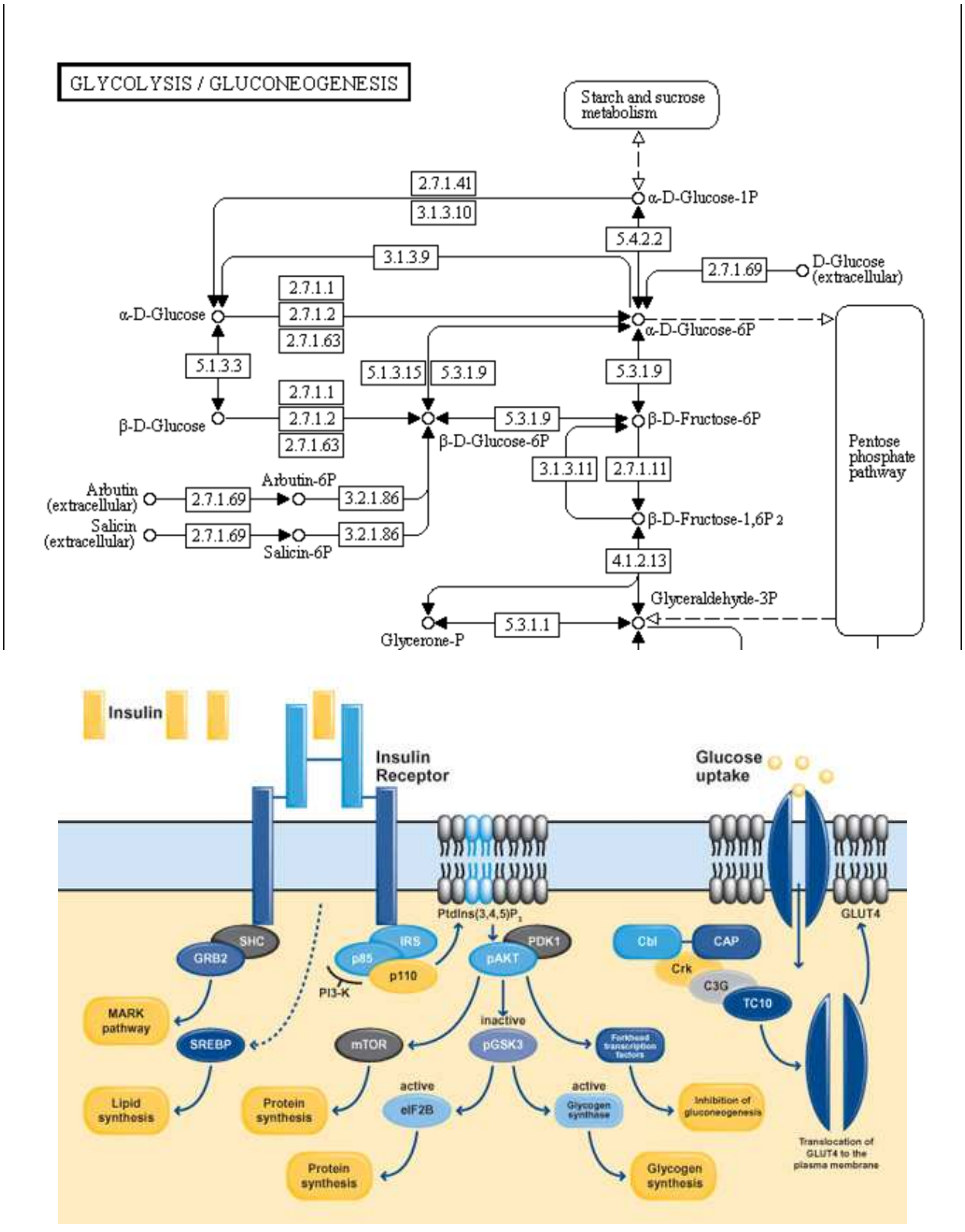
- S é o conjunto de todas entidades participantes da reação
- S' é o subconjunto de S que são reagentes da reação.
- S'' é o subconjunto de S que são produtos da reação.

- k é uma taxa representada por uma função parametrizada com imagem em $\mathbb{R}$ ou um valor constante.

Estas leis de reação são responsáveis por determinar a concentração dos reagentes e produtos após um determinado período. No caso das vias de interação proteína-proteína por exemplo as leis de reação são constatadas com bases em estudos das propriedades estruturais e físico-químicas de suas superfícies de interfaces. As leis que regem as associações entre duas proteínas A e B complexo C podem ser expressas portanto da seguinte forma:

$$A + B \xrightarrow{k} C \tag{2.2}$$

Na figura 2.1 são apresentadas algumas representações dos tipos de vias.



Fonte : (LUCIANO; STEVENS, 2007)

Figura 2.1: Representações de vias biológicas a) Via metabólica (glicólise) b) Via de sinalização (insulina humana).

Estas interações podem resultar em complexos estáveis (permanentes) ou instáveis que existem de forma transitória (ou seja as proteínas que aglutinam na forma de um complexo que podem se separar).

A detecção de interações de proteína-proteína pode ser feita experimentalmente pelo método clássico denominado *yeast two hybrid method* (Y2M). A estratégia é usar dois domínios (grupos funcionais) e comparar o comportamento de dois proteomas (XIONG, 2006). Dados dois domínios A e B, caso exista interação dos domínios em um proteoma, haverá a formação de uma proteína de fusão que será ativa. Em outro proteoma os genes A e B codificam proteínas de interação que podem realizar uma função comum à proteína codificada pelo gene C. É detectada então a presença de interação proteína-proteína.

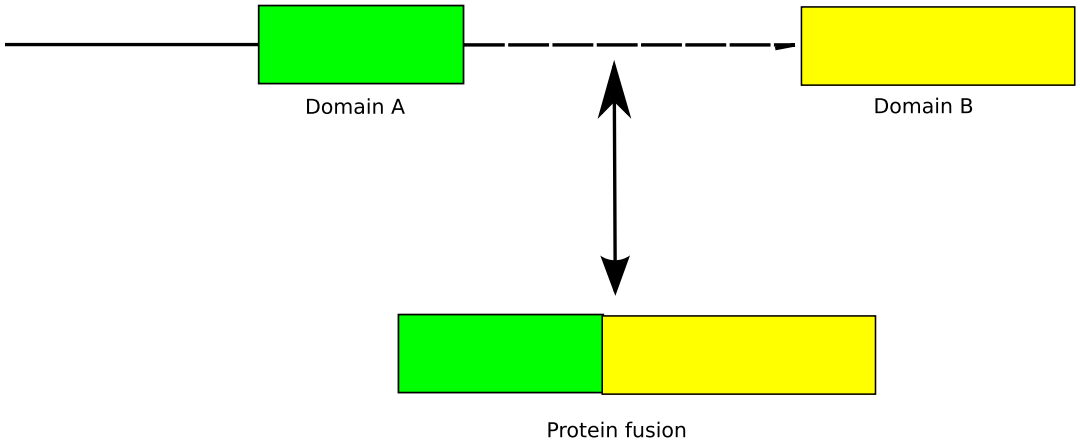


Figura 2.2: Yeast Two Hybrid Method.

Bancos de dados de interações proteína-proteína denominados *interactomes* (CUSICK et al., 2005), acumularam grandes quantidades de dados. Geralmente as interações de proteínas são expressas na forma de matrizes, onde cada célula representa a interação entre duas proteínas. Manipular e compreender tais vias com esta representação torna-se uma tarefa custosa com milhares de proteínas interagindo.

Reactions involving SHC1

Physical Interaction

Interacting molecule	Pathway
IL2RB	IL-2
GRB2	IL-2
GRB2	IL-3
CSF2RB	IL-3
INPP5D	IL-3
GAB2	IL-3
SOS1	IL-3
RAPGEF1	IL-3
CRKL	IL-3
IL4R	IL-4
PLCG1	IL-4
INPP5D	IL-4
GRB2	IL-5

Figura 2.3: Matriz de interações proteína-proteína

Na próxima seção apresentaremos uma notação gráfica utilizada para especificar vias biológicas denominada diagramas de processos biológicos. A partir dela será proposta uma formalização da notação na forma de uma linguagem formal visual.

2.1 Diagramas de processos biológicos

A modelagem de sistemas biológicos visa o estudo da evolução das concentrações das moléculas participantes. Este aspecto pode ser visualizado na forma de diagramas de processos contendo entidades químicas participantes e as possíveis reações entre elas. Ao contrário da química que já estabeleceu os padrões de notação com a IUPAC (LANGE, 1999), não há ainda uma simbologia gráfica padrão que seja formalmente definida para expressar estes diagramas na biologia.

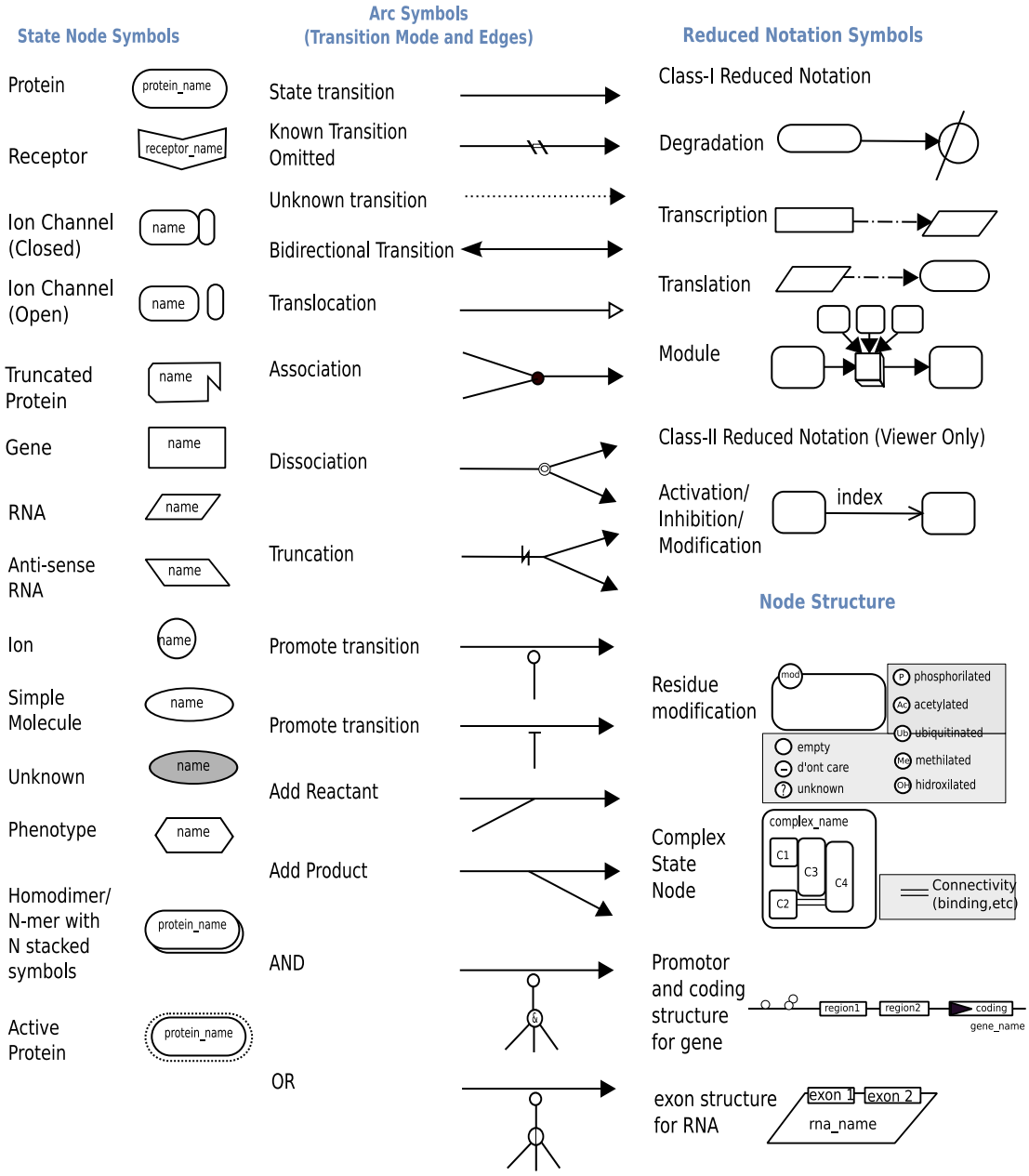
Algumas notações foram propostas por (MAIMON; BROWNING, 2001), (MOODIE et al., 2006), (LEE M HYUN S, 2003) sem que fossem firmadas como padrões de plena aceitação. Isto se deve ao fato destas últimas citadas serem diagramas adaptados de modelos computacionais como diagramas de circuitos integrados e diagramas UML. Estes diagramas muitas vezes não possuem expressividade suficiente para representar todas as possíveis reações que podem ocorrer em sistemas biológicos. Um grupo de pesquisa liderado por Kitano e com apoio de membros da Universidade de Edinburgo estabeleceram recentemente algumas diretrizes para a especificação de um padrão denominado SBGN (System Biology Graphical Notation) (NOVERE et al., 2008). A base da SBGN é o conjunto de símbolos gráficos usados por (KITANO et al., 2005) com algumas modificações e inclusões de novos tipos de símbolos.

Com base em estudo comparativos das notações, já foi defendida por alguns autores a idéia do diagrama de processos como um padrão na especificação de sistemas biológicos (KIM et al., 2008), (SCHILLING et al., 2008). Entre os principais pontos que favorecem a esta escolha estão : um nível de compreensão *user-friendly* adequado aos biólogos já que foi criada sob medida para descrição de vias biológicas e possuir uma boa documentação para o uso da notação. Devido a estas características e a adoção desta notação como um padrão para especificação de vias biológicas escolhemos formalizar a notação dos diagramas de processos.

A seguir apresentamos uma listagem dos símbolos gráficos, uma semântica informal dos símbolos e alguns diagramas exemplos da linguagem com base nas definições de (KITANO et al., 2005) mostrada na figura 2.4. Estes diagramas buscam representam todos os possíveis reagentes e produtos gerados das reações de vias biológicas, sem explicitar a sequência temporal dos eventos. Todos os estados das reações antes (reagentes) e depois (produtos) são representados no mesmo diagrama.

Os símbolos gráficos da notação são divididos em:

- Nodos de estado - representam os tipos de substâncias que podem originar ou que serão produzidas numa reação;
- Nodos de transição - representam os vários tipos de reações que podem ocorrer num sistema biológico;
- Notação reduzida - representa uma notação simplificada de detalhes do sistema biológico. É uma forma de simplificação gráfica da notação principal. É dividida em 2 categorias:
  - Categoria I - abstração de processos intermediários cujo detalhamento não é o foco de estudo do sistema biológico, como transcrição, tradução e degradação.
  - Categoria II - representação simplificada dos símbolos dos nodos de transição. Todas as reações são representadas por uma seta simples e um rótulo que diferencia cada tipo de reação.



Fonte : (KITANO et al., 2005)

Figura 2.4: Notação gráfica de Kitano para vias biológicas.

Os vários tipos de nodos de estado que podem ser descritos com o diagrama são listados a seguir:

- *Protein* - representa uma estrutura sequencial linear de aminoácidos codificada por um gene.
- *Receptor* - representa um tipo específico de proteína que capta sinais extracelulares.
- *Ion Channel* (canal de íon) - representa um tipo especial de proteína geralmente associada a membrana de algum compartimento celular. Este tipo de proteína é responsável pelo fluxo de algum tipo de substância para o interior ou exterior de um compartimento celular. Apresenta-se em dois estados: aberto (habilitada ao transporte de uma substância) e fechado (não habilitada).

- *Truncated Protein* - uma proteína que não atingiu sua forma apropriada por algum motivo, como perda de alguns resíduos de aminoácidos presentes na proteína normal. Normalmente este tipo de proteína perde sua função original.
- *Gene* - é a unidade fundamental da hereditariedade, formada por uma seqüência específica de ácidos nucléicos que podem codificar uma proteína ou controlar uma característica fenotípica.
- *RNA* - assim como o DNA também é uma molécula formada de uma sequência específica de ácidos nucléicos.
- *Anti-sense RNA* - representa uma sequência complementar de nucleotídeos de uma sequência de RNA normal.
- *Simple Molecule* - representa todas as outras classes de substâncias que podem estar presentes numa via biológica como açúcares, lipídios etc.
- *Unknon Substance* - alguma substância que faz parte da via biológica que não foi devidamente identificada.
- *Íon* - uma espécie química eletricamente carregada, um átomo ou molécula que perdeu ou ganhou elétrons. Íons são representados por círculos isolados ou associados a proteínas ou outras substâncias.
- *Phenotype*- são as características visíveis de um indivíduo, que são definidas pela expressão de um gene.
- *Homodimer* - complexo formado por duas moléculas que se comportam como uma única substância.
- *Multimer* - complexo formado por duas ou mais moléculas diferentes entre si que no entanto se comportam como uma única substância.
- *Empty Set* - representa uma substância degradada.

O conjunto de nodos de transições definem as reações que possibilitam as mudanças de estado de uma via biológica. As transições propostas são listadas abaixo:

- *State Transition* - representa uma reação que transforma um entidade reagente em outra entidade produto.
- *Known Transition Ommited* - representa uma reação conhecida que entretanto foi omitida da via biológica.
- *Unknown Transition* - representa uma reação desconhecida.
- *Add Reactant* - representa o acréscimo de um reagente à uma transição.
- *Add Product*- representa o acréscimo de um produto à transição.
- *Bidirectional Transition* - representa uma transição reversível de uma substância para outro estado.
- *Translocation*- representa a saída de uma substância de um compartimento.

- *Association*- representa uma associação entre duas substâncias, para a formação de um complexo molecular.

- *Dissociation* - representa o desmembramento de um complexo molecular em subcomplexos ou substâncias simples.

- *Truncation* - representa o desmembramento de um complexo molecular ou de uma proteína resultando como produtos substâncias resíduo da reação e a proteína com alguma alteração na sua sequência de aminoácidos.

- *Promote Transition* - representa uma transição que só ocorre na presença de uma substância promotora.

- *Inhibit Transition* - representa uma transição que pode ser inibida por uma determinada substância.

- *Add Reactant* - representa a entrada de um reagente durante uma transição de estado.

- *Add Product*- representa a saída de um produto após a transição.

- *And* - representa a inibição ou estímulo de uma transição por um conjunto de substâncias onde todos participantes presentes são necessários para realizar o bloqueio ou estímulo da reação.

- *Or* - representa a inibição ou estímulo de uma transição por um conjunto de substâncias onde pelo menos um dos participantes presentes pode realizar o bloqueio ou estímulo da reação.

Na figura 2.5 são apresentados alguns exemplos de diagramas escritos com a notação de diagrama de processos.

a) Na primeira reação temos uma transformação da lactose em galactose e glicose estimulada pela proteína beta-galactosidase, ou seja enquanto houver quantidade suficiente da proteína a reação será catalisada.

b) Na segunda reação temos uma associação de duas proteínas *prot1* e *prot2* que formam um complexo molecular que pode se dissociar preservando as substâncias incorporadas.

c) Neste diagrama é exposta o transporte de um íon do interior para o exterior de uma célula por meio de um canal de íon aberto que habilita sua passagem.

d) Representa uma reação que ocasiona degradação de uma proteína modificada em relação a original pela ausência de alguns aminoácidos.

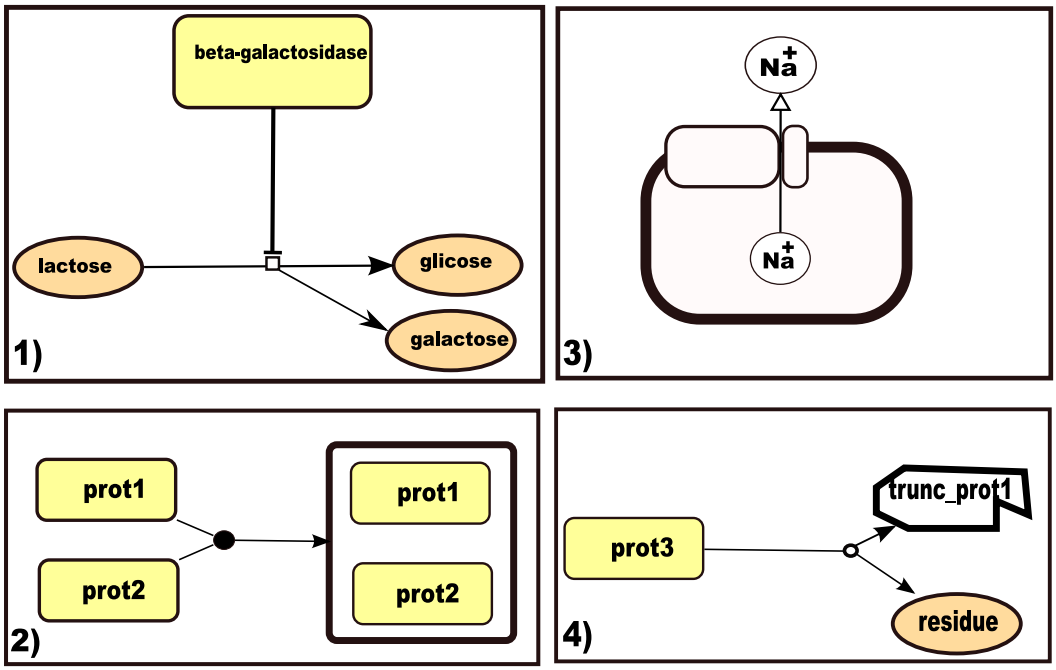


Figura 2.5: Diagramas exemplos escritos em notação de diagrama de processos.

2.2 Considerações Finais

Este capítulo apresentou a definição de vias biológicas, além disso como são geradas e representadas em diagramas. Embora exista uma descrição formal de grafos que representa um diagrama de processos, esta descrição aborda somente partes dos aspectos dos diagramas. A descrição formal de uma linguagem precisa de regras de construção para tais diagramas bem como o significado (semântica). Por isso podemos considerar a definição existente para diagrama de processos como uma descrição semi-formal.

Descrições formais eliminam ambiguidades e dúvidas que podem ocorrer na especificação. Elas também reduzem a possibilidade de construção de modelos errôneos (modelos que não correspondem a realidade), pois permitem raciocínio formal. Com um modelo mais fidedigno, predições mais efetivas podem ser feitas pela análise do modelo. No próximo capítulo serão apresentados trabalhos relacionados com a especificação e modelos semânticos de simulação das vias biológicas.

3 TRABALHOS RELACIONADOS E MOTIVAÇÃO

Neste capítulo apresentamos os trabalhos relacionados em quatro seções: na primeira são apresentadas linguagens de especificação de vias biológicas baseadas em XML voltadas principalmente para armazenamento de vias biológicas em bancos de dados. Na segunda seção, apresentamos alguns modelos semânticos usados para representar a evolução das concentrações das substâncias. Na terceira seção apresentamos os problemas das técnicas usadas atualmente e comparamos com a nossa proposta.

3.1 Linguagens de especificação de vias biológicas

Nesta seção são apresentadas algumas linguagens de especificação de vias biológica. Na subseção 3.1.1 , apresentamos SBML que se tornou um padrão XML para armazenamento de vias biológicas definindo estruturas de listas para todos os componentes do sistema. Na subseção 3.1.2, é apresentada BioPNML uma linguagem de especificação onde os dados a serem armazenados devem ser estruturados diretamente em redes de Petri.

3.1.1 SBML

SBML é uma linguagem de especificação de vias metabólicas baseada em XML, e segundo (HUCKA et al., 2003) foi definida usando UML. A definição básica é o resultado da análise de características comuns nas representações usadas em equações diferenciais e simuladores estocásticos de cálculos de processos.

Uma reação química é formada de reagentes, produtos, reações, estequiometria e taxas associadas as leis de reações. Em SBML os modelos são listas de um ou mais destes componentes:

- Compartimento - um invólucro de volume finito para substâncias fortemente ativas onde ocorre reações;
- Espécie - uma substância química ou entidade que faz parte de uma reação. Ex: Íons de cálcio e moléculas como ATP;
- Reação - representa transformações que podem ocorrer em uma ou mais espécies. Reações tem taxas associadas que descrevem a maneira no qual elas são consumidas ou aumentadas no decorrer do tempo;
- Parâmetro - um quantidade que possui um nome simbólico. Em SBML é possível definir parâmetros que são globais ao modelo, bem como os parâmetros que são locais a uma reação isolada;

- Definição de unidade - um nome para uma unidade usada na expressão de quantidades no modelo;
- Regras - uma expressão matemática que é adicionada para o modelo de equações construído do conjunto de reações. Regras podem ser usadas para mudar valores de parâmetros estabelecer transferências quantitativas entre reagentes;

Como exemplo é mostrada uma etapa do circuito oscilatório de um gene em uma célula eucariótica na figura 3.1.

