UNIVERSIDADE ESTADUAL DE CAMPINAS

FACULDADE DE ENGENHARIA ELÉTRICA

DEPARTAMENTO DE ENGENHARIA DA COMPUTAÇÃO E AUTOMAÇÃO INDUSTRIAL

LÓGICA TEMPORAL DE TEMPO REAL GENERALIZADA APLICADA AO CONTROLE E SIMULAÇÃO DE SISTEMAS DINÂMICOS A EVENTOS DISCRETOS

Este exemplar corresponde à redação final da tese defendida por Braz Izaias da Silva Júnior e aprovada pela Comissão Julgadora em 13 / 07 / 92.

Rafael Santos Mendes
Orientador

Autor : Braz Izaias da Silva Júnior

Orientador: Rafael Santos Mendes

Tese apresentada à Faculdade de Engenharia Elétrica da Universidade Estadual de Campinas como parte dos requisitos exigidos para a obtenção do grau de Mestre em Engenharia Elétrica.

Julho de 1992

"A Deus seja a glória, a honra, a força e o poder, hoje e para todo o sempre."

AGRADECIMENTOS

Apresentar uma tese, longe de ser um trabalho individual, consiste num esforço conjunto sem o qual esta tarefa seria árida, penosa e improdutiva, senão impossível. Sendo assim, quero expressar meus agradecimentos a algumas pessoas que me foram muito valiosas durante este período da minha vida.

Agradeço ao meu orientador, Prof. Rafael Santos Mendes, pela orientação e parceria na elaboração da tese, pela amizade sincera, pelo diálogo constante, pelas ricas lições acadêmicas, profissionais e de vida que, com certeza, foram e serão úteis no meu amadurecimento profissional.

Agradeço ao Prof. Waldomiro Pelagio Diniz de Carvalho Loyolla pelo esforço conjunto na aplicaçao do meu trabalho em simulação de sistemas e pela simulação dos exemplos.

Agradeço ao Conselho de Desenvolvimento Científico e Tecnológico (CNPq) que, através dos seus recursos, manteve o indispensável suporte financeiro durante o mestrado.

Agradeço aos meus pais e irmãos pelo esforço, dedicação, paciência, apoio, sabedoria e amor genuíno com o qual acompanharam todo o meu processo de crescimento e, em particular, minha estada em Campinas.

Agradeço à Silvânia, minha esposa, cujo amor, dedicação, paciência, compreensão, perseverança e longanimidade me deram forças para continuar durante estes dois anos e meio de trabalho. A ela, meu carinho especial.

Agradeço aos meus colegas de sala, José Reginaldo Hughes Carvalho e Irene Andrea Velásquez Alegre, pelo companheirismo e amizade que demonstraram.

Agradeço aos meus co-orientandos, Ely Carneiro de Paiva, Juan José Lopensino e Humberto Xavier de Araújo, pela presença, amizade e cuidado no dia-a-dia do nosso trabalho.

Agradeço aos integrantes da Igreja Evangélica Congregacional de Campinas - SP pelo carinho e acolhida demonstrados e aos integrantes da Igreja Evangélica Congregacional de Jaboatão dos Guararapes - PE pela amizade e incentivo.

Agradeço a todos que direta ou indiretamente ajudaram-me a concluir esta etapa. Agradeço à Srta. Janete Sayoko Toma pela paciente correção e digitação do manuscrito.

Por último, mas não menos importante, agradeço àquele cujo temor é o princípio de toda a sabedoria, Deus.

APRESENTAÇÃO

O presente trabalho propõe uma metodologia para estudo, análise, controle e simulação de Sistemas Dinâmicos a Eventos Discretos (DEDS) baseada em resultados da lógica temporal. Os DEDS são sistemas onde a mudança de estados só se dá com a ocorrência de eventos, que ocorrem em momentos discretos de tempo, e estes sistemas não são satisfatoriamente descritos por equações diferenciais. A Lógica temporal de Tempo Real Generalizada (GRTTL), nome do formalismo desenvolvido, deriva de sistemas lógicos já propostos e é uma generalização dos mesmos para aplicações em DEDS que apresentem comportamento não-determinísticos e com limitantes de tempo real. Uma abordagem dirigida para controle de sistemas é adotada, onde ao sistema a ser controlado (planta) é adicionado um controlador que garante o cumprimento das especificações desejadas (equações de malha-fechada).

A Simulação de sistemas é conseguida associando a GRTTL a um simulador, o Sistema de Simulação Baseado em Conhecimento (SSBC), obtendo uma metodologia de conversão do 1º para o 2º.

ABSTRACT

This work presents a temporal framework for control and simulation of Discrete Event Dynamic Systems (DEDS). DEDS are systems which the state changes occur only when an event occurs, and events occur in discrete times. This kind of systems are not well described by differential equations. The Generalized Real-Time Temporal Logic (GRTTL), the name of our formalism, is a generalization of existing models for DEDS in a Real-Time non-deterministic approach. A control point of view is adopted, where to the system to be controlled (plant) is associated a controller which may enforce the satisfaction of the desired specifications (closed-loop assertions).

The system simulation is obtained with the association between the GRTTL and a simulator, the Knowledge-Based System Simulator (KBSS), where a methodology to convert from GRTTL to KBSS is outlined.

ÍNDICE

CAPÍTULO 1

INTRODUÇÃO

Em aplicações em engenharia, têm surgido vários sistemas que não têm se adequado à modelagem pelas técnicas clássicas, originando uma busca incessante de novos conceitos e formalismos que possam preencher esta lacuna. Alguns destes sistemas apresentam características peculiares como estados discretos e mudanças de estado decorrentes pela ocorrência de eventos, que por sua vez ocorrem discretamente no tempo. Estes são os Sistemas Dinâmicos a Eventos Discretos.

Os Sistemas Dinâmicos a Eventos Discretos têm sido estudados em várias áreas da engenharia com o objetivo de se ter uma ferramenta concisa, formal e completa para a sua modelagem e controle. Entre estes formalismos pode-se citar as Cadeias de Markov, Álgebra minimax, Teoria de filas e Processos Recursivos finitos [3,13,8 e 20].

Aprofundando um pouco mais a discussão em torno dos Sistemas Dinâmicos a Eventos Discretos, observa-se que esta dificuldade reflete uma possível inadequação do atual conjunto de técnicas e conceitos (paradigma) na abordagem destes sistemas. O paradigma baseado nas equações diferenciais, ou a diferenças, tem sido insuficiente para o estudo satisfatório de sistemas como redes de comunicação, células de manufatura, sistemas de tráfego e outros desta natureza. Esta ausência tem originado a criação de uma gama extensa de procedimentos heurísticos visando uma solução do problema através de regras pré-definidas de atuação. Um dos pontos de controvérsia entre os estudiosos da área é se é possível se chegar num paradigma para os Sistemas Dinâmicos a Eventos Discretos, tal como se tem as equações diferenciais, ou se tal façanha está fadada ao insucesso.

Do ponto de vista formal, a lógica temporal vem sendo utilizada na especificação formal de estruturas de Software e Hardware, bem como na verificação da correção de Software. Suas noções baseiam-se em alguns conceitos da lógica modal, da qual a lógica temporal é um caso particular. A

lógica temporal é uma generalização da lógica clássica e como tal tem uma capacidade de expressão maior do que aquela. Esta propriedade faz com que seu uso traga vantagens na descrição de processos que se alteram com o tempo, e de propriedades de sistemas que são satisfeitas eventualmente ou de um certo ponto em diante. Recentemente, a lógica temporal começou a ser introduzida no estudo e modelagem de sistemas para controle, apresentando um grande potencial para descrever tais sistemas de maneira sucinta e formal.

Este trabalho fornece uma contribuição ao estudo e controle dos Sistemas Dinâmicos a Eventos Discretos à medida que apresenta um novo formalismo baseado em lógica temporal para a modelagem, análise e desenvolvimento de controladores para os mesmos. O formalismo proposto denomina-se Lógica Temporal de Tempo Real Generalizada (GRTTL) e generaliza resultados anteriores, obtendo um método formal para o tratamento de Sistemas Dinâmicos a Eventos Discretos não-determinísticos e com limitantes de tempo real. Uma outra contribuição apresentada refere-se à aplicação da lógica temporal na simulação de sistemas a eventos discretos. Inicialmente, são apresentados requisitos para que uma lógica temporal qualquer possa ser compatibilizada com um simulador genérico; em seguida, é feito um mapeamento das características da GRTTL para um simulador específico, o SSBC, desenvolvido na FEE/UNICAMP; finalmente, uma metodologia de transformação entre a GRTTL e o SSBC é apresentada, seguida de exemplos práticos da mesma.

Nos capítulos que se seguem, o assunto é apresentado da seguinte maneira: Inicialmente os Sistemas Dinâmicos a Eventos Discretos são descritos, bem como os formalismos mais utilizados na sua modelagem. Após, a lógica temporal é definida e todos os seus axiomas e regras de transformação analisados. Os principais formalismos no estudo de Sistemas Dinâmicos a Eventos Discretos que usam lógica temporal são apresentados e em seguida, a GRTTL é proposta, acompanhada de exemplos da sua utilização. Aspectos de simulação envolvendo a GRTTL e o SSBC são apresentados, juntamente com exemplos de sua aplicação. Conclusões e perspectivas são apresentadas ao final.

CAPÍTULO 2

SISTEMAS DINÂMICOS A EVENTOS DISCRETOS

2.1. INTRODUÇÃO

Até bem pouco tempo atrás, a maioria dos sistemas tratados pela ciência, e em particular pela engenharia, era satisfatoriamente modelado, estudado e sintetizado utilizando-se técnicas derivadas do estudo e aplicação das equações diferenciais. Qualquer sistema satisfazendo requisitos de linearidade poderia, a princípio, ser descrito por uma função de transferência que representa, em essência, o comportamento do sistema. Para sistemas discretos no tempo, ou discretizados, as equações diferenciais metamorfoseiam-se nas equações a diferenças, mantendo contudo a mesma idéia.

Em aplicações mais recentes, tornaram-se evidentes alguns casos onde alguns sistemas importantes que apareciam em várias situações não podiam ser descritos satisfatoriamente por equações diferenciais, ou à diferenças. Neles, o estado só muda em instantes discretos (não previsíveis) de tempo e isto só com a ocorrência de um evento. Estas mudanças de estado podem ser determinísticas ou não-determinísticas. A estes sistemas, convencionou-se chamar de Sistemas Dinâmicos a Eventos Discretos (DEDS) em oposição aos Sistemas Dinâmicos a Variáveis Contínuas (CVDS) já bem conhecidos pela comunidade científica e modelados por equações diferenciais. Exemplos de DEDSs são normalmente encontrados em sistemas de comunicações, sistemas de tráfego, automação da manufatura, etc. A figura 1 abaixo representa uma trajetória de estado típica para os DEDSs. Os trechos constantes representam os períodos (tempos) de retenção num dado estado e as mudanças de estados sinalizam a ocorrência de eventos.

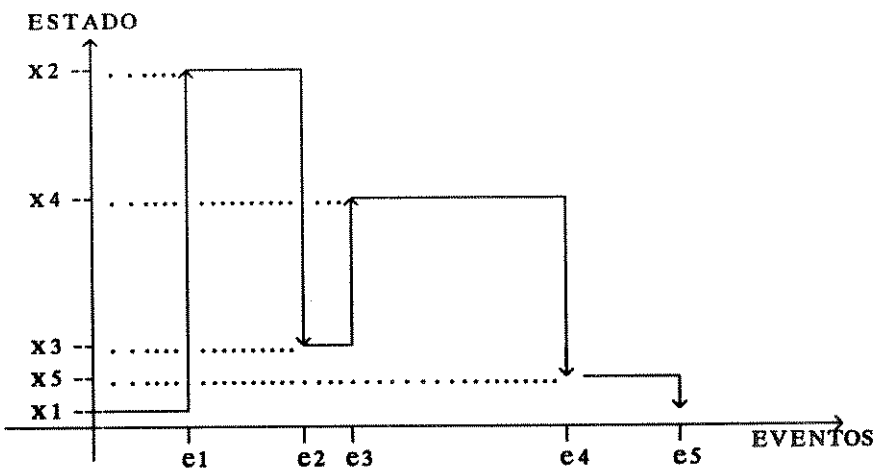


figura 1

Uma outra diferença entre os DEDSs e os CVDSs está na compreensão do

que seja o estado do sistema. Para os CVDSs, de uma maneira formal, o estado do sistema é o conjunto de informações que junto como as entradas caracterizam os próximos estados. Para os DEDS, a noção de estado nada mais é do que o valor das variáveis do sistema, ou seja, o estado livre ou ocupado de servidores numa rede de comunicação, o status de uma máquina numa linha de montagem, etc. Vê-se que a noção de estado pra um DEDS é mais informal e incluída na noção de estado para um CVDS.

Neste capítulo, os Sistemas Dinâmicos a Eventos Discretos são apresentados e as suas principais características delineadas de maneira sucinta. Uma atenção especial é dada aos formalismos mais utilizados na modelagem dos DEDSs, tendo-se o cuidado de descrevê-los com algum detalhe. Referências mais detalhadas sobre cada formalismo são dadas em cada subseção.

2.2. CARACTERÍSTICAS DOS DEDS

Enquadrando os DEDSs como um conjunto autônomo e bem definido de sistemas, algumas características gerais podem ser ressaltadas. São elas:

i) **Eventos ocorrem em tempos discretos (não previsíveis) e têm valores discretos** - Esta ocorrência discreta causa problemas para se definir um ferramental teórico que descreva o sistema em todas as suas fases.

ii) **Os processos são orientados a eventos e não a tempos** - Isto simboliza o fato de que as mudanças no sistema, e consequentemente nos processos, ocorrem a partir da ocorrência de eventos.

iii) **Não-determinismo** - Os processos são normalmente não-determinísticos, no sentido que são permitidas escolhas por algum mecanismo não-modelado pelo projetista.

iv) **Interação** - A interação entre processos (DEDS) pode ser síncrona ou assíncrona.

v) **Tempo-Real** - É comum a existência de prazos a serem cumpridos.

Os Sistemas Dinâmicos a Eventos Discretos apareceram recentemente

enquanto um campo de pesquisas autônomo, mas já existiam há bastante tempo como um ítem de estudo em outras áreas. De uma maneira mais direta, os DEDS têm-se manifestado em áreas como Pesquisa Operacional, Teoria de Controle, Teoria da Computação e Inteligência Artificial. Em cada uma delas os DEDSs têm sido abordados por um prisma diferente, como é mostrado a seguir:

1) **Pesquisa Operacional** (PO) - Para a pesquisa operacional, os DEDSs são vistos como sistemas que se guiam através de regras de operação.

2) **Teoria de Controle** (TCt) - Os conceitos clássicos da teoria de controle, como estabilidade, controlabilidade, observabilidade, etc, são adaptados para fornecerem informações úteis no estudo dos DEDSs. Vale salientar que todo o esforço nesta área tenta adaptar o velho paradigma de controle a este tipo de sistema.

3) **Teoria da Computação** (TC) - A identificação de DEDS na teoria da computação é um fato corriqueiro visto que um programa pode ser encarado como um DEDS onde cada passo do mesmo pode ser associado com um estado, e a chegada de um comando ou mensagem como um evento que altera o estado do sistema. Os algoritmos para evitar regiões críticas (semáforos), bem como restrições de existência despontam como as ferramentas mais usadas em TC para o controle de DEDS.

4) **Inteligência Artificial** (AI) - A interação de sistemas Inteligentes com o homem tem suscitado a adoção de uma interface homem-máquina cada vez mais elaborada para esta tarefa. Neste aspecto, a Inteligência Artificial tem conseguido muitos resultados sobre como a comunicação de DEDSs com o operador humano deve se processar, bem como as restrições impostas sobre a implementação destes sistemas. A abordagem aqui é feita quase que exclusivamente em termos de regras heurísticas, podendo envolver a utilização de lógica nebulosa.

2.3. MODELOS PARA DEDS

Para se ter um estudo sistemático de Sistemas Dinâmicos a Eventos Discretos é necessário ter um modelo que represente com razoável proximidade o comportamento destes sistemas. O fato dos DEDSs terem surgido num contexto

multidisciplinar fez com que se tivesse vários modelos complementares. De uma maneira resumida, ainda não existe uma estratégia de modelagem para os DEDSs que tenha atingido tal supremacia perante os demais que tenha se tornado um paradigma. Sendo assim, vários modelos têm disputado ferenhamente tal posição de destaque tendo em vista ampliar o seu poder de modelagem. Como mencionado em [1], o formalismo que se arvore como ideal para a modelagem de DEDS deve preencher as seguintes características:

1) Tratar a natureza descontínua dos eventos discretos;
2) Medir de maneira contínua o desempenho;
3) Possibilitar uma abordagem Probabilística;
4) Possibilitar análise hierárquica;
5) Capturar a dinâmica do comportamento transitório dos DEDS e
6) Não apresentar explosão computacional.

Dentre os formalismos usados na modelagem de DEDS, destacam-se os seguintes:

* **Cadeias de Markov/modelos autômatos-** (Ex.: Redes de Petri, Máquinas de estado estendidas,etc);
* **Modelos de Filas;**
* **Modelos de Álgebra Minimax;**
* **Modelos baseados em computador e**
* **Modelos de processos semi-Markovianos generalizados.**

Dentre estes modelos pode-se identificar vários aspectos que os dividem. O modelo pode ser temporizado ou não-temporizado; lógico, algébrico ou de desempenho; determinístico ou estocástico. A figura 2,proposta por Ho [1], apresenta os formalismos divididos de acordo com as suas características básicas:

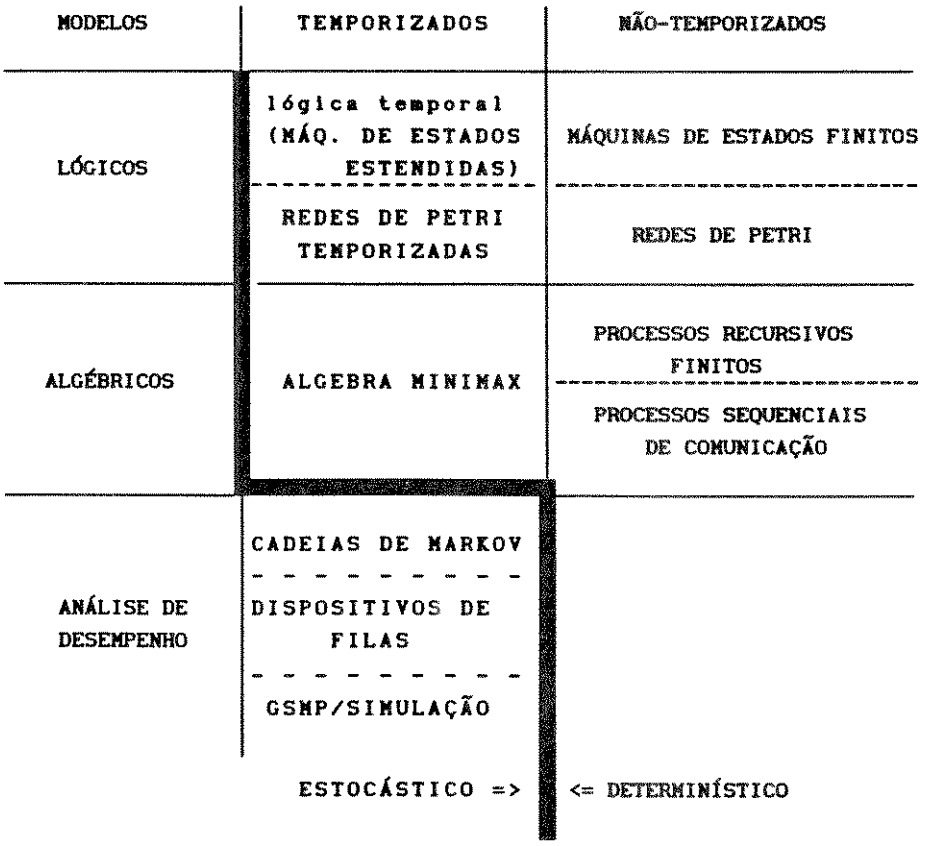


figura 2

Segue-se uma breve apresentação de cada formalismo listado tendo-se o cuidado de ressaltar suas características para um tipo de objetivo pretendido.

2.3.1. Cadeias de Markov

Em 1907, A.A. Markov iniciou a publicação de uma série de artigos desenvolvendo uma teoria que mais tarde se denominaria de Cadeias de Markov. Um fato interessante é que ele descobriu estas cadeias estudando a alternância de ocorrência das vogais e consoantes na literatura russa, mais especificamente no romance Eugène Oneguine, de Pouckine. Foram considerados espaços de estados finitos e mostrou-se que o limite das probabilidades de transição entre dois estados i e j, $P^n(i,j)$ existe, no caso de uma cadeia aperiódica com só uma classe recorrente (só um grupo de estados recorrentes, ou seja, que serão visitados pela primeira vez num tempo finito).

Formalmente, pode-se estudar as cadeias de Markov utilizando-se de

três abordagens distintas, mas interligadas. São elas:

Abordagem Probabilística

Seja X_n, n=0,1,..., um processo estocástico que toma valores em um conjunto de valores finito ou enumerável, como por exemplo o conjunto dos números inteiros {0,1,2,...}. Se $X_n=i$ então diz-se que o processo está no estado i no tempo n. Sempre que o processo está num estado i existe uma probabilidade P_{ij} de transição para o estado j, esta probabilidade é dada como segue:

$$P\{ X_{n+1} = j \,/\, X_n = i,\ X_{n-1} = i_{n-1},\ldots,X_1 = i_1,\ X_0 = i_0\} = P_{ij}$$

Para todos os estados $i_0,i_1,\ldots,i_{n-1},i,j$ e $n\geq 0$. Este processo é conhecido como uma **Cadeia de Markov**.

De uma maneira mais geral, uma Cadeia de Markov é dita ser de ordem m, se a distribuição condicional de qualquer estado X_{n+1} só depende dos m últimos estados precedentes, ou seja:

$$P\{X_{n+1}=j/X_n=i,X_{n-1}=i_{n-1},\text{—},X_1=i_1,X_0=i_0\}=P\{X_{n+1}=j/X_n=i,X_{n-1}=i_{n-1},\ldots,X_{n-m+1}=i_1\}$$

Para o caso onde se tem m=1, a Cadeia de Markov é dita ser de ordem 1. Neste caso, pode-se dizer que a probabilidade condicional da ocorrência de $X_{n+1}=j$ só depende do estado $X_n=i$, ou seja, o estado atual, se X_{n+1} é o próximo estado. Frequentemente, quando se fala de uma Cadeia de Markov refere-se só às cadeias de ordem 1, de modo que as próximas referências às Cadeias de Markov sempre serão àquelas de ordem 1. A probabilidade p(i,j) é denominada de probabilidade de transição. Se a probabilidade de transição não depende do momento em que ocorre a transição, mas só dos estados i e j envolvidos, a Cadeia de Markov é dita ser homogênea, ou estacionária, no tempo.

Abordagem Algébrica

Esta abordagem utiliza noções da Álgebra linear como valores próprios e vetores próprios. Considere uma Cadeia de Markov homogênea. Pode-se formar uma matriz de probabilidade de transição do estado i para j. A matriz P tem a seguinte forma:

$$P = \begin{bmatrix} P_{00} & P_{01} & P_{02} \ldots \\ P_{10} & P_{11} & P_{12} \ldots \\ \vdots & \vdots & \vdots \\ P_{n0} & P_{n1} & P_{n2} \ldots \\ \vdots & \vdots & \vdots \end{bmatrix}$$

Onde $P_{ij} \geq 0, i, j \geq 0; \sum_{j=0}^{\infty} P_{ij} = 1, \quad i = 0,1,\ldots$

Utilizando-se a Teoria de Matrizes pode-se associar muitos dos resultados da mesma com interpretações práticas nos sistemas descritos pelas Cadeias de Markov.

