4

Redes de Autômatos Estocásticos para Avaliação de Desempenho

Professores:

Afonso Sales [1]
(afonso.sales@pucrs.br)
Ricardo M. Czekster [2]
(ricardo.czekster@pucrs.br)
Paulo Fernandes [3]
(paulo.fernandes@pucrs.br)
Thais Webber [4]
(thais.webber@pucrs.br)

Resumo:

Avaliar o desempenho de sistemas é um ponto-chave para detectar gargalos de execução, possíveis otimizações e, sobretudo para prover aplicações com alto grau de confiabilidade e desempenho. Muitas são as formas para se avaliar o desempenho de sistemas, por exemplo, monitoração, medições, simulações e modelagem analítica. Muitas vezes, não se dispõe da aplicação em funcionamento para monitorá-la ou para realizar medições no ambiente real. As alternativas são então simulação, onde se imita o comportamento do sistema real, entretanto, o esforço de programação envolvido pode por vezes inviabilizar sua adoção. Neste sentido, a técnica da modelagem analítica descreve sistemas através de equações matemáticas como um conjunto de entradas e saídas em estados discretos, que em equilíbrio podem prover dados probabilísticos acerca das ocorrências destes estados. Esta forma de avaliação de desempenho tem sido difundida e

[1] Doutor em Informática pelo Institut National Polytechnique de Grenoble (2009). Mestre em Ciência da Computação (2003) e Bacharel em Informática (1999) pela PUCRS. Professor da Faculdade de Informática da PUCRS.

[2] Doutor (2010), Mestre (2006) e Bacharel (2002) em Ciência da Computação pela PUCRS. Realizou seu doutorado sanduíche no Laboratoire d'Informatique de Grenoble, França (2007/2008). Professor da Faculdade de Informática da PUCRS.

[3] Doutor em Informática pelo Institut National Polytechnique de Grenoble (1998). Mestre em Ciência da Computação pela UFRGS (1990) e Bacharel em Ciência da Computação pela mesma universidade (1987). Professor da Faculdade de Informática da PUCRS e coordenador do Programa de Pós-Graduação em Ciência da Computação na mesma universidade.

[4] Doutora (2009), Mestre (2003) e Bacharel (2001) em Ciência da Computação pela PUCRS. Realizou seu doutorado sanduíche no Laboratoire d'Informatique de Grenoble, França (2007/2008). Professora da Faculdade de Informática da PUCRS.

utilizada em diversos âmbitos da ciência da computação, biologia, engenharias, economia, entre outros. Na ciência da computação, descrevem-se modelos matemáticos (com características estocásticas, ou seja, aleatórias), principalmente no contexto de Redes de Computadores, para avaliar vazão, tempo de resposta, utilização de recursos, entre outros índices de desempenho. Um formalismo para modelagem estocástica vigente desde os anos 80, e que vem sofrendo extensões e melhorias nas últimas décadas, é o formalismo de Redes de Autômatos Estocásticos (conhecido como formalismo SAN - Stochastic Automata Networks). Neste curso, serão apresentados os principais fundamentos de modelagem deste formalismo, alguns casos práticos de aplicação de modelos descritos em SAN, sua relação com Cadeias de Markov, bem como o uso de ferramentas de apoio à solução destes modelos.

4.1. Introdução

Em última análise, avaliar um sistema é pronunciar-se sobre suas características. Dada uma problemática qualquer, sua avaliação pode ser definida como toda e qualquer observação feita sobre a mesma. Em princípio, toda avaliação tem por objetivo o estabelecimento de um julgamento qualitativo sobre o sistema avaliado. No entanto, toda avaliação científica é feita sobre resultados quantitativos. A aplicação prática da avaliação de desempenho é o conhecimento da situação (estado) do sistema avaliado em um dado momento, podendo observar a sua evolução ou o seu comportamento, e também avaliar situações futuras, com a finalidade de previsão e planejamento. Existem diferentes técnicas de avaliação de desempenho de sistemas, que são tradicionalmente divididas em três abordagens distintas [PID 92]: *monitoração*, *simulação* e *métodos analíticos*.

Monitoração: consiste na observação (monitoração) de sistemas reais, logo propicia maior fidelidade à realidade nos índices obtidos, pois não é feita nenhuma abstração (modelagem) do sistema em questão. Porém, há algumas desvantagens desta abordagem como, por exemplo, a necessidade da existência do sistema a ser avaliado. Isto pode gerar problemas em relação ao custo e ao tempo, pois o sistema implementado pode não satisfazer as necessidades, tendo que ser abandonado [PID 92]. Uma outra desvantagem é a questão da amostragem. É necessário que se faça o uso correto de técnicas de estatística para que os dados recolhidos tenham validade.

Simulação: consiste em construir um modelo que simule o funcionamento do sistema real. Este modelo deve descrever as características funcionais do sistema em uma escala adequada de tempo [PID 92] apenas com os detalhes importantes referentes ao sistema, *i.e.*, há um certo nível de abstração. Comparativamente à monitoração, a simulação costuma ser menos dispendiosa e consumir menos tempo para que os índices sejam calculados, permitindo que sejam feitos quantos experimentos forem necessários. Porém, por se tratar de uma abstração da realidade, a fidelidade das medidas tende a ser menor na simulação se compararmos com a monitoração. Além disso, da mesma forma que na monitoração, a quantidade e representatividade das amostras consideradas é muito importante para a obtenção de resultados corretos.

Métodos analíticos: assim como a simulação, também se baseia no desenvolvimento de um modelo do sistema real, porém com um nível de abstração mais alto. Neste caso, o modelo é puramente matemático. Neste tipo de modelo, o funcionamento do sistema real é reduzido a relações matemáticas. Modelos analíticos podem ser determinísticos ou estocásticos. Em um modelo determinístico, todos os parâmetros do sistema são previamente determinados. Já em um modelo estocástico, o comportamento do sistema é analisado probabilisticamente, ou seja, os parâmetros do sistema são descritos por variáveis aleatórias, com distribuições de probabilidade convenientes. No último caso, o sistema é descrito em termos de um conjunto de estados em que o mesmo pode se encontrar e de transições estocásticas entre esses estados (uma transição estocástica é aquela cuja ocorrência é descrita por uma variável aleatória [KLE 75]). Uma vantagem desta técnica em relação as outras descritas é que não há a necessidade de se preocupar com

um conjunto específico de amostras de funcionamento do sistema para a obtenção dos índices de desempenho. Desta forma, o domínio de métodos analíticos para avaliação de sistemas compreende um ferramental importante para profissionais da área de Ciência da Computação.

Este curso aborda o desenvolvimento de modelos analíticos, que normalmente exige maior abstração de aspectos da realidade, se comparado a modelos de simulação. Para o desenvolvimento de modelos normalmente adotam-se um ou mais formalismos, dependendo da realidade em estudo. Existem diferentes formalismos propostos com um conjunto de regras definidas e uma gramática para descrever um sistema de maneira não ambígua. Para o caso dos formalismos de descrição com vistas à análise de desempenho e soluções analíticas é comum verificar a existência de três entidades principais: estados, transições e eventos. Os diversos formalismos definidos na literatura são distinguidos entre si de muitas maneiras, entretanto, todos são fundamentalmente baseados no formalismo das Cadeias de Markov [STE 94]. Neste curso em particular, Redes de Autômatos Estocásticos serão abordadas. Conhecido como formalismo SAN - *Stochastic Automata Networks*), encontra-se vigente desde os anos 80, e este vem sofrendo extensões e melhorias nas últimas décadas. Serão apresentados os principais fundamentos de modelagem deste formalismo, alguns casos práticos de aplicação de modelos descritos em SAN, sua relação com Cadeias de Markov, bem como o uso de ferramentas de apoio à solução destes modelos.

Cabe salientar que dentre as soluções utilizadas para a resolução de sistemas através de formalismos, destacam-se principalmente os métodos numéricos e a simulação. Os métodos numéricos consistem, normalmente, no emprego de métodos matemáticos iterativos, tais como o *Método da Potência* [STE 2009], o *Método de Arnoldi* [ARN 51] e o *Método GMRES* (*Generalized Minimal RESidual method*) [SAA 86]. A simulação baseia-se geralmente na geração de números pseudo-aleatórios que atuam nas decisões relacionadas ao disparo de transições no modelo e na interação entre os componentes do sistema analisado. A simulação também consiste em uma aproximação da solução do modelo, sendo necessário executá-la por um tempo de simulação suficientemente grande para obter-se um conjunto de amostras significativas. Resumindo, para resolver um dado modelo, calcula-se o vetor de probabilidades correspondente à estacionariedade do sistema ou a algum período transiente. Pode-se dizer que o sistema chegou em um momento onde, dadas as suas taxas iniciais, este não mais evolui, atingindo um estado de equilíbrio. Dessa forma, extraem-se os índices de desempenho comparando-os com diferentes formas de modelar o problema, adicionando ou excluindo componentes, inspecionando as mudanças ocorridas nos índices. No caso de degradação dos índices, recomenda-se alterar as transições e as taxas, bem como os estados e os eventos. O objetivo é o de descobrir a melhor forma de composição do sistema para refletir a realidade estudada, propondo mudanças que impactarão diretamente em uma melhoria em termos de desempenho [CZE 2010].

4.2. Redes de Autômatos Estocásticos

O formalismo de *Redes de Autômatos Estocásticos* (SAN - *Stochastic Automata Networks*) é um formalismo estruturado baseado na teoria das Cadeias de Markov [STE 94]. Este formalismo foi inicialmente proposto por Brigitte Plateau em 1984 [PLA 84]. No começo da década de 90 foram formalizadas as primeiras soluções para modelos SAN em escala de tempo contínua e discreta [PLA 91]. Na virada do século, os conceitos básicos do formalismo SAN foram novamente revistos face aos eficientes algoritmos para modelos à escala de tempo contínua [FER 98]. Nesta mesma ocasião, foi disponibilizada a versão 2.0 da ferramenta PEPS (*Performance Evaluation of Parallel Systems*), a qual provê métodos iterativos[6] para resolução de modelos SAN.

