Introdução às Redes de Petri Coloridas usando a ferramenta CPN Tools

Giovanni Cordeiro Barroso[1 2], **José Marques Soares**[3]

[1]Departamento de Física – Universidade Federal do Ceará (UFC)

[2]Cursos de Pós-Graduação em Engenharia de Teleinformática
Universidade Federal do Ceará (UFC)

[3]Departamento de Engenharia de Teleinformática
Universidade Federal do Ceará (UFC)

Resumo. *Neste texto é apresentada uma introdução às redes de Petri Coloridas e à ferramenta de modelagem, análise e simulação CPN Tools, desenvolvida na Univesidade de Aarhus (Dinamarca). O texto é baseado, em parte, no livro intitulado* Simulating of Telecommunication Systems with CPN Tools*, de autoria de D. A. Zaitsev e T. R. Shmeleva e no* Help *da ferramenta CPN Tools. O referido livro é usado como livro texto da disciplina* Mathematical Modeling of Information Systems *do curso de Mestrado em Comunicações da* Odessa National Academy of Telecommunication *da Ucrânia. O Help do CPN Tools é encontrado facilmente na internet. Foram feitas várias modificações no texto para adaptá-lo às necessidades da disciplina* Sistemas Híbridos *do curso de Pós-Graduação em Engenharia de Teleinformática da UFC.*

1. Introdução

O CPN Tools é uma ferramenta de modelagem, análise e simulação de redes de Petri coloridas. A ferramanta foi desenvolvida na Universidade de Aarhus, Dinamarca, e é distribuído livremente para organizações não comerciais via web, no endereço http://cpntools.org/. Esta ferramenta tem sido utilizada em várias aplicações e projetos importantes, especialmente nas áreas de telecomunicações e sistemas de manufatura.

2. Classes de redes de Petri implementadas no CPN Tools

O CPN Tools suporta a descrição de modelos de uma classe de redes de Petri muito poderosa, denominada *Redes de Petri de Alto Nível* [Jensen and Kristensen 2009]. Esta classe de redes de Petri suporta hierarquia e restrições de tempo.

As redes de Petri coloridas (RPC) receberam esta denominação porque quando foram propostas, as mesmas utilizavam diferentes tipos de fichas especificadas por números naturais e representadas visualmente por cores: 1-vermelha, 2-azul, 3-verde, etc. O conceito de RPCs utilizadas no CPN Tools é um pouco mais complicado. Tais redes são frequentemente chamadas de redes de Petri coloridas generalizadas porque o tipo de ficha é descrito como um tipo de dado abstrato, como em uma linguagem de programação. O termo `colorida` é usado ainda por razões hitóricas.

Redes de Petri com restrições de tempo utilizam o conceito de tempo para representar a duração de ações, tarefas e eventos em um sistema real. Apesar do disparo das transições ocorrerem instantaneamente nas redes de Petri, o disparo de uma transição em

uma rede de Petri com restrições de tempo representa o retardo de tempo relativo à própria dinâmica de um sistema real. Isso permite a análise de sistemas reais, tais como tempo de resposta em uma rede de computadores, que é uma característica de qualidade de serviço (QoS) da mesma.

Redes hierárquicas permitem a contrução de modelos mais complexos. Em tais redes, um subsistema e sua dinâmica podem ser modelados por uma outra rede (sub-rede). No CPN Tools, uma transição pode ser substituída por uma rede adicional. Assim, pode-se ter uma construção aninhada (rede e subredes). O número de hierarquias não tem, em princípio, limitações. Note que a ideia é largamente utilizada em linguagem de programação, onde procedimentos e funções são utilizados para se manter o entendimento de um programa mais complexo.

2.1. O Grafo das Redes de Petri Coloridas e a Linguagem CPN ML

No CPN Tools, a descrição de um modelo é constituída da combinação da estrutura da rede de Petri com a linguagem de programação CPN ML (*Markup Language*).

A estrutura da rede de Petri é constituída de um grafo direcionado bipartido, possuindo dois tipos de vertices: lugares, representados por círculos ou elipses, e transições, representadas por barras ou boxes (retângulos ou quadrados). Arcos direcionados conectam os lugares às transições e as transições aos lugares. Veja Figura 1

Figura 1. Estrutura de uma rede de Petri

Além de sua estrutura, as redes de Petri apresentam fichas nos lugares. Uma ficha é um objeto dinâmico que se movimenta através dos lugares como resultado do disparo das transições.

Nas redes de Petri lugar/transição (ou simplesmente redes de Petri) [Murata 1989] todas as fichas são valores inteiros positivos e indistinguíveis. Nas redes de Petri coloridas existem vários tipos de fichas, portanto, as mesmas são distinguíveis. Veja Figura 2.

Na figura é apresentada uma RPC com três lugares e uma transição, cuja marcação inicial (Figura 2(a)) apresenta cinco fichas do tipo `INT` no lugar `p1 (5‘3)` e uma ficha do tipo `INT` no lugar `p2 (1‘4)`. As fichas em `p1` possuem o mesmo valor (cinco cópias de valor 3) e a ficha em `p2` possui valor 4. Na Figura 2(b) é apresentada a nova marcação da RPC após o disparo da transição. Foi retirada uma ficha de valor 3 do lugar `p1`, a ficha de valor 4 do lugar `p2` e foi acrescentada ao lugar `p3` uma ficha cujo valor é a soma das fichas retiradas dos lugares de entrada da transição `(1‘7)`.

Como pode ser visto, no CPN Tools uma linguagem de programação especial é incluída para a descrição de atributos dos elementos da rede. Esta linguagem fornece

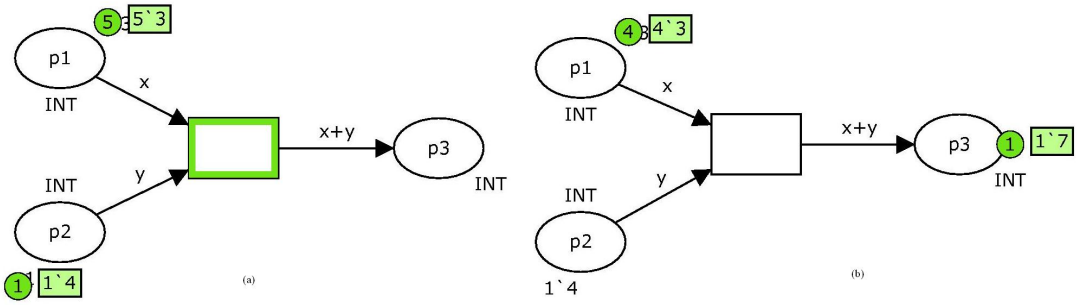

Figura 2. Rede de Petri colorida. (a) marcação da RPC antes do disparo da transição; (b) marcação da RPC após o disparo da transição

declarações de conjuntos coloridos (tipos), variáveis, constantes, funções e procedimentos. No exemplo da Figura 2 as seguintes declarações foram utilizadas:

```
colset INT = int;
var x, y :  INT;
```

Um conjunto de cores foi definido: `INT`, do tipo inteiro. Assim, todos os lugares da rede possuem o conjunto de cores `INT` associado. Foram também definidas as variáveis `x` e `y`, as quais são do tipo `INT`.

No exemplo da figura, a transição está habilitada, pois pode-se fazer a ligação da variável `x` com uma das cinco fichas do lugar `p1`, bem como da variável `y` com a ficha de valor 4 do lugar `p2`:

$x \rightarrow 3;$
$y \rightarrow 4;$

Assim, a marcação da rede após o disparo da transição apresenta quatro fichas no lugar `p1`, zero ficha no lugar `p2` e uma ficha no lugar `p3`, cujo valor é a soma das fichas retiradas, respectivamente, dos lugares de entrada da transição.

2.2. Um exemplo mais complexo

Considere um exemplo menos simples para ajudar no estudo do CPN Tools. O mesmo é uma adaptação do exemplo apresentado em [Zaitsev and Shmeleva 2006] e foi tirado do conhecido conto de fadas Cinderela:

> A madrasta de Cinderela mandou que ela separasse grãos de diferentes tipos, mas no exemplo, camundongos amigos de Cinderela separam os grãos enquanto ela vai ao baile. Veja Figura 3

Neste modelo RPC, as seguintes declarações de conjuntos de cores e variáveis foram utilizadas:

```
colset A = unit with abobora;
colset C = unit with Cinderela;
colset G = with arroz | aveia | trigo;
colset R = unit with rato;
colset F = unit with Fada;
```

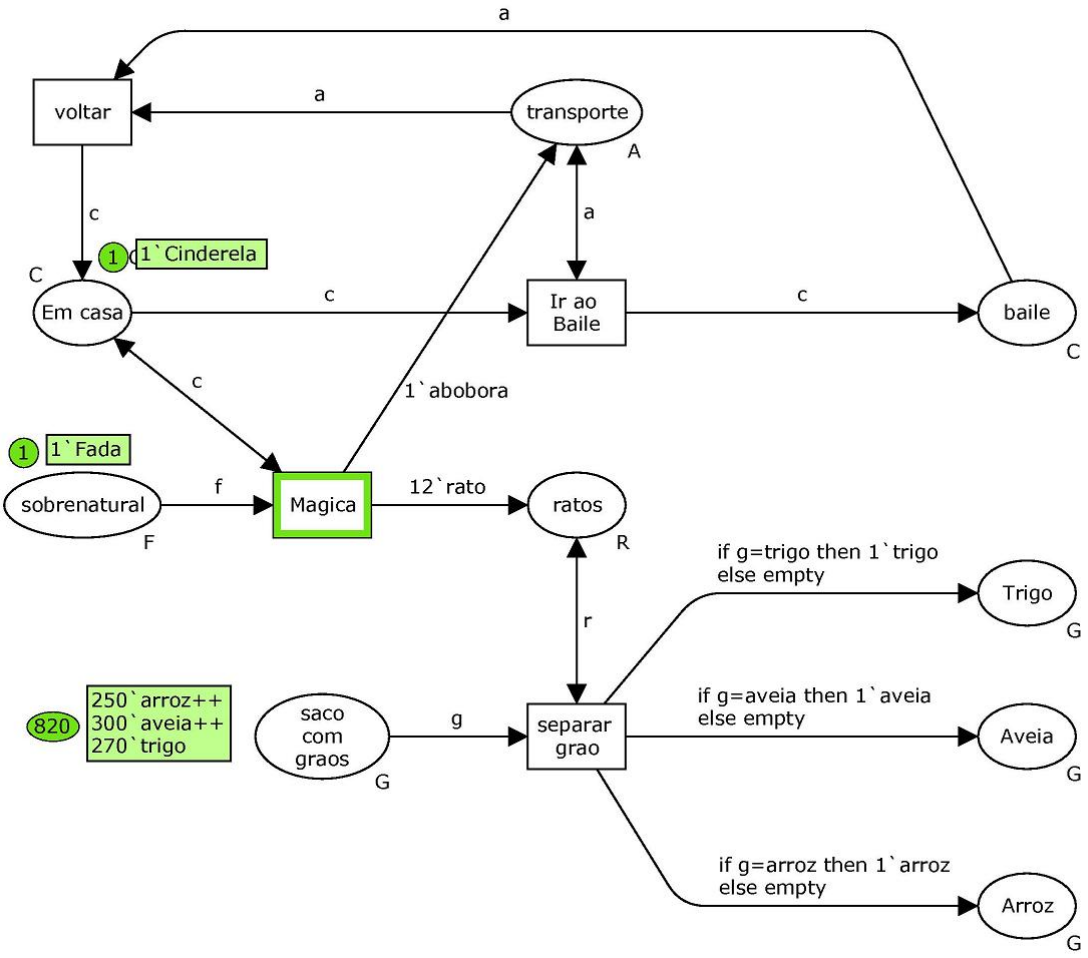

Figura 3. RPC que modela a ida de Cinderela ao baile, enquanto seus amigos separam os grãos por ela

```
var a :  A;
var c :  C;
var g :  G;
var r :  R;
var f :  F;
```

Neste exemplo, foram criados cinco conjuntos de cores, a saber: `A`, com uma ficha denominada `abobora`; `C`, com uma ficha denominada `Cinderela`; `G`, com três fichas denominadas, respectivamente, `arroz, aveia` e `trigo`; `R`, com uma ficha denominada `rato` e; `F` com uma ficha denominada `Fada`. Na marcação inicial, somente a transição `Magica` está habilitada. Veja que ela se encontra em destaque, possuindo uma aura verde. Conversando com Cinderela, a fada cria doze ratos, uma abóbora e desaparece (disparo da transição Magica). A viagem de Cinderela até o baile e a separação dos grãos são eventos concorrentes e, assim, podem ocorrer simultaneamente e em qualquer ordem. A abóbora é usada como recurso para habilitação e disparo das transições `ir ao baile` e `voltar`, para levar Cinderela ao baile e trazê-la de volta. Assim, um auto-laço é usado (arco bidirecional ligando o lugar `transporte` à transição `ir ao baile`, (Figura 3). Os ratos são usados como recursos para habilitação e disparo da transição `separar`

grao, para separação dos diferentes tipos de grãos.

Considerando o sentido dos arcos e suas respectivas inscrições, o disparo da transição Magica não modifica a marcação do lugar Em casa (que possui uma ficha de cor Cinderela), e o disparo da transição Ir ao baile não muda a marcação do lugar transporte (ficha de cor abobora), visto que o mesmo recurso é utilizado para trazer de volta Cinderela para sua casa (disparo da transição voltar). Os outros arcos são unidirecionais. Um arco direcionado de um lugar para uma transição, significa que uma ficha (ou mais fichas) será retirada do lugar no momento do disparo da respectiva transição. Isso é feito de acordo com a inscrição do arco de entrada da transição. No exemplo, todas as inscrições são variáveis do conjunto de cores correspondente. Por exemplo, o arco de entrada da transição separar grao possui a inscrição g. g é uma variável do conjunto de cores G, assim, um grão é retirado, de forma arbitrária, do lugar saco com graos, no momento do disparo de separar. Inscrições de arco mais complexas serão estudadas nas seções seguintes.

Os arcos de saída da transição, no momento de seu disparo, irão criar novas fichas, as quais poderão, ou não coincidir com aquelas retiradas dos lugares de entrada da transição, ou seja, novas fichas poderão ser criadas. Por exemplo, no disparo da transição Ir ao baile, a ficha Cinderela é retirada do lugar Em casa, através da variável c associada ao arco que liga este lugar à transição. Uma ficha de mesmo valor, ou seja, Cinderela, será colocada no lugar baile, de acordo com a inscrição c do arco que liga a transição Ir ao baile a este lugar.

A transição Magica é um pouco mais complicada. No disparo desta transição, a ficha Fada é retirada do lugar sobrenatural, através da variável f e desaparece, visto que esta variável não está associada ao arco de saída da transição. Já a marcação do lugar Em casa não é afetada devido ao auto-laço (arco bidirecional) com a variável c associada. Como resultado ainda do disparo desta transição, doze fichas rato e uma ficha abobora são criadas, conforme respectivas inscrições nos arcos de saída da transição. Veja na Figura 4 a marcação final do modelo após 823 passos de simulação.

Nesta marcação, Cinderela retornou do baile e se encontra em casa; a fada desapareceu e; seus amigos ratos terminaram a separação dos grãos, ficando 270 grãos de trigo, 300 de arroz e 250 de aveia, nos respectivos lugares.

Note que o disparo separar grao retira aleatoriamente uma ficha de saco com graos através da variável g, mas este grão só será colocado em somente um dos lugares de saída da transição (arroz, aveia ou trigo). Os arcos de saída da transição contêm funções associadas que selecionam somente o grão do tipo requerido, ou um tipo de ficha especial, denominada empty. A escolha de uma ficha empty, na realidade, significa que nenhuma ficha será colocada no lugar de saída da transição.

Considerando o modelo RPC estudado, o mesmo não apresenta várias das características do conto de fadas. Por exemplo, o tempo não é considerado e, por isso, o aviso que a Fada dá à Cinderela sobre a meia noite não está modelado. Recomenda-se, então, que o mesmo seja analisado para se enumerar outras características do conto e, após um estudo mais aprofundado do CPN Tools, poder-se construir um modelo mais próximo do conto original.

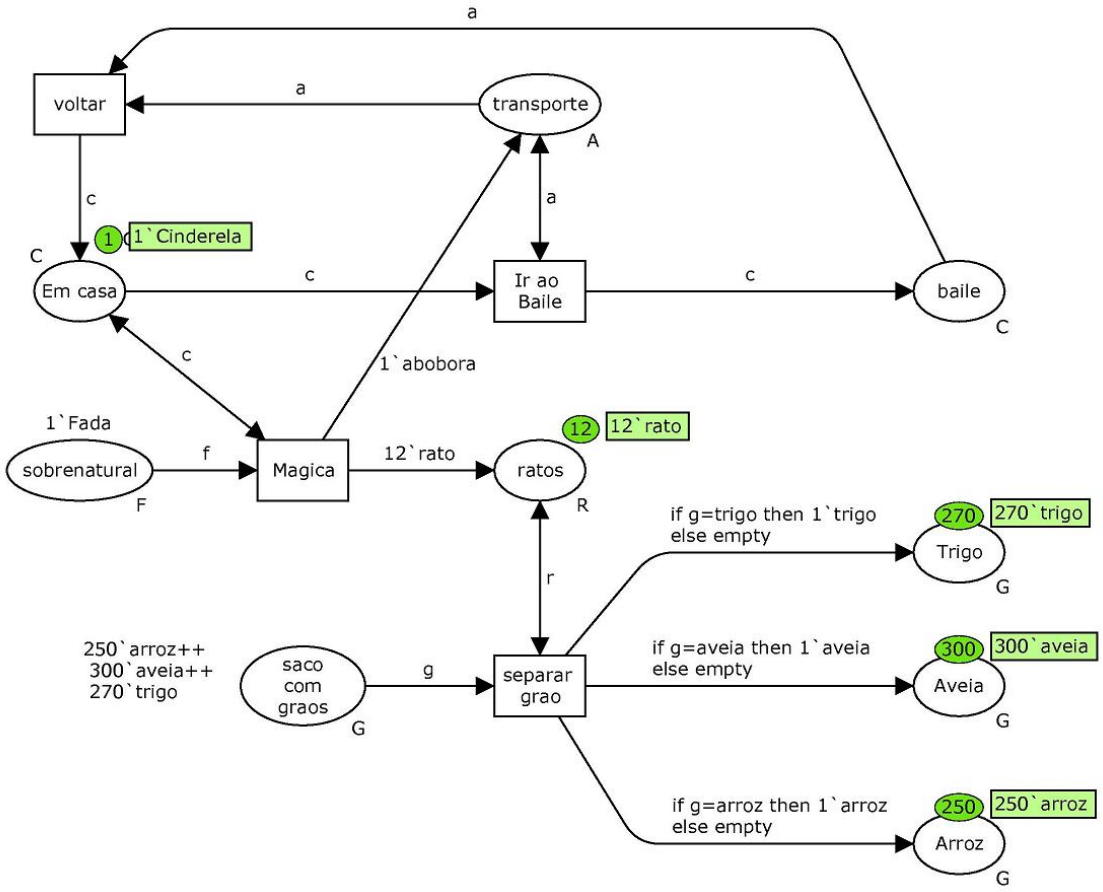

Figura 4. RPC que modela a ida de Cinderela ao baile: marcação final

É necessário salientar que o pequeno exemplo apresentado fornece um pouco de experiência para o desenvolvimento de modelos de sistemas reais, tais como sistemas de telecomunicações ou células de manufatura. O trabalho dos ratos, por exemplo, se assemelha à função de um roteador em uma rede de computadores ou de telecomunicações, ou ainda ao trabalho de um robô na separação de vários tipos de peças.

3. O CPN Tools e suas Funções Básicas

O CPN Tools é uma ferramenta destinada à síntese de redes de Petri coloridas e suas respectivas análises. Ele é uma ferramenta indispensável para o desenvolvimento de sistemas complexos em vários campos da engenharia, assim, o mesmo é largamente usado nas áreas de gerenciamento de processos e produtos, planejamento e controle de operações militares, controle de sistemas de produção, robôs, veículos e mísseis. A lista completa de suas aplicações reais pode ser encontrada na home-page do CPN Tools `http://www.daimi.au.dk/CPNTools/`. Atualmente, o CPN Tools pode ser instalado tanto na plataforma Windows, como na plataforma Linux. O CPN Tools é, na realidade, uma nova geração do antigo sistema Design-CPN.

Na área de telecomunicações, o CPN Tools é usado para especificação e verificação de protocolos, estimativas de taxas de transferências e QoS, projeto de redes e dispositivos de telecomunicações.

Anteriormente, um modelo era usado somente para estimação de características de sistemas durante o processo de desenvolvimento dos mesmos. No desenvolvimento orientado a modelo (model-driven development), um modelo inicial e simples é, sequencialmente transformado na especificação final do sistema. O processo de desenvolvimento constitui-se no processo de se modelar mais e mais detalhes do sistema real, até que o modelo se torne a especificação técnica necessária para a sua produção ou instalação. A vantagem desta abordagem é a possibilidade de se analisar o sistema em cada estágio do projeto. Isto permite um projeto bem próximo do ótimo porque, para sistemas reais mais complexos, a solução formal para otimização pode ser uma tarefa muito difícil e, em muitos casos, impossível.

As redes de Petri coloridas Hierárquicas temporizadas, modeladas no CPN Tools, permitem a descrição de um objeto arbitrário. Mais ainda, a linguagem das redes de Petri coloridas é direcionada à especificação de sistemas, principalmente aqueles que possuem interações complicadas entre componentes. O conceito de eventos assíncronos permite uma forma de descrição que preserva o paralelismo natural do comportamento do sistema. Isto é altamente conveniente para a implementação em processadores paralelos, ou arquiteturas de fluxo de dados de computadores.

3.1. Funções Básicas no CPN TOOLS

As funções básicas no CPN Tools consistem em:

- Criação (edição) de um modelo;
- Análise do comportamento do modelo através de simulação;
- Criação e análise do espaço de estados do modelo.

Para a criação de modelos, é fornecido um editor gráfico especial de redes de Petri coloridas. O editor permite que se desenhe uma rede de Petri na tela do computador, bem como se escreva os atributos dos elementos da rede e declarações adicionais escritas na linguagem CPN ML. O modelo pode ser constituído de várias páginas. Estas páginas são conectadas para fornecer a estrutura hierárquica.

Para modelos muito simples, a geração de seu espaço de estados completo (grafo de alcançabilidade) é possível. O CPN Tools fornece um relatório do espaço de estados gerado, com as conclusões das propriedades padrão das redes de Petri, tais como: limitação (boundedness) e vivacidade (liveness). Além disso, é fornecida uma linguagem especial, baseada em CPN ML, para a descrição de consultas sobre propriedades não padrão do espaço de estados em que o usuário esteja interessado. Infelizmente, para modelos mais complexos, o espaço de estados pode ser muito grande e sua criação ser impraticável.

A única forma de análise de modelos mais complexos é a simulação do comportamento dos mesmos. O CPN Tools fornece simulação passo-a-passo para depuração do modelo, bem como simulação automática com um certo número de passos. Simulação em grandes intervalos de tempo é uma forma de se fazer uma análise estatística do comportamento do modelo. Ela é util para a estimação de características de taxa de transferência e QoS.

4. Organização da interface do CPN Tools

Para o CPN Tools, foi implementado uma nova concepção de interação gráfica, baseada nas características do MS Open GL. Isto permite uma maior velocidade de entrada e

edição de modelos, usando caixas de ferramentas e menus sensíveis ao contexto.