Abordagem por Grafos

Esta abordagem funciona mais como um recurso visual e de análise qualitativa do que um ferramental preciso para o estudo de sistemas usando as Cadeias de Markov. Se o espaço de estados for finito, com cardinalidade r, sempre se pode associar à matriz P um grafo G= (E,Γ), onde E representa os estados e Γ é uma relação definida como segue: $\Gamma(E_i) = \{ E_j, p_{ij}>0\}$. De uma maneira simples, o conjunto E define os nós do grafo enquanto a relação Γ define os arcos e seus respectivos pesos. A este grafo pode ser associada uma matriz de adjacência X, como segue:

$$\forall\ i,j \in \{1,r\}\ ; \begin{cases} X_{ij}=0 \text{ se } p_{ij}=0 \\ \\ X_{ij}=1 \text{ se } p_{ij}>0 \end{cases}$$

Como exemplo, considere-se a matriz de transição P abaixo apresentada:

	E_1	E_2	E_3	E_4
E_1	1/2	1/4	1/4	0
E_2	0	1/2	0	1/2
E_3	1/3	0	1/3	1/3
E_4	1/2	0	1/2	0

onde cada elemento P_{ij} representa a probabilidade de transição do estado E_i para o estado E_j. O grafo correspondente à matriz P é o seguinte:

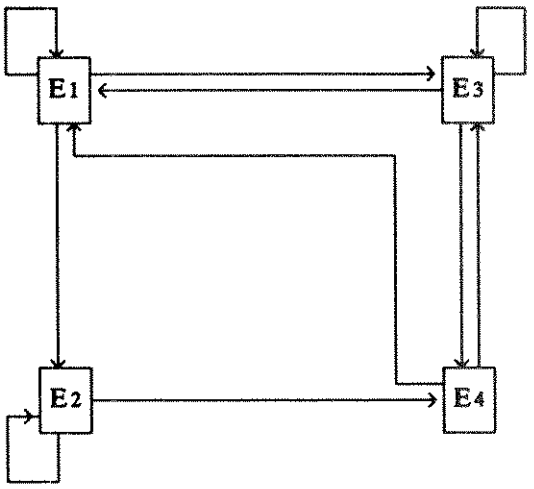


figura 3

Na aplicação das Cadeias de Markov, as três abordagens aqui expostas são usadas conjuntamente. Para Sistemas Dinâmicos a Eventos Discretos, as cadeias de Markov representam uma alternativa de formalismo, onde a probabilidade de ocorrência do estado seguinte só depende do estado atual. Uma característica destes processos é que eles não têm memória, ou seja, as informações passadas não influenciam nas decisões futuras. Um exemplo típico de modelagem de um DEDS por Cadeias de Markov é o sistema composto de duas máquinas e um buffer, apresentado em [2], o qual é descrito como um processo Nascimento-morte. Maiores informações sobre Cadeias de Markov podem ser encontradas em [3], [4] e [5], e suas aplicações para DEDSs em [1] e [2].

2.3.2. TEORIA DE FILAS

Sistemas de filas representam um exemplo de uma classe abrangente de sistemas dinâmicos, referidos como Sistemas de Fluxo. Um sistema de fluxo é aquele nos qual existe um fluxo, ou um movimento, ou ainda uma transferência de artigos através de um ou mais canais de um ponto a outro do sistema.

Quando se analisa um sistema de fluxo é comum dividi-lo em duas classes: Fluxo seguro ou estável (steady) e não-estável (unsteady). Um sistema de fluxo estável é aquele no qual se conhece exatamente o valor de fluxo durante o período de interesse. O tempo relacionado ao início deste fluxo no canal, e a quantidade de fluxo que se localiza neste canal são conhecidos e constantes. Da segunda classe fazem parte problemas de fluxo Aleatório ou

estocástico. Isto significa que o tempo de serviço (utilização dos canais) é imprevisível ou incerto e a quantidade da demanda do fluxo que se localiza no canal é imprevisível também.

Na especificação de um sistema de filas, deve-se identificar os processos estocásticos que descrevem a sequência de chegada tanto quanto a estrutura e a disciplina de atendimento. Geralmente, o processo de chegada é descrito em termos da distribuição de probabilidade do tempo entre chegadas denotado por A(t), onde A(t) = P[tempo entre chegadas≤t]. Na maioria das filas, estes tempos são variáveis aleatórias identicamente distribuidas (independentes). Além da distribuição do tempo entre chegadas, ou seja A(t), uma outra quantidade estatística que deve ser descrita é o montante de tempo de permanência neste canal, denomidado de tempo de serviço. A distribuição probabilística do tempo de serviço é dada por B(x) = P[tempo de serviço≤x]

Com estes dois parâmetros, A(t) e B(x), juntamente com outros derivados dos mesmos, é possível construir várias estruturas de filas para se analisar DEDSs. Existem vários tipos de filas diferentes, cada uma com suas peculiaridades e aplicações. Na descrição de tipos de filas, geralmente se usa uma notação especial que rapidamente comunica os atributos do sistema em questão. Ele é formado por um descritor de três partes A/B/m, onde A e B decrevem a distribuição probabilística do tempo entre chegadas e do tempo de serviço, respectivamente, e m fornece o comprimento máximo da filas de espera. A e B podem assumir os seguintes valores:

M - exponencial
E - Erlangiana
H - Hiperexponencial
D - Determinística
G - Geral

No caso do processo nascimento-morte descrito nas cadeias de Markov, o sistema era do tipo M/M/1 [2]. Para uma bibliografia mais extensa sobre a teoria de filas recomenda-se as referências [6], [7], [8].

2.3.3. PROCESSOS GENERALIZADOS SEMI-MARKOVIANOS

A idéia explorada aqui é definir um tipo particular de processo estocástico, chamado de Processo Generalizado Semi-Markoviano (GSMP), que objetiva capturar a estrutura dinâmica dos DEDSs.

Pode-se considerar os DEDS como modelos tendo trajetórias constantes por partes. Em função desta visão, se (S(t): t≥0) é o processo de saída correspondente ao sistema a evento discreto, tem-se a seguinte expressão:

$$S(t) = \sum_{n=0}^{\infty} S_n I(\Lambda(n) \leq t < \Lambda(n+1))$$

Onde tem-se:

S_n - representa o estado (na saída) quando da ocorrência da n-ésima transição.

I(A) - representa uma função que assume os valores um ou zero dependendo se A ocorre ou não.

$\Lambda(n)$ - é o instante no qual a n-ésima transição ocorre. É importante notar que $0 = \Lambda(0) < \Lambda(1) < \ldots$

$\Delta_{n+1} = \Lambda(n+1) - \Lambda(n)$ é definido como o tempo entre a n-ésima e a n-ésima-primeira transições de S(t).

Para caracterizar a dinâmica do processo de saída, S(t), assume-se a existência de uma sequência $X = (X_n: n \geq 0)$ que descreve a evolução no tempo do estado interno do sistema. O estado S(t) e o tempo de permanência $\Delta(n)$ num estado estarão relacionados com X da seguinte forma: $(S_n,\Delta_n)=(h_1(X_n),h_2(X_n))$, onde X_n, h_1 e h_2 são conhecidos. Dado S(t): $0 \leq t \leq \Lambda(n)$ pode-se estender o cálculo para o intervalo $(\Lambda(n),\Lambda(n+1))$ obtendo-se X_{n+1}. Com X_{n+1} pode-se calcular Δ_{n+1}, $\Lambda(n+1)$ e S_{n+1} da seguinte maneira:

$$\begin{aligned} \Delta_{n+1} &= h(X_{n+1}) \\ \Lambda(n+1) &= \Lambda(n) + \Delta_{n+1} \\ S_{n+1} &= h_1(X_{n+1}) \end{aligned}$$

A abordagem usando o GSMP na modelagem de DEDSs tenta formalizar procedimentos ou linguagens de simulação para DEDSs. As duas variáveis citadas, o estado e o tempo de permanência num estado, são as informações

necessárias em uma dada transição. Pela ótica de simulação, a sequência de evolução seria a seguinte:

1) Para cada estado, tem-se uma lista de eventos factíveis, que consiste de um conjunto de eventos habilitados quando o sistema está neste estado;

2) Para cada evento na lista, gera-se um prazo para a ocorrência deste evento, se nenhum outro evento ocorrer primeiro desabilitando-o; aquele evento que tiver o menor prazo de ocorrência, ocorrerá;

3) Condições lógicas associadas com funcões de distribuição de probabilidade, indicam qual o próximo estado a ser alcançado; uma nova lista de eventos factíveis é gerada e o ciclo se repete.

Para uma análise mais profunda do GSMP, recomenda-se as referências [1], [2] e [9].

2.3.4. MÁQUINAS DE ESTADOS FINITOS /SUPERVISOR

Considere um conjunto de estados S e um conjunto de eventos E. A cada mudança de estado está associado um evento. Ao modelar uma planta por este conjunto de estados, algumas transições são permitidas, outras não. Esta proibição não é devida a imposições externas (especificações), mas deve-se à dinâmica da planta. Caso só se tenha acesso às sequências de eventos na execução do sistema, pode-se identificar sequências permitidas e não-permitidas. Além destas, o sistema pode funcionar em conjunto com um controlador que permita ou não o acesso a certos estados. Este controlador restringe ainda mais o universo de sequências permitidas. Uma linguagem seria o conjunto de todas as sequências de eventos desejadas. Uma linguagem associada às especificações em malha-fechada de um sistema qualquer seria controlável se existisse um controlador capaz de gerar uma linguagem desejada.

Dado um sistema, pode-se ter acesso aos seus estados e eventos ou só aos seus eventos. Se o sistema é tal que só se pode observar os seus eventos, uma réplica é construída e passa a ser alimentada com os eventos gerados pela planta. Deste modo, o controlador tem um estimador que estima (exatamente) qual o estado da planta, possibilitando uma ação eficaz de controle. Ao

estimador mais uma função de controle $\phi(w)$ denomina-se de **Supervisor**. É comum ter-se a mesma ação de controle para vários estados distintos, significando que não faz diferença para o controlador a distinção entre estes estados. Na construção do supervisor, isto pode ser levado em conta, pois pode-se simplificar o diagrama de estados da planta eliminando estas redundâncias. Ao supervisor na sua forma mais "reduzida" dá-se o nome de **Supervisor quociente**.

Pelo que foi exposto acima, percebe-se que a figura do supervisor aproxima-se bastante com o estimador da teoria clássica de controle estocástico, onde para alguns sistemas é necessário ter um estimador de estado que fornece uma aproximação do estado do sistema. Um problema comum nesta área, o Problema de Controle Supervisório, versa sobre a possibilidade ou não de se implementar um controlador que permita a ocorrência de uma certa linguagem. Caso não seja possível tal implementação, tenta-se, pelo menos, obter a linguagem mais próxima da requerida. Conceitos clássicos como controlabilidade, observabilidade e estabilidade são transportados para este formalismo. Maiores detalhes sobre o mesmo podem ser encontrados em [1] e [10].

Como uma crítica feita a esta abordagem aponta-se o fato da complexidade computacional crescer bastante com o número de estados do sistema envolvido. Entre as tentativas feitas para superar estes obstáculos tem-se supervisão modular, descentralizada e hierárquica [11] e [12]. Além de controle da manufatura, outras áreas de aplicação desta teoria são protocolos de comunicação e gerenciamento de base de dados.

2.3.5. ÁLGEBRA MINIMAX

Os modelos usando Álgebra Minimax foram propostos em [13] onde só tratavam de sistemas com eventos determinísticos. Recentes trabalhos tentam ampliar a faixa de aplicação deste formalismo para sistemas com características estocásticas. Uma rápida noção dos conceitos envolvidos com a Álgebra Minimax é feita e, em seguida, a modelagem de um exemplo simples é fornecida.

A Álgebra Minimax está baseada na definição de uma estrutura denominada Dióide ($\mathcal{D}$) que é um conjunto provido de duas operações. Um exemplo de dióide é o conjunto $\mathbb{Z} \cup \{-\infty\}$, onde $\mathbb{Z}$ é o conjunto dos nº inteiros, provido das operações seguintes:

Adição: $a \oplus b \equiv \text{Max}(a,b)$

Multiplicação: $a \otimes b \equiv a + b$

Onde:

* Tanto $\oplus$ como $\otimes$ são associativas.
* $\oplus$ é comutativa.
* $\otimes$ é distributiva em relação a $\oplus$.
* Existe "e" e "ε" $\in \mathcal{D}$ tal que $\forall\ a \in \mathcal{D}$, $\begin{cases} a \oplus \varepsilon = a; \\ e \otimes a = a \otimes e = a; \end{cases}$
* O elemento neutro ε é absorvente para $\otimes$, i.e. $\forall a \in \mathcal{D}$, $\varepsilon \otimes a = a \otimes \varepsilon = \varepsilon$;
* $\oplus$ é idempotente, i.e. $\forall a \in \mathcal{D}$, $a \oplus a = a$;

Se $\otimes$ é comutativa, $\mathcal{D}$ é chamado de Dióide Comutativo. Com a Álgebra Minimax, é possivel escrever as especificações da planta de uma maneira bem familiar à teoria clássica de controle, lembrando que as operações de adição e de multiplicação são como descritas acima. Conforme [2], apresenta-se a seguir um exemplo simples de representação de DEDS através da Álgebra Min-Max. Considere um sistema onde o estado inicial é dado por (x_0,y_0) e suas equações dinâmicas sejam dadas por

$$x_1 = x_0 + a$$
$$x_2 = x_1 + a$$
$$y_1 = \max\{y_0+b,\ x_1+b\}$$

Onde a e b são constantes e as variáveis que nos interessam são X_{n+1} e Y_n. Em Álgebra Minimax estas mesmas equações seriam:

$$x_2 = x_1 \otimes a$$
$$y_1 = x_1 \otimes b \oplus y_0 \otimes b$$

Para descrever o comportamento de sistema através de equações matriciais, escolhe-se um número negativo H tal que $-H < x_1 - y_0$ e tem-se

$$\begin{pmatrix} x_2 \\ y_1 \end{pmatrix} = \begin{pmatrix} a & -H \\ b & b \end{pmatrix} \begin{pmatrix} x_1 \\ y_0 \end{pmatrix}$$

Onde é importante lembrar que a adição e a multiplicação são como definidas no início. Se é definido

$$Z_n = \begin{pmatrix} x_{n+1} \\ y_n \end{pmatrix} \quad e \quad M = \begin{pmatrix} a & -H \\ b & b \end{pmatrix}$$

Tem-se $Z_1 = MZ_0$, $Z_2 = MZ_1 = M^2Z_0$. O valor de M^n pode ser calculado da seguinte maneira: Inicialmente define-se $c = a - b$ e $J = H + a$. Então tem-se $b = a - c = a \otimes (-c)$ e $-H = a \otimes (-J)$. Deste modo, $Z_n = M^nZ_0 = a^nK^nZ_0 = (na)K^nZ_0$ onde K é dado por

$$K = \begin{pmatrix} 0 & -J \\ -c & -nc \end{pmatrix} \text{ e } K = \begin{pmatrix} 0 & -J \\ -c & \max(-nc\ -J-c) \end{pmatrix}$$

Onde K^n pode assumir dois valores:

$$K^n = \begin{pmatrix} 0 & -J \\ -c & -J-c \end{pmatrix} \text{ se } (n-1)c > J \quad e \quad K^n = \begin{pmatrix} 0 & -J \\ -c & -nc \end{pmatrix} \text{ se } (n-1)c < J$$

Em Álgebra Minimax, uma matriz M é dita ser de ordem periódica d se e somente se existe um inteiro n_0 tal que para todo $n > n_0$, $M^{n+d} = M^n$. Sendo assim, a mariz K é de ordem periódica 1. Para um P e Z_0 definidos abaixo

$$P = \begin{pmatrix} 0 & -J \\ c & -J \end{pmatrix} \text{ e } Z_0 = \begin{pmatrix} x_1 \\ y_0 \end{pmatrix}$$

e para n suficientemente grande tem-se $Z_n = (na)Pz_0$, ou $X_{n+1} = (n+1)a$ e $Y_n = (n+1)a + (-c) = na + b$. Em termos intuitivos a idéia é a seguinte: depois do início, ou transitório, os valores de Y_n são os mesmos de X_n acrescidos de b. Para um aprofundamento na teoria dos dióides, e consequentemente, na Álgebra minimax, recomenda-se [14] e [15].

2.3.6. REDES DE PETRI

O formalismo das Redes de Petri procura representar o sistema através de um conjunto de **lugares** (places), **conexões** e **portas** onde a dinâmica do sistema é simulada com a ajuda de **senhas** (tokens). Nas linhas que se seguem são definidos precisamente todos estes conceitos. Vale salientar que a

definição de Redes de Petri apresentada neste trabalho consiste numa variação da já consagrada na literatura. Isto é feito com o intuito de enriquecer o trabalho, conferindo-lhe um aspecto mais inovador. A abordagem clássica das Redes de Petri pode ser encontrada em [17], enquanto que a aqui utilizada é extraída de [18].

Seja uma Rede de Petri definida pelos seguintes conjuntos: um conjunto de lugares L, um conjunto de conexões T e um conjunto de Portas P. Uma porta conecta uma conexão a um lugar ou um lugar a uma conexão. Um lugar pode se conectar a uma conexão de quatro maneiras diferentes, que são os quatro tipos de portas existentes. Uma porta pode ser caracterizada como 1) porta de entrada ou saída e 2) porta alteradora ou restauradora.Uma porta é dita ser uma porta de entrada se ela é representada por uma seta entrando na conexão e de saída se ela é representada por uma seta saindo da conexão. Uma porta alteradora é aquela onde só existe um sentido possível para a porta e restauradora se existem dois sentidos possíveis para a mesma. A figura abaixo, como encontrada em [18], representa de maneira clara as relações entre os diversos tipos de portas.

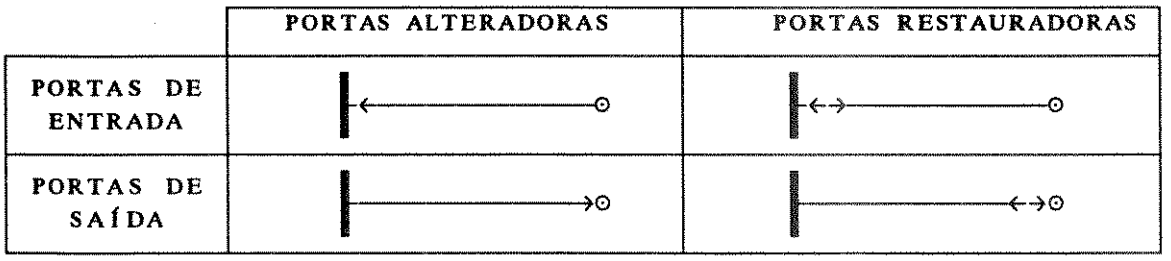


figura 4.

Os lugares da rede representam unidades funcionais **passivas**, i.e., unidades funcionais do tipo estado. As conexões representam unidades funcionais **ativas**, i.e., unidades funcionais do tipo transição. Formalmente tem-se que uma Rede de Petri RP é dada por (L ,T , P), Onde:

$$L = \{l_1, l_2, \ldots, l_n\}$$
$$T = (t_1, t_2, \ldots, t_m)$$
$$P = P_e \cup P_s$$
$$P_e \subseteq L \times T$$
$$P_o \subseteq T \times L$$

Onde P_e representa as portas de entrada (lugares de entrada de conexão) e P_o representa as portas de saída (regiões de saída de conexão), ambas

alteradoras. Com estes elementos é possível descrever a planta de uma maneira estática. Para simplicidade da notação não foram incluidas as portas restauradoras entre os tipos de portas possíveis. A diferença entre uma porta alteradora e uma restauradora será fornecida após a definição de redes marcadas.

A dinâmica da planta é dada por um ente abstrato chamado de **token** (senha). Os tokens movem-se de lugar para lugar, simbolizando o andamento do processo. Diz-se de um lugar que ele está **marcado** quanto ele possui um ou mais tokens. Ao se iniciar o processo, deve-se colocar os tokens em alguns lugares simbolizando o estado inicial. Este processo é conhecido como **marcação**. Quando uma conexão é disparada, um ou mais tokens passam de um lugar para outro. O número total de tokens pode se alterar durante a execução. Para ocorrer o disparo de uma conexão, é necessário que todas as condições de entrada estejam vigorando e nenhuma condição de saída esteja vigorando. Em outras palavras, todos os lugares conectados à conexão mencionada via portas de entrada devem estar marcados e nenhum lugar conectado via portas de saída deve estar marcado.

Como já mencionado, toda vez que ocorre um disparo de uma conexão, os tokens passam de um lugar marcado para outro não marcado, nesta versão das Redes de Petri. Suponha que se queira que o disparo de uma conexão não altere a marcação de um lugar, para um dado propósito. Uma tentativa frustrada seria elaborar uma rede onde se tenha uma porta de entrada e de saída conectando o lugar e a conexão em questão, pois como condição para o disparo desta última o lugar de destino não pode estar marcado. O disparo nunca ocorreria. Uma maneira de contornar isto é conectando o lugar à conexão via uma porta restauradora de entrada. Isto simboliza que o token tem o efeito de habilitar o disparo, mas o lugar de entrada em questão continua marcado (daí o nome restauradora). Analogamente, uma porta restauradora de saída tem a propriedade de permitir o disparo da conexão sem a marcação do correspondente lugar de destino. De uma maneira geral, pode-se dizer que uma porta restauradora faz o mesmo papel da alteradora na habilitação de disparos, não desencadeando, contudo, os seus efeitos (aparecimento/desaparecimento de tokens).

Até agora só se falou de Redes de Petri que não envolvem temporalidade nas suas conexões, ou seja, não existem limites de tempo para que determinadas conexões ocorram. Uma das maneiras disto pode ser contornado é a definição das Redes de Petri temporizadas. Nestas redes, tem-se uma função de distribuição de probabilidade do tempo em que o token ficará retido num determinado lugar até poder transicionar para o lugar seguinte. Graficamente,

uma conexão temporizada é representada por uma barra vazia (oca) enquanto que uma conexão livre, ou seja sem limitantes de tempo, é vista como uma barra cheia. Os lugares são vistos como circunferências com ou sem pontos pretos no seu interior, simbolizando a existência ou não de tokens nele, respectivamente; e as portas como segmentos orientados de reta ligando um lugar a uma conexão e vice-versa. A figura 5 ilustra uma Rede de Petri.

As Redes de Petri são utilizadas largamente na modelagem de sistemas de comunicação, células de manufatura, bem como dispositivos para computação (impressoras de linha, acionadores de disco, etc). Maiores detalhes podem ser vistos em [2, 16, 17, 18 e 19].

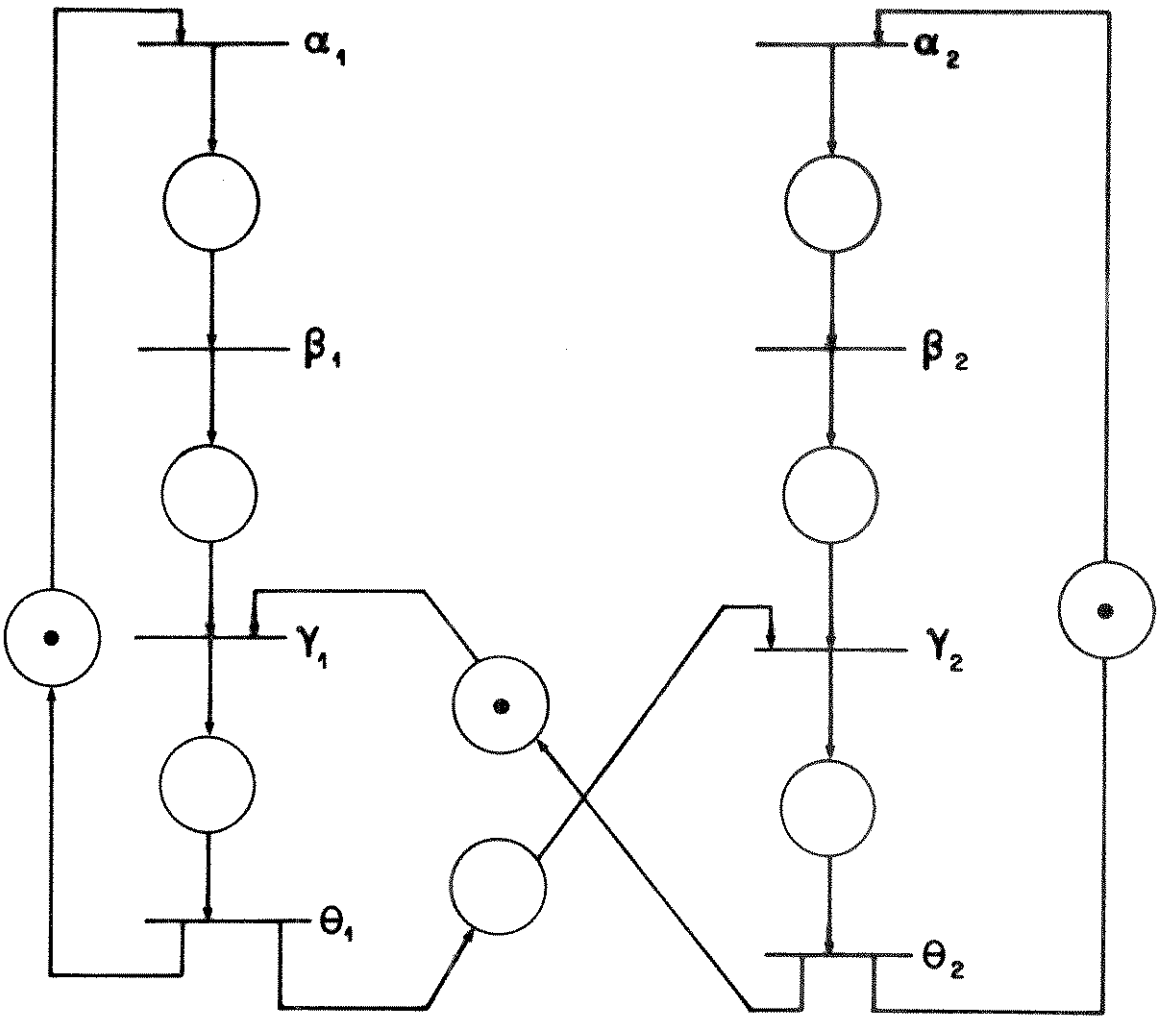


figura 5

2.3.7. PROCESSOS RECURSIVOS FINITOS

Dentre os modelos para o estudo de DEDS, os Processos Recursivos Finitos (FRP) aplicam-se quando se deseja uma análise do comportamento lógico do sistema, sem entrar em questões de comportamento temporal (desempenho). Além deste aspecto atemporal, os FRPs possuem uma mecanismo de modelagem bem diferente das abordagens vistas até agora. De uma forma bem geral, um processo Recursivo finito é uma representação matemática finita de um conjunto infinito L.

