II CONGRESSO NACIONAL DE ENGENHARIA MECÂNICA
II NATIONAL CONGRESS OF MECHANICAL ENGINEERING
12 a 16 de Agosto de 2002 - João Pessoa – PB

SIMULAÇÃO DE SISTEMAS HÍBRIDOS USANDO MATLAB / SIMULINK

Emilia Villani
Dpto. Eng. Mecatrônica e de Sistemas Mecânicos, Escola Politécnica da Universidade de São Paulo
Av. Prof. Mello Moraes, 2231 São Paulo Brasil
evillani@usp.br

Chen Yen-Tsang
Dpto. Eng. Mecatrônica e de Sistemas Mecânicos, Escola Politécnica da Universidade de São Paulo
Av. Prof. Mello Moraes, 2231 São Paulo Brasil
yentsang@hotmail.com

Paulo Eigi Miyagi
Dpto. Eng. Mecatrônica e de Sistemas Mecânicos, Escola Politécnica da Universidade de São Paulo
Av. Prof. Mello Moraes, 2231 São Paulo Brasil
pemiyagi@usp.br

Resumo. *Considerando o desenvolvimento de sistemas híbridos, um aspecto de particular importância é a escolha da técnica de modelagem. Dentro das técnicas disponíveis, aquelas que definem uma interface entre um formalismo discreto e um formalismo contínuo, apresentam um maior poder de modelagem. No entanto, não se pode garantir a existência de uma solução analítica para os sistemas de equações diferenciais. Assim, para sistemas complexos, a simulação apresenta-se, em muitos casos, como única ferramenta efetiva para análise do comportamento do sistema. Neste contexto, este trabalho introduz uma nova abordagem para simulação de modelos construídos através da associação de redes de Petri à sistemas de equações diferenciais baseando-se no software MatLab/Simulink. Enquanto as redes de Petri são simuladas através de subrotinas do MatLab, a parte contínua é modelada em diagramas de blocos no Simulink. A padronização da interface entre os dois subsistemas é realizada através de blocos do Simulink.*

Palavras-chave: *sistemas híbridos, simulação, redes de Petri.*

1. INTRODUÇÃO

Dentro do conceito de sistemas mecatrônicos, um aspecto importante é a integração de sistemas de diferentes tecnologias. Esta integração tem sido objeto de diversos estudos e, em muitos casos, é um fator limitante para o aperfeiçoamento de sistemas de controle supervisório. Neste sentido, um problema freqüentemente abordado, que é um resultado direto desta necessidade de integração, é o desenvolvimento de sistemas supervisórios híbridos, isto é, sistemas supervisórios que apresentam ao mesmo tempo variáveis contínuas e discretas.

Para o projeto de sistemas supervisórios híbridos, a escolha da técnica de modelagem é de fundamental importância, uma vez que, entre outras coisas, ela delimita as abordagens e ferramentas a serem utilizadas para análise do sistema projetado. Entre as técnicas de modelagem disponíveis, algumas consistem em extensões de modelos contínuos, como equações diferenciais ordinárias nas quais são incluídas variáveis cujos valores podem ser modificados de forma descontínua no tempo [Antsaklis & Nerode, 1998]. Outras abordagens consistem na modificação de técnicas de modelagem utilizadas em sistemas a eventos discretos, onde são introduzidos novos

elementos que permitem a representação da dinâmica contínua do sistema, como nas Redes de Petri Híbridas [Alla & David, 1998]. Existem, também, abordagens intermediárias que combinam modelos de sistemas contínuos, descritos por equações diferenciais, e de sistemas discretos descritos por autômatos finitos ou Redes de Petri, onde é introduzida uma interface para a comunicação entre os dois tipos de modelos, como em [Valentin-Roubinet, 1998].

De uma forma geral, os sistemas supervisórios híbridos implicam na execução e/ou monitoração de seqüências de atividades, na resolução de conflitos e indeterminismos, etc., que resultam em uma lógica de controle discreta (isto é, composta por estados que se alternam devido a ocorrência de eventos modelados como instantâneos) relativamente complexa, inviabilizando a utilização de abordagens que são extensão de formalismos contínuos. Por outro lado, os processos produtivos a serem supervisionados freqüentemente também envolvem relações complexas entre as diversas variáveis contínuas do modelo a ser construído, o que inviabiliza a utilização de extensões de formalismos discretos. Assim, este trabalho considera particularmente as abordagens que especificam uma solução a partir da definição de uma interface entre um formalismo discreto e outro contínuo. Estas abordagens, em geral, apresentam maior flexibilidade e poder de modelagem.

Entre as abordagens deste grupo, neste trabalho considerou-se em especial aquelas onde o formalismo discreto é baseado em redes de Petri. Esta escolha foi realizada em base às já bem conhecidas propriedades deste formalismo, como habilidade para representar a sincronização de processos, eventos concorrentes, causalidade, compartilhamento de recursos, presença de conflitos e a possibilidade de utilização não apenas da descrição gráfica, mas também da descrição matemática e formal equivalente.