A ideia principal do formalismo SAN é modelar um sistema em vários *subsistemas*, isto é, a visão do sistema é como se ele fosse composto por módulos "quase independentes", dando uma ideia de paralelismo ao seu comportamento. Logo, a expressão "quase independente" denota a possibilidade de ocorrer interação entre cada subsistema (*i.e.*, entre os módulos). Uma das principais vantagens na modelagem deste tipo de comportamento pelo formalismo SAN é a possibilidade de representar um sistema com um grande espaço de estados, *i.e.*, o problema modelado possui uma representação de milhões de estados. Essa característica de modularização de uma problemática qualquer pelo formalismo SAN permite o armazenamento e a solução eficiente de sistemas complexos por evitar os prejuízos da *explosão do espaço de estados* que ocorre no formalismo de Cadeias de Markov, o qual SAN tem equivalência de representação.

Cada subsistema (ou módulo) é representado por um *autômato estocástico* e por *transições* entre os estados deste autômato. As transições entre os estados de um autômato são modeladas por um processo estocástico de tempo *contínuo* ou *discreto* definidos respectivamente por distribuições *exponenciais* ou *geométricas*.

É interessante ressaltar que um modelo em SAN pode ser representado por um único autômato estocástico que contém todos os estados possíveis do sistema modelado. Esse único autômato corresponde à cadeia de Markov subjacente ao modelo em SAN[7].

Repare que a visão geral de modelagem através do formalismo SAN apresentada nesta seção aplica-se tanto à escala de tempo contínua como à escala de tempo discreta. Entretanto, as explicações e exemplos apresentados ao longo deste curso fazem referência somente à escala de tempo contínua (*i.e.*, *taxas de ocorrência*) e não à escala de tempo discreta (*i.e.*, *probabilidades de ocorrência*). A diferenciação entre as duas escalas de tempo ocorre apenas na obtenção do *descritor Markoviano*[8] de cada modelo. Enquanto um modelo em SAN à escala de tempo contínua gera uma cadeia de Markov à escala de tempo contínua (CTMC - *Continuous-Time Markov Chain*), um modelo em SAN descrito

[6] No que se refere à solução computacional de modelos, os métodos iterativos tais como *Método da Potência*, *Método de Arnoldi* e *GMRES*, implementados na ferramenta PEPS, são mais eficientes que os métodos diretos, como por exemplo *Eliminação de Gauss*, por aproveitar as características das técnicas de armazenamento esparsas e consequentemente não requerer tanta memória quanto os métodos diretos.

[7] Esta correspondência entre SAN e Cadeias de Markov será detalhada na Seção 4.2.5.1.

[8] Detalhes sobre a obtenção de um descritor Markoviano de um modelo em SAN podem ser encontrados na literatura a respeito de SAN [FER 98a, WEB 2009, SAL 2009, CZE 2010].

em escala de tempo discreta produz uma cadeia de Markov à escala de tempo discreta (DTMC - *Discrete Time Markov Chain*).

A seguir, nas próximas seções, cada primitiva de modelagem do formalismo SAN será explicada em detalhes.

4.2.1. Autômatos

Um *autômato estocástico* é basicamente um modelo matemático de um sistema que possui entradas e saídas discretas. O sistema pode se encontrar em qualquer um dentre o número finito de seus possíveis estados (ou das configurações internas). De fato, seu estado interno reuni todas as informações sobre as entradas anteriores e indica ainda o que é necessário para determinar o comportamento do sistema para as entradas seguintes [HOP 79].

Baseado nessa definição, pode-se descrever um autômato estocástico como um *conjunto finito de estados* e um *conjunto finito de transições* entre esses estados. O termo *estocástico* atribuído a esses autômatos dá-se pela razão do tempo ser tratado como uma *variável aleatória*, a qual obedece a uma distribuição *exponencial* para a escala de tempo *contínua* e uma distribuição *geométrica* para a escala de tempo *discreta*.

O *estado local* do sistema modelado em SAN é o estado individual de cada autômato do modelo. Em contra partida, o *estado global* do modelo é definido pela combinação dos estados locais de todos os autômatos que compõem o modelo. Desta forma, a mudança do estado global do sistema ocorre pela mudança do estado local de qualquer um dos autômatos do modelo.

A mudança de um estado local qualquer para outro é feita através de *transições* entre estes estados. As transições são primitivas que possibilitam a mudança entre um estado e outro. Um ou mais *eventos* podem ser associados às transições. Para que uma transição ocorra, é necessário que ocorra obrigatoriamente o disparo de um dos eventos associados a esta transição.

A fim de auxiliar na compreensão da representação gráfica de um modelo em SAN, a Figura 4.1 apresenta um modelo em SAN com dois autômatos ($\mathcal{A}^{(1)}$ e $\mathcal{A}^{(2)}$) completamente independentes, *i.e.*, não há interação entre estes autômatos.

Neste primeiro exemplo, o autômato $\mathcal{A}^{(1)}$ do modelo possui três estados locais representados pelas letras B, C e D. O autômato $\mathcal{A}^{(2)}$ também possui três estados locais, entretanto, representados pelas letras X, Y e Z. Desta forma, visto que o estado global de um modelo em SAN é definido pela combinação dos estados locais de todos os autômatos, os estados globais do modelo são obtidos pelas combinações entre os estados locais B, C e D do autômato $\mathcal{A}^{(1)}$ e os estados locais X, Y e Z do autômato $\mathcal{A}^{(2)}$. Logo, os nove estados globais do modelo em SAN da Figura 4.1 são: BX, BY, BZ, CX, CY, CZ, DX, DY e DZ.

O modelo em SAN apresentado na Figura 4.1 possui seis eventos: e_1, e_2, e_3, e_4, e_5 e e_6. Os eventos e_1, e_2 e e_3 ocorrem somente no autômato $\mathcal{A}^{(1)}$, enquanto os outros três eventos (e_4, e_5 e e_6) ocorrem somente no autômato $\mathcal{A}^{(2)}$.

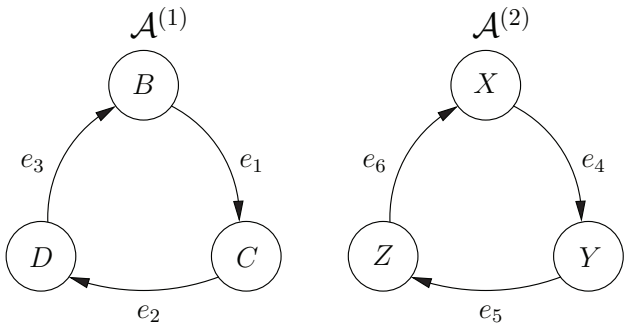


Figura 4.1: Modelo em SAN com dois autômatos independentes.

Todo evento em um modelo em SAN possui uma taxa de ocorrência associada a este evento. Na Tabela 4.1, apresenta-se as respectivas taxas de ocorrência para os eventos do modelo em SAN apresentado na Figura 4.1. Neste exemplo, as taxas de ocorrência são representadas pelos letra grega τ. De uma maneira geral, o valor de uma taxa de ocorrência de um evento é um valor real positivo e não nulo.

Evento	Taxa de ocorrência
e_1	τ_1
e_2	τ_2
e_3	τ_3
e_4	τ_4
e_5	τ_5
e_6	τ_6

Tabela 4.1: Taxas de ocorrência dos eventos do modelo em SAN da Figura 4.1.

Como comentado anteriormente, note que no modelo da Figura 4.1 não há interação entre os dois autômatos, *i.e.*, existe apenas *eventos locais* em cada um deles. Na próxima seção, será vista a definição e os tipos de eventos que podem ser utilizados nos modelos em SAN.

4.2.2. Eventos

Evento é uma primitiva de modelagem usada no modelo para representar a ocorrência de uma transição, a qual consequentemente muda o *estado global* do modelo. Um ou mais eventos podem estar associados a uma mesma transição e esta é disparada através da ocorrência de qualquer um destes eventos.

Basicamente, dois tipos de eventos podem ser utilizados nos modelos em SAN. Desta forma, um evento pode ser classificado como: *local* ou *sincronizante*.

4.2.2.1. Eventos Locais

Os eventos locais são utilizados nos modelos em SAN para alterar o estado local de *um único autômato*, sem que essa alteração ocasione qualquer mudança de estado em qualquer outro autômato do modelo. Esse tipo de evento é particularmente interessante, pois permite que vários autômatos tenham um comportamento paralelo, trabalhando independentemente sem que haja interação entre eles. Sendo assim, o uso dos eventos locais é que dão o caráter *paralelo* no comportamento do sistema modelado.

Note que a Figura 4.1 é composta exclusivamente por esse tipo de evento, pois o disparo de um destes eventos muda o estado de apenas um autômato do modelo. Logo, os eventos e_1, e_2, e_3, e_4, e_5 e e_6 da Figura 4.1 são todos eventos locais.

4.2.2.2. Eventos Sincronizantes

Ao contrário dos eventos locais, os eventos sincronizantes alteram o estado local de dois ou mais autômatos *simultaneamente*, *i.e.*, a ocorrência de um evento sincronizante em um autômato *força* necessariamente a ocorrência deste mesmo evento em todos os outros autômatos em que este evento apareça. Através do uso dos eventos sincronizantes, é possível fazer a interação entre autômatos. Essa interação ocorre sob a forma de sincronismo no disparo das transições. Sendo assim, o uso dos eventos sincronizantes na modelagem de um sistema dá o caráter *síncrono* no comportamento do modelo que o representa.

A classificação de um evento em *local* ou *sincronizante* é dada pela presença do identificador do evento no conjunto de eventos de um autômato. Caso o identificador do evento apareça apenas no conjunto de eventos de um único autômato, o evento é classificado como *evento local*. Este é o caso para os seis eventos apresentados na Figura 4.1. Entretanto, se o mesmo identificador aparecer no conjunto de eventos de *vários* autômatos, o evento é classificado como *evento sincronizante*. A fim de melhor exemplificar um evento sincronizante, a Figura 4.2 apresenta basicamente o mesmo modelo descrito anteriormente, porém, neste exemplo, o evento e_3 é classificado como evento *sincronizante*, visto que o identificador deste evento (e_3) aparece tanto no autômato $\mathcal{A}^{(1)}$ quanto no autômato $\mathcal{A}^{(2)}$.