Reaction	Rate
$src \rightarrow RNAP$	$V_i/(1+P/K_i)$
$RNAP \rightarrow waste$	$V_{kd} \cdot RNAP$
$RNA_{nuc} \rightarrow mRNA_{nuc}$	$\frac{V_{m1} \cdot RNAP \cdot RNA_{nuc}}{K_{m1} + RNA_{nuc}}$
$mRNA_{nuc} \rightarrow mRNA_{cyt}$	$k_1 \cdot mRNA_{nuc}$
$mRNA_{cyt} \rightarrow RNA_{cyt}$	$\frac{V_{m2} \cdot mRNA_{cyt}}{mRNA_{cyt} + K_{m2}}$
$RNA_{cyt} \rightarrow RNA_{nuc}$	$k_2 \cdot RNA_{cyt}$
$AA \rightarrow P$	$\frac{V_{m3} \cdot mRNA_{cyt} \cdot AA}{AA + K_{m3}}$
$P \rightarrow AA$	$(V_{m4} \cdot P)/(P + K_{m4})$

Fonte: (HUCKA et al., 2003)

Figura 3.1: Reações do circuito oscilatório de um gene.

Na figura 3.2 apresentamos uma descrição em SBML para um simples processo de transformação do RNA nuclear para RNA nuclear mensageiro definido pela lei de reação 3.1 e taxa 3.2.

$$RNA_{nuc} \rightarrow mRNA_{nuc} \quad \text{reação} \tag{3.1}$$

$$\frac{Vm_1 \cdot RNAP \cdot RNA_{nuc}}{Km_1 + RNA_{nuc}} \quad \text{taxa} \tag{3.2}$$

```
    <listOfCompartments>
<compartment name="Nuc"outside="Cyt"/>
.
</listOfCompartments>

    <listOfSpecies>
<species name="mRNA_nuc"compartment="Nuc"initialAmount="0.0032834"/>
<species name="RNA_nuc"compartment="Nuc"initialAmount="96.117"/>
.
.
</listOfSpecies>

    <listOfReactions> <reaction name="R1"reversible="false»
<listOfReactants>
<species Reference species="mRNA_nuc"/>
</listOfReactants>
<listOfProducts>
<species Reference species="RNA_nuc"/>
</listOfProducts>
<kineticLaw formula="Vm1 · RNAP· RNA_nuc/Km1 + RNA_nuc"/>
</reaction>
```

Fonte (HUCKA et al., 2003).

Figura 3.2: Descrição SBML do processo de transformação do RNA nuclear.

3.1.2 BioPNML

BioPNML (Biology Petri Net Markup Language) é também uma linguagem baseada em XML, no entanto formalmente definida usando Redes de Petri. Proposta por (CHEN et al., 2002), a meta é representar redes metabólicas com redes de Petri. Para isto foi proposto um mapeamento dos termos metabólicos para os termos de redes de Petri conforme a ilustra a tabela 3.1.

Metabólitos,genes,promotores,sinais	Lugares
Reação, interação,outros processos	Transições
Reagentes dos processos	Arcos de Entrada
Produtos dos processos	Arcos de saída
Token inicial ou estado do sistema	Marcação inicial

Fonte: (CHEN et al., 2002).

Tabela 3.1: Mapeamento de termos.

A estrutura XML do BioPNML utiliza as mesmas tags usadas em SBML, acrescidas de *idref tags* que são usadas para ligar os lugares e as transições às suas respectivas tags espécies (CHEN et al., 2002).

3.2 Modelos semânticos propostos para simulação de vias biológicas

3.2.1 Equações diferenciais

Nesta primeira abordagem a criação do modelo é focada na identificação de uma lei de reação que represente a evolução da concentração de cada entidade biológica em estudo. Diferentes formatos de equações diferenciais já foram propostos em sistemas biológicos (equações de Hill, de ação de massa generalizada, sistemas de equações, etc.) (VOIT, 2001).

Em um dos mais simples, descrito em (L. BORTOLUSSI, 2006), os sistemas de equações descrevem a evolução de variáveis dependentes $(X_1,...,X_n)$ e independentes $(X_{n+1},...,X_{n+m})$, representando quantidades associadas ao substrato e fixadas no experimento. O formato geral para estas equações é é representado pela equação 3.3.

$$X_i = V_i^+(X_1,...,X_{n+m}) - V_i^-(X_1,...,X_{n+m}) \tag{3.3}$$

A velocidade na qual X_i aumenta ou diminui é determinado pelo termos de produção e degradação $((V_i^+)$ e (V_i^-), respectivamente). Os termos V contém todas as variáveis que diretamente influenciam o comportamento dos termos X_i expressos pelas equações 3.4 e 3.5 respectivamente:

$$V(X_1,,X_{n+m}) = \alpha\Pi_{j=1}^{n+m} * X_j^{h_j} \tag{3.4}$$

$$X_i = \alpha_i\Pi_{j=1}^{n+m} X_j^{g_{ij}} - \beta_i\Pi_{j=1}^{n+m} X_j^{h_{ij}} \tag{3.5}$$

Os parâmetros α_i, β_i são taxas constantes que representam as taxas básicas de produção e degradação para cada variável dependente. Os parâmetros g_{ij} e h_{ij} representam a força da interação da entidade biológica X_j na produção ou degradação de X_i. Caso o valor seja positivo (efeito de aumento), negativo (efeito de inibição) ou zero (nenhum efeito).

Todos os termos envolvidos tem um significado biológico e representam um conjunto de comportamentos. Um exemplo de modelo possível descrito com sistemas de equações onde LacI, tetR e λci são proteínas regulatórias é descrito pelo conjunto de equações a seguir:

$$\begin{aligned} tetR &= \alpha_1 LacI^{-1} - \beta_1 tetR^{0.5}, \quad \alpha_1 = 0.2 \quad \beta_1 = 1 \\ \lambda ci &= \alpha_2 LacI^{-1} - \beta_2 \lambda ci^{0.5}, \quad \alpha_2 = 0.2 \quad \beta_2 = 1 \\ LacI &= \alpha_3 \lambda ci^{-1} - \beta_1 LacI^{0.5}, \quad \alpha_3 = 0.2 \quad \beta_3 = 1 \end{aligned}$$

Equações Diferenciais são modelos matemáticos onde o nível de abstração é baixo dificultando a verificação de propriedades (JONG, 2002), principalmente ao formular propriedades que se refiram à estrutura que o conjunto de equações representa. Propriedades estruturais relacionadas a grafos e cadeias de Markov, por exemplo, são úteis ao abranger uma gama maior de análise, que vai além da avaliação de valores resultantes de variáveis e equações.

3.2.2 Cálculos de processos

Foi proposto o uso de vários cálculos de processos como por exemplo CCS (PINTO et al., 2007), CLS (BARBUTI et al., 2006), cálculo-π (CURTI M.; DEGANO; BALDARI, 2004), para descrever modelos de redes bioquímicas. Em cálculo-π as moléculas

e seus domínios individuais são tratados como processos computacionais, onde seus determinantes químicos e estruturais correspondem aos canais de comunicação.

Um exemplo simples modelado em cálculo-π é a reação de ionização de uma solução de Na (sódio) e Cl (cloro). Estas duas substâncias em solução alternam mudanças de estados apresentando-se ora como íons Na^+ e Cl^- , ora nos seus estados originais. Admitindo-se por exemplo que a conversão de Na para Na^+ e de Cl para Cl^- possua uma taxa estocástica 100 e a reação reversa o valor 10, a especificação completa deste processo seria descrita como:

$$
\begin{aligned}
Na &\overset{(\overline{ionize},100)}{\longrightarrow} Na^+ \\
Na^+ &\overset{(\overline{deionize},10)}{\longrightarrow} Na \\
Cl &\overset{(ionize,100)}{\longrightarrow} Cl^- \\
Cl^- &\overset{(deionize,10)}{\longrightarrow} Cl \\
&\qquad\qquad Na|Cl
\end{aligned}
$$

As portas *ionize* e *deionize* são os canais de comunicação dos processos envolvidos. Nesta variante do cálculo-π as taxas estocásticas representam o comportamento probabilístico dos processos.

3.3 Motivação

Alguns estudos já foram feitos relacionando a topologia de redes biológicas com grafos (KLAMT; HAUS; THEIS, 2009), onde os nodos representam metabólitos, proteínas, ou outras entidades biológicas, e as arestas representam relações ou interações entre estes nodos como associação, catalização, inibição, transição etc. Muitos processos biológicos, entretanto, podem relacionar em uma aresta mais do que dois participantes necessitando de uma representações baseada em hipergrafos. Propriedades já amplamente estudadas em teoria dos grafos podem ser usadas na verificação de caraterísticas estruturais de redes biológicas. Torna-se natural então formalizar estas descrições biológicas com grafos.

As linguagens de especificação baseadas em XML, como SBML, a sintaxe é apenas textual e são necessárias ferramentas de apoio a verificação sintática com algum formalismo que defina as regras de transformação das especificações e associe os diagramas à sintaxe abstrata XML. Num documento SBML temos um sistema de tipos muito simples para especificar vias onde a tipagem é feita apenas nas listas sem tipagem dos elementos que a compõem. Ferramentas que usem esta linguagem de descrição como sintaxe abstrata podem gerar diagramas inconsistentes.

Como exemplo, no padrão dos diagramas de processos uma associação representa uma ligação não covalente entre duas substâncias (moléculas simples, proteínas, etc) na formação de um complexo ou um multímero (NOVERE et al., 2008). O diagrama ilustrado na figura 3.3 representa a associação entre duas proteínas (s6 e s7 respectivamente) formando uma proteína s10. Temos um descrição correta sintaticamente definindo as listas de espécies e reações segundo o padrão SBML. Porém, este não é um diagrama de processos válido (sintaticamente correto).

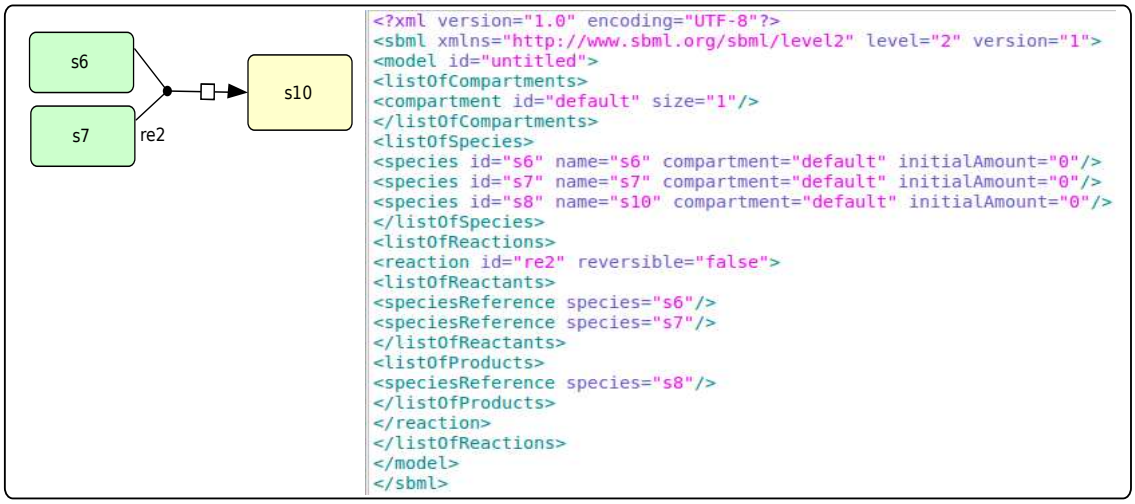


Figura 3.3: Diagrama de processos inválido.

Outro problema encontrado é como verificar que os diagramas de processos estão corretamente traduzidos para uma rede de Petri ou cálculo de processos. Suponha que um usuário altere um diagrama de processos onde temos três proteínas A, B e C e um complexo D. Ao manipular a especificação ele cria uma associação entre A e B gerando como produto a proteína C. Logo após o usuário modelar a especificação gera uma rede Petri com três lugares e uma transição com dois arcos de entrada e um arco de saída. Sintaticamente foi gerada uma rede Petri correta. Entretanto esta rede não corresponde a um diagrama de processos válido.

Já foram feitas traduções de gramáticas de grafos para modelos simuláveis baseados em cadeia de Markov. Um trabalho apresentado por (HECKEL; LAJIOS; MENGE, 2006) propõe uma extensão ao *framework* original no qual uma taxa é associada a cada regra de transformação. Entretanto ao usar cadeias de Markov diretamente recai-se no problema da explosão de estados. Há modelos markovianos mais abstratos como Redes de Petri Estocásticas Generalizada (GSPNs) (BRENNER et al., 2005) onde há técnicas que permitem uma otimização da geração do espaço de estados que normalmente tendem a crescer exponencialmente quando grandes vias são modeladas.

3.4 Considerações Finais

Este capítulo apresentou trabalhos relacionados ao estado da arte sob dois aspectos:

- No primeiro foi mostrado como são feitas as descrições de vias biológicas com as linguagens de marcação baseadas em XML.
- No segundo foram mostradas linguagens usadas para simular a evolução das concentrações das substâncias participantes por meio de métodos formais matemáticos.

A partir dessas observações foram apontados problemas relacionados aos sistemas de tipos das linguagens de descrição, que podem gerar especificações sintaticamente incorretas e aos modelos semânticos usados para definir a evolução das concentrações, que devido ao baixo grau de abstração dificultam estudos relacionados a verificação de propriedades das vias biológicas.

Em nossa abordagem, será apresentada uma formalização dos diagramas de processos usando gramática de grafos tipadas com atributos. Esta gramática foi proposta em

(BARDOHL, 1999) como um modelo formal para geração de linguagens visuais. Serão apresentadas nas próximas seções metamodelos formais que possibilitarão a verificação sintática da especificação com a checagem de erros da construção da via e a associação formal do diagrama e sua especificação textual. Também será apresentada uma tradução semântica para as redes de Petri estocásticas generalizadas que possibilita a verificação da evolução das concentrações e a verificação de propriedades inerentes ao sistema. Este framework possibilita que as etapas que procedem a especificação da via biológica (simulação e verificação de propriedades) se tornem automáticas (mostradas em destaque na figura 3.4).

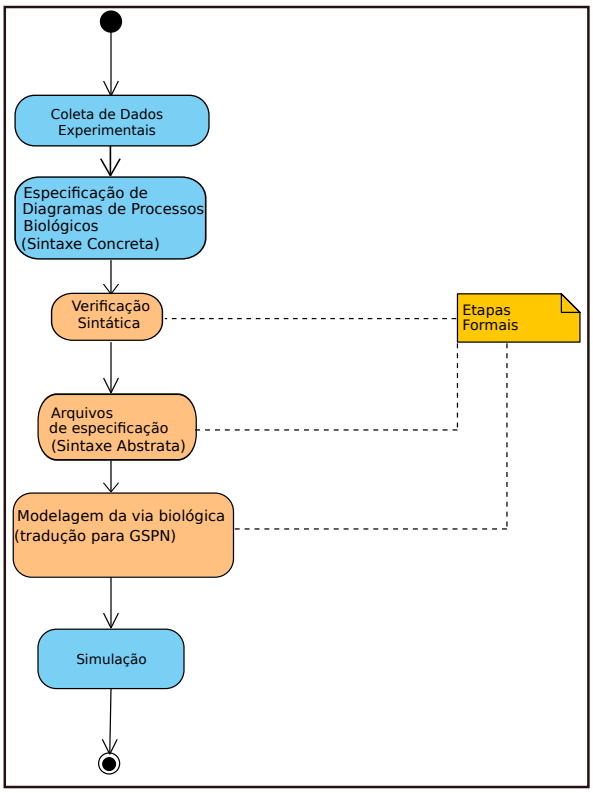


Figura 3.4: Solução Proposta

4 METAMODELO PARA OS DIAGRAMAS DE PROCESSOS BIOLÓGICOS

Neste capítulo será apresentado um metamodelo formal para diagramas de processos biológicos em duas seções: na seção 4.1 são apresentados conceitos básicos de uma linguagem formal visual, já na seção 4.2 são apresentadas as sintaxes abstrata e concreta da linguagem visual formalizada nesta dissertação.

4.1 Definição de uma Linguagem Formal Visual

Antes de apresentamos o conceito de linguagens visuais apresentamos o conceito geral de uma linguagem formal textual:

- Alfabeto - conjunto finito não vazio de símbolos e elementos que podem ser usados numa linguagem. Ex: $V = \{a, b, c..., z\}$
- Palavra ou sentença - é uma cadeia de símbolos gerada a partir de um alfabeto.
- Tamanho de uma sentença - o tamanho de uma sentença *w* é definida pelo número de símbolos que a compõe representado por $|w|$. Ex: $|aba| = 3$
- Fechamento transitivo - representa o conjunto de todas as palavras possíveis de serem construídas a partir de um determinado alfabeto. É representado pelo símbolo $\star$. Ex: $V^\star = \{a, aba, abbab, ...\}$
- Fechamento positivo - representa o conjunto de todas as palavras possíveis de serem construídas com um determinado alfabeto excetuando a palavra vazia (representado por ε). É representada pelo símbolo $+$. Onde $V^+ = V^\star - \{\varepsilon\}$.

Definição 4.1.1 (Gramática de uma Linguagem Formal) *A gramática de uma linguagem formal é definida por uma quádrupla: $LF = (T, N, P, S)$ onde:*

- *T é um alfabeto de símbolos chamados terminais.*
- *N é um alfabeto de símbolos chamados não-terminais.*
- *P é um conjunto de produções que representam as regras de reescrita de termos que geram as sentenças da linguagem, representado por um conjunto finito de pares (α, β) :*
  - *$\alpha \in (T \cup N)^\star N (T \cup N)^\star$ - é o lado esquerdo da regra.*
  - *$\beta \in (T \cup N)^\star$ - é o lado direito da regra.*

- *$S \in N$ é um símbolo inicial da linguagem, ponto de partida para a escrita de qualquer sentença.*

Definição 4.1.2 (Derivação) *consiste na aplicação sucessiva de regras de produção substituindo uma subpalavra de acordo com uma regra de produção. É formalmente definida por um par da relação de derivação denotado por $\Rightarrow$ com:*

- *Domínio em $(T \cup N)^+$ e codomínio $(T \cup N)^\star$*
- *Um par $\langle a, b \rangle$ da relação denotado de forma infixada por $a \Rightarrow b$.*

$\Rightarrow$ é definida indutivamente da seguinte forma:

- *Para toda produção na forma $S \rightarrow b$ onde S é o símbolo inicial da gramática tem-se que $S \Rightarrow b$*
- *Para todo par $k \Rightarrow \rho v \sigma$, da relação de derivação , se $v \rightarrow t$ é regra de P, então $k \Rightarrow \rho t \sigma$.*

Definição 4.1.3 (Linguagem Formal) *define-se como o conjunto de todas as sentenças que podem ser derivadas pela aplicação do conjunto de produções de uma gramática, ou seja : $L = \{w \in T^\star | S \Rightarrow^\star w\}$ onde w representa uma possível sentença que pode pertencer à linguagem.*

Em uma linguagem formal visual a sintaxe concreta é definida em termos de diagramas (elementos gráficos). A sintaxe abstrata por sua vez define o componente lógico da linguagem descrito por grafos. A definição de uma linguagem visual deve explicitar como a ligação entre a sintaxe abstrata e concreta é feita. Na figura 4.1 ilustra-se a ideia básica da representação das duas sintaxes.

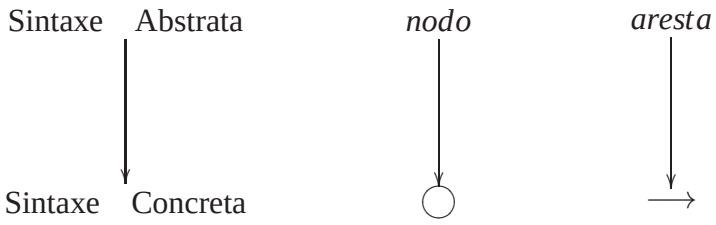


Figura 4.1: Sintaxe abstrata e concreta de um grafo simples.

A linguagem visual é portanto o conjunto de todas as sentenças visuais (diagramas) que podem ser derivadas da gramática. A geração das sentenças de linguagens visuais é feita a partir de regras de produções da gramática da linguagem visual. A seguir será apresentado o conceito de produção numa gramática de grafos e como este conceito foi inserido no modelo conceitual de linguagens visuais.

A semântica formal de uma linguagem mapeia cada sentença da linguagem para um modelo matemático que define seu significado. O conjunto de sentenças de uma linguagem só adquire um significado (semântica) no contexto do modelo que interpreta a estrutura(TURBAK; GIFFORD, 2008). O modelo matemático mais adequado à interpretação semântica varia com a linguagem que está sendo descrita.

4.2 Sintaxe

Sistemas biológicos são complexos para abordar e descrever de modo intuitivo. As representações diagramáticas utilizadas na biologia podem ser modeladas por grafos cujos vértices podem representar um conjunto extenso de substâncias bioquímicas e os arcos representam reações (transições de estados nas substâncias envolvidas).

Nesta seção definimos o metamodelo para diagrama de processos. Este metamodelo será usado como base para as regras da gramática que definem a linguagem visual. O metamodelo é um tipo especial de grafo com uma representação visual associada. Este grafo será, portanto, uma representação abstrata do diagrama de processos. Nós usaremos regras sobre grafos para definir a geração de todos diagramas de processos sintaticamente corretos. A base para nossa abordagem é a definição de hipergrafos com atributos (figura 4.2), que são grafos nos quais cada aresta pode ter mais de um vértice origem e destino, e nos quais os vértices e arestas possuem atributos (valores) associados.

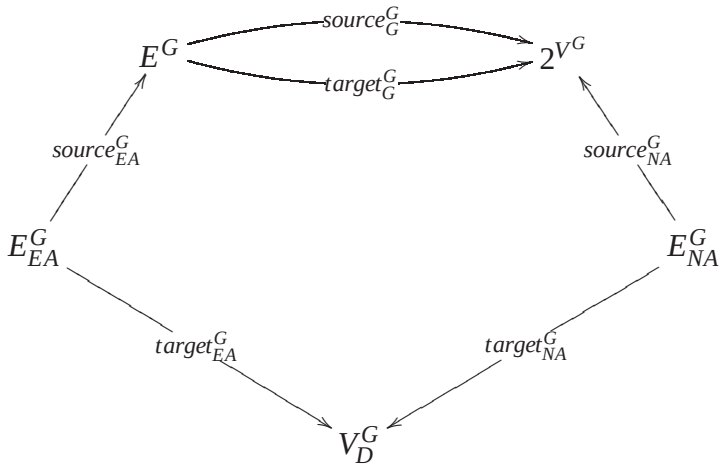


Figura 4.2: Um grafo com atributos associados aos arcos e arestas.

A seguir apresentaremos nas definições 4.2.1, 4.2.2, 4.2.3 os conceitos genéricos de um hipergrafo com atributos, morfismo de grafos e grafo tipado. Tais definições são necessárias para a compreensão da seção 4.3 na qual são apresentadas as definições do metamodelo proposto.

Definição 4.2.1 (Grafo) *Um (hiper)grafo com atributos é uma tupla :*
$G = (V^G_G, V^G_D, E^G_G, E^G_{NA}, E^G_{EA}, (source^G_j, target^G_j)_{j\in\{G,NA,EA\}})$*, onde:*

- V^G_G *e* V^G_D *são conjuntos cujos elementos são os vértices do grafo e os vértices de dados respectivamente;*
- E^G_G, E^G_{NA} *e* E^G_{EA} *são conjuntos cujos elementos são as arestas do grafo, os atributos do nodos e os atributos das arestas, respectivamente;*
- $source^G_G : E^G_G \to V^{G*}_G$, $target^G_G : E^G_G \to V^{G*}_G$ *são funções totais que definem listas de vértices como origem e destino de cada aresta do grafo;*

- *$source^G_{EA} : E^G_{EA} \to E^G_G$, $target^G_{EA} : E^G_{EA} \to V^G_D$ são funções totais que definem vértices origem e o destino para cada atributo de aresta;*
- *$source^G_{NA} : E^G_{NA} \to V^G_G$, $target^G_{NA} : E^G_{NA} \to V^G_D$ são funções totais que definem vértices origem e o destino para cada atributo de vértice;*

Dois grafos são relacionados entre si por morfismos que mapeiam os nodos e arestas do primeiro grafo para os do segundo grafo, preservando a origem e destino de cada aresta.