Seja A um conjunto fixo de eventos, e A^* o conjunto de todos os subconjuntos de A. Um **Processo** (determinístico ou não) P é uma tripla (trP, αP, τP), onde:

trP $\subset A^*$ - Informa o conjunto de **traces** que P pode executar. Um **trace** de P é uma sequência de eventos possível de ser executada a partir do estado inicial.

αP: trP $\longrightarrow 2^A$ - αP(s) informa qual o próximo (ou próximos) eventos depois da ocorrência de α.

τP: trP $\longrightarrow \{0,1\}$ - É a função de terminação, e τP(s) especifica se P termina ou continua depois de executar s.

Formalmente, um Processo Recursivo Finito é composto de um conjunto Π chamado de espaço de processos. Neste conjunto estão todos os processos possíveis. Cada processo é uma coleção de sequências finitas ou infinitas. Cada sequência de um processo é constituída por eventos (eventos estes que representam todas as ocorrências possíveis do sistema em questão). Além disto, existe um conjunto de operadores básicos sobre estes eventos. Estes operadores são em número de cinco e são: a **escolha determinística**, a **composição síncrona**, a **composição sequencial**, a **mudança global** e a **mudança local do conjunto de eventos**. Eles mapeiam elementos de Π em Π. Com este conjunto de operações pode-se simular todas as saídas desejadas pelo sistema. Como uma ilustração, suponha que se deseje representar um processo cuja sequência de ocorrência de eventos é dada por ($\alpha\beta\gamma\alpha\beta\gamma$...). Nos Processos Recursivos Finitos, a modelagem desta sequência é feita através da seguinte expressão recursiva:

$$X = (\alpha \rightarrow (\beta \rightarrow (\gamma \rightarrow X)))$$

Esta expressão demonstra a utilização de um dos operadores possíveis sobre Π que é o operador escolha determinística "$\rightarrow$" que simplesmente serializa a ocorrência de um evento antes de um processo. Utilizando artifícios como o citado acima, pode-se ter uma representação finita de um processo infinito. Uma das vantagens deste formalismo é que a expressão final fica em muito semelhante às fórmulas dadas pelas equações diferenciais para os CVDS. Em termos práticos, um processo Y é chamado de um processo recursivo finito (FRP) se ele pode ser representado como

$$X = f(Y)$$
$$Y = g(X)$$

Onde f e g são funções sobre Π, e f satisfaz algumas determinadas condições. Como já dito, Uma visível vantagem desta representação está na analogia com a equação a diferenças para controle:

$$x(t+1) = f(x(t)),\ y(t) = g(x(t)),\ t = 0,1,...$$

Para um estudo mais acurado sobre os Processos Recursivos Finitos recomenda-se as referências [1] e [20].

2.4. CONCLUSÃO

Os Sistemas Dinâmicos a Eventos Discretos vêm contando atualmente com um número crescente de pesquisadores envolvidos no seu estudo e análise. As abordagens apresentadas neste capítulo mostram que dependendo da base matemática usada, dos critérios que se quer estudar e das aplicações envolvidas, pode-se ter uma gama muito grande e diversificada de métodos de análise. Apesar desta falta de consenso quanto a um formalismo padrão, numa coisa os autores concordam: com as noções de estado e tempo de retenção para descrição das trajetórias de DEDSs e na caracterização dos modelos de DEDSs pelos critérios de desempenho,lógicos e algébricos.

Nos próximos capítulos os DEDSs são modelados através de formalismos baseados em lógica temporal.

CAPÍTULO 3

LÓGICA TEMPORAL

3.1. INTRODUÇÃO

A lógica é um dos ramos do conhecimento que tem sido estudado desde os tempos remotos. A lógica clássica tem-se baseado nos conceitos de verdadeiro ou falso, onde através deles pode-se sempre determinar se uma dada sentença declarativa é verdadeira ou falsa, sob determinadas condições. Por trás desta propriedade de sempre se poder atribuir o valor verdadeiro ou falso a uma sentença está o conceito de funcional-veritatividade. Um operador funcional-veritativo é aquele em que dados os valores verdade dos seus operandos sempre se pode decidir o valor verdade da expressão inteira. Como exemplo disto temos a fórmula p ∧ q, onde "∧" tem o sentido do "**e**" lógico, em que a tabela-verdade abaixo fornece o valor da mesma, dados os valores de p e q:

p	q	p ∧ q
F	F	F
F	V	F
V	F	F
V	V	V

A idéia de uma sentença funcional-veritativa seria então aquela onde se tem o seu valor-verdade simplesmente conhecendo o valor-verdade dos seus elementos (variáveis, constantes, etc.). Se a funcional-veritatividade é relaxada, aparecem sentenças onde não se pode afirmar nada sobre o seu valor-verdade, embora se conheça o valor-verdade dos seus elementos. Sendo assim pode-se divisar sentenças que são **necessariamente** verdadeiras ("Todos os casados não são solteiros", "hoje é sábado ou não é sábado") e as que são verdadeiras, simplesmente. Também vê-se a diferença entre uma determinada sentença ser simplesmente falsa ou necessariamente falsa. Somam-se às idéias de verdadeiro e falso as noções de **proposição necessária, proposição impossível, proposição contingente** e **proposição possível**. Estas noções, que já eram conhecidas na idade média [22], são chamadas de **noções modais**. Uma proposição necessária é aquela que **tem** que ser falsa; uma proposição possível é aquela que não é necessária nem impossível. Uma lógica que trabalhe com proposições que utilizam estas noções é chamada de lógica modal. É interessante notar que qualquer uma das quatro noções pode ser escolhida para representar as outras três. Normalmente, utiliza-se a noção de possibilidade

ou a de necessidade para a representação. Os símbolos utilizados para representá-las são "◇" e "□", respectivamente. Com estes operadores, ditos operadores modais, pode-se proceder à formalização da Lógica Modal. Se em termos sintáticos estes operadores bastam para a especificação do sistema, em termos semânticos introduz-se um modelo baseado numa tripla (W,R,V), onde W representa um conjunto de objetos (mundos), R é uma relação diádica reflexiva definida sobre W, e V é uma atribuição de valores dentro de certas regras. R fornece informações sobre a "acessibilidade" entre os "mundos possíveis" dados por W. Dependendo de como seja esta relação tem-se vários sistemas modais (T,S4,S5). Uma proposição será necessariamente verdadeira se ela é válida em todos os arranjos possíveis de valores-verdade para as variáveis (mundos acessíveis). Ela será possivelmente verdadeira se for verdadeira em, pelo menos, um mundo acessível, e assim por diante.

O conceito de mundo acessível pode assumir tantas variações quanto possível. Numa destas, pode-se imaginar que existe uma ordenação nos mundos, de maneira a se ter uma seqüência de ocorrência. Deste modo, um mundo só seria acessível pelos anteriores e acessaria todos os posteriores. Se se pensa nestes mundos como instantes de tempo, tem-se uma interpretação temporal da lógica modal. Esta interpretação dá origem à lógica temporal. O operador "◇" passa a ter a noção de "é possível que uma dada proposição seja verdadeira em algum instante de tempo" e o operador "□" indica que "a proposição será verdadeira deste instante em diante."*

A lógica temporal pode ser classificada de várias maneiras [22]. Se a lógica temporal é contínua, ou seja, existe um mapeamento dos instantes de tempo e a reta real, ela é derivada do sistema modal S.4.3. Caso contrário, tem-se uma lógica temporal discreta, onde faz sentido falar no instante de tempo seguinte. Este sistema é o S.4.3.1, ou como é mais conhecido, sistema D [21].

Com relação à semântica, a lógica temporal pode ser de intervalo ou pontual, de acordo com [23]. A semântica pontual pode ainda ser subdividida em semântica linear, semântica de desvio (Branching) e semântica de ordem parcial. A tabela a seguir fornece uma visão geral das características de cada semântica adotada.

---

* Na realidade, só um dos operadores é necessário, pois sempre se pode escrever um em função do outro.

SEMÂNTICA		CARACTERÍSTICA
INTERVALO		Baseia-se em trechos finitos do comportamento do sistema. As fórmulas descrevem o que ocorre durante o intervalo. Um intervalo pode ser dividido em dois outros (operador Chopp).
PONTUAL		As fórmulas são interpretadas em função de um ponto da trajetória, representando o comportamento do sistema naquele instante de tempo.
PODE SER	Linear	Só existe um futuro possível para cada instante as fórmulas são interpretadas sobre seqüências.
	Desvio	Cada instante pode ter múltiplas escolhas de futuro possível. As fórmulas são interpretadas sobre árvores semânticas.
	Ordem Parcial	As fórmulas são interpretadas sobre estruturas de estados (ou eventos) onde só se exige a obediência a duas relações: procedência e conflito A concorrência é bem caracterizada.

figura 6

A lógica a ser tratada neste trabalho tem as características de ser linear, portanto pontual, embora permita escolhas não-determinísticas [24]. Também é discreta, pois deriva da de Manna e Pnueli [25] que é discreta [26].

3.2. SISTEMA LÓGICO FORMAL - SINTAXE

A especificação de um sistema lógico formal é composta de:

1) lista de um conjunto primitivo de símbolos;
2) definição de quais são as fórmulas-bem-formadas (fbf) do sistema;
3) um subconjunto de fórmulas-bem-formadas chamadas de axiomas;
4) um conjunto de regras de inferências.

A próxima seção traz um sistema, incluindo símbolos e regras derivadas. Após isto, apresenta-se uma semântica para o sistema. O sistema apresentado é quase o mesmo que é apresentado em [23]. No capítulo 5 algumas inclusões serão feitas a este sistema com o intuito apresentar o formalismo proposto nesta tese.

ESPECIFICAÇÃO DO SISTEMA FORMAL

1) SÍMBOLOS

Os símbolos do sistema podem ser dos seguintes tipos:

A - Símbolos Lógicos:

igualdade: =

conectivos: $\neg$, $\rightarrow$

parenteses: (,)

símbolos variáveis globais: Letras maiúsculas do alfabeto latino de U a Z, com ou sem índices inferiores e/ou superiores.

símbolos variáveis locais: Letras minúsculas do alfabeto latino de u a z, com ou sem índices inferiores e/ou superiores.

símbolos operadores temporais: $\odot$, $\mathcal{U}$, $\Diamond$

B - Parâmetros:

Símbolo quantificador: $\exists$

Símbolos constantes globais: Letras maiúsculas do alfabeto latino de A até F, com ou sem índices inferiores e/ou superiores.

Letras de predicado: Letras maiúsculas do alfabeto latino de G até T, com ou sem índices inferiores e/ou superiores que representam relações n-árias para qualquer inteiro positivo n.

Letras de funções: Letras minúsculas do alfabeto latino de f até t, com ou sem índices inferiores ou superiores representando relações n-árias para qualquer inteiro positivo n.

A linguagem será uma linguagem multi-sorte, de acordo com Enderton [27]. Isto significa que se tem um conjunto não-vazio I, cujos membros são chamados sortes. Cada sorte tem seus próprios símbolos constantes globais, símbolos variáveis globais e locais, predicados e funções. Cada sorte

corresponde a um diferente conjunto, de maneira análoga ao exposto em [28]. Para cada h > 0 e cada (n+1)-upla de sortes $(i_1,i_2,...,i_{n+1})$, existe um conjunto enumerável de letras de funções n-árias ditas serem de sorte $(i_1,i_2,...,i_{n+1})$. Também, para cada n > 0 e cada (n+1)-upla de sortes $(i_1,i_2,...,i_{n+1})$, existe um conjunto enumerável de letras de predicados n-ários ditas serem de sorte $(i_1,i_2,...,i_{n+1})$.

2) FÓRMULAS

Termos - Para qualquer sorte i, os termos de sorte i da linguagem são definidos indutivamente como segue:

- Todos os símbolos constantes globais são termos.
- Todos os símbolos variáveis globais são termos.
- Todos os símbolos variáveis locais são termos.
- Se $t_1,t_2,...,t_n$ são termos e f é uma letra de função n-ária, então $f(t_1,t_2,...,t_n)$ é um termo.
- Se t é um termo $\odot t$ é um termo.
- Nenhuma outra seqüência é um termo.

As fórmulas-bem-formadas da linguagem são:

i) Se $t_1,t_2,...,t_n$ são termos de sorte $i_1,i_2,...,i_n$ respectivamente, e P é qualquer letra de predicado de sorte $i_1,i_2,...,i_n$, então $P(t_1,t_2,...,t_n)$ é uma fórmula-bem-formada.

ii) Se w é uma fórmula-bem-formada então $(\neg w)$, $(\odot w)$ e $(\Diamond w)$ são fórmulas-bem-formadas (fbf).

iii) Se w_1 e w_2 são fórmulas-bem-formadas, então $(w_1 \rightarrow w_2)$ e $(w_1 \mathcal{U} w_2)$ são fórmulas-bem-formadas.

iv) Se V é um símbolo variável global e w é uma fbf então $(\exists_i V\ (w))$ é uma fórmula-bem-formada.

v) Nenhuma outra seqüência é uma fbf.

3) AXIOMAS

Antes da apresentação dos axiomas serão definidas as noções de variáveis livres e substituição simultânea. Estas definições podem ser encontradas em [23]. Seja x qualquer símbolo variável (local ou global), C qualquer símbolo constante individual, e $t_1,t_2,...,t_n$ quaisquer termos.

- **Mapeamento VAR:** Define-se o mapeamento VAR da seguinte maneira:

  $VAR(C) \triangleq \emptyset$, $VAR(X) \triangleq \{x\}$

  $VAR(f(t_1,t_2,...,t_n)) \triangleq VAR(t_1) \cup VAR(t_2) \cup ... \cup VAR(t_n)$, onde VAR(t) é o conjunto composto de todas as variáveis que ocorrem em t.

- **Variáveis Livres:** Define-se o conjunto das variáveis livres em uma fórmula w (denotado por livre (w)) como segue:

  $Livre\ (t_1 = t_2) \triangleq VAR(t_1) \cup VAR(t_2)$

  $Livre\ (p(t_1,t_2,...,t_n) \triangleq VAR(t_1) \cup ... \cup VAR(t_n)$

  $Livre\ (\neg w_1) \triangleq Livre\ (w_1)$

  $Livre\ (w_1 \rightarrow w_2) \triangleq Livre\ (w_1) \cup Livre\ (w_2)$

  $Livre\ (\exists V : w_1) \triangleq Livre\ (w_1) - \{V\}$(onde "-" significa retirar do conjunto.

  $Livre\ (\odot w_1) \triangleq Livre\ (w_1)$

  $Livre\ (\Diamond w_1) \triangleq Livre\ (w_1)$

  $Livre\ (w_1 \mathcal{U} w_2) \triangleq Livre\ (w_1) \cup Livre\ (w_2)$

Dada uma fórmula w, e uma variável global X, X é dita ser livre se ela não ocorre em w na forma (∃X)v onde v é uma fórmula contida em w e X ocorre em v. Uma ocorrência de uma variável X numa fórmula w é ligada se ela está dentro de uma ocorrência, em w, de uma fórmula da forma (∃X)v.

- **Substituição simultânea** - Sejam $g_1,g_2,...,g_n$ termos tais que para todo i, a variável x_i pertence à mesma sorte que g_i. Então a nova fórmula $w\begin{bmatrix} x_1...x_n \\ g_1...g_n \end{bmatrix}$ pode ser obtida de w substituindo-se simultaneamente cada variável x_i pelo termo g_i. Ou seja, onde se encontra

x_i, coloca-se g_i. Este procedimento ocorre da seguinte maneira:

constante $$C\begin{bmatrix} x_1 \ldots x_n \\ g_1 \ldots g_n \end{bmatrix} \triangleq C$$

variável $$X\begin{bmatrix} x_1 \ldots x_n \\ g_1 \ldots g_n \end{bmatrix} \triangleq \begin{cases} x_i \text{ se } x \neq x_i \text{ para todo } i \\ g_i \text{ se } x = x_i \end{cases}$$

função $$f(t_1,t_2,\ldots,t_n)\begin{bmatrix} x_1 \ldots x_n \\ g_1 \ldots g_n \end{bmatrix} \triangleq f\left(t_1\begin{bmatrix} x_1 \ldots x_n \\ g_1 \ldots g_n \end{bmatrix} \ldots t_n\begin{bmatrix} x_1 \ldots x_n \\ g_1 \ldots g_n \end{bmatrix}\right)$$

igualdade $$(t_1 = t_2)\begin{bmatrix} x_1 \ldots x_n \\ g_1 \ldots g_n \end{bmatrix} \triangleq (t_1\begin{bmatrix} x_1 \ldots x_n \\ g_1 \ldots g_n \end{bmatrix} = t_2\begin{bmatrix} x_1 \ldots x_n \\ g_1 \ldots g_n \end{bmatrix})$$

predicados $$p(t_1,t_2,\ldots,t_n)\begin{bmatrix} x_1 \ldots x_n \\ g_1 \ldots g_n \end{bmatrix} \triangleq p\left(t_1\begin{bmatrix} x_1 \ldots x_n \\ g_1 \ldots g_n \end{bmatrix},\ldots,t_n\begin{bmatrix} x_1 \ldots x_n \\ g_1 \ldots g_n \end{bmatrix}\right)$$

negação $$(\neg w_1)\begin{bmatrix} x_1 \ldots x_n \\ g_1 \ldots g_n \end{bmatrix} \triangleq \neg\left(w_1\begin{bmatrix} x_1 \ldots x_n \\ g_1 \ldots g_n \end{bmatrix}\right)$$

implicação $$(w_1 \rightarrow w_2)\begin{bmatrix} x_1 \ldots x_n \\ g_1 \ldots g_n \end{bmatrix} \triangleq \left(w_1\begin{bmatrix} x_1 \ldots x_n \\ g_1 \ldots g_n \end{bmatrix} \rightarrow w_2\begin{bmatrix} x_1 \ldots x_n \\ g_1 \ldots g_n \end{bmatrix}\right)$$

próximo $$(\circ w_1)\begin{bmatrix} x_1 \ldots x_n \\ g_1 \ldots g_n \end{bmatrix} \triangleq \circ\left(w_1\begin{bmatrix} x_1 \ldots x_n \\ g_1 \ldots g_n \end{bmatrix}\right)$$

eventualmente $$(\Diamond w_1)\begin{bmatrix} x_1 \ldots x_n \\ g_1 \ldots g_n \end{bmatrix} \triangleq \Diamond\left(w_1\begin{bmatrix} x_1 \ldots x_n \\ g_1 \ldots g_n \end{bmatrix}\right)$$

até que $$(w_1, \mathcal{U} w_2)\begin{bmatrix} x_1 \ldots x_n \\ g_1 \ldots g_n \end{bmatrix} \triangleq \left(w_1\begin{bmatrix} x_1 \ldots x_n \\ g_1 \ldots g_n \end{bmatrix} \; \mathcal{U} \; w_2\begin{bmatrix} x_1 \ldots x_n \\ g_1 \ldots g_n \end{bmatrix}\right)$$

Existe $$(\exists V{:}w_1)\begin{bmatrix} x_1 \ldots x_n \\ g_1 \ldots g_n \end{bmatrix} \triangleq \begin{cases} (\exists V{:}w_1\begin{bmatrix} x_1 \ldots x_{i-1} x_{i+1} \ldots x_n \\ g_1 \ldots g_{i-1} g_{i+1} \ldots g_n \end{bmatrix} & \text{Se } x_i = V \\ & \text{para algum } i \\ (\exists V{:}w_1\begin{bmatrix} x_1 \ldots x_n \\ g_1 \ldots g_n \end{bmatrix} & \text{Se } x_i \neq V \text{ para todo } i \end{cases}$$

Axiomatização do Sistema

Seja w_1, uma fórmula-bem-formada nas formas abaixo. Então w e $\Box$w são axiomas:

A1) w, onde w é uma instância de uma tautologia.

A2) $\Box(w_1 \rightarrow w_2) \rightarrow (\Box w_1 \rightarrow \Box w_2)$

A3) $\Box w \rightarrow w$

A4) $\odot \neg w \leftrightarrow \neg \odot w$

A5) $\odot(w_1 \rightarrow w_2) \rightarrow (\odot w_1 \rightarrow \odot w_2)$

A6) $\Box w \rightarrow \odot w$

A7) $\Box w \rightarrow \odot \Box w$

A8) $\Box(w \rightarrow \odot w) \rightarrow (w \rightarrow \Box w)$

A9) $w \, , \, \mathcal{U} w_2 \leftrightarrow w_2 \vee (w_1 \wedge \odot(w_1 \, \mathcal{U} \, w_2))$

A10) $w_1 \, \mathcal{U} \, w_2 \rightarrow \Diamond w_2$

Antes da apresentação dos outros axiomas, é dada a definição de **substituibilidade.**

Substituibilidade: Seja x qualquer variável (local ou global). Então a relação dada por "um termo substitui globalmente x na fórmula w" (globsub(t,x,w)), é definida indutivamente sobre as fórmulas como segue:

1) Se w é uma fórmula atômica, então globsub(t,x,w) ocorre.
2) globsub(t,x, $\neg$w) ocorre se e somente se globsub(t,x,w) ocorre.
3) globsub(t,x,w $\rightarrow$ w_2) ocorre se e somente se globsub(t,x,w_1) e globsub(t,x,w_2) ocorrem.
4) Seja * qualquer operador temporal monádico. Então globsub(t,x,*w) ocorre se e somente se:
   - x $\notin$ livre(*w) ou
   - não existem ocorrências de variáveis locais em t e globsub(t,x,w) ocorre.
5) globsub(t,x,$w_1 \mathcal{U} w_2$) se e somente se:
   - x $\notin$ livre($w_1 \mathcal{U} w_2$), ou
   - não existem ocorrências de variáveis locais em t e globsub(t,x,w_1) e globsub(t,x,w_2) ocorrem.
6) globsub(t,x,$\exists$V:w) se e somente se:
   - x $\notin$ livre($\exists$V:w), ou
   - V $\notin$ var(t) e globsub (t,x,w) ocorre

Sendo assim, se globsub(t,x,w) ocorre, pode-se fazer a substituição global da variável x pelo termo t, ou seja, $w\begin{bmatrix} x \\ z \end{bmatrix}$ não implica no aparecimento em w de ocorrências de novas variáveis globais ligadas e de novas variáveis locais dentro do escopo dos operadores temporais.

A11) $(\exists V{:}w) \rightarrow w\left[\begin{smallmatrix} V \\ t \end{smallmatrix}\right]$ onde globsub(t,V,w) ocorre.