4.1. Áreas da Janela Principal

Ao executar o CPN Tools, duas janelas aparecerão na tela do computador. A primeira delas é uma janela auxiliar de fundo preto que apresenta o histórico de mensagens referentes a subprocessos iniciados. Veja Figura 5.

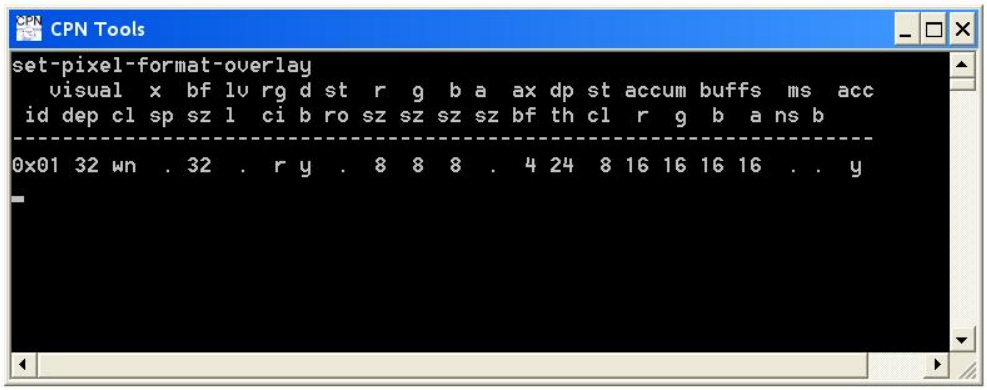

Figura 5. Janela auxiliar de apresentação de mensagens

A segunda janela é a janela principal do CPN Tools. Veja Figura 6.

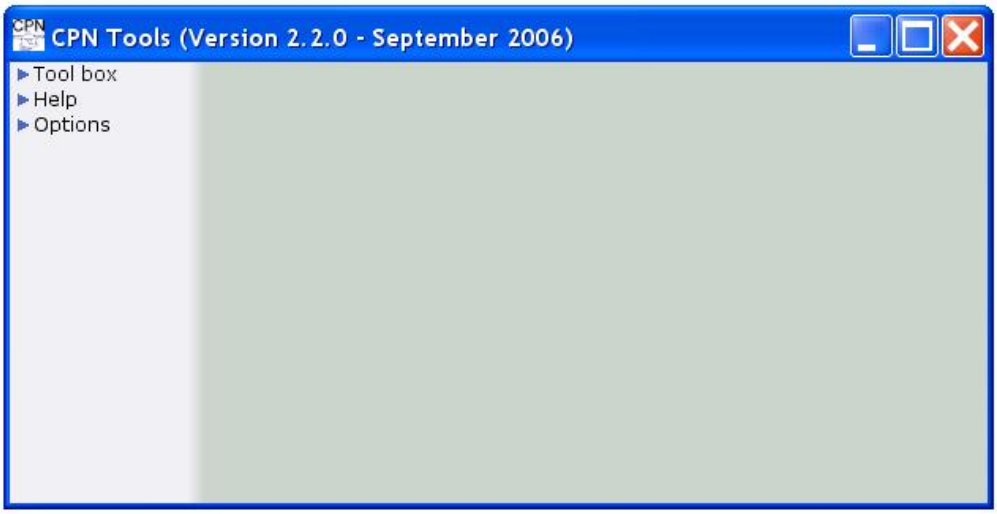

Figura 6. Janela Principal do CPN Tools

Esta janela contém duas áreas: área de de trabalho (de cor cinza escuro) e área de índice (de cor cinza claro). A área de índice apresenta três ferramentas básicas: `Tool box`, `Help` e `Options`. Abaixo delas é que serão colocadas as descrições das redes de Petri coloridas. Nesta área, os pequenos triângulos significam que o respectivo item pode ser aberto (subitens) quando se clica em cima deles. Veja Figura 7

Nas figuras 6 e 7 nenhuma rede foi carregada ou criada. Na área de trabalho serão visualizadas as páginas das redes. A forma de se abrir um palete de ferramentas é

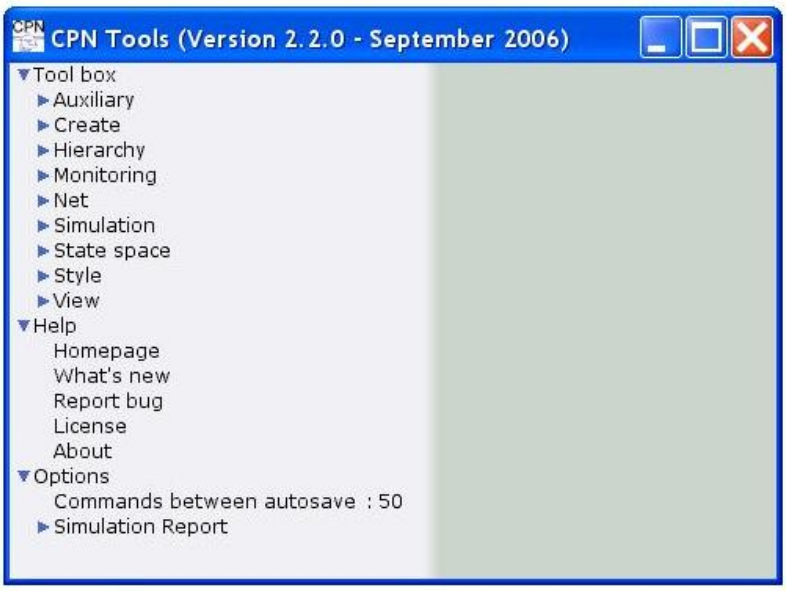

Figura 7. Clicando nos pequenos triângulos associados a `Tool box`, `Help` e `Options`

pressionar o botão esquerdo do mouse sobre o item desejado e arrastá-lo (com o botão pressionado) até a área de trabalho. Desta forma, todos os itens da área de índice podem ser abertos. Por exemplo, na Figura 8 a ferramenta `Create` foi aberta.

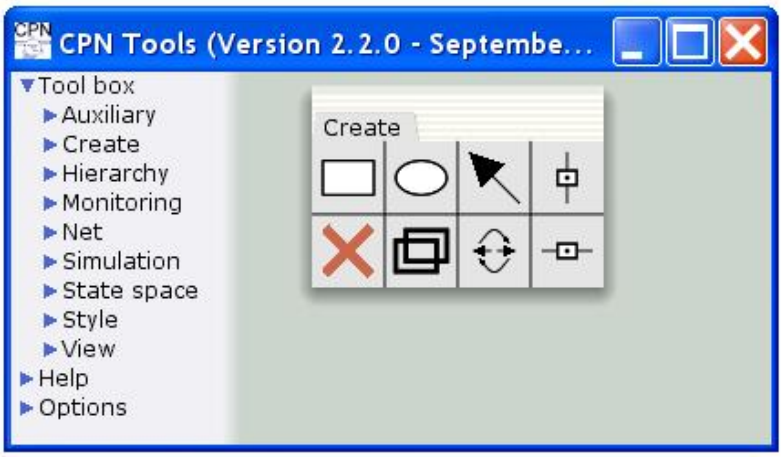

Figura 8. Palete da ferramenta Create arrastado para a área de trabalho

Para selecionar qualquer uma das ferramentas do palete, basta clicar em cima dele com o botão esquerdo do mouse. Neste instante, o cursor toma a forma da ferramenta selecionada. Veja na Figura 9 que `place` foi selecionado do palete `Create`.

Para abandonar a ferramenta selecionada, basta arrastá-la para seu ícone no palete e clicar novamente o mouse, ou então, pressionar a tecla `Esc`.

Algumas ferramentas do índice possuem opções especiais que podem ser mostradas e modificadas clicando no quadrado respectivo na área de índice. Por exemplo, se na ferramenta `Print` a opção `Black/White` for selecionada, a rede será impressa

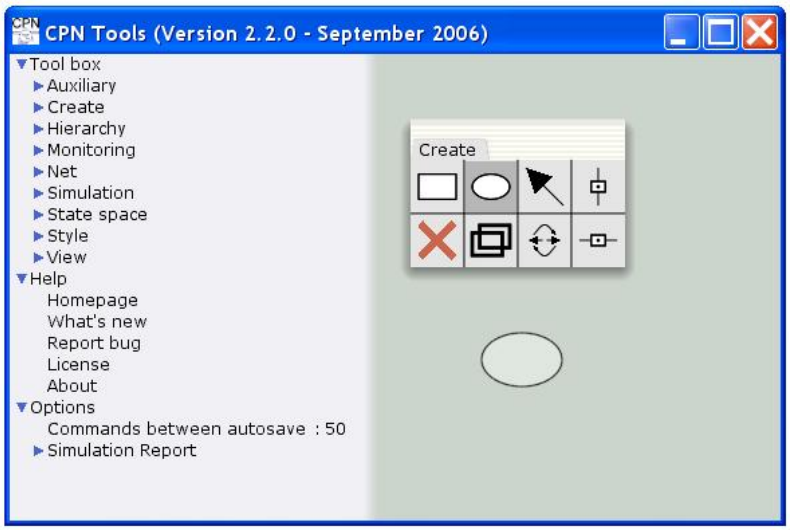

Figura 9. Seleção do item place **no palete** Create

em preto e branco. Esta opção pode ser selecionada também usando o palete `Net`, através da ferramenta de impressão do mesmo. Pressionando-o com o botão direito do mouse, aparecerá a opção `set options`. Soltando o botão do mouse em cima desta opção, aparecerá uma pequena janela e a seleção da opção desejada poderá ser feita. Veja as duas formas de seleção na Figura 10.

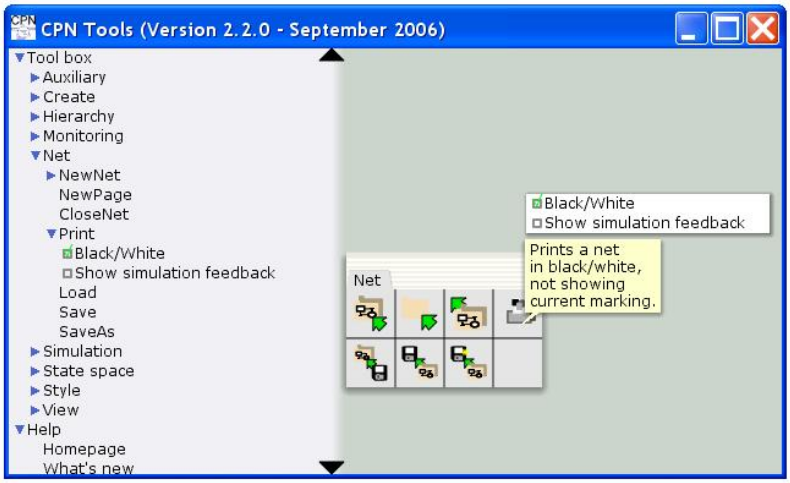

Figura 10. Opções de impressão de uma rede.

4.2. Menus sensíveis ao contexto

Para maior interação com o usuário, o CPN Tools fornece uma grande quantidade de menus sensíveis ao contexto, aparecendo na tela quando se pressiona o botão direito do mouse. Os menus possuem a forma de um círculo e são divididos em itens na forma de fatias de pizza. Para manter o menu na tela, é necessário continuar pressionando o botão direito do mouse, enquanto se arrasta o mouse para o item que se deseja selecionar. Na maioria das vezes, os menus sensíveis ao contexto também existem nos paletes. Veja figuras 11 e 12

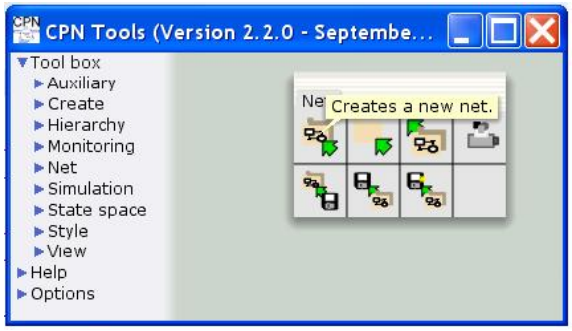

Figura 11. Selecionando item no palete.

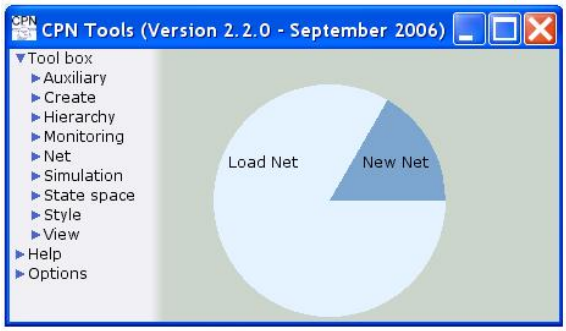

Figura 12. Selecionando item no menu sensível ao contexto.

Os menus sensíveis ao contexto tornam a interação mais natural e mais rápida. É necessário somente pressionar o botão direito do mouse em um dado objeto e escolher a ação desejada. No exemplo da Figura 13 é apresentada a ação de fechar o palete `Create`.

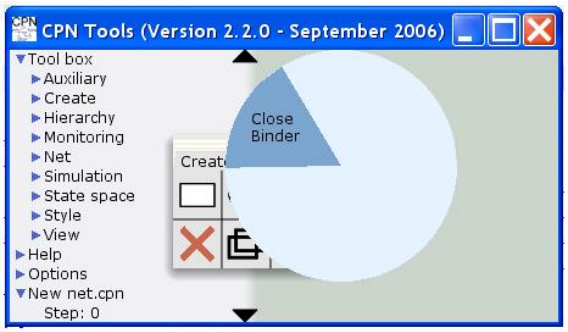

Figura 13. Ação de fechar palete Create **usando menu sensível ao contexto.**

4.3. Estrutura do modelo

No CPN Tools, cada modelo de rede de Petri colorida possui um nome. As descrições de cada modelo estão situadas na área de índice, abaixo dos itens padrão. Veja Figura 14, onde são apresentadas estas descrições após a criação de uma nova rede.

Cada rede no CPN Tools possui:

- New `net.cpn` - nome da nova rede, que pode ser modificado pelo usuário;
- `Step` - número de passos executados em uma simulação;

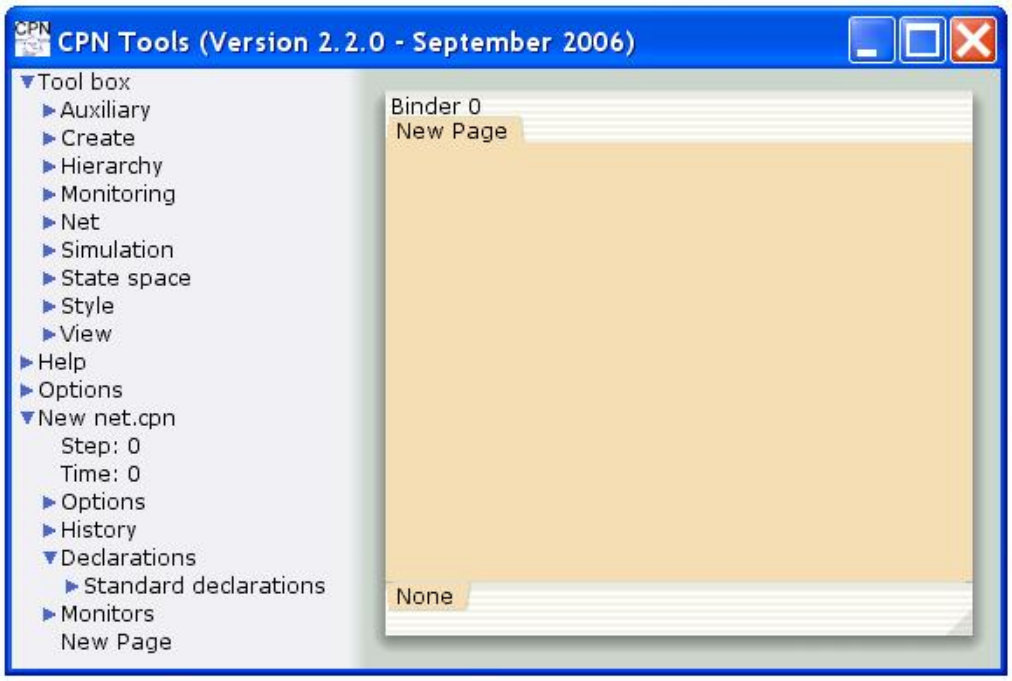

Figura 14. Descrições na área de índice e área de trabalho após a criação de uma nova rede.

- `Time` - tempo de simulação atual do modelo;
- `History` - lista de comandos realizados sobre a rede (ver subseção 4.1);
- `Declarations` - declarações de conjunto de cores, funções, variáveis e constantes;
- `Monitors` - conjunto de funções de monitoramento de simulações;
- `New page` - nome da página da rede. Se existir mais de uma página, então os nomes aparecerão em sequência. Os nomes das páginas poderão ser modificados pelo usuário.

Para abrir uma página de uma rede, é necessário arrastá-la da área de índice com o mouse para a área de trabalho. Para a criação de novas declarações e novas páginas são utilizados os menus sensíveis ao contexto. Veja Figura 15.

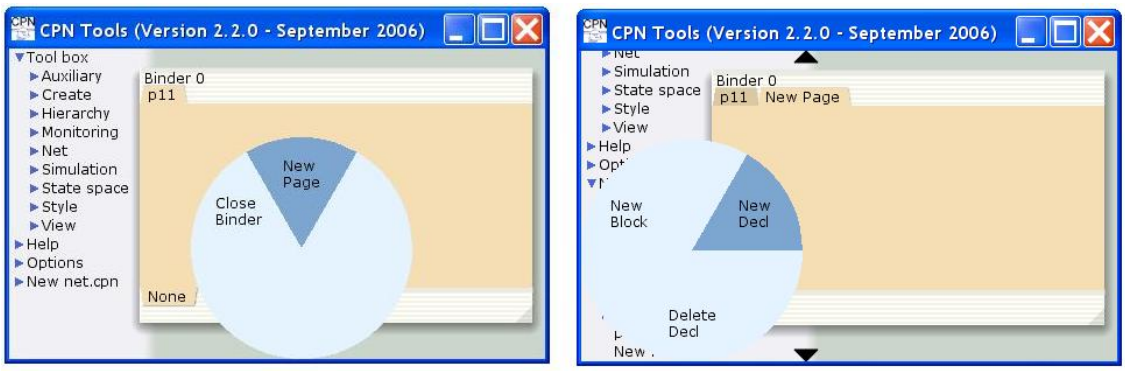

Figura 15. Exemplos de criação de uma nova página e de uma nova declaração.

4.4. Organização do sistema Help

O CPN Tools possui três tipos de Help:

- `bolha de conversação` (speech bubble);
- `offline help`;
- `online help`.

A bolha de conversação aparece na tela quando se mantém o cursor sobre um determinado item, por alguns segundos. Ela descreve o objeto apontado pelo cursor. Veja figuras 10 e 11, apresentadas na subseção 4.2;

Clicando no item `Help` na área de índice, um navegador é inicializado com a página de help do CPN Tools. Ela contém informações sobre o CPN Tools acompanhadas de exemplos de redes. Informações mais específicas e de atualizações podem ser encontradas na `homepage` do CPN Tools em Aarhus. Se necessário, o `help online` chama estas informações.

O `offline help`, é um programa armazenado no computador quando o CPN Tools é instalado. Para acessá-lo (se o seu sistema operacional é o Windows), basta ir no menu:

- `iniciar` → `Todos os programas` → `CPN Tols` → `Help` → `CPN Tools help [local]`.

O seu navegador irá abrir com uma página help igual àquela do `online help`, só que você não precisa estar conectado à internet.

4.5. Mensagens de status do CPN Tools

O CPN Tools fornece mensagens gráficas que refletem o estado atual do sistema. Existem vários tipos de mensagens, tais como:

- `Speech bubbles` (bolhas de conversação);
- `Status bubbles` (bolhas de status);
- `Auras`;
- `Mudança de ícone do cursor`.

A bolha de conversação é um retângulo amarelo que fornece informação sensitiva ao contexto. Algumas bolhas de conversação aparecem automaticamente, enquanto outras só aparecem quando o cursor pára por alguns segundos sobre um determinado objeto ou item. Por exemplo, movendo o cursor para uma declaração com erro de sintaxe, aparecerá uma bolha de conversação contendo uma mensagem de erro. Veja Figura 16

As bolhas de conversação são usadas para mostrar:

- Mensagens de erro durante a verificação de sintaxe;
- Mensagens de erro durante a simulação de uma rede;
- Dicas de ferramentas dos paletes;
- Informações detalhadas relativas às bolhas de status;
- Resultado da aplicação da ferramenta de avaliação ML;
- Caminho completo de onde uma rede foi salva. Para ver onde a rede foi salva, mova o cursor para cima do nome da rede na área de índice.

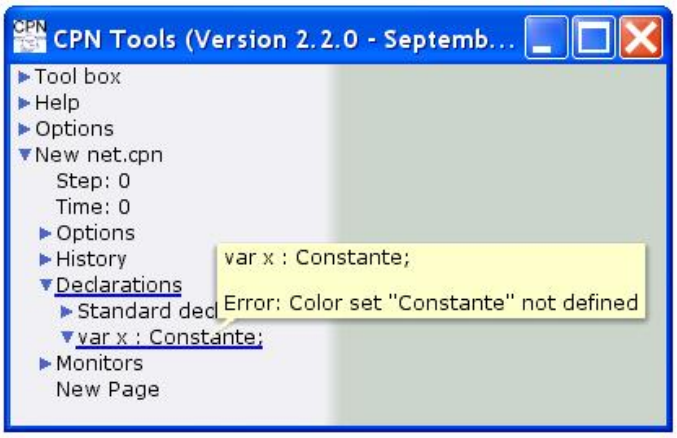

Figura 16. Exemplo de bolha de conversação com mensagem de erro.

As bolhas de status são bolhas coloridas que sempre aparecerão (quando for o caso) no canto inferior esquerdo da área de índice. Movendo o cursor para a bolha de status, aparecerá a bolha de conversação correspondente. As cores e o respectivo significado das bolhas de status são:

- Verde - indica que uma operação foi completada com sucesso;
- Vermelha - indica que um erro ocorreu quando da execução de uma operação;
- Roxa - indica que uma operação de longa duração (como uma simulação longa) está sendo executada.

Na Figura 17 é apresentada uma bolha de status verde, significando que a operação de `salvar arquivo` foi realizada com sucesso, e uma bolha de conversação especificando esta operação.

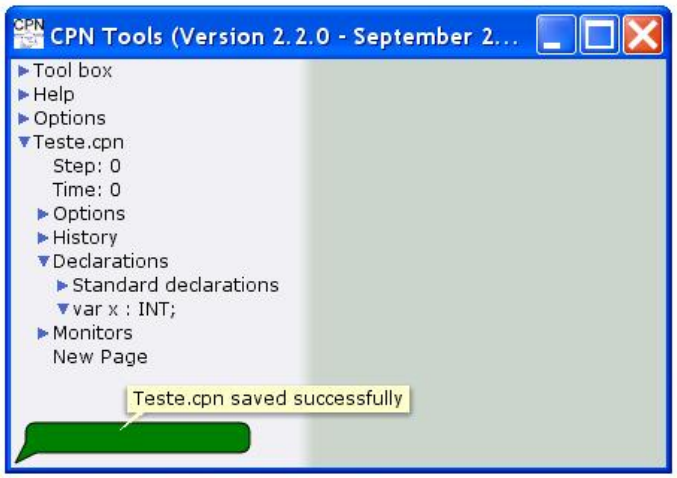

Figura 17. Bolha de status verde e bolha de conversação indicativa da operação realizada.