Definição 4.2.2 (Morfismo de grafos) *Dados dois grafos com atributos G^1 e G^2, $G^i = (V^{G_i}_G, V^{G_i}_D, E^{G_i}_G, E^{G_i}_{NA}, E^{G_i}_{EA}, (source^{G_i}_j, target^{G_i}_j)_{j \in \{G_i, NA, EA\}})$ para $k = 1,2$. Um morfismo parcial de grafos $g : G^1 \to G^2$ é uma tupla $f = (f_{V_G}, f_{V_D}, f_{E_G}, f_{E_{NA}}, f_{E_{EA}})$ onde $f_{V_k} : V^{G_1}_k \to V^{G_2}_k$ e para $k \in \{G, D\}$ e $f_{E_j} : E^{G_1}_j \to E^{G_2}_j$ para $j \in \{G, NA, EA\}$, são funções parciais, tal que f comuta em todas as funções source e target , isto é, respeitem origem e destino. Se todos os componentes de um morfismo são totais nós dizemos que o morfismo é total.*

Diagramas de processos biológicos, apresentados no capítulo 2, tem uma estrutura particular. Nós usamos um mecanismo de tipagem para vértices e arestas para distinguir entre os diferentes tipos de elementos que aparecem nestes diagramas. Formalmente, este conceito de tipagem será dado pela definição de um grafo no qual vértices e arestas representam os tipos de elementos, chamado grafo de tipos. Os tipos dos elementos de cada grafo serão definidos pelo mapeamento deste grafo para o grafo de tipos (através de um morfismo).

Definição 4.2.3 (Grafo tipado) *Dado um grafo TG do qual chamamos de grafo de tipos. Um grafo tipado sobre TG é uma tupla $(G, type)$ onde G é um grafo e $type : G \to TG$ é um morfismo total de grafos.*

4.3 Definições do Metamodelo

Os diagramas de processos são uma classe especial de grafos tipados, onde cada nodo representa alguma das espécies químicas e cada aresta representa uma reação química. Antes de definirmos o grafo de tipos destes diagramas, apresentaremos seis definições que a comporão nesta ordem: conjunto de vértices (Def. 4.3.1) e arestas (Def. 4.3.2) do grafo, o conjunto de vértices de dados (Def. 4.3.3), os atributos dos vértices e das arestas (4.3.4 e 4.3.5) e finalmente as funções *source* e *target* (Defs. 4.3.6 , 4.3.7 e 4.3.8).

Definição 4.3.1 (Tipos de vértices) *O conjunto de nodos que fazem parte do grafo tipo denotado por V^{Bio}_G é definido por:*

$$\begin{aligned} V^{Bio}_G = \{&Protein, Receptor, Ion\_Channel, Truncated\_Protein, Simple\_Molecule,\\ &Unknown\_Molecule, Phenotype, Homodimer,\\ &Ion, Gene, RNA, Antisense\_RNA, Complex, Compartment,\\ &And, Or, Empty\_Set, Legends\_Concentrations\} \end{aligned}$$

A descrição das substâncias representadas pelos tipos são as mesmas apresentadas na seção 2.1. Além disso, acrescentamos tipos especiais como *Legends_Concentrations* que representa as faixas de concentrações das substâncias. Ele define intervalos de valores que as concentrações das substâncias podem assumir.

Definição 4.3.2 (Tipos de arestas) *O conjunto de arestas que fazem parte do grafo tipo denotado por E_G^{Bio} é definido por:*

$$E_G^{Bio} = \{State\_Transition, Unknown\_Transition, Known\_Transition\_Ommited, \\ Translocation, Bidirectional\_Transition, \\ Association, Dissociation, Truncation, In\_Compartment, \\ In\_And, In\_Or, In\_Complex\}$$

Foram definidos 8 tipos de arestas já presentes da notação definida na seção 2.1. Entretanto, nas vias biológicas não são expressas somente reações químicas. Há representações de compartimentos pelos tipos *Compartment* e *Complex,* além dos nodos lógicos *And* e *Or*. Para representar estas relações foram especificados 4 novos tipos de arestas:

- *In_Compartment* - que associa uma instância do tipo *Compartment* a uma lista de substâncias englobadas;
- *In_Complex* - que associa uma instância do tipo *Complex* a uma lista de substâncias agregadas.
- *In_And* - que associa uma instância do tipo *And* a uma lista de substâncias onde todos os todos os participantes são necessários para inibir ou estimular a reação.
- *In_Or* - que associa uma instância do tipo *Or* a uma lista de substâncias onde pelo menos um dos participantes são necessários para inibir ou estimular a reação.

A cada vértice e aresta foram associados alguns atributos. Originalmente, os grafos que representam diagramas de processos não tem atributos associados a vértices e arestas. Entretanto, tais atributos são necessários para obter modelos que são adequados para análise, e devem portanto fazer parte da sintaxe abstrata da linguagem. A escolha dos atributos usados aqui foi baseada no banco de dados biológicos BioModels (LE NOVèRE et al., 2006). Eles são os parâmetros necessários para a simulação/análise de processos biológicos, bem como os parâmetros da descrição textual.

Definição 4.3.3 (Definição dos vértices de dados) *O conjunto formado pelos vértices de dados é definido pelo conjunto V_D^{Bio}. Este conjunto define o tipo dos atributos dos vértices e das arestas:*

$$V_D^{Bio} = \{Integer, Double, String, Boolean, List[String], \\ List[Integer], List[pair(Integer, Integer)]\} \tag{4.1}$$

Definição 4.3.4 (Definição dos atributos dos vértices) *O conjunto de atributos dos vértices é denotado pelo conjunto E_{NA}^{Bio} :*

$$E_{NA}^{Bio} = \{name\_elem, amount, state, units, intervals, colors\} \tag{4.2}$$

Intuitivamente os nomes dos atributos tem os seguintes significados e tipos:

- *name_elem* - nome científico da substância participante. O tipo deste atributo é *String*.
- *amount* - quantidade da substância presente na via biológica. O tipo deste atributo é *Integer*.

- *state* - estado da proteína que indica se ela está no estado normal ou ativada do tipo *Boolean*.

- *units* - define que tipo de unidades são usadas para definir as faixas de concentrações. Seu tipo associado é *String*.

- *intervals* - define as faixas de concentrações que cada substância pode assumir. Seu tipo associado é uma lista de pares de inteiros onde o primeiro elemento representa o valor mínimo do intervalo e o segundo elemento o valor máximo do intervalo.

- *colors* - define as cores associadas às faixas de intervalo. Seu tipo associado é uma lista de inteiros.

Definição 4.3.5 (Definição dos atributos das arestas) *O conjunto de atributos das arestas é denotado pelo conjunto* E_{EA}^{Bio} *:*

$$E_{EA}^{Bio} = \{name\_react, rate\_law, react\_parameter\_tax, \\ participants, prod\_parameter\_tax, condition\_trigger\}$$

Intuitivamente os nomes dos atributos tem os seguintes significados e tipos:

- *name_react* - nome científico que identifica a reação. Seu tipo associado é uma String.

- *rate_law* - indica a velocidade com a qual a reação ocorre. Pode ser uma taxa constante ou uma expressão dependente de parâmetros como a quantidade das substâncias participantes da reação. Seu tipo é definido por uma String.

- *react_parameter_tax* - lista de parâmetros que representam os coeficientes estequiométricos de cada reagente da reação. Seu tipo é uma lista de inteiros (*List[Integer]*).

- *priority* - define uma prioridade para eventos que ocorrem de forma imediata com um valor *Integer*. Quando há reações concorrentes de um pathway disputando um mesmo reagente, pode-se atribuir um valor que devidamente normalizado determina a maior ou menor probabilidade de uma delas ocorrer.

- *participants* - lista de rótulos que definem como a substância participará da reação seja como simples participante, inibidor ou promotor da reação. O tipo definido para este atributo é uma lista cujos elementos são do tipo *String*.

- *condition_trigger* - condição necessária para que ocorra a reação. Por exemplo quando uma determinada substância (que pode fazer parte ou não da reação) atinge um determinado quantidade. Seu tipo é *Boolean*, quando nenhuma condição é estipulada seu valor é *true*.

- *prod_parameter_tax* - lista de parâmetros que representam os coeficientes estequiométricos dos produtos da reação. Seu tipo é uma lista de inteiros (*List[Integer]*).

Agora serão definidas as funções source e target para as arestas dos grafos determinando quais tipos de nodos podem ser relacionados as funções source e target de cada uma. Para definir essas funções usaremos uma notação baseada em BNF (Backus-Naur Form) (KNUTH, 1964). Os elementos dos conjuntos V^{Bio}, V_D^{Bio}, E^{Bio}, E_{EA}^{Bio} e E_{NA}^{Bio} serão interpretados como símbolos terminais. Listas serão representadas entre colchetes []. O operador | indica escolha (basicamente, união de conjuntos). O operador + indica que a lista deve ter pelo menos um elemento. Por exemplo, dada uma aresta $x \in E^{Bio}$ e vértices $ny, nz, nw \in V_G^{Bio}$, as funções $source(x) = [(ny|nz)^+]$ e $target(x) = [(nw)]$ definem que a origem da aresta x deve ser uma lista de qualquer tamanho finito não-vazia cujos elementos podem ser os tipos ny ou nz e o destino desta aresta é uma lista unitária contendo o tipo nw.

Definição 4.3.6 (Definição das funções $source_G^{Bio}$ e $target_G^{Bio}$) *Sejam:*

$$\begin{aligned} simple\_sub &= \begin{array}{c} Protein|Ion\_Channel|Receptor|Simple\_Molecule \\ Ion|Unknown\_Molecule|Truncated\_Protein \end{array} \\ genetic\_elem &= Gene|RNA|Antisense\_RNA \\ complex\_sub &= Homodimer|Complex \\ logic &= And|OR \\ transitions &= \begin{array}{c} State\_Transition|Known\_Transition\_Omitted| \\ Unknown\_Transition|Bidirectional\_Transition \end{array} \end{aligned} \tag{4.3}$$

As funções $source_G^{Bio}$ e $target_G^{Bio}$ que associam a cada aresta de E_G^{Bio} uma lista de vértices de V_G^{Bio}, são definidas por:

i) arestas que representam transições $\forall t \in transitions$

$$\begin{aligned} source_G^{Bio}(t) &= [(type\_src\_transition)^+] \\ target_G^{Bio}(t) &= [(type\_tgt\_transition)^+] \end{aligned}$$

onde :

$$\begin{aligned} type\_src\_transition &= simple\_sub|genetic\_elem|complex\_sub|logic|Empty\_Set \\ type\_tgt\_transition &= simple\_sub|genetic\_elem|complex\_sub|Empty\_Set \end{aligned} \tag{4.4}$$

ii) arestas representando translocação:

$$\begin{aligned} source_G^{Bio}(Translocation) &= [(type\_src\_transition)^+] \\ target_G^{Bio}(Translocation) &= [(type\_tgt\_transition)^+] \end{aligned}$$

onde :

$$\begin{aligned} type\_src\_translocation &= simple\_sub|genetic\_elem|complex\_sub|logic \\ type\_tgt\_translocation &= simple\_sub|genetic\_elem|complex\_sub \end{aligned} \tag{4.5}$$

iii) arestas representando associação:

$$source_G^{Bio}(Association) = [(type\_src\_association)^+]$$
$$target_G^{Bio}(Association) = [type\_tgt\_association]$$

onde :

$$type\_src\_association = simple\_sub|complex\_sub$$
$$type\_tgt\_association = Homodimer|Complex \tag{4.6}$$

iv) arestas representando dissociação:

$$source_G^{Bio}(Dissociation) = [type\_src\_dissociation]$$
$$target_G^{Bio}(Dissociation) = [(type\_tgt\_dissociation)^+]$$

onde :

$$type\_src\_dissociation = type\_tgt\_association$$
$$type\_tgt\_dissociation = type\_src\_association$$

v) arestas representando truncagem:

$$source_G^{Bio}(Truncation) = [type\_src\_truncation]$$
$$target_G^{Bio}(Truncation) = [Truncated\_Protein, (type\_tgt\_truncation)^+]$$

onde :

$$type\_src\_truncation = Protein|Complex|Homodimer$$
$$type\_tgt\_truncation = simple\_sub \tag{4.7}$$

vi) arestas representando complexos biologicos:

$$source_G^{Bio}(In\_Complex) = [(type\_src\_in\_complex)^+]$$
$$target_G^{Bio}(In\_Complex) = [Compartment] \tag{4.8}$$

onde :

$$type\_src\_in\_complex = simple\_sub|genetic\_elem|complex\_sub$$

vii) arestas representando compartimentalização:

$$source_G^{Bio}(In\_Compartment) = [(type\_src\_in\_compartment)^+]$$
$$target_G^{Bio}(In\_Compartment) = [Compartment] \tag{4.9}$$

onde :

$$type\_src\_in\_compartment = simple\_sub|genetic\_elem|Complex|Homodimer \tag{4.10}$$

viii) arestas representando os operadores lógico And e Or:

$$\begin{aligned} source_G^{Bio}(In\_Or) &= [(type\_src\_in\_logical)^+] \\ target_G^{Bio}(In\_Or) &= [Or] \\ source_G^{Bio}(In\_And) &= [(type\_src\_in\_logical)^+] \\ target_G^{Bio}(In\_And) &= [And] \end{aligned}$$

onde :

$$type\_src\_in\_logical = simple\_sub|genetic\_elem|complex\_sub \tag{4.11}$$

Definição 4.3.7 (Definição das funções $source_{NA}^{Bio}$ e $target_{NA}^{Bio}$) *As funções $source_{NA}^{Bio} : E_{NA}^{Bio} \rightarrow V_G^{Bio}$ e $target_{NA}^{Bio} : E_{NA}^{Bio} \rightarrow V_D^{Bio}$ das arestas E_{NA}^{Bio}são definidas por :*

i) aresta representando o atributo name_react:

$$\begin{aligned} source_{NA}^{Bio}(name\_elem) &= type\_src\_name \\ target_{NA}^{Bio}(name\_elem) &= String \end{aligned}$$

onde :

$$type\_src\_name = simple\_sub|genetic\_elem|complex\_sub \tag{4.12}$$

ii) aresta representando o atributo amount:

$$\begin{aligned} source_{NA}^{Bio}(amount) &= type\_src\_amount \\ target_{NA}^{Bio}(amount) &= Integer \end{aligned}$$

onde :

$$type\_src\_amount = simple\_sub|genetic\_elem|complex\_sub \tag{4.13}$$

iii) aresta representando o atributo active:

$$\begin{aligned} source_{NA}^{Bio}(active) &= type\_src\_amount \\ target_{NA}^{Bio}(active) &= Integer \end{aligned}$$

onde :

$$type\_src\_active = simple\_sub|genetic\_elem|complex\_sub \tag{4.14}$$

iv) aresta representando o atributo units:

$$\begin{aligned} source_{NA}^{Bio}(units) &= Legends\_Concentrations \\ target_{NA}^{Bio}(units) &= String \end{aligned} \tag{4.15}$$

v) aresta representando o atributo intervals:

$$\begin{aligned} source_{NA}^{Bio}(intervals) &= Legends\_Concentrations \\ target_{NA}^{Bio}(intervals) &= List[pair(Integer, Integer)] \end{aligned} \tag{4.16}$$

vi) aresta representando o atributo colors:

$$\begin{aligned} source_{NA}^{Bio}(color) &= Legends\_Concentrations \\ target_{NA}^{Bio}(colors) &= List[Integer] \end{aligned}$$

Definição 4.3.8 (Definição das funções $source_{EA}^{Bio}$ e $target_{EA}^{Bio}$) *Sejam:*

$$\begin{aligned} reaction = &State\_Transition, Unknown\_Transition, \\ &Translocation, Bidirectional\_Transition, \\ &Association, Dissociation, Truncation \end{aligned} \tag{4.17}$$

As funções $source_{EA}^{Bio} : E_{EA}^{Bio} \to E_G^{Bio}$ e $target_{EA}^{Bio} : E_{EA}^{Bio} \to V_D^{Bio}$ das arestas de E_G^{Bio} são definidas por:

i) aresta que representa o atributo name_react:

$$\begin{aligned} source_{EA}^{Bio}(name) &= reaction \\ target_{EA}^{Bio}(name) &= String \end{aligned} \tag{4.18}$$

ii) aresta que representa o atributo rate_law:

$$\begin{aligned} source_{EA}^{Bio}(rate\_law) &= reaction \\ target_{EA}^{Bio}(rate\_law) &= String \end{aligned} \tag{4.19}$$

iii) aresta que representa o atributo priority:

$$\begin{aligned} source_{EA}^{Bio}(priority) &= reaction \\ target_{EA}^{Bio}(priority) &= Integer \end{aligned} \tag{4.20}$$

iv) aresta que representa o atributo participants:

$$\begin{aligned} source_{EA}^{Bio}(participants) &= reaction \\ target_{EA}^{Bio}(participants) &= List[String] \end{aligned} \tag{4.21}$$

v) aresta que representa o atributo condition_trigger:

$$\begin{aligned} source_{EA}^{Bio}(condition\_trigger) &= reaction \\ target_{EA}^{Bio}(condition\_trigger) &= Boolean \end{aligned} \tag{4.22}$$

vi) aresta que representa o atributo react_parameter_tax:

$$\begin{aligned} source_{EA}^{Bio}(react\_parameter\_tax) &= reaction \\ target_{EA}^{Bio}(react\_parameter\_tax) &= List[Integer] \end{aligned} \tag{4.23}$$

vii) aresta que representa o atributo react_parameter_tax:

$$\begin{aligned} source_{EA}^{Bio}(prod\_parameter\_tax) &= reaction \\ target_{EA}^{Bio}(prod\_parameter\_tax) &= List[Integer] \end{aligned} \tag{4.24}$$

Agora, podemos definir o grafo tipo que será usado como sintaxe abstrata para os diagramas de processos.

Definição 4.3.9 (Bio-Type Graph) *O grafo tipo para representar diagramas de processos é definido por:* $G^{Bio} = (V_G^{Bio}, V_D^{Bio}, E_G^{Bio}, E_{NA}^{Bio}, E_{EA}^{Bio}, (source_j^{Bio}, target_j^{Bio})_{j\in\{G,NA,EA\}})$ *onde:*

- *Os conjuntos* V_G^{Bio}, V_D^{Bio}, E_G^{Bio}, E_{NA}^{Bio}, E_{EA}^{Bio} *são descritos nas Defs. 4.3.1, 4.3.2, 4.3.3, 4.3.4 e 4.3.5 respectivamente;*
- *as funções source e target são descritas nas definições 4.3.6 , 4.3.7 e 4.3.8 respectivamente.*

Denotamos por $Value_D^{Bio}$ a união disjunta dos conjuntos correspondentes a todos nomes de tipos de V_D^{Bio}.

Definição 4.3.10 (BioProc) *Dado o grafo de tipos* G^{Bio}*, uma instância de* G^{Bio} *é qualquer grafo* $(G, type)$ *tipado sobre* G^{Bio}*, com* $V_D^G = Value_D^{Bio}$.

Dado um BioProc $(G, type)$, para qualquer atributo *attr* de um vértice ou aresta x considerado, nós escreveremos $attr(x)$ para d tal que um vértice *attrInst* e $source_i^G(attrInst) = x$, $target_i^G(attrInst) = d$ e $type(attrInst) = attr$, com $i \in \{NA, EA\}$

Na figura 4.3 nós apresentamos uma representação compacta do grafo tipo da linguagem. O quadrado em linhas pontilhadas representa os atributos. Para ilustrar de uma forma clara como cada aresta relaciona aos tipos *source* e *target*, nós agrupamos em pequenos blocos com tipos que tem características em comum : *simple_sub*, *complex_sub*, *genetic_elem* e *logic* representado respectivamente com as formas triângulo, quadrado, círculo e diamante. Cada aresta tem tipos que podem ser associados com as funções *source* ou *target*. Se nós associamos uma forma para cada uma destas funções todos os tipos do bloco podem ser associados com esta aresta.

Para ilustrar o relacionamento entre as duas sintaxes propostas, nós mostramos um exemplo de reação que transforma a substância *lactose* em *glicose* e *galactose*, e é catalisada pela proteína *beta-galactosidase*. Cada símbolo gráfico das substâncias inserido no diagrama é uma instância de um vértice do grafo tipo ($s1$, $s2$ e $s3$ são instâncias do tipo *Simple_Molecule* já $p1$ é uma instância do tipo *Protein*). Cada substância possui um atributo associado *name* que por sua vez tem o tipo *String*. As reações são tipadas de acordo com as arestas do grafo tipo, no exemplo a reação instanciada $t1$ é do tipo *State_Transition*. A reação associa uma lista de substâncias reagentes pela função *source* e uma lista de produtos pela função *target*.

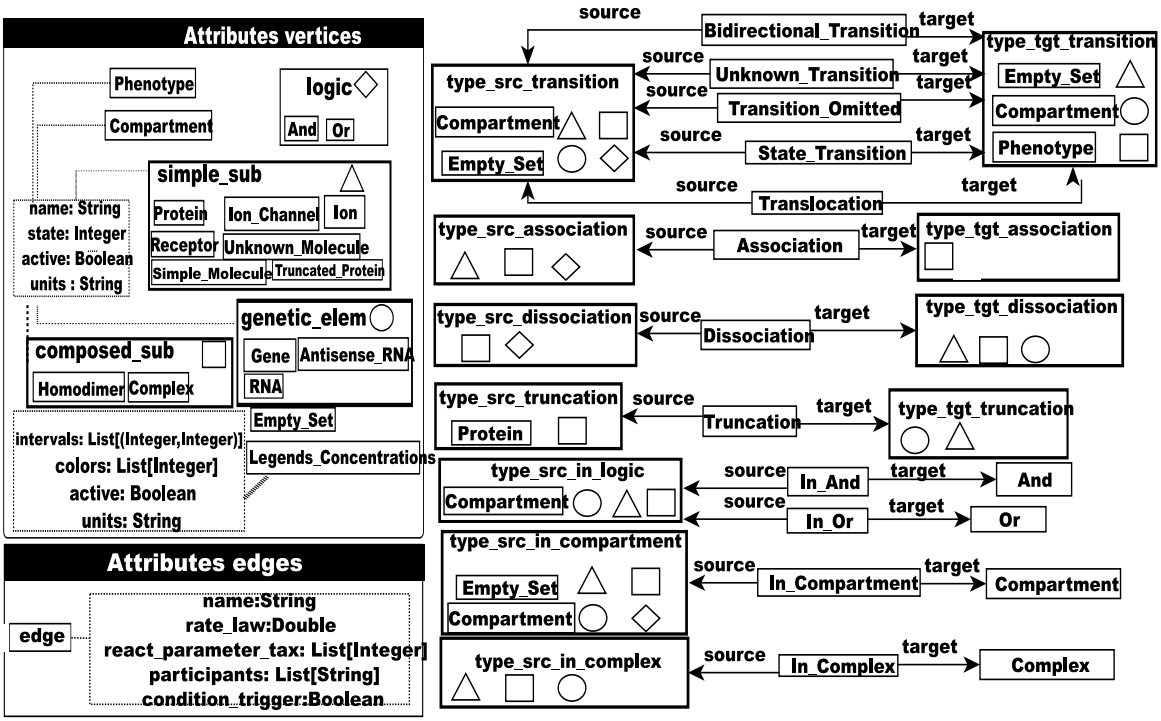


Figura 4.3: Representação dos tipos associados as arestas de transição e seus atributos no grafo tipo da linguagem.

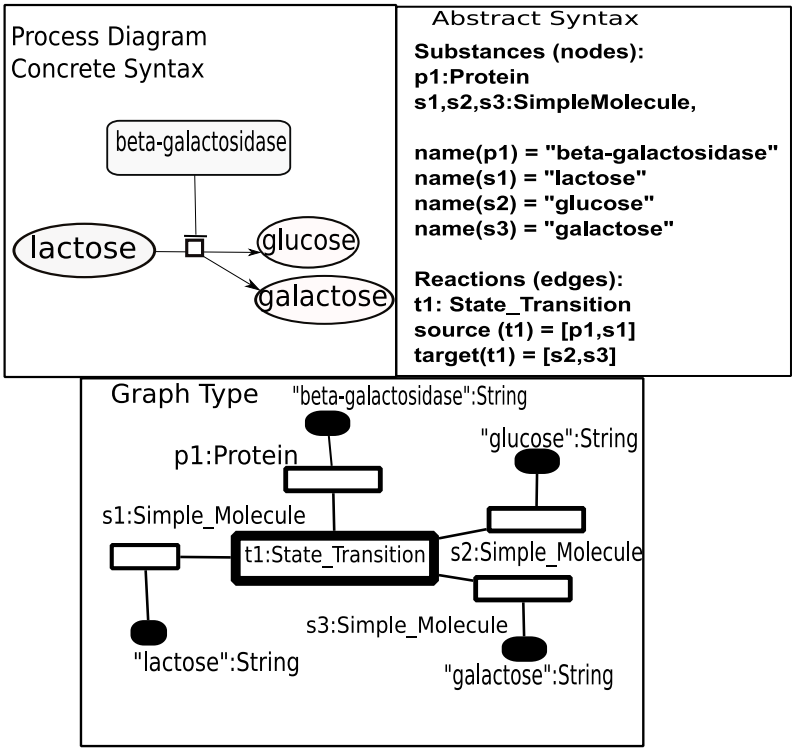


Figura 4.4: Representação da sintaxe abstrata e da sintaxe concreta de um diagrama BioProc.