O modelo em SAN apresentado na Figura 4.2 possui quatro eventos locais (e_1, e_2, e_4 e e_5) e um evento sincronizante (e_3). Desta maneira, o disparo do evento sincronizante e_3 força obrigatoriamente a ocorrência da transição do estado local D para o estado local B no autômato $\mathcal{A}^{(1)}$, ao mesmo tempo que ocorre a transição do estado local Z para o estado local X no autômato $\mathcal{A}^{(2)}$. Logo, pode-se dizer que houve uma sincronização entre os autômatos $\mathcal{A}^{(1)}$ e $\mathcal{A}^{(2)}$ partindo do estado global DZ para o estado global BX do modelo em SAN da Figura 4.2. Da mesma forma, pode-se afirmar que o evento sincronizante e_3 poderá ser disparado somente a partir do estado global DZ, *i.e.*, se o modelo encontrar-se no estado global DZ então o evento sincronizante e_3 poderá ser disparado, visto que ele se encontra apto a ser disparado em todos os autômatos em que ele se encontra. Como contra exemplo, tomamos o estado global DX. A partir deste estado global o evento sincronizante e_3 *encontra-se apto a disparar* no autômato $\mathcal{A}^{(1)}$ (partindo do estado

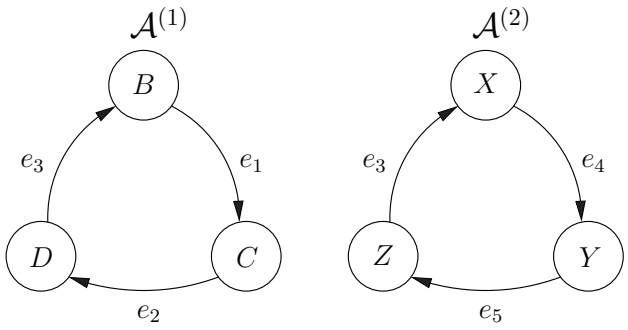


Figura 4.2: Modelo em SAN com um evento sincronizante (e_3).

local D), entretanto ele *não é disparável* no autômato $\mathcal{A}^{(2)}$ (partindo do estado local X). De onde conlui-se que para que um evento sincronizante seja disparado, este evento tem que *estar apto a ser disparável em todos* os autômatos os quais possuam o identificador deste evento sincronizante.

4.2.3. Probabilidades de rotação

As probabilidades alternativas de rotação (representadas de uma forma geral pela letra grega π) são utilizadas quando um evento tem *duas ou mais* alternativas de transição. Dessa maneira, probabilidades de rotação são utilizadas para indicar em que *proporções* o evento seguirá por uma transição ou por outra. A probabilidade de rotação pode ser omitida caso esta seja igual a 1, *i.e.*, caso haja somente uma alternativa de transição do evento entre os estados de origem e destino. No caso de duas ou mais probabilidades de rotação serem definidas, um ponto importante a ser considerado é que a soma destas probabilidades de rotação de um mesmo evento deve ser sempre igual a 1 (*i.e.*, as proporções devem somar 100%).

Um exemplo de uso de probabilidades de rotação é apresentado na Figura 4.3. Neste exemplo, o evento local e_4 possui duas alternativas de disparo partindo de um mesmo estado local (*i.e.*, estado local X). O evento e_4 pode disparar a transição partindo do estado local X para o estado local Y com uma probabilidade igual a π_1, ou este evento também pode disparar a transição que parte do estado local X para o estado local Z com uma probabilidade igual a π_2. Para melhor ilustrar o uso de probabilidades de rotação, vamos assumir que a taxa de ocorrência deste evento é igual a 10 (*i.e.*, $\tau_4 = 10$). E que as probabilidades de rotação π_1 e π_2 são iguais respectivamente a $0,4$ e $0,6$, onde $\pi_1 + \pi_2 = 0,4 + 0,6 = 1$). Logo, pode-se concluir que o evento e_4 ocorre com uma taxa igual a 4 (*i.e.*, $\tau_4 \times \pi_1 = 10 \times 0,4 = 4$) partindo do estado local X para o estado local Y, e este evento ocorre com uma taxa igual a 6 (*i.e.*, $\tau_4 \times \pi_2 = 10 \times 0,6 = 6$) partindo do estado local X para o estado local Z.

A aplicação do uso de probabilidades de rotação não é de exclusividade dos eventos locais. Assim como nos eventos locais, as probabilidades de rotação também podem

ser empregadas nos eventos sincronizantes, descrevendo com que proporção o evento seguirá cada possível transição em cada autômato. Do mesmo modo que descrito para os eventos locais, quando somente uma transição é possível partindo de um mesmo estado associada a um mesmo evento sincronizante, então pode-se omitir a probabilidade de rotação deste evento, visto que ela é igual a 1.

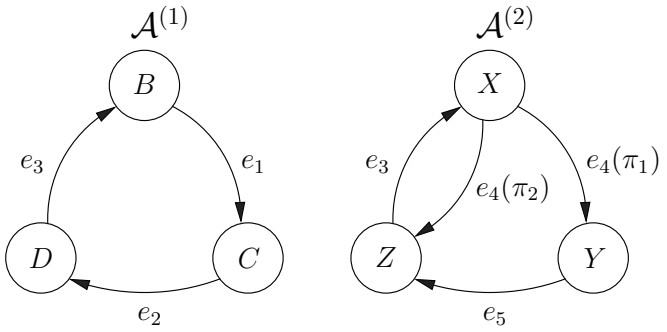


Figura 4.3: Modelo em SAN com probabilidades de rotação no evento e_4.

De uma maneira geral, todo evento deve possuir uma taxa de ocorrência e uma probabilidade de rotação associada a ele. Tanto a taxa de ocorrência como a probabilidade de rotação podem ter associados valores *constantes* ou valores *funcionais*. Taxas e probabilidades funcionais assumem valores diferentes conforme os estados dos outros autômatos do modelo.

4.2.4. Taxas ou Probabilidades Funcionais

A primeira, e aparente, possibilidade de interação entre os autômatos é através do uso de eventos sincronizantes. Taxas e probabilidades funcionais constituem a segunda possibilidade de interação entre autômatos nos modelos em SAN. A utilização de *funções* para definir taxas e/ou probabilidades permite associar a um mesmo evento diferentes valores conforme o estado global do modelo.

As taxas e probabilidades funcionais são expressas por funções que levam em consideração os estados atuais dos autômatos do modelo, podendo desta forma variar seu valor conforme os estados em que se encontram os autômatos envolvidos na função. Para exemplificar o uso de taxa funcional, considere o modelo em SAN da Figura 4.4. Neste exemplo, o evento local e_1 no autômato $\mathcal{A}^{(1)}$ não possui mais um valor constante associado a sua taxa (*i.e.*, o valor constante τ_1), mas possui uma função f que descreve qual é a taxa de ocorrência deste evento de acordo com o estado local do autômato $\mathcal{A}^{(2)}$.

Na Tabela 4.2, apresenta-se as respectivas taxas de ocorrência para os eventos do modelo em SAN apresentado na Figura 4.4.

A função f associada à taxa de ocorrência do evento local e_1 do autômato $\mathcal{A}^{(1)}$ é

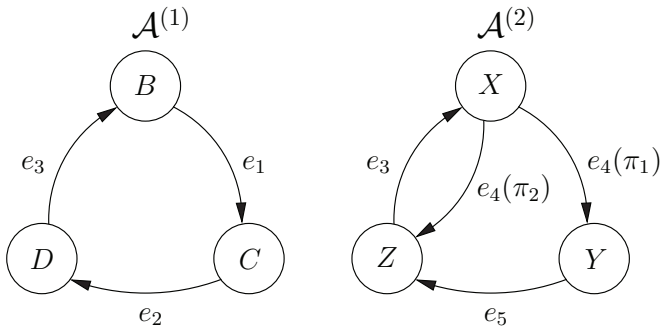


Figura 4.4: Modelo em SAN com taxa funcional no evento e_1.

Tipo	Evento	Taxa de ocorrência
loc	e_1	f
loc	e_2	τ_2
syn	e_3	τ_3
loc	e_4	τ_4
loc	e_5	τ_5
loc	e_6	τ_6

Tabela 4.2: Taxas de ocorrência dos eventos do modelo em SAN da Figura 4.2.

definida como:

$$f = \begin{cases} 0 & \text{se autômato } \mathcal{A}^{(2)} \text{ está no estado } X; \\ \lambda & \text{se autômato } \mathcal{A}^{(2)} \text{ está no estado } Y; \\ \gamma & \text{se autômato } \mathcal{A}^{(2)} \text{ está no estado } Z. \end{cases}$$

Como pode-se observar na definição da função f, a taxa de ocorrência da transição do estado local B para o estado local C no autômato $\mathcal{A}^{(1)}$ é igual a λ caso o autômato $\mathcal{A}^{(2)}$ esteja no estado local Y. Isso equivale a dizer que a transição do estado local B para o estado local C no autômato $\mathcal{A}^{(1)}$ ocorre com taxa igual a λ se o modelo em SAN se encontrar no estado global BY, visto que a taxa de ocorrência do evento local e_1 é igual a λ somente quando o estado local do autômato $\mathcal{A}^{(2)}$ for igual a Y. Da mesma forma que a taxa de transição do estado local B para o estado local C no autômato $\mathcal{A}^{(1)}$ é igual a γ quando o autômato $\mathcal{A}^{(2)}$ estiver no estado local Z. E a transição do estado local B para o estado local C *não ocorrerá* (*i.e.*, a taxa de ocorrência do evento será igual a 0) caso o autômato $\mathcal{A}^{(2)}$ esteja no estado local X. Da mesma forma, pode-se dizer que quando o modelo estiver no estado global BX, a avaliação da função f será igual a 0 (visto que o autômato $\mathcal{A}^{(2)}$ estará no estado local X) e consequentemente o evento local e_1 não ocorrerá, impossibilitando a transição entre os estados locais B e C no autômato $\mathcal{A}^{(1)}$.

Assim como as taxas de ocorrência podem ser expressas por funções, o mesmo pode ocorrer com as probabilidades alternativas de rotação de cada evento. A definição de funções usadas para expressar as probabilidades funcionais de rotação são exatamente

iguais as funções usadas para definir as taxas de ocorrência. Sendo assim, as probabilidades de rotação também podem ter valores que variam de acordo com o estado local de outros autômatos.

4.2.5. Atingibilidade

Devido à natureza dos eventos e dada a dinâmica do modelo, os modelos estruturados geralmente apresentam combinações de estados que *não são válidas* ou que *não são atingíveis* dado um estado inicial do modelo. Esta computação de quais são os *estados atingíveis* de um modelo dado um estado inicial nem sempre é uma tarefa simples de ser executada.

O *espaço de estados produto* (PSS - *Product State Space*) de um modelo em SAN é facilmente calculado pelo produto cartesiano de todos os espaços de estados locais dos autômatos. Estretanto, dado um estado inicial do modelo, pode-se calcular o *espaço de estados atingíveis* (RSS - *Reachable State Space*) do modelo a partir de sucessivos disparos dos eventos até que nenhum outro novo estado atingível seja encontrado.