As mensagens do tipo `Aura` são codificadas por cores e são usadas para realçar objetos com características particulares, ou para indicar diferentes tipos de relações entre os objetos. Auras são associadas com lugares, transições, arcos, inscrições, declarações,

tabulação de páginas e entradas indexadas, tais como, nome de páginas e nome de redes. As Auras possuem as seguintes cores:

- Vermelho Clara - indica objetos com erro durante a verificação de sintaxe e durante a simulação de uma rede;
- Vermelho Escura - indica nomes de lugares ou transições duplicados;
- Verde - indica transições habilitadas;
- Azul escura - indica dependência entre declarações e outros elementos, tais como lugares, transições e páginas;
- Água marinha - indica que uma inscrição pertence a um determinado objeto;
- Laranja - indica que não se iniciou ainda a verificação de sintaxe de um determinado objeto;
- Amarela - indica que está em curso a verificação de sintaxe de um determinado objeto;
- Rosa - indica que lugares de fusão pertencem a um dado conjunto de fusão;
- Água marinha - quando se está trabalhando com redes hierárquicas, indica as relações entre porta/soquete e super/subpáginas.

Na Figura 18 são apresentadas algumas mensagens tipo `Aura`.

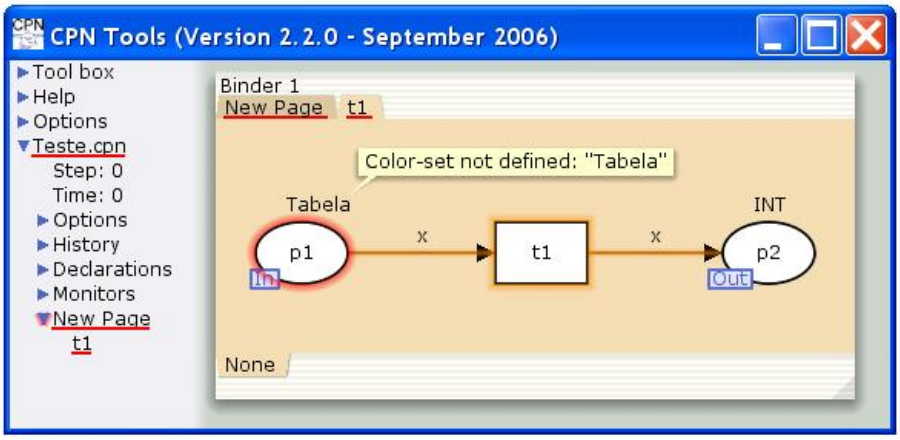

Figura 18. Mensagens do tipo aura.

O lugar `p1` possui uma aura vermelho clara, indicando que o conjunto de cores Tabela, associado a ele, não foi definido. A cor alaranjada contornando os dois arcos e a transição indica que a sintaxe destes elementos ainda não foi verificada.

A modificação do ícone do cursor indica que uma determinada ação irá ser iniciada ou já está sendo realizada, por exemplo:

- O cursor padrão é uma ponta de flecha;
- Uma pequena mão indica que um objeto (lugar, transição, etc) pode ser movido de posição;
- Uma cruz indica que é possível modificar a posição de toda a rede na área de trabalho;
- Uma seta de duas pontas indica que um objeto pode ser redimensionado. Passando o cursor padrão na borda de um objeto (um lugar, por exempo), o cursor se

modificará para uma seta de duas pontas e pressionando e arrastando o mesmo, o objeto aumentará ou diminuirá de tamanho, conforme a direção de arrastamento do cursor;

- pressionando o botão esquerdo do mouse sobre qualquer ferramenta de um palete, o cursor assumirá o formato da ferramenta;
- Para ferramentas de múltiplas fases, isto é, ferramentas que são aplicadas clicando em mais de um objeto, o cursor indicará qual será a próxima fase da ferramenta.

5. Tool Box do CPN Tools

O `Tool Box` fornece os seguintes paletes de ferramentas (veja Figura 19):

- Ferramentas `Auxiliary` - para melhoria da legibilidade da rede;
- Ferramentas `Create` - para edição de redes;
- Ferramentas `Hierarchy` - para criação de redes hierárquicas;
- Ferramentas `Monitoring` - para análise da simulação do comportamento das redes;
- Ferramentas `Net` - para operações com toda a rede;
- Ferramentas `Simulation` - para simulação do comportamento das redes;
- Ferramentas `State Space` - pra criação e análise do espaço de estados das redes;
- Ferramentas `Style` - para modificações na aparência das redes;
- Ferramentas `View` - para escolha de escalas de visualização e destaque de grupos de elementos.

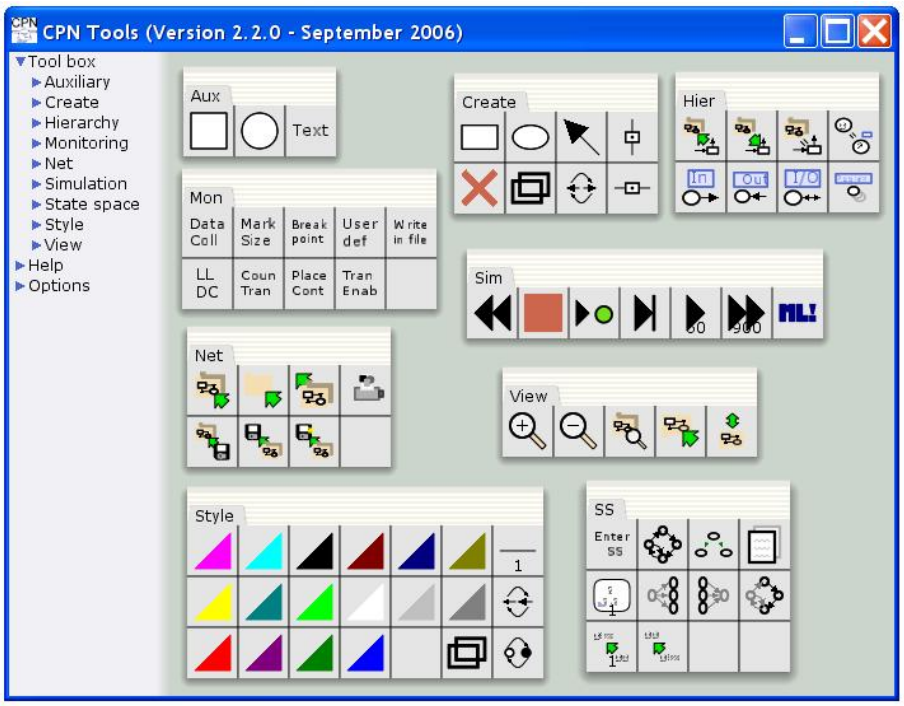

Figura 19. Paletes de ferramentas do Tool Box.

5.1. Ferramentas Net

Cada item do palete de ferramentas `Net` possui, respectivamente, o seguinte significado (veja Figura 20):

- `creates a new net` - criação de uma nova rede;
- `creates a new page` - criação de uma nova página;
- `closes a net` - fechar uma rede;
- `prints a net` - imprimir uma rede.
- `loads in a net` - abrir uma rede já existente;
- `saves a net` - salvar rede;
- `saves a net with a new name` - salvar uma rede com outro nome;

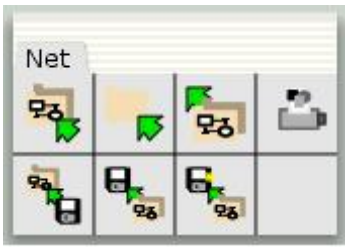

Figura 20. Palete de Ferramentas para criar, salvar, carregar e imprimir uma rede.

Para criar uma nova rede, deve-se iniciar com a ferramenta `creates a new net` e finalizar com a ferramenta `saves a net`. Para abrir uma rede já existente, deve-se iniciar com a ferramenta `loads in a net`. As redes são impressas para um arquivo de formato `.eps` (Extended Post Script) e podem ser inseridas como figuras em um arquivo de texto (por exemplo, documentos Latex ou MS Word). Novas páginas são criadas, principalmente, para redes hierárquicas.

5.2. Ferramentas Create

Cada item do palete de ferramentas `Create` possui, respectivamente, o seguinte significado (veja Figura 21):

- `creates a transition` - criação de uma nova transição;
- `creates a place` - criação de um novo lugar;
- `creates an arc` - criação de um arco ligando uma transição a um lugar ou viceversa;
- `creates a vertical magnetical guideline` - cria uma linha vertical que alinhará os elementos que estão próximos a essa linha;
- `deletes an element` - apaga um elemento da rede (lugar, transição ou arco) e suas inscrições associadas;
- `clones an element` - faz uma cópia de um elemento da rede, juntamente com suas inscrições associadas;
- `cycles between the possible directions or arc` - modifica a direção do arco ou cria um arco duplo;
- `creates a horizontal magnetical guideline` - cria uma linha horizontal que alinhará os elementos que estão próximos a essa linha;

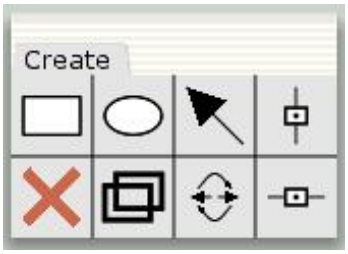

Figura 21. Palete de ferramentas para edição de uma rede.

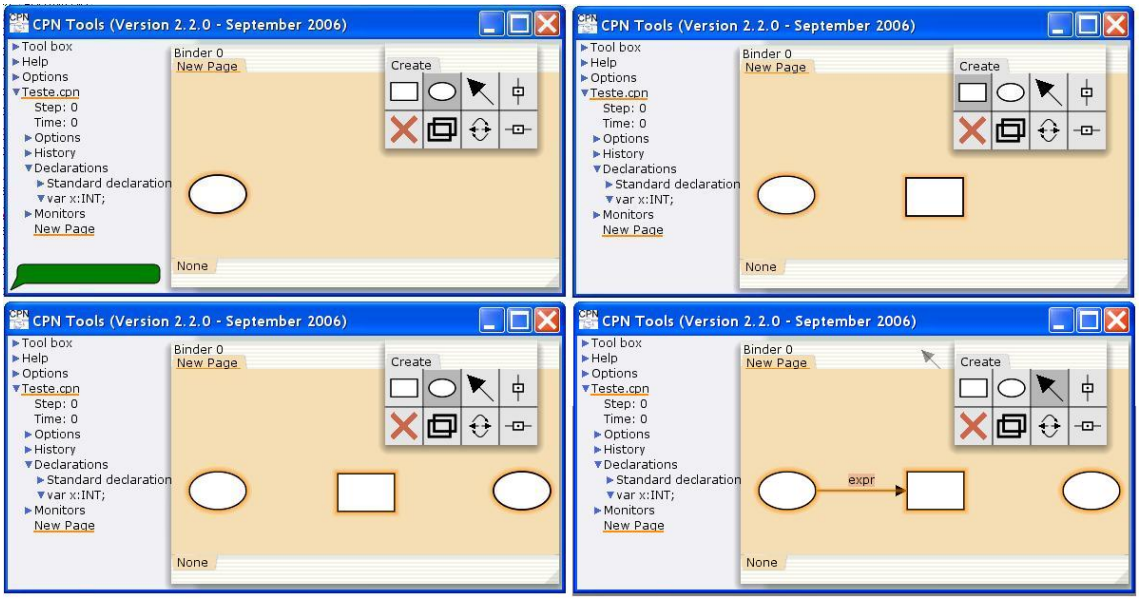

Figura 22. Processo de criação de uma rede simples.

Observe na Figura 22 a utilização do palete para a criação de uma rede com dois lugares e uma transição.

Note que na figura, todos os elementos possuem uma aura alaranjada. Isto significa que sua sintaxe ainda não está correta, pois faltam ainda os atributos associados aos elementos. A estrutura da rede apresentada na Figura 22 será completada criando-se um arco da transição para o lugar que se encontra à sua direita, colocando-se os atributos de cada elemento e uma marcação inicial no lugar denominado de `p1`. Veja Figura 23.

A rede agora está correta, ou seja, foi verificada sintaticamente e não aparece mais nenhuma mensagem de erro. Note que o lugar `p1` possui 5 fichas: 2 fichas de valor 14 e 3 fichas de valor 55.

Cada elemento da rede possui seus próprios atributos, assim, após a criação de cada elemento, é necessário criar também seus atributos. Na Figura 24 são apresentados os atributos de cada elemento.

Após clicar com o botão esquerdo do mouse sobre um determinado elemento, você poderá entrar com seus atributos, ou trocar de atributo usando a tecla `Tab`.

No exemplo apresentado na Figura 23, os atributos da transição não são usados. A marcação atual da rede aparece na cor verde próxima ao lugar correspondente.

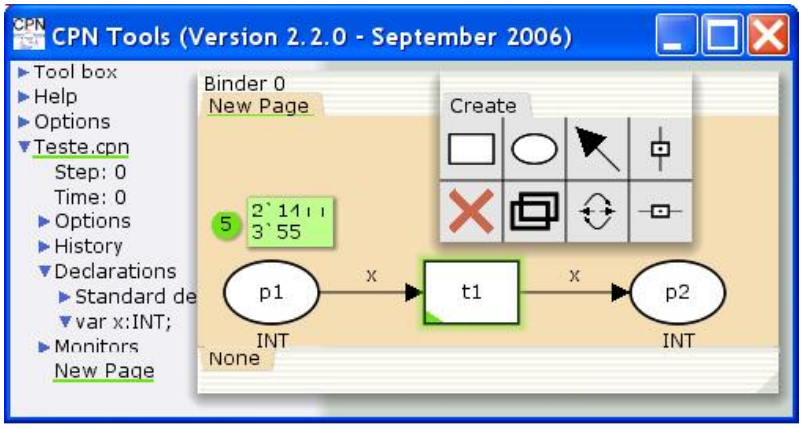

Figura 23. Exemplo de criação de uma rede simples.

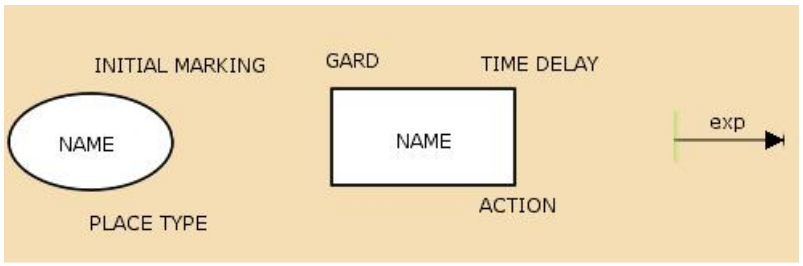

Figura 24. Atributos de cada elemento da rede.

5.3. Ferramentas Simulation

Cada item do palete de ferramentas `Simulation` possui, respectivamente, o seguinte significado (veja Figura 25):

Figura 25. Palete de ferramentas para simulação de uma rede.

- `Goes to the initial state` - retorna a marcação inicial da rede;
- `Stops an ongoing simulation` - pára a simulação em execução;
- `Executes a transition with a chosen binding` - fornece ao usuário a chance de escolher qual ligação deseja fazer para disparar a transição. Esta opção, serve para uma verificação mais específica de uma dada ação. Neste caso, pode-se escolher manualmente uma das fichas que possa ser ligada à inscrição do arco de entrada da transição;
- `Executes a transition` - dispara uma transição específica quando o cursor, com o formato do ícone associado a esta opção, está próximo ou sobre a mesma. É utilizado para verificação de erros por meio de simulação passo-a-passo. Clicando em cima de uma transição habilitada, a mesma é disparada, ou pode-se clicar em um espaço vazio da área de trabalho para que a escolha de qual transição habilitada disparar fique com o CPN Tools.;

- Executes the specified number of transitions, showing the intermediate markings - executa um número de passos (disparo de transições) equivalente ao número especificado no ícone. Entre cada disparo, a marcação atual é apresentada;
- Executes the specified number of transitions, without showing the intermediate markings - executa um número de passos (disparo de transições) equivalente ao número especificado no ícone e apresenta somente a marcação após o último disparo.
- Evaluates a text as ML code - avalia um texto como código ML. Usando este item, é forçada uma verificação de sintaxe em um texto de construção de linguagem.

5.4. Resumo de outras ferramentas

Outras ferramentas, tais como Auxiliary, Style e View, são apenas acessório.

As ferramentas Auxiliary permitem a criação de retângulos, círculos e comentários de textos que não possuem qualquer significado semântico, mas podem facilitar o entendimento da rede.

As ferramentas Style são usadas para enfatizar determinadas estruturas da rede com cores diferenciadas, linhas mais grossas, etc. Nenhuma destas ferramentas possui significado semântico.

As ferramentas View são usadas para se modificar a visão de uma página e seus elementos através de agrupamentos e aumento e diminuição de tamanho dos elementos.

As ferramentas Hierarchy e State Space serão apresentadas mais detalhadamente nas seções seguintes, dado que as mesmas são um pouco mais complicadas.

As ferramentas Hierarchy são usadas para a edição das estruturas hierárquicas de uma rede. O palete contém ferramentas para a estruturação *bottom-up* e *top-down* de uma rede.

As ferramentas State Space são usadas para calcular o espaço de estados (grafo de marcações) de uma rede; transferir estados entre o simulador e a ferramenta state space; e gerar os relatórios do espaço de estado construído.

6. Introdução ao CPN ML

O CPN Tools usa a linguagem de programação CPN ML para criar as declarações e as inscrições da rede. A CPN ML fornece declarações de conjunto de cores (tipos de dados), variáveis, funções e valores (constante). Cada lugar da rede possui associado um conjunto de cores como atributo, assim, ele só pode conter fichas daquele conjunto de cores específico. Variáveis e funções são usadas como inscrições de transições e arcos.

As declarações são parte integrante da rede e estão situadas na área de índice. Existem declarações de conjunto de cores padrão pré-definidas, tais como: E - elementary; INT - inteiro; BOOL - Booleano; STRING - caracter. Declarações do usuário podem ser adicionadas após as declarações padrão, usando-se o menu sensível ao contexto. Adicionalmente, o CPN Tools aceita declarações externas que podem ser carregadas a partir de um arquivo, para redes mais complexas.

Quando uma rede é criada ou carregada na área de trabalho, o CPN Tools faz a verificação automática da sintxe da rede. Através da cor da aura dos elementos e da cor da linha que sublinha as declarações, é possível saber em que estágio de verificação a rede se encontra. As cores aparecem na área de índice, sublinhando o nome da página à qual a cor pertence. Se a rede foi aberta em uma das abas da área de trabalho, então, o nome da rede aparece no topo da aba e estará sublinhada também. A aura de cor laranja indica que um elemento ainda não foi verificado. Quando uma rede é carregada, a verificação da sintaxe da mesma pode levar vários segundos, ou até minutos, para ser completada. Durante este tempo, os elementos da rede irão mudando a cor de sua aura de laranja para amarelo e, finalmente, perderão a aura se sua sintaxe estiver correta, caso contrário, ficarão com uma aura vermelha. Caso a aura de cor laranja permaneça por muito tempo, é possível que tenha ocorrido algum erro na hora da verificação, ou que o elemento esteja com algum erro.

As declarações são verificadas começando no topo. Se uma declaração depende de uma outra, ela apresentará um erro se a outra não tiver sido declarada anteriormente. Declarações com erro serão verificadas novamente, caso se faça alguma mudança em qualquer declaração. A declaração com erro será sublinhada por uma linha vermelha. O nome da rede à qual a declaração pertence, e o nome das páginas da rede relacionadas a esta declaração também serão sublinhadas com uma linha de cor vermelha.

Um aura vermelha em um determinado elemento, significa que o mesmo foi verificado, mas existe um erro. Nesse caso, uma bolha de conversação irá aparecer, especificando o erro encontrado. Outros elementos conectados àquele com erro, não serão verificados até que o erro seja retirado.

6.1. Conjuntos de Cores Simples

O CPN Tools fornece um conjunto de cores padrão simples, a saber: `Unit, Boolean, Integer, String, Enumerated, Index`.

Conjunto de cores UNIT

O conjunto de cores **unit** compreende um único elemento. Sua declaração possui a seguinte sintaxe:

```
colset name = unit [with new_unit];
```

Se a opção (`new_unit`) não for utilizada, o nome da ficha coincide com o nome do conjunto de cores. No exemplo da Cinderela os seguintes conjuntos de cores `unit` foram utilizados:

```
colset A = unit with abobora;
colset C = unit with Cinderela;
colset R = unit with rato;
colset F = unit with Fada;
```

Conjunto de cores BOOLEAN

O conjunto de cores **boolean** compreende dois valores `true` e `false`. Sua declaração possui a seguinte sintaxe:

```
colset name = bool [with new_false, new_true];
```

As opções (`new_false, new_true`) permitem novos nomes para `false` e `true`. Por exemplo, `nao` e `sim`.

```
colset Pergunta = bool with (nao, sim);
```

As seguintes operações podem ser aplicadas às variáveis booleanas:

not b	negação do valor booleano de b;
b1 **andalso** b2	conjunção booleana `and`;
b1 **orelse** b2	disjunção booleana `or`.

Conjunto de cores INTEGER

O conjunto de cores **Integer** compreende os valores inteiros. Sua declaração possui a seguinte sintaxe:

```
colset name = int with int-exp1 ..  int-exp2;
```

A opção `with` permite restringir o conjunto de cores inteiro pelo intervalo determinado pelas duas expressões `int-exp1` e `int-exp2`:

```
colset Duzia = int with 1..12;
```

As seguintes operações podem ser aplicadas às variáveis inteiras:

`+, -, div, mod, abs, Int.min, Int.max.`

Conjunto de cores STRING

O conjunto de cores **String** é especificado por uma sequência de caracteres ASCII entre aspas. Sua declaração possui a seguinte sintaxe:

```
colset name = string [with string-exp1..string-exp2
[and int-exp1..int-exp2]];
```

A opção `with` especifica o intervalo de caracteres válidos, por exemplo:

```
colset minusculas = with ``a''..``z'';
```

As seguintes operações podem ser aplicadas às variáveis string:

^ (concatenação), `String.size, sbstring`

Conjunto de cores ENUMERATED

O conjunto de cores **Enumerated** explicita todos os identificadores na sua declaração. Sua sintaxe é assim definida:

```
colset name = with id0 | id1 | ...  | idn;
```

No exemplo seguinte são enumerados os dias do final de semana:

```
colset FimdeSemana = with sabado | domingo;
```

Conjunto de cores INDEXED

Os conjuntos de cores **Indexed** são sequências de valores que contêm um inteiro e um índice especificador. Sua declaração possui a seguinte sintaxe:

```
colset name = index id with int-exp1 ..
int-exp2;
```

Valores indexados possuem o seguinte formato: `id i` ou `id(i)`, em que

`i` é um inteiro e `int-exp1` $\leq$ `i` $\leq$ `int-exp2`.

No exemplo a seguir são indexados os dias da semana:

```
colset DiaSemana = index dia with 1..7;
```

e assim, pode-se associar o domingo a dia(1), a segunda a dia(2), e assim por diante.