4.4 Mapeamento das sintaxes concreta e abstrata dos diagramas

Após a definição dos conjuntos de tipos da linguagem estabelecemos a associação entre as sintaxes abstrata e concreta de todos os elementos do grafo BioProc. Primeiramente

listaremos a associação dos nodos com os símbolos da notação de diagrama de processos.

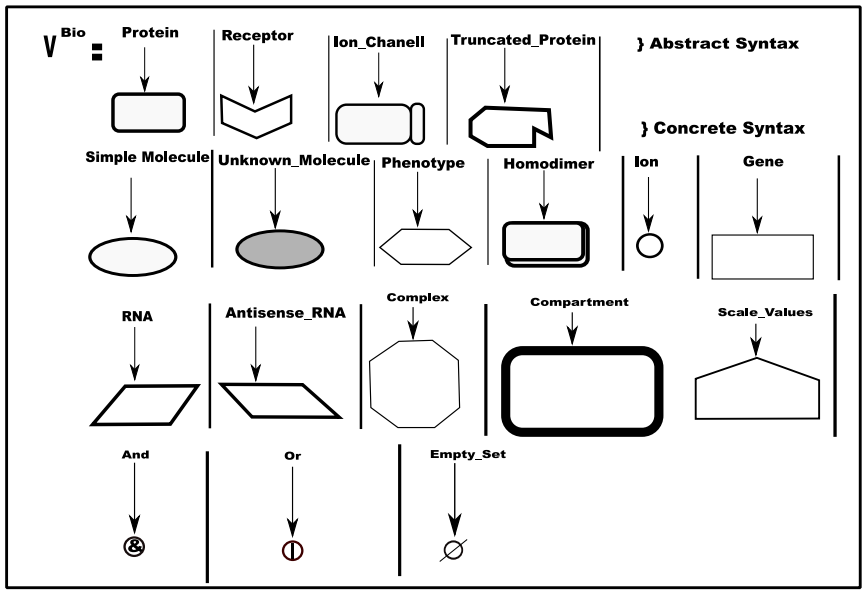


Figura 4.5: Sintaxe abstrata e concreta dos nodos.

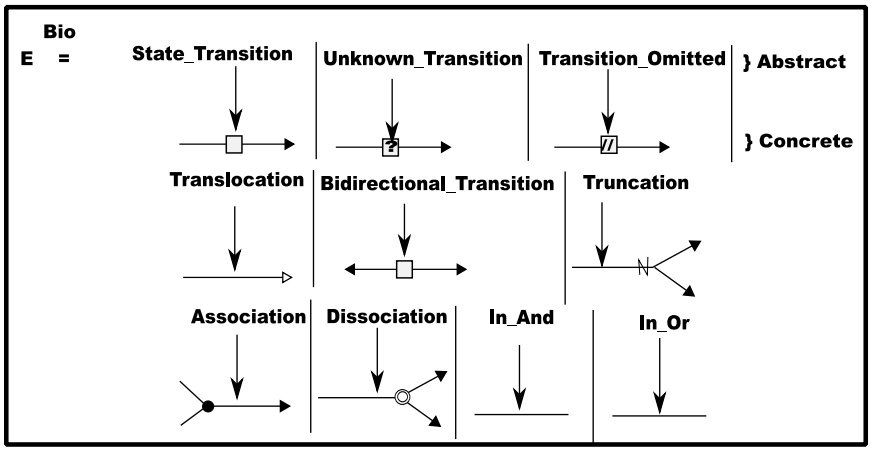


Figura 4.6: Sintaxe abstrata e concreta das arestas do grafo.

4.5 Regras que definem diagramas BioProc

Uma produção ou regra de uma gramática (de strings) é definida como $L := R$ onde L é chamado LHS (*left hand side*) e R (RHS, *right hand side*). Uma produção pode ser aplicada se há uma instância do LHS em uma string. Usando grafos a idéia permanece essencialmente a mesma, somente a noção é definida em forma de grafos ao invés de strings.

Definição 4.5.1 (Regra de uma gramática de grafos) *Uma regra* $p : (L \xrightarrow{r} R)$ *consiste de um nome de regra p e um morfismo parcial injetivo de grafo r, chamado morfismo de regra. Os grafos L e R são repectivamente (left-hand-side e right-handside) de p.*

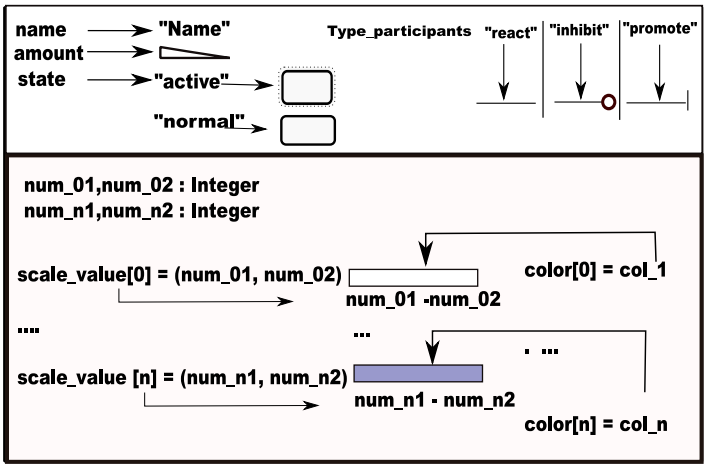


Figura 4.7: Sintaxe abstrata e concreta dos atributos do grafo.

A aplicação de uma regra a um grafo G consiste em encontrar L como subgrafo de G e substituí-lo pelo grafo R obtendo um novo grafo H. Formalmente temos a definição de derivação de um grafo.

Definição 4.5.2 (Match e derivação) *Um match para $p : L \rightarrow R$ em algum grafo G é um morfismo total $m : L \rightarrow G$. Dado a regra r e um match m para p em um grafo G, a derivação direta de G com r em m, escrito $G \overset{p,m}{\Longrightarrow} H$, é o pushout de p e m.*

Conforme (ROZENBERG, 1997) intuitivamente um *pushout* é uma união generalizada. Em transformações de grafos uma operação de *pushout* pode ser exemplificada na figura 4.8. Dada uma regra de uma gramática formada pelos grafos L e R e um grafo G a ser transformado pela aplicação da regra, encontramos L como subgrafo (*match*) do grafo G substituindo este subgrafo pela estrutura do grafo R em G, que gerará o novo grafo H.

Mostraremos agora no exemplo da figura 4.9 como foi construído o diagrama usando as regras de construção da gramática de grafos. Neste exemplo inicia-se com um diagrama vazio e aplica-se as seguintes regras:

1) Insert_simple_molecule - uma molécula simples s1 é inserida no diagrama com seus atributos (*name ,state,amount*);

2) Insert_simple_molecule - uma molécula simples s2 é inserida no diagrama com seus atributos (*name ,state,amount*);

3) Insert_State_Transition - uma reação de transição é inserida no diagrama com os atributos da aresta associando s1 como *source* e s2 como *target*;

4) Insert_protein - uma proteína é inserida no diagrama no diagrama com seus atributos (*name ,state,amount*);

5) Add_react_ST - associo a proteína como promotora da reação;

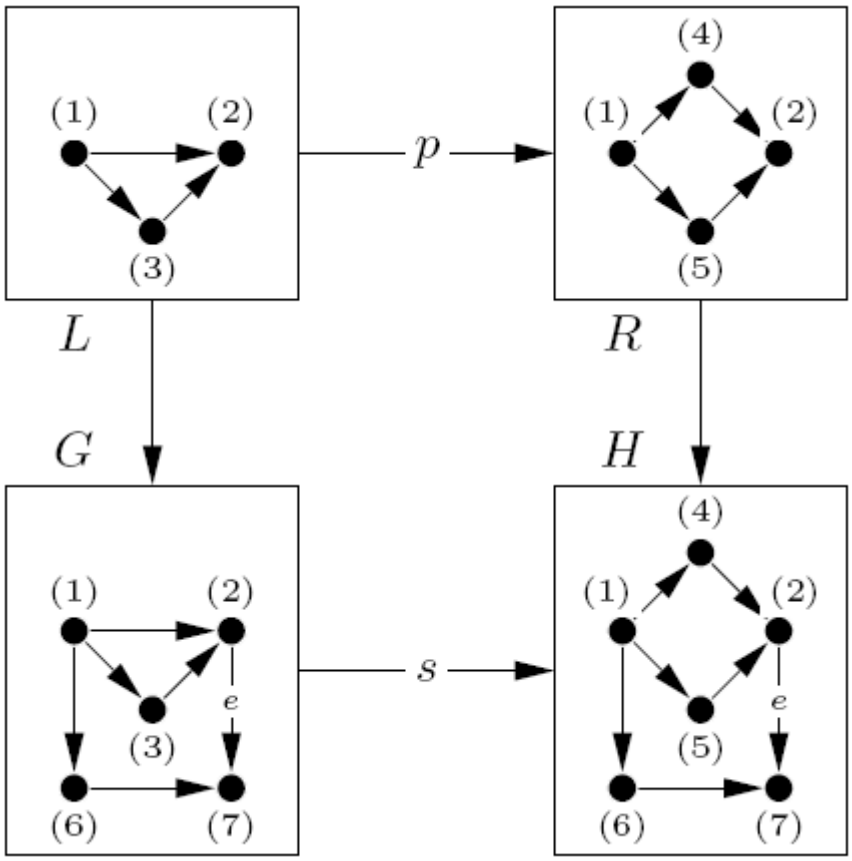


Figura 4.8: Derivação de grafo SPO.

4.6 Considerações Finais

Neste capítulo apresentamos o metamodelo formal de construção dos diagramas de processos definindo suas sintaxes: concreta e abstrata. Além disso foi apresentada a interação passo a passo da construção de um diagrama sintaticamente correto utilizando regras da gramática de grafos proposta.

Modelar vias implica não só na construção e armazenamento das vias em bancos de dados biológicos. É necessário definir formalmente a semântica da simulação destes diagramas. Apresentaremos no próximo capítulo uma tradução formal do metamodelo de especificação em Redes de Petri Estocásticas Generalizadas. Com esta tradução poderemos modelar o comportamento dos diagramas de processos e verificar propriedades já definidas para o modelo e suas interpretações no contexto biológico.

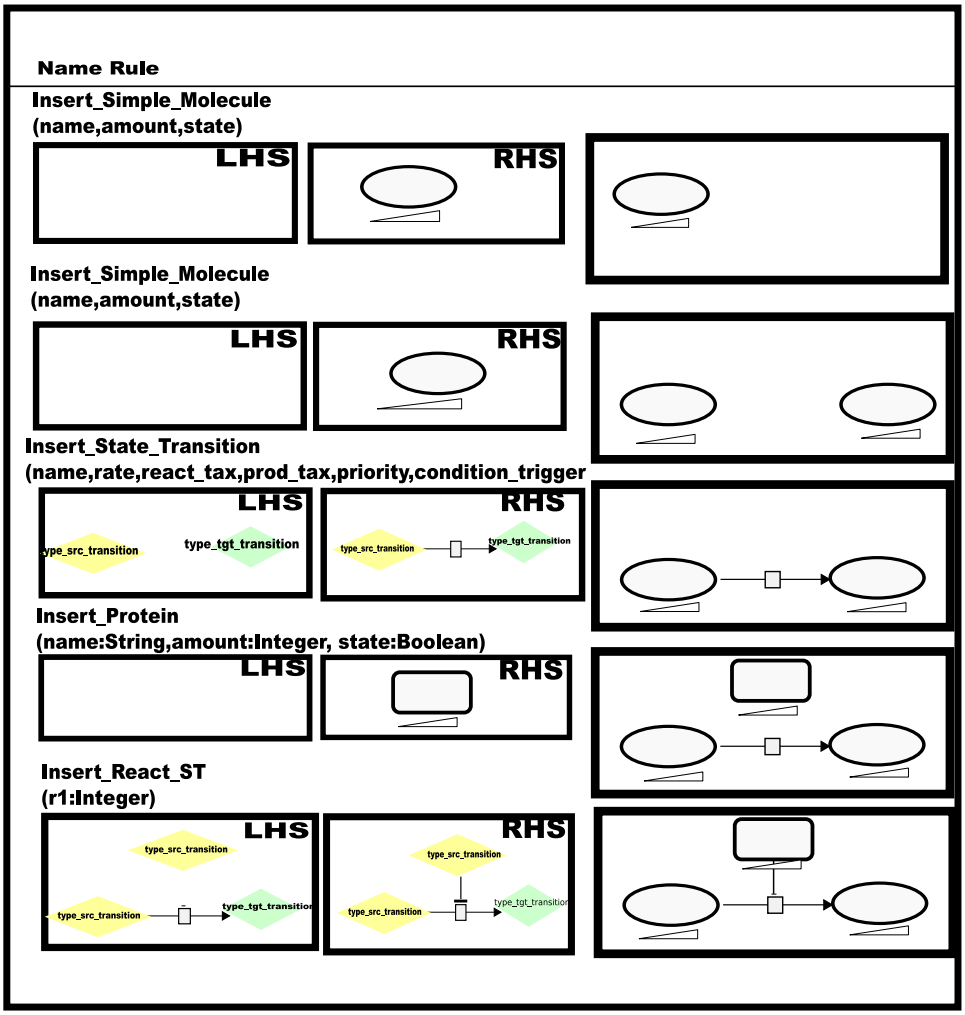


Figura 4.9: Geração de um diagrama exemplo a apartir de regras.

5 SEMÂNTICA DOS DIAGRAMAS BIOPROC

No capítulo anterior foi apresentada a formalização de uma linguagem para especificação de vias biológicas. Neste capítulo apresentaremos uma tradução formal da linguagem descrita anteriormente para um método formal que possibilite a simulação do comportamento das vias biológicas. Isto compreende o estudo da variação das quantidades de reagentes que interagem na via bem como os produtos resultantes. Além disso verificar propriedades que possam simular comportamentos experimentalmente verificáveis do sistema.

Vários aspectos de vias biológicas necessários à simulação já foram amplamente estudados em sistemas concorrentes computacionais. Redes de Petri é um um método formal de verificação de sistemas reativos capaz de caracterizar aspectos dinâmicos como concorrência, sincronização, exclusão mútua, e conflito (KARTSON et al., 1994). Neste capítulo será apresentada a semântica para os diagramas Bioproc. Para isto definimos uma tradução formal destes diagramas para Redes de Petri Estocásticas Generalizadas (GSPNs) .

5.1 Generalized Stochastic Petri Nets

As redes de Petri (PN) originalmente formulada na tese de doutorado de Carl Petri (PETRI, 1962), tiveram várias extensões e adaptações de forma a integrar novas características ao modelo, como tempo e propriedades estocásticas.

Uma PN básica é formada por lugares, transições, arcos e tokens:

- Lugares são usados para descrever possíveis estados de sistemas;
- Transições são usados para descrever eventos que podem modificar o estado do sistema;
- Arcos especificam relações entre os estados e as transições;
- *Tokens* são marcadores presentes nos lugares, e são usados para especificar o estado da PN (marcação). Se um lugar é usado para descrever uma condição, ele pode conter zero ou um *token*, a condição é verdadeira se há um *token* presente e falso contrário. Se um lugar define uma situação, o número de tokens contido no lugar é usado para determinar o estado daquele lugar;

Graficamente um modelo PN é representado por um grafo bipartido direcionado no qual os lugares são representados por círculos e as transições são barras retangulares. Tokens são representados pontos pretos dentro de lugares ou por simplificação visual um número inteiro que representa quantidade de *tokens*.

Generalized Stochastic Petri Nets (GSPNs) estendem a definição básica em alguns aspectos:

- Inclusão de propriedades estocásticas com a definição de dois tipos de transição:
  - *Timed transitions* - são transições onde, após sua habilitação, o disparo ocorre após um determinado intervalo de tempo (determinado por uma taxa). Sua representação gráfica é um retângulo não preenchido;
  - *Imediate transitions* - são transições onde, após sua habilitação, o disparo ocorre imediatamente. Sua representação gráfica é um retângulo preto;
- Definição de prioridades - para resolver o problema de conflito de transições habilitadas para o disparo em GSPN ao mesmo tempo, pode ser usada uma função que determina qual transição possui maior probabilidade de ocorrência;

Apresentada a visão geral das GSPNs apresentamos a definição formal:

Definição 5.1.1 (GSPN) *Uma rede de Petri estocástica generalizada é uma 8-tupla :* $M_{GSPN} = (P, T, \Pi, I, O, G, M_0, W)$ *where:*

- P *é um conjunto de lugares;*
- T *é um conjunto de transições, onde* $T \cap P = \emptyset$*;*
- $\Pi : T \to \mathbb{N}$ *é a função de prioridade que mapeia uma transição imediata a um número natural que representa a prioridade da mesma;*
- $I, O : T \to Bag(P)$ *são as funções de entrada, saída das transições respectivamente, onde* $Bag(P)$ *é o multiconjunto P. Formalmente multiconjuntos são conjuntos com elementos repetidos e podem ser presentados por listas de pares, onde cada par é representado por um elemento do conjunto e o número de ocorrências. Nas GSPNs o par é composto pelo lugar e o número de tokens do arco necessário para a ocorrência da transição.*
- $G : T \to \mathbb{C}$ *função guard que associa uma condição necessária mas não suficiente para o disparo de uma transição* $t \in T$*, onde* $\mathbb{C}$ *é o conjunto de expressões booleanas que podem ser associadas as transições da GSPN;*
- $M_0 : P \to \mathbb{N}$ *é a marcação inicial dos lugares;*
- $W : T \to \mathbb{R}$ *é a função que mapeia uma transição a uma função real positiva com imagem no conjunto dos reais;*

Um exemplo de construção de GSPN é exposto na figura 5.1. Onde:

- $P = \{P0, P1, P2\}$ representa o conjunto de lugares;
- $T = \{T0\}$ representa o conjunto de transições;

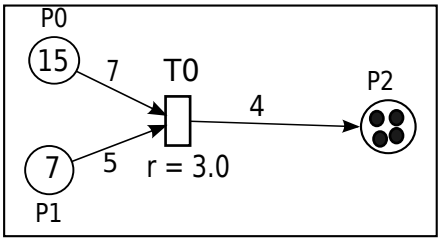


Figura 5.1: Exemplo de GSPN.

- $I(T0) = \langle (P0,7),(P1,5) \rangle$ e $O(T0) = \langle (P2,4) \rangle$ representam respectivamente as funções de entrada e saída da transição $T0$ que a associam aos multiconjuntos representado por listas de pares, onde o primeiro elemento do par é o lugar e o segundo representa o número de tokens consumidos (função de entrada) ou produzidos (função de saída). Na figura exemplo 5.1 as ocorrências são representadas com os números junto aos arcos;

- $G(T0) = T$ é associada o valor *true* a transição;

- $M_0(P0) = 15$, $M_0(P1) = 7$ e $M_0(P2) = 4$;

- $W(T0) = 3.0$ representa a taxa associada a transição $T0$.

A dinâmica de uma PN é regida pela regra do disparo de transições. Um disparo consiste na retirada de um ou mais tokens dos lugares origem dos arcos de entrada e gera um ou mais *tokens* nos lugares de saída. A remoção e a produção dos *tokens* durante o disparo é uma operação atômica.

Na definição 5.1.2 temos a tradução dos diagramas BioProc para GSPNs.

Definição 5.1.2 (Semântica dos diagramas BioProc) *Dado um grafo* $G^{Bio} = (G, type)$, *onde* $G = (V^G, V_D^G, E^G, E_{NA}^G, E_{EA}^G, (source_j^G, target_j^G)_{j \in \{G,NA,EA\}})$ *e* $type : G \to Bio$, *a semântica de* G^{Bio} *é definida pela GSPN* $P = (S, T, \Pi, I, O, H, M_0, W)$ *onde:*

- O conjunto de vértices usados na representação de diagramas são mapeados em conjunto de lugares da GSPN, excluem-se os tipos de vértices que não possuem representação de concentração.

$$S = V^{Bio} - \{And, Or\}$$

- O conjunto de arestas que representam reações são mapeadas no conjunto de transições da GSPN, excluem-se os tipos dos quais não possuem atributos relativos a taxas das reações, representado pelo subconjunto formado pelos tipos {In_Compartment,In_Complex,In_Logical}.

$$T = E^{Bio} - \{In\_Compartment, In\_Complex, In\_Logical\}$$

- Os arcos de entrada de uma transição são os tentáculos da aresta do hipergrafo que possuem atributos relativos a taxas das reações. Um arco de entrada numa GSPN é um *multiset* do conjunto S de lugares. Podemos representar normalmente um *multiset* por uma lista de pares onde o primeiro elemento de cada par da lista é um elemento do *multiset* e o segundo elemento do par é o número de repetições do

elemento:
Seja $\forall\, t \in T$

$$I(t) = [(s_i, t_i), ...(s_n, t_n)] \tag{5.1}$$

onde

- $s_i \in source^{Bio}(t)$;
- $t_i \in reaction\_parameter\_tax(t)$;
- $participants(i) = "normal"$;

- Os arcos de saída de uma transição são os tentáculos da aresta do hipergrafo que possuem atributos relativos a taxas das reações. Um arco de saída numa GSPN também é representado por um *multiset* do conjunto S de lugares.
  Seja $\forall\, t \in T$

$$O = [(k_i, p_i), ...(k_n, p_n)] \tag{5.2}$$

onde

- $k_i \in target^{Bio}(t)$;
- $p_i \in prod\_parameter\_tax(t)$;

- $M_0 = \forall s \in S \quad [M_0(s) = amount(s)]$;
- $W = \forall t \in t \quad [W(t) = rate\_law(t)]$;

Para demonstrar os aspectos dinâmicos, nós propomos uma extensão à notação dos diagramas de processos a representação do número de tokens nas GSPNs. O programador deve associar a um diagrama escala de cores que representem as variações de concentrações (poderia ser concentrações após a simulação ou simplesmente intervalos de valores absolutos). Durante a simulação cada lugar da rede de Petri pode aumentar ou reduzir seu número de tokens.

Definição 5.1.3 (Evolução de um diagrama BioProc) *Dados um BioProc G^{Bio} e sua semantica P. Se P evolui para a rede P' através do disparo de uma ou varias transições, então dizemos que o diagrama G^{Bio} evolui para o diagrama H^{Bio} definido pelo mesma estrutura de vértices $V_{G^{Bio}}$ e arestas de s $E_{G^{Bio}}$ com os valores atualizados do atributo amount $\forall v \in V_{G^{Bio}}$.*

5.2 Exemplos

Exemplo 5.2.1 Neste exemplo demonstramos o uso da linguagem visual, e como a evolução da GSPN gerada também altera o estado do diagrama BioProc. A reação exemplificada é uma interação proteína-proteína (Figura 5.2). As cores (tons de cinza) das substâncias foram determinadas com base na escala de valores definidas pelo usuário.

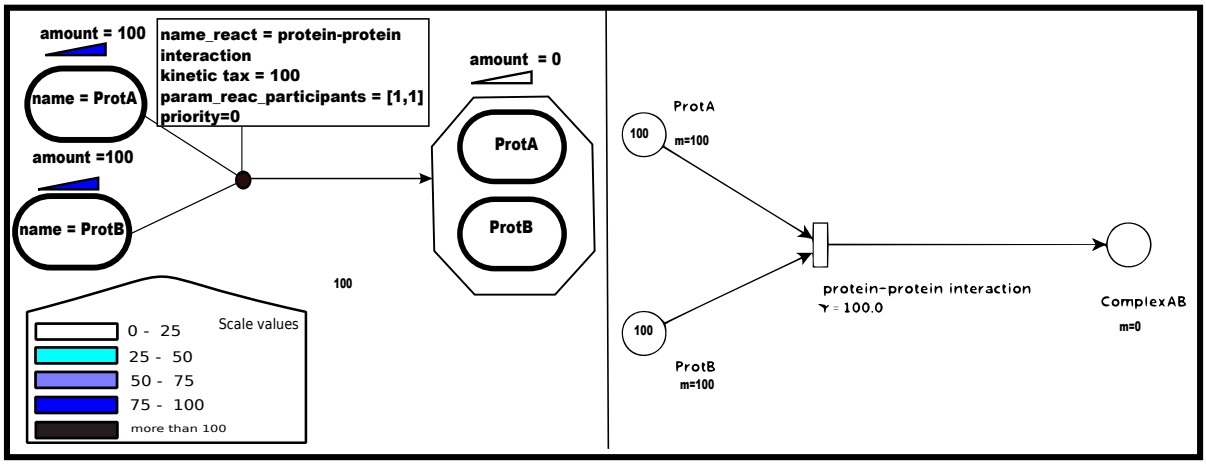


Figura 5.2: Exemplo de interação proteína-proteína antes da simulação.