A12) $\odot(\exists V{:}w) \rightarrow (\exists V{:}\odot w)$

A13) $\odot f(t_1,...,t_n) = f(\odot t_1,...,t_n)$
(Para qualquer função f e termos $t_1,...,t_n$)

A14) $\odot p(t_1,...,t_n) \leftrightarrow p(\odot t_1,...,\odot t_n)$
(Para qualquer símbolo predicado P)

A15) Reflexividade da igualdade

$t = t$ para qualquer termo t

A16) Substituibilidade da igualdade

$t_1 = t_2 \rightarrow (\phi(t_1,t_1) \leftrightarrow \phi(t_1,t_2))$
Onde (i) ϕ é uma fórmula de estado (sem operadores temporais)
(ii) globsub $(t_2,t_1,\phi(t_1,t_2))$ ocorre.
$\phi(t_1,t_2)$ é a formula obtida de $\phi(t_1,t_1)$ substituindo t_1 por t_2 em zero ou mais lugares.

A17) Axioma do Frame

$V = \odot V$ onde V é qualquer variável global.

4) REGRAS DE INFERÊNCIA

O sistema em questão terá duas regras de inferência básicas, em oposição ao sistema encontrado em [23] que possui três. Tem-se então

R1) Modus Ponens - MP

$$\frac{w_1,\ w_1 \rightarrow w_2}{w_2}$$

R2) Inserção de $\forall$ - $\forall$I

$$\frac{w_1 \rightarrow w_2}{w_1 \rightarrow (\forall V{:}w_2)} \text{ onde } V \notin \text{livre}(w_1)$$

A notação $\dfrac{w_1, w_2, \ldots, w_n}{w}$ significa que se pode inferir w das fórmulas $w_1,...,w_n$. A diferença de [23] consiste no não-aparecimento da regra

"Inserção $\Box$" que diz que $\frac{w}{\Box w}$. Isto é explicado porque no sistema deste trabalho as fórmulas sem operadores temporais dizem respeito ao estado inicial. Maiores detalhes sobre estas diferenças serão dados no Cap. 4.

Os axiomas e regras aqui apresentados estão sumarizados no apêndice bem como outros axiomas, teoremas e regras derivadas do Cap.5.

3.3. SEMÂNTICA

A semântica de uma linguagem formal trata das interpretações dadas às seqüências de símbolos. Com o já mencionado anteriormente, a linguagem é uma linguagem multi-sorte.

As fórmulas da linguagem são avaliadas com respeito a uma estrutura S composta de uma Interpretação I e de uma Trajetória σ. Estas considerações estão de acordo com o sistema proposto por Thistle e Wonham em [28], em contraste com Ostroff [23] que apresenta uma tripla I = (S,A,σ) composta de uma Estrutura (S), uma Atribuição (A) e uma Trajetória (σ). Neste desenvolvimento utiliza-se a noção de estrutura S = (I,σ), onde I é a interpretação e σ a trajetória. Comparando os formalismos, vê-se que aqui a interpretação I faz as funções da estrutura S, em [23]. A ausência da atribuição A na presente elaboração deve-se ao fato de que como se definiu como símbolo quantificador primitivo o $\exists$, e não o $\forall$, não se faz necessário este elemento na definição semântica dele, como será visto mais adiante.

Uma interpretação I, será dada por um mapeamento cujo domínio é formado pelos parâmetros. Sendo assim tem-se:

1- I designa um domínio D_j para cada símbolo quantificador $\exists_j$ de sorte j.
2- I designa cada símbolo constante de cada sorte j um elemento em D_j.
3- I designa para cada símbolo variável global de cada sorte j um valor em D_j.
4- I designa para cada símbolo predicado de sorte $(j_1,j_2,...,j_n)$ um predicado $\mathcal{P} \subset D_{j1} \times D_{j2} \times D_{jn-1} \rightarrow D_{jn}$.

Antes de definir a trajetória σ faz-se necessário definir o conceito

de estado. Um estado s é um mapeamento que a cada variável local v_j, de cada sorte j, designa um valor $s(v_j)$ em D_j. Uma trajetória σ é constituída de uma sequência finita ou infinita de estados. Para uma trajetória qualquer $\sigma = s_0 s_1 \ldots$, chama-se de sufixo-K da trajetória σ para qualquer K (σ^K) à trajetória $s_K s_{K+1} \ldots$.

Até o presente, todos os símbolos constantes, funções, predicados foram apresentados sem nenhuma interpretação. A abordagem semântica possibilita que isto seja feito. Sendo assim, define-se:

1. S[t] como o valor atribuído ao termo t por S
2. S[c] como o valor atribuído ao símbolo constante C por S
3. S[v] como o valor atribuído ao símbolo variável local v sob o estado inicial da trajetória σ de S. É importante observar que qualquer fórmula sem operadores temporais sempre faz referência ao estado inicial.
4. S[f] como a função atribuída ao símbolo f por S.
   Para qualquer símbolo de função n-ário f e quaisquer termos $t_1,\ldots,t_n$ $S[f(t_1,\ldots,t_n)] = S[f](S[t_1], S[t_2],\ldots,S[t_n])$.
5. S[$\mathcal{P}$] como o predicado atribuído ao símbolo $\mathcal{P}$ por S. O predicado $\mathcal{P}$ seleciona um conjunto de n-uplas ordenadas dentro do domínio em questão.
6. Para qualquer termo da forma ($\odot$t), onde t é um termo
   $S[(\odot t)] = S^{(1)}[t] = I\sigma^1[t]$
   Ou seja, $\odot$t corresponde intuitivamente ao valor de t no próximo instante de tempo.

Diz-se de uma fbf w que ela é satisfeita por uma estrutura S (representado por $\models^S w$), para qualquer estrutura S, se w preenche um dos seguintes requisitos:

i) $\models^S (t_1 = t_2)$ se e somente se $S[t_1] = S[t_2]$ (t_1 e t_2 da mesma sorte).

ii) $\models^S p(t_1,t_2,\ldots,t_n)$ se e somente se $(S[t_1],S[t_2],\ldots,S[t_n]) \in S[p]$, onde $\mathcal{P}$ é uma letra de predicado e $t_1,t_2,\ldots,t_n$ são termos.

iii) $\models^S (\neg w)$ se e somente se não é o caso de $\models^S w$.

iv) $(w_1 \rightarrow w_2)$ se e somente se ou $\models^S w_2$ ou não é o caso de $\models^S w_1$.

v) $\models^S (\exists V{:}w)$ se e somente se existe pelo menos uma estrutura $S'=(I',\sigma)$ tal que $\models^{S'} w$, onde I' é uma interpretação idêntica a

I exceto pela atribuição que faz à variável global V, em questão.

vi) $\models^{s}(\odot w)$ se e somente se $\models^{s(1)}w$.

vii) $\models^{s}(\Diamond w)$ se e somente se existe um $K \geq 0$ tal que $S^{(K)} \models w$

viii) $\models^{s}(w_1 \mathcal{U} w_2)$ se e somente se para algum $K \geq 0$, $\models^{s(K)} w_2$ e para todo i, $0 \leq i < K$, $\models^{s(i)} w_1$.

Intuitivamente, os operadores temporais têm as seguintes paráfrases:

$\odot w$ - w será verdadeiro no próximo instante.

$\Diamond w$ - w será eventualmente verdadeiro num estado futuro.

$w_1 \mathcal{U} w_2$ - w_2 será verdadeiro em algum estado futuro, e w_1 será verdade (pelo menos) até lá.

Algumas abreviações são feitas abaixo, no intuito de simplificar o trabalho de manipulação de fórmulas.

i) $w_1 \vee w_2$ abrevia $((\neg w_1) \rightarrow w_2)$ (Disjunção)

ii) $w_1 \wedge w_2$ abrevia $\neg(w_1 \rightarrow (\neg w_2)$ (Conjunção)

iii) $(w_1 \leftrightarrow w_2)$ abrevia $(w_1 \rightarrow w_2) \wedge (w_2 \rightarrow w_1)$ (Bicondicional)

iv) $(\Box w)$ abrevia $(\neg(\Diamond(\neg w))$ (daqui por diante)

v) $w_1 \cup w_2$ abrevia $(\Box w_1) \vee (w_1 \mathcal{U} w_2))$ (A menos que)

vi) $w_1 \mathcal{P} w_2$ abrevia $(\neg((\neg w_1) \mathcal{U} w_2))$ (precede)

vii) $(\forall V{:}w)$ abrevia $(\neg(\exists V{:}(\neg w)))$ (quantificador universal)

viii) $(t_1 \neq t_2)$ abrevia $(\neg(t_1 = t_2))$ (diferente ou não igual)

Sejam Φ e Φ_1 símbolos que denotem quaisquer conjuntos de fórmulas, e seja R uma classe de estruturas qualquer. Então tem-se:

$S \models \Phi$ se e somente se para toda fórmula $w \in \Phi$, $S \models w$. Diz-se que S é um modelo para Φ.

$R \models \Phi$ se e somente se para toda estrutura $S \in R$, $S \models \Phi$. Diz-se que Φ é R-válida. No caso onde R é a classe de todas as estruturas possíveis, Φ é dita ser válida ($\models \Phi$).

$\Phi_1 \models^{R} \Phi$ se e somente se $\models^{R} \Phi_1 \Rightarrow \models^{R} \Phi$. Diz-se que Φ é uma R-conseqüência de Φ_1. No caso onde R é a classe de todas as estruturas possíveis, Φ é dita ser uma conseqüência de Φ_1 ($\Phi_1 \models \Phi$).

As noções básicas fornecidas aqui na composição do sistema lógico formal (sintaxe e semântica) foram adaptadas de [23]. Este sistema serve como base para inovações a serem introduzidas nos próximo capítulos. É importante lembrar que no apêndice é apresentada uma lista de axiomas e regras de inferência condensando todos os resultados obtidos.

CAPÍTULO 4

LÓGICA TEMPORAL APLICADA À MODELAGEM DE DEDS

4.1. INTRODUÇÃO

A lógica temporal é um formalismo que tem tido uma larga aplicação em engenharia e ciência da computação. Na área computacional, a lógica temporal tem-se prestado à verificação de software e hardware e síntese de programas concorrentes. Nestes casos, a lógica temporal descreve o comportamento dos programas garantindo que certas propriedades ocorrerão e outras não. Garantir que "coisas boas" ocorrem é conhecido como garantir a "existência" ou "liveness". Garantir que "coisas ruins" não ocorram é visto como uma questão de segurança, ou "safety". Em geral estas são as duas noções utilizadas na implementação de programas concorrentes e na verificação de software e hardware [23], [29], [30], onde é necessária uma argumentação formal sobre as propriedades dos sistemas em questão. Um trabalho fundamental nesta área foi desenvolvido por Manna e Pnueli [25]. Nele, os autores desenvolvem os conceitos usados e citados pela maioria dos autores na área.

Para aplicações em controle, o uso da lógica temporal é relativamente recente. Para este fim algumas adaptações foram feitas na sintaxe e semântica da lógica temporal, tentando compatibilizá-la com os sistemas encontrados em controle, mais detidamente em controle de sistemas dinâmicos a eventos discretos. Neste contexto, uma estratégia de controle que restrinja os estados possíveis de serem alcançados pelo sistema, funciona como uma restrição de segurança das aplicações em computação. Garantir que a planta cumprirá uma certa restrição, seja o cumprimento de um limitante de tempo ou atingir um dado estado, pode ser visto como uma assertiva de existência ("liveness"). Sendo assim, pode-se aplicar toda a teoria desenvolvida para a lógica temporal, com algumas alterações, na verificação e síntese de controladores para sistemas dinâmicos a eventos discretos. Esta passagem exige a criação de novos elementos e a supressão de outros para adaptar o formalismo aos tipos de sistemas tratados. Dependendo de como se encara a modelagem dos DEDSs, pode-se ter enfoques diferentes para o papel da lógica temporal. Neste capítulo, uma breve apresentação sobre os trabalhos nesta linha é feita. Os modelos utilizando lógica temporal podem ser primariamente classificados em duas categorias: a abordagem **primal** (single language) e a abordagem **dual** (dual language). A abordagem **primal** advoga a utilização da lógica temporal na descrição da planta, na especificação de malha-fechada e na síntese e descrição do controlador. A abordagem **dual**, por sua vez, insiste em só utilizar a lógica temporal nas especificações de malha-fechada, reservando um

outro formalismo para a descrição da planta e do controlador. Os dois formalismos combinados completam o trabalho, daí o termo **dual**. A importância deste capítulo está no fato de que ele fornece os conceitos básicos para o entendimento da lógica temporal de tempo real generalizada (GRTTL) proposta no capítulo 5.

4.2. ABORDAGEM DO GRÁFICO - W

Um dos primeiros trabalhos que utilizou a lógica temporal no controle de sistemas dinâmicos a eventos discretos (DEDS) foi o trabalho de Fusaoka et al., em 1983. Neste trabalho a lógica temporal é descrita segundo Manna e Pnueli [23] e a síntese de controladores é feita através de uma estrutura chamada de gráfico-w. O gráfico tem uma função semelhante ao método do "tableau", que é citado em [29]. A idéia básica é a seguinte: divide-se toda a fórmula temporal em duas partes, uma que fornece a parte que acontece agora e outra que faz referência aos passos futuros. Sendo assim, a divisão é feita em termos de presente e futuro. Repetindo esta divisão, exaure-se todas as possibilidades desde que o número de "situações" das fórmulas seja finito.

Formalmente falando, um gráfico-w é criado a partir de um conjunto de nós e de um conjunto de arcos. Cada nó é uma "instância" da fórmula em questão e cada arco a alteração desta instância. A partir destas noções desenvolve-se todo um processo de argumentação para a síntese de controladores. Esta síntese consiste em comparar o gráfico-w obtido do conjunto de fórmulas com o exigido pelas especificações de malha-fechada e completar o que falta no próprio gráfico. Isto indica quais fórmulas precisam ser incluídas para se ter o controle. É importante notar que este procedimento é heurístico e só funciona para casos onde o número de possibilidades (estados) é finito, e mais ainda, reduzido. Aqui utiliza-se claramente o conceito **primal**, pois uma mesma linguagem é usada para a modelagem da planta e do controlador, e ainda para as especificações em malha-fechada. O gráfico-w é só um procedimento tabular utilizado para a argumentação sobre as expressões em lógica temporal.

4.3. ABORDAGEM DE THISTLE-WONHAM

Um trabalho posterior ao de Fusaoka bem relevante para o presente desenvolvimento é o de Thistle e Wonham, em 1986 [28]. A axiomatização da lógica temporal segue de perto a proposta de Manna e Pnueli [23]. Uma diferença, porém, é o fato de que uma certa regra de inferência utilizada em Manna e Pnueli não é usada no sistema proposto por Thistle e Wonham. Para proceder à síntese do controlador, tem-se os seguintes procedimentos: Modela-se a planta através de fórmulas temporais; acrescenta-se equações representando o controlador e, finalmente, prova-se que as especificações de malha-fechada são obtidas manipulando-se formalmente o conjunto de equações da planta mais o controlador.

Este formalismo será mais detalhado aqui, por razões que ficarão claras adiante. Serão apresentados somente as diferenças em relação ao sistema proposto no Capítulo 3. Para um aprofundamento no sistema desta seção recomenda-se a referência [28].

Símbolos - Como é uma lógica temporal proposicional, fica eliminado o quantificador existencial e o universal.

Axiomas - Retira-se todos os que fazem alusão aos quantificadores.

Entre os axiomas são incluídos os seguintes:

A18. Se w é uma fórmula que é verdadeira sob uma determinada avaliação, então w e $\Box$w são axiomas.

Este esquema de axioma é chamado de axioma de domínio. Com ele se pode introduzir axiomas diversos dependendo do sistema que se está tratando.

Ao se trabalhar com controle, é importante desenvolver a noção de um conjunto de eventos. Sendo assim, uma sorte da linguagem é separada para ser a sorte de eventos. Os símbolos constantes da sorte são os próprios eventos e é definida uma variável local δ que assume o valor do evento que acontece naquele momento. Para casos onde exista intertravamento ou o sistema pare de transicionar, um símbolo especial ϕ é definido, simbolizando o evento nulo, quando este evento ocorre todas as variáveis mantém o mesmo valor, ou seja:

A19. $\Box[\delta = \phi \Rightarrow (0x) = x)]$ Onde x é uma variável local de uma sorte qualquer.

A regra de inferência

$$\frac{v}{\Box v}$$

existe no sistema de Manna e Pnueli. Ela significa que se uma determinada fórmula v é dedutível, então $\Box v$ também o será. Neste caso, a dedutibilidade de v está ligada à uma dada estrutura S. Se a estrutura S se altera, a fórmula v pode não mais ser dedutível e, consequentemente, não se pode mais deduzir $\Box v$. É importante fazer uma distinção entre esta regra e o conceito de validade. A regra $v \vdash \Box v$ $\left(\text{ou } \frac{v}{\Box v}\right)$ é um conceito sintático e aponta a dedutibilidade de $\Box v$ partindo de v. No lado semântico, $v \models \Box v$ significa que se a fórmula v é válida (verdadeira sob qualquer interpretação) então $\Box v$ também o será, pois neste caso w é satisfeita por todas as estruturas. O sistema Thistle-Wonham elimina a regra sintática $v \vdash \Box v$. As únicas fórmulas onde tanto v quanto $\Box v$ são dedutíveis são aquelas na forma dos esquemas de axiomas apresentados nesta seção e anteriormente no capítulo 3.

A diferença principal está no que se pode deduzir a partir de uma hipótese. Isto tem um efeito significativo no momento de associar os conceitos da lógica temporal às estruturas de controle. Para o sistema Thistle-Wonham, qualquer fórmula que não contenha operadores temporais refere-se única e exclusivamente ao estado inicial do sistema descrito.

Uma consequência advinda da não inclusão desta regra no corpo de regras de inferência manifesta-se na completude forte do sistema. Um sistema é correto se tudo o que é demonstrado (ou dedutível) é válido e é completo se tudo o que é válido pode ser demonstrado. Note que os conceitos de dedutibilidade e validade são conceitos sintáticos e semânticos, respectivamente. Um sistema que apresenta a propriedade da completude forte garante que se pode deduzir de uma fórmula w todas as suas consequências lógicas. O exemplo citado em Thistle-Wonham [28] esclarece bem este fato: "Assuma que a fórmula v não contém nenhum operador temporal; então, a fórmula $\Box v$ é satisfeita por qualquer estrutura que satisfaz o conjunto de fórmulas {v, (⊙v), (⊙(⊙v)), ...}. Não se pode deduzir $\Box v$ deste conjunto". Para maiores considerações sobre esta regra, recomenda-se a referência [205], e

para a justificação da exclusão dela por parte do sistema Thistle-Wonham, recomenda-se a referência [28].

Estes são os axiomas incluídos no sistema do capítulo 3. É interessante notar que a única regra de inferência permitida é o modus ponens. Este sistema é claramente baseado nos conceitos da abordagem primal.

O sistema Thistle-Wonham possibilita tratar casos onde o conjunto de estados do controlador é muito grande ou até infinito, desde que o número de esquemas de fórmulas seja finito. Em contrapartida, não se tem resultados concretos em termos de decidibilidade, ou seja, dado uma planta e uma especificação de malha-fechada, não sempre possível sintetizar um controlador seguindo um dado procedimento, num número finito de passos.

4.4. SISTEMA DE KNIGHT-PASSINO

Numa abordagem mais recente, Knight e Passino [31] apresentaram um sistema baseado em lógica temporal que permite considerações de decidibilidade. Isto quer dizer que se pode num número finito de passos, decidir se uma dada estratégia de controle satisfaz os requisitos de malha-fechada ou não. Este sistema usa a abordagem da linguagem dual, ou seja, usa-se um formalismo para a descrição da planta e do controlador (Máquinas de Estados Finitos) e um outro (lógica temporal) para as especificações em malha-fechada.

Nesta seção pretende-se esboçar rapidamente o sistema em questão. Para um aprofundamento sugere-se a referência [31].

Para a modelagem e controle da planta usam-se máquinas de estados finitos. A planta é representada como sendo a quádrupla $P = (X,Q,\delta,X_0)$ onde δ é uma relação δ: $Q \times X \rightarrow \mathcal{P}(X) - \{\phi\}$, onde o mapeamento feito por δ é não determinístico, pois $\mathcal{P}(X)$ equivale ao conjunto de todos os subconjuntos de X. X e Q representam o conjunto de estados da planta e do regulador (variante do termo controlador), respectivamente, e são finitos. $X_0 \in X$ denota o conjunto de estados iniciais possíveis. O regulador é representado pela quádrupla $R=(Q,X,\xi,q_0)$ onde ξ é a função de transição do regulador, $\xi: XxQ \longrightarrow Q$, $q_0 \in Q$ denota o estado inicial do regulador e Q e X são como já descritos anteriormente. O arranjo entre a planta e o regulador pode ser visualizado no diagrama de blocos que segue:

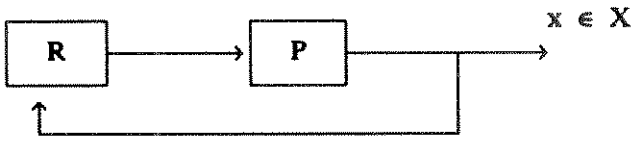


figura 7

O interesse desta modelagem está na análise das seqüências de estados da planta. Se uma dada seqüência é permitida pelo regulador R, então ela é uma seqüência R-permissível. Para cada regulador R tem-se um conjunto diferente de seqüências permissíveis. Dada a planta P, o regulador será obtido em função das condições desejadas de operação. Estas condições serão expressas por equações em lógica temporal. A seguir apresenta-se considerações sobre a lógica temporal utilizada no contexto deste trabalho.

A lógica temporal utilizada em conjunto com as máquinas de estados finitos é uma lógica temporal proposicional e é muito semelhante à apresentada em [29], exceto pelo fato de não ser incluído o operador "until", como é comum em outras apresentações. Uma outra diferença reside no fato de que Manna e Wolper obtém resultados sobre decidibilidade e síntese usando o método do tableau semântico. O presente trabalho utiliza uma abordagem baseada no trabalho de Buchi [32]. Enquanto na abordagem de Thistle-Wonham, o número de estados do controlador pode ser infinito, aqui eles são limitados. Com isto, perde-se um pouco na generalidade da abordagem, mas ganha-se em resultados sobre a decidibilidade de uma expressão. Desta forma, pode-se sempre verificar se um dado regulador satisfaz ou não as especificações em lógica temporal ou utilizar as especificações para a síntese de um regulador. Fica claramente identificado o caráter dual desta abordagem.

A seguir são apresentados vários resultados que culminam com dois teoremas garantindo a decidibilidade de especificação e a síntese de reguladores. Como uma investigação acurada destes resultados está fora do escopo deste trabalho, só serão mencionados os ítens mais importantes, seguidos de uma discussão intuitiva. Para os interessados, uma boa leitura é a referência [29], que foi usada como fonte para o resumo que se segue.

EXECUÇÃO

Suponha que o sistema a ser analisado consista de uma Planta P e de um regulador R. Seja w o conjunto dos números naturais. Uma execução (Run) é uma seqüência infinita de pares $(q_i,x_i)_{i\in w}$, onde q_i e x_i são os estados do

regulador R e da planta P no passo i da execução (cada mudança de estado determina um passo), respectivamente. q_0 é o estado inicial do regulador e $x_0 \in X_0$ é um dos possíveis estados iniciais da planta. Para cada i, $q_{i+1} = \xi(x_i q_i)$ e $x_{i+1} \in \delta(\xi(x_i,q_i)x_i)$. A seqüência de saída do sistema é $(x_i)_{i\in w}$. O conhecimento de ξ e q_0, bem como da seqüência de saída, permite recuperar a execução.

Para a execução, existe uma pequena diferença, ou assimetria, na contagem dos estados da planta e do controlador. Quando o estado x_i é enviado para o regulador, a resposta dele será q_{i+1} que por sua vez suscitará o estado x_{i+1} na planta, e assim por diante. Sendo assim, x_{i+1} é gerado a partir de (q_i,x_i), e não de (q_{i+1},x_i).

POSTO (RANK)

Uma classificação útil para as fórmulas temporais pode ser feita a partir da definição do posto (rank) de uma fórmula. Seja $\varphi\in B$ uma fórmula bem-formada do nosso sistema, define-se $v(\varphi)$: $B \to w$ como o posto desta fórmula, onde w é o trainentos onde:

a) $v(\varphi) = 0$, se φ consiste de uma única variável proposicional.
b) $v(\sim\varphi) = v(\varphi)$;
c) $v((\varphi \wedge \psi)) = v((\varphi \vee \psi)) = v((\varphi \to \psi)) = v((\varphi \leftrightarrow)) = \sup\{v(\varphi),v(\psi)\}$ onde $\sup\{.,.\}$ é um operador que dá o maior valor dentre os operandos;
d) $v(0\varphi) = v(\Box\varphi) = r(\Diamond\varphi) = v(\varphi) + 1$.