Para exemplificar a noção de atingibilidade, tomamos como exemplo o modelo de compartilhamento de recursos, onde se tem N processos disputando R recursos. Este sistema pode ser modelado em SAN usando um autômato com dois estados para cada processo: um estado que representa que o recurso *não está sendo utilizado* pelo processo, enquanto o outro estado representa que o recurso *está em uso* pelo processo. É fácil imaginar que, se o número R de recursos for menor do que o número N de processos, o estado global deste modelo que representa todos os processos utilizando um recurso *nunca poderá ser atingido*, visto que este estado não corresponde à realidade do modelo. Desta forma, o RSS deste modelo é um subconjunto do seu PSS e os estados que se encontram no PSS e não se encontram no RSS possuem probabilidade igual a *zero*, pois o modelo jamais vai se encontrar nestes estados globais.

Recentemente, foi proposto um método de geração do espaço de estados atingíveis de um modelo em SAN partindo de um estado inicial [SAL 2009a, SAL 2009]. Este método se apresentou muito eficaz tanto em termos de memória utilizada para armazenamento do RSS quanto em termos de tempo de execução para a computação do RSS de modelos com grande espaço de estados. Esta eficiência em termos de memória e tempo só foi possível graças ao uso de *Diagramas de Decisão Multi-valorados* (MDD - *Multi-valued Decision Diagrams*) [KAM 98], os quais permitem operações como união e intersecção de maneira eficiente sobre conjuntos de espaço de estados.

4.2.5.1. Cadeia de Markov Subjacente

Uma vez descoberto todos os estados atingíveis de um modelo em SAN, é possível representar este modelo estruturado por um único autômato conhecido como *autômato global*. Este autômato é justamente a representação da *cadeia de Markov subjacente* ao modelo, onde ao invés de eventos as transições entre os estados da cadeia de Markov possuem diretamente os valores das taxas/probabilidades associadas aos eventos no modelo.

Para melhor exemplificar a obtenção da cadeia de Markov subjacente, tomamos como exemplo o modelo em SAN da Figura 4.4. Dado o estado global BX como estado inicial deste modelo, realizando sucessivos disparos dos eventos do modelo, é possível descobrir sete estados atingíveis dos nove estados globais possíveis do modelo. A Figura 4.5 apresenta a cadeia de Markov subjacente do modelo da Figura 4.4.

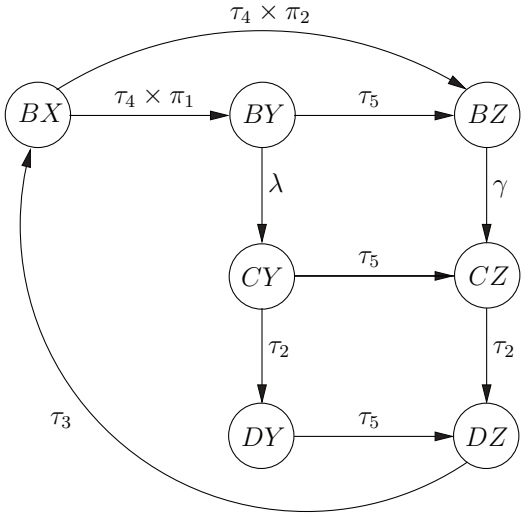


Figura 4.5: Cadeia de Markov subjacente do modelo em SAN da Figura 4.4.

Como pode ser observado na Figura 4.5, somente os sete estados globais (BX, BY, BZ, CY, CZ, DY, DZ) são atingíveis. Os estados globais CX e CZ não são atingíveis a partir do estado inicial BX do modelo. Note também que na cadeia de Markov não há mais a noção de evento associada às transições. Desta forma, por exemplo, ao invés do evento e_5 estar associado à transição que leva do estado global CY para o estado global CZ, o valor da taxa (τ_5) deste evento é que está associado à esta transição. Da mesma forma, este fato ocorre nas transições que levam do estado global BX para os estados globais BY e BZ, onde o valor da taxa (τ_4) do evento e_4 é multiplicado, respectivamente, pelas probabilidades de rotação π_1 e π_2.

É também interessante de ressaltar os valores de taxa associado ao evento e_1, visto que este evento possui uma taxa funcional. Neste caso, como pode ser visto na cadeia de Markov da Figura 4.5, a função f quando avaliada no estado global BY resulta no valor λ, o qual é associado à transição que leva do estado global BY para o estado global CY. De maneira análoga, a função f quando avaliada no estado global BZ resulta no valor γ, o qual é associado à transição que leva do estado global BZ para o estado global CZ. E por esta mesma razão que os estados globais BY e BZ são inatingíveis, pois a função f quando avaliada no estado global BX resulta no valor 0 (*zero*). Consequentemente, não há transição entre os estados globais BX e CX, da mesma maneira que também não há entre os estados globais CX e CZ na cadeia de Markov subjacente do modelo em SAN da Figura 4.4.

4.2.6. Funções de Integração

Define-se *funções de integração* para a obtenção de resultados numéricos sobre o modelo em SAN. As funções de integração avaliam qual a probabilidade do modelo em SAN encontrar-se em um ou mais estados globais.

Desta forma, pode-se compor funções de integração que levem em conta a probabilidade do modelo se encontrar em um conjunto de estados, podendo assim se obter índices de desempenho e confiabilidade do modelo. Essas funções de integração são avaliadas sobre o *vetor de probabilidades*[9] π que contém a probabilidade do modelo se encontrar em cada um dos estados globais e a soma dos valores de todos as posições deste vetor é igual a 1. Ainda assumindo como exemplo o modelo em SAN da Figura 4.4, o vetor de probabilidade π teria nove posições, cada uma associada a um estado global do modelo, onde as posições que representam os estados globais CX e DX teriam o valor 0 (*zero*), visto que estes estados são inatingíveis:

	BX	BY	BZ	CX	CY	CZ	DX	DY	DZ
π				0			0		

Tendo em mente o modelo em SAN da Figura 4.4, assumimos f_B como a função de integração que expressa a probabilidade do autômato $\mathcal{A}^{(1)}$ estar no estado local B. A probabilidade resultante de f_B é calculada pela integração das probabilidades do vetor π nos índices onde o estado local B compõe o estado global do modelo. Desta forma, seria a simples soma das probabilidades do vetor π nos índices BX, BY e BZ:

$$f_B = \pi[BX] + \pi[BY] + \pi[BZ]$$

Na seção seguinte, apresentamos as ferramentas de solução para os modelos em SAN, de forma que uma vez dada sua descrição, o vetor de probabilidade π é computado e consequentemente as funções de integração são calculadas automaticamente.

4.3. Ferramentas de Solução

Solucionar modelos em SAN não é uma tarefa trivial e o uso de ferramentas especializadas para a obtenção da solução destes modelos torna-se obrigatório. Existem duas ferramentas que são utilizadas para a solução de modelos em SAN:

- *Performance Evaluation of Parallel Systems* (PEPS) [BRE 2007];
- SAN LITE-SOLVER [SAL 2011].

[9]Não confunda as probabilidades de rotação π_1 ou π_2 do exemplo com o vetor de probabilidade π dos estados globais do modelo. O vetor de probabilidade π caracteriza a distribuição de probabilidade do modelo.

4.3.1. PEPS

PEPS é a ferramenta mais antiga e consequentemente a mais utilizada para a solução de modelos em SAN. PEPS provê tanto a solução estacionária quanto solução transiente para os modelos. Além do mais, esta ferramenta também permite o uso de métodos iterativos avançados, tais como método da *Potência* [STE 94], *Arnoldi* [ARN 51] e GMRES [SAA 86], para a solução dos modelos em SAN, bem como métodos de simulação, tais como *Amostragem Perfeita* [FER 2008] e *Simulação baseada em Bootstrap* [CZE 2010a].

A solução aproximada aplicada pelo PEPS consiste em um processo iterativo da multiplicação de um vetor de probabilidade por uma estrutura não-trivial (conhecida como *descritor de Kronecker*). Este processo iterativo é conhecido como *Multiplicação Vetor-Descritor* (*Vector-Descriptor Product* - VDP).

PEPS é uma ferramenta composta de três módulos correspondentes a cada fase do processo de avaliação de desempenho do modelo: *descrição*, *compilação* e *solução*. A descrição de um modelo em SAN será detalhada na Seção 4.3.3.

Detalhes sobre a ferramenta de solução PEPS podem ser encontrados na página *http://www-id.imag.fr/Logiciels/peps/index.html*.

4.3.2. SAN LITE-SOLVER

SAN LITE-SOLVER é uma ferramenta de linha de comando, simples e fácil de usar para a solução de modelos em SAN. A principal ideia empregada na solução dos modelos é que ao invés de multiplicar um vetor de probabilidades por um descritor, a cadeia de Markov subjacente ao modelo em SAN é calculada e então o processo iterativo de multiplicação é através de um vetor de probabilidades por uma matriz esparsa (*i.e.*, a cadeia de Markov subjacente). Este processo é conhecido como *Multiplicação Vetor-Matriz* (*Vector-Matrix Product* - VMP).

O cálculo da cadeia de Markov subjacente do modelo em SAN só é possível de ser realizado de maneira eficiente [SAL 2009a] graças ao uso dos *Diagramas de Decisão Multi-valorados* (MDD - *Multi-valued Decision Diagrams*) [KAM 98]. Uma vez calculado o espaço de estados atingíveis (RSS) de um modelo em SAN, ele é armazenado em um MDD e utilizado para o cálculo de todas as transições do modelo, desta forma, construindo a cadeia de Markov subjacente.

Ao contrário da ferramenta PEPS, SAN LITE-SOLVER prove apenas a solução estacionária do modelo em SAN utilizando o método iterativo da Potência. Na verdade, a principal ideia desta ferramenta é facilitar o uso do modelador na obtenção da solução tão rápida quanto possível de um modelo em SAN. Quando comparada com a ferramenta PEPS, SAN LITE-SOLVER gasta em alguns casos mais memória para armazenar a cadeia de Markov subjacente do modelo (visto que ela é representada por uma matriz esparsa ao invés de um descritor), entretanto permite empregar uma abordagem *mais leve* para solucionar os modelos em SAN (*i.e.*, aplica-se a multiplicação vetor-matriz ao invés de vetor-descritor).