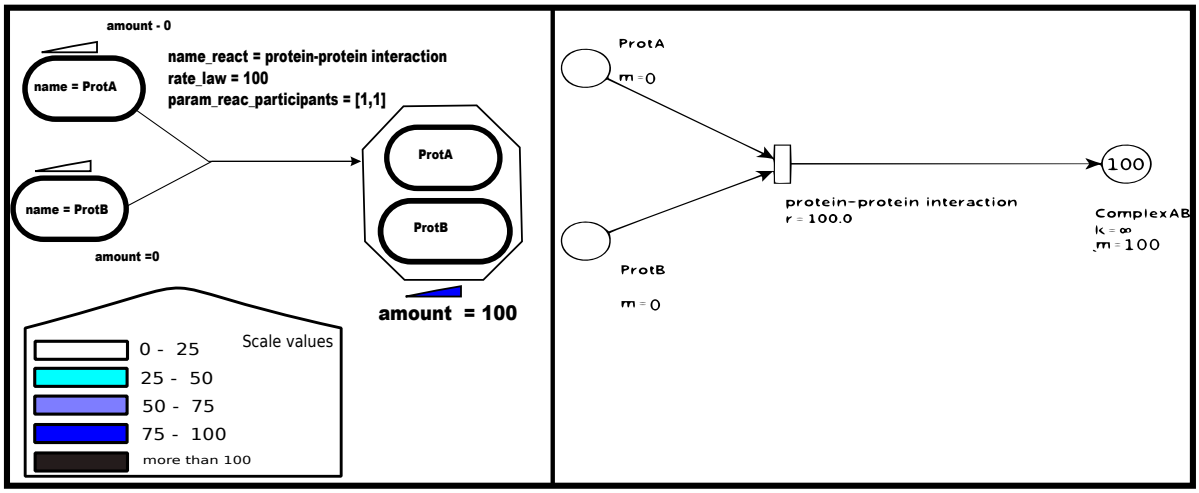


Figura 5.3: Exemplo de interação proteína-proteína após a simulação.

Exemplo 5.2.2 O próximo exemplo é mais elaborado. Esta via biológica é conhecida como *Extracellular signal regulated kinase*(ERK) os dados relativos às quantidades iniciais dos reagentes e as taxas das reações foram extraídos de (CHO et al., 2003). Esta via é responsável pela proliferação e diferenciação de diferentes tipos de célula. Ela é composta de dois tipos de de reações: fosforilações (uma transição na qual uma substância adquire um íon de fósforo) e interações(associações e dissociações) proteína-proteína. A seguir estão enumeradas as etapas de reações.

1. *RKIP* (uma proteína repressora) associa-se com a proteína *Raf-1* kinase* gerando complex (*Raf1*/RKIP*).

2. O complexo *(Raf1*/RKIP)* pode ligar com *ERK-PP* e formar o complexo *Raf-1*/RKIP/ERK-PP*.

3. Neste estágio a proteína *Raf-1** abandona o complexo quando o *ERK-PP* fosforila *RKIP* em *RKIP-P*.

4. Após esta etapa há novamente uma associação entre *RKIP-P* e *RP* formando o complexo *(RKIP-P/RP)*.

5. *RKIP-P* então retorna para o estado original e associa-se novamente com Raf-1* passa atuar como inibidor da fosforilação e ativação de *MEK*.

6. *ERK* livre asocia com MEK-PP integrando outro complexo. *(MEK-PP/ERK)*. Este complexo dissocia novamente gerando *MEK-PP* e promove a fosforilação do *ERK* em *ERK-PP*.

Com base nestas descrições definimos um digrama BioProc mostrado na figura 5.4. Usamos nossa tradução dos diagramas para GSPN e simulamos a rede Petri no software Pipe (BONET et al., 2007). Realizamos a simulação com 200 disparos na rede de Petri num intervalo de 50 milisegundos. Após a simulação obtivemos as variações de concentrações apresentadas nas figuras 5.4 e 5.5.

5.3 Análise Estrutural de uma GSPN

Propriedades importantes de GSPNs podem ser definidas e checadas com ferramentas de análise (KARTSON et al., 1994). Tais propriedades podem ser importantes nos estudos de sistemas biológicos. Em (CHAOUIYA, 2007) e (ZEVEDEI-OANCEA; SCHUSTER, 2003) foram definidas interpretações das propriedades das redes de Petri no contexto de sistemas biológicos. Os programas construídos usando o metamodelo BioProc foram verificados quanto a 3 propriedades :

- Atingibilidade e Reversibilidade.

- Análise Invariante.

- Minimal Traps e siphons.

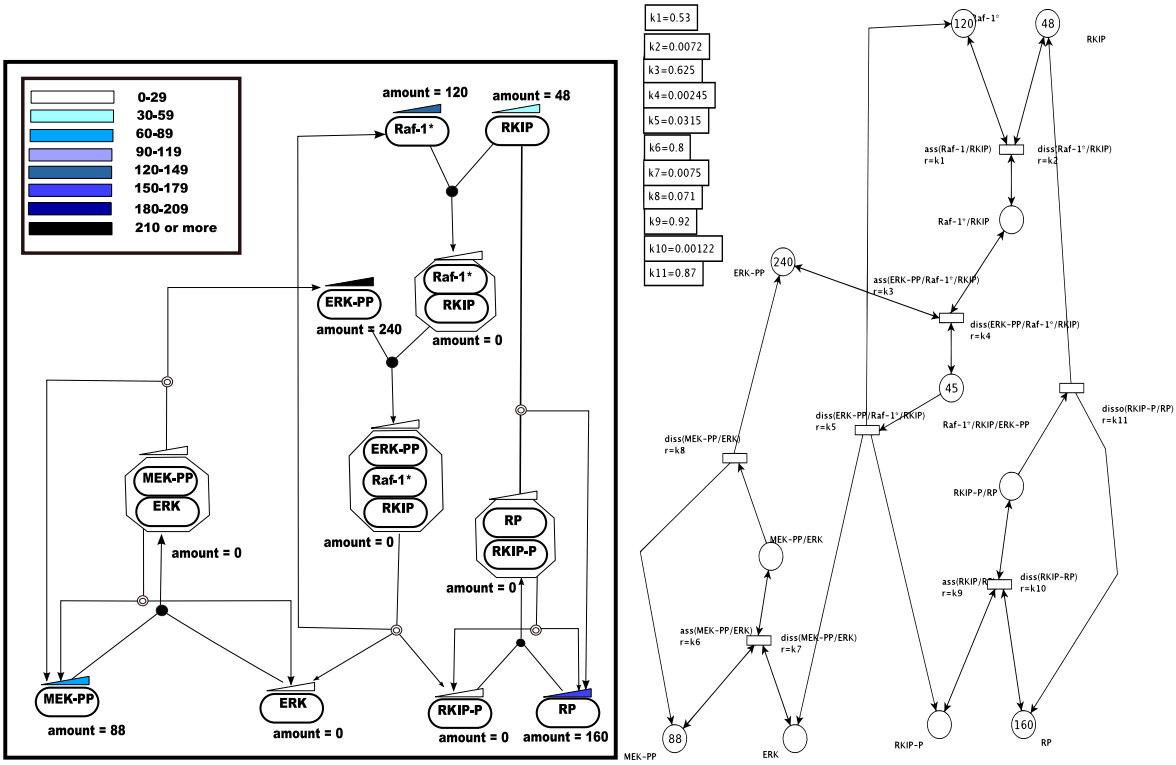


Figura 5.4: Pathway RKIP com a marcação inicial.

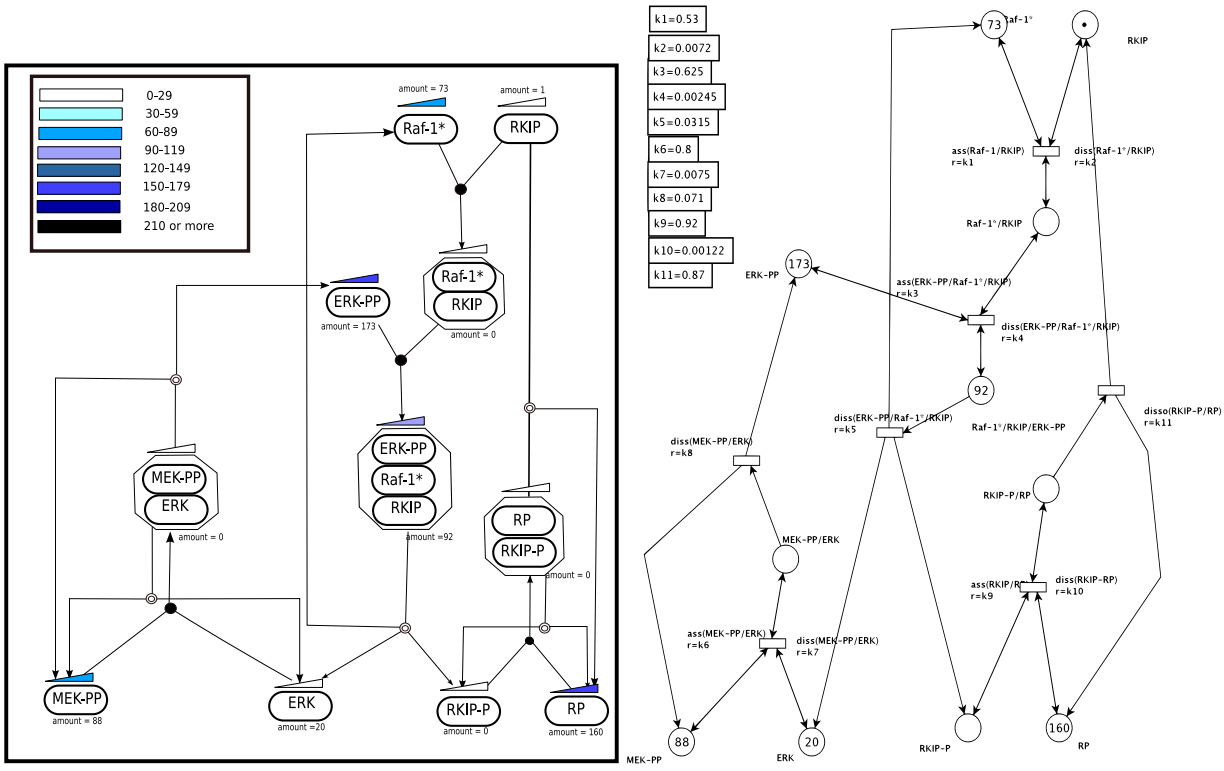


Figura 5.5: Pathway RKIP com a marcação inicial.

5.3.1 Atingibilidade e Reversibilidade

Atingibilidade e reversibilidade são propriedades que determinam se uma marcação específica M_n é atingível a partir de outra configuração M_i. Isto significa que existe uma

sequência de transições σ_M onde M_i atinge M_n ($M_i[\sigma_M\rangle M_n$). Na via de estudo definimos uma marcação unária para alguns lugares para a geração de um pequeno grafo atingível e compreensível da via do exemplo 5.2.2.

A seguinte sequência $M0$ representa respectivamente a marcação inicial das seguintes substâncias: ⟨ Raf-1*/RKIP,Raf-1*,MEK-PP,RKIP,Raf-1*/RKIP/ERK-PP,ERK-PP,ERK,RKIP-P, RKIP-P/RP, RP, MEK-PP/ERK ⟩.

$$M0 = \langle 0,1,1,1,0,1,0,1,0,1,0 \rangle$$

Todos estados são atingíveis da via com esta configuração mostrada na figura 5.7.

	Raf-1*/RKIP	Raf-1*	MEK-PP	RKIP	Raf-1*/RKIP/ERK-PP	ERK-PP	ERK	RKIP-P	RKIP-P/RP	RP	MEK-PP/ERK
M0	0	1	1	1	0	1	0	1	0	1	0
M1	0	1	1	1	0	1	0	0	1	0	0
M2	1	0	1	0	0	1	0	1	0	1	0
M3	0	1	1	2	0	1	0	0	0	1	0
M4	1	0	1	0	0	1	0	0	1	0	0
M5	0	0	1	0	1	0	0	1	0	1	0
M6	1	0	1	1	0	1	0	0	0	1	0
M7	0	0	1	0	1	0	0	0	1	0	0
M8	0	1	1	0	0	0	1	2	0	1	0
M9	0	0	1	1	1	0	0	0	0	1	0
M10	0	1	1	0	0	0	1	1	1	0	0
M11	0	1	0	0	0	0	0	2	0	1	1
M12	0	1	1	1	0	0	1	1	0	1	0
M13	0	1	0	0	0	0	0	1	1	0	1
M14	0	1	1	0	0	1	0	2	0	1	0
M15	0	1	1	1	0	0	1	0	1	0	0
M16	0	1	0	1	0	0	0	1	0	1	1
M17	1	0	1	0	0	0	1	1	0	1	0
M18	0	1	1	0	0	1	0	1	1	0	0
M19	0	1	1	2	0	0	1	0	0	1	0
M20	0	1	0	1	0	0	0	0	1	0	1
M21	1	0	1	0	0	0	1	0	1	0	0
M22	1	0	0	0	0	0	0	1	0	1	1
M23	0	1	0	2	0	0	0	0	0	1	1
M24	1	0	1	1	0	0	1	0	0	1	0
M25	1	0	0	0	0	0	0	0	1	0	1
M26	1	0	0	1	0	0	0	0	0	1	1

Figura 5.6: Conjunto dos estados atingíveis.

Do ponto de vista biológico estas propriedades podem ser usadas para verificar que substâncias são necessárias para promover uma produção de um determinado elemento e as transições necessárias. Com estas informações podemos manipular o fluxo de processamento da via biológica sabendo que concentrações de substâncias devem ser controladas para que adquiram um determinado comportamento.

5.3.2 P-invariants e T-invariants

P-invariants e T-invariants são também propriedades verificadas em GSPNs. P-invariants representam relações de lugares onde o número total de tokens independe da sequência de disparos de transições e qualquer escolha para marcação inicial. Se todos os lugares são participantes de pelo menos um conjunto P-invariant, o número máximo de tokens da rede é considerado limitado (KARTSON et al., 1994). Em uma via biológica representa que a via é fechada e a quantidade de participantes sempre será finita e controlada. No exemplo mostrado de acordo com as propriedades de *P- invariants* verifica-se que o sistema é limitado, pois todas as entidades deste exemplo fazem parte de pelo menos uma *P-invariant*.

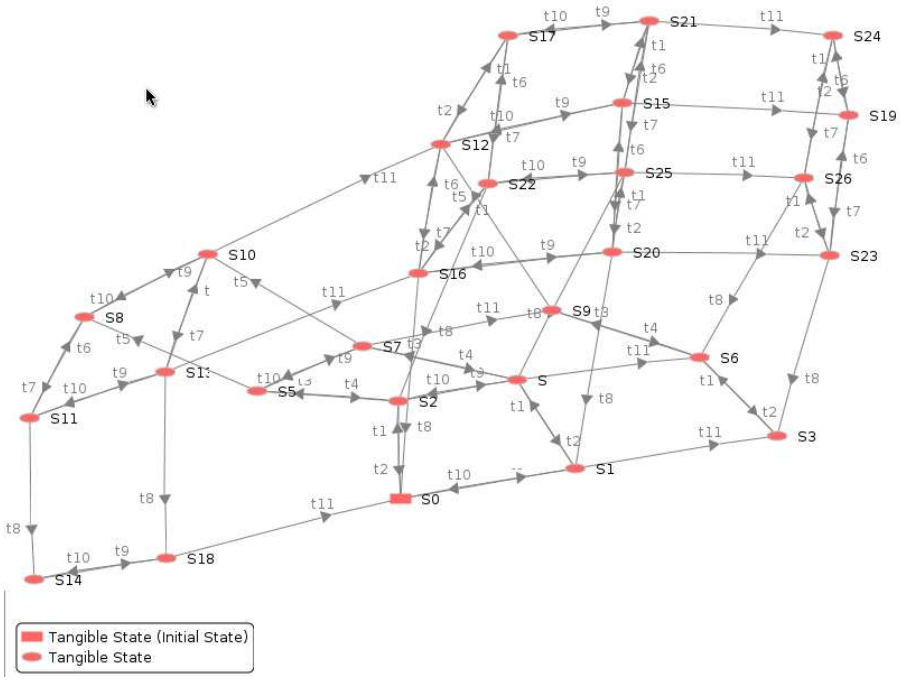


Figura 5.7: Grafo de atingibilidade.

1. M(Raf-1*/RKIP) + M(Raf-1*) + M(Raf-1*/RKIP/ERK-PP) = 1

2. M(MEK-PP) + M(MEK-PP/ERK) = 1

3. M(Raf-1*/RKIP/ERK-PP) + M(ERK-PP) + M(ERK) + M(MEK-PP/ERK) = 1

4. M(Raf-1*/RKIP) + M(RKIP) + M(Raf-1*/RKIP/ERK-PP) + M(RKIP-P) + M(RKIP-P/RP) = 2

5. M(RKIP-P/RP) + M(RP) = 1

T-invariants são relações onde dados uma marcação inicial M e uma sequência de transição σ, a marcação obtida ao final da sequência de transição M' é igual a inicial (KARTSON et al., 1994). Na tabela 5.1 são mostradas as T-invariants do exemplo, dadas as transições rotuladas de $t0$ a $t11$, temos 5 T-invariants. Na T-Invariant da linha 3, por exemplo, temos que o disparo do conjunto de transições formado por $\langle t1, t3, t4, t6, t8, t9, 10\rangle$ resultará na mesma marcação obtida antes da sequência. Esta idéia segundo (CHAOUIYA, 2007) pode ser usada para determinar comportamentos cíclicos das vias ao verificar quais reações promovem a manutenção de uma determinada configuração de concentrações de substâncias.

5.3.3 *Siphons* e *Traps* de uma GSPN

Há técnicas de análise estrutural em redes de Petri possibilitam a checagem de possíveis *deadlocks* (estados nos quais nenhuma transição pode ser disparada) verificando a existência de *siphons* e *traps*. Intuitivamente temos:

t0	ass(Raf-1*/RKIP)
t1	diss(Raf-1*/RKIP)
t2	ass(ERK-PP/(Raf-1*/RKIP))
t3	disso(ERK-PP/(Raf-1*/RKIP))
t4	disso(RKIP-P/RP)
t5	ass(RKIP/RP)
t6	disso(RKIP/RP)
t7	disso(ERK-PP/Raf-1*/RKIP)
t8	disso(ERK-PP/Raf-1*/RKIP)
t9	ass(MEK-PP/ERK)
t10	disso(MEK-PP/ERK)
t11	disso(MEK-PP/ERK-PP)

	t1	t2	t3	t4	t5	t6	t7	t8	t9	t10	t11
1	1	1	0	0	0	0	0	0	0	0	0
2	0	0	0	1	1	0	0	0	0	0	0
3	1	0	1	1	0	1	0	1	1	1	0
4	0	0	0	0	0	0	1	1	0	0	0
5	0	0	1	0	0	0	0	0	0	0	1

Tabela 5.1: T-invariants detectados no pathway exemplo.

- Um *siphon* é um conjunto de lugares onde o conjunto de transições de entrada está contido no conjunto de transições de saída. Uma importante propriedade de *siphons* é que estes conjuntos uma vez desmarcados (ou seja sem *tokens*) assim permanecem sem que estes conjuntos se reativem.

- Um *trap* é um conjunto de lugares onde o conjunto de transições de saída está contido no conjunto de transições de entrada. Uma importante proriedade das traps é que estes conjuntos uma vez marcados nunca perderão completamente seus *tokens*;

Com isso podemos verificar se a rede possui *deadlocks* a partir da seguinte propriedade: se todos *siphons* de uma rede contém uma *trap* e todas as traps são suficientemente marcadas, garantimos que o estado de deadlock não será atingível.

No exemplo foram detectadas 5 *siphons* e 6 *traps* listadas nas tabelas 5.2 e 5.3. Como pode ser analisado na tabela todos os *siphons* contém pelo menos um *trap* que com a marcação inicial do exemplo garante que a não ocorrência de um *deadlock* completo.

	Siphons
1	RKIP-P/RP, RP
2	Raf-1*/RKIP, Raf-1*, Raf-1*/RKIP/ERK-PP
3	Raf-1*/RKIP/ERK-PP, ERK-PP, ERK, MEK-PP/ERK
4	MEK-PP, MEK-PP/ERK
5	Raf-1*/RKIP, RKIP, Raf-1*/RKIP/ERK-PP, RKIP-P, RKIP-P/RP

Tabela 5.2: Siphons detectados na via RKIP

Em (ZEVEDEI-OANCEA; SCHUSTER, 2003) é descrito o teste de *deadlock-freeness* como uma ferramenta para verificar quando um sistema atingiu um estado bloqueado podendo configurar uma situação já descrita na química de falso equilíbrio. Um falso equilíbrio seria uma situação onde o sistema não se apresenta reativo devido a uma configuração de reagentes e produtos das reações que não habilitem nenhum tipo de reação.

	Traps
1	RKIP-P/RP, RP
2	Raf-1*/RKIP/ERK-PP, ERK-PP, ERK, MEK-PP/ERK
3	Raf-1*/RKIP, Raf-1*, Raf-1*/RKIP/ERK-PP
4	Raf-1*/RKIP, Raf-1*, Raf-1*/RKIP/ERK-PP, ERK-PP
5	MEK-PP, MEK-PP/ERK
6	Raf-1*/RKIP, RKIP, Raf-1*/RKIP/ERK-PP, RKIP-P, RKIP-P/RP

Tabela 5.3: Traps detectados na via RKIP

5.4 Análise Probabilística com CSL

Nesta seção será apresentada a sintaxe da lógica estocástica contínua (CSL) e como utilizá-la em análises de modelos isomórficos a cadeias de Markov (caso das GSPNs) na formulação e verificação de propriedades destes modelos.

5.4.1 CSL

Na seção anterior foram apresentadas propriedades estruturais, que são checáveis sem a construção do espaço de estados (grafo de atingibilidade) (CEROTTI et al., 2006). Como vimos antes tais características são importantes em predição de comportamento de sistemas biológicos, entretanto os estudos que podem ser feitos com estas propriedades são limitados já que não é verificado o comportamento dinâmico e evolutivo do sistema.

Em (AZIZ et al., 2000) foi proposta uma lógica estocástica para análise de cadeias de Markov de tempo contínuas (CTMC) chamada CSL (*continuous stochastic logic*). Lógica estocástica contínua é uma extensão da lógica temporal CTL, para especificar propriedades a serem verificadas em performance de sistemas

Basicamente a sintaxe CSL é definida como segue conforme (HECKEL; LAJIOS; MENGE, 2006). Supondo que uma função de rotulação $L : S \rightarrow 2^{AP}$ é dada associando todo estado s do conjunto de proposições atômicas $L(s) \subseteq AP$ que são válidas em s.

Definição 5.4.1 (Sintaxe CSL) *é definida por:*

$$\phi = true|a|\neg\phi|\varphi_1 \wedge \varphi_2|S_{\lhd p}(\varphi)|P_{\lhd p}\varphi_1 \mathscr{U}^I \varphi_2) \tag{5.3}$$

Onde $\lhd \in \{<, >, \leq, \geq\}$ $p \in [0,1]$. $a \in AP$ e $I \subseteq R$ é um intervalo não vazio. Os outros conectivos são definidos como segue $false = \neg true$, $\phi \vee \psi = \neg(\neg\phi \wedge \neg\psi)$. O operador S denota se uma fórmula ϕ é verdadeira no limite definido pela probabilidade p num estado estacionário. Já o operador P denota uma fórmula ϕ é verdadeira num intervalo de estados (*path*) $\varphi_1 \mathscr{U}^I \varphi_2$ é verdadeiro no limite definido pela probabilidade p.