Intuitivamente, o posto de uma fórmula φ fornece o número de operadores temporais na mesma. Isto pode ser associado com o grau de complexidade da fórmula em relação às fórmulas proposicionais com operadores não temporais.

O conjunto de todas as fórmulas temporais em questão, B, pode ser dividido em subconjuntos de todas as fórmulas de posto i.

Seqüência R-Permissível

Para qualquer conjunto Z, seja Z^w o conjunto de todas as seqüências infinitas de elementos de Z. Se $z \in Z^w$, z_i é o i-ésimo termo na seqüência z. Z^I representa uma seqüência idêntica a z onde são retiradas os primeiros i termos.

Seja $P = (X,Q,\delta,X_0)$ uma planta fixa; seja R um regulador, com estado inicial q_0 e função de transição ξ. Se $\alpha \in X^w$, α é dita ser R-permissível se ela é a seqüência de saída derivada de uma execução do sistema regulador que consiste de r e P.

Satisfação

Seja α uma seqüência R-permissível e seja $((\sigma_i,\alpha_i))_{i\in w}$ a execução correspondente. $q^{R,\alpha,i}$ denota o estado σ_i alcançado por R depois e i passos na execução de que α é derivada. $R^{\alpha,i}$ denota o regulador cuja função de transição é a mesma de R. Após estas definições preliminares pode-se definir a noção de satisfação. Seja P uma planta fixa e sejam fixos os conjuntos Q e X. Dado um regulador R para P e uma seqüência R-permissível α, diz-se que uma fórmula φ é satisfeita pelo par (R,α), $(R,\alpha) \models \varphi$, se as condições abaixo são satisfeitas:

i) $(R,\alpha) \models p$ se $p\in Q$ e $p = q_0$ ou se $p\in X$ e $p=\alpha_0$,

ii) $(R,\alpha) \models \neg\varphi$ se não ocorre $(R,\alpha) \models \varphi$,

iii) $(R,\alpha) \models (\varphi \wedge \psi)$ se $(R,\alpha) \models \varphi$ e $(R,\alpha) \models \psi$,

iv) $(R,\alpha) \models (\varphi \vee \psi)$ se $(R,\alpha) \models \varphi$ ou $(R,\alpha) \models \psi$,

v) $(R,\alpha) \models (\varphi \rightarrow \psi)$ se não ocorre $(R,\alpha) \models \varphi$ e $(R,\alpha) \models \neg\psi$,

vi) $(R,\alpha) \models (\varphi \Leftrightarrow \psi)$ se $(R,\alpha) \models (\varphi \rightarrow \psi)$ e $(R,\alpha) \models (\psi \rightarrow y)$,

vii) $(R,\alpha) \models \circ\varphi$ se $(R^{\alpha,1},\alpha^1) \models \varphi$,

viii) $(R,\alpha) \models \Box\varphi$ se para $i \geq 0$, $(R^{\alpha,i},\alpha^i) \models \varphi$,

ix) $(T,\alpha) \models \Diamond\varphi$ se para algum $i \geq (R^{\alpha,i},\alpha^i) \models \varphi$,

Relação de Equivalência

Um outro conceito útil para os resultados desta abordagem é o de relação de equivalência. Uma relação de equivalência é uma relação entre dois elementos (neste caso, de Z) que é reflexiva, simétrica e transitiva. Dado um conjunto Z, uma relação diádica de equivalência N e um elemento $z \in Z$, uma N-classe de equivalência é o conjunto $\{y \in Z : y \sim Z\}$. Define-se uma família de relações de equivalência $\sim_{ni}R$ sobre o conjunto de sequências R-permissíveis, tal que uma equivalência $\sim_{n,R}$ implica em satisfação, por parte das sequências, das mesmas fórmulas de posto no máximo $\sim$. Sejam α e β duas sequências R-permissíveis. Então

i) $\alpha \sim_{0,R} \beta$ se $\alpha_0 = \beta_0$

ii) $\alpha \sim_{n+1,R} \beta$ se a) $\alpha \sim_{n,R} \beta$

b) $q^{R,\alpha,1} = q^{R,\beta,1}$ e $\alpha^1 \sim_{n,R} {}^{\alpha,1}\beta^1$

c) $\forall i$ existe um j (e vice-versa) tal que $q^{R,\alpha,i} = q^{R,\beta,j}$ e

$\alpha^i \sim_{n,R} {}^{\alpha,i}\beta^j$

Para se compreender bem a relação $\sim_{n,R}$ é importante lembrar que $\alpha \sim_{n,R} \beta$ significa dizer que α e β satisfazem o mesmo conjunto de fórmulas, de posto até n. Sendo assim, a condição para que $\alpha \sim_{0,R} \beta$ ocorra, é que o primeiro termo de α e β, α_0 e β_0, respectivamente, sejam iguais. Para $\sim_{n+1,R}$, pode-se argumentar por indução finita. Note que o item a) apoia-se na indução, enquanto b) e c) garantem a satisfação das fórmulas de posto n+1 para os casos onde aparecem os conectivos $\circ$, $\Diamond$ e $\Box$, respectivamente. Vale salientar que se $n > m$ então a classe $\sim_{m,R}$ contém a classe $\sim_{n,R}$, pois para todo elemento α tal que $\alpha \sim_{n,R} \beta$, $\alpha \sim_{m,R} \beta$.

Lema 1 - "Seja φ uma fórmula tal que $r(\varphi) \leq n$ e seja α e β sequências R-permissíveis tais que $\alpha \sim_{n,R} \beta$. Então $(R,\alpha) \models \varphi$ se e somente se $(R,\beta) \models \varphi$."

Este resultado garante que provar a satisfação de uma fórmula, ou conjunto de fórmulas, em relação a uma sequência α equivale a provar isto para outra sequência β, desde que $\alpha \sim_{n,R} \beta$. Deste modo, ao se deparar com dificuldades na prova de que $(R,\alpha) \models \varphi$, podemos recorrer a uma outra sequência β, onde $\alpha \sim_{n,R} \beta$, e proceder a prova mais facilmente.

Lema 2 - "Dada uma fórmula φ, um regulador R e um par de sequênci finitas (v,τ) determinando uma sequência $\alpha \in X^w$, pode-se decidir efetivamente se $(R,\alpha) \models \varphi$ ou não.

Este lema fornece a preciosa informação de que sempre se pode chegar a uma conclusão sobre a condição de satisfação de uma fórmula φ pelo par (R,α), desde que as sequências determinantes de α, (v,τ), sejam finitas.

Uma sequência α e X^w, é dita ser ultimamente periódica, com período n_2 e início de periodicidade n_1, se α tem a forma $v\hat{}\tau\hat{}\tau\hat{}\tau\hat{}\tau...$, onde v e τ são sequências finitas de comprimento n1 e n2,respectivamente, e o símbolo "^" significa a concatenação das sequências à esquerda e à direita dele (Ex.: a^b é a concatenção de a e b). Diz-se que α é determinada pelo par (v,τ). Fica

patente a analogia entre v e um transiente, enquanto que τ se identifica com o regime permanente.

Lema 3 - "Para qualquer regulador R, qualquer $n \in w$ e uma sequência R-permissível α qualquer, existe uma sequência periódica β tal que $\alpha \sim_{n,R} \beta$. Ainda mais, dados n, $|X|$ e $|Q|$ onde $|X|$ e $|Q|$ representam a cardinalidade do conjunto X e Q, respectivamente, pode-se calcular cotas para o início (onset) e o período.

Este lema estabelece que a qualquer sequência α R-permissível sempre se pode associar uma outra β que é finalmente periódica. Se o valor n for dado, bem como $|Q|$ e $|X|$, é possível até se fornecer os valores de n_1 e n_2. A importância deste lema é facilmente vista se se comparam os três lemas conjuntamente.

Suponha que se tem uma fórmula φ, um regulador R e uma sequência $\alpha \in Z^w$. Se deseja-se verificar que $(R,\alpha) \models \varphi$, ou não, pode-se seguir o seguinte procedimento:

1) Encontrar uma sequência β ultimamente periódica tal que $\alpha \sim_{n,R} \beta$ (Lema 3).
2) Sendo β ultimamente periódica, verifica-se se $(R,\beta) \models \varphi$ (Lema 2).
3) Como $\alpha \sim_{n,R} \beta$ então $(R,\alpha) \models \varphi$ se e somente se $(R,\beta) \models \varphi$ (Lema 1).

Se o desejado era provar que o Regulador R satisfaz as condições impostas pela fórmula φ, tudo se passa como se fosse necessário provar φ para todas as sequências R-permissíveis.

Em termos de síntese, o procedimento é uma busca exaustiva. Dada uma fórmula φ (especificação em malha-fechada) e a Planta $P = (X,Q,\delta,X_0)$. Sabendo que o universo dos reguladores R é finito, pode-se tentar todas as combinações possíveis variando-se a função de transição e o estado inicial.

Após este breve resumo do sistema proposto por Knight-Passino, cabe observar que é utilizada a abordagem dual para verificação e síntese de controladores. Outras referências são: [32] que traz as noções utilizadas para a prova dos resultados e [33] onde se trata os mesmos resultados usando a lógica de desvio.

4.5. ABORDAGEM OSTROFF-WONHAM

A abordagem Ostroff-Wonham é um outro exemplo onde a linguagem dual é utilizada. Neste caso, a lógica temporal de tempo real é utilizada em parceria com as "Máquinas de estado estendidas "ESM". A lógica temporal de tempo real (RTTL) é um formalismo muito semelhante à lógica temporal apresentada no capítulo 3, acrescentando-se considerações de tempo real, ou seja, limites inferiores e superiores para o tempo entre dois eventos. Uma outra diferença é a inclusão da regra de inferência inserção-$\Box$ dada por: $\frac{v}{\Box v}$ onde v é uma fórmula bem-formada.

A modelagem de DEDS via este formalismo é feita por etapas. Para verificação de um conjunto planta-controlador em relação a um conjunto de fórmulas em RTTL (especificações em malha-fechada) usa-se um sistema de prova chamado de PS-RTTL. Com este sistema de prova pode-se verificar se o conjunto planta-controlador (a ESM representando o controlador) satisfaz os requisitos apresentados pelas fórmulas, para alguns casos, automaticamente. A síntese de um controlador também pode ser obtida através deste procedimento, em casos especiais. Na maioria dos casos, procede-se tanto a verificação como a síntese de uma maneira não automática.

Além da ESM, Ostroff elaborou um procedimento de conversão de ESM para instruções numa linguagem chamada CONIC. Com isto, pode-se ter o seguinte ciclo para o projeto de um controlador:

1) A planta é transformada numa ESM, se estava em CONIC;
2) Utilizando as fórmulas em RTTL e a ESM da planta, encontra-se uma ESM para o controlador;
3) Finalmente, converte-se a ESM do controlador para o CONIC, e tem-se um programa, que é o controlador, para ser usado na aplicação necessária

A análise exaustiva do sistema Ostroff-Wonham está fora do escopo deste trabalho, portanto ele é visto rapidamente nas linhas que se seguem. O ESM é analisado com um pouco de detalhe. Para uma abordagem mais profunda deste formalismo, recomenda-se a leitura das referências [23,34,35].

ESM

Um dos primeiros formalismos que surgiu para se analisar sistemas onde o estado do sistema muda discretamente com a ocorrência de eventos foi o baseado em máquinas de estado. Embora muito utilizado, este formalismo apresenta as seguintes dificuldades: não permite a hierarquização de grupos de estados e dificuldade na representação de estados que ocorrem eventualmente. Uma tentativa de se contornar alguns destes obstáculos resultou na criação das Máquinas de Estados Estendidos (Extended State Machine - ESM).

A ESM inclui em seu bojo elementos do CSP, Processos Sequenciais de Comunicações, tais como: ações de comunicação e ações compartilhadas. As ações de comunicação referem-se ao envio e recebimento de mensagens via canais de comunicação. Com o termo ação compartilhada deseja-se expressar o sincronismo na ocorrência de certos eventos em sistemas diferentes, mas com o mesmo rótulo. Embora possa não haver comunicação através de canais entre os processos em questão, os eventos ocorrem simultaneamente e podem ter uma relação causa-efeito bastante explorada para sincronização. Uma outra característica do ESM é que ele permite a adoção de cotas inferiores e superiores de tempo entre dois eventos e possibilita a existência de um processo "relógio" que auxilia na temporização geral. Como outra característica marcante, tem-se a introdução de variáveis de monitoração (activity) e de dados. As variáveis de monitoração fornecem informações necessárias ao controle do sistema pelo controlador. As variáveis de dados representam informações numéricas. A união destes dois tipos de variáveis fornece o estado do sistema.

Em termos formais, uma ESM básica é uma quintupla $(X,Y,C,\mathcal{L},A)$ onde:

- X é um conjunto unitário {x} representando a variável de monitoração x, assumindo valores do tipo type (x).
- Y é um conjunto de variáveis de dados onde cada variável de dados $y \in Y$ assuma valores do tipo type(y).
- C é um conjunto de canais de comunicação.
- $\mathcal{L}$ é um conjunto de (Rótulos de) eventos.
- A é um conjunto de ações básicas. Uma ação básica é uma dupla (x,E) onde x é uma variável de monitoração e E é um evento.

Um evento é definido como uma quádrupla (a_s, guarda, operação, a_d) onde:

- a_s - valor de x que originou o evento (valor-fonte), $a_s \in$ type(x).
- a_d - valor de x após a ocorrência do evento (valor-destino), $a_d \in$ type(x).
- guarda - condição booleana a ser satisfeita para que o evento ocorra.
- operação - ação "concreta" tomada pelo sistema, Pode ser uma atribuição, um envio, ou recebimento.

As ações básicas podem ser usadas para compor ações mais complexas como interações. Formalmente tem-se:

Ações - Uma ação ou é uma ação-básica ou uma interação. Uma interação pode ser uma ação compartilhada ou uma ação de comunicação

Ação Compartilhada - É definida indutivamente da seguinte maneira:

A. Sejam A_i e A_j ações (básicas) de atribuição de ESMs diferentes tendo o mesmo rótulo de evento α. Então $A_i \cup A_j$ é também uma ação compartilhada com rótulo de evento associado α.

B. Sejam A_i e A_j, cada uma por si mesma, ou uma ação de atribuição ou uma ação compartilhada tendo rótulos de eventos compartilhados α. Então $a_i \cup A_j$ é também uma ação compartilhada com rótulo de evento α.

Ação de comunicação - Sejam A_s e A_r ações (básicas) complementares de envio e recepção, respectivamente. Então $A_s \cup A_r$ é uma ação de comunicação.

Se $A_i \cup A_j$ é uma interação, então A_i e A_j são chamadas ações casadas.

Ainda se tratando de ações, pode-se ter composição paralela de conjuntos de ações. Para uma abordagem mais profunda recomenda-se a referência [23]. É representado por $A_i // A_j$, onde A_i e A_j são conjuntos de ações.

É possível definir composição paralela de ESMs. Novos ESMs são gerados de ESMs básicos pela aplicação recursiva das seguintes regras:

1) Qualquer ESM básico é um ESM
2) Sejam $M_i = (X_i, Y_i, C_i, \mathcal{L}_i, A_i)$ e $M_j = (Y_j, Y_j, C_j, \mathcal{L}_j, A_j)$ dois ESMs distintos ($X_i \cup X_j = \phi$ e $Y_i \cap Y_j = \phi$). Então $M_i // M_j \triangleq (X_i \cup X_j, Y_i \cup Y_j, (C_i \cup C_j) - (C_i \cap C_j), \mathcal{L}_i \cup \mathcal{L}_j, A_i // A_j))$. É um ESM.

Um conceito importante para a associação de toda a parte sintática dos ESMs, anteriormente vista, com a RTTL é o conceito de trajetória. A trajetória, definida sobre um espaço de estados S, é uma sequência infinita de estados. O conjunto de todas as trajetórias de um ESM M é denominado S^w. Uma outra razão para se ter as trajetórias é que elas possibilitam uma descrição precisa da dinâmica de M.

Um gerador de trajetórias para um ESM $M = (X,Y,C,\mathcal{L},A)$, denotado por G_M, é uma sextupla dada por

$$G_M = (V_M, R_M, S_M, \theta_M, T_M, J_M)$$

V_M - conjunto de variáveis
R_M - conjunto dos tipos das variáveis
S_M - espaço de estados
θ_M - conjunto de estados iniciais
T_M - conjunto de todas as transições de M
J_M - "Família de Justiça".

De todas as trajetórias de S^w, destacam-se algumas que têm propriedades especiais. Elas são as **trajetórias legais**. Uma trajetória σ é qualquer sequência infinita $s_0 s_1 s_2 \ldots$ de estados em S, onde S é o espaço de estados de um ESM.

O sistema Ostroff-Wonham utiliza a abordagem dual nos mesmos moldes que a abordagem dual composta pelo modelo Fair Transition System (FTS) em conjunto com a RTTL. Em geral, a RTTL se presta para verificação e desenvolvimento de controladores para qualquer aplicação onde se tenha a aplicação do FTS na modelagem [36]. Uma informação interessante acerca do FTS é que ele pode, sózinho, ser utilizado para a descrição da planta e do controlador e para a especificação do comportamento em malha-fechada [24]. Desta forma tem-se uma abordagem em linguagem única, ou primal.

4.6. ABORDAGEM LIN-IONESCU

A presente abordagem foi proposta por Lin e Ionescu na referência [36]. Os autores desenvolvem uma lógica temporal adaptada de Thistle-Wonham [28] onde introduzem noções de não-determinismo. Com isto, a lógica temporal é estendida para tratar DEDS não determinísticos. Esta abordagem pode ser incluída na lista de abordagens de linguagem única, ou primal. Nela, a lógica temporal generalizada modela a planta, o controlador e fornece as especificações em malha-fechada.

Quando um DEDS é apresentado através de fórmulas em lógica temporal, a este conjunto de fórmulas pode ser associado um modelo estocástico limitado (BSM). Este modelo apresenta uma modelagem abrangente e concisa dos DEDS, onde as fórmulas temporais têm a função de orientar a dinâmica do sistema fornecendo condições a serem satisfeitas para que certos eventos ocorram e, consequentemente, certos estados sejam alcançados. Segue-se uma breve exposição dos Modelos Estocásticos Limitados (BSMs). Esta exposição pode ser encontrada mais detalhadamente em [36]. Uma outra referência para as idéias preliminares usadas em [36] é a referência [24].

BSM

Dado um conjunto F de fórmulas, um BSM é uma quintupla $(S,E,s_0,\ell,\mathcal{P})$ onde:

S - conjunto não-vazio e enumerável de estados;

E - conjunto de eventos possíveis, $e_{ij} \in E$, e e_{ij} é o evento que sinaliza a transição entre s_i e s_j, onde $s_i,s_j \in S$. E é enumerável;

s_0 - estado inicial, $s_0 \in S$;

ℓ - $\ell:S \rightarrow F^*$ (F^{*0} denota os conjuntos de todos os sub-conjuntos de F) é uma função que associa a cada estado de S um subconjunto de fórmulas de F. Estas fórmulas dão o comportamento do sistema naquele estado. As condições para que um dado evento efetue a transição deste para um outro estado, devem estar prescritas neste conjunto de fórmulas.

$\mathcal{P}$ - $\mathcal{P}:E \rightarrow [0,1]$ associa a todo evento possível e_{ij} uma probabilidade $\mathcal{P}_{ij} \triangleq \mathcal{P}(e_{ij})$, de modo que para todo estado $s_i \in S$ e todos os

eventos $e_{ij}:s_i \rightarrow s_j$, $j \in J$ tem-se $\sum_{j\in J} \mathcal{P}_{ij} = 1$. O conjunto de estados pode ser infinito, mas existe um número real positivo $\alpha>0$ tal que para todo evento e_{ij} se $\mathcal{P}_{ij} > 0$ então $\mathcal{P}_{ij} > \alpha$, ou seja $\mathcal{P}_{ij}$ é limitada.

A noção de BSM é colocada para caracterizar os aspectos semânticos introduzidos na lógica temporal. Se o conjunto J é unitário, {j}, tem-se que os DEDSs a serem analisados serão determinísticos, e recaiu-se na abordagem Thistle-Wonham. Nesta linha, um BSM M é interpretado como uma trajetória possível que começa em s_0, executa transições de estados e gera uma sequência de eventos, ou trajetória. Fazendo uma alteração na notação, de modo que $e_{ij}:s_i \rightarrow s_j$ se torne $e_{k+1}:s_k \rightarrow s_{k+1}$, $k = 0,1,2,...$, pode-se representar uma trajetória de s_0 até s_n como sendo $\sigma_n \triangleq s_0s_1...s_n$. S^* denota o conjunto de todas as finitas ou infinitas trajetórias sobre S e W_s o conjunto das trajetórias de S que têm como estado inicial o estado $s = s_0$.

Baseado na probabilidade da ocorrência de um evento, pode-se definir uma distribuição de probabilidade para um conjunto de trajetórias W_s, iniciadas no estado s. Esta função de probabilidade é denotada por $\tilde{P}_s$. De forma mais detalhada, se B é um conjunto, mensurável, das trajetórias de W_s compostas de n termos ($\sigma_n = s_0s_1...s_n$), tem-se que $\tilde{P}s_0(B) = P(e_1) \times P(e_2) \times ... \times P(e_n)$, onde $e_i:s_{i-1} \rightarrow s_i$. Para cada conjunto de estados diferentes, a probabilidade assume valores diferentes.

Após esta exposição sobre o BSM, apresenta-se um resumo das principais alterações feitas na lógica temporal utilizada por Thistle e Wonham, com o intuito de adequá-las às necessidades dos DEDS não-determinísticos. Esta expansão da lógica temporal introduz operadores que tratam com noções de certeza (probabilidade 1) e possibilidade, num sentido probabilístico (probabilidade maior que zero). Os axiomas que tratam só destes novos operadores são os mesmos de [36] bem como os que relacionam os operadores temporais com os operadores modais envolvendo incerteza.

LÓGICA TEMPORAL GENERALIZADA

A lógica temporal modificada será apresentada rapidamente. Como a GRTTL, a ser apresentada no capítulo 5, deriva desta lógica temporal, é de suma importância entender a abordagem Lin-Ionescu. Com o objetivo de se ter

uma apresentação sucinta, só serão ressaltados os acréscimos ao Sistema de Thistle e Wonham.

Símbolos: símbolos constantes individuais (globais);
símbolos variáveis locais;
letras de função;
letras de predicado;
conectivos lógicos ($\neg$ e $\vee$);
operadores temporais ($\odot$, $\Box$ e $\mathcal{U}$);
operadores modal: ∇ - operador certeza, probabilidade 1.

Fórmulas-bem-formadas:

i) Para quaisquer termos $t_1,...,t_n$, qualquer símbolo de predicado $\mathcal{P}$, $\mathcal{P}(t_1,...,t_n)$ é uma fórmula;
ii) Para qualquer fórmula w, ($\neg$w), ($\odot$w), ($\Box$w) e (∇w) são fórmulas;
iii) Para quaisquer fórmulas v e w, (v $\vee$ w) e v$\mathcal{U}$w são fórmulas.

Em termos semânticos, as fórmulas são avaliados em função da dupla (M,σ), onde M é um modelo $M = (S,E,s_0,\ell,\mathcal{P})$ que especifica uma interpretação e σ fornece uma trajetória. A noção de satisfação é como já descrito anteriormente, acrescida de um item que se refere à satisfação de uma fórmula ∇w, ou seja:

$M\sigma \models (\nabla w)$ se e somente se $\tilde{P}s_0 \left(\{ \tau / \tau \in Ws_0 \text{ e } M\tau \models w \} \right) = 1$, onde Ws_0 é o conjunto de todos os caminhos de S^* que começam em s_0.

Uma abreviação distinta das já citadas, é a abreviação do operador possível:

operador possível: (Δw) abrevia ($\neg(\nabla(\neg w))$).

O sistema de prova é visto em níveis, cada nível relacionando-se com uma classe de fórmulas diferentes. O primeiro nível trata das fórmulas temporais, semelhantes às tratadas por Thistle-Wonham. O segundo nível enfoca os axiomas necessários para a análise de fórmulas envolvendo certeza (∇ e Δ). São eles:

A20) $\nabla(w_1 \Rightarrow w_2) \Rightarrow (\nabla w_1 \Rightarrow \nabla w_2)$;

A21) $\Delta\nabla w \Leftrightarrow \nabla w$;

A22) $\nabla w \Rightarrow w$;

Além destes 3 axiomas, uma regra é introduzida para a argumentação formal.