Detalhes sobre a ferramenta de solução SAN LITE-SOLVER podem ser encontrados na página *http://www.inf.pucrs.br/afonso.sales/san-lite-solver*.

4.3.3. Formatação de modelo em SAN

Nesta seção, apresentamos a formatação de um arquivo que representa um modelo em SAN e que é utilizado pelas ferramentas de solução PEPS e SAN LITE-SOLVER. Basicamente, um modelo descrito em SAN possui cinco seções:

- **identifiers**: definição dos identificadores e domínios utilizados no modelo;
- **events**: definição dos eventos;
- **partial reachability**: definição de um estado global inicial do modelo;
- **network**: definição da rede de autômatos;
- **results**: definição das funções de integração.

Para ilustrar a formatação de um modelo em SAN, tomamos como exemplo o modelo em SAN da Figura 4.4. Desta forma, na seção dos identificadores do modelo, são declarados os identificadores para as taxas dos eventos, *i.e.*, são declarados os valores para τ_2, τ_3, τ_4, τ_5, bem como a descrição da função f (*i.e.*, a taxa funcional do evento e_1) e os respectivos valores de λ, γ, π_1 e π_2.

Definição dos identificadores e domínios

```
identifiers

  lambda = 1;  // taxa do evento e1 se A2 estiver em Y
  gamma = 6;   // taxa do evento e1 se A2 estiver em Z
  tau2 = 2;    // taxa do evento e2
  tau3 = 3;    // taxa do evento e3
  tau4 = 4;    // taxa do evento e4
  tau5 = 5;    // taxa do evento e5

  pi1 = 0.4;   // probabilidade de rotação de X para Y
  pi2 = 0.6;   // probabilidade de rotação de X para Z

  // função da taxa funcional de e1
  f = ((st A2 == Y) * lambda) + ((st A2 == Z) * gamma);
```

Note que "st" é uma palavra-reservada utilizada em conjunto com o identificador do autômato. Neste caso, "st A2" identifica qual o estado do autômato $\mathcal{A}^{(2)}$. Logo, quando declara-se "st A2 == Y" deseja-se saber se o autômato $\mathcal{A}^{(2)}$ encontra-se no estado local Y. Da forma como f está declarada, existem três possíveis resultados para a função f:

- valor de lambda – caso o autômato $\mathcal{A}^{(2)}$ esteja em Y;
- valor de gamma – caso o autômato $\mathcal{A}^{(2)}$ esteja em Z;
- valor 0 (*zero*) – pois o autômato $\mathcal{A}^{(2)}$ está em X e consequentemente as avaliações de "st A2 == Y" e "st A2 == Z" serão iguais a 0.

Após a definição dos identificadores, define-se o conjunto de eventos do modelo em SAN.

Definição dos eventos

```
events

  loc e1 (f);      // evento local com taxa funcional definida em f
  loc e2 (tau2);   // evento local com taxa definida em tau2
  syn e3 (tau3);   // evento sincronizante com taxa definida em tau3
  loc e4 (tau4);   // evento local com taxa definida em tau4
  loc e5 (tau5);   // evento local com taxa definida em tau5
```

Note que foi associado como taxa de ocorrência ao evento e_1 a função f, a qual define a taxa funcional deste evento. Para os demais eventos e_2, e_3, e_4 e e_5 foram associadas como taxas os respectivos identificadores *tau2*, *tau3*, *tau4* e *tau5* correspondentes a τ_2, τ_3, τ_4 e τ_5 do modelo em SAN.

A definição do estado global inicial do modelo em SAN é feita na seção *partial reachability*.

Definição do estado global inicial

```
partial reachability = ((st A1 == B) && (st A2 == X));
```

Como pode ser visto, definiu-se como estado global inicial do modelo em SAN da Figura 4.4 o estado BX, onde o autômato $\mathcal{A}^{(1)}$ deve estar no estado local B e o autômato $\mathcal{A}^{(2)}$ no estado local X. A escolha deste estado foi meramente arbitrária, pois um outro estado atingível qualquer do modelo poderia ter sido escolhido para ser o estado global inicial. Uma vez determinado os identificadores, taxas, eventos e o estado inicial do modelo, define-se a rede de autômtatos propriamente dita.

Definição da rede de autômatos

```
network ModeloSAN (continuous)

  aut A1
    stt B to (C) e1
    stt C to (D) e2
    stt D to (B) e3

  aut A2
```

```
    stt X to (Y) e4(pi1)
          to (Z) e4(pi2)
    stt Y to (Z) e5
    stt Z to (X) e3
```

Na seção *networks*, note que foi empregada a palavra-reservada "*continuous*" que caracteriza que o modelo em SAN está descrito em tempo contínuo. Em relação a tempo discreto, a palavra-reservada "*discrete*" deverá ser utilizada, entretanto, a modelagem em SAN em tempo discreto não se encontra disponível tanto na ferramenta PEPS quanto na ferramenta SAN LITE-SOLVER. Neste exemplo, a palavra *ModeloSAN* é a nome utilizado no modelo em SAN apresentado.

Também é importante ressaltar outras três palavras-reservadas: "aut", "stt" e "to". A palavra "aut" é utilizada para determinar o nome do autômato, "stt" é empregado para determinar o nome do estado local no autômato, enquanto "to" determina para qual estado ocorrerá a transição. Por exemplo, a definição do autômato $\mathcal{A}^{(1)}$ é dada por "aut A1" e a transição entre o estado local B para o estado local C através da ocorrência do evento e_1 é descrita em "stt B to (C) e1".

Note que no autômato $\mathcal{A}^{(2)}$ (*i.e.*, "aut A2") há o emprego das probabilidades de rotação no evento e_4. Desta forma, a transição que parte do estado local X para o estado local Y ocorre através do evento e_4 com probabilidade π_1 (*i.e.*, "e4(pi1)"), enquanto que do estado local X para o estado local Y também ocorre através do evento e_4 mas com probabilidade π_2 (*i.e.*, "e4(pi2)").

E por fim são definidas as funções de integração, as quais apresentam as probabilidades resultantes do estados do modelo de maneira integrada.

Definição das funções de integração

```
results

  BX = ((st A1 == B) && (st A2 == X));
  BY = ((st A1 == B) && (st A2 == Y));
  BZ = ((st A1 == B) && (st A2 == Z));
  CX = ((st A1 == C) && (st A2 == X));
  CY = ((st A1 == C) && (st A2 == Y));
  CZ = ((st A1 == C) && (st A2 == Z));
  DX = ((st A1 == D) && (st A2 == X));
  DY = ((st A1 == D) && (st A2 == Y));
  DZ = ((st A1 == D) && (st A2 == Z));
```

Nove resultados (*i.e.*, funções de integração) foram criados neste exemplo. Esses resultados nos dizem a probabilidade de cada estado global do modelo em SAN. Por exemplo, a função de integração descrita em BX determina a probabilidade do modelo se encontrar no estado local B no autômato $\mathcal{A}^{(1)}$ e no estado local X no autômato $\mathcal{A}^{(2)}$, *i.e.*, a probabilidade do modelo se encontrar no estado global BX em regime estacionário.

Executando este exemplo tanto na ferramenta PEPS quanto no SAN LITE-SOLVER, os resultados das funções de integração são:

BX = 0,1909
BY = 0,0509
BZ = 0,1188
CX = 0,0000
CY = 0,0073
CZ = 0,3746
DX = 0,0000
DY = 0,0029
DZ = 0,2546

Como pode ser observado nos resultados das funções de integração, e como esperado dado os estados atingíveis do modelo, os estados globais CX e DX possuem probabilidades igual a *zero*, visto que são estados inatingíveis do modelo em SAN, assumindo como estado inicial o estado global BX.

A seguir, apresentamos alguns exemplos de modelos em SAN e seus respectivos arquivos formatados para o formalismo SAN, de forma a melhor ilustrar o uso e a modelagem do formalismo de Redes de Autômatos Estocásticos.

4.4. Exemplos de Modelagem

Esta seção apresenta alguns exemplos de modelagem em SAN visando aplicar os conceitos e as primitivas de modelagem vistas na Seção 4.2. Desta forma, é feita a apresentação dos modelos em SAN de cada exemplo exposto, bem como a respectiva formatação textual dos modelos.

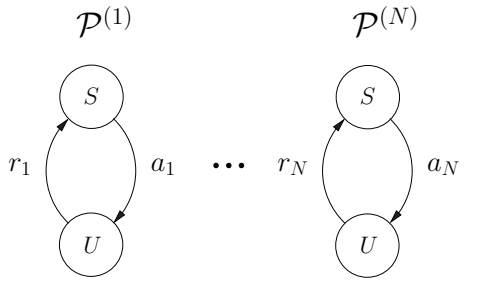
Apresenta-se um modelo de compartilhamento de recurso na Seção 4.4.1. Em seguida, é apresentado o modelo de comunicação com barramento exclusivo na Seção 4.4.2. E por fim apresenta-se a modelagem de um projeto de desenvolvimento de software globalmente distribuído na Seção 4.4.3.

4.4.1. Compartilhamento de Recursos (CR)

O modelo de *Compartilhamento de Recursos*[10] (CR), é bastante utilizado na avaliação de como N processos compartilham R recursos, *i.e.*, qual a probabilidade de ocupação dos recursos dadas taxas de requisição e liberação dos mesmos. A Figura 4.6 representa um sistema de compartilhamento de recursos onde cada processo é representado por um autômato $\mathcal{P}$ composto por dois estados: S (dormindo, ou *sleeping*) e U (usando ou *using*).

A liberação de um recurso pelo processo (mudança do estado U para o estado S) é representada por um evento local r, esse evento ocorre com taxa μ de forma completamente independente, ou seja, a liberação de um recurso não depende do estado do sistema

[10]Do inglês *Resource Sharing*.



Tipo	Evento	Taxa	Tipo	Evento	Taxa
loc	a_1	f_1	loc	r_1	μ_1
⋮	⋮	⋮	⋮	⋮	⋮
loc	a_N	f_N	loc	r_N	μ_N

$$f_i = (nb\ [\mathcal{P}^{(1)}..\mathcal{P}^{(N)}]\ U < R) * \lambda_i$$

Figura 4.6: Modelo de *Compartilhamento de Recursos* em SAN.