A verificação e validação de modelos construídos através da definição de uma interface entre sistemas de equações diferenciais e um modelo em redes de Petri é um problema frequentemente apontado por pesquisadores da área de sistemas híbridos. No entanto, as soluções propostas tratam apenas de casos específicos. Um dos pontos principais deste problema é o fato de que a consideração de variáveis contínuas leva a um número infinito de estados para o sistema, o que inviabiliza a verificação de propriedades baseadas na árvore de alcançabilidade da rede. Simultaneamente, dado que não são consideradas restrições para a parte contínua, não é possível garantir a existência de uma solução analítica para os sistemas de equações diferenciais, fazendo com que a simulação seja muitas vezes a única ferramenta disponível.

Neste sentido, este trabalho apresenta uma solução para a simulação de sistemas híbridos baseada na utilização do software MatLab/Simulink. O Capítulo 2 aborda o problema de simulação de sistemas híbridos de uma forma geral, apresentando as principais ferramentas existentes. O Capítulo 3 apresenta a abordagem proposta. Em seguida, o Capítulo 4 descreve brevemente sua aplicação para um estudo de caso. Finalmente, o Capítulo 5 apresenta as principais conclusões obtidas, bem como as diretrizes para o prosseguimento do trabalho.

2. SIMULAÇÃO DE SISTEMAS HÍBRIDOS

De acordo com [Champagnat et al, 2001], a simulação de sistemas pode ser dividia em duas abordagens principais. A primeira é utilizada para a simulação de sistemas dirigidos a eventos discretos. Neste grupo se encontram os simuladores de redes de Petri de uma forma geral. Entre estes se destacam os simuladores para modelos com tempo, como redes de Petri temporizads e estocásticas. Neste último caso, os eventos são armazenados em um calendário, na ordem de ocorrência (eventos armazenados para a mesma a data ocorrem em ordem aleatória). Na ocorrência de cada evento, ele é retirado do calendário e pode ainda implicar na adição ou remoção de outros eventos. A segunda abordagem é utilizada para a simulação de sistemas contínuos no tempo, modelados por sistemas de equações diferenciais. Neste caso a simulação é do tipo determinística e não existem duas mudanças de estado simultâneas (ocorrendo numa mesma data).

Como um reflexo das abordagens propostas para modelagem, a simulação de sistemas híbridos também pode ser dividida em três grupos. O primeiro consiste em extensões de simuladores discretos. Neste caso o calendário é preservado e a dinâmica da parte contínua é resumida a

equações explícitas no tempo, onde é possível calcular, no instante de ocorrência de cada evento, a data de ocorrência de mudanças de estado decorrentes da dinâmica contínua, como ultrapassagem de limites. Estas mudanças de estado são consideradas como eventos discretos e agendadas no calendário. Um exemplo é a extensão proposta em [Hochon et al, 1998] para o software MISS-RdP.

O segundo caso consiste na extensão de simuladores contínuos, onde não existe o conceito de calendário e os eventos devem ser expressados sobre a forma de uma variável contínua atingindo um limite. Este é o caso do software Modellica. Finalmente, o terceiro modo requer a integração e sincronização de dois simuladores: um discreto e um contínuo.

Em [Mosterman, 1999] pode-se encontrar uma revisão sobre alguns softwares para simulação híbrida em geral. A maioria dos softwares analisados considera autômatos finitos para a modelagem da parte contínua. No que se refere a simuladores onde a parte discreta é baseada em redes de Petri, além da extensão do MISS-RdP, foi também proposto o Visual Object Net [Drath, 1998], que, no entanto, é baseado em redes de Petri Híbridas [Alla & David, 1998], o que, conforme apresentado no Capítulo 1, limita sua aplicação no caso de sistemas supervisórios híbridos.

Visando propor uma solução mais abrangente para simulação de sistemas modelados através associação de redes de Petri a sistemas de equação diferencias, foi desenvolvida a abordagem apresentada a seguir, utilizando o software comercial MatLab/Simulink, da MathWorks.

3. ABORDAGEM PROPOSTA

3.1. Modelagem do Sistema

O formalismo considerado para construção do modelo a ser simulado é baseado nas redes Predicado/Transição Diferenciais (redes PTD) [Champagnat, 1998]. Resumidamente, as redes PTD são definidas através da adição das seguintes características a uma rede de Petri ordinária [Cardoso e Valette, 1995]:

- Á cada rede é associado um vetor de variáveis contínuas.
- Á cada *lugar* é associado um sistemas de equações diferenciais algébricas que determina a evolução dinâmica das variáveis contínuas quando o *lugar* está marcado.
- Á cada *transição* é associada uma função de habilitação, que habilita o disparo da *transição* em função dos valores das variáveis contínuas.
- Á cada *transição* é associada uma função de junção, que determina os valores das variáveis contínuas após o disparo das *transições*.

Observa-se que, em relação ao modelo original proposto por [Champagnat, 1998], este trabalho adota como restrição a capacidade unitária para os *lugares* associados a sistemas de equações diferenciais. Por outro lado, adota-se como extensão a incorporação de *arcos habilitadores* e *inibidores*, que podem ser oriundos de outros *lugares* da rede ou representar um sinal externo a rede. Este ultimo caso é particularmente importante para modelagem da interação do modelo com elementos externos.