Com a posterior tradução de nossa via exemplo em cadeias de Markov podemos usá-la como entrada em softwares como PRISM (KWIATKOWSKA; NORMAN; PARKER, 2002) e realizar a verificação de propriedades 5.4 e 5.5 :

$$P_{>0.25}(RKIP/RAF - 1^* = 0 \quad \mathscr{U}^{[T,T']} \quad (RKIP = 0 \wedge RKIP/RAF - 1* = 0) \tag{5.4}$$

$$P_{>0.9}[(Raf1^*\_RKIP\_ERKPP < 100)U(Raf1^*\_RKIP < 20)] \tag{5.5}$$

Estas expressões em CSL formalizam as seguintes propriedades :

- A probabilidade que a proteína RKIP degrade no intervalo de tempo $[T, T']$ e não associe a proteína RAF*1 logo após este processo é maior que 25 %. Esta propriedade foi falsa no intervalo [1,10] conforme simulação feita no PRISM.

- A probabilidade que o complexo $Raf1^*\_RKIP\_ERKPP$ atinja um valor menor que 100 até o momento que a proteína $Raf1^*\_RKIP$ atinja uma concentração menor que 20 é maior que 95 %. Esta propriedade é verdadeira conforme experimento realizado no PRISM.

5.5 Considerações Finais

Este capítulo apresentou uma tradução formal dos diagramas de processos para Redes de Petri Estocásticas Generalizadas. Foi apresentado um estado de caso com dois exemplos: no primeiro foi modelado um sistema biológico simples, no segundo foi modelada uma via biológica real já amplamente estudada. Neste último foi feito o estudo de verificação de propriedades estruturais das Redes de Petri e a interpretação de seus resultados em contextos biológicos. Algumas das propriedades estruturais inclusive podem ser verificadas sem a exploração do espaço de estados gerado pela combinação de interações que podem ocorrer na via biológica.

Também foram exploradas propriedades probabilísticas utilizando Lógica Estocástica Contínua, que possibilitam uma maior flexibilidade na formulação de propriedades que precisam ser verificadas no modelo após a construção do espaço de estados da via biológica. Desta maneira o estudo de caso realizado atingiu o objetivo ao demonstrar a utilização dos métodos propostos neste trabalho num exemplo real, e a interpretação de importantes propriedades que podem ser constatadas e checadas antes e durante a simulação *in silico*.

6 CONCLUSÕES E TRABALHOS FUTUROS

Esta dissertação propôs uma linguagem formal visual como uma interface para a descrição textual das especificações de vias biológicas. O metamodelo definido possibilita técnicas de checagem de consistências nos diagramas ao determinar por meio de regras de construção que tipos de reações podem ocorrer entre determinadas entidades participantes de um sistema biológico.

Apresentamos uma tradução semântica para a linguagem visual que se adequa a modelagem de processos biológicos, possibilitando a verificação de propriedades estruturais e dinâmicas já estudadas em redes de Petri estocásticas. Modelos são constantemente mudados: novos tipos de vértices/arestas/atributos podem se tornar necessários, outros podem se tornar obsoletos, novas representações gráficas podem se tornar necessárias etc. Nós podemos usar o grafo Bio Type como base para um modelo de transformação. Gramáticas de grafos tem sido usadas como uma ferramenta para descrição de transformação de modelos (EHRIG; EHRIG, 2006). A idéia é que nós podemos usar regras de grafos para descrever num nível de metamodelo como o modelo evolui.

Alguns exemplos de regras que poderiam ser usados para mostrar a idéia de evolução são mostrados na figura 6.1. Primeiro, supondo que nós gostaríamos de integrar dois atributos em um. A regra *red_att_cols* por exemplo integra *intervals* (uma lista de pares) e *colors* (uma lista de valores) criando um novo atributo chamado *interval_colors*, que é uma lista de tuplas (o primeiro e o segundo elemento descrevem os intervalos, o terceiro é a cor associada a aresta). Um outro exemplo poderia ser a inclusão no modelo de um atributo do tipo *boolean* para indicar se uma proteína é uma enzima ou não.

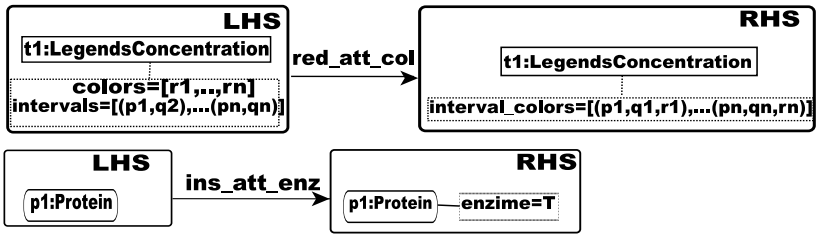


Figura 6.1: Algumas Regras de Evolução.

Apesar das contribuições que a fundamentação formal promove, algumas limitações também foram constadas. GSPNs são isomórficos a cadeias de Markov onde os estados anteriores são irrelevantes para a predição dos estados seguintes (ausência de memória). Entretanto em alguns sistemas biológicos há condições de disparos de eventos que são dependentes de estados anteriores para serem analisados, não possibilitando a simulacão com GSPNs. A seguir são apresentadas algumas possibilidades de trabalhos futuros :

- Proposição de novas traduções do modelo metamodelo proposto para novos modelos semânticos;
- Formalização e inclusão de novos símbolos e estruturas que forem criados do projeto SBGN;
- Implementações de ferramentas que automatizem as etapas de tradução dos modelos Bioproc para a etapa de simulação;
- Avaliação experimental da abordagem proposta, verificando se novas propriedades verificadas nos experimentos *in silico* são detectadas nos experimentos *in vivo*;

7 APÊNDICE

Neste capítulo de apêndice listamos todas as regras de construção de diagramas de processos. As regras estão agrupadas em:

- Regras de inserção e remoção de vértices.
- Regras de inserção e remoção de arestas.

Por exemplo as regras de construções dos vértices foram definidas com o seguinte padrão

- no LHS no temos um diagrama vazio
- no RHS temos o digrama modificado com a inserção de um vértice da linguagem visual. Nas regras especificadas também são inseridos em uma única operação todos os atributos relativos ao vértice.
- na exclusão de um nodo ocorre o inverso o RHS é um diagrama vazio e no LHS o nodo a ser excluído, com uma NAC (negative application condition) associada. NACs intuitivamente são condições que impedem a aplicação de uma regra, por exemplo excluir um vértice que seja origem ou destino de uma aresta.

Já as regras de Inserção de arestas foram definidas com o seguinte padrão

- no LHS no temos um diagrama contendo pelo menos dois nodos que deveram ser dos tipos dos quais podem ser o source e o target da aresta.
- no RHS temos o diagrama modificado com a inserção da aresta bem como o conjunto de atributos.
- na exclusão de uma aresta no LHS temos um diagrama com pelo menos dois nodo conectados pela aresta e no RHS o mesmo diagrama sem a aresta.

A partir de um grafo tipo podemos derivar automaticamente o conjunto de regras necessários para construir instâncias corretas deste grafo. Se nós temos uma sintaxe concreta associada, estas regras podem ser usadas para construir o editor correspondente. Em (BARDOHL, 1999) são mostrados detalhes de como gerar automaticamente estas regras tanto para inserção quanto para remoção de símbolos de um diagrama. Todas as regras que descrevem a gramática de BioProc encontram-se na seção de apêndice desta dissertação.

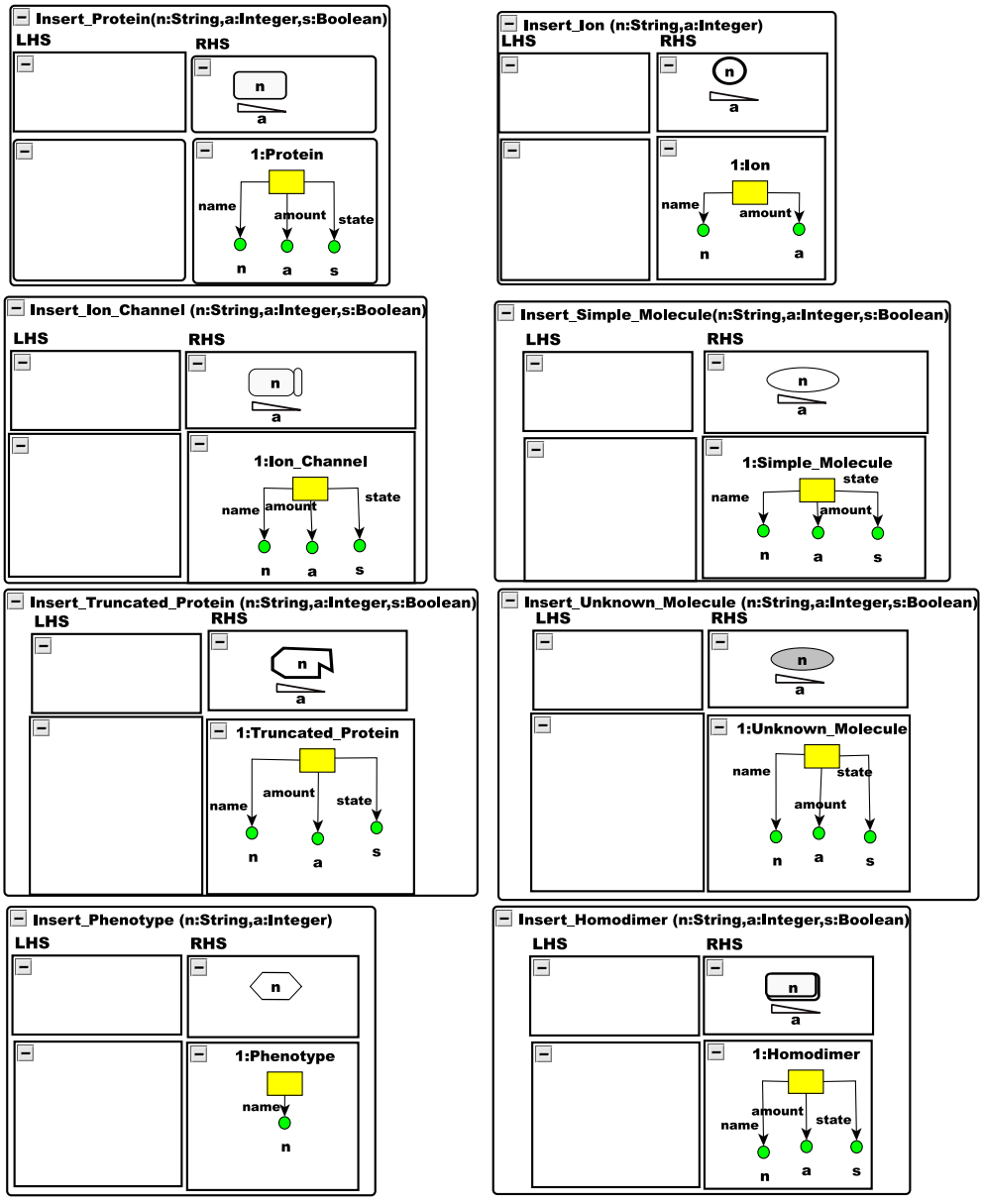


Figura 7.1: Regras de Inserção dos Nodos (parte1).

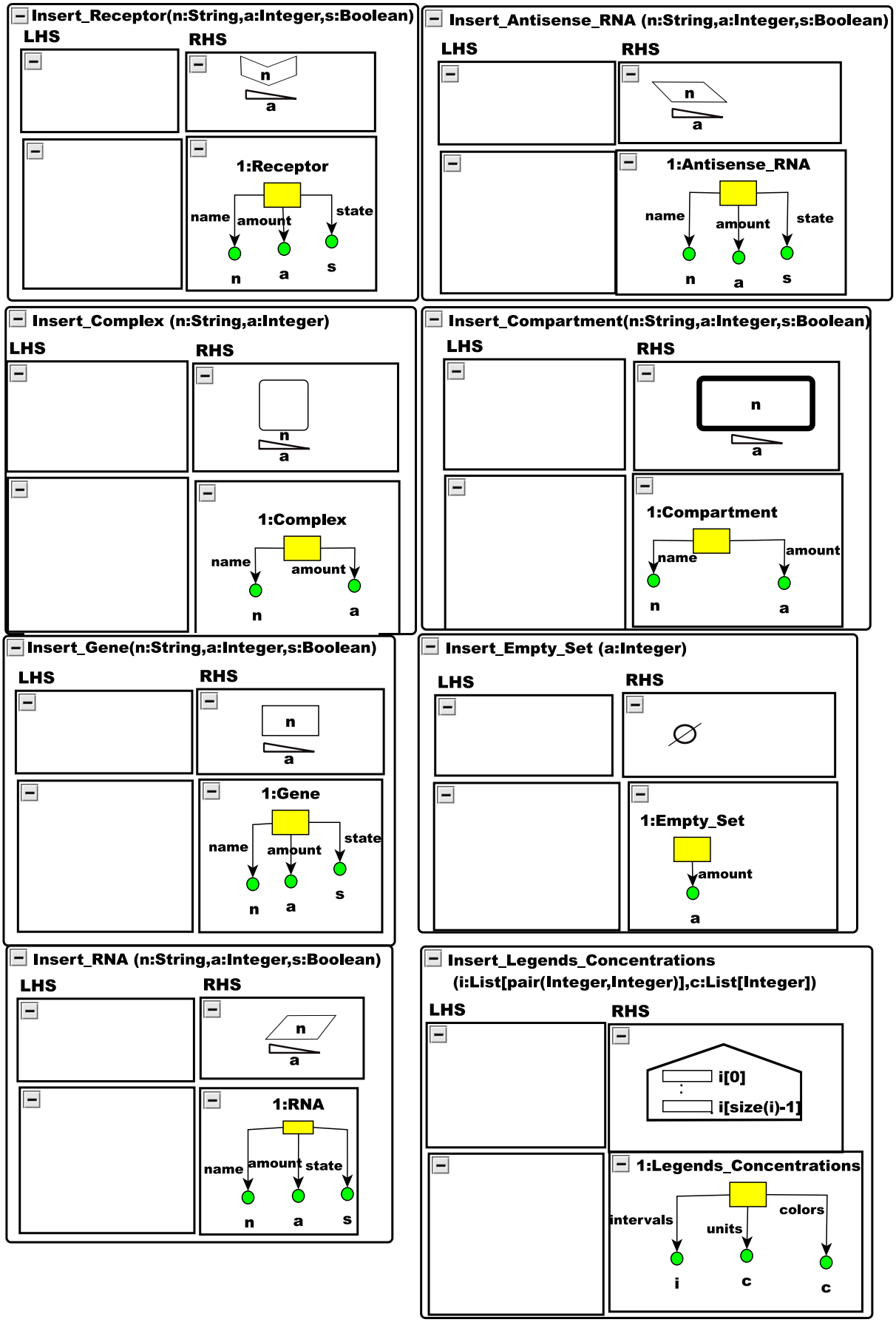


Figura 7.2: Regras de Inserção dos Nodos (parte2).