(*i.e.*, estado dos demais autômatos). Por outro lado, a alocação de um recurso (mudança do estado S para o estado U) é representada por um evento local a, o qual é disparado por uma taxa de ocorrência funcional (f). É esta taxa f quem regula (modela) a contenção de recursos. Esta função deve resultar um valor *nulo* (valor igual a *zero*) quando não houver recurso disponível e uma taxa não-nula (valor definido em λ) caso contrário. Nesta função, a palavra-reservada "nb" indica a quantidade de autômatos que se encontram em um determinado estado. Neste exemplo, na função f, "nb" verifica dentre os autômatos do modelo (*i.e.*, do primeiro autômato $\mathcal{P}^{(1)}$ até o último autômato $\mathcal{P}^{(N)}$) quantos se encontram no estado S e se este valor é inferior ao número de recursos disponíveis R. Se esta condição é satisfeita (*i.e.*, retorna o valor 1), então o resultado é multiplicado pela taxa λ do evento. Este modelo descreve toda a interação entre os autômatos através da taxa funcional (f), logo não foi necessário utilizar nenhum evento sincronizante.

Note que neste modelo é representado de forma compacta 2^N estados globais, entretanto nem todos estes estados são *atingíveis*[11]. Na verdade, todos os estados globais cuja composição de estados locais represente mais de R recursos utilizados são considerados inatingíveis.

Para exemplificar a formatação deste modelo em SAN, tomamos como exemplo $N = 3$ processos que compartilham $R = 2$ recursos.

Exemplo de arquivo do modelo CR (N=3, R=2)

```
identifiers

  // Número de recursos
  R = 2;
  // Taxas de alocação
  lambda1 = 1;
  lambda2 = 1;
  lambda3 = 1;
  // Funções de alocação
  f1 = (nb [P1..P3] U < R) * lambda1;
  f2 = (nb [P1..P3] U < R) * lambda2;
  f3 = (nb [P1..P3] U < R) * lambda3;
```

[11] O número de estados atingíveis para um modelo com N processos compartilhando R recursos é igual a $\sum_{i=0}^{R} \binom{N}{i}$, onde $\binom{N}{i}$ representa o número de combinações não ordenadas de i elementos tirados de um conjunto contendo um total de N elementos.

```
  // Taxas de liberação
  mu1 = 2;
  mu2 = 2;
  mu3 = 2;

events

  loc a1 (f1);
  loc a2 (f2);
  loc a3 (f3);
  loc r1 (mu1);
  loc r2 (mu2);
  loc r3 (mu3);

partial reachability = ((st P1 == S) && (st P2 == S) && (st P3 == S));

network RS3_2 (continuous)

aut P1
  stt S to (U) a1
  stt U to (S) r1

aut P2
  stt S to (U) a2
  stt U to (S) r2

aut P3
  stt S to (U) a3
  stt U to (S) r3

results

  P1EmUso = (st P1 == U);
  P2EmUso = (st P2 == U);
  P3EmUso = (st P3 == U);
```

4.4.2. Comunicação com Barramento Exclusivo (CBE)

O modelo *Comunicação com Barramento Exclusivo* (CBE) representa um modelo de comunicação em rede que se dá através de um único barramento. O objetivo é avaliar como N nodos podem transmitir e receber mensagens neste barramento com as restrições impostas. A Figura 4.7 ilustra o esquema de conexão entre os nodos e o barramento. Somente é possível transmitir mensagens entre dois nodos quando o nodo receptor estiver em estado ocioso (*i.e.*, livre para recepção de mensagens), e se nenhum outro nodo estiver transmitindo ou recebendo mensagens via barramento.

A Figura 4.8 apresenta o modelo SAN com $N = 3$ autômatos que representam os nodos em seus possíveis estados na conexão. Cada autômato *Nodo*$^{(i)}$ é composto por três estados: T (transmitindo mensagens), O (ocioso) e R (recebendo mensagens).

Na Tabela 4.3, apresenta-se as respectivas taxas de ocorrência para os eventos do modelo em SAN apresentado na Figura 4.8.

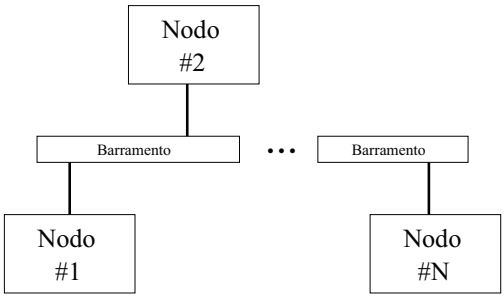


Figura 4.7: Esquema de Conexão no Barramento Exclusivo.

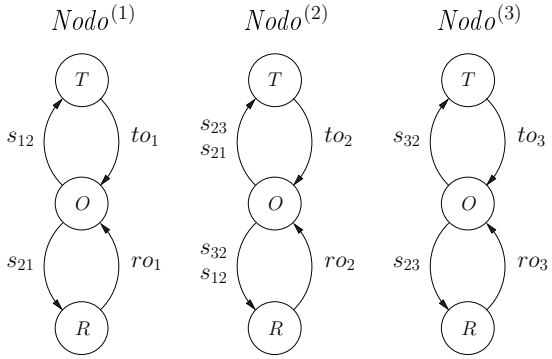


Figura 4.8: Modelo de *Comunicação com Barramento Exclusivo* em SAN ($N = 3$).

Eventos locais (to e ro) quando disparados representam o término da recepção ou transmissão de mensagens, neste modelo com taxas constantes β. Eventos sincronizantes s_{kl} sincronizam transmissões e recepções de mensagens entre dois nodos (*Nodo*$^{(k)}$ e *Nodo*$^{(l)}$), cujas taxas de disparo α_{kl} dependem da avaliação de funções f_{kl}. Estas funções verificam se o barramento está livre (*i.e.*, se ninguém está recebendo ou transmitindo mensagens). Mais especificamente, as funções verificam se todos os autômatos que representam os nodos se encontram no estado O (ocioso).

A seguir, apresenta-se a descrição de um modelo em SAN para este exemplo de comunicação com barramento exclusivo para $N = 3$ nodos.

Exemplo de arquivo do modelo CBE (N=3)

Tipo	Evento	Taxa de ocorrência	Tipo	Evento	Taxa de ocorrência
loc	to_1	β_1	syn	s_{12}	$f_{12} = (nb\ [Nodo^{(1)}..\ Nodo^{(3)}]\ O == N) * \alpha_{12}$
loc	to_2	β_2	syn	s_{23}	$f_{23} = (nb\ [Nodo^{(1)}..\ Nodo^{(3)}]\ O == N) * \alpha_{23}$
loc	to_3	β_3	syn	s_{32}	$f_{32} = (nb\ [Nodo^{(1)}..\ Nodo^{(3)}]\ O == N) * \alpha_{32}$
loc	ro_1	β_4	syn	s_{21}	$f_{21} = (nb\ [Nodo^{(1)}..\ Nodo^{(3)}]\ O == N) * \alpha_{21}$
loc	ro_2	β_5			
loc	ro_3	β_6			

Tabela 4.3: Taxas de ocorrência dos eventos do modelo em SAN da Figura 4.8.

```
identifiers
  // Número de nodos ligados ao barramento
  N = 3;
  // Taxas de fim de transmissão/recepção
  taxa_beta = 2;
  beta1 = taxa_beta;
  beta2 = taxa_beta;
  beta3 = taxa_beta;
  beta4 = taxa_beta;
  beta5 = taxa_beta;
  beta6 = taxa_beta;
  // Taxa de início de comunicação
  alpha12 = 1;
  alpha23 = 1;
  alpha32 = 1;
  alpha21 = 1;

  f12 = (nb [Nodo1..Nodo3] O == N) * alpha12;
  f23 = (nb [Nodo1..Nodo3] O == N) * alpha23;
  f32 = (nb [Nodo1..Nodo3] O == N) * alpha32;
  f21 = (nb [Nodo1..Nodo3] O == N) * alpha21;

events

  loc to1 (beta1);  syn s12 (f12);
  loc to2 (beta2);  syn s23 (f23);
  loc to3 (beta3);  syn s32 (f32);
  loc ro1 (beta4);  syn s21 (f21);
  loc ro2 (beta5);
  loc ro3 (beta6);
partial reachability = ((st Nodo1 == O) && (st Nodo2 == O) &&
                        (st Nodo3 == O));

network CBE3 (continuous)

  aut Nodo1
    stt O to (T) s12
          to (R) s21
    stt T to (O) to1
    stt R to (O) ro1

  aut Nodo2
    stt O to (T) s23 s21
          to (R) s32 s12
    stt T to (O) to2
    stt R to (O) ro2

  aut Nodo3
    stt O to (T) s32
          to (R) s23
    stt T to (O) to3
    stt R to (O) ro3

results
  BarramentoOcioso = (nb [Nodo1..Nodo3] O == N);
```

4.4.3. Desenvolvimento Global de Software (DGS)

O modelo *Desenvolvimento Global de Software* (DGS) foi proposto em Fernandes *et al.* [FER 2011] representando equipes de desenvolvimento de software distribuídas em diferentes fuso horários e configurações para executar um projeto de desenvolvimento de software chamado ALPHA.

Em linhas gerais, o projeto ALPHA foi executado em 11 meses, envolvendo quatro países: Brasil, Estados Unidos, Malásia e India. Este projeto global atuou com uma infraestrutura adequada para possibilitar comunicação síncrona e assíncrona entre os seus membros, interações entre os mesmos e com a equipe central, compartilhamento de informações e conhecimento com uso de ambiente *web*.

As equipes foram formadas por membros distribuídos em *sites* (locais de desenvolvimento), e estes participantes foram classificados em *junior* (iniciantes) e *senior* (experientes) em diferentes atividades para guiar a construção das equipes. Tem-se 14 participantes, sendo 5 deles instalados no Brasil, 6 nos Estados Unidos, 1 na Malásia e 2 na Índia. A Figura 4.9 ilustra a configuração dos autômatos e seus eventos em uma rede de autômatos estocásticos para representar este projeto.

No modelo SAN, os membros de cada site são representados por autômatos com estados que representam suas atividades em um dia de trabalho de 8 horas. Cada autômato tem três estados: W (trabalhando, ou *working*), S (buscando uma solução, ou *seeking solution*), e C (colaborando, ou *collaborating*). Há também autômatos que representam as atividades da equipe central (coordenadora do projeto, *i.e.*, em inglês *project and delivery managers*, autômato *Activities* com estados M (gerenciando, ou *management*) e C (colaborando, ou *collaboration*)) e sua disponibilidade para interação com as demais equipes (autômato *Availability* com estados A (disponível, ou *available*) e U (indisponível, ou *unavailable*)). Note que a equipe central interage somente com um participante por vez neste modelo, além disto, considera-se que as tarefas estão igualmente distribuídas entre os membros.