Considera-se ainda que, além das redes PTD para a modelagem da parte híbrida do sistema, o modelo a ser simulado pode ainda conter uma parte puramente discreta, modelada em redes de Petri Ordinárias com *arcos habilitadores* e *inibidores*, e uma parte puramente continua, modelada por sistemas de equações diferenciais autônomos, isto é, que não são associados a nenhum *lugar* e são, portanto, sempre válidos. A Figura 1 apresenta um esquema dos sistemas a serem simulados.

3.2. Simulação no *MatLab/Simulink*

O problema de simulação de modelos descritos de acordo a técnica apresentada no item 3.1 pode ser dividido em três partes:

- Simulação da parte discreta;
- Simulação da parte contínua;
- Sincronização e comunicação das duas partes.

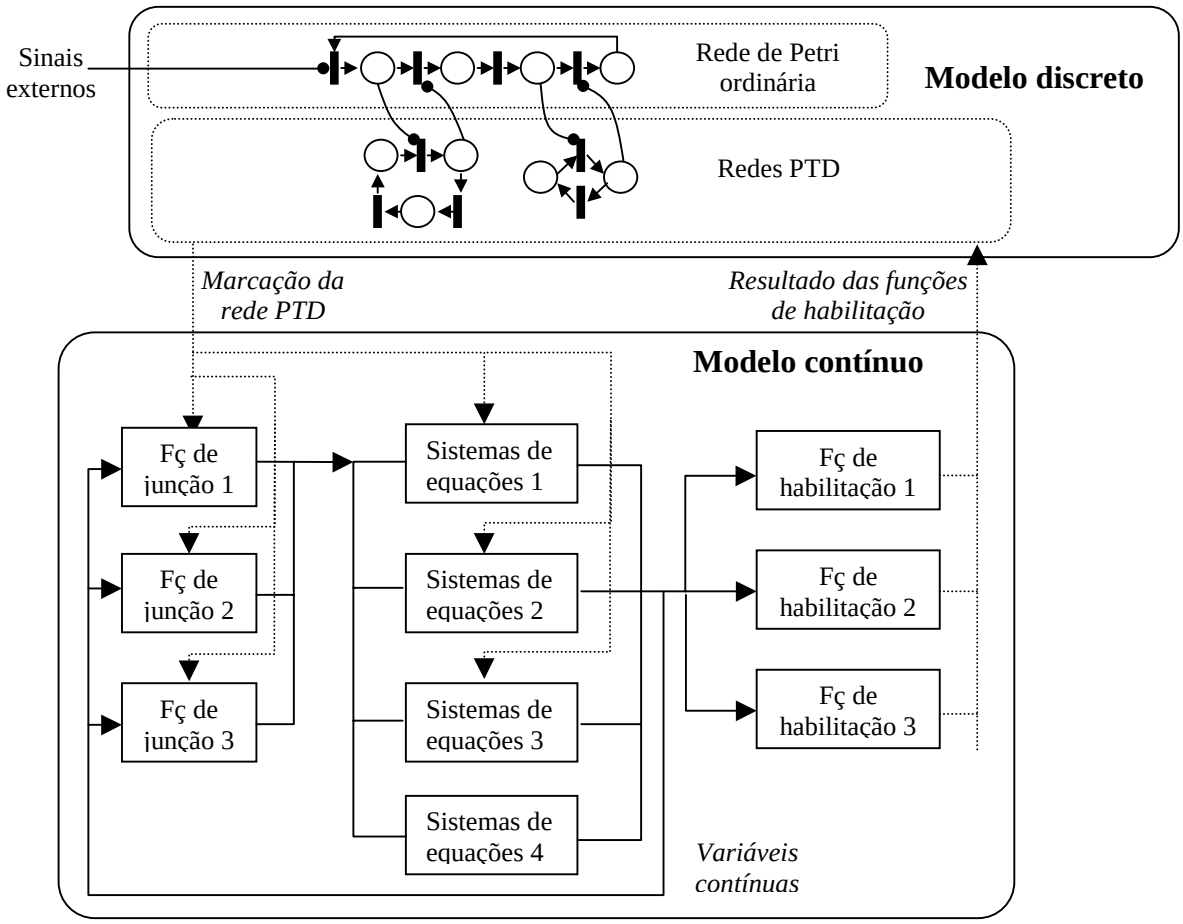


Figura 1. Estrutura do modelo a ser simulado.

De uma forma geral a simulação da parte discreta é relativamente simples graças a possibilidade de representação na forma matricial dos modelos em redes de Petri. Por outro lado, a simulação da parte contínua é de uma maior complexidade, uma vez que não existem restrições quanto ao tipo de sistema de equações diferenciais. O simulador da parte contínua deve ser capaz de resolver sistemas não lineares, com diversas variáveis, assim como possibilitar a utilização de diferentes métodos de resolução.