6.2. Conjuntos de Cores Compostos

Os conjuntos de cores compostos são formados pela combinação de conjuntos de cores simples. A linguagem CPN ML fornece os seguintes conjuntos de cores compostos: `product, record, union, list, subset, alias`.

Os conjuntos `product` e `record` são conjuntos de dados formados pelo produto cartesiano dos elementos de outros conjuntos. A única diferença entre eles é que os componentes do conjunto de cores `product` não são nomeados, enquanto os componentes do conjunto de cores `record` são nomeados. Existe uma grande semelhança com o tipo de dado `record` na linguagem de programação Pascal ou com o tipo de dado `structure` na linguagem C.

Conjunto de cores PRODUCT

O conjunto de cores **product** possui a seguinte sintaxe:

```
colset name = product name1 * name2 * ...  *
namen;
```

Os valores neste conjunto de cores possuem a seguinte forma:

`(v1, v2, ..., vn)` em que `vi` possui o tipo `namei` para 1$\leq$`i`$\leq$`n`;

Para extrair o i-ésimo elemento de um `product`, utiliza-se a seguinte operação:

```
#i name;
```

Exemplo de declaração de um `product` é apresentado a seguir:

```
colset abscissa = INT;
colset ordenada = INT;
colset Ponto = product abscissa * ordenada;
var p :  Ponto;
```

Conjunto de cores RECORD

O conjunto de cores **record** possui a seguinte sintaxe:

```
colset name = record id1:name1 * id2:name2 * ...
* idn:namen;
```

Os valores neste conjunto de cores possuem a seguinte forma:

`(id1=v1, id2=v2, ..., idn=vn)` em que `vi` possui o tipo `namei` para $1 \leq i \leq n$;

Para extrair o i-ésimo elemento de um `record`, utiliza-se a seguinte operação:

```
#idi name;
```

Considere o mesmo exemplo apresentado para o conjunto de cores `product`, agora modificado para o tipo `record`:

```
colset abscissa = INT;
colset ordenada = INT;
colset Ponto1 = record x:abscissa * y:ordenada;
var p1 :  Ponto1;
```

Veja que um ponto em um plano cartesiano pode ser representado tanto por um conjunto de cores `product (Ponto)`, como por um conjunto de cores `record (Ponto1)`. Por exemplo, para o conjunto `Ponto`, a variável `p = (2,5)`. Para o conjunto `Ponto1`, a variável `p1 = (x=2, y=5)`.

Para extrair um elemento específico do conjunto `Ponto`, (por exemplo o segundo elemento) escreve-se: `#2 p`. Esta operação retorna o valor `5`.

Para extrair um elemento específico do conjunto `Ponto1`, (por exemplo o segundo elemento) escreve-se: `#y p1`. Esta operação retorna o valor `5`.

Conjunto de cores UNION

O conjunto de cores **union** produz a união de dois ou mais conjuntos de cores previamente definidos. Normalmente, em um lugar só pode existir fichas pertencentes a um único conjunto de cores. O conjunto de cores **union** ajuda a superar esta limitação, pois é possível associar diferentes conjuntos de cores a um único lugar. O conjunto de cores **union** possui a seguinte sintaxe:

```
colset name = union id1[:name1] + id2[:name2] +
...  + idn[:namen];
```

Se `namei` é omitido, então, `idi` é tratado como um novo valor (constante) e pode ser referido simplesmente como `idi`. Operadores simples podem ser usados para recuperar valores de conjuntos de cores pertencentes à união. Veja exemplo na Figura 26:

Na figura, têm-se duas declarações do tipo `union`, os conjuntos `Uniao` e `U`. O primeiro faz a união do conjunto de cores `Small`, que é do tipo `int`, com o conjunto de cores `Dados`, que é do tipo `string`. O segundo faz a união do conjunto de cores `Small` com um novo valor denominado `Ack`.

Desta forma, é possível colocar no lugar `p2` duas fichas de tipos distintos. Uma do tipo inteiro `Int(5)` e outra do tipo string `Dado("A")`. Note que a variável `par` tanto pode assumir o valor inteiro como o valor string, indistintamente.

No lugar `p4`, na rede da figura, existe uma ficha do tipo inteiro `I(4)`. Ao disparar `t2`, essa ficha será retirada e uma nova ficha de valor `Ack` será colocada em `p2`.

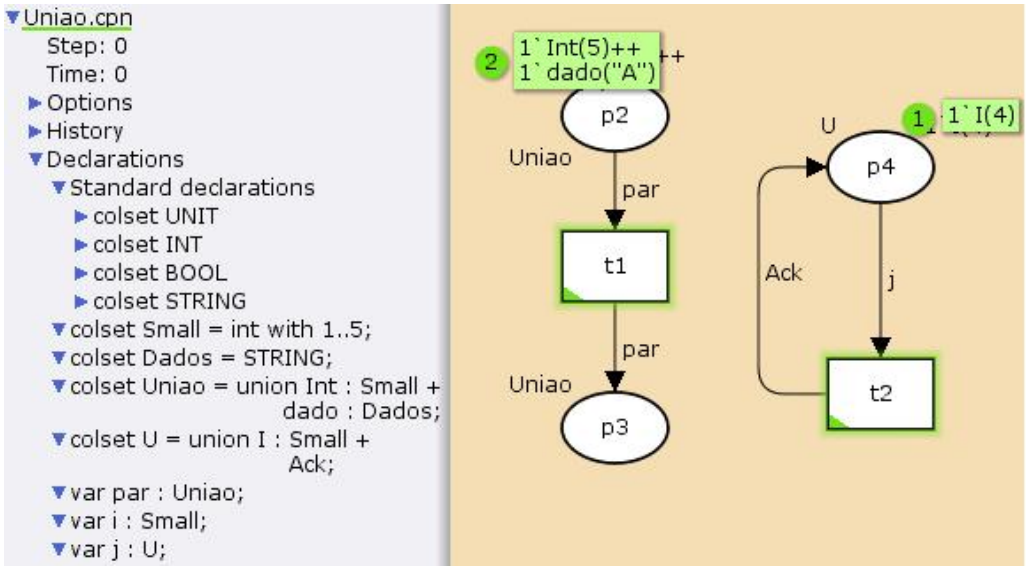

Figura 26. Exemplo de declaração do conjunto de cores `union`

Conjunto de cores LIST

O conjunto de cores `list` possui tamanho variável e se constitui de uma sequência de elementos de um mesmo conjunto de cores previamente definido. Funções padrão permitem acessar o primeiro e o último elemento de uma lista. Para acessar elementos do interior da lista, funções recursivas têm que ser usadas.

O conjunto de cores **list** possui a seguinte sintaxe:

```
colset name = list name0 [with int-exp1 ..
int-exp2];
```

A cláusula `with` especifica o menor e o maior tamanho da lista. Os elementos de uma lista possuem a seguinte forma:

`[v1, v2, ..., vn]`, em que `vi` é do tipo `name0` para `i = 1..n`.

A seguir são listadas algumas das operações que se aplicam às listas:

`nil`	lista vazia (o mesmo que [])
`e::l`	coloca o elemento `e` como cabeça da lista `l`
`l1^^l2`	concatena as duas listas `l1` e `l2`
`hd l`	`head`, primeiro elemento da lista `l`
`tl l`	Toda a lista `l`, exceto o primeiro elemento
`length l`	Retorna o tamanho da lista `l`
`rev l`	Retorna uma lista inversa à lista `l`
`map f l`	Usa a função `f` em todos os elementos da lista `l` e retorna uma lista com todos os resultados
`mem l x`	Retorna verdadeiro se o elemento `x` pertence à lista `l`

`List.nth(l,n)`	Retorna o `n`-ésimo elemento da lista `l`, em que $0 \leq$ `n` $<$ `length l`
`List.take(l,n)`	Retorna os primeiros `n` elementos da lista `l`
`List.drop(l,n)`	Retorna o que resta na lista após retirar os primeiros `n` elementos da lista `l`
`List.exists p l`	Retorna verdadeiro se p é verdadeiro para algum elemento da lista `l`
`List.null l`	Retorna verdadeiro se a lista `l` é vazia

Normalmente, no disparo de uma transição em uma rede de Petri colorida qualquer, retiram-se fichas dos lugares de entrada da transição de forma aleatória, ou seja, a ligação entre as variáveis dos arcos e as fichas é feita de forma aleatória, desde que essa ligação habilite a transição. Entretanto, em alguns setores do conhecimento, tais como telecomunicações, linha de montagem, etc., existe um ordenamento de chegada e saída de informação ou produtos. Existe, então, a necessidade de políticas de prioridade. Algumas destas políticas são, por exemplo, a `FIFO` (First In First Out), ou primeira que chega é a primeira que sai; e a `LIFO` (Last In First Out), ou última que chega é a primeira que sai.

Na figura 27 é apresentado um exemplo de FIFO. Primeiro, valores inteiros são retirados do lugar `p1` de forma aleatória e armazenados por ordem de chegada em uma lista no lugar `p2` (`1`[9,7,5,3]`). Quando `t2` estiver habilitada, o disparo da mesma vai retirar sempre a ficha que chegou primeiro ao lugar `p2`. Na figura, após `t2` disparar duas vezes, as fichas `9` e `7` são retiradas consecutivamente de `p2`.

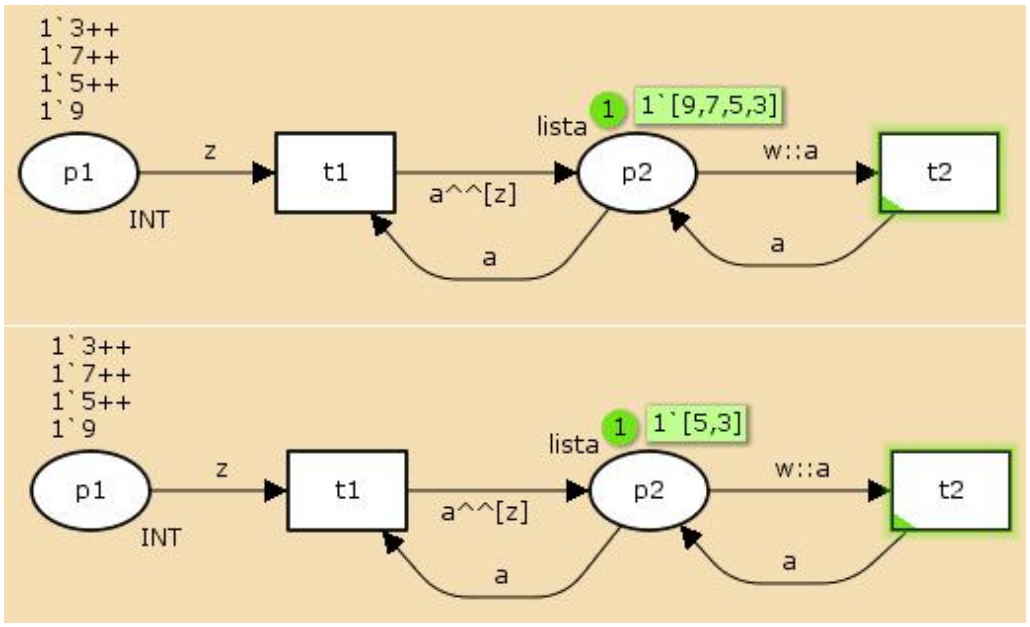

Figura 27. Exemplo de modelagem de uma FIFO

Conjunto de cores ALIAS

O conjunto de cores **alias** possui a seguinte sintaxe:

```
colset name = name0;
```

Este conjunto foi introduzido para que se possa usar um nome diferente em um conjunto de cores definido previamente. Veja o exemplo:

```
colset Ponto1 = record x:abscissa * y:ordenada;
colset Novoponto = Ponto1;
```

Nesse caso, o conjunto `Novoponto` pode ser utilizado no lugar de `Ponto1`, visto que os mesmos são iguais.

6.3. Declaração de Variáveis e Constantes

Uma variável é um identificador cujo valor pode ser modificado durante a execução de um modelo de rede de Petri colorida. As variáveis são utilizadas nas inscrições associadas aos elementos da rede.

A declaração **variavel** possui a seguinte sintaxe:

```
var id1, id2, ..., idn :  cs_name;
```

em que `idi` é um identificador e `cs_name` é o nome de um conjunto de cores previamente definido. Por exemplo:

```
var p :  Ponto;
var p1, p2 :  Ponto1;
```

A declaração **value** associa um valor a um identificador, o qual funcionará como uma constante, e possui a seguinte sintaxe:

```
val id1 = exp;
```

em que `id` é um identificador e `exp` é uma expressão CPN ML. A expressão representa o valor a ser associado com o identificador. Por exemplo:

```
val Resposta = "sim";
val NotaMaxima = 10;
```

6.4. Funções

As funções no CPN ML implementam as estruturas de controle padrão de uma linguagem de programação, tais como os operadores `if` e `case`. No entanto, como a ML é uma `linguagem de programação funcional` [Goldberg 1996], o seu maior poder é revelado através de funções recursivas. A declaração de uma função recursiva possui a seguinte sintaxe:

```
fun id pat1 = exp1
     | id pat2 = exp2
     | ...
     | id patn = expn;
```

em que `pat1, pat2, ..., patn` são padrões e `exp1, exp2, ..., expn` são expressões do mesmo tipo. Esta declaração significa que, no caso dos argumentos atuais satisfazerem o padrão `pati`, então o valor da função é calculado como `expi`. Pode-se calcular o fatorial de um número inteiro usando recursão. Veja exemplo a seguir:

```
fun fact ( 0 ) = 1
     | fact ( i ) = i * fact( i-1 );
```

As estruturas de controle **if-then-else** e **case** estão disponíveis para a descrição de funções:

```
if bool-esp then exp1 else exp2;
```

em que `exp1` e `exp2` são do mesmo tipo.

```
case exp of
     pat1 => exp1
     | pat2 => exp2
     | ...
     | patn => expn;
```

em que `exp1, exp2, ..., expn` são todas do mesmo tipo. O significado desta função é o mesmo que em outras linguagens de programação.

Por exemplo, uma função que retorna o sinal de um número pode ser escrita da seguinte forma:

```
fun sign (x) = if x > 0 then 1
     else if x < 0 then ~1
     else 0;
```

lembrando que o operador unário negativo no CPN Tools é o `(~)`. Assim, para representar o valor `-10`, escreve-se `~10`.

A função que retorna o nome do sinal do número pode ser escrita da seguinte forma:

```
fun namesign (x) =
     case sign(x) of
          1 => ``positivo''
          | ~1 => ``negativo''
          | _ => ``zero'';
```

O símbolo `_` na última linha da função indica que a palavra `zero` será escolhida dentre todas as opções checadas pela expressão `sign(x)`.

A construção **let** permite a declaração de variáveis locais na definição de uma função:

```
let
     val pat1 = exp1
     val pat2 = exp2
     ...
     val patn = expn

in
     exp
```

```
end;
```

Por exemplo, para apresentar em metros uma medida que foi calculada em milímetros, pode-se implementar a seguinte função:

```
fun metros (x) =
      let
            val mM = 1000
      in
            x div mM
      end;
```

6.5. Funções Aleatórias

As funções aleatórias (randômicas) fornecem as facilidades para a modelagem de características estatísticas. Por exemplo, elas permitem a descrição da intensidade de tráfico, ou da taxa de mensagens enviadas em um sistema de telecomunicações. O CPN Tools permite algumas formas de descrição de escolha aleatória: são elas:

- variáveis livres;
- função `ran`;
- Funções de distribuição aleatória especiais.

Variáveis livres são variáveis associadas aos arcos de saída de uma transição e que não possuem nenhuma ligação nem com os arcos de entrada da transição, nem com sua guarda. Veja Figura 28:

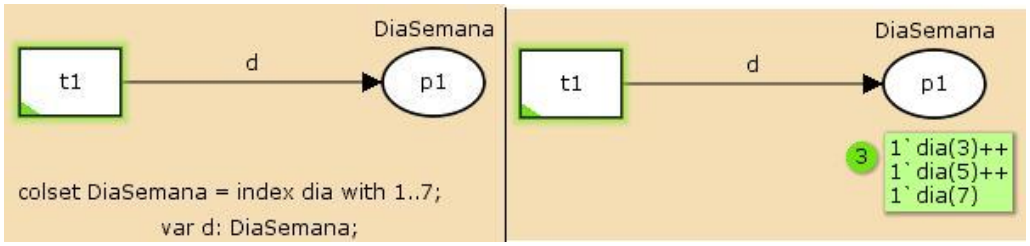

Figura 28. Transição fonte com arco de saída associado à variável livre `d`

A função `ran` gera um valor aleatório para grandes conjuntos de cores. No exemplo da Figura 29, a função `ranRange()` gera um valor aleatório entre 1..10000:

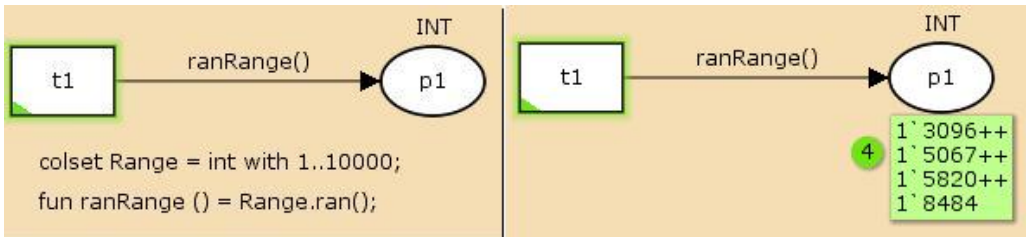

Figura 29. Transição fonte com arco de saída associado à função `ranRange()`

O CPN Tools fornece ainda uma série de funções de distribuição especiais, tais como as bem conhecidas `bernoulli`, `binomial`, `chisq`, `erlang`, `exponential`, `normal`, `poisson`, `student`, `uniform` (discreta e contínua). A seguir, apresenta-se a sintaxe destas funções no CPN Tools:

- `fun Bernoulli() = bernoulli(p);`, em que *p* é real e $0.0 \leq p \leq 1.0$. Retorna uma distribuição de Bernoulli com média *p* e variância $p(1-p)$;

- `fun Binomial() = binomial(n, p);`, em que *n* é inteiro, *p* é real, $n \geq 1$ e $0.0 \leq p \leq 1.0$. Retorna uma distribuição binomial com média *np* e variância $np(1-p)$;

- `fun Chisq() = chisq(n);`, em que *n* é inteiro e $n \geq 1$. Retorna uma distribuição Chisq com média *n* e variância *2n*;

- `fun Erlang() = erlang(n, r);`, em que *n* é inteiro, *r* é real, $n \geq 1$ e $r > 0,0$. Retorna uma distribuição Erlang com média $\frac{n}{r}$ e variância $\frac{n}{r^2}$;

- `fun Exponencial() = exponential(r);`, em que *r* é real e $r > 0,0$. Retorna uma distribuição exponencial de média $\frac{1}{r}$ e variância $\frac{1}{r^2}$;

- `fun Normal() = normal(n, v);`, em que *n* e *v* são reais. Retorna uma distribuição normal com média *n* e variância *v*;

- `fun Poisson() = poisson(m);`, em que *m* é real e $m > 0,0$. Retorna uma distribuição de Poisson com média e variância iguais a *m*;

- `fun Student() = student(n);`, em que $n > 1$ e *n* é inteiro. Retorna uma distribuição Student (também chamada de distribuição-t) com média igual a zero e variância $\frac{1}{n-2}$;

- `fun Uniform() = uniform(a, b);`, em que *a* e *b* são reais e $a \leq b$. Retorna uma distribuição uniforme com média $\frac{a+b}{2}$ e variância $\frac{(b-a)^2}{12}$.

6.6. Multi-conjuntos (Multi-sets)

Os multi-conjuntos são largamente utilizados no CPN Tools para representação das marcações nos lugares e outros propósitos. Multi-conjuntos são também denominados de `bolsas (bags)` e, diferentemente dos conjuntos, podem possuir mais de uma cópia de um mesmo elemento.

O operador `(`) -- acento grave --` é o construtor do multi-conjunto. Por exemplo, 7 `4 é o multi-conjunto com sete cópias da cor 4. Um multi-conjunto possui a seguinte sintaxe:

- `i`c` – o inteiro `i` é não negativo

O inteiro `i` tem que ser não negativo, pois determina a multiplicidade de cada elemento do conjunto de cores. O operador de multi-conjunto, combinado com os operadores de adição `(++)` e subtração `(--)` dos multi-conjuntos fornecem um método simples para a especificação de multi-conjuntos. Por exemplo, na Figura 3, descrito na Seção 2.2, o lugar `saco com graos` apresenta a seguinte marcação inicial:

- `250`arroz ++ 300`aveia ++ 270`trigo.

Isto significa que o referido lugar contém 250 grãos de arroz, 300 de aveia e 270 de trigo. Não esqueça que o operador de construção do multi-conjunto é o acento grave (`) e não o apóstrofo (‘).

As seguintes operações, constantes e funções estão disponíveis para multi-conjuntos:

`empty`	a constante `empty` constroi um multi-conjunto vazio que é idêntico a todo tipo de multi-conjunto;
`mas1 == ms2`	igualdade de multi-conjuntos;
`mas1 <><> ms2`	desigualdade de multi-conjuntos;
`ms1 >> ms2`	`ms1` maior que `ms2`;
`ms1 >>== ms2`	`ms1` maior ou igual a `ms2`;
`ms1 << ms2`	`ms1` menor que `ms2`;
`ms1 <<== ms2`	`ms1` menor ou igual a `ms2`;
`ms1 ++ ms2`	adição em multi-conjuntos;
`ms1 -- ms2`	subtração em multi-conjuntos;
`i ** ms`	multiplicação escalar;
`size ms`	cardinalidade do multi-conjunto ms;
`random ms`	retorna uma cor de ms pseudo aleatória;
`cf(c,ms)`	retorna o número de cópias da cor `c` em `ms`;
`filter p ms`	toma um predicado `p` e um multi-conjunto `ms` e produz um outro multi-conjunto com todas as cores de `ms` que satisfazem ao predicado `p`.

Sejam os conjuntos de cores `m1` e `m2`, como a seguir:

```
m1 = 3`7 ++ 5`2 ++ 8`14;
m2 = 1`7 ++ 2`2 ++ 5`14;
```

assim,

```
m1 ++ m2 = 4`7 ++ 7`2 ++ 13`14;
m1 -- m2 = 2`7 ++ 3`2 ++ 3`14;
m1 >> m2         é verdadeiro;
size m1 = 16;
cf(14,m2) = 5.
```

No CPN Tools, qualquer marcação (inicial ou corrente) de um lugar, é representada por um multi-conjunto do conjunto de cores associado ao lugar. No instante do disparo de uma transição, é feita uma escolha aleatória de uma ficha pela variável associada ao arco de saída do lugar.

6.7. Multi-conjuntos com Restrições de Tempo

Multi-conjuntos com restrições de tempo são usados no CPN Tools para representação de retardos de tempo no modelo de uma rede de Petri colorida. A declaração de um conjunto de cores com restrições de tempo deve ser acompanhada com o modificador `timed`. Os operadores @, @+, e @@+ são usados para adicionar temporização às cores (fichas). Associar um tempo `t` a uma cor `c`, significa adicionar um valor de tempo a essa cor que

é igual ao `tempo atual do modelo + t`. As seguintes operações são válidas para multi-conjuntos com restrições de tempo:

`c @ t`	associa um tempo `t` (com o tipo `Time.time`) à cor `c`;
`ms @+ i`	adiciona o tempo inteiro `i` a todas as cores do multi-conjunto `ms`;
`tms1 +++ tms2`	adição de multi-conjuntos temporizados.