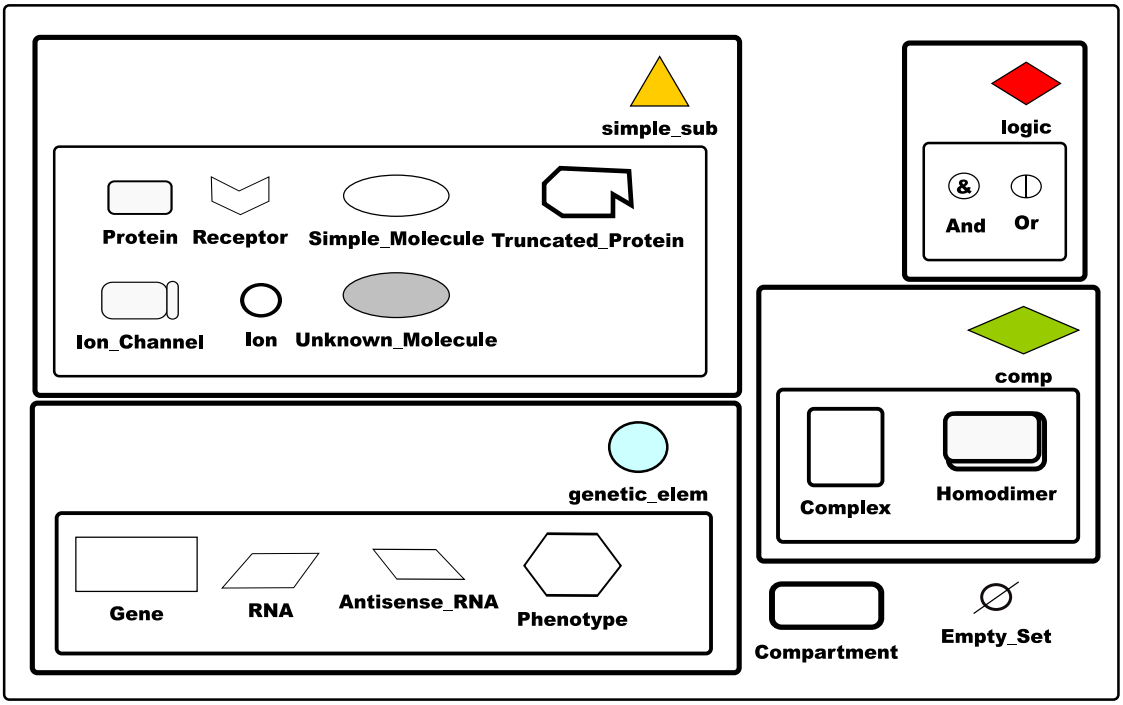


Figura 7.3: Agrupamento de Nodos

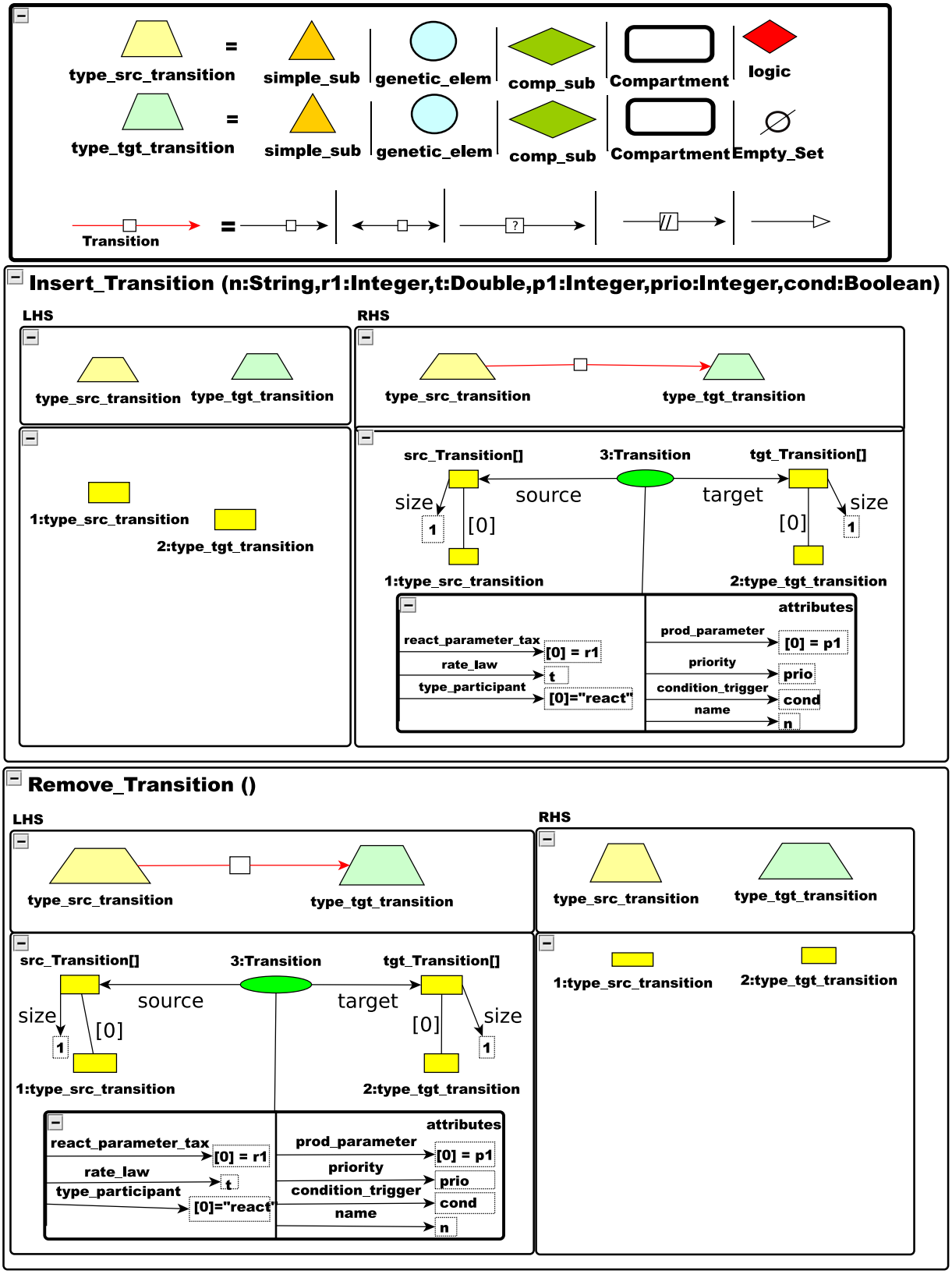


Figura 7.4: Regras de Inserção e Remoção de Transições

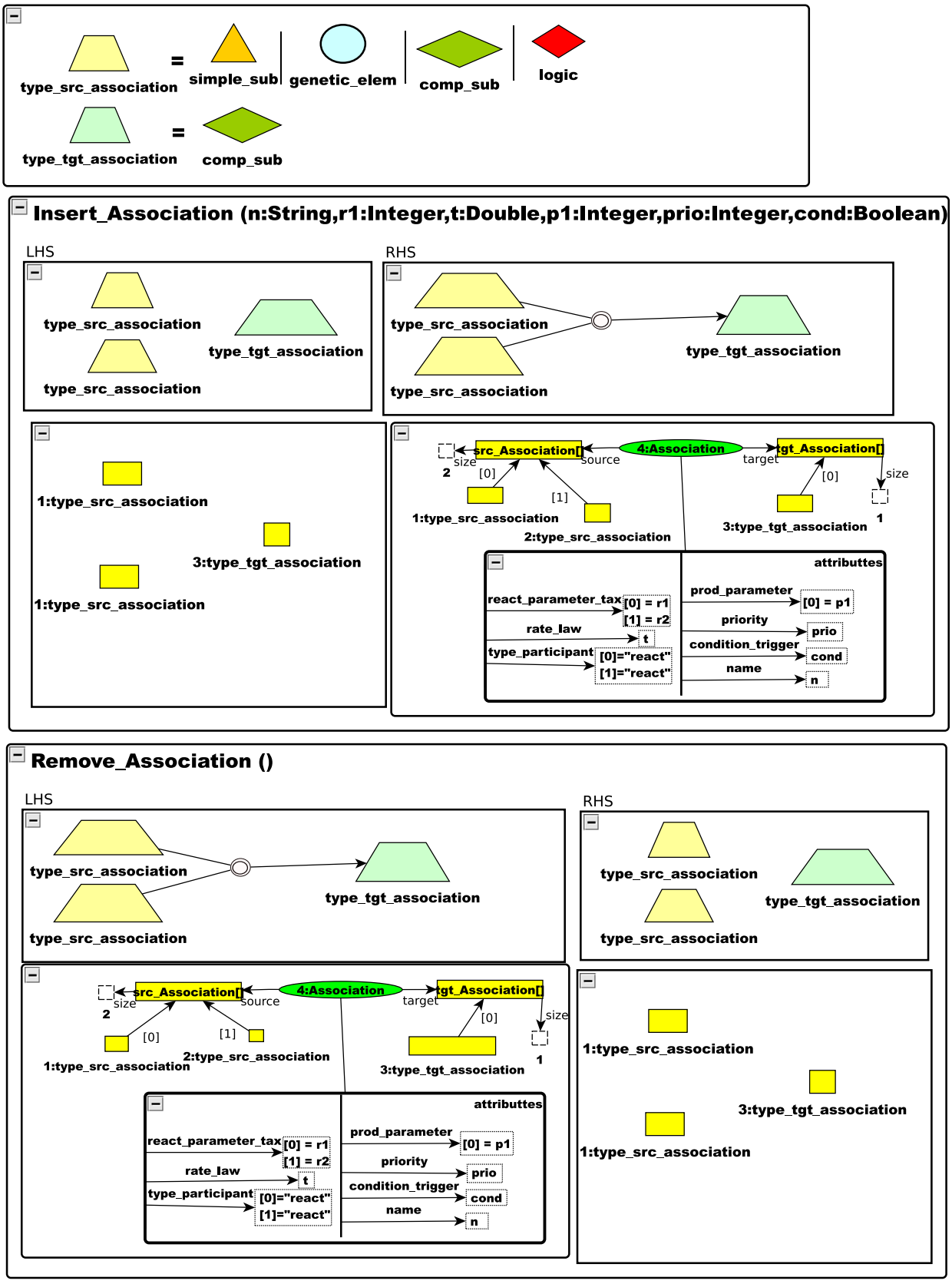


Figura 7.5: Regras de Inserção e Remoção da aresta Association

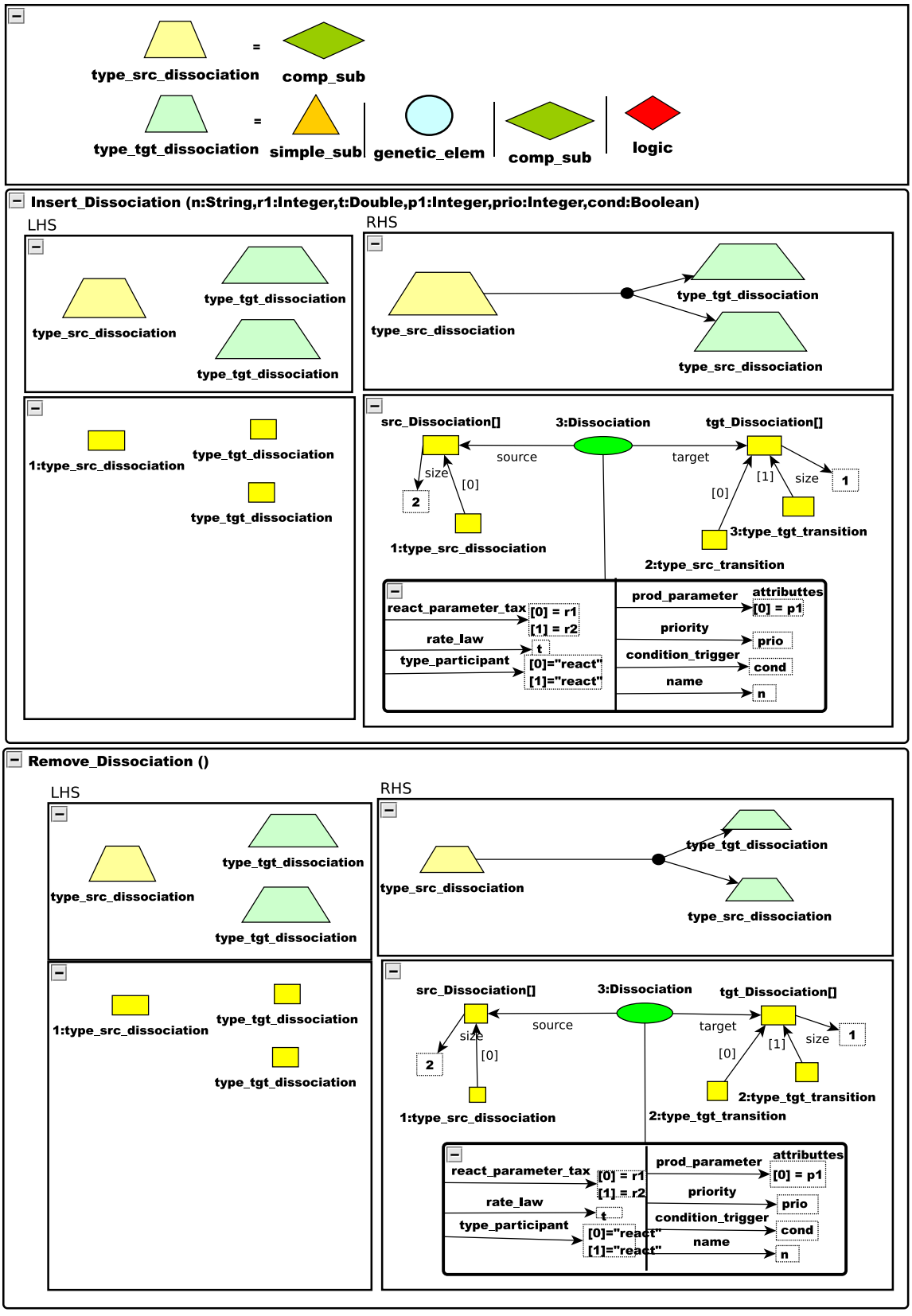


Figura 7.6: Regras de Inserção e Remoção da aresta Dissociation

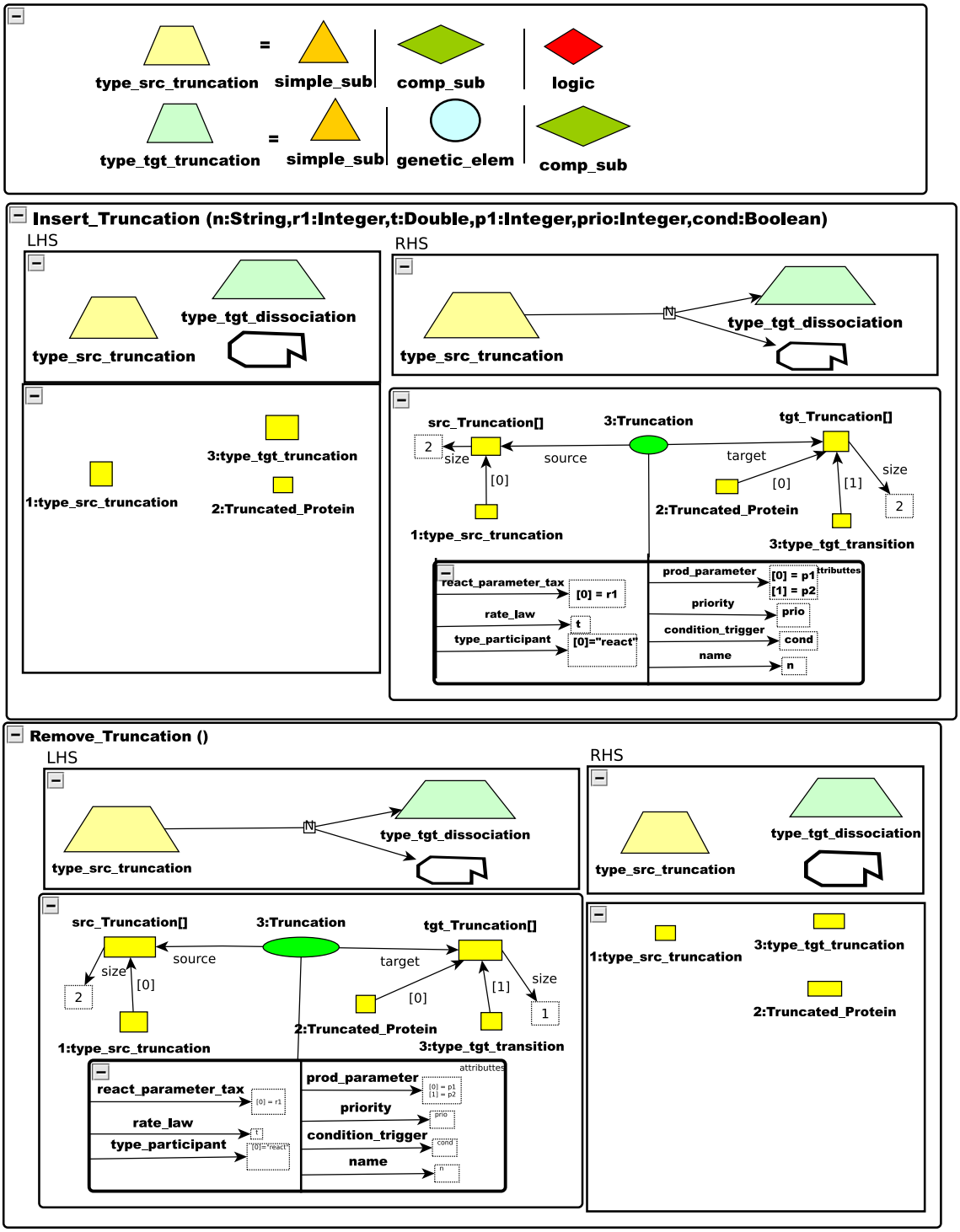


Figura 7.7: Regras de Inserção e Remoção da aresta Truncation

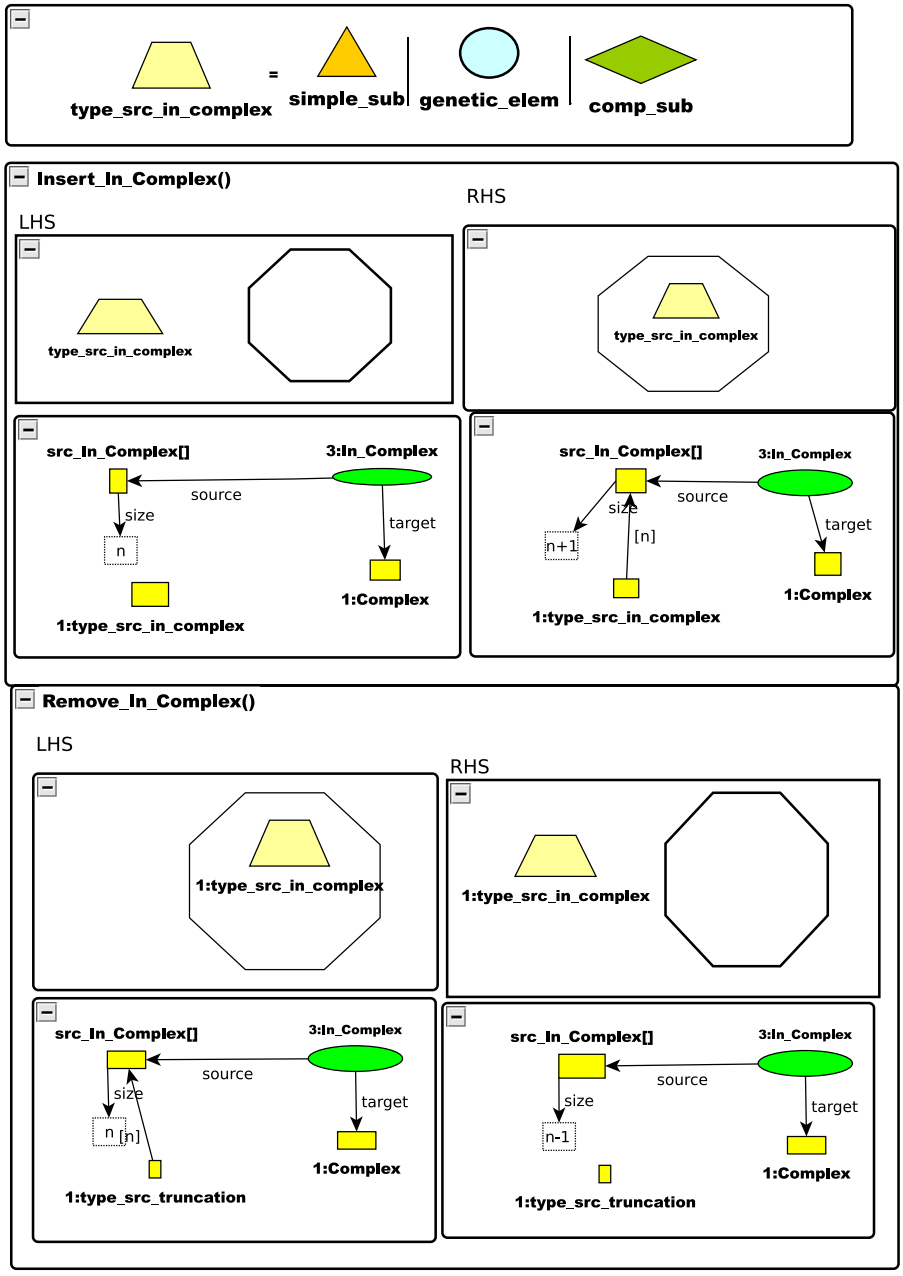


Figura 7.8: Regras de Inserção e Remoção da aresta InComplex

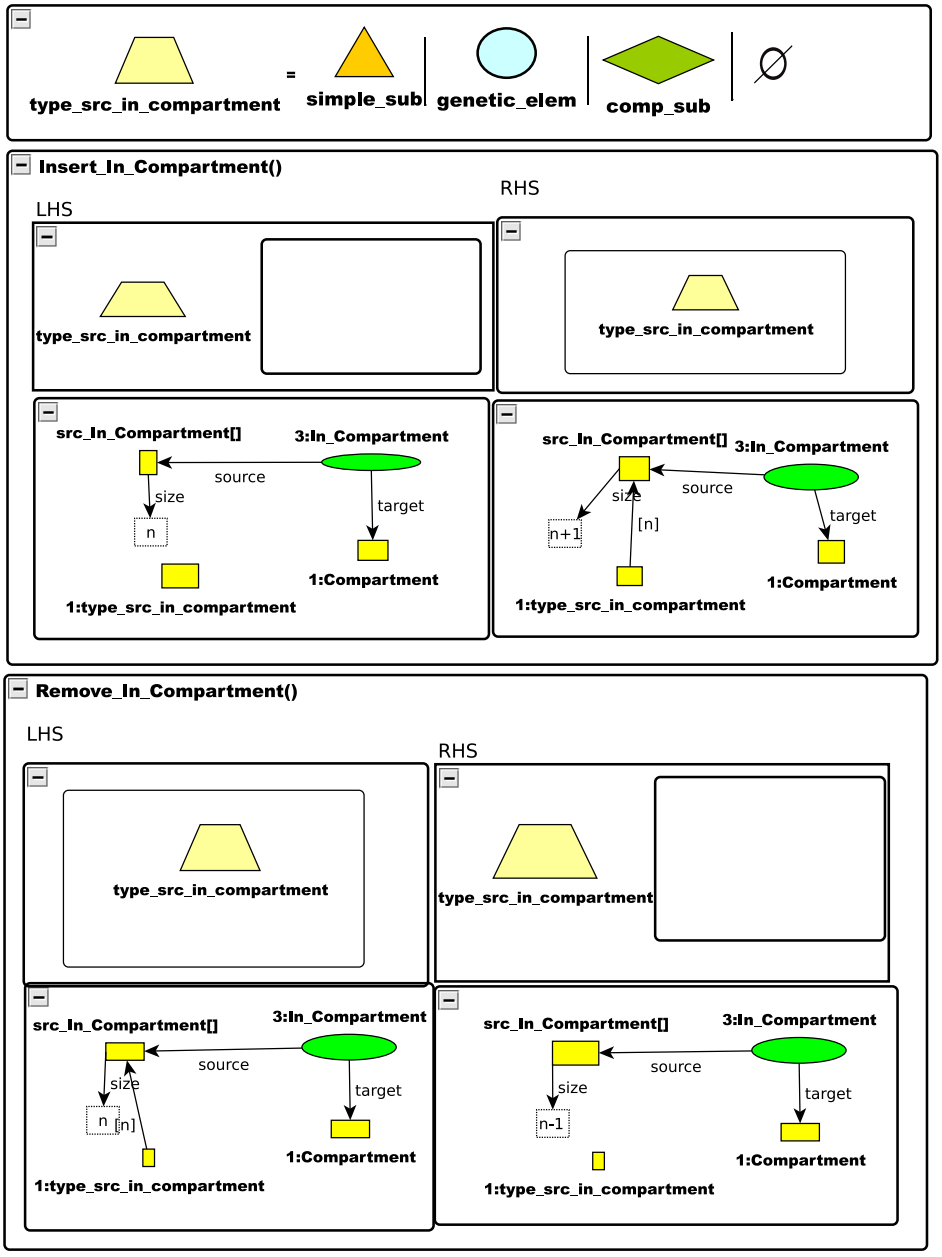


Figura 7.9: Regras de Inserção e Remoção da aresta InCompartment

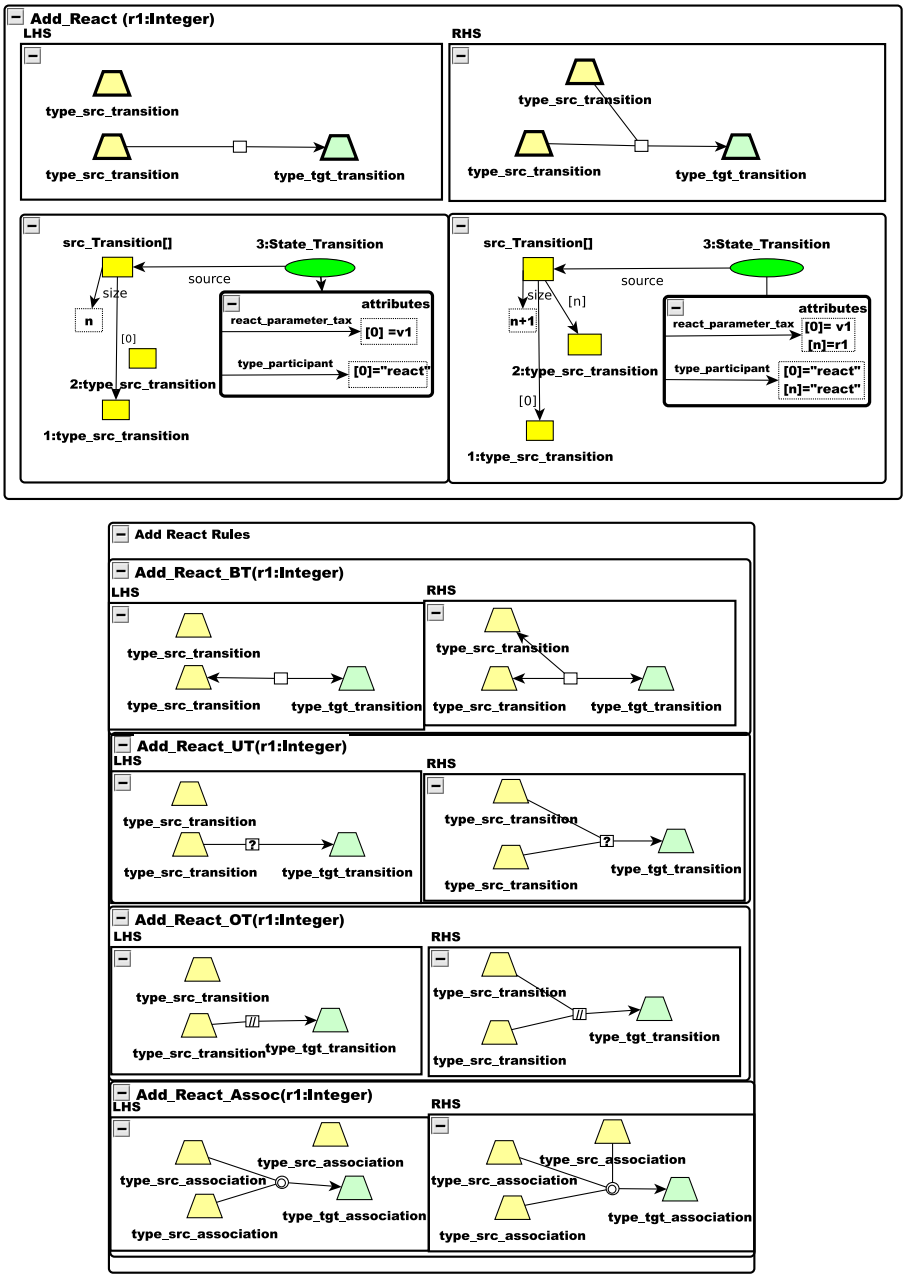


Figura 7.10: Regras de Inserção de um Reagente (Sintaxe concreta e abstrata)

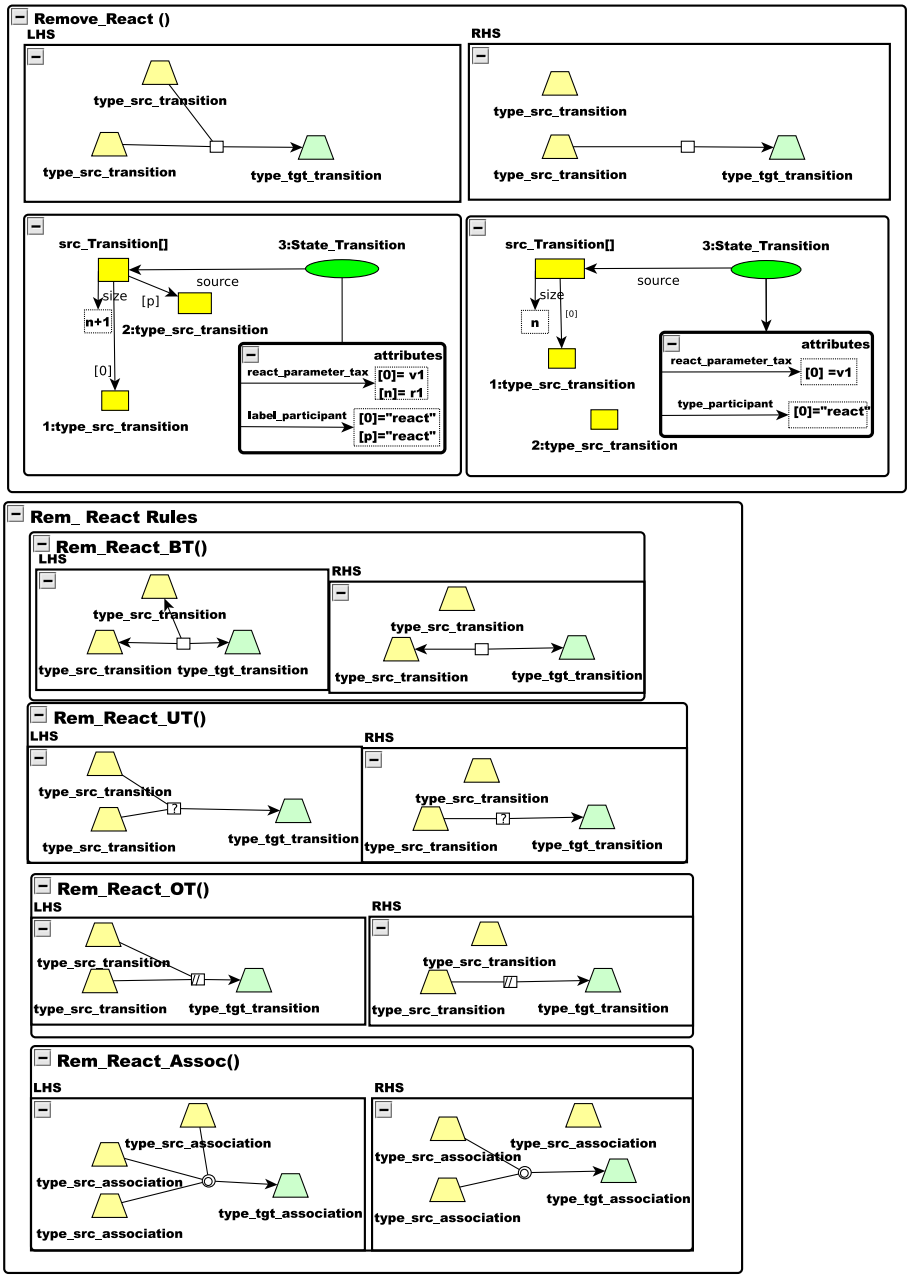


Figura 7.11: Regras de Remoção de um Reagente (Sintaxe concreta e abstrata)