Por estas razões, neste trabalho optou-se pelo desenvolvimento e introdução de um simulador discreto a um simulador contínuo já existente. O simulador escolhido foi o *Simulink*. Entre os motivos que levaram a esta escolha estão a interface intuitiva e a facilidade de utilização do programa, que é do tipo *point and click*. Outro fator de grande peso foi o fato de ele ser uma extensão do *MatLab*, um programa largamente difundido e conhecido pela sua capacidade e pelos recursos oferecidos para manipulação de matrizes, o que favorece particularmente a implementação do simulador para a parte discreta. Outro ponto importante é que além da possibilidade de programação tradicional (através dos M files) onde os comandos são interpretados durante a execução, somou-se a possibilidade de programação usando a linguagem C, que é previamente compilada aumentando significativamente a velocidade de simulação e execução de programas. Uma última vantagem é que ambos o *Simulink* e o *MatLab* estão disponíveis para uma grande variedade de plataformas, como (PC/Windows, MacIntosh, Unix/X, Windows, VAX/VMS).

Em linhas gerais, a solução apresentada neste trabalho para simulação de sistemas híbridos consiste em executar as redes de Petri através de subrotinas do MatLab®, enquanto a simulação da parte contínua é realizada através de modelos em diagramas de blocos no *Simulink*. A padronização da interface entre os dois subsistemas é realizada através de blocos do *Simulink*, que garantem a sincronização das atividades.

3.3 Simulação da parte discreta

A simulação da parte discreta é realiza por um conjunto de subrotinas escritas em linguagem M (linguagem de programação tradicional do MatLab) que executam a rede de Petri, isto é, que determinam a seqüência de disparo das *transições* e a evolução da marcação da rede utilizando a notação matricial [Cardoso & Valette, 1995].

Na notação matricial, uma *transição* está habilitada se:

$$M_0 >= Pre \cdot \sigma \quad (1)$$

$$Pos \cdot \sigma + M_0 <= C \quad (2)$$

onde M_0 é vetor de marcação inicial;
Pos é matriz de pós-condição da rede de Petri;
Pre é matriz de pré-condição da rede de Petri;
σ é vetor da *transição* habilitada.
C é vetor capacidade dos *lugares* da rede de Petri.

Semelhantemente, a marcação da rede após o disparo de uma *transição* é dada por:

$$M = M_0 + (Pos - Pre) \cdot \sigma \quad (3)$$

onde M é vetor de marcação atual.

Para incorporação dos *arcos habilitadores* e *inibidores* na notação matricial utiliza-se uma abordagem semelhante à apresentada em [Matsusaki, 1998] para os gates do MFG. Resumidamente, considera-se que do ponto de vista exclusivo da habilitação de uma *transição*, um *lugar* (ou um sinal externo) conectado a uma *transição* por um *arco habilitador* corresponde a uma um *lugar* de entrada, e um *lugar* (ou um sinal externo) conectado por um *arco inibidor* corresponde a um *lugar* de saída com capacidade unitária (Figura 2). Estes *lugares* correspondentes são chamados de *pseudo-lugares* e sua marcação corresponde a marcação do *lugar* de origem do *arco habilitador/ inibidor*, ou do sinal externo. A marcação dos *pseudo-lugares* não é alterada pelo disparo da *transição*, mas deve ser atualizada antes de uma nova verificação das *transições* habilitadas.

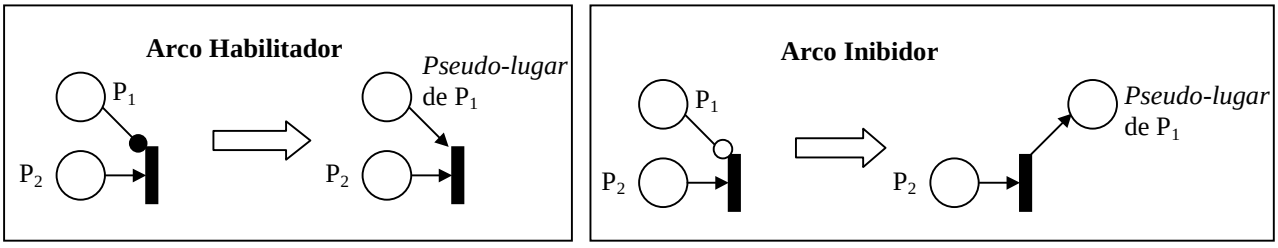
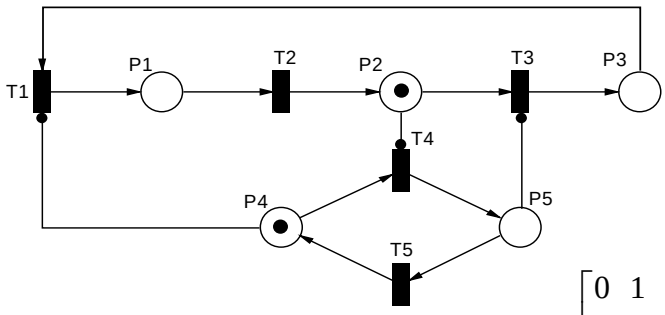


Figura 2. Relação entre *arcos habilitadores/inibidores* e *pseudo-lugares*.