R3) $\vdash w$, então $\vdash \nabla w$

É possível fazer uma associação deste nível com o modelo do sistema LPC + S5 da lógica modal [21]. No LPC + S5 tem-se noções de certeza definidas, desde que se assuma que não há diferença entre satisfação e satisfação com probabilidade 1.

Completando os níveis, tem-se o terceiro nível que é composto de axiomas que relacionam os conceitos temporais com o conceito modal de incerteza. É importante frisar bem neste momento que os operadores de incerteza (∇, Δ) são atemporais, ou sejam, não são avaliados sobre uma trajetória [24]. Seguem-se os axiomas:

A23) $\nabla\odot w \Rightarrow \odot\nabla w$

Em termos intuitivos, o avanço do tempo não tem influência sobre algo que ocorrerá com probabilidade um.

A24) $\Box\Diamond_\Delta \left[\bigwedge_{i=0}^{k} \odot^{(i)}\nabla w_i\right] \Rightarrow \Diamond\left[\bigwedge_{i=0}^{k} \odot^{(i)}\nabla w_i\right]$

ou de uma forma expandida:

$$\Box\Diamond(\nabla w_0 \wedge \Delta\odot(\nabla w_1 \wedge \Delta\odot(\nabla w_2 \wedge \Delta\odot(\ldots \wedge \Delta\odot\nabla w_k)\ldots)))) \Rightarrow \Diamond(w_0 \wedge \odot w_1 \wedge \odot\odot w_2 \wedge \ldots \wedge \odot^{(k)} w_k)$$

Embora a visualização do significado intuitivo deste axioma não seja imediata, ele garante que sucessivas escolhas aleatórias são independentes, no sistema em questão.

Em linhas gerais, está é a abordagem Lin-Ionescu. Por ser derivada da abordagem Thistle-Wonham, tem-se que ela apresenta a mesma falta de completude forte apresentada nesta última. A diferença básica entre as duas abordagens está na inclusão dos operadores ∇ e Δ, e nos resultados daí provenientes.

4.7. CONCLUSÃO

Neste capítulo foram apresentadas várias abordagens que utilizam a lógica temporal ora como ferramenta única de modelagem, verificação e síntese de controladores,ora como ferramenta auxiliar nestas tarefas. De uma maneira resumida, pode-se ter uma visão conjunta das vantagens e desvantagens de cada método. Um método pode ser classificado dentro dos seguintes conjuntos de características:

* Conforme Abordagem: Primal ou Dual;
* Conforme Tipo de sistema descrito: Determinístico ou não-determinístico;
* Conforme restrições de tempo real: Com ou sem restrições;
* Conforme espaço de estados do sistema: Finito ou Infinito.

O quadro sinóptico abaixo fornece uma rápida visualização das características dos formalismos envolvidos:

MÉTODO	ESTRATÉGIA DE ANÁLISE	ABORDAGEM	TIPO DE SISTEMA	RESTRIÇÕES TEMPO REAL	ESPAÇO DE ESTADOS
GRÁFICO W	TABLEAU SEMÂNTICO	PRIMAL	DETER-MINÍSTICO	SEM	FINITO
THISTLE WONHAM	DEDUÇÃO LÓGICA	PRIMAL	DETER-MINÍSTICO	SEM	INFINITO
KNIGHT PASSINO	SEQUÊNCIAS PERIÓDICAS	DUAL	NÃO DETER-MINÍSTICO	SEM	FINITO
OSTROFF WONHAM	DEDUÇÃO LÓGICA	DUAL	NÃO DETER-MINÍSTICO	COM	INFINITO
LIN IONESCU	DEDUÇÃO LÓGICA	PRIMAL	NÃO DETER-MINÍSTICO	SEM	INFINITO

figura 8

Uma importante lacuna é notada na existência de uma abordagem por linguagem primal, usando lógica temporal, onde se tenha: 1) facilidades de descrição de cotas inferiores e superiores do tempo de permanência num estado, ou melhor, do tempo decorrido entre a ocorrência de dois eventos; 2) capacidade de tratar DEDS não-determinísticos através da própria lógica temporal, sem recorrer a um formalismo auxiliar, como na abordagem dual. O próximo capítulo traz uma proposta que visa preencher estes requisitos, constituindo-se a principal contribuição deste trabalho.

CAPÍTULO 5

APRESENTAÇÃO DA GRTTL

5.1. INTRODUÇÃO

No estudo dos Sistemas Dinâmicos a eventos discretos (DEDS) depara-se com sistemas que têm características não-determinísticas na sua dinâmica. Estes DEDS não-determinísticos são caracterizados pela possibilidade de sair de um estado X_A e ir para qualquer estado dentro de um conjunto de estados acessíveis X_K, K = 1,...,n, com n qualquer, sem condições de se estabelecer, a priori, para qual estado o sistema transicionará. Aliado a este não-determinismo é freqüente o aparecimento de sistemas onde existem restrições de tempo real, ou seja, limites inferiores e superiores de tempo em que o sistema deve mudar de estado sob o risco de problemas na sua dinâmica. Exemplos clássicos disto aparecem em células de manufatura onde se tem limites de tempo para execução de certas tarefas. O controlador deve garantir que estas tarefas ocorram nos prazos, e para isto certas ações são por ele tomadas.

Das abordagens examinadas até o presente, em particular as citadas no capítulo 3, nenhuma delas fornece um formalismo teórico consistente para o estudo de DEDS não-determinísticos com limitantes de tempo real que use os conceitos de linguagem primal. O sistema Ostroff-Wonham analisa este tipo de DEDS, mas por uma abordagem dual. Entre as vantagens citadas em se obter uma linguagem primal para estes sistemas está a facilidade de se trabalhar com um só formalismo na modelagem, verificação, desenvolvimento e síntese de controladores. Este conceito tem mais proximidade com a teoria para os Sistemas dinâmicos a variáveis contínuas (CVDS) que utiliza as equações diferenciais, ou as diferenças, no seu estudo.

Neste capítulo é apresentado um formalismo baseado na lógica temporal, chamado de Lógica Temporal de Tempo Real Generalizada (Generalized Real-Time Temporal Logic) que será simbolizado pela sigla GRTTL. A GRTTL incorporará elementos de três abordagens anteriormente vistas. Da abordagem Thistle-Wonham ela herda todo o arcabouço axiomático e o sistema de prova, inclusive a exclusão da regra "$\Box$-inserção" presente no sistema Manna-Pnueli [25]. Da abordagem Ostroff-Wonham, ou melhor da RTTL, a GRTTL absorve os conceitos de limites inferiores e superiores do tempo de permanência num estado, os elementos do sistema de prova referentes às restrições de tempo real e a introdução dos quantificadores existencial ($\exists$) e, seu dual, universal ($\forall$), transformando-se numa lógica de primeira ordem. A abordagem Lin-Ionescu

contribui com os operadores "certeza" ou "com probabilidade 1" (∇) e, seu dual, possibilidade **probabilística** (Δ) que fornece à GRTTL ferramentas no trato de DEDS não-determinísticos. O formalismo emergente da fusão destes elementos traz características interessantes na análise de DEDS que apresentam não-determinismo e restrições temporais na sua dinâmica.

A metodologia para desenvolvimento de controladores usada é a mesma introduzida em [28]. Primeiro, descreve-se a planta (sistema a ser controlado) em GRTTL. Esta descrição modela, inclusive, o caráter não-determinístico da planta. Em seguida, especifica-se em GRTTL o comportamento desejado para o conjunto planta-controlador, ou seja, especificações de malha-fechada. Finalmente, propõe-se um conjunto de equações, em GRTTL, que somadas àquele da planta possibilitarão a satisfação das equações de malha-fechada. Este conjunto de equações introduzido derradeiramente nada mais é do que o controlador. É importante observar que, muito embora a planta possa ser não-determinística, o controlador é determinístico.

Para fins de modelagem pela GRTTL, os DEDS serão vistos como Modelos Estocásticos Limitados de Tempo Real, ou RT-BSM (Real-Time Bounded Stochastic Model). Estes são uma generalização dos BSMs apresentados em [24] e aprimorados inicialmente em [36]. Sobre os sistemas não-determinísticos, vale salientar que faz-se a distinção entre sistemas não-determinísticos e sistemas estocásticos. Embora esteja associada uma probabilidade p a uma transição, esta probabilidade não será utilizada na análise do sistema ou síntese do controlador. Sendo assim, ela só denota que a escolha é aleatória sem necessitar de uma análise estocástica (média, variância, etc) da situação. A justificativa para tal diferença é que existem casos onde faz pouca diferença o conhecimento quantitativo das grandezas probabilísticas envolvidas. Nesta situação, o conhecimento de que certos eventos ocorrem certamente (probabilidade 1) ou possivelmente (probabilidade entre 0 e 1, exclusive) fornece subsídios suficientes para o tratamento do sistema. Para uma análise mais acurada desta distinção ver [24] e referências ali indicadas.

Nas seções que se seguem apresenta-se um arcabouço para o RT-BSM, bem como para a GRTTL.

5.2. RT-BSM

Os modelos estocásticos limitados de tempo real são um formalismo utilizado na descrição de DEDS não-determinísticos com limitantes de tempo-real para o uso da GRTTL. A grosso modo, a qualquer sistema $\sum$ pode ser associado um conjunto de eventos E, que ocorrem em certo instantes de tempo, e um conjunto de fórmulas temporais F, que fornecem as condições para que os eventos ocorram ou não, ou seja, o comportamento dinâmico do sistema. É interessante notar que só com estes dois conjuntos pode-se englobar a dinâmica de um sistema $\sum$. O conceito de estado aparece associado a estes dois primeiros. Um evento $e \in E$ pode ser encarado como aquele que modifica o subconjunto de fórmulas que regia o comportamento do sistema até aquele instante. Por definição, a ocorrência de um evento leva o sistema de um estado para outro. Comparando estas duas últimas noções (subconjunto de fórmulas e estado) vê-se que o estado do sistema pode ser representado por um subconjunto de fórmulas; cada estado é regido por um subconjunto.

Cada seqüência de eventos define um RT-BSM. Como um sistema pode ter várias seqüências de eventos possíveis, diz-se que um sistema $\sum$ é composto por vários RT-BSMs. O formalismo aqui apresentado é uma expansão do BSM proposto por Lin-Ionescu em [36], onde se permite tratar com características de tempo real. Uma bibliografia anterior ao trabalho de Lin-Ionescu, fornece vários tipos de Modelos (gerais, limitados, finitos) [37]. Recomenda-se a mesma para um melhor entendimento das relações entre DEDS não-determinísticos e os modelos estocáticos.

Os RT-BSMs são definidos formalmente como segue:

Definição 3.1. Dado um conjunto F, de fórmulas, um modelo estocástico limitado de tempo real é uma sétupla (S,E,s_0,A,P,ℓ,u) onde:

S - é um conjunto enumerável e não-vazio de estados.

E - é um conjunto de eventos possíveis, cada evento $e_{ij} \in E$ sendo uma transição de s_i a s_j como $s_i,s_j \in S$ e, deste modo, E também é enumerável.

s_0 - $s_0 \in S$ é o estado inicial.

A - $A:S \rightarrow F^*$ (F^* denota o conjunto de todos os subconjuntos de F) é uma função que associa cada estado de S a um conjunto de

fórmulas em F^*.

P - $P:E \rightarrow [0,1]$ associa a cada evento possível e_{ij} uma probabilidade $P_{ij} \triangleq p(e_{ij})$, de modo que para todo estado $s_i \in S$ e todo evento $e_{ij}:s_i \rightarrow s_j$ $(j \in J)$, $\sum_{j \in J} P_{ij} = 1$. O número de estados pode ser infinito, mas é garantido que existe um $\alpha > 0$ tal que para todo e_{ij}, se $P_{ij} > 0$ então $P_{ij} > \alpha$. Isto garante que o número de estados s_j que podem ser alcançados de s_i seja limitado. Esta é razão porque este modelo é chamado de **limitado**. Se o conjunto J é unitário J = (j) tem-se que o sistema é determinístico.

$\ell:S \rightarrow \mathbb{R}^+$ - Dado um estado $s \in S$, ℓ corresponde ao tempo mínimo de permanência neste estado. Em termos de eventos, ele é o tempo mínimo entre a ocorrência de um evento que leva o sistema ao estado s e a ocorrência de um evento que leva o sistema do estado s para um outro qualquer.

$u:S \rightarrow \mathbb{R}^+$ - De um modo análogo ao já exposto para ℓ, u corresponde ao tempo máximo de permanência num estado $s \in S$.

No caso de $\ell = 0$ e $u = \infty$, recai-se num caso particular onde o sistema é autônomo (não existem limites de tempo-real). Este é o caso do sistema proposto em [36].

Considerando um RT-BSM M pode-se observar que ele indica uma determinada trajetória do sistema, pois tem-se a indicação do estado inicial, $s_0 \in S$, e toda a seqüência de eventos, que possibilita obter a seqüência de estados resultante. É comum escrever o evento $e_{ij}: s_i \rightarrow s_j$ como $e_{k+1}:s_k \rightarrow s_{k+1}$, $k = 0,1,...$, renumerando os subscritos. Em termos notacionais, um trecho de uma trajetória σ que se inicia em s_0 e finda em s_n é descrito como: $\sigma_n = s_0 s_1 s_2 ... s_n$. Representando o conjunto de todas as seqüências de S por S^*, tem-se que W_s é o subconjunto de S^* cujos elementos são seqüências que iniciam por s, ou seja, $s_0 = s$.

A cada transição de um estado s_k para um outro s_{k+1}, está associada uma probabilidade p. Observando uma trajetória finita qualquer que se inicia em s, tem-se uma probabilidade para a ocorrência da mesma. O modelo em questão pode ser visto como uma Cadeia de Markov, pois a probabilidade de transição depende exclusivamente do estado anterior (s_k). Formalmente falando, pode-se definir uma função distribuição de probabilidade sobre o conjunto W_s. Esta função é chamada $\tilde{\mathcal{P}}_s$. Para que esta função fique bem estabelecida (sem casos anômalos) basta mostrar que o conjunto Q de todas as seqüências σ de S^*

satisfazendo a condição de ser limitada ($s_0 s_1 s_2 ... s_n$, $0 \leq n$), é mensurável. Satisfeito isto tem-se que

$$\tilde{P}_{s0}(Q) = p(e_1) \times p(e_2) \times ... \times p(e_n)$$

Onde $\tilde{P}_{s0}(Q)$ simboliza a função distribuição de probabilidade de todas seqüências Q que começam em s_0. Neste trabalho, esta função não será muito explorada. Sugere-se as referências [39,38] para uma abordagem mais detalhada das Cadeias de Markov e suas implicações probabilísticas.

Dada uma seqüência $\sigma \in S^*$, $\sigma = s_0 s_1 s_2 s_3 ...$, denota-se como $\sigma^{(1)}$ a seqüência $s_1 s_2 s_3 ...$, $\sigma^{(2)}$ como $s_2 s_3 s_4 ...$, e assim por diante.

5.3. SISTEMA FORMAL

O sistema formal da GRTTL é formado por componentes sintáticos e pelo sistema de prova. A semântica é vista na próxima seção. A sintaxe do sistema fornece uma definição clara dos símbolos utilizados na linguagem, enquanto que o sistema de prova apresenta uma ferramenta para a argumentação lógica, dedução e geração de teoremas. Muito deste capítulo foi apresentado anteriormente, de forma esparsa.

5.3.1. SINTAXE

A Seguir, apresenta-se os símbolos, termos e fórmulas usados.

A - SÍMBOLOS

Os símbolos podem ser símbolos lógicos ou parâmetros. Como já foram apresentados no capítulo 3, só serão mencionados aqui. Será introduzido um novo símbolo, que trata de **certeza** (noção modal).

- **Símbolos lógicos:**
  Igualdade: =
  Conectivos: $\neg$, $\longrightarrow$
  Parênteses: (,)
  Símbolos variáveis globais: U_0, U_1, ...

Símbolos variáveis locais: u_0, u_1, ...

Símbolos operadores temporais: $\odot$, $\mathcal{U}$, $\Diamond$

Símbolo operador certeza: ∇

- **Parâmetros:**

Símbolo quantificador: $\exists$

Símbolos constantes globais: C_0, C_1, ...

Letras de predicado: relações n-árias, n > 0

Letras de função: funções n-árias, n > 0

Seguindo a orientação do capítulo 3, a GRTTL é uma linguagem multi-sorte, onde existe um conjunto I não-vazio, cujos membros são chamados de sortes. Cada sorte tem seus próprios símbolos variáveis (globais e locais), seus símbolos constantes globais, predicados, funções e o símbolo quantificador. Cada sorte i tem um quantificador $\exists_i$ que quantifica sobre os elementos desta sorte.

B - TERMOS

Para qualquer sorte i, os termos de sorte i da linguagem são definidos indutivamente como segue:

- Todos os símbolos constantes globais são termos.
- Todos os símbolos variáveis (globais e locais) são termos.
- Se $t_1,...,t_n$ são termos e f é uma letra de função n-ária, então $f(t_1,...,t_n)$ é um termo.
- Se t é um termo, $\odot t$ é um termo.
- Nenhuma outra seqüência, diferente das anteriores, é um termo.

C - FÓRMULAS

As fórmulas-bem-formadas da linguagem são:

i) Se $t_1,t_2,...,t_n$ são termos de sorte $t_1,...,t_n$, respectivamente, e P é qualquer letra de predicado de sorte $i_1,i_2,...,i_n$, então $P(t_1,t_2,...,t_n)$ é uma fórmula-bem-formada.

ii) Se w é uma fórmula-bem-formada, então ($\neg$w), ($\odot$w), ($\Diamond$w) e (∇w) são fórmulas-bem-formadas.

iii) Se w_1 e w_2 são fórmulas-bem-formadas, então ($w_1 \rightarrow w_2$) e ($w_1 \mathcal{U} w_2$)

são fórmulas-bem-formadas.

iv) Se V é um símbolo variável global e W é uma fórmula-bem-formada, então (∃V:w) é uma fórmula-bem-formada.

v) Nenhuma outra seqüência é uma fórmula-bem-formada.

Adota-se aqui as mesmas definições de variáveis livres, mapeamento e substituição simultânea que as adotadas no capítulo 3, atentando-se para as inclusões abaixo referentes ao operador ∇.

- Variável livre: $livre(\nabla w_1) \triangleq livre(w_1)$, onde w_1 é uma fórmula qualquer.
- Substituição simultânea:

certeza $$(\nabla w_1)\begin{bmatrix} x_1 & \dots & x_n \\ g_1 & \dots & g_n \end{bmatrix} \triangleq \nabla\left(w_1\begin{bmatrix} x_1 & \dots & x_n \\ g_1 & \dots & g_n \end{bmatrix}\right)$$

5.3.2. SISTEMA DE PROVA

O sistema de prova de uma linguagem apresenta os axiomas e as regras de inferência, sendo usados na dedução de fórmulas (teoremas) úteis para a aplicação em questão. Tanto nos axiomas quanto nas regras de inferência, nota-se partes que dizem respeito à lógica temporal proposicional, às inclusões para a lógica temporal de 1ª ordem e aos acréscimos referente às noções de certeza e possibilidade. O sistema não será dividido em níveis, como feito em [36], mas faz-se um breve comentário ao final de cada bloco de axiomas e após uma regra de inferência relacionados com cada subdivisão acima exposta. **É importante notar que a numeração dos axiomas neste capítulo difere da adotada no capítulo 3 e 4. No apêndice, a numeração seguida será a deste capítulo.**

A - AXIOMAS (Esquemas de Axiomas)

Sejam w, w_1 e w_2 fórmulas-bem-formadas nas formas abaixo. Então w e $\Box w$ são axiomas (onde $\Box w = \neg \Diamond \neg w$):

A1) w, onde w é uma instância de uma tautologia.

A2) $\Box(w_1 \longrightarrow w_2) \longrightarrow (\Box w_1 \longrightarrow \Box w_2)$

A3) $\Box w_1 \longrightarrow w_1$

A4) $\odot \neg w_1 \longleftrightarrow \neg \odot w_1$

A5) $\odot((w_1 \longrightarrow w_2) \longrightarrow (\odot w_1 \longrightarrow \odot w_2)$

A6) $\Box w_1 \longrightarrow \odot w_1$

A7) $\Box w_1 \longrightarrow \odot \Box w_1$

A8) $\Box(w_1 \longrightarrow \odot w_1) \longrightarrow (w_1 \longrightarrow \Box w_1)$

A9) $w_1 \mathcal{U} w_2 \longleftrightarrow w_2 \vee (w_1 \wedge \odot (w_1 \mathcal{U} w_2))$

A10) $w_1 \mathcal{U} w_2 \longrightarrow \Diamond w_2$

Os próximos axiomas refletem o fato de que constantes, variáveis globais, funções e predicados mantém-se inalteradas com o avanço da trajetória. A definição de globsub(t,x,w) é como no capítulo 3.

A11) t = t para qualquer termo t.

A12) $t_1 = t_2 \longrightarrow (\phi(t_1,t_1) \longleftrightarrow \phi(t_1,t_2))$

Onde i) ϕ é uma fórmula de estado (sem operadores temporais)

ii) globsub($t_2,t_1,\phi(t_1,t_2)$) ocorre.

A13) $\odot \mathcal{P}(t_1,t_2,...,t_n) \Longleftrightarrow \mathcal{P}((\odot t_1),(\odot t_2),...,(\odot t_n))$, para qualquer letra de predicado n-ário e termos $t_1,...,t_n$.

A14) $\odot f(t_1,t_2,...,t_n) = f((\odot t_1),(\odot t_2),...,(\odot t_n))$, para qualquer letra de função f e termos $t_1,t_2,...,t_n$.

A15) C = (⊙C), para qualquer símbolo constante global C.

A16) V = (⊙V) onde V é qualquer variável global.

Estes axiomas dizem respeito à lógica temporal proposicional.

Antes da apresentação dos axiomas referentes à lógica temporal de 1ª ordem, é importante ter em mente a definição de substituibilidade, como apresentada no capítulo 3.

A17) $(\exists V{:}w) \longrightarrow w \left[\begin{smallmatrix} V \\ t \end{smallmatrix} \right]$ onde globsub(t,V,w) ocorre.

A18) $\odot(\exists V{:}w) \longrightarrow (\exists V{:}\odot w)$

Um outro grupo de axioma diz respeito à introdução do operador ∇ no sistema de prova da GRTTL representando a noção de certeza (probabilidade 1). Nestes axiomas o símbolo Δ foi introduzido para abreviar $\Delta w = \neg \nabla \neg w$, sendo

mais precisamente explicado posteriormente.

A19) $\nabla(w_1 \longrightarrow w_2) \longrightarrow (\nabla w_1 \longrightarrow \nabla w_2)$;

A20) $\Delta\nabla w \longleftrightarrow w$;

A21) $\nabla w \longrightarrow w$;

A22) $\nabla \odot w \longrightarrow \odot \nabla w$

A23) $\Box \Diamond \Delta \left[\Lambda_{i=0}^{k} \odot^{(i)} \nabla w_i\right] \longrightarrow \Diamond \left[\Lambda_{i=0}^{k} \odot^{(t)} \nabla w_i\right]$

Este quatro axiomas delineiam verdades gerais sobre certeza (A19-A21) e as interrelações entre tempo e aleatoriedade, num resumo do sistema Lin-Ionescu [36].

O operador ∇ é introduzido neste sistema e proporciona a primeira diferença conceitual do sistema proposto por Ostroff [23]. O aparecimento de ∇ indica que, na escolha não determinística que se faz, com probabilidade 1, ocorrerá o descrito na fórmula que é seu operando. Maiores detalhes serão fornecidos quando for abordada a semântica da GRTTL.

Uma outra observação importante é que o operador ∇, e seu dual Δ, podem ser aplicados a fórmulas e, em particular, a variáveis locais. Como o não determinismo depende basicamente dos valores das variáveis locais, pode-se dizer que o operador ∇ atua sobre as variáveis locais, realizando uma tarefa não permitida ao operador $\exists$ (e seu dual $\forall$) que só quantificam variáveis globais.

Dois axiomas da lógica temporal proposicional foram deixados para o final por necessitarem de esclarecimentos adicionais.