Em relação aos autômatos dos participantes, eles apresentam eventos locais e e r, com taxas que refletem o nível de experiência *junior* ou *senior*. Os eventos sincronizantes co representam a alocação dos recursos, e não a frequencia de cooperações entre os participantes, logo, representam a alocação da equipe central. Uma vez que o participante está no estado S e a equipe central está disponível (*i.e.*, no estado A), a cooperação inicia imediatamente - vide taxa funcional $disp$ que permite duas avaliações possíveis:

(i) eventos co tem uma taxa virtualmente infinita, se o autômato *Availability* está no estado A, *i.e.*, ele é disparado imediatamente se há disponibilidade da equipe central;

(ii) eventos co tem uma taxa igual a zero, se o autômato *Availability* está no estado U, logo não dispara.

A dinâmica do modelo é dada pela definição de eventos que habilitam transições na rede de autômatos. A Tabela 4.4 mostra os valores estimados para as taxas dos eventos,

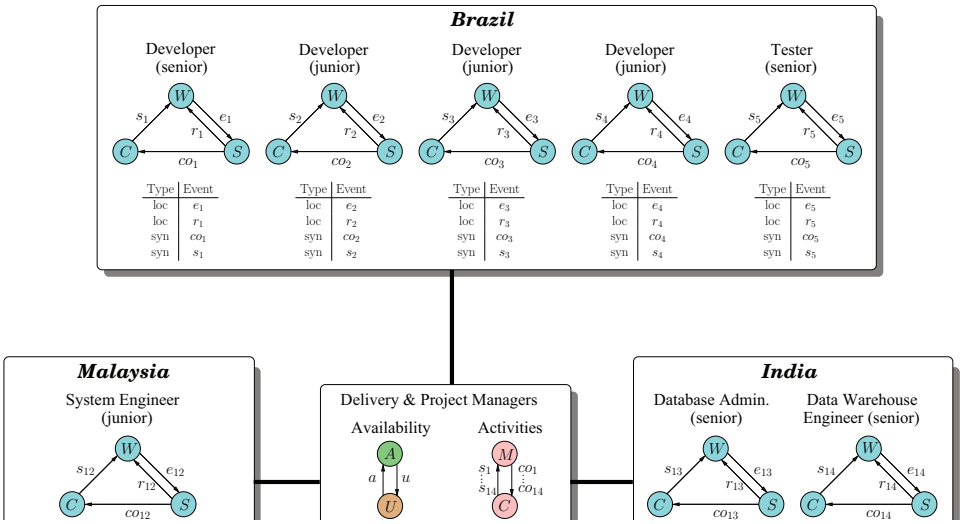


Figura 4.9: Modelo de *Desenvolvimento Global de Software* em SAN [FER 2011].

considerando um dia de trabalho de 8 horas. Os valores foram obtidos de dados estatísticos e históricos do projeto ALPHA, bem como entrevistas com os gerentes e participantes.

Quando resolvido numericamente este modelo pode calcular o desempenho das equipes em diferentes cenários variando a disponibilidade da equipe central e o nível de experiência dos participantes.

A seguir, apresenta-se a descrição de um modelo em SAN para este exemplo de comunicação com barramento exclusivo para $N = 3$ nodos.

Tabela 4.4: Taxas estimadas para os eventos considerando 8h por dia.

Tipo	Evento	Descrição	Taxa
loc	a	equipe central está disponível para colaborar com os participantes em média 2 horas por dia de trabalho, *i.e.*, equipe central colabora há uma taxa de 4 vezes ao dia.	$\frac{8}{2}$
loc	u	equipe central está indisponível para cooperar com os participantes em média 6 horas por dia de trabalho.	$\frac{8}{6}$
loc	e	participantes junior trabalham em média 1 hora por dia sem algum tipo de suporte da equipe central.	$\frac{8}{1}$
		devido à sua experiência, participantes senior trabalham em média 7 horas por dia de trabalho sem algum tipo de suporte da equipe central.	$\frac{8}{7}$
loc	r	participantes junior gastam em média 7 horas por dia de trabalho procurando soluções.	$\frac{8}{7}$
		participantes senior gastam em média somente 1 hora por dia de trabalho procurando soluções devido à sua experiência.	$\frac{8}{1}$
syn	co	quando o participante precisa colaborar com a equipe central, a colaboração ocorre imediatamente se a equipe central está disponível.	$disp$
syn	s	devido à qualidade do suporte da equipe central (de média para baixa), a colaboração demora em média 2 horas por dia de trabalho.	$\frac{8}{2}$

Exemplo de arquivo do modelo DGS

```
identifiers

  UN        = 8; // 8 horas de trabalho
  disp      = (st CT_avail == A);
  taxa_a_u  = (UN/6);
  taxa_u_a  = (UN/2);

  // Developer Senior Brazil
  taxa_e1  = (UN/7);
  taxa_r1  = (UN/1);
  taxa_s1  = (UN/2);
  taxa_co1 = (UN/0.1)*disp;
  // Developer Junior Brazil
  taxa_e2  = (UN/1);
  taxa_r2  = (UN/7);
  taxa_s2  = (UN/2);
  taxa_co2 = (UN/0.1)*disp;
  taxa_e3  = (UN/1);
  taxa_r3  = (UN/7);
```

```
taxa_s3  = (UN/2);
taxa_co3 = (UN/0.1)*disp;
taxa_e4  = (UN/1);
taxa_r4  = (UN/7);
taxa_s4  = (UN/2);
taxa_co4 = (UN/0.1)*disp;

// Tester Senior Brazil
taxa_e5  = (UN/7);
taxa_r5  = (UN/1);
taxa_s5  = (UN/2);
taxa_co5 = (UN/0.1)*disp;

// Business analyst Senior USA
taxa_e6  = (UN/7);
taxa_r6  = (UN/1);
taxa_s6  = (UN/2);
taxa_co6 = (UN/0.1)*disp;
taxa_e7  = (UN/7);
taxa_r7  = (UN/1);
taxa_s7  = (UN/2);
taxa_co7 = (UN/0.1)*disp;

// System engineer Junior Malaysia
taxa_e8  = (UN/1);
taxa_r8  = (UN/7);
taxa_s8  = (UN/2);
taxa_co8 = (UN/0.1)*disp;

// Database administrator Senior India
taxa_e9  = (UN/7);
taxa_r9  = (UN/1);
taxa_s9  = (UN/2);
taxa_co9 = (UN/0.1)*disp;

// ETL engineer Senior India
taxa_e10  = (UN/7);
taxa_r10  = (UN/1);
taxa_s10  = (UN/2);
taxa_co10 = (UN/0.1)*disp;

// ETL engineer Junior USA
taxa_e11  = (UN/1);
taxa_r11  = (UN/7);
taxa_s11  = (UN/2);
taxa_co11 = (UN/0.1)*disp;

// User Senior USA
taxa_e12  = (UN/7);
taxa_r12  = (UN/1);
taxa_s12  = (UN/2);
taxa_co12 = (UN/0.1)*disp;
taxa_e13  = (UN/7);
taxa_r13  = (UN/1);
taxa_s13  = (UN/2);
```

```
  taxa_co13 = (UN/0.1)*disp;
  taxa_e14  = (UN/7);
  taxa_r14  = (UN/1);
  taxa_s14  = (UN/2);
  taxa_co14 = (UN/0.1)*disp;

events

  loc a_u   (taxa_a_u);
  loc u_a   (taxa_u_a);

  loc e1    (taxa_e1);    loc e2    (taxa_e2);    loc e3    (taxa_e3);
  loc r1    (taxa_r1);    loc r2    (taxa_r2);    loc r3    (taxa_r3);
  syn s1    (taxa_s1);    syn s2    (taxa_s2);    syn s3    (taxa_s3);
  syn co1   (taxa_co1);   syn co2   (taxa_co2);   syn co3   (taxa_co3);
  loc e4    (taxa_e4);    loc e5    (taxa_e5);    loc e6    (taxa_e6);
  loc r4    (taxa_r4);    loc r5    (taxa_r5);    loc r6    (taxa_r6);
  syn s4    (taxa_s4);    syn s5    (taxa_s5);    syn s6    (taxa_s6);
  syn co4   (taxa_co4);   syn co5   (taxa_co5);   syn co6   (taxa_co6);
  loc e7    (taxa_e7);    loc e8    (taxa_e8);    loc e9    (taxa_e9);
  loc r7    (taxa_r7);    loc r8    (taxa_r8);    loc r9    (taxa_r9);
  syn s7    (taxa_s7);    syn s8    (taxa_s8);    syn s9    (taxa_s9);
  syn co7   (taxa_co7);   syn co8   (taxa_co8);   syn co9   (taxa_co9);
  loc e10   (taxa_e10);   loc e11   (taxa_e11);   loc e12   (taxa_e12);
  loc r10   (taxa_r10);   loc r11   (taxa_r11);   loc r12   (taxa_r12);
  syn s10   (taxa_s10);   syn s11   (taxa_s11);   syn s12   (taxa_s12);
  syn co10  (taxa_co10);  syn co11  (taxa_co11);  syn co12  (taxa_co12);
  loc e13   (taxa_e13);   loc e14   (taxa_e14);
  loc r13   (taxa_r13);   loc r14   (taxa_r14);
  syn s13   (taxa_s13);   syn s14   (taxa_s14);
  syn co13  (taxa_co13);  syn co14  (taxa_co14);

partial reachability = ((st Partic1 == W)  && (st Partic2 == W)  &&
                        (st Partic3 == W)  && (st Partic4 == W)  &&
                        (st Partic5 == W)  && (st Partic6 == W)  &&
                        (st Partic7 == W)  && (st Partic8 == W)  &&
                        (st Partic9 == W)  && (st Partic10 == W) &&
                        (st Partic11 == W) && (st Partic12 == W) &&
                        (st Partic13 == W) && (st Partic14 == W) &&
                        (st CT_avail == A) && (st CT_activ == M));

network DGS (continuous)

  aut CT_avail
    stt A to (U) a_u
    stt U to (A) u_a

  aut CT_activ
    stt M to (C) co1 co2 co3 co4 co5 co6 co7
                 co8 co9 co10 co11 co12 co13 co14
    stt C to (M) s1 s2 s3 s4 s5 s6 s7 s8 s9 s10 s11 s12 s13 s14

  // Developer Senior Brazil
  aut Partic1
    stt W to (S) e1
```

```
  stt S to (W) r1
        to (C) co1
  stt C to (W) s1

// Developer Junior Brazil
aut Partic2
  stt W to (S) e2
  stt S to (W) r2
        to (C) co2
  stt C to (W) s2

// Developer Junior Brazil
aut Partic3
  stt W to (S) e3
  stt S to (W) r3
        to (C) co3
  stt C to (W) s3

// Developer Junior Brazil
aut Partic4
  stt W to (S) e4
  stt S to (W) r4
        to (C) co4
  stt C to (W) s4

// Tester Senior Brazil
aut Partic5
  stt W to (S) e5
  stt S to (W) r5
        to (C) co5
  stt C to (W) s5

// Business analist Senior USA
aut Partic6
  stt W to (S) e6
  stt S to (W) r6
        to (C) co6
  stt C to (W) s6

// Business analist Senior USA
aut Partic7
  stt W to (S) e7
  stt S to (W) r7
        to (C) co7
  stt C to (W) s7

// System engineer Junior Malaysia
aut Partic8
  stt W to (S) e8
  stt S to (W) r8
        to (C) co8
  stt C to (W) s8


// Database administrator Senior India
```

```
  aut Partic9
    stt W to (S) e9
    stt S to (W) r9
          to (C) co9
    stt C to (W) s9

  // ETL engineer Senior India
  aut Partic10
    stt W to (S) e10
    stt S to (W) r10
          to (C) co10
    stt C to (W) s10

  // ETL engineer Junior USA
  aut Partic11
    stt W to (S) e11
    stt S to (W) r11
          to (C) co11
    stt C to (W) s11

  // User Senior USA
  aut Partic12
    stt W to (S) e12
    stt S to (W) r12
          to (C) co12
    stt C to (W) s12

  // User Senior USA
  aut Partic13
    stt W to (S) e13
    stt S to (W) r13
          to (C) co13
    stt C to (W) s13

  // User Senior USA
  aut Partic14
    stt W to (S) e14
    stt S to (W) r14
          to (C) co14
    stt C to (W) s14

results

  // Equipe Central
  CT_avail_A = (st CT_avail == A);
  CT_avail_U = (st CT_avail == U);
  CT_activ_M = (st CT_activ == M);
  CT_activ_C = (st CT_activ == C);

  // Participante 1 (Developer Senior Brazil)
  Partic1_W  = (st Partic1 == W);
  Partic1_S  = (st Partic1 == S);
  Partic1_C  = (st Partic1 == C);
```