Na utilização da notação matricial são criadas duas novas matrizes de pré e pós condições, chamadas de Pre_A e Pos_A, onde são adicionados os *pseudo-lugares*. Estas novas matrizes substituem as matrizes Pre e Pos na detecção das *transições* habilitadas, isto é, nas expressões (1) e (2), mas não entram na determinação da nova marcação depois do disparo da *transição*, isto é, na equação (3). Um exemplo é apresentado na Figura 3. Nas matrizes Pre_A e Pos_A tem-se que o *lugar* P_6 corresponde ao *pseudo-lugar* de P_2, P_7 ao *pseudo-lugar* P_4 e P_8 ao *pseudo-lugar* P_5.

A simulação de redes de Petri está organizada da seguinte forma. A subrotina principal é o *Jogador_de_Marcas*. As principais subrotinas por ela utilizada são *Função_Habilitação*, que detecta quais são as *transições* habilitadas, *Função_Conflito*, que resolve eventuais conflitos, e *Função_Atualiza*, que atualiza a marcação da rede após o disparo de *transições*. Os dados de entrada para a simulação são as matrizes Pre, Pos, Pre_A e Pos_A, e os vetores C e M_0, além dos eventuais sinais externos e do resultado das funções de habilitação proveniente da interface com a parte contínua.



$$\text{Pre} = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}, \text{Pos} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 \end{bmatrix}, \text{Pre\_A} = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \end{bmatrix}, \text{Pos\_A} = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Figura 3. Exemplo de representação matricial.

A implementação da simulação da rede de Petri no ambiente do *Simulink* é realizada utilizando-se o modelo apresentado na Figura 4. Neste modelo, tem-se que:

- As matrizes Pre, Pos, Pre_A e Pos_A, e os vetor C são implementados através do bloco *Constant* (Biblioteca *Sources*) uma vez que não variam ao longo do tempo.
- Para a marcação da rede utilizam-se os blocos *Data Store Memory*, *Data Store Write*, *Data Store Read* (Biblioteca *Signals & Systems*), que permitem a atualização e armazenagem dinâmica da marcação da rede no decorrer da simulação. O mesmo procedimento é usado para o resultado das funções de habilitação.
- Os sinais externos podem ser modelados através de combinações de blocos da Biblioteca *Sources*.
- O simulador é encapsulado no bloco *Matlab Function* (Biblioteca *Functions & Tables*), que tem como função a subrotina *Jogador_de_Marcas*.

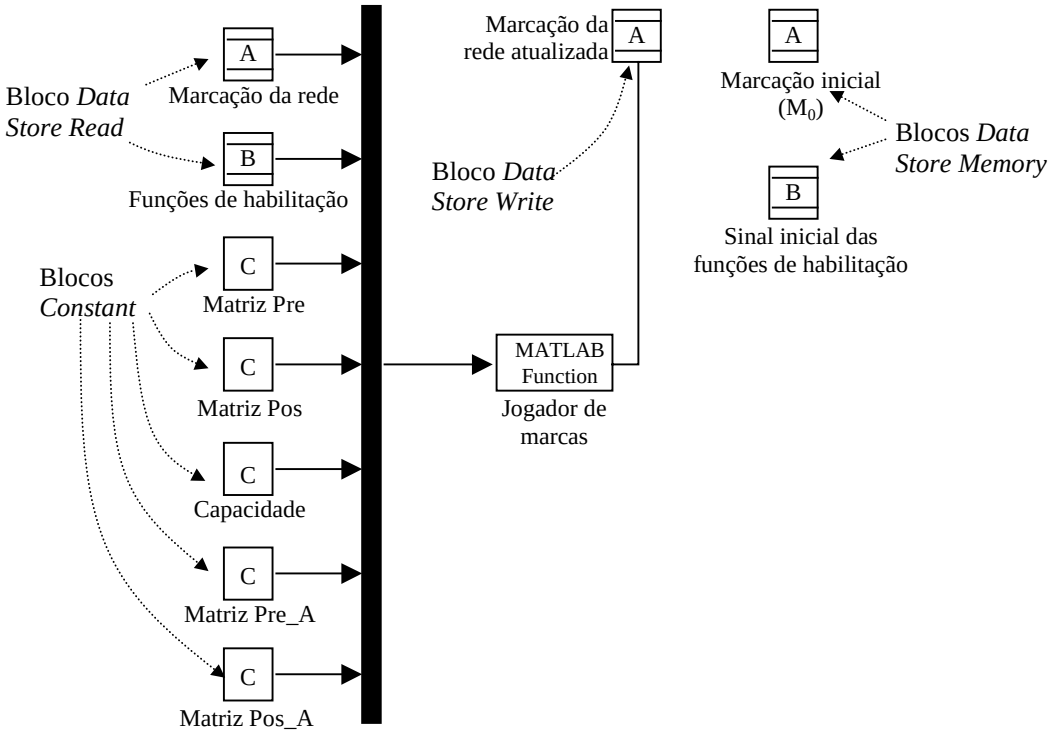


Figura 4. Implementação do simulador de redes de Petri no ambiente Simulink.

3.3 Simulação da parte contínua

Cada sistema de equação diferencial é definido através de um bloco *SubSystem* (Biblioteca *Signals & Systems*) dentro do qual deve ser inserido o diagrama de blocos correspondente ao sistema, de acordo com os recursos oferecidos pelo Simulink. Algumas diretrizes de modelagem podem ser encontradas em [Matsumoto, 2001].