Na abordagem de DEDSs pela GRTTL, os sistemas são descritos em fórmulas da linguagem, e isto com o objetivo de se fazer argumentação lógico-matemática sobre o domínio desta interpretação. Para que fórmulas auxiliares que são obviamente verdadeiras possam ser consideradas como axiomas, o seguinte esquema de axioma é enunciado:

A24) Se w é uma fórmula que é verdadeira sob uma valoração pretendida, então w e $\Box$w são axiomas.

Em outras palavras, este esquema de axioma permite a inclusão no corpo de axiomas de resultados óbvios (por exemplo, axiomas de Peano para se trabalhar com os números naturais).

Quando se trabalha com uma linguagem dual, normalmente uma das linguagens trata de representar o conjunto de eventos do sistema. Na abordagem proposta por Ostroff, o ESM se encarrega desta representação. Quando, porém, a linguagem é primal, ela mesma deve providenciar uma maneira para representar o conjunto de eventos. Na GRTTL, uma sorte é separada para ser a sorte de eventos. Seus símbolos constantes globais são chamados de símbolos de eventos e ela possui um símbolo variável local representado por δ que denota o evento que ocorre neste instante de tempo. Sempre que numa fórmula ocorrer o termo $\delta = \alpha$, onde α é um símbolo de evento, significa que, naquela fórmula, o evento α ocorre naquele instante de tempo. Se o sistema pára de transicionar ou entra num estado de travamento, isto é representado pelo evento nulo $\delta = \phi$ que não altera nenhuma das variáveis do sistema. Para formalizar isto tem-se o seguinte axioma:

A25) $\Box [\delta = \phi \longrightarrow (\circ x) = x]$, onde x é um símbolo variável local.

Introduzindo as questões de tempo real na GRTTL, foi necessário criar uma sorte da linguagem para representação do **tempo associado às transições**. (equivalente conceitualmente ao **tempo de permanência num estado**). Os símbolos constantes globais, são números reais. O símbolo t é incluído entre as variáveis locais desta sorte. Sendo assim, o termo t = T, onde T é uma meta-variável, significa que o tempo, na fórmula que contém este termo, vale T unidades. Dado que a GRTTL é uma lógica temporal discreta (ver capítulo 3), pode-se falar no operador "next", ou seja, os eventos ocorrem em instantes discretos. A partir disto, é possível associar aos símbolos ℓ e u do RT-BSM, um análogo na GRTTL, onde ℓ é a cota de tempo mínima entre a ocorrência de dois eventos e u é a cota máxima. O tempo entre a ocorrência de dois eventos também pode ser encarado como o tempo de permanência no estado que foi alcançado após o primeiro evento ter acontecido, e que é abandonado após a ocorrência do segundo evento em questão.

Sejam O, A e B estados O, A, B $\in$ S e α, $\beta \in$ E eventos que levam de O a A e de A a B, respectivamente, como na figura abaixo:

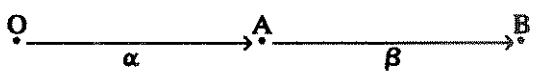


figura 9

Sejam ainda $\geq$ e $\leq$ predicados binários com as noções intuitivas de "maior ou igual a" e "menor ou igual a" respectivamente. Então, pode-se usar ℓ_A e u_A como abreviatura das seguintes fórmulas, respectivamente:

ℓ_A = a abrevia $\quad \Box\left[(\delta=\alpha \wedge t=T) \rightarrow \Diamond\ [\delta=\beta \wedge t \geq T+a)]\right]$

u_A = b abrevia $\quad \Box\left[(\delta=\alpha \wedge t=T) \rightarrow \Diamond\ [\delta=\beta \wedge t \leq T+b)]\right]$

onde a e b são meta-variáveis sobre a sorte do tempo.

Portanto, ℓ_A e u_A representam, respectivamente, o tempo mínimo e máximo de permanência no estado A.

B - REGRAS DE INFERÊNCIA

As regras de inferência podem ser vistas como uma forma de se gerar novas fórmulas, a partir dos axiomas apresentados, que não sejam "inconsistentes" com estes. Serão enunciadas três regras primitivas.

R1) Modus Ponens - MP

$$\frac{w_1, w_1 \rightarrow w_2}{w_2}$$

R2) $\forall$ - Inserção ($\forall$I)

$$\frac{w_1 \rightarrow w_2}{w_1 \rightarrow (\forall V : w_2)} \text{ onde } V \in \text{livre } (w_1)$$

R3) ∇ - Inserção (∇I)

$$\frac{w}{\nabla w}$$

A regra R3 diz que não há diferença entre satisfação e satisfação com probabilidade 1.

Nas regras acima $\frac{w_1, w_2 \ldots w_n}{w}$ significa que se pode inferir w das fórmulas $w_1, w_2, \ldots, w_n$. Note que este é um conceito sintático, e não semântico. A principal diferença aqui, além da inclusão de ∇I, em relação ao sistema proposto em [23] é a ausência da regra $\frac{w}{\Box w}$. Esta regra é retirada porque, neste trabalho, uma fórmula sem operadores temporais diz respeito ao estado inicial. Isto acarreta a perda da **completude forte** como será visto no final do capítulo.

A partir do conjunto de axiomas e das regras (primitivas) de inferência pode-se gerar uma infinidade de teoremas, ou seja, fórmulas que são derivadas dos primeiros através de um processo chamado de **dedução**. A definição de dedução aqui é a mesma encontrada em Ostroff [23] e em tantas outras referências em lógica.

Definição - w é uma dedução de um conjunto de fórmula ϕ (notação: $\phi \vdash w$), se e somente se existe uma seqüência finita de fórmulas $w_1,...,w_n$ tal que $w_n = w$, e cada w_i é uma das possibilidades seguintes:

i) é um axioma,

ii) é uma fórmula de ϕ ou

iii) obtida de fórmulas anteriores por uma regra de inferência

Definição - w é um teorema (notação:$(\vdash w)$, se e somente se $\{\ \} \vdash w$, onde $\{\ \}$ representa o conjunto vazio.

Além do conjunto de axiomas, pode-se usar o processo de dedução para se ter novos teoremas, que auxiliam na prova de resultados. Do mesmo modo, pode-se ter regras de inferências derivadas, que são utilizadas com o intuito de simplificar a argumentação em vários casos. No apêndice são listados vários teoremas e regras de inferência derivadas. Como a GRTTL é em muito semelhante à RTTL, vários teoremas e regras derivadas foram transcritas de [23], tendo-se o cuidado de se observar as alterações introduzidas pela ausência da regra $\Box$-inserção. Pelo fato da GRTTL ser uma generalização (sistema mais forte) do que o proposto por Thistle-Wonham [28] e por Lin-Ionescu [36], todos os resultados ali enunciados são aceitos aqui.

Entre as regras derivadas, as que tratam de argumentação sobre questões de tempo real são as mais interessantes para a GRTTL, visto que este é o ponto crucial de suas diferenças para com o sistema de Lin-Ionescu. A seguinte regra derivada é similar àquela apresentada em [23] onde se tem o cuidado de se adaptar os resultados para a GRTTL. Muito embora a prova seja em muito semelhante ao exposto em [23], ela difere no uso do operador $\Box$ mais externo e por estar num outro sistema. A argumentação que se segue ilustra a maneira pela qual os resultados contidos em [23] podem ser estendidos para a GRTTL, desde que observadas as premissas e regras usadas na prova. A regra a seguir expressa resultados sobre soma de limites de tempo superiores. Vale salientar que $(\forall V{:}w) \stackrel{\Delta}{=} \neg(\exists V{:}(\neg w))$.

Sejam w_1, w_2 e w_3 quaisquer fórmulas temporais e sejam d_1 e d_2

constantes quaisquer da sorte de tempo e T uma variável global da mesma sorte. Então:

$$\frac{\Box\left[(\forall T{:}(w_1 \wedge t{=}T) \rightarrow \Diamond\, (w_2 \wedge t{\leq}T{+}d_1))\right] \quad \Box\left[(\forall T{:}(w_2 \wedge t{=}T) \rightarrow \Diamond\, (w_3 \wedge t{\leq}T{+}d_2))\right]}{\Box\left[(\forall T{:}(w_1 \wedge t{=}T) \rightarrow \Diamond\, (w_3 \wedge t{\leq}T{+}d_1{+}d_2))\right]}$$

No esboço de prova que se segue, os passos são enumerados e estes números passam a fazer a identificação desta expressão nos passos seguintes. Do lado direito da expressão, é apresentada uma coluna onde se indica os passos anteriores e as regras (primitivas e/ou derivadas) para a geração deste passo. As premissas são introduzidas tendo elas mesmas como fórmulas primitivas. Os teoremas e as regras citadas no desenvolvimento são resultados apresentados por Ostroff [23] que têm aplicação imediata na GRTTL, por enfocarem aspectos comuns à RTTL e à GRTTL. PR significa argumentação proposicional.

Prova:

1) $\Box\left[(\forall T{:}w_1 \wedge t{=}T \rightarrow \Diamond\, (w_2 \wedge t \leq T{+}d_1))\right]$ (1)

Premissa 1.

2) $\Box\left[(\forall T{:}w_2 \wedge t{=}T \rightarrow \Diamond\, (w_3 \wedge t \leq T{+}d_2))\right]$ (2)

Premissa 2.

3) $(\forall T{:}w_1 \wedge t{=}T \rightarrow \Diamond\, (w_2 \wedge t \leq T{+}d_1))$ A3,1

Retirada do operador $\Box$.

4) $(\forall T{:}w_2 \wedge t{=}T \rightarrow \Diamond\, (w_3 \wedge t \leq T{+}d_2))$ A3,2

Retirada do operador $\Box$.

5) $(\forall T_1{:}t < T_1) \leftrightarrow (\exists T_2{:}t_1{=}T_1{-}T_2 \wedge T_2 \neq 0)$ A24,T1,T2

Axioma de domínio, novas variáveis (T1,T2)

6) $(t < T_1+d_1) \leftrightarrow (\exists T_2 : t=T_1-T_2+d_1 \wedge T_2 \neq 0)$ — 5,A17

Axioma da quantificação.

7) $w_2 \wedge t < T_1+d_1 \rightarrow w_2 \wedge (\exists T_2 : t=T_1-T_2+d_1 \wedge T_2 \neq 0)$ — 6,PR

Argumentação proposicional com 5.

8) $w_2 \wedge t < T_1+d_1 \rightarrow (\exists T_2 : w_2 \wedge t=T_1-T_2+d_1 \wedge T_2 \neq 0)$ — T44, e T2 não ocorre livre em w_2

quantificação passa a englobar w_2.

9) $w_2 \wedge t = T_1-T_2+d_1 \rightarrow \Diamond(w_3 \wedge t \leq T_1-T_2+d_1+d_2)$ — 4,A17,PR

Instanciação de T.

10) $(t \leq T_1-T_2+d_1+d_2) \rightarrow (t \leq T_1+d_1+d_2 \wedge T_2 \geq 0)$ — A24

Afirmação matemática.

11) $w_3 \wedge t \leq T_1-T_2+d_1+d_2 \rightarrow w_3 \wedge t \leq T_1+d_1+d_2$ — 10,PR

Argumentação proposicional.

12) $w_2 \wedge t = T_1-T_2+d_1 \rightarrow \Diamond(w_3 \wedge t \leq T_1+d_1+d_2)$ — 9,11,$\Diamond$ Q

Inclusão do eventualmente.

13) $w_2 \wedge t = T_1-T_2+d_1 \wedge (T_2 \neq 0) \rightarrow \Diamond(w_3 \wedge t \leq T_1+d_1+d_2)$ — 12,PR

Permitido, pois $T_2 \geq 0$.

14) $(\exists T_2 : w_2 \wedge t = T_1-T_2+d_1 \wedge T_2 \neq 0) \rightarrow \Diamond(w_3 \wedge t \leq T_1+d_1+d_2)$ — 13,$\exists$I, e T2 não ocorre livre em w_2 ou w_3

Generalização existencial.

15) $(w_2 \wedge t \leq T_1+d_1) \rightarrow \Diamond(w_3 \wedge t \leq T_1+d_1+d_2)$ — 8,14,PR

Argumentação proposicional.

16) $w_1 \wedge t = T_1 \rightarrow \Diamond(w_2 \wedge t \leq T_1+d_1)$ — 3,A17,PR

Instanciação de T.

17) $(t \leq T_1+d_1) \leftrightarrow (t = T_1+d_1) \vee (t < T_1+d_1)$ — A17,PR,A24

Afirmação matemática.

18) $w_1 \wedge (t = T_1) \rightarrow \Diamond\left[(w_2 \wedge t = T_1+d_1) \vee (w_2 \wedge t < T_1+d_1)\right]$ 16,17,ER

Regra da equivalência.

19) $w_2 \wedge (t = T_1+d_1) \rightarrow \Diamond(w_3 \wedge t \leq T_1+d_1+d_2)$ 4,A17,PR

Instanciação de T.

20) $(w_2 \wedge t < T_1+d_1) \vee (w_2 \wedge t = T_1+d_1) \rightarrow \Diamond(w_3 \wedge t \leq T_1+d_1+d_2)$ 15,19,PR

Argumentação proposicional.

21) $w_1 \wedge (t = T_1) \rightarrow \Diamond\Diamond(w_3 \wedge t \leq T_1+d_1+d_2)$ 18,20,PR,$\Diamond$I

Inclusão do eventualmente.

22) $w_1 \wedge (t = T) \rightarrow \Diamond(w_3 \wedge t \leq T_1+d_1+d_2)$ 19,T4,ER

Regra da equivalência.

23) $((\forall T{:}w_1 \wedge t = T \rightarrow \Diamond(w_2 \wedge t \leq T_1+d_1)) \wedge (\forall T{:}w_2 \wedge t = T \rightarrow$
$\rightarrow \Diamond(w_3 \wedge t \leq T+d_2)) \rightarrow (w_1 \wedge (t = T) \rightarrow \Diamond(w_3 \wedge t \leq T_1+d_1+d_2))$ 3,4,22,PR

Argumentação proposicional.

24) $((\forall T{:}w_1 \wedge (t=T \rightarrow \Diamond(w_2 \wedge t \leq T+d_1)) \wedge (\forall T{:}w_2 \wedge t=T \rightarrow \Diamond(w_3 \wedge t \leq T+d_2))$
$\rightarrow(\forall T(w_1 \wedge (t=T) \rightarrow \Diamond(w_3 \wedge t \leq T_1+d_1+d_2))$ 23,R2

Inserção do $\forall$.

25) $\Box((\forall T{:}w_1 \wedge t=T \rightarrow \Diamond(w_2 \wedge t \leq T+d_1)) \wedge (\forall T{:}w_2 \wedge t=T \rightarrow \Diamond(w_3 \wedge t \leq T+d_2))$ 24
$\rightarrow \Box(\forall T(w_1 \wedge (t=T) \rightarrow \Diamond(w_3 \wedge t \leq T_1+d_1+d_2))$ D4($\Box$ $\Box$)

Inclusão do $\Box$.

26) $\Box((\forall T{:}w_1 \wedge t=T \rightarrow \Diamond(w_2 \wedge t \leq T+d_1)) \wedge (\forall T{:}w_2 \wedge t=T \rightarrow \Diamond(w_3 \wedge t \leq T+d_2))$ 1,2,T7

Conjunção das duas fórmulas.

27) $\Box((\forall T{:}w_1 \wedge (t=T) \rightarrow \Diamond(w_3 \wedge t \leq T+d_1+d_2))$ 25,26,R1

Modus Ponens.

C.Q.D.

Note que as diferenças básicas entre a dedução acima e a apresentada por Ostroff são devidas a introdução do operador "$\Box$" nas expressões da regra. Os teoremas T_1, T_2, T_4, T_7 e T_{44}, bem como as regras $\Diamond$I, $\Diamond$Q, $\exists$I, ER e D4, são apresentados no Apêndice, e são transcritos do Apêndice B de [23].

Apresentado o resultado acima, pode-se pensar numa generalização do mesmo, em muito semelhante à regra da cadeia generalizada apresentada em Ostroff. Sejam $w_1, w_2, \ldots, w_n$ fórmulas temporais. Então:

$$\frac{\begin{array}{l}\Box\left[(\forall T: w_1 \wedge t=T \rightarrow \Diamond(w_2 \wedge t \leq T+d_1))\right] \\ \vdots \\ \Box\left[(\forall T: w_{n+1} \wedge t=T \rightarrow \Diamond(w_n \wedge t \leq T+d_n))\right]\end{array}}{\Box\left[\forall T(w_1 \wedge (t=T) \rightarrow \Diamond(w_n \wedge t \leq T+\sum_{i=1}^{n} d_i))\right]}$$

Prova: Usando o princípio da indução finita tem-se:

para n =1 **Trivial**, pois reduz-se a uma fórmula.

para n =2 Teorema já demonstrado (DR8)

supor verdadeiro para n = K, provar para n = K + 1

1) $\Box\left[w_1 \wedge (t=T) \rightarrow \Diamond(w_K \wedge t \leq T+\sum_{i=1}^{K} d_i))\right]$ (1)

2) $\Box\left[w_K \wedge (t=T) \rightarrow \Diamond(w_{K+1} \wedge t \leq T+ d_{K+1}))\right]$ (2)

3) $\Box\left[\forall T(w_1 \wedge (t=T) \rightarrow \Diamond(w_{K+1} \wedge t \leq T+\sum_{i=1}^{K+1} d_i))\right]$ DR8,1,2

C.Q.D.

Esta regra é mais conservativa do que a regra da cadeia temporizada (TCHAIN) apresentada por Ostroff, pois naquele caso, o tempo máximo entre a primeira fórmula envolvida e a última é, no máximo, igual à soma de todos os tempos. No nosso caso, se cada tempo for exatamente d_i, o total será a soma deles. Em outras palavras a regra aqui demonstrada não prevê a desconsideração de uma das fórmulas, apresentando a análise do pior caso.

5.4. SEMÂNTICA

Enquanto a sintaxe possibilita definir uma forma precisa para a linguagem e o sistema de prova torna possível a geração de novos teoremas a partir de axiomas e teoremas pré-existentes, a semântica visa atribuir um conteúdo às fórmulas em GRTTL apresentadas. É a semântica quem faz a ligação da GRTTL com os sistemas dinâmicos a eventos discretos, pois associa os RT-BSM com as estruturas da GRTTL.

As fórmulas da linguagem serão interpretadas em função de uma estrutura S composta de uma interpretação I e de uma trajetória σ, ou seja, $S=(I,\sigma)$. Seja o RT-BSM $M=(S,E,s_0,A,P,\ell,\omega)$, então a interpretação I será dada pelo RT-BSM M. Deste modo, pode-se dizer que a estrutura S é formada por M e σ. A utilização de M como elemento que faz a interpretação é baseada na definição de $A:S \rightarrow F^*$ (onde F^* é o conjunto de todos os subconjuntos de F, o conjunto das fórmulas). Se a cada estado s está associado um conjunto de fórmulas F_s, e um estado é composto de uma valoração das constantes e variáveis (globais e locais) de todas as sortes, então o conjunto de fórmulas pode ser associado com o conjunto de variáveis e constantes do sistema, descrevendo assim os valores que estes últimos assumem através das fórmulas.

A semântica da GRTTL baseia-se em muito na descrição semântica de Lin-Ionescu [36], contendo também elementos da descrição semântica da RTTL, apresentada por Ostroff [23].

Uma interpretação M (a interpretação I dada por M) será dada por um mapeamento cujo domínio é formado pelos parâmetros. Tem-se deste modo:

1 - M atribui um domínio D_j para cada símbolo quantificador $\exists_j$ de sorte j.

2 - M atribui para cada símbolo constante de cada sorte j um elemento em D_j.

3 - M atribui para cada símbolo variável global de cada sorte j um valor em D_j.

4 - M designa para cada símbolo predicado de sorte $(j_1,j_2,j_3,...,j_n)$ um predicado $P \subset Dj_1 \times Dj_2 \times ... \times Dj_n$.

5 - M atribui para cada símbolo de função de sorte $(j_1,j_2,...,j_n)$ uma função $f: Dj_1 \times Dj_2 \times ... \times Dj_{n-1} \rightarrow Dj_n$.

Para o modelo $M=(S,E,s_0,A,P,\ell,u)$ com seqüência $\sigma = s_0 s_1 s_2 \ldots$, e para qualquer inteiro $K \geq 0$, $M\sigma^{(K)}$ representa o modelo $M=(S,E,s_{k+1},A,P,\ell,u)$ com seqüência $\sigma^{(K)} = s_K s_{K+1} \ldots$, e $M\sigma^{(0)} = M\sigma$. Na interpretação semântica, onde $S=(M,\sigma)$, denota-se a atribuição a um termo t por uma estrutura S indutivamente, da seguinte maneira:

1 - S[C] como valor atribuído ao símbolo constante C por S.

2 - S[V] como valor atribuído ao símbolo variável global V por S.

3 - $S[v] \in A(s_0)$ como valor atribuído a qualquer símbolo variável local v pelo estado inicial s_0 da trajetória σ de M.

4 - S[f] como a função atribuída ao símbolo f por S. Para qualquer símbolo de função n-ário f e quaisquer termos $t_1,\ldots,t_n$

$$S\left[f(t_1,\ldots,t_n)\right] = S[f]\left(S[t_1], S[t_2],\ldots,S[t_n]\right)$$

5 - S[P] como predicado atribuído ao símbolo P por s. Para qualquer símbolo de predicado n-ário P e quaisquer termos $t_1,\ldots,t_n$

$$S\left[P(t_1,\ldots,t_n)\right] = S[P]\left(S[t_1], S[t_2],\ldots,S[t_n]\right)$$

6 - Para qualquer termo da forma $(\odot t)$, onde t é um termo

$$S[\odot t] = S^{(1)}[t] = M\sigma^{(1)}[t]$$

Diz-se que uma fórmula-bem-formada w é satisfeita por uma estrutura S, representado por $\models^S w$, se w preeenche os seguintes requisitos:

i) $\models^S (t_1=t_2)$ se e somente se $S[t_1] = S[t_2]$ (t_1 e t_2 da mesma sorte).

ii) $\models^S P(t_1,t_2,\ldots,t_n)$ se e somente se $(S[t_1],S[t_2],\ldots,S[t_n]) \in S[P]$. Onde P é uma letra de predicado e $t_1,t_2,\ldots,t_n$ são termos.

iii) $\models^S (\neg w)$ se e somente se não é o caso de $\models^S w$.

iv) $(w_1 \rightarrow w_2)$ se e somente se $\models^S w_2$ ou não é o caso de $\models^S w_1$.

v) $\models^S (\exists V{:}w)$ se e somente se existe pelo menos uma estrutura $S'= (M',\sigma)$, tal que $\models^{S'} w$, onde M' é uma interpretação idêntica a M exceto pela atribuição que faz à variável global V, em questão.

vi) $\models^S (\odot w)$ se e somente se $\models^{S^{(1)}} w$

vii) $\models^{s}(\Diamond w)$ se e somente se existe um $K \geq 0$ tal que $S^{(K)} \models w$

viii) $\models^{s}(w_1 \mathcal{U} w_2)$ se e somente se para algum $K \geq 0$, $\models^{s}{}^{(K)} w_2$ e para todo $i \leq K$ $\models^{s}{}^{(i)} w_1$

ix) $\models^{s}(\nabla w)$ se e somente se $\tilde{p}_{s0}(\{\sigma \in w_{s0_*} \text{ e } S \models w\}) = 1$, onde $S=(M,\sigma)$, e w_{s0} é o conjunto de todas as trajetórias de S^0 que começam por s .

No item ix) acima vê-se que a satisfação de (∇w) não depende de instantes posteriores ao instante inicial, ou estado inicial. Isto demonstra que ∇ não é um operador temporal. Porém, dado o valor-verdade de w no instante considerado, não se pode saber o valor-verdade de (∇w) indicando assim que ∇ é um operador modal.

A interpretação intuitiva dos operadores temporais e do ∇ é a seguinte:

$\odot w$ - w será verdadeiro no próximo instante de tempo (ou no próximo estado).

$\Diamond w$ - w será eventualmente verdadeiro num estado futuro ou presente.

$w_1 \mathcal{U} w_2$ - w_2 será verdadeiro em algum estado futuro, e w_1 será verdadeiro (pelo menos) até lá.

∇w - com probabilidade 1, w será verdadeiro no instante atual.