As seguintes declarações são, então, válidas:

```
colset Tint = int timed;
var t1, t2 :  Tint;
t1 = 1`5@150;
t2 = 1`3@40;
```

isto significa que o conjunto de cores `Tint` é temporizado, as variáveis `t1` e `t2` são do tipo `Tint` e podem assumir cores do tipo inteiro temporizadas. No exemplo, o CPN Tools só poderá utilizar a ficha `1`3@40` quando o tempo do relógio global do modelo for igual a 40, ou seja, antes desse tempo, a ficha não estará disponível para habilitação ou disparo de nenhuma transição.

7. A Linguagem de Descrição do Modelo

No CPN Tools, cada elemento da rede de Petri possui atributos descritos na linguagem CPN ML. Usando o palete `Create` desenha-se um elemento em uma página do modelo. Após desenhado o elemento, os atributos do mesmo podem ser adicionados. Para tanto, basta clicar com o botão esquerdo do mouse sobre o elemento e usar a tecla `Tab` para acessar seguidamente os atributos. Pressionando a tecla `Esc`, ou clicando novamente sobre o elemento selecionado no palete, estingue-se a seleção. A seguir serão apresentados os atributos dos respectivos elementos de um modelo.

7.1. Inscrições nos Lugares

Existem três inscrições que podem ser associadas a um lugar. Uma é obrigatória e as outras duas são opcionais. Veja Figura 30:

- `PLACE TYPE` (obrigatória) – inscrição do conjunto de cores associado ao lugar;
- `INIT MARK` (opcional) – inscrição da marcação inicial de um lugar;
- `NAME` (opcional) – nome do lugar, que não tem significado semântico, mas é importante para a compreensão do modelo.

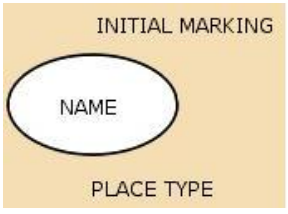

Figura 30. Atributos de um lugar de uma rede de Petri colorida.

Ao se criar um lugar, o primeiro atributo que aparece para ser preenchido é o nome do lugar. Esse atributo é escrito no centro do lugar. Após preencher o nome (se assim se desejar), teclando `Tab` aparecerá `PLACE TYPE`, que obrigatoriamente deverá ser preenchido. Desta forma, o último atributo a aparecer é o da marcação inicial.

Quando o lugar possui uma marcação inicial diferente de vazio, ao se completar a inscrição da marcação, o CPN TOOLS cria automaticamente a marcação atual (marcação corrente). Essa marcação é composta de um círculo (ou elipse) e um retângulo verdes. No círculo verde é apresentado o número total de fichas do lugar naquela marcação e no retângulo verde são apresentados os detalhes da marcação. Veja exemplo na Figura 31:

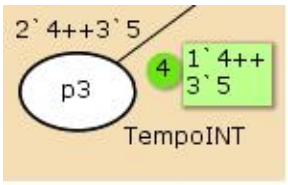

Figura 31. Atributos de um lugar: nome (p3); conjunto de cores (TempoINT); marcação inicial (`2`4++3`5`); marcação atual (`4 (1`4++3`5)` em verde).

7.2. Inscrição nos Arcos

A inscrição no arco possui a seguinte forma (Figura 32)

Figura 32. Atributo de um arco.

A expressão do arco `expr` tem que coincidir com o conjunto de cores associado ao lugar ligado ao arco, caso contrário, uma mensagem de erro aparecerá próxima ao arco, durante a verificação de sintaxe.

Existe uma diferença básica entre as inscrições dos arcos de entrada e as dos arcos de saída de uma transição.

A expressão do arco de entrada da transição é constituída pela escolha de fichas dos lugares de entrada da transição.

A expressão do arco de saída da transição é um construtor para a criação de novas fichas. Este construtor, frequentemente, usa variáveis das inscrições dos arcos de entrada da transição. Veja Figura 33

Na rede apresentada nessa figura, as variáveis `x` e `y` escolhem aleatoreamente uma ficha de seus respectivos lugares de entrada. A expressão do arco de saída da transição é o construtor de uma nova ficha criada a partir da soma dos valores que as variáveis `x` e `y` assumem no momento do disparo da transição. No exemplo, `x=3` e `y=4`.

As expressões dos arcos de saída das transições podem ser temporizadas. No exemplo apresentado na Figura 34, a cada 50 unidades de tempo uma cópia da ficha que

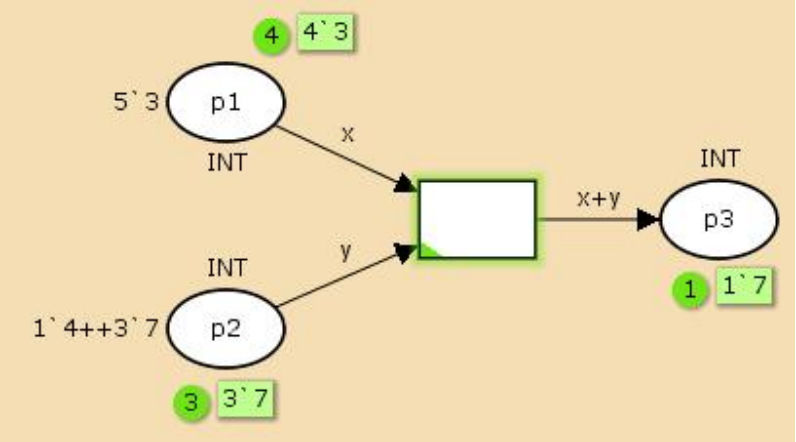

Figura 33. Expressões nos arcos de entrada e de saída de uma transição.

se encontra em `p1` é colocada no lugar `p2` e uma nova ficha de mesmo valor (4) é colocada em `p1` com sua temporização acrescentada de +50 unidades de tempo.

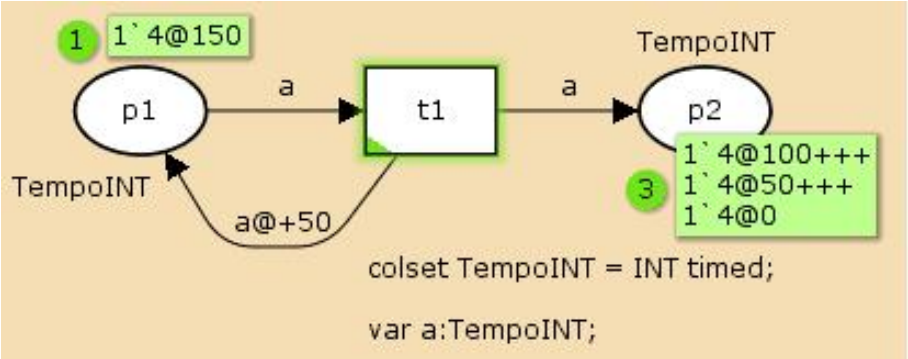

Figura 34. Expressão temporizada em um arco de saída de uma transição.

7.3. Inscrições nas Transições

São quatro as inscrições que podem ser associadas às transições. Todas são opcionais (Veja Figura 35):

- Nome (NAME);
- Guarda (GARD);
- Retardo de tempo (TIME DELAY);
- Segmento de Código (CODE SEGMENT).

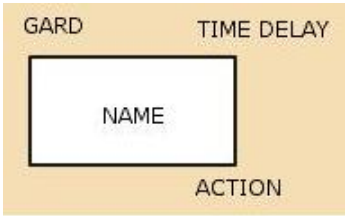

Figura 35. Inscrições que podem ser associadas a uma transição.

A inscrição `TIME DELAY` tem que ser uma expressão inteira positiva. A mesma é precedida pelos símbolos @+, o que significa que a inscrição possui a seguinte forma:

@+ delay-expr.

O retardo de tempo é sempre relativo ao tempo atual, ou seja, se, por exemplo, o tempo atual é 20 e o retardo de tempo associado à transição é @+5, então, o tempo associado às fichas colocadas nos lugares de saída da transição, quando do seu disparo, será igual @+25.

Se a transição não possui tempo associado, isto indica que o retardo de tempo associado ao disparo da transição é zero. Diferente do retardo de tempo associado a um arco, que associa este retardo somente às fichas relativas àquele arco, o retardo de tempo associado a uma transição associa esse retardo a todas as fichas colocadas nos lugares de saída da mesma. Veja na Figura 36 a diferença da temporização associada ao arco de saída da transição em relação à temporização associada à própria transição.

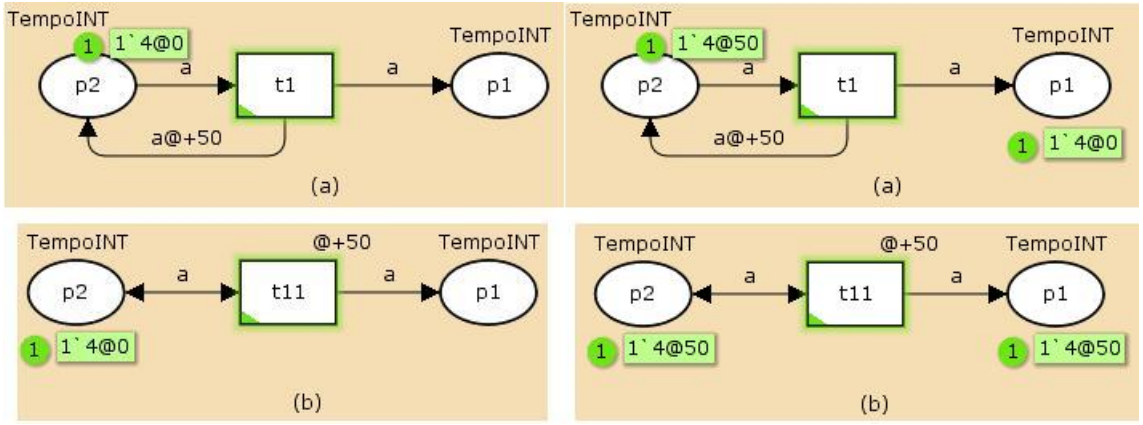

Figura 36. Diferença da temporização de uma transição para a temporização de um arco de saída da transição.

Na Figura 36(a) é apresentada uma rede com temporização somente no arco que liga `t1` a `p2`. Na marcação inicial a rede possui uma ficha (`1 '4`@+0) no lugar `p2`. Após o disparo de `t1`, note que uma ficha `1 '4`@+0 foi colocada em `p1` e uma ficha `1 '4`@+50 foi colocada em `p2`. Como não existe temporização nem na transição, nem no arco que liga `t1` a `p1`, então a ficha nesse lugar continua com a mesma temporização da ficha em `p2` na marcação anterior.

Na Figura 36(b) a temporização é associada à transição (@+50). Assim, na marcação inicial a rede possui uma ficha (`1 '4`@+0) no lugar `p2`. Após o disparo de `t11`, as fichas nos lugares de saída da transição possuem uma temporização de @+50, pois este é o tempo acrescentado às mesmas no disparo da transição.

A inscrição `GUARD` é uma expressão booleana da linguagem CPN ML, que é avaliada como `verdadeiro` ou `falso`. O valor inicial da guarda é sempre `verdadeiro`, antes que uma expressão seja adicionada. Uma guarda pode ser uma única expressão ou uma lista de expressões booleanas, `[b-expr1, b-expr2, ..., b-exprn]`. Quando uma expressão de guarda é associada a uma transição, esta só poderá disparar se a expressão for verdadeira. Veja exemplo apresentado na Figura 37.

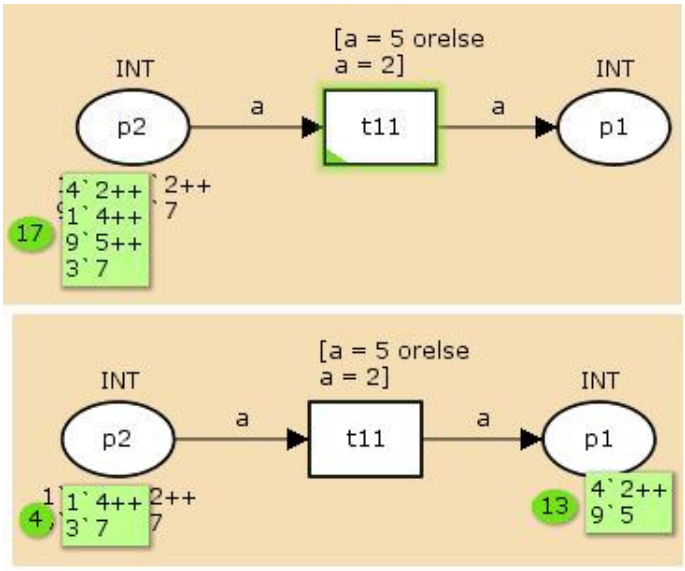

Figura 37. Expressão de guarda em uma transição restringe o disparo da mesma para quando a expressão for verdadeira.

A marcação inicial do lugar `p2` é (`4‘2++1‘4++9‘5++3‘7`), a expressão de guarda da transição `t11` é `[a=5 orelse a=2]`. Neste caso, `t11` está habilitada, pois existem quatro fichas `2` e nove fichas `5` em `p2`. Após `t11` ser disparada treze vezes, a nova marcação da rede é (`1‘4++3‘7`) em `p2` e (`4‘2++9‘5`) em `p1`. Nesta nova marcação, a transição não está mais habilitada, pois sua expressão booleana é avaliada como falsa, já que não existe mais nenhuma ficha em `p2` de valor 2 ou 5.

Usando a expressão de guarda, é possível se comparar fichas (ou parte) de diferentes lugares de entrada da transição. Na figura 38 o valor do segundo campo da ficha em `p1` é comparado com o valor da ficha em `p2`.

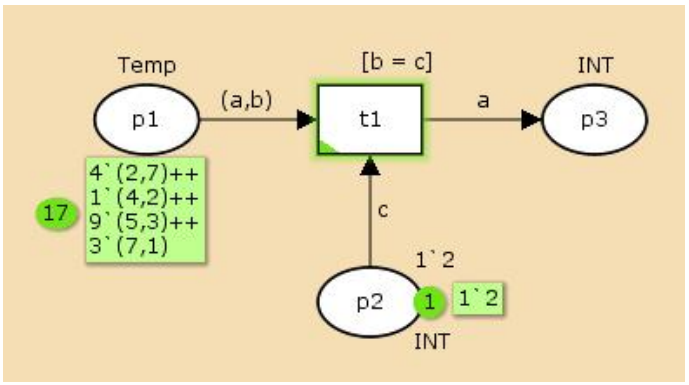

Figura 38. Exemplo de aplicação da expressão de guarda de uma transição.

No exemplo da figura, a transição está habilitada, porque `p1` possui uma ficha (4,2) e `p2` uma ficha de valor `2`. Na hora do disparo, a variável `b` é associada ao segundo campo da ficha (4,2) em `p1` e a variável `c` é associada à ficha `2` do lugar `p2`, assim, a expressão `[b = c]` é verdadeira.

A inscrição `CODE SEGMENT`, também opcional, contém um código ML. Um segmento de código é executado quando a transição, à qual o mesmo está associado, é disparada. Cada segmento de código pode utilizar variáveis CPN e pode associar estas variáveis aos arcos de saída de uma transição, mesmo que elas não tenham sido associadas a nenhuma ficha dos lugares de entrada da transição. Um segmento de código pode conter:

- Padrão de entrada (`Input pattern`) – opcional;
- Padrão de saída (`Output pattern`) – opcional;
- Código (`Code action`) – obrigatório;

Um `Input pattern` é um conjunto de variáveis CPN, precedidas pela palavra reservada `input`. O `Input Pattern` lista as variáveis CPN que podem ser usadas no `Code action`, sem que os valores destas variáveis possam ser modificados. Se o `Input pattern` não é declarado, então, nenhuma variável CPN pode ser usada no `Code action`.

Um `Output pattern` é um conjunto de variáveis CPN, precedidas pela palavra reservada `output`. O `Output pattern` lista as variáveis CPN a serem modificadas como resultado da execução do `Code action`. Se o `Output pattern` não é declarado, então, nenhuma variável CPN pode ser calculada.

Um `Code action` é uma expressão ML precedida pela palavra reservada `action`. O `Code action` não pode conter nehuma declaração de conjunto de cores, variáveis CPN, ou variáveis de referência. No entanto, ele pode utilizar constantes pré-declaradas ou declaradas pelo usuário, operações e funções. Em adição, novas funções e constantes podem ser definidas localmente por meio da cláusula `let-in-end`. O `Code action` é executado como uma declaração local em um ambiente contendo variáveis CPN especificadas no `Input pattern`. Isso garante que o `Code action` não pode modificar diretamente quaisquer variáveis CPN, mas tão somente cópias locais das mesmas. Quando o `Code action` é executado, seu resultado é aplicado às variáveis CPN declaradas no `Output pattern`. Quando o `Code action` é avaliado em um ambiente contendo um `Input pattern`, as variáveis, obrigatoriamente, produzem um resultado do mesmo tipo das variáveis do `Output pattern`.

Normalmente, o segmento de código é usado para um processamento mais complexo das fichas de entrada. A soma de fichas apresentada no exemplo da Figura 33 pode ser representada usando segmento de código. Veja Figura 39.

8. Particularidades dos Modelos com Restrições de Tempo no CPN TOOLS

No CPN Tools pode-se construir modelos de redes de Petri coloridas com e sem restrições de tempo. Se nenhum dos conjuntos de cores associados ao modelo possuir o modificador `timed`, então a rede não possuirá restrições de tempo. Neste caso, o CPN Tools utiliza o seguinte algoritmo para simulação da rede (Figura 40):

Vale lembrar que a escolha da transição a disparar pode ser feita manualmente (passo-a-passo), ou automaticamente, de forma aleatória, pelo próprio CPN Tools. Isso é feito utilizando o palete `simulation`.

Para o caso de redes com restrições de tempo, o CPN Tools utiliza o seguinte algoritmo para simulação da rede (Figura 41):

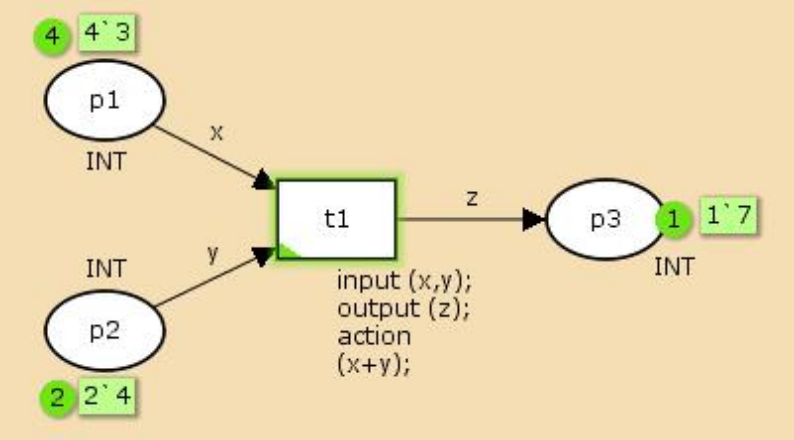

Figura 39. Exemplo de segmento de código associado a uma transição.

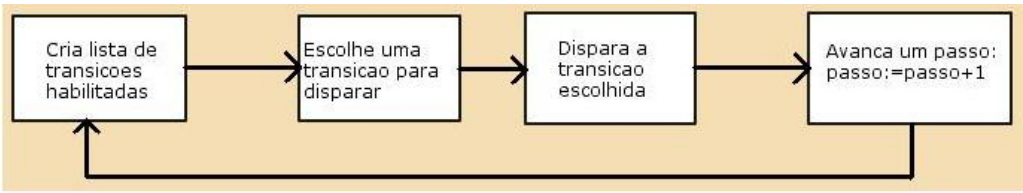

Figura 40. Algoritmo de simulação de RPC sem restrição de tempo.

O próximo instante de tempo do modelo não será `Time+1`, mas sim o tempo do próximo evento `Time+Next_Time`. Neste próximo evento são executadas todas as transições que estejam habilitadas naquele instante. Desta forma, é necessário muito cuidado quando se combina transições temporizadas com transições não temporizadas, pois estas últimas serão disparadas até que não haja mais nenhuma habilitada. Isso pode causar um laço infinito (livelock) no comportamento do modelo. No exemplo apresentado a seguir na Figura 42, o tempo não avança e somente a transição `t1` irá disparar para sempre, pois ela estará sempre habilitada no tempo @+0. Em consequência, a transição `t2` nunca irá disparar porque o tempo nunca avançará para @+1.

Desta forma, é implementada no CPN Tools uma classe de redes de Petri coloridas com restrição de tempo simples e, ao mesmo tempo, poderosa, devido à associação de tempo com as fichas. A cada ficha é associado um tempo @+t e um relógio global é usado para todo o modelo. Para os instantes em que o tempo do modelo (tempo global – `Tg`) é menor que o tempo associado a uma certa ficha (`Tg < t`), a ficha não é processada pelo simulador, isto é, ela fica indisponível para habilitação e disparo de qualquer transição.

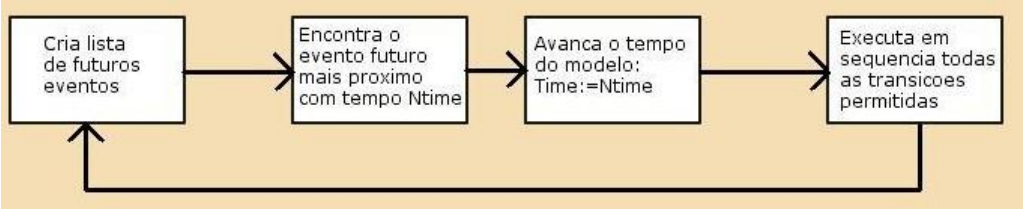

Figura 41. Algoritmo de simulação de RPC com restrição de tempo.

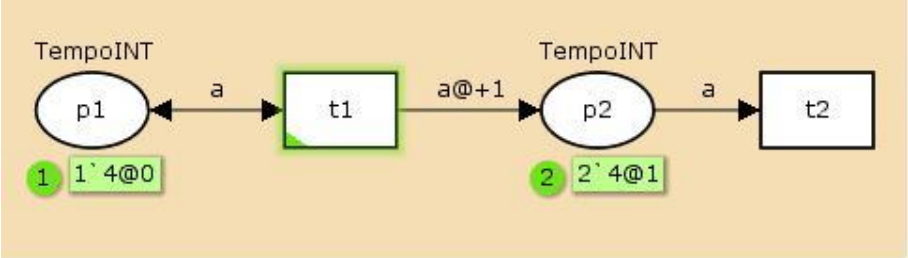

Figura 42. O tempo não avança porque `t1` estará sempre habilitada.

No instante em que `Tg = t`, a ficha se torna, então, disponível e já pode ser usada pelo simulador para habilitação e disparo das transições. Fichas com temporização @+t, que chegam nos lugares de saída de uma transição, esperam seu tempo de indisponibilidade nos respectivos lugares de saída. O exemplo apresentado na Figura 43 mostra como o tempo é processado no CPN Tools.