3.4 Interface entre a parte continua e discreta

Modelagem das funções de habilitação

As funções de habilitação são modeladas através do bloco *Function* (Biblioteca *Functions & Tables*). A comparação é realizada através do bloco *Relational Operator* (Biblioteca *Math*), tendo o zero como segundo parâmetro (Figura 5). O resultado de todas as funções de habilitação é colocado em um vetor e armazenado através do bloco *Data Store Write* (Biblioteca *Signals & Systems*), para então ser utilizado pelo *Jogador de marcas* da parte discreta.

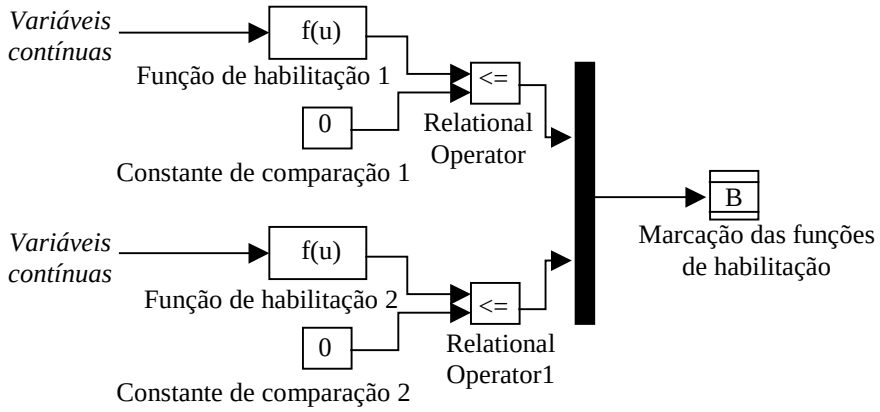


Figura 5. Implementação das funções de habilitação.

Modelagem das funções de junção

As funções de junção são modeladas através da combinação de três blocos (Figura 6). O bloco *Function* (Biblioteca *Functions & Tables*) recebe como entrada a marcação da rede de Petri. Quando a marcação da rede é modificada, a função associada ao bloco verifica em base a marcação atual e antiga, se houve o disparo da *transição*. Neste caso, a saída do bloco é chaveada de 0 para 1. Esta mudança de sinal é detectada pelo primeiro bloco *Subsystem* (Biblioteca *Signals & Systems*) sensível a borda de subida, que imediatamente calcula o valor das variáveis continuas após o disparo. Estes valores são armazenados pelo bloco *Subsystem* (Biblioteca *Signals & Systems*) sensível a nível, que os fornece aos sistemas de equações diferenciais até o disparo da próxima *transição* associada a uma função de junção.

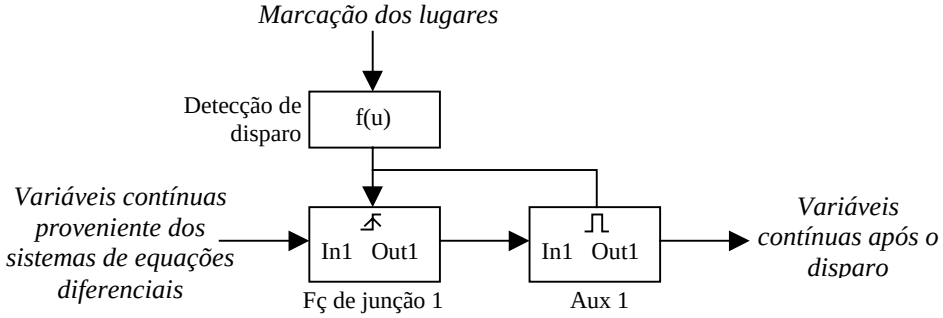


Figura 6. Implementação da função de junção.

Seleção dos sistemas de equação diferencial

A seleção do sistema de equação diferencial válido para a determinação da evolução das variáveis contínuas do modelo é realizada através do bloco *Multiport Switch* (Biblioteca *Nonlinear*), que recebe como entrada uma combinação linear da marcação dos *lugares* da rede de Petri, calculada pelo bloco *Function* (Biblioteca *Functions & Tables*). Observa-se que os sistemas de equação diferencial recebem como entrada o resultado das funções de junção, que são atualizados apenas nos disparos das *transições*, isto é, permanecem constante entre dois disparos. Por outro lado, a saída dos sistemas de equações diferenciais evoluem continuamente no tempo.