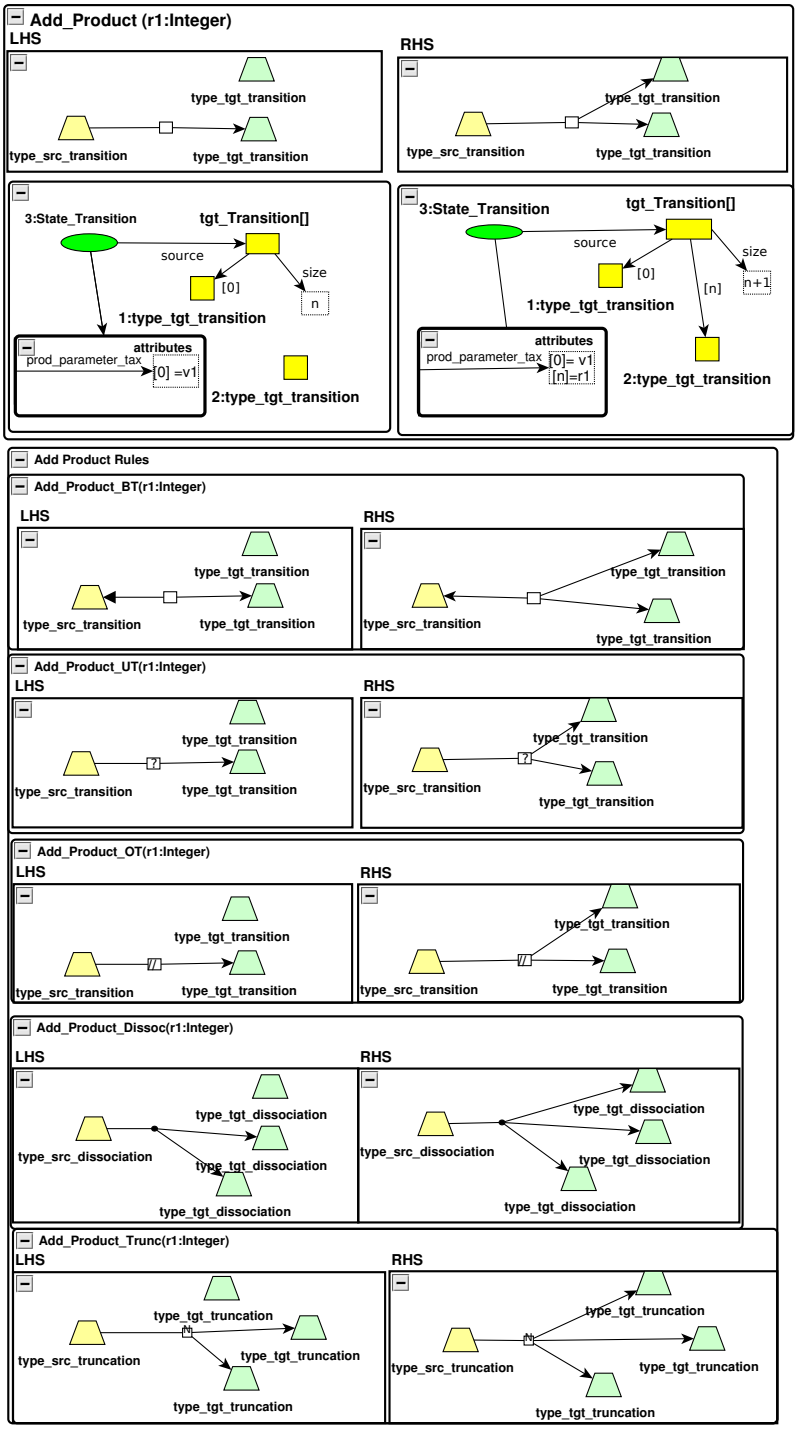


Figura 7.12: Regras de Inserção de um Inibidor (Sintaxe concreta e abstrata)

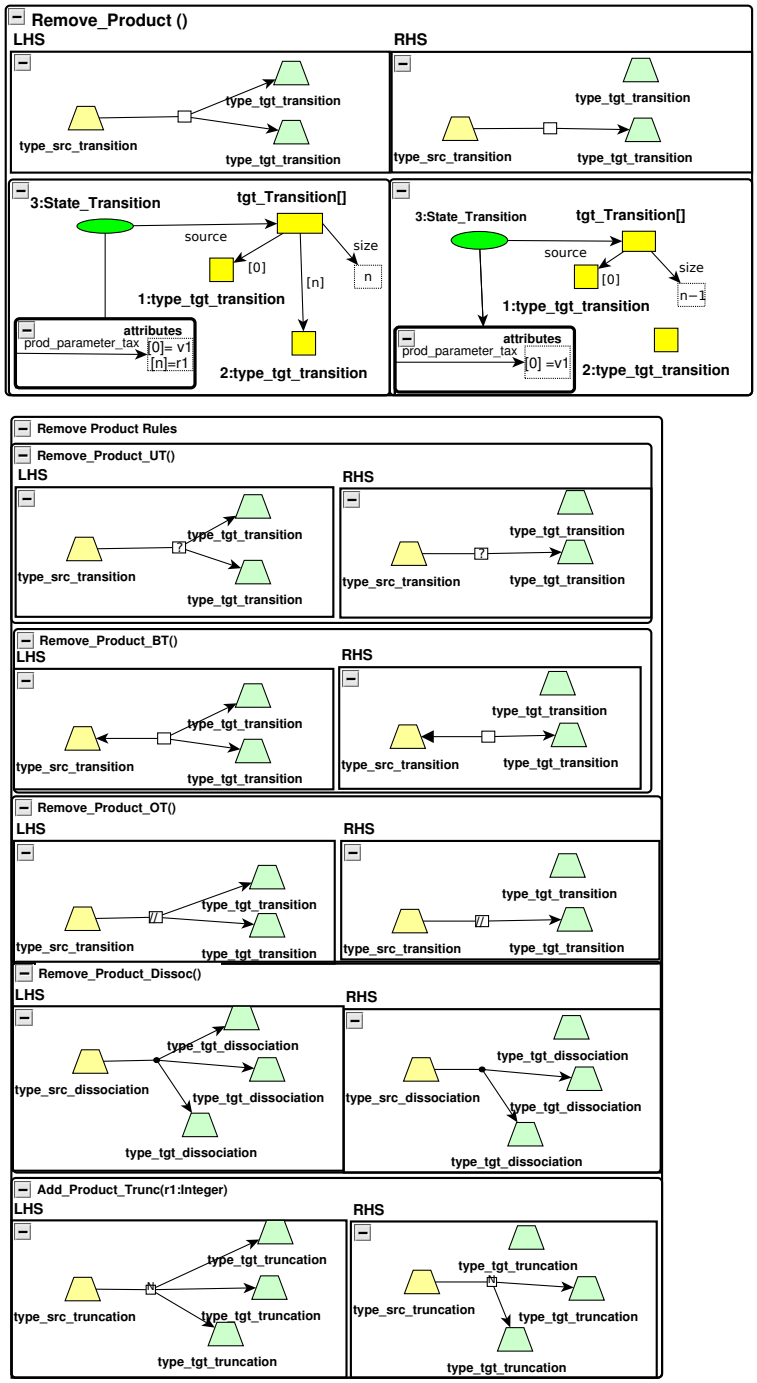


Figura 7.13: Regras de Remoção de um Inibidor (Sintaxe concreta e abstrata)

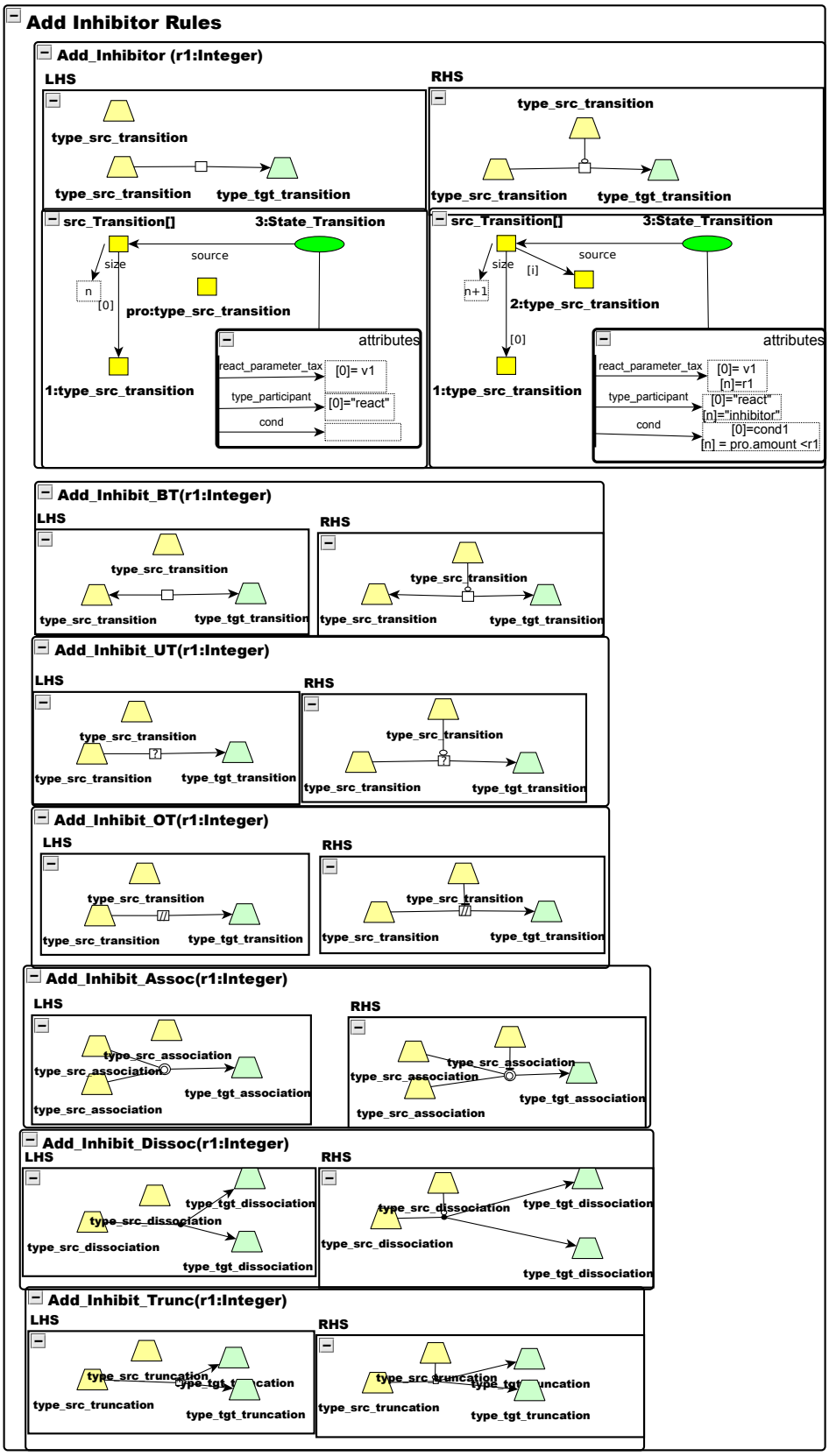


Figura 7.14: Regras de Inserção de um Inibidor (Sintaxe concreta e abstrata)

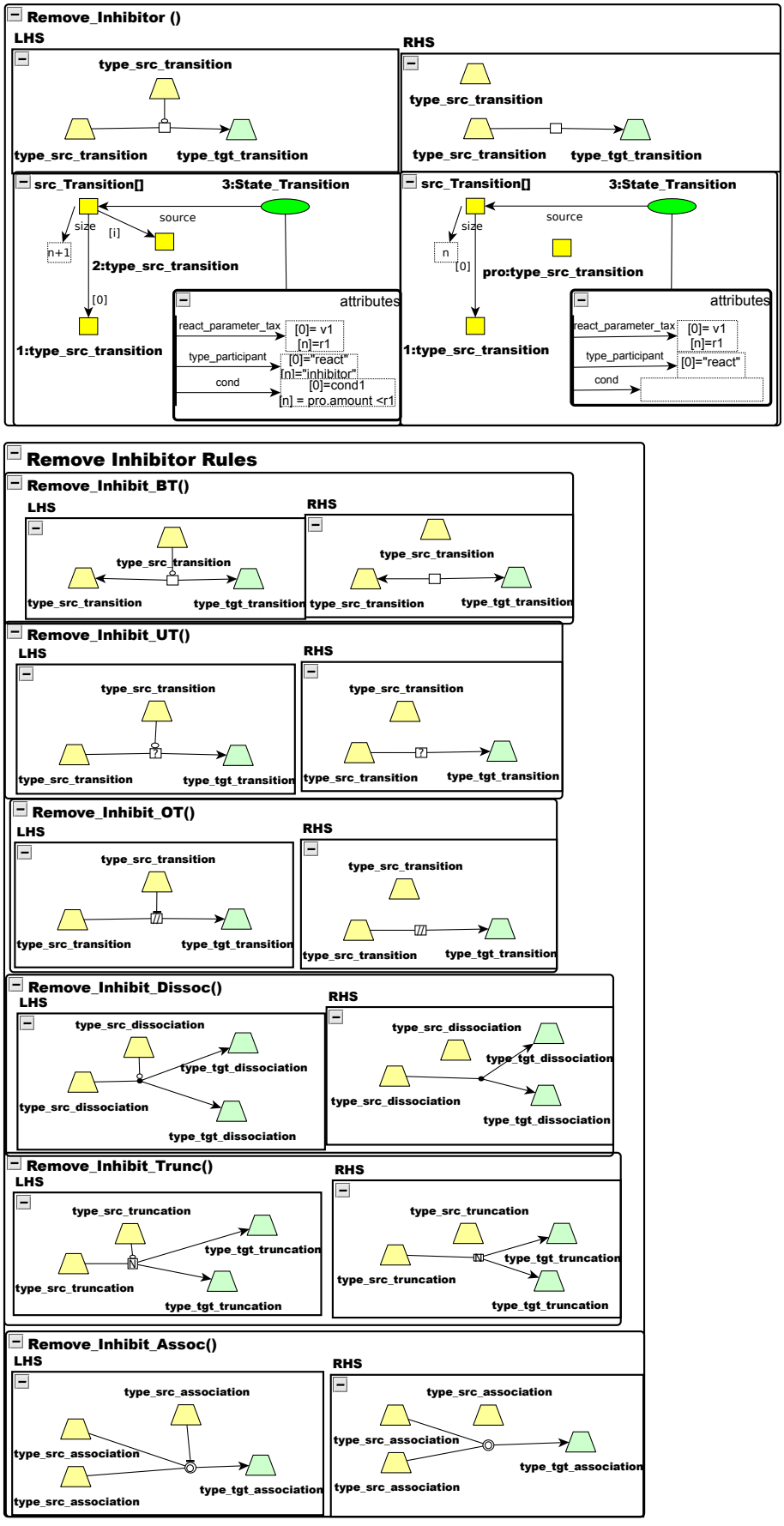


Figura 7.15: Regras de Remoção de um Inibidor (Sintaxe concreta e abstrata)

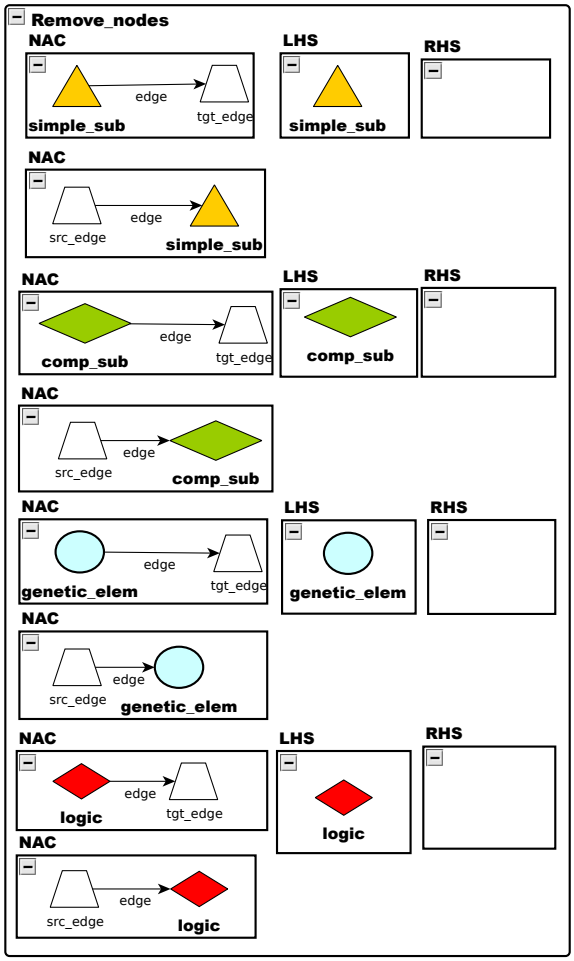


Figura 7.16: Regras de Remoção de nodos(Sintaxe concreta e abstrata)

REFERÊNCIAS

AZIZ, A. et al. Model-checking continuous-time Markov chains. **ACM Trans. Comput. Logic**, New York, NY, USA, v.1, n.1, p.162–170, 2000.

BARBUTI, R. et al. A Calculus of Looping Sequences for Modelling Microbiological Systems. **Fundamenta Informaticae**, [S.l.], v.72, p.1–15, 2006.

BARDOHL, R. **GenGED - Visual Definition of Visual Languages based on Algebraic Graph Transformation**. 1999. Tese (Doutorado em Ciência da Computação) — Technische Universität Berlin.

BONET, P. et al. PIPE v2.5: a petri net tool for performance modelling. In: LATIN AMERICAN CONFERENCE ON INFORMATICS (CLEI 2007), 23. **Anais...** [S.l.: s.n.], 2007.

BRENNER, L. et al. A framework to decompose GSPN models. In: APPLICATIONS AND THEORY OF PETRI NETS 2005. **Anais...** Springer, 2005. p.128–147.

CEROTTI, D. et al. CSL Model Checking for Generalized Stochastic Petri Nets. In: QEST '06: PROCEEDINGS OF THE 3RD INTERNATIONAL CONFERENCE ON THE QUANTITATIVE EVALUATION OF SYSTEMS, Washington, DC, USA. **Anais...** IEEE Computer Society, 2006. p.199–210.

CHAOUIYA, C. Petri net modelling of biological networks. **Brief Bioinform**, [S.l.], v.8, p.210–219, July 2007.

CHEN, M. et al. The biology Petri net markup language. In: PROMISE. **Anais...** GI, 2002. p.150–161. (LNI, v.21).

CHO, K.-H. et al. Mathematical Modeling of the Influence of RKIP on the ERK Signaling Pathway. In: CMSB '03: PROCEEDINGS OF THE FIRST INTERNATIONAL WORKSHOP ON COMPUTATIONAL METHODS IN SYSTEMS BIOLOGY, London, UK. **Anais...** Springer-Verlag, 2003. p.127–141.

CURTI M.; DEGANO, P. . P. C.; BALDARI, C. Modelling Biochemical Pathways through Enhanced pi-calculus. **Theoretical Computer Science**, [S.l.], v.325, p.11–140, 2004.

CUSICK, M. E. et al. Interactome: gateway into systems biology. **Hum Mol Genet**, Center for Cancer Systems Biology and Department of Cancer Biology, Dana-Farber Cancer Institute, 44 Binney Street, Boston, MA 02115, USA. Michael_cusick@dfci.harvard.edu, v.14 Spec No. 2, October 2005.

EHRIG, H.; EHRIG, K. Overview of Formal Concepts for Model Transformations Based on Typed Attributed Graph Transformation. **Electr. Notes Theor. Comput. Sci.**, [S.l.], v.152, p.3–22, 2006.

EHRIG, H. et al. **Fundamentals of Algebraic Graph Transformation (Monographs in Theoretical Computer Science. An EATCS Series)**. [S.l.]: Springer, 2006.

FUNAHASHI, A. et al. CellDesigner 3.5: a versatile modeling tool for biochemical networks. **Proceedings of the IEEE**, [S.l.], v.96, n.8, p.1254–1265, Aug. 2008.

HANDBOOK OF GRAPH GRAMMARS AND COMPUTING BY GRAPH TRANSFORMATIONS, VOLUME 1: FOUNDATIONS. **Anais...** World Scientific, 1997.

HECKEL, R.; LAJIOS, G.; MENGE, S. Stochastic Graph Transformation Systems. **Fundam. Inf.**, Amsterdam, The Netherlands, The Netherlands, v.74, n.1, p.63–84, 2006.

HUCKA, M. et al. The systems biology markup language (SBML): a medium for representation and exchange of biochemical network models. **Bioinformatics**, [S.l.], v.19, n.4, p.524–31, 2003.

JONG, H. de. Modeling and Simulation of Genetic Regulatory Systems: a literature review. **Journal of Computational Biology**, Institut National de Recherche en Informatique et en Automatique (INRIA), Unité de Recherche Rhône-Alpes, 655 avenue de l'Europe, Montbonnot, 38334 Saint Ismier CEDEX, France. Hidde.de-Jong@inrialpes.fr, v.9, n.1, p.67–103, January 2002.

KARTSON, D. et al. **Modelling with Generalized Stochastic Petri Nets**. New York, NY, USA: John Wiley & Sons, Inc., 1994.

KIM, D. H. et al. Current Research Trends in Systems Biology. **Animal Cells and Systems**, [S.l.], v.12, p.181–191, Dec 2008.

KITANO, H. Scientific Challenges in Systems Biology. In: CHOI, S. (Ed.). **Introduction to Systems Biology**. [S.l.]: Humana Press, 2007. p.3–13.

KITANO, H. et al. Using process diagrams for the graphical representation of biological networks. **Nature Biotechnology**, [S.l.], v.23, n.8, p.961–966, August 2005.

KLAMT, S.; HAUS, U.-U.; THEIS, F. Hypergraphs and Cellular Networks. **PLoS Comput Biol**, [S.l.], v.5, n.5, p.e1000385+, May 2009.

KNUTH, D. E. backus normal form vs. Backus Naur form. **Commun. ACM**, New York, NY, USA, v.7, n.12, p.735–736, 1964.

KOLPAKOV, F. BioUML—open source extensible workbench for systems biology. In: THE FOURTH INTERNATIONAL CONFERENCE ON BIOINFORMATICS OF GENOME REGULATION AND STRUCTURE, Novosibirsk, Russia. **Proceedings...** [S.l.: s.n.], 2004. p.77–80.

KWIATKOWSKA, M.; NORMAN, G.; PARKER, D. PRISM: probabilistic symbolic model checker. In: INTERNATIONAL CONFERENCE ON MODELLING TECHNIQUES AND TOOLS FOR COMPUTER PERFORMANCE EVALUATION (TOOLS'02), 12. **Proceedings...** Springer, 2002. p.200–204. (LNCS, v.2324).

L. BORTOLUSSI, A. P. **Connecting Process Algebras and Differential Equations for Systems Biology**. London, 2006.

DEAN, J. (Ed.). **Lange's Handbook of Chemistry**. [S.l.]: McGraw-Hill, 1999.

LE NOVèRE, N. et al. BioModels Database: a free, centralized database of curated, published, quantitative kinetic models of biochemical and cellular systems. **Nucleic Acids Res**, European Bioinformatics Institute EMBL, Wellcome-Trust Genome Campus, Hinxton, CB10 1SD, UK. lenov@ebi.ac.uk, v.34, n.Database issue, January 2006.

LEE M HYUN S, P. S. UniPath: a knowledge representation system for biological pathways. **Genome Informatics**, [S.l.], v.14, p.681–682, 2003.

LUCIANO, J. S.; STEVENS, R. D. e-Science and biological pathway semantics. **BMC Bioinformatics**, Genetics Department, Harvard Medical School, Boston, MA 02115, USA.jluciano@genetics.med.harvard.edu, v.8 Suppl 3, p.1–21, 2007.

LüDEMANN, E. et al. PaVESy: pathway visualization and editing system. **Bioinformatics**, [S.l.], v.20, p.2841–2844, 2004.

MAIMON, R.; BROWNING, S. Diagrammatic notation and computational structure of gene networks. In: INTERNATIONAL CONFERENCE ON SYSTEMS BIOLOGY, 2., Madison, WI. **Proceedings...** Omnipress, 2001. p.311–317.

MOODIE, S. L. et al. A Graphical Notation to Describe the Logical Interactions of Biological Pathways. **J Integr Bioinfo**, [S.l.], v.3, p.36+, 2006.

NOVERE, N. L. et al. **Systems Biology Graphical Notation**: process diagram level 1. 2008.

PETRI, C. A. **Kommunikation mit Automaten.** 1962. Tese (Doutorado em Ciência da Computação) — Bonn: Institut für Instrumentelle Mathematik, Schriften des IIM Nr. 2. Second Edition:, New York: Griffiss Air Force Base, Technical Report RADC-TR-65–377, Vol.1, 1966, Pages: Suppl. 1, English translation.

PINTO, M. C. et al. Modelling, property verification and behavioural equivalence of lactose operon regulation. **Comput. Biol. Med.**, Elmsford, NY, USA, v.37, n.2, p.134–148, 2007.

SCHILLING, M. et al. Standardizing experimental protocols. **Current Opinion in Biotechnology**, [S.l.], v.19, n.4, p.354 – 359, 2008. Protein technologies / Systems biology.

KITANO, H. (Ed.). **Foundations of Systems Biology**. [S.l.]: The MIT Press, 2001. p.2–36.

PRESS, T. M. (Ed.). **Design Concepts in Programming Languages**. [S.l.]: The MIT Press, 2008.

PRESS, C. U. (Ed.). **Computational Analysis of Biochemical Systems**. [S.l.]: Cambridge University Press, 2001. 311p. v.25, n.3.

XIONG, J. **Essential Bioinformatics**. New York.: Cambridge University Press, 2006.

ZEVEDEI-OANCEA, I.; SCHUSTER, S. Topological analysis of metabolic networks based on Petri net theory. **In Silico Biology**, [S.l.], v.3, p.29, 2003.