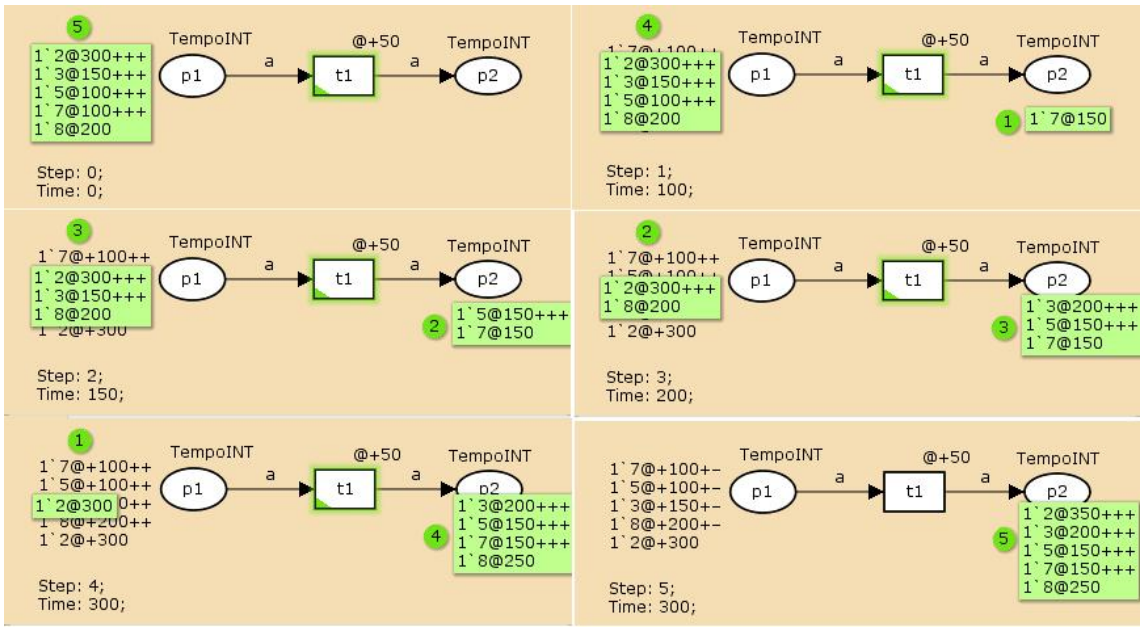

Figura 43. Processamento do tempo no CPN Tools.

No tempo t=0, nenhuma transição estará habilitada, pois nenhuma ficha possui um tempo associado igual a 0. Nesse caso, o relógio global avança para o próximo evento em t=100. Nesse instante, a transição é disparada e uma ficha é colocada no lugar `p2`. Essa ficha (`1‘7@150`), no instante do disparo possuia um tempo associado igual a 100. No disparo, a esse tempo é acrescido o valor +50, que é o retardo associado à transição. Note que, após este disparo, o relógio não avançará porque no lugar `p1` existe ainda uma ficha cujo tempo associado é 100. No passo (step) 2 essa ficha é retirada de `p1` e colocada em `p2`, com um tempo associado de +150. Note que isto significa que estes dois eventos ocorreram concorrentemente. Após o disparo de `t1` no tempo t=100, o relógio avança para o tempo t=150 e `t1` dispara, retirando a ficha `1‘3@150` de `p1` e depositando a ficha `1‘3@200` em `p2`. Novamente, o relógio avança, agora para t=200 e o disparo de `t1` retira a ficha `1‘8@200` de `p1` e deposita a ficha `1‘8@250` em `p2`. Por fim, o

relógio avança para t=300 e o disparo de `t1` retira a ficha `1 '2@300` de `p1` e deposita a ficha `1 '2@350` em `p2`. Nesse instante o relógio não mais avançará pois não existe mais nenhuma transição habilitada.

Para descrever retardos de tempo, pode-se usar tempos associados às transições e/ou tempos associados aos arcos. Retardos de tempo associados às transições são, muitas vezes, mais compreensíveis porque as transições modelam ações, portanto, possuem naturalmente um retardo de tempo associado. No entanto, o uso de retardo de tempo nos arcos pode dar uma maior flexibilidade à construção do modelo.

Por exemplo, para modelar tráfico (de rede de computadores, telefônico, de carros, etc) é conveniente se usar uma função aleatória. A grande variedade de funções aleatórias disponíveis (veja Seção 6.5) permite a descrição das particularidades de um tráfego. No exemplo apresentado na Figura 44 é gerado um fluxo de números inteiros com uma distribuição de Poisson com média 5.

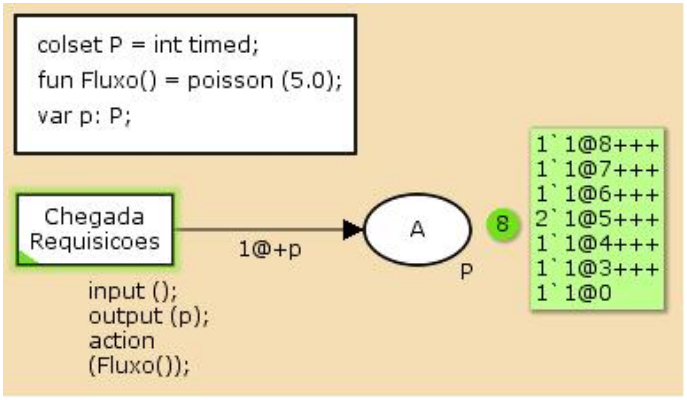

Figura 44. Geração de fluxo de inteiros segundo uma distribuição de Poisson de média 5.

9. Fragmentando um Modelo para Melhor Visualização

Vários editores gráficos, tais como o Corel Draw, fornecem suporte para operações que destacam partes de uma figura. O conceito de fragmentos de rede é comum no CPN Tools e é formulado através da criação de grupos de elementos. Um grupo pode ter qualquer formato e o mesmo pode ser formado pela seleção de alguns elementos da rede. Após a formação do grupo, o mesmo pode ser duplicado (copiado) e movido para qualquer local usando-se a ferramenta `Clone` do palete `Create`.

Para criar um grupo, use a ferramenta `New Group`. O menu correspondente aparece quando se clica no botão direito do mouse no canto inferior esquerdo da `área de trabalho`. Veja Figura 45.

Selecionando a entrada `New Group` aparecerá uma nova película, (denominada `Group 1`), na área de trabalho, deixando os elementos da rede quase invisíveis. Veja Figura 46.

Para ativar (visualizar) um grupo, basta clicar no nome dele na parte inferior da área de trabalho. Clicando no primeiro grupo (denominado `None`) toda a rede ficará visível.

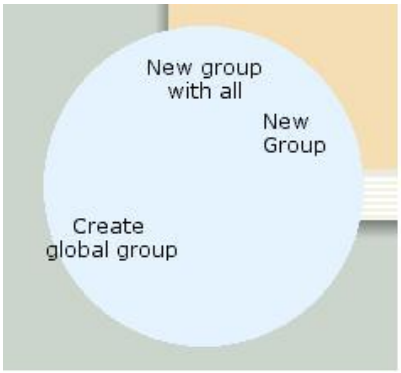

Figura 45. Menu para criação de novos grupos.

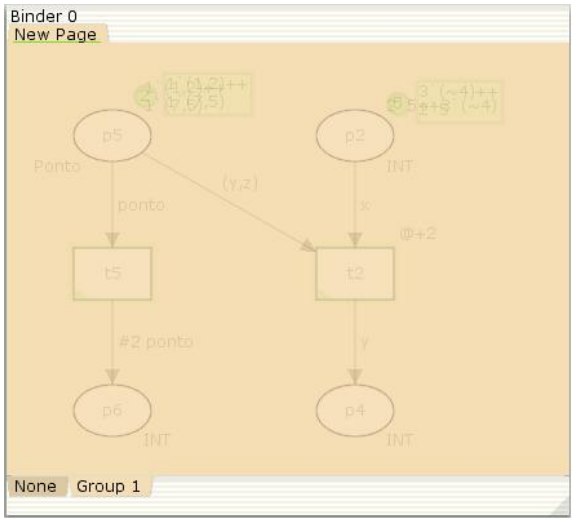

Figura 46. Criação de um novo grupo.

Para adicionar um novo elemento (e respectivas inscrições) ao grupo ativo, usa-se a ferramenta `Toggle Group`. Para tanto, basta levar o mouse para cima do elemento que se deseja adicionar, pressionar o botão direito do mouse e selecionar a opção `Toggle Group`. Veja Figura 47

Se o elemento já pertencer ao grupo ativo, realizando esta mesma operação, o elemento deixará de fazer parte do grupo.

Grupos podem ser usados para diferentes propósitos:

- Destacar um grupo de elementos;
- Modificar atributos;
- Mover um grupo de elementos;
- Clonar um grupo de elementos;
- Deletar um grupo de elementos.

Os grupos podem ser usados para destacar partes de um modelo, para facilitar o entendimento do mesmo.

Os grupos podem ser usados para modificar atributos de vários elementos ao mesmo tempo. Por exemplo, quando se deseja modificar a cor de um grupo de elementos,

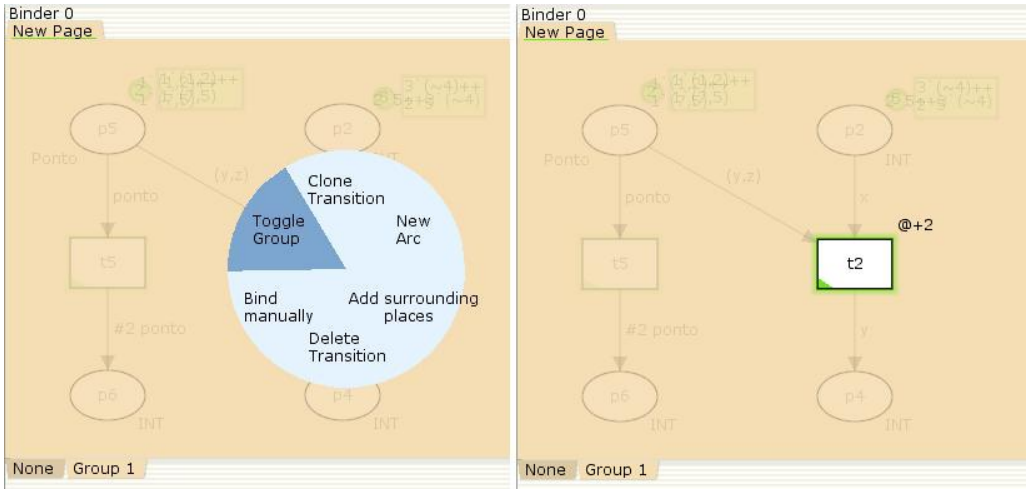

Figura 47. Adicionando um elemento ao grupo.

cria-se um grupo com os mesmos e, usando uma cor do palete `Style` basta clicar com o mouse em qualquer um dos elementos do grupo que todos terão sua cor modificada. Esta operação é importante para a criação de redes que possuem estruturas reusáveis.

Os grupos podem ser usados para mover vários elementos ao mesmo tempo. Para tanto, basta pressionar o botão esquerdo do mouse em qualquer um dos elementos do grupo, arrastar o cursor para a posição desejada e todos os elementos do grupo serão movidos ao mesmo tempo.

Da mesma forma, os grupos podem ser usados para clonar elementos, assim, em vez de clonar um elemento por vez, clona-se todos os elementos do grupo ao mesmo tempo. Uma vez clonados, os elementos do grupo podem ser copiados uma ou mais vezes para uma ou mais páginas da rede.

A ação de deletar um dos elementos do grupo é a mesma ação realizada com a rede normal.

Existem dois tipos de grupos: `Normal Group` e `Global Group`. As ações relativas ao `Normal Group` foram explicitadas acima. Construindo um `Global Group` é possível manipular elementos de todas as páginas de uma rede, se a mesma for hierárquica. Ao se criar um `Global Group`, o nome do mesmo aparece no canto inferior direito da área de trabalho.

Algumas operações que podem ser realizadas em um `Normal Group` não têm significado em um `Global Group`. `Move to subpage` e `clone elements` são exemplos destas operações. Não se poderia mover para uma única página elementos que fazem parte de submodelos diferentes, como não faz sentido deletar um elementos que está representando várias instâncias e se encontra em vários submodelos diferentes.

10. Fusion Places (Lugares de Fusão)

`Fusion Places`, ou lugares de fusão, possibilitam a construção de modelos mais fáceis de visualizar e possibilitam a conexão entre duas ou mais páginas de uma rede. Pode-se considerar que os lugares de fusão são o primeiro passo para a criação de redes hierárquicas, tanto que, a ferramenta `Fuses places into a single fusion`

`set` é uma das ferramentas do palete `Hierarchy`.

Cada lugar pertencente a um conjunto de lugares de fusão possui associado uma etiqueta com um único nome que identifica o conjunto. Todos os lugares de um conjunto de fusão são considerados pelo CPN Tools como um único lugar. Se uma marcação muda em um dos lugares de fusão, ela mudará também em todos os outros lugares do conjunto. Desta forma, o mesmo conjunto de cores é associado a todos os lugares de um mesmo conjunto de fusão. Veja o exemplo apresentado na Figura 48 e observe a marcação nos lugares `A`, `B` e `C`, antes e depois do disparo da transição.

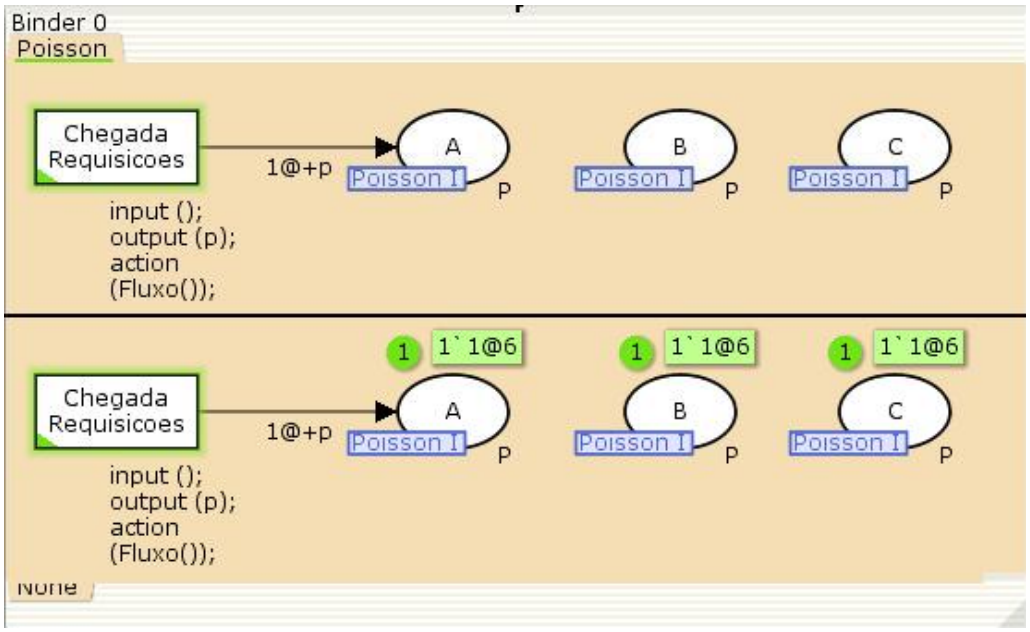

Figura 48. Um conjunto de lugares de fusão representa um único lugar.

Quando a marcação em um dos lugares do conjunto de fusão `Poisson I` é vazia, em todos os outros lugares do conjunto a marcação também é vazia. Após o disparo da transição, uma cópia da mesma ficha (`1`1@6`) é colocada em todos os lugares do grupo.

11. Construção de Modelos Hierárquicos

Na engenharia e na computação é comum a construção de modelos hierárquicos. Um equipamento eletrônico é constituído por placas de circuito impresso, cada placa é constituída de circuitos integrados, cada circuito integrado é constituído de transistores, capacitores, resistores, etc. Um programa em uma determinada linguagem formal é constituído de módulos (procedimentos, funções, classes, etc). Um modelo hierárquico significa uma construção aninhada (uma rede dentro de outra). De uma forma geral, um lugar ou uma transição em uma rede de Petri pode ser substituída por uma outra rede que refina a ação modelada por aquele lugar ou aquela transição. No CPN Tools é utilizada apenas a `transição de substituição` para a criação de modelos hierárquicos.

11.1. Transição de Substituição

A transição de substituição é a forma usada pelo CPN Tools para se poder substituir uma transição de uma rede de mais alto nível por uma rede de mais baixo nível.

Seja o exemplo apresentado na Figura 49. Nele, a página de mais alto nível `Principal` possui a transição de substituição `t2`, que é susbstituída pela subpágina `secundaria`. A substituição é assinalada pela etiqueta secundaria que aparece próxima à transição `t2` na página `principal`. Note também que a subpágina associada a `t2` possui o mesmo nome da etiqueta.

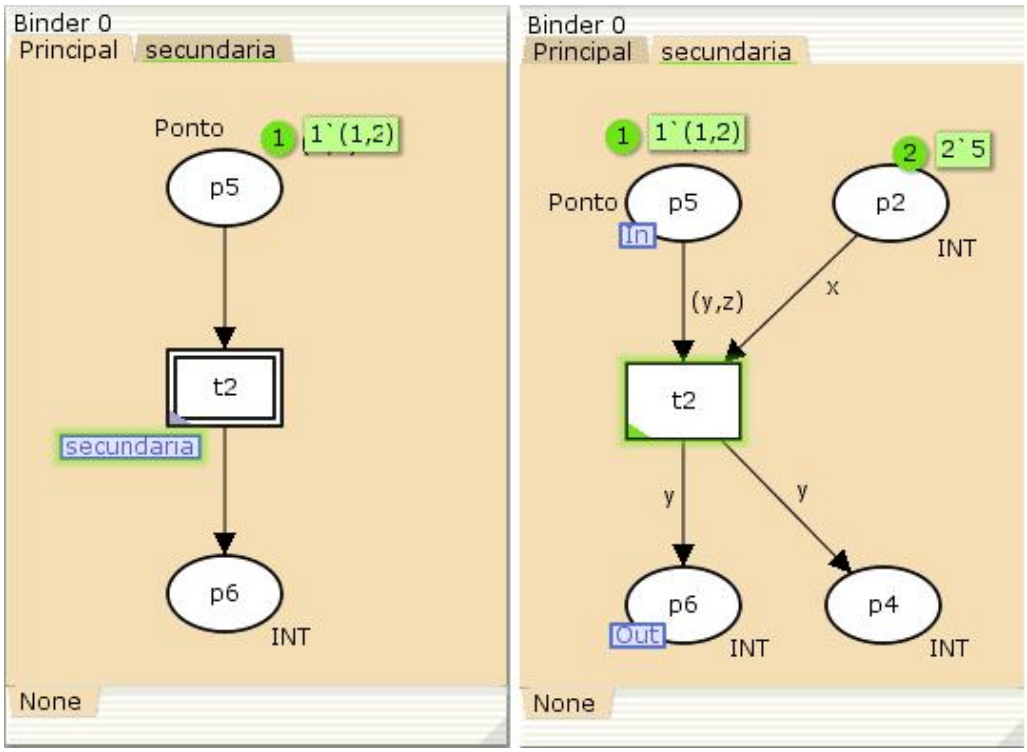

Figura 49. Transição de substituição `t2` e sua subrede associada `secundaria`.

Observe que os lugares `p5` e `p6`, que são respectivamente entrada e saída de `t2`, aparecem tanto na página `principal`, como na página `secundaria`. Na página `principal` estes lugares são chamados de `sockets` e na `secundaria` são chamados de `ports` e são etiquetados, respectivamente, com as etiquetas `In` e `Out`.

Os modelos hierárquicos podem ser criados utilizando-se as famosas abordagens `Top-Down` e `Botton-Up`.

11.1.1. Abordagem Top-Down

Nesta abordagem, primeiro o projetista cria um modelo mais abstrato e em seguida vai refinando o mesmo através da criação de submodelos mais detalhados. No CPN Tools isso é feito da seguinte forma:

- Cria-se o modelo mais abstrato (Figura 50).
- Transforma-se uma transição comum em uma transição de substituição. Para criar uma transição de substituição, faça da seguinte forma:
  - Arraste o palete `Hierarchy` para a área de trabalho;

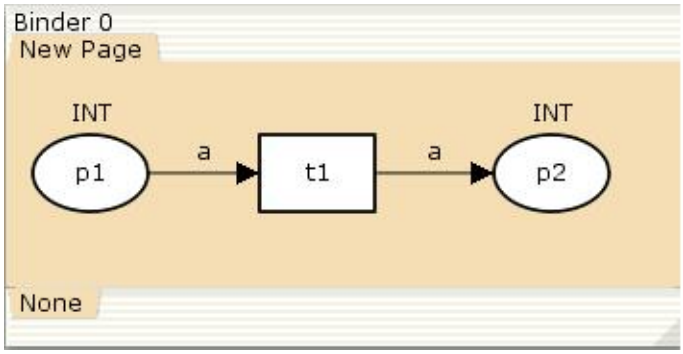

Figura 50. Modelo mais abstrato.

- Clique na ferramenta `Moves a transition to a subpage` (primeira do palete);
- Clique sobre a transição que se deseja transformar em transição de substituição (Figura 51).

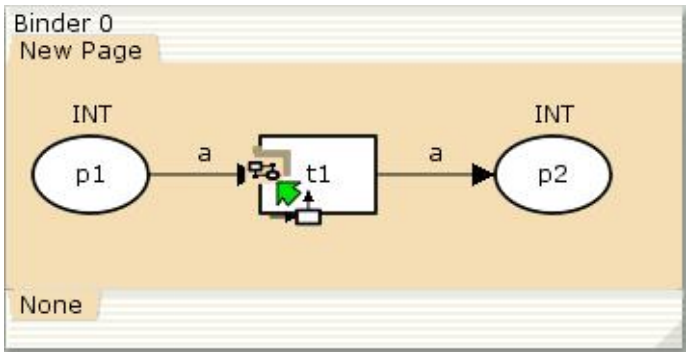

Figura 51. Criando uma transição de substituição.

Veja que a transição `t1` agora é graficamente representada por dois retângulos concêntricos e uma etiqueta azulada ao lado, também denominada de `t1` (Figura 52).

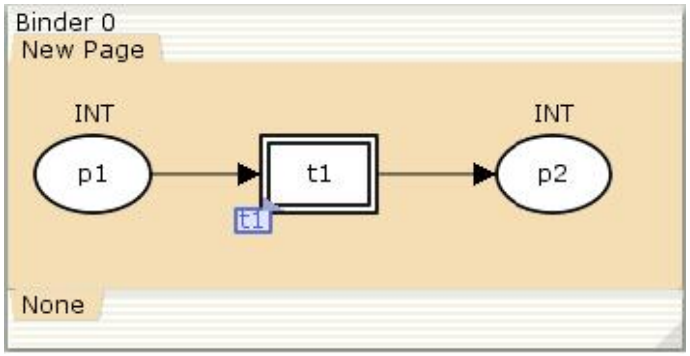

Figura 52. Transição de substituição criada.

Nesse instante também é criada uma subpágina associada a essa transição de substituição. A subpágina também é denominada de `t1` e aparece na área de índice logo abaixo do nome da página principal. Esta subpágina pode, então, ser arrastada para a área de trabalho (Figura 53).