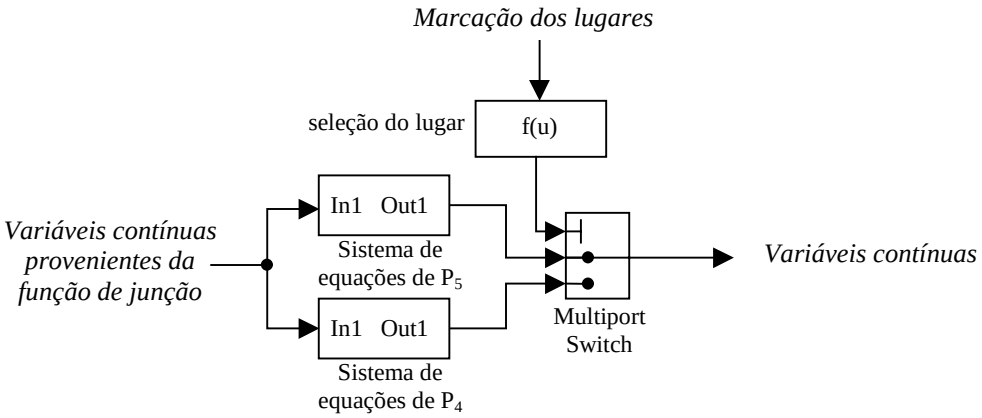


Figura 7 – Implementação da seleção dos sistemas de equações diferenciais.

Uma vez construído o modelo de cada parte do sistema as diversas partes são integradas. Os resultados podem ser armazenados e analisados utilizando as diversas ferramentas disponibilizadas pelo Simulink.

4. ESTUDO DE CASO – SISTEMA DE AR CONDICIONADO

A abordagem descrita acima foi utilizada para simulação de um sistema de ar condicionado de um conjunto de salas em um edifício. O modelo foi baseado no sistema de ar condicionado do Prédio dos Ambulatórios do Hospital das Clínicas da Faculdade de Medicina da Universidade de São Paulo.

Resumidamente, o sistema de ar condicionado modelado é formado por dois ventiladores, uma caixa de mistura e uma serpentina. O ar é retirado da zona pelo ventilador de retorno, é parcialmente renovado por ar exterior na caixa de mistura, atravessa a serpentina de resfriamento, onde tem sua temperatura reduzida, e é então enviado às salas condicionadas pelo ventilador de insuflamento.

O controle do sistema de ar condicionado é dividido em duas partes. O controle local é formado por um regulador PI (proporcional integral) que determina a posição da válvula que controla o fluxo de água gelada na entrada da serpentina em função da temperatura no ambiente. O controle supervisório chaveia entre diferentes estratégias de controle e modifica o estado dos equipamentos e do sistema de controle local de acordo com sinais recebidos do sistema de gerenciamento do edifício e/ou de usuários nas salas. Um exemplos de estratégia de controle é a *estratégia em caso de incêndio*, onde modifica-se as velocidades dos ventiladores de forma a manter a pressão nas salas abaixo da pressão ambiente e evitar que a fumaça se espalhe por todo o edifício.

Os modelos adotados, assim como maiores informações sobre o sistema, podem ser encontrado em [Villani, 2000]. Resumidamente, os modelos das estratégias de controle são puramente discretos, sendo portanto construídos em redes de Petri. Os modelos dos equipamentos são híbridos, pois do ponto de vista do sistema supervisório, cada equipamento pode assumir diferentes configurações discretas (ex.: ligado, desligado, velocidade alta, etc.) e do ponto de vista das salas condicionadas, cada equipamento modifica de forma dinamicamente contínua as propriedades do fluxo de ar insuflado no ambiente (ex.: temperatura, vazão). Assim, cada equipamento é modelado

por uma rede PTD. Finalmente, o modelo do ambiente é composto pela equacionamento da troca de calor e por eventos discretos que influenciam a carga térmica (ex. entrada e saída de pessoas).

Neste caso, o objetivo da simulação é determinar quais são os efeitos das diferentes estratégias de controle na temperatura do ar das salas condicionadas. A Figura 8 apresenta como exemplo apresenta alguns dos resultados obtidos para um cenario onde considera-se a ocorrência de varios eventos discretos no decorrer da simulação. Os gráficos construídos devem servir de base para a análise das estratégias de controle.

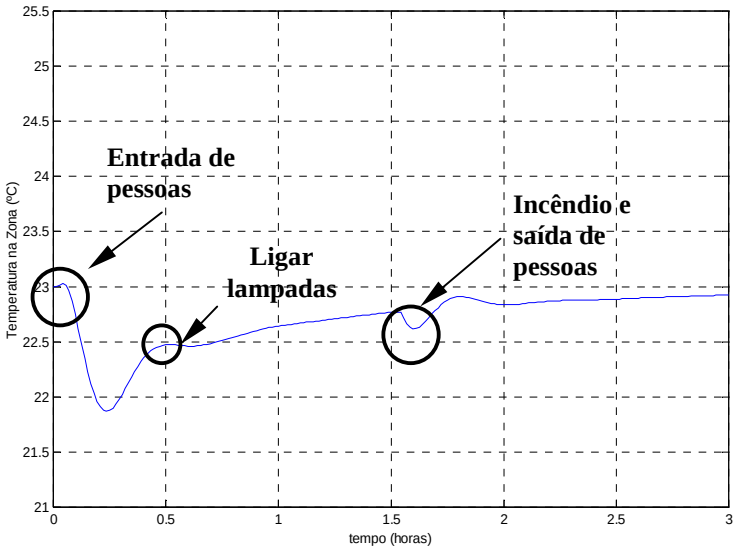


Figura 8 - Exemplo de resultado fornecido pelo simulador.