4.5. Considerações Finais

A avaliação de desempenho de diferentes problemáticas é utilizada para detectar problemas e propor alternativas que deixem os sistemas modelados mais rápidos e eficazes. Como visto anteriormente, a avaliação pode compreender a monitoração do sistema, a simulação ou a modelagem analítica. O enfoque principal deste curso foi direcionado à modelagem onde o formalismo escolhido foi Redes de Autômatos Estocásticos (SAN) para a descrição das problemáticas. Dentre a grande gama de formalismos existentes de modelagem estocástica, SAN apresenta primitivas simples de modelagem para endereçar sincronismo e dependência de sistemas modulares. Outros formalismos são poderosos quanto à representação visual e/ou verificação formal, entretanto, as vantagens de SAN não se restringem apenas ao nível de modelagem.

Em comparação a outros formalismos, como por exemplo, Cadeias de Markov [STE 94] e Redes de Petri Estocásticas [AJM 95], a grande vantagem do formalismo SAN é a utilização de álgebra tensorial generalizada [FER 98] que permite a definição de um formato tensorial (descritor Markoviano) que se mostra extremamente econômico face a outros métodos de armazenamento e resolução de sistemas. É possível empregar uma série de técnicas para a resolução dos modelos usando algoritmos que encontram-se no estado-da-arte, realizando a multiplicação de um vetor de probabilidades por um descritor Markoviano de maneira eficiente. Cabe lembrar que entende-se como resolução de modelos estocásticos de avaliação de desempenho a obtenção de probabilidades estimadas (estacionariedade) para as diversas situações em que o sistema possa se encontrar.

Por se tratar de um formalismo relativamente recente, as SAN continuam sendo um fértil campo de pesquisa, sempre englobando novos conceitos e técnicas para armazenamento e soluções de sistemas. Por exemplo, ainda encontram-se em abertos problemas fundamentais no que tange à solução eficiente de estruturas tensoriais quando funções são empregadas, seu relacionamento com simulação para modelos extremamente grandes em termos do espaço de estados produto bem como questões ligadas à verificação de modelos (*model checking*). Estas questões auxiliarão na descoberta de formas mais eficazes de resolver os modelos em menos tempo e permitir a descoberta de relacionamentos e propriedades entre os estados dos mesmos, auxiliando a fase de análise.

Os tópicos abordados neste curso não esgotam o assunto da modelagem estocástica de problemáticas complexas aplicadas (ou não) a realidades computacionais. Em um futuro próximo esperamos que o formalismo seja adaptado para possuir mais primitivas de modelagem que permitam a descrição de outras realidades com comportamentos não previstos até o presente momento. A utilização de SAN por parte de pesquisadores vem crescendo com o passar do tempo e esperamos que este interesse se renove constantemente à medida que surgirem outros modelos, mostrando melhores otimizações dos recursos existentes e apontando gargalos de desempenho.

4.6. Bibliografia

[AJM 95] AJMONE-MARSAN, M. et al. **Modelling with Generalized Stochastic Petri Nets**. [S.l.]: John Wiley & Sons, 1995.

[ARN 51] ARNOLDI, W. E. The principle of minimized iterations in the solution of the matrix eigenvalue problem. **Quarterly of Applied Mathematics**, v.9, p.17–29, 1951.

[BRE 2007] BRENNER, L. et al. PEPS2007 - Stochastic Automata Networks Software Tool. In: INT. CONF. ON THE QUANTITATIVE EVALUATION OF SYSTEMS (QEST'07), 2007, Edinburgh, UK. **Anais...** IEEE Computer Society, 2007. p.163–164.

[CZE 2010a] CZEKSTER, R. et al. Simulation of Markovian models using Bootstrap method. In: SUMMER COMPUTER SIMULATION CONFERENCE (SCSC), 2010, Ottawa, ON, Canada. **Anais...** Society for Computer Simulation International, 2010. p.564–569.

[CZE 2010] CZEKSTER, R. M. **Solução numérica de descritores Markovianos a partir de re-estruturações de termos tensoriais**. 2010. Tese (Doutorado em Ciência da Computação) — Pontifícia Universidade Católica do Rio Grande do Sul (PUCRS), Brasil.

[FER 2011] FERNANDES, P. et al. Performance Evaluation of Software Development Teams: a Practical Case Study. **Electronic Notes in Theoretical Computer Science (ENTCS)**, v.275, p.73–92, 2011.

[FER 98] FERNANDES, P.; PLATEAU, B.; STEWART, W. J. Efficient descriptor-vector multiplication in Stochastic Automata Networks. **Journal of the ACM**, v.45, n.3, p.381–414, 1998.

[FER 2008] FERNANDES, P.; VINCENT, J. M.; WEBBER, T. Perfect Simulation of Stochastic Automata Networks. In: Int. Conf. on Analytical and Stochastic Modelling Techniques and Applications (ASMTA'08), 2008. **Anais...** [S.l.: s.n.], 2008. p.249–263. (LNCS, v.5055).

[FER 98a] FERNANDES, P. **Méthodes numériques pour la solution de systèmes Markoviens à grand espace d'États**. 1998. Tese (Doutorado em Ciência da Computação) — Institut National Polytechnique de Grenoble, France.

[HOP 79] HOPCROFT, J. E.; ULLMAN, J. D. **Introduction to automata theory, languages and computation**. [S.l.]: Addison-Welsey, 1979.

[KAM 98] KAM, T. et al. Multi-valued decision diagrams: theory and applications. **Multiplie-Valued Logic**, v.4, n.1-2, p.9–62, 1998.

[KLE 75] KLEINROCK, L. **Queueing Systems**. New York: John Wiley & Sons, 1975.

[PID 92] PIDD, M. **Computer Simulation in Management Science**. [S.l.]: John Wiley & Sons, 1992.

[PLA 91] PLATEAU, B.; ATIF, K. Stochastic Automata Networks for modelling parallel systems. **IEEE Transactions on Software Engineering**, v.17, n.10, p.1093–1108, 1991.

[PLA 84] PLATEAU, B. **De l'Evaluation du Parallélisme et de la Synchronisation**. 1984. Tese (Doutorado em Ciência da Computação) — Paris-Sud, Orsay.

[SAA 86] SAAD, Y.; SCHULTZ, M. H. GMRES: a Generalized Minimal RESidual algorithm for solving nonsymmetric linear systems. **SIAM Journal on Scientific and Statistical Computing**, Philadelphia, PA, USA, v.7, n.3, p.856–869, 1986.

[SAL 2009a] SALES, A.; PLATEAU, B. Reachable state space generation for structured models which use functional transitions. In: INTERNATIONAL CONFERENCE ON THE QUANTITATIVE EVALUATION OF SYSTEMS (QEST'09), 2009, Budapest, Hungary. **Anais...** IEEE Computer Society, 2009. p.269–278.

[SAL 2009] SALES, A. **Réseaux d'Automates Stochastiques: Génération de l'espace d'états atteignables et Multiplication vecteur-descripteur pour une sémantique en temps discret**. 2009. Tese (Doutorado em Ciência da Computação) — Institut Polytechnique de Grenoble (Grenoble INP), Grenoble, France. Brigitte Plateau (advisor).

[SAL 2011] SALES, A. **SAN Lite-Solver**: a user-friendly software tool to solve SAN models. http://sites.google.com/site/afonsosales/tools#SANLite.

[STE 94] STEWART, W. J. **Introduction to the numerical solution of Markov chains**. [S.l.]: Princeton University Press, 1994.

[STE 2009] STEWART, W. J. **Probability, Markov Chains, Queues, and Simulation**. USA: Princeton University Press, 2009.

[WEB 2009] WEBBER, T. **Reducing the Impact of State Space Explosion in Stochastic Automata Networks**. 2009. Tese (Doutorado em Ciência da Computação) — Pontifícia Universidade Católica do Rio Grande do Sul (PUCRS), Brasil.