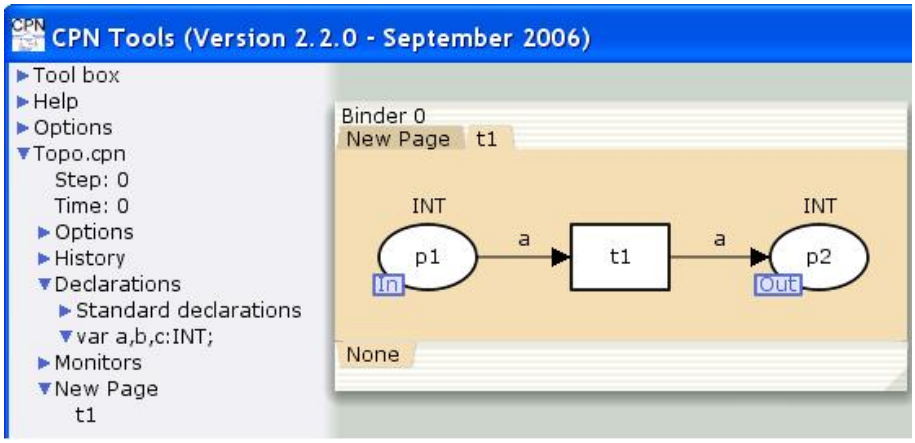

Figura 53. Subpágina associada à transição de substituição criada.

Note que, antes de `t1` ser transformada em transição de substituição, seus respectivos arcos de entrada e saída possuiam uma inscrição associada (Figura 50). Após `t1` ser transformada (Figura 52), as inscrições desaparecem. Em compensação, na subpágina `t1`, é criada uma subrede, contendo uma transição denominada de `t1` com os respectivos lugar de entrada (etiquetado por `In`) e lugar de saida (etiquetado por `Out`). As inscrições que havia na página principal, agora aparecem nos arcos ligados à transição criada na subpágina (Figura 53).

A subpágina criada pode ser modificada, acrescentando-se novos lugares e transições, desde que os lugares `Port (In, Out, I/O)` que existirem no instante de sua criação sejam mantidos.

11.1.2. Abordagem Botton-UP

Nesta abordagem, o projetista cria vários modelos detalhados e depois faz a ligação dos mesmos através da criação de super e subpáginas associadas. No CPN Tools isso é feito da seguinte forma:

- Crie duas ou mais páginas independentes, por exemplo `Pagina1` e `Pagina2`, conforme Figura 54. Neste exemplo, a `Pagina2` será transformada em subpágina associada à transição `t1` da `Pagina1`.

  Esse processo é dividido em duas fases:

- Atribua um dos tipos `Port` aos lugares desejados na subpágina a ser criada, usando uma das três ferramentas `Sets the port type to (In, Out, I/O)` do palete `Hierarchy`. Na Figura 55 foi atribuída a etiqueta `In` ao lugar `p5` e a etiqueta `Out` ao lugar `p6`. Note que aparecem duas mensagens de erro porque ainda não foi associada nenhuma transição de substituição à `Pagina2`.

- Use a ferramenta `Assigns subpage for a substitution transition` (terceira da linha de cima do palete `Hierarchy`. Após selecionar a ferramenta, clique em cima da transição que será transformada em transição de substituição e após clique na área de trabalho da página que será transformada em subpágina associada.

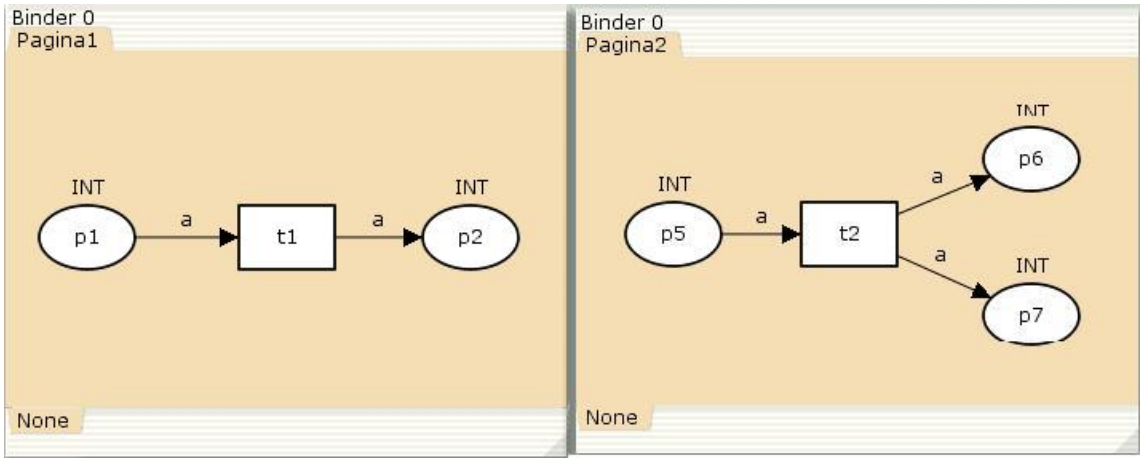

Figura 54. Dois modelos independentes (`Pagina1` e `Pagina2`).

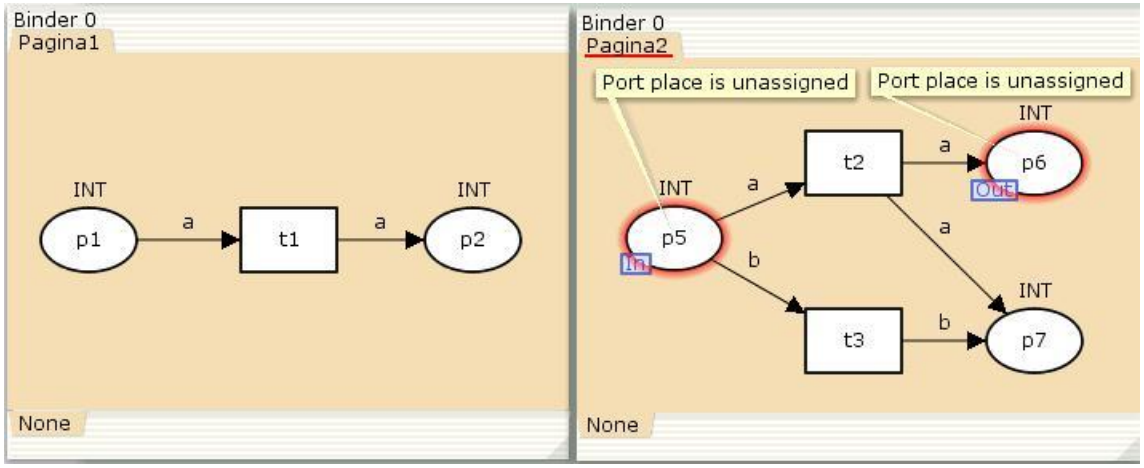

Figura 55. Atribuição dos tipos `Port` aos lugares da `Pagina2`.

No exemplo, clicou-se em `t1` de `Pagina1` e depois na área de trabalho de `Pagina2` (veja Figura 56). Note que neste caso as inscrições associadas aos arcos de entrada e saída da transição de substituição `t1` não desaparecem. Note também que a etiqueta associada a `t1`, por definição, é criada com o nome da subpágina associada, e não com o nome da própria transição, como é no caso da abordagem `Top-Down`.

12. Análise de Redes de Petri Coloridas

O CPN Tools fornece duas formas de análise de redes de Petri coloridas (RPC), são elas: simulação do comportamento da rede e geração do espaço de estados da rede. Através da análise, o projetista poderá ter mais confiança de que o modelo corresponde às especificações desejadas e que funciona adequadamente. Nesse estágio, pode-se fazer correção de erros (depuração) e modificações no modelo para adequá-lo às características desejadas.

Usando geração de espaço de estado, é possível analisar várias propriedades de uma rede, tais como reversibilidade (reversibility), limitação (boundedness) e vivacidade (liveness).

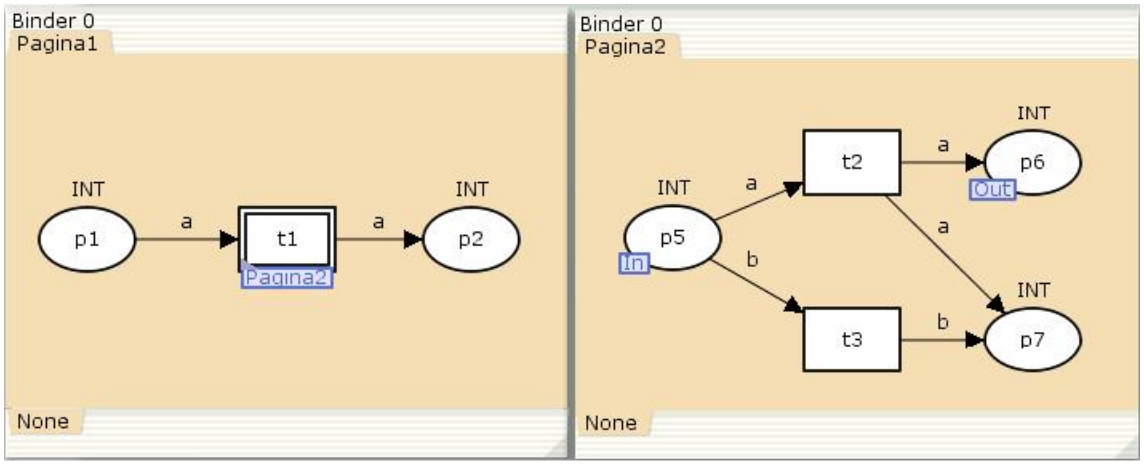

Figura 56. Criando subpáginas na abordagem `Botton-UP`.

Através da simulação, é possível se utilizar de uma outra ferramenta do CPN Tools, denominada `Monitor` e, assim, analisar aspectos específicos de uma rede, tais como: quantas vezes uma determinada transição foi disparada; uma determinada marcação foi alcançada; tempo médio de realização de uma determinada tarefa, etc.

12.1. Verificação de um modelo

Verificar modelos envolve a verificação de sintaxe e a simulação passo-a-passo. A verificação de sintaxe de uma rede, ou modelo, é feita automaticamente pelo CPN Tools quando uma rede está sendo criada ou quando uma rede já existente é carregada. Como já visto na Seção 4.5, é possível ver, através da indicação de auras coloridas e palavras sublinadas, em que estágio se encontra a verificação.

As indicações de cores são mostradas na área de índice sublinhando o nome da página que está sendo verificada naquele momento. Se a página está aberta na área de trabalho, o nome da página (na aba superior da área de trabalho) também será sublinhada com a cor correspondente ao estágio de verificação. Da mesma forma, aparecerá uma aura nos elementos da página que estão sendo verificados. A cor laranja indica que um elemento ainda não foi verificado.

Quando uma rede é carregada, a verificação de sintaxe é completada em alguns minutos. Durante este tempo, a aura dos elementos muda sua cor de laranja para amarelo e finalmente para vermelho se houver erro, ou os elementos perderão a aura caso a rede seja sintaticamente correta. Se a aura laranja permanece, é porque ocorreu um erro no processo de verificação ou existe um erro em um elemento da rede. Um brilho amarelo indica que o lugar/transição/arco/página/rede está sendo verificado neste instante.

Declarações são verificadas iniciando no topo. Se uma declaração depende de uma outra declarada posteriormente (abaixo da primeira), então ocorrerá um erro indicando que a outra declaração ainda não foi definida. Declarações com erro são novamente verificadas quando ocorre alguma modificação em qualquer declaração.

Uma aura vermelha indica que o elemento já foi verificado, mas contém um erro. Uma bolha de conversação aparecerá, mostrando uma mensagem com o erro encontrado. Elementos conectados ao elemento com erro também apresentarão uma aura vermelha e

não serão verificados até que o erro seja removido.

Se existir erro em alguma das declarações, aquela com erro será sublinada com a cor vermelha. O nome da rede e todos os nomes de páginas ligadas àquela declaração também serão sublinhados de vermelho. Para ver a mensagem de erro relativa à declaração, basta mover o cursor para cima da declaração e esperar alguns segundos que uma bolha de conversação aparecerá com a respectiva mensagem de erro.

Simulação passo-a-passo é usada para que se possa traçar o caminho seguido pelas fichas no modelo. Por exemplo, pode-se escolher manualmente as ligações a serem feitas entre as variáveis dos arcos de entrada de uma dada transição habilitada e as fichas disponíveis nos lugares de entrada dessa transição e, após, disparar a transição sob estas condições.

12.1.1. Análise do Espaço de Estados

Encontrar o espaço de estados de uma rede de Petri colorida é um pouco mais complicado que encontrar o grafo de alcançabilidade (árvore de alcançabilidade, árvore de cobertura) de uma rede de Petri. Em uma rede de Petri as marcações nos lugares são representadas por vetores cujos elementos são números naturais, enquanto que nas redes de Petri coloridas estas marcações são representadas por multi-conjuntos com ou sem restrições de tempo.

A análise do espaço de estados é possível para modelos pequenos ou mais simples devido ao conhecido fenômeno da esplosão de estados. O número de estados alcançados por uma rede de Petri *k-limitada* e *m* lugares é estimado em k^m. A análise do espaço de estados é realizada quando se deseja conhecer as propriedades da rede, tais como `limitação, vivacidades e reversibilidade`.

O palete `Statespace (SS)` é apresentado na Figura 57 e contém as seguintes ferramentas:

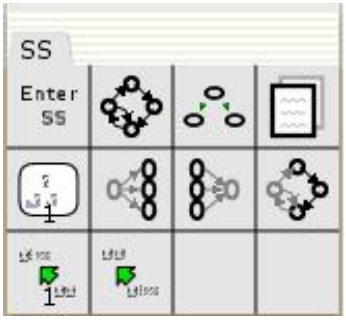

Figura 57. Palete de ferramentas para criação do espaço de estados de uma rede.

- `Enter SS` – entra no modo calcula espaço de estados;
- `Calculate State Space` – calcula o espaço de estados;
- `Calculate SCC graph` – calcula os componentes fortemente conectados (strongly connected components);

- `Save Report` – salva relatório em arquivo de texto de todos os dados e propriedades encontrados do espaço de estados calculado;

- `Displays the node with the specified number` – desenha uma marcação do espaço de estados calculado. Oferece opções de qual marcação desenhar;

- `Display the successors to this node` – apresenta todos os sucessores de um nó gerado graficamente;

- `Display the predecessors to this node` – apresenta todos os antecessores de um nó gerado graficamente;

- `Displays a partial SS graph according to the expression specified in the target Aux` – apresenta um grafo parcial do espaço de estados de acordo com uma expressão avaliada a partir de um texto auxiliar;

- `State Space To Sim` – muda o estado do CPN Tools do modo `SS` para `Simulação`;

- `Sim To State Space` – muda o estado do CPN Tools do modo `Simulação` para `SS`.

Entrando no Modo Espaço de Estados

Para entrar com sucesso no modo `State Space`, as seguintes condições são necessárias:

- A rede não pode possuir erros de sintaxe;
- Todos os lugares, transições e páginas da rede têm que ser unicamente nomeados;
- Lugares, transições e páginas têm que possuir nomes ML únicos[1]

Erros de sintaxe e nomes ML iguais serão identificados na verificação de sintaxe.

Antes de calcular e analisar o espaço de estados de uma rede, é necessário gerar o código do espaço de estados, isto é, o código ML que é usado para calcular e analisar o espaço de estados. Este código é gerado quando se usa a ferramenta `Enter SS` em uma das páginas da rede. Veja Figura 58.

Esta operação levará algum tempo. Evite pressionar mais de uma vez a ferramenta, pois isso pode causar um erro de execução do CPN Tools. Uma resposta textual é exibida na janela auxiliar de mensagens para indicar o status da operação. Veja Figura 59.

Quando a operação `Enter SS` tiver sido finalizada com sucesso, uma bolha de status verde aparecerá no canto inferior esquerdo da área de índice, caso contrário, se um erro ocorre, aparecerá uma bolha vermelha. Posicionando o cursor em cima desta bolha, uma mensagem de erro aparecerá.

[1] Um nome ML é um texto que é obtido de uma inscrição de nome durante a verificação de sintaxe. Um nome ML começa com uma letra e não possui espaços em branco entre nomes. Pode ser formado por letras, numeros e o separador _.

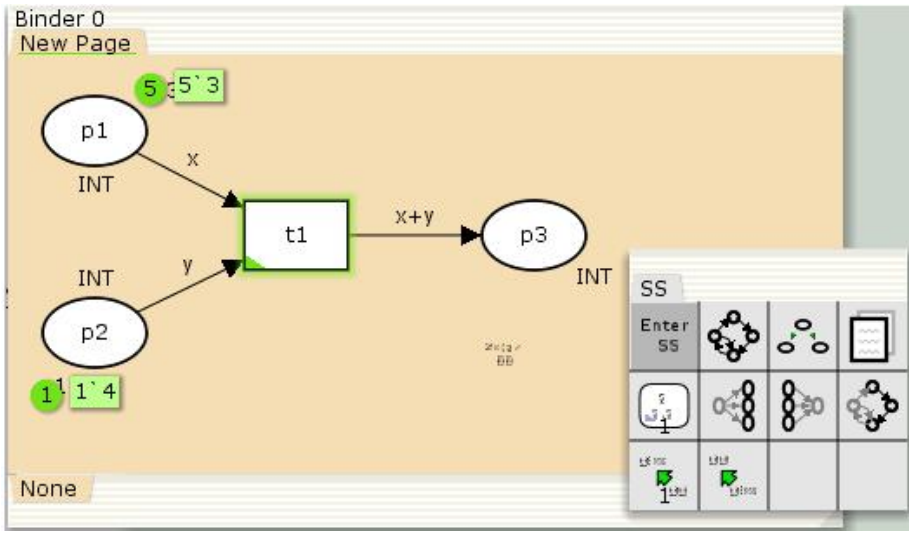

Figura 58. Geração do código do espaço de estados de uma rede.

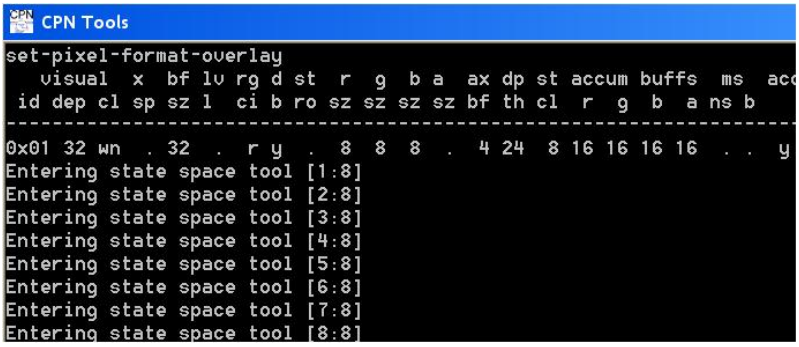

Figura 59. Janela auxiliar apresentando o status da operação.

Calculando o Espaço de Estados

Após entrar com sucesso no espaço de estados, pode-se calcular o espaço de estados. Isso é feito usando-se a ferramenta `Calculate State Space`.

Se é esperado que o espaço de estados seja pequeno, após selecionar a ferramenta no palete, basta simplesmente arrastar o cursor para uma das páginas da rede e clicar com o botão esquerdo do mouse.

Se é esperado que o espaço de estados seja grande, então, podem-se modificar as opções da ferramenta para que a simulação tenha algum critério de parada, como tempo, número de nós alcançados, etc. Estas modificações podem ser feitas diretamente na área de índice, conforme apresentado na Figura 60:

Se o cálculo foi realizado com sucesso, aparecerá uma bolha de status verde no canto inferior esquerdo da área de índice.

Se alguma das opções de parada é alcançada, então, uma bolha de status amarela aparecerá no canto inferior esquerdo da área de índice.

Se o cálculo falhou, então, aparecerá uma bolha de status vermelha. Posicionando o cursor sobre a bolha, aparecerá uma mensagem de erro.

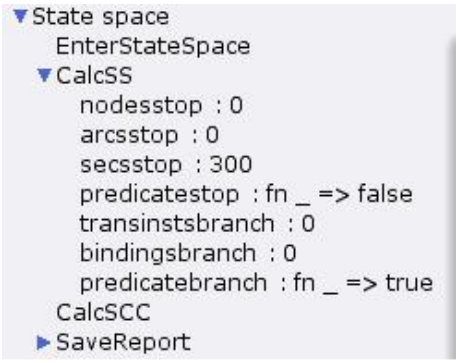

Figura 60. Opções da ferramenta `Calculate State Space`**.**

Calculando o SCC Graph

Após calcular com sucesso o espaço de estados, podem-se calcular os componentes fortemente conectados da rede, clicando com a ferramenta `SCC Graph` sobre uma das páginas da rede.

Se o cálculo foi realizado com sucesso, uma bolha de status verde aparecerá no canto inferior esquerdo da área de índice. Caso contrário, ou seja, se o cálculo falhou, uma bolha vermelha aparecerá. Posicionando o cursor no topo da bolha, uma mensagem de erro aparecerá.

Salvando Relatório do Espaço de Estados Calculado

Após calcular o espaço de estados e, opcionalmente, o `SCC Graph`, é possível salvar um relatório de todos os cálculos realizados. Para tanto, basta usar a ferramenta `Save Report` em uma das páginas da rede. Uma janela de diretórios aparecerá para que se escolha o nome do arquivo e onde o mesmo será salvo.

O conteúdo do relatório será determinado pelas opções da ferramenta `Save Report`, as quais podem ser modificadas no `tool box`, conforme apresentado na Figura 61.

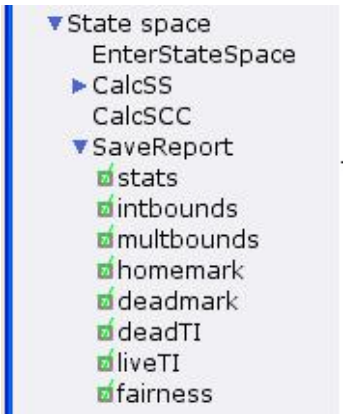

Figura 61. Opções da ferramenta `Save Report`**.**

Desenhando Espaço de Estados (SS)

Várias ferramentas do palete `SS` podem ser usadas para desenhar partes de um espaço de estados. Estes espaços só podem ser desenhados após entrar e calcular com sucesso o espaço de estado.

Usando a ferramenta `Display the node with the specified number` em uma página do CPN Tools, um determinado nó do `SS` será desenhado. O nó que será desenhado depende da opção escolhida para esta ferramenta. Veja Figura 62.

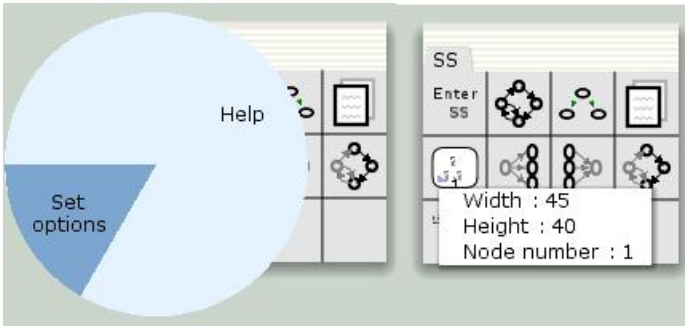

Figura 62. Opções da ferramenta `Display the node with the specified number`**.**

Na Figura 63 é apresentado o desenho de um nó de um espaço de estados. O primeiro número representa o número do nó (neste exemplo o número do nó é 1 – marcação inicial). Os dois números na parte de baixo do nó representam, respectivamente, o número de antecessores e o número de sucessores do nó que foram calculados. Neste caso, zero antecessores e dois sucessores.