5. CONCLUSÃO

Visando propor uma solução para o problema de análise de sistemas híbridos, este trabalho introduz uma nova abordagem para construção e simulação de modelos construídos através da associação de redes de Petri à sistemas de equações diferenciais baseando-se no software MatLab/Simulink. São apresentadas as principais etapas para construção do modelo e utilização das subrotinas desenvolvidas, assim como sua utilização para um estudo de caso.

O prosseguimento do trabalho deve incluir a criação de uma interface gráfica que possibilite a geração automática dos modelos na plataforma Simulink a partir da rede fornecida no seu formato original.

Entre as vantagens da abordagem adotada está a possibilidade de usufruir dos recursos fornecidos pelo MatLab/Simulink. Neste sentido é possível a extrapolação do modelo original proposto em [Champagnat, 1998], onde a parte contínua é modelada através de equações diferenciais, para inclusão elementos de inteligência artificial na modelagem do sistema, como redes neurais ou fuzzy.

6. AGRADECIMENTOS

Os autores agradecem o apoio das entidades FAPESP, CNPq, CAPES e FINEP/RECOPE para o desenvolvimento deste trabalho.

4. REFERÊNCIAS

Alla, H. & David, R., 1998, “Continuous and Hybrid Petri Nets” Journal of Circuits, Systems and Computers, vol.8, n.1

Antsaklis, P. J. & Nerode, A., 1998, “Hybrid Control Systems” IEEE Transactions on Automatic Control, vol 43, n.4, pp 457-459.

Cardoso, J. & Valette, R., 1997, Redes de Petri, Editora da UFSC, Florianópolis.
Champagnat, R., 1998, “Supervision des Systèmes Discontinus: Definition d’un Modèle Hybride et Pilotage en Temps-rèel” PhD Thesis, Univ. Paul Sabatier, Toulouse, France.
Champagnat, R., Valette, R., Hochon, J., Pingaud, H., 2001, “Modeling, simulation and analysis of batch production systems”, Discrete Event Dynamic Systems: Theory and Application, Vol 11, n.1/2.
Drath, R., 1998, “Hybrid Object Nets: An Object Oriented Concept for Modelling Complex Hybrid Systems” ADMP’98 3rd International Conference on Automation of Mixed Processes, Reims.
Hochon, J., Champagnat, R., Valette, R., 1998, “Modélisation et simulation hybride à l'aide des réseaux de Petri Prédicats-Transitions couplés à des équations algébro-différentielles”, Actes du 4e Colloque Africain sur la Recherche en Informatique, CARI'98, Dakar (Sénégal).
Matsusaki, C. T. M., 1998, Redes F-MFG (Functional Mark Flow Graph) e sua aplicação no projeto de sistemas antropocêntricos. Dissertação de Mestrado, EPUSP, São Paulo.
Matsumoto, 2001, E.Y; MATLAB 6 – Fundamentos de programação; Ed. Érica LTDA, São Paulo.
Mosterman, P. J., 1999, "An Overview of Hybrid Simulation Phenomena and Their Support by Simulation Packages," in Hybrid Systems: Computation and Control '99, Lecture Notes in Computer Science vol. 1569, Frits W. Vaandrager and Jan H. van Schuppen (eds.).
Valentin-Roubinet, C. (2000). “Hybrid Dynamic System verification with Mixed Petri Nets”, Proc. ADPM 2000 - The 4th International Conference on Automation of Mixed Processes: Hybrid Dynamic Systems, Dortmund, Germany.
Villani, E., 2000, Abordagem Híbrida para Modelagem de Sistemas de Ar Condicionado em Edifícios Inteligentes Dissertação de Mestrado, EPUSP, São Paulo.

HYBRID SYSTEM SIMULATION USING MATLAB/SIMULINK

Emilia Villani
Dpto. Eng. Mecatrônica e de Sistemas Mecânicos, Escola Politécnica da Universidade de São Paulo
Av. Prof. Mello Moraes, 2231 São Paulo Brasil
evillani@usp.br

Chen Yen-Tsang
Dpto. Eng. Mecatrônica e de Sistemas Mecânicos, Escola Politécnica da Universidade de São Paulo
Av. Prof. Mello Moraes, 2231 São Paulo Brasil
yentsang@hotmail.com

Paulo Eigi Miyagi
Dpto. Eng. Mecatrônica e de Sistemas Mecânicos, Escola Politécnica da Universidade de São Paulo
Av. Prof. Mello Moraes, 2231 São Paulo Brasil
pemiyagi@usp.br

Abstract*. In the development of hybrid supervisory systems, the choice of the modeling technique is of particular importance. Generally, the techniques that define an interface between a continuous formalism, such as differential equation systems, and a discrete one, such as Petri nets, have more flexibility. On the other hand, it is not possible to assure an analytical solution to the differential equation system and simulation is sometimes the only tool available for the behavior analysis of complex systems. In this context, this works introduces a novel approach for hybrid system simulation using the software MatLab/Simulink. For the Petri nets simulation, subroutines of MatLab are proposed, while the continuous part is modeled by the traditional block diagram approach of Simulink. The interface between the discrete and continuous part is also patronized using the blocks of Simulink.*

Keywords*. Hybrid systems, simulation, Petri nets.*