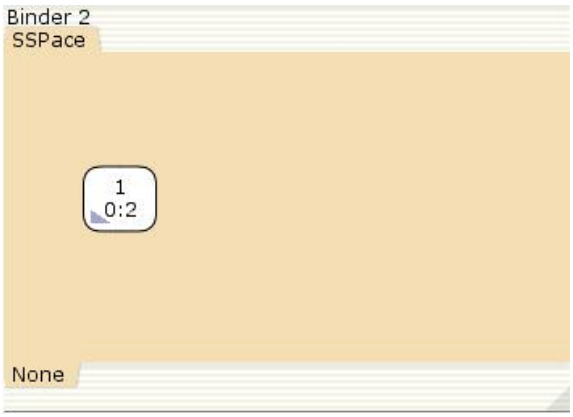

Figura 63. Marcação inicial de um espaço de estados (nó 1).

Usando a ferramenta `Display the successors to this node` pode-se desenhar os respectivos sucessores de um nó já desenhado. Da mesma forma, usando a ferramenta `Display the predecessors to this node` pode-se desenhar os respectivos antecessores de um nó já desenhado. Na Figura 64 são desenhados os dois sucessores do nó 1 apresentado na Figura 63.

Neste caso, o nó 2 possui um antecessor (nó 1) e um sucessor (ainda não desenhado). O nó 3 possui um antecessor (nó 1) e dois sucessores (ainda não desenhados). Veja na Figura 66 o desenho de um espaço de estados completo para a rede apresentada na Figura 65.

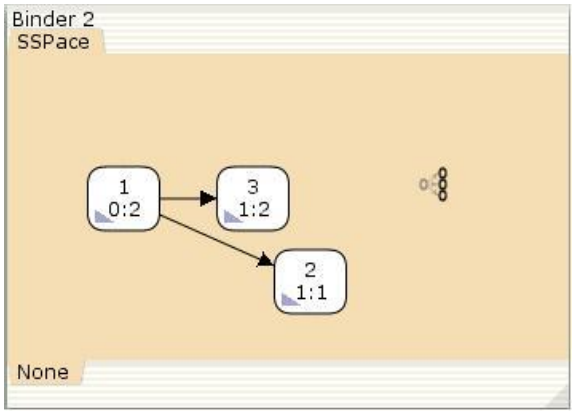

Figura 64. Desenhando sucessores de um nó do espaço de estados.

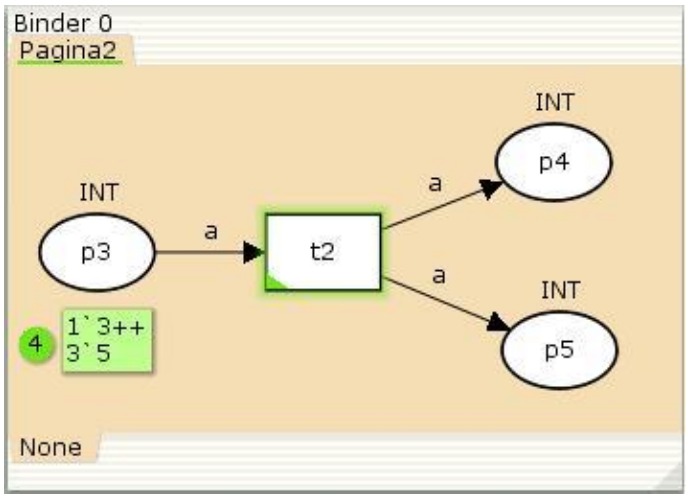

Figura 65. Exemplo de uma RPC e sua marcação inicial para criação de seu espaço de estados.

Clicando no pequeno triângulo no canto inferior esquerdo de um nó, aparecerá a descrição do nó, a qual mostra todos os detalhes da marcação associada ao nó. Veja Figura 67

Este nó corresponde à marcação 2 do exemplo apresentado na Figura 65. Esta marcação apresenta três fichas de cor 5 no lugar `p3`, uma ficha de cor 3 no lugar `p4` e uma ficha de cor 3 no lugar `p5`.

A descrição do nó poderá desaparecer novamente, bastando clicar novamente no triângulo.

De maneira similar, clicando em um arco do espaço de estados aparecerá uma descrição dos elementos de ligação associados ao arco. Veja Figura 68.

Neste exemplo, a rede se encontrava na marcação 1 e a transição `t2` disparou com a variável `a` de seu arco de entrada sendo ligada a uma ficha de cor 3.

Se o cursor passar sobre uma determinada descrição de nó ou arco, o elemento ao qual a descrição pertence será destacado.

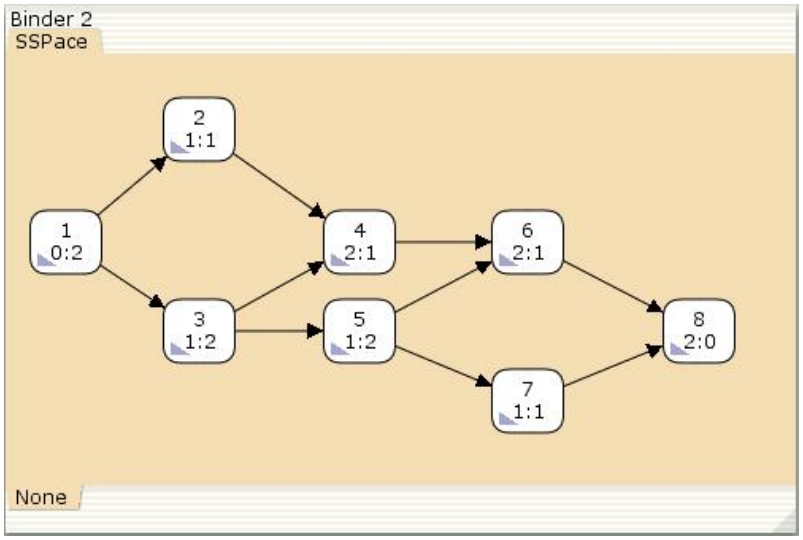

Figura 66. Espaço de estados completo da RPC apresentada na Figura 65.

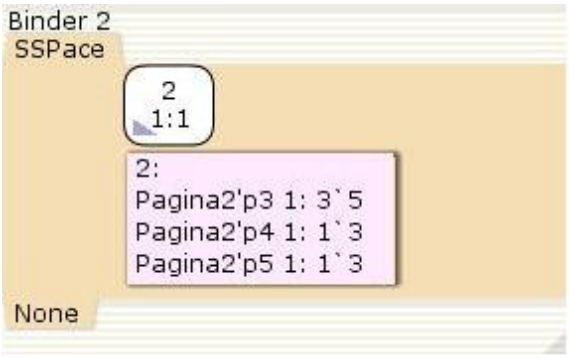

Figura 67. Descrição de um nó do SS.

Desenhando Nós e Arcos Usando Textos Auxiliares

A ferramenta `Displays a partial SS graph` é usada para desenhar grupos arbitrários de nós ou arcos. A ferramenta tem que ser aplicada a um texto auxiliar que contenha uma expressão CPN ML, a qual será avaliada como uma lista de nós ou uma lista de arcos. A expressão CPN ML poderá ser simples ou mais complexa. Na Figura 69 é apresentada uma expressão simples escrita com a ferramenta `Creates a text` do palete `Auxiliar`. Esta expressão contém uma lista de nós – [1,2,4]. Quando foi aplicada a ferramenta `Displays a partial SS graph` a esta lista (selecionando a ferramenta e clicando em cima da lista) os respectivos nós foram desenhados.

Se a ferramenta for aplicada a uma expressão CPN ML que contenha erro de sintaxe, ou que não retorna uma lista de nós ou de arcos, então, uma bolha de status vermelha aparecerá. Pasando o cursor em cima da bolha, a mensagem de erro correspondente aparecerá.

Todos os elementos do espaço de estados podem ser movidos para que se possa modificar o leiout e melhorar a visualização do mesmo.

Transferindo estados entre o simulador e o espaço de estados

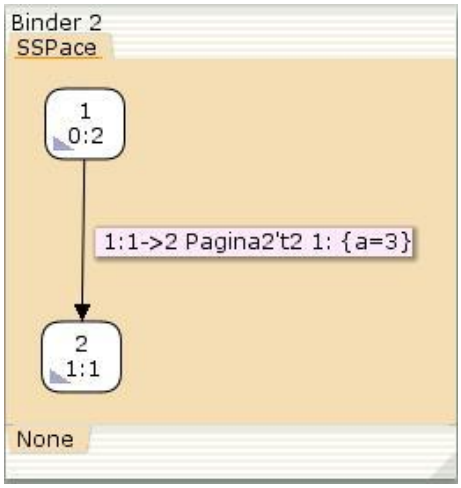

Figura 68. Descrição de um arco do SS.

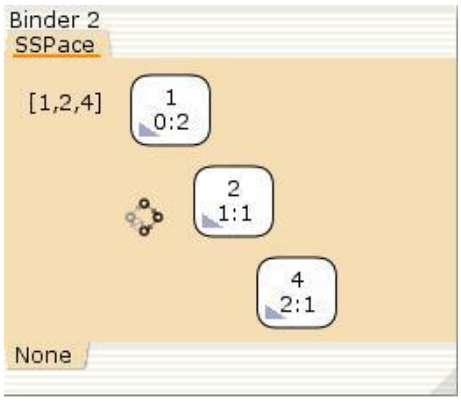

Figura 69. Desenho de parte do SS usando a ferramenta `Displays a partial SS graph`**.**

Para transferir um nó (marcação) do espaço de estados para o simulador, usa-se a ferramenta `State Space To Sim` em uma página da rede. O número do nó a ser transferido aparece nas opções da ferramenta. Qualquer nó do espaço de estados pode ser escolhido, bastando modificar a opção, que originalmente apresenta o número 1, ou marcação inicial. Para modificar a opção, pressione o botão direito do mouse sobre a ferramenta e escolha `Set options` no menu. Aparecerá uma pequena janela com um número que poderá ser modificado.

Na Figura 70(a) é apresentada uma rede e sua marcação inicial (`1`3++3`5`) no lugar `p3`. Aplicando a ferramenta `State Space To Sim` sobre a rede com a opção `state:3`, a marcação da rede mudará para `3`5` em `p3`, `1`3` em `p4` e `1`3` em `p5`, pois esta é a marcação de número 3 no espaço de estados. Veja Figura 70(b).

Para transferir um nó do simulador para o estado de espaços, usa-se a ferramenta `Sim To State Space` a uma página da rede. Uma bolha de status indicará se a operação foi realizada com sucesso.

Para maiores detalhes sobre a utilização do palete `State Space (SS)` veja o `State Space Tool Manual` [Jensen et al. 2006].

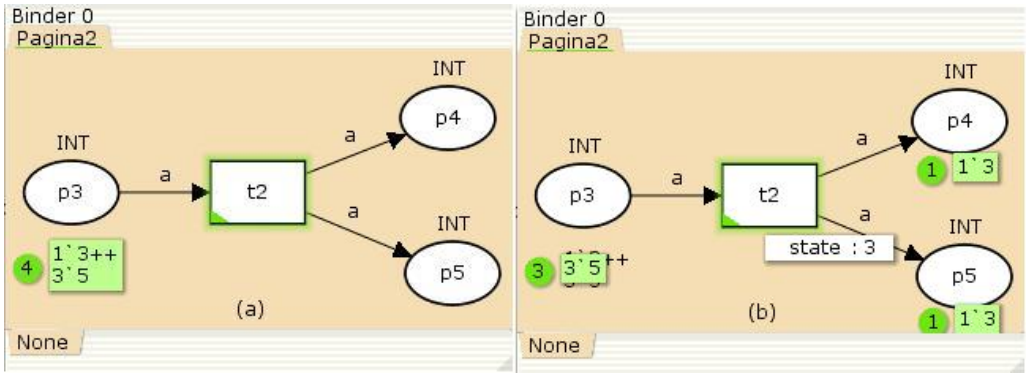

Figura 70. Transferência de uma marcação do SS para o simulador.

12.1.2. Simulação do Comportamento de uma Rede

O CPN Tools pode ser usado como um sistema típico de simulação. Quando o comportamento de uma rede é complexo, pode-se simular esse comportamento em grandes intervalos de tempo e tirar conclusões sobre as características do sistema modelado. Especificamente, quando funções aleatórias são largamente utilizadas no modelo, está-se interessado mais nas propriedades estatísticas do mesmo, do que em seu espaço de estados.

Para iniciar a simulação da dinâmica de uma rede, basta arrastar o palete `Sim` para a área de trabalho, como já visto na Seção 5.3. A seguir, selecione uma das ferramentas de simulação e aplique em uma das páginas da rede.

Critérios de Parada de uma Simulação

Uma simulação irá parar se uma das condições a seguir for encontrada:

- Não existe mais nenhuma transição habilitada;
- O número de passos (steps) especificado nas opções da ferramenta `Play` ou da ferramenta `Fast Forward` já tiver sido executado;
- A ferramenta `Stop` for aplicada após o início de uma simulação;
- A condição estabelecida para um dos monitores de parada (breakpoint monitors) for verdadeira.

Quando uma das condições de parada for encontrada, uma bolha de status verde e uma bolha de conversação indicarão, respectivamente, que a simulação parou com sucesso e qual condição foi encontrada.

O simulador irá verificar, em diferentes pontos, se alguma condição de parada foi encontrada, dependendo de qual ferramenta foi usada para iniciar a simulação.

Realimentação da Simulação

Enquanto a simulação está em execução (usando a ferramenta `Play`), são apresentadas as seguintes informações:

- A marcação atual dos lugares é apresentada próxima aos mesmos:
  - O número de fichas em um lugar é apresentado em um círculo verde;

  - Os valores correspondentes das fichas são apresentados em um retângulo verde.
- Auras verdes nas transições habilitadas são apresentadas e os nomes das páginas com transições habilitadas são sublinhados de verde;
- Passos (steps) e tempo de simulação são apresentados na área de índice, logo após o nome da rede.

Na Figura 71 são destacadas estas realimentações.

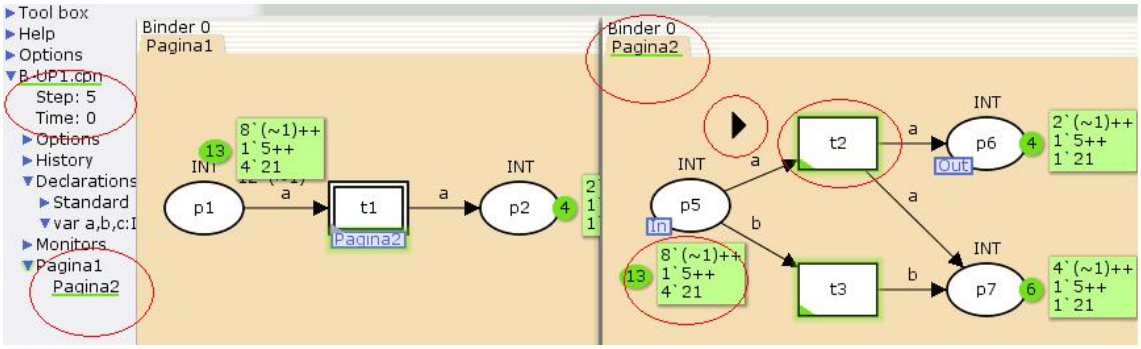

Figura 71. Realimentação durante uma simulação.

Uma bolha de status verde aparecerá quando a simulação for encerrada com sucesso. A razão da parada também será apresentada quando se passar o cursor sobre a bolha.

Simulação Usando a Ferramenta `Fast Forward`

Nenhuma realimentação é apresentada quando a simulação é realizada com a ferramenta `Fast Forward`. Ao final da simulação, a rede é atualizada com a última marcação alcançada.

Após iniciar a simulação com esta ferramenta, uma bolha de status roxa aparecerá, indicando que a simulação está em curso.

Escolha Manual das Ligações

É possível escolher qual ligação (variável-ficha) será feita para disparar uma transição. Isto é feito usando a ferramenta `Executes a transition with a chosen binding`, ou clicando no triângulo verde no canto inferior esquerdo de uma transição habilitada. Uma caixa de índice será aberta, mostrando as variáveis envolvidas no disparo da transição e os respectivos valores que elas podem assumir naquele instante. Uma interrogação (?) após a variável indica que nenhum valor ainda foi escolhido para aquela variável. Veja Figura 72.

Na caixa de índice só aparecerão os valores possíveis de serem ligados à variável naquele instante. Na Figura, a variável `a` pode assumir o valor -1 ou 21 ou 5, para o disparo de `t2`. Para disparar a transição, basta clicar em cima da caixa de índice, após a escolha do(s) valor(es). Após o disparo, a cor da caixa de índice mudará de azul para amarelo e nela só aparecerão os valores ligados às variáveis. Para fechar a caixa, basta clicar novamente em cima dela.

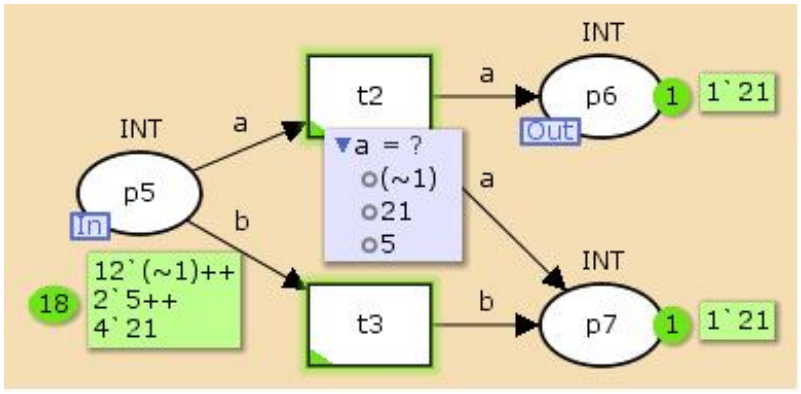

Figura 72. Escolha manual de uma ligação na hora do disparo de uma transição.

Relatório de Simulação

Um relatório de simulação é um arquivo texto que contém informações relativas às transições disparadas durante a simulação. Este arquivo é salvo em um diretório denominado `output`, que é um subdiretório da pasta onde foi salva a rede que foi simulada. Como padrão, o arquivo de relatório é salvo com o nome `simrep-x.txt` em que `x` é um número que é incrementado cada vez que um novo relatório de simulação é salvo.

O arquivo de simulação é salvo somente se a opção `Save Report` estiver marcada na entrada `Tool Box` → `Options` → `Simulation Report` da área de índice. Veja Figura 73.

Figura 73. Marcação da opção Save Report.

Para cada passo de simulação, é armazenada no arquivo uma linha, cuja sintaxe é:

`step time TransitionName @ (Instance:PageName)`

Um relatório de simulação mais detalhado será salvo se a opção `Save Bindings` estiver também marcada. Veja Figura 73.

A seguir, na Figura 74 é apresentado um exemplo de relatório de simulação com as opções `Save Report` e `Save Bindings` marcadas.

Na segunda linha do arquivo é apresentado o caminho de onde está armazenado o arquivo da rede que está sendo simulada. Na terceira linha são apresentas a data e a hora de criação do arquivo de simulação. Na quinta linha é especificado que no passo (step) 1 a transição `t3` ocorreu na instância 1 da página `Pagina2`. Na sexta linha é especificado que, quando `t3` ocorreu, a ficha de valor (-1) foi associada à variável `b` . Desta forma, nas próximas linhas são apresentadas as transições disparadas seguidas das variáveis associadas no disparo das mesmas.

Replicação de Simulações

```
CPN Tools simulation report for:
D:\Documentos\ApostilaCPNTools\Modelos\B-UP1.cpn
Report generated: Fri Oct 15 16:28:13 2010

1        0       t3 @ (1:Pagina2)
 - b = (~1)
2        0       t3 @ (1:Pagina2)
 - b = 5
3        0       t3 @ (1:Pagina2)
 - b = (~1)
4        0       t2 @ (1:Pagina2)
 - a = (~1)
5        0       t2 @ (1:Pagina2)
 - a = (~1)
```

Figura 74. Exemplo de relatório de simulação armazenado em arquivo txt.

Uma única simulação pode ser executada aplicando-se uma das ferramentas de simulação do palete `Sim`. Para iniciar uma nova simulação a ferramenta `Goes to the initial state` tem que ser aplicada.

Frequentemente, é útil fazer várias replicações de uma simulação. Por exemplo, quando se deseja fazer análise de desempenho, é quase sempre necessário executar várias simulações para se coletar dados estatísticos confiáveis.

A função `CPN'Replications.nreplications` pode ser usada para executar um certo número de simulações automaticamente. Aplicando-se a ferramenta `Evaluate ML` do palete `Sim` ao código apresentado na Figura 75, três simulações serão executadas (desde que algum critério de parada seja alcançado nas três simulações).

Figura 75. execução automática de 3 simulações.

Se a função for avaliada e em alguma das simulações um critério de parada não for encontrado, então a chamada da função nunca terminará. Neste caso, a única forma de parar a simulação é fechando o CPN Tools.

Erros Durante uma Simulação

Se um erro ocorrer durante uma simulação, uma mensagem de erro aparecerá. Veja Figura 76.

Neste caso, o conjunto de cores `SmallInt` possui somente cinco valores, variando de 1 a 5. No exemplo, toda vez que `t1` dispara, uma ficha com valor incrementado de uma unidade é colocada em `p1`. Quando esta ficha alcança o valor 6, a simulação é interrompida, pois este valor não pertence ao conjunto de cores `SmallInt`.

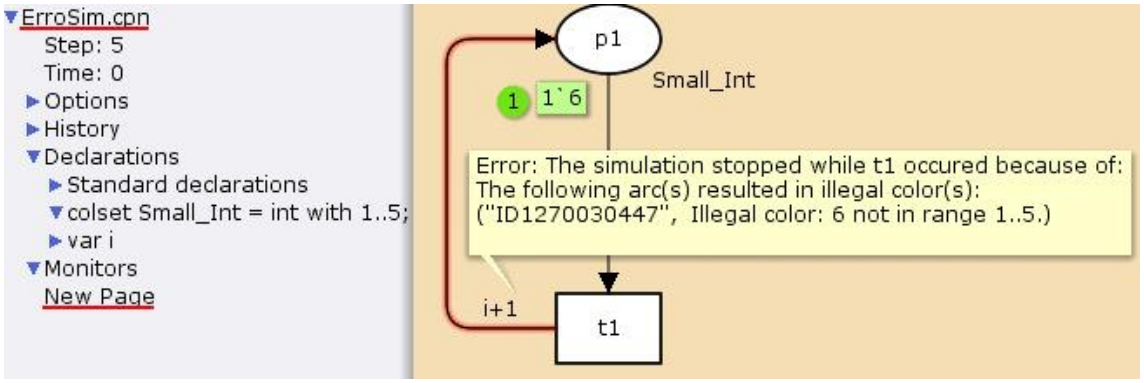

Figura 76. Ocorrência de erro durante uma simulação.

Referências

Goldberg, B. (1996). Functional programming languages. *ACM Computings Surveys*, 28(1):249–251.

Jensen, K., Christensen, S., and Kristensen, L. M. (2006). Cpn tools state space manual.

Jensen, K. and Kristensen, L. M. (2009). *Coloured Petri Nets, Modelling and Validation of Concurrent Systems*. Springer.

Murata, T. (1989). Petri nets: Properties, analysis and applications. *Proceedings of the IEEE*, 77(4):541–580.

Zaitsev, D. A. and Shmeleva, T. R. (2006). Simulating of telecommunication systems with cpn tools. Technical report, Odessa National Academy of Telecommunication, Department of Communication Networks.