Para simplificar a manipulaçao de fórmulas, algumas definições adicionais são acrescentadas ao sistema. São eles:

i)	$w_1 \vee w_2$	abrevia	$((\neg w_1) \rightarrow w_2)$	(disjunção)
ii)	$w_1 \wedge w_2$	abrevia	$\neg(w_1 \rightarrow (\neg w_2))$	(conjunção)
iii)	$w_1 \leftrightarrow w_2$	abrevia	$(w_1 \rightarrow w_2) \wedge (w_2 \rightarrow w_1)$	(bicondicional)
iv)	$(\Box w)$	abrevia	$(\neg \Diamond (\neg w))$	(daqui-por-diante)
v)	$w_1 \cup w_2$	abrevia	$(\Box w_1) \vee (w_1 \mathcal{U} w_2)$	(a-menos-que)
vi)	$w_1 \mathcal{P} w_2$	abrevia	$(\neg(\neg w_1) \mathcal{U} w_2)$	(precede)
vii)	$(\forall V{:}w)$	abrevia	$(\neg(\exists V{:}(\neg w)))$	(quantificador-universal)
viii)	$(t_1 \neq t_2)$	abrevia	$(\neg(t_1 = t_2))$	(diferente ou não-igual)
ix)	(Δw)	abrevia	$(\neg(\nabla \neg(w)))$	(é possível)

5.5. CORREÇÃO E COMPLETUDE DA GRTTL

Antes de se discutir a correção e a completude da GRTTl, alguns conceitos básicos são introduzidos, semelhantes aos apresentados em [23].

Def. 4.2. Sejam Φ e Φ_1 símbolos que denotem quaisquer conjuntos de fórmulas, e seja R uma classe de estruturas qualquer. Então tem-se:

$\models^S \Phi$ se e somente se para toda fórmula $w \in \Phi$, $\models^S w$. Diz-se que S é um modelo para Φ.

$\models^R \Phi$ se e somente se para toda estrutura $S \in R$, $\models^S \Phi$. Diz-se que Φ é R-válido. No caso onde R é a classe de todas as estruturas possíveis, Φ é dito ser válido ($\models^S \Phi$).

$\Phi_1 \models^R \Phi$ se e somente se $\models^R \Phi_1 \rightarrow \models^R \Phi$. Diz-se que Φ é uma R-conseqüência de Φ_1. No caso onde R é a classe de todas as estruturas possíveis, Φ é dito ser uma conseqüência de Φ_1 ($\Phi_1 \models \Phi$).

Def. 4.3. Um sistema de prova é **correto** se e somente se $\vdash w$ implica em $\models w$, para toda fórmula w. Em outras palavras, se uma fórmula w é um teorema (dedutível a partir do conjunto vazio), então ela é válida e é satisfeita por todas as interpretações).

Def. 4.4. Um sistema de prova é **fortemente correto** se e somente se $\Phi \vdash w$ implica $\Phi \models w$, para todo conjunto de fórmulas Φ e toda fórmula w.

Def. 4.5. Um sistema de prova é **completo** se e somente se $\models w$ implica em $\vdash w$, para toda fórmula w.

Def. 4.6. Um sistema de prova é **fortemente completo** se e somente se $\Phi \models w$ implica $\Phi \vdash w$, para todo conjunto de fórmulas Φ e toda fórmula w.

O conjunto de axiomas e regras de inferência que formam o sistema de prova da GRTTL provém basicamente do sistema de prova da RTTL [23] e do sistema de prova proposto por Lin-Ionescu [36]. Sendo assim, o sistema em questão exibe as seguintes características, quanto à correção e à completude:

1) É correto, pois os axiomas apresentados são válidos, e as

regras de inferências preservam a validade, de acordo com [23], [36] e [24].

ii) É completo, pois os sistemas em [23], [36] e [24] também o são.

iii) Não é fortemente completo. Isto se deve aos mesmo fatores expostos em [28], ou seja, não se pode deduzir de uma fórmula todas as suas conseqüências lógicas. Note que a regra $w \vdash \Box w$ não é incluída no sistema da GRTTL.

iv) O sistema é fortemente correto. Isto pode ser visto da seguinte maneira: toda a parte do sistema de prova da GRTTL que é semelhante ao Sistema da RTTL garante a correção forte, pois não se utiliza a regra $\Box$-inserção, justamente a regra que retira a correção forte da RTTL. Com relação à regra $w \vdash \nabla w$, vale salientar que o operador ∇ é modal, mas não é temporal, ou seja, é avaliado naquele estado e não sob uma trajetória. Sendo assim, para qualquer trajetória σ onde w é dedutível, tem-se: $w \vdash \nabla w$ e $w \models \nabla w$, pois w ocorre com probabilidade 1.

5.6. CONCLUSÃO

Neste capítulo a GRTTL foi apresentada formalmente, destacando-se a sintaxe, o sistema de prova e a semântica. Este formalismo pode ser usado na descrição de DEDSs, mais especificamente plantas a serem controladas, especificação em malha-fechada e controladores. A ligação dos DEDSs com a GRTTL é feita encarando-se os DEDSs como Modelos Estocásticos Limitados de Tempo Real (RT-BSM) e associando aos RT-BSMs um conjunto de fórmulas temporais. Com este ferramental (GRTTL) é possível se analisar, com uma abordagem de linguagem primal, DEDS que são não-determinísticos e possuem limitantes de tempo real. No capítulo que se segue, um exemplo é apresentado como uma aplicação prática da GRTTL, e, nos capítulos 7 e 8, são discutidos aspectos e resultados da associação da GRTTL com um ambiente de simulação.

CAPÍTULO 6

EXEMPLO: CÉLULA FLEXÍVEL DE MANUFATURA

6.1. DESCRIÇÃO DO SISTEMA

Este exemplo é uma extensão do exemplo apresentado em [36] onde são feitas algums adaptações para se ressaltar questões de tempo real.

Uma célula flexivel de manufatura consiste de duas máquinas que podem processar peças de duas classes C1 e C2, diferentes. Estas peças devem ser processadas de acordo com as seguintes regras:

R1 - Toda peça de ambas as classes C1 e C2 têm que visitar ambas as máquinas m1 e m2, não importando qual máquina seja visitada primeiro.

R2 - As peças são processadas em exclusão mútua: duas peças não podem ser processadas na mesma máquina ao mesmo tempo. Em outras palavras, peças são processadas uma por vez em cada máquina.

R3 - As peças têm a seguinte distribuição para ter acesso às máquinas: peças de classe C1 podem visitar a máquina m1 com probabilidade p_{11} e a máquina m2 com probabilidade p_{12}; peças de classe C2 podem visitar a máquina m1 com probabilidade p_{21} e a máquina m2 com probabilidade p_{22}.

R4 - Sendo considerado o tempo de processamento das duas máquinas igual a Z, o intervalo entre os processamentos da peça pelas duas máquinas, nunca poderá ser maior que 2Z.

As três primeiras regras são idênticas às apresentadas no exemplo original. A última foi incluída para termos considerações de tempo real na modelagem, especificação e síntese do controlador para o problema. A figura 10 fornece uma idéia das transições do sistema.

Na descrição dos estados tem-se:

O - A peça está fora da estação de trabalho.
W - A peça está esperando para ser processada.
P - A peça está sendo processada pela máquina 1.
Q - A peça está sendo processada pela máquina 2.
C - A peça está esperando para ser processada pela máquina 2.
D - A peça está esperando para ser processada pela máquina 1.

Na descrição dos eventos tem-se:

α_i - Chegada de uma peça.
β_i - Escolha estocástica para processamento por m1 ou m2.
γ_i - Entrada na fila de espera para processamento pela máquina m1.
ρ_i - Entrada na fila de espera para processamento pela máquina m2.
υ_i - Começo de processamento pela máquina m1.
μ_i - Começo de processamento pela máquina m2.
λ_i - Término do processamento na máquina m1 e saída da estação de trabalho.
ω_i - Término do processamento na máquina m2 e saída da estação de trabalho.

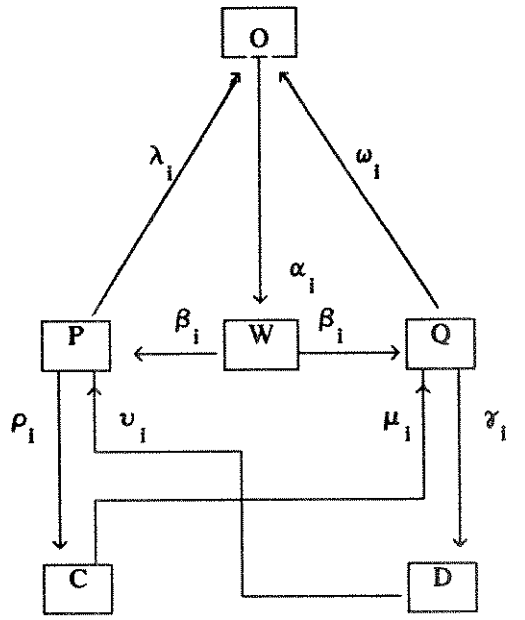


figura 10

Esta foi uma descrição informal do sistema a ser estudado nas seções que se seguem. Para um enfoque de controle, se torna muito importante conhecer quais as variáveis de controle deste conjunto de duas máquinas. O controlador não tem acesso à escolha feita pela peça. Ele só pode controlar a entrada na estação de trabalho, através da ocorrência, ou não, dos eventos β_i, ν_i e μ_i. Basicamente esta é a única ação de controle que o controlador pode tomar.

Uma questão que pode causar uma certa confusão neste exemplo, refere-se ao modo como vai-se modelar esta célula de manufatura. No intuito de ressaltar as questões ligadas à lógica temporal, toda a modelagem é feita em função das peças que serão processadas, e não em relação aos estados das máquinas.

6.2. MODELAGEM EM GRTTL

A modelagem do sistema em GRTTL consiste na modelagem já apresentada em [36] com acréscimos de considerações de tempo real. A prova das especificações de malha fechada é dada ao final.

AXIOMAS DA PLANTA

$$\text{P1)}\quad \Box[\bigvee_{i=1}^{n}(\delta=\alpha_i \vee \delta=\beta_i \vee \delta=\rho_i \vee \delta=\gamma_i \vee \delta=\mu_i \vee \delta=\upsilon_i \vee \delta=\lambda_i \vee \delta=\omega_i)]$$

Estes são os eventos possíveis.

$$\text{P2)}\quad \Box\{\bigwedge_{i=1}^{n}[\delta=\alpha_i \Rightarrow x_i=O \wedge (\odot x_i)=W \wedge (\bigwedge_{i\neq j=1}^{n}(\odot x_j)=x_j)]\}$$

Esta fórmula descreve o efeito da chegada da peça i; seu estado muda de O para W, enquanto os estados das outras peças permanecem o mesmo.

$$\text{P3)}\quad \Box\{\bigwedge_{i=1}^{n}[\delta=\beta_i \Rightarrow x_i=W \wedge [(\odot x_i)=P \vee (\odot x_i)=Q] \wedge \Delta(\odot x_i)=P \wedge \Delta(\odot x_i)=Q$$

$$\wedge (\bigwedge_{i\neq j=1}^{N}(\odot x_j)=x_j) \wedge (l_P=Z \wedge u_P=Z) \wedge (l_Q=Z \wedge u_Q=Z)]\}$$

Esta fórmula descreve as mudanças de estado motivadas pelas escolhas estocásticas da estação de trabalho. Se o evento foi β_i, o próximo estado tanto pode ser P (máquina m1) como Q (máquina m2). O tempo entre a ocorrência de β_i e ρ_i (β_i e γ_i) é de Z unidades de tempo.

$$\text{P4)}\quad \Box\{\bigwedge_{i=1}^{n}[\delta=\rho_i \Rightarrow x_i=P \wedge (\odot x_i)=C \wedge (\bigwedge_{i\neq j=1}^{n}(\odot x_j)=x_j)]\}$$

$$\text{P5)}\quad \Box\{\bigwedge_{i=1}^{n}[\delta=\gamma_i \Rightarrow x_i=Q \wedge (\odot x_i)=D \wedge (\bigwedge_{i\neq j=1}^{n}(\odot x_j)=x_j)]\}$$

P4) e P5) descrevem o efeito dos eventos ρ_i e γ_i. Se o evento for $\rho_i(\gamma_i)$, a peça termina o processamento na máquina 1(2), ou seja, sai do estado P(Q), e irá para o estado C(D). Os outros estados se mantém.

$$\text{P6)}\quad \Box\{\bigwedge_{i=1}^{n}[\delta=\mu_i \Rightarrow x_i=C \wedge (\odot x_i)=Q \wedge (\bigwedge_{i\neq j=1}^{n}(\odot x_j)=x_j) \wedge (l_Q=Z \wedge u_Q=Z)]\}$$

P7) $\Box\{\bigwedge_{i=1}^{n} [\delta=\upsilon_i \Rightarrow x_i=D \wedge (\odot x_i)=P \wedge (\bigwedge_{i\neq j=1}^{n}(\odot x_j)=x_j) \wedge (l_P=Z \wedge u_P=Z)]\}$

P6) e P7) descrevem o efeito dos eventos μ_i e υ_i. Se o evento for $\mu_i(\upsilon_i)$, a peça é direcionada para a máquina 2(1), ou seja, sairá do estado C(D) e irá para o estado Q(P). Os outros estados ficam inalterados e o tempo entre os eventos ρ_i e μ_i (γ_i e υ_i) é livre.

P8) $\Box\{\bigwedge_{i=1}^{n} [\delta=\lambda_i \Rightarrow x_i=P \wedge (\odot x_i)=O \wedge (\bigwedge_{i\neq j=1}^{n}(\odot x_j)=x_j)]\}$

P9) $\Box\{\bigwedge_{i=1}^{n} [\delta=\omega_i \Rightarrow x_i=Q \wedge (\odot x_i)=O \wedge (\bigwedge_{i\neq j=1}^{n}(\odot x_j)=x_j)]\}$

P8) e P9) descrevem o efeito dos eventos λ_i e ω_i. Se o evento for $\lambda_i(\omega_i)$, a peça sairá da máquina 1(2), ou seja, sairá do estado P(Q) e irá para o estado O. Os outros estados permanecem inalterados e o tempo entre os eventos υ_i e λ_i (μ_i e ω_i) é livre.

P10) $\Box[O\neq W \wedge O\neq P \wedge O\neq Q \wedge O\neq C \wedge O\neq D \wedge W\neq P \wedge W\neq Q \wedge W\neq C \wedge W\neq D \wedge P\neq Q \wedge$

$\wedge P\neq C \wedge P\neq D \wedge Q\neq C \wedge Q\neq D \wedge C\neq D]$

Os estados são distintos.

P11) $\bigwedge_{i=1}^{n} x_i=O.$

Todas as peças estão fora da estação de trabalho no estado inicial.

ESPECIFICAÇÕES DE MALHA FECHADA EM GRTTL

MF1) $\Box[\bigwedge_{i=1}^{n}(x_i=P \Rightarrow \bigwedge_{i\neq j=1}^{n} x_j\neq P)]$

MF2) $\Box[\bigwedge_{i=1}^{n}(x_i=Q \Rightarrow \bigwedge_{i\neq j=1}^{n} x_j\neq Q)]$

As peças são processadas uma por vez em cada máquina.

MF3) $\Box[\bigwedge_{j=1}^{n}(\bigvee_{i\neq 1}^{n}(x_i=C \vee x_i=D)=>\delta\neq\beta_j)]$

As escolhas estocásticas não são feitas se existem peças-pf esperando para serem processadas na segunda máquina. Em outras palavras, peças-pf têm prioridade. As peças-pf são aquelas que já foram processadas uma vez por qualquer uma das máquinas.

$$MF4)\ \Box[\bigwedge_{\substack{i=1\\ j=1\\ i\neq j}}^{n}(x_i\neq C \Rightarrow ((\delta=\rho_i\ \mathcal{P}\ \delta=\rho_j) \Rightarrow (\delta=\mu_i\ \mathcal{P}\ \delta=\mu_j)))]$$

$$MF5)\ \Box[\bigwedge_{\substack{i=1\\ j=1\\ i\neq j}}^{n}(x_i\neq D \Rightarrow ((\delta=\gamma_i\ \mathcal{P}\ \delta=\gamma_j) \Rightarrow (\delta=\upsilon_i\ \mathcal{P}\ \delta=\upsilon_j)))]$$

As assertivas acima garantem que as peças já processadas uma vez (peças-pf) são reprocessadas na ordem em que chegam: Se a próxima chegada da peça *i* precede a próxima chegada da peça j, então a peça *i* é enviada antes da peça j.

$$MF6)\ \Box[((\bigwedge_{i=1}^{n}(x_i=O \vee x_i=W)) \wedge \neg(\bigwedge_{k=1}^{n}x_k=O)) \Rightarrow \bigvee_{j=1}^{n}\delta=\beta_j]$$

Se todas as peças estão fora da estação ou esperando para serem processadas, mas nem todas estão fora da estação, então um β_j deve ocorrer.

$$MF7)\ \Box[\wedge((\delta=\gamma_i)\wedge t=T)\Rightarrow(\Diamond(\delta=\upsilon_i)\wedge t\leq T+2Z)]$$

$$MF8)\ \Box[\wedge(((\delta=\rho_i)\wedge t=T)\Rightarrow(\Diamond(\delta=\mu_i)\wedge t\leq T+2Z)]$$

Se uma peça saiu do processamento de qualquer uma das máquinas e vai para a fila de espera de reprocessamento, ela é reprocessada em , no máximo, 2Z unidades de tempo.

ESPECIFICAÇÃO DO CONTROLADOR

Alguns símbolos adicionais são usados para representar os dados armazenados pelo controlador. Os símbolos p e q representarão as filas para as máquinas m2 e m1, respectivamente. A cada elemento destas filas associa-se inteiros de 1 a N , ou o símbolo ϵ que denota fila vazia. O símbolo variável local $\underline{c}$ representa o número de peças na estação de trabalho que não foram ainda processadas (não são peças-pf). Ele assume valores inteiros não negativos.

Além dos símbolos acima citados, algumas operações sobre listas são representadas através dos seguintes símbolos:

1) "*" denota a inclusão de um elemento numa fila (⊙p=p*i significa que o elemento i é colocado na fila p).

2) "$|^1$" denota a retirada de um elemento na cabeça de uma fila. ($p|^1$ significa que foi retirado um elemento no topo de p.

3) "<" denota a comparação de um elemento com a cabeça de uma fila. (i< p simboliza que o elemento i é igual ao da cabeça de p.

O controlador esboçado em [36] garante que as equações de malha fechada (teoremas) MF1 a MF6 sejam satisfeitas. Porém sobre as questões de tempo real introduzidas neste trabalho e exigidas em MF7 e MF8, nada se garante. Sendo assim, necessário se faz acrescentar considerações ao controlador para que o mesmo possa satisfazer os requisitos de malha fechada. Isto é feito criando-se um partilhamento entre controlador e a planta dos eventos relativos à saída da peça do primeiro processamento na máquina M1 (M2) e dos eventos relativos à saída da peça do segundo processamento na máquina M2 (M1). O detalhe é que para o controlador estas transições terão um limite inferior de Z e um limite superior de 2Z unidades de tempo. A figura 11 ilustra o diagrama de estados do controlador.

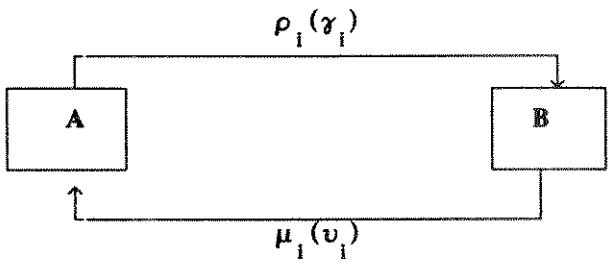


figura 11

A variável Yi do controlador é introduzida. Ela sincroniza o início do segundo processamento. Quando o evento ρ_i ou γ_i ocorre, a variável Y_i vai para o estado **B**. A partir deste estado, o controlador impõe que o evento partilhado μ_i ou υ_i ocorra entre Z e 2Z unidades de tempo. Isto garante que as especificações MF7 e MF8 sejam cumprida.

As equações do controlador são:

C1) $\Box[\bigwedge_{i=1}^{n}(\delta=\rho_i=>((Y_i=A) \wedge (\odot Y_i=B) \wedge (\odot p)=p^*i \wedge (\odot c)=c-1)) \wedge$

$l_B = Z \wedge u_B = 2Z)]$

C2) $\Box[\bigwedge_{i=1}^{n}(\delta=\gamma_i=>((Y_i=A) \wedge (\odot Y_i=B) \wedge (\odot p)=p^*i \wedge (\odot c)=c-1))]$

$l_B = Z \wedge u_B = 2Z)]$

C1 (C2, respectivamente) diz que se uma peça-pf, i, vem da máquina m1(m2), o número i é colocado na fila p(q), e o valor do contador c é decrementado de 1. Note as cotas inferiores e superiores de tempo.

C3) $\Box[\bigwedge_{i=1}^{n}((\delta=\mu_i)=>((Y_i=B) \wedge (\odot Y_i=A) \wedge i < p \wedge (\odot p)=p \wedge (\odot c)=c)]$

C4) $\Box[\bigwedge_{i=1}^{n}((\delta=\upsilon_i)=>((Y_i=B) \wedge (\odot Y_i=A) \wedge i < p \wedge (\odot p)=p \wedge (\odot c)=c)]$

C3 (C4, respectivamente) diz que se uma peça-pf é enviada para a máquina m2 (m1) só se ela está na cabeça da fila p(q). Os conteúdos da fila e do contador c não se alteram. A variável de estado do controlador muda de B para A.

C5) $\Box[\bigwedge_{i=1}^{n}(\delta=\omega_i=>(\odot p)=p|^1 \wedge (\odot c)=c)]$

C6) $\Box[\bigwedge_{i=1}^{n}(\delta=\lambda_i=>(\odot q)=q|^1 \wedge (\odot c)=c)]$

C5 (C6, respectivamente) diz que se uma peça sai da máquina m2 (m1), a fila p(q) é diminuída do primeiro elemento e o valor do contador fica inalterado.

C7) $\Box[\bigwedge_{i=1}^{n}(\delta=\beta_i=>(q=\epsilon \wedge (\odot q)=q \wedge p=\epsilon \wedge (\odot p)=p \wedge (\odot c)=c))]$

C7) estabelece que uma escolha estocástica é feita só se as filas p e q estão vazias. Quando esta escolha ocorre, o valor do contador e o conteúdo das filas permanecem inalterados.

C8) $\Box[\bigwedge_{i=1}^{n}(\delta=\alpha_i=> \wedge (\odot q)=q \wedge (\odot p)=p \wedge (\odot c)=c+1)]$

A expressão acima atesta que se uma peça chega na estação, o valor do contador c é incrementado de 1. Os conteúdos das filas p e q permanecem inalterados.

C9) $q=\epsilon \wedge^{n} p=\epsilon \wedge c=0 \ (\wedge_{i=1}^{n} Y_i = A)$

Esta equação simplemente fornece o estado inicial do controlador. No início, as filas p e q estão vazias, o contador possui o valor zero e os estados das variáveis Y_i são inicializados como A, que significa o controle do tempo de espera desativado.

Sendo assim termina-se a descrição formal do controlador. A prova de que as especificações de malha fechada são dadas a seguir.

PROVA DAS ESPECIFICAÇÕES DE MALHA FECHADA

Como já dito num parágrafo anterior, as equações MF1 a MF6 de malha fechada são satisfeitas trivialmente pelo conjunto de fórmulas P1 a P11 e C1 a C9. Note que os acréscimos feitos às assertivas do controlador em nada atrapalham a satisfação de MF1 a MF6. Resta provar as equações MF7 e MF8. Apresenta-se abaixo um esboço da prova.

PROVA DE MF7 E MF8

Como as equações são semelhantes, a prova será feita para MF7 $\wedge$ MF8.

$$\Box[(\delta=\rho_i \wedge t=T) \Rightarrow \Diamond(\delta=\mu_i \wedge t\leq T+2Z)] \quad [1]$$

Definição de u_B em MF7.

$$\Box[(\delta=\gamma_i \wedge t=T) \Rightarrow \Diamond(\delta=\nu_i \wedge t\leq T+2Z)] \quad [2]$$

Definição de u_B em MF8.

$$\Box[(\delta=\rho_i \wedge t=T) \Rightarrow \Diamond(\delta=\mu_i \wedge t\leq T+2Z)] \wedge \Box[(\delta=\gamma_i \wedge t=T) \Rightarrow \Diamond(\delta=\nu_i \wedge t\leq T+2Z)] \quad [3]$$

Argumentação proposicional com [1] e [2].

$$\Box[((\delta=\rho_i \wedge t=T) \Rightarrow \Diamond(\delta=\mu_i \wedge t\leq T+2Z)) \wedge ((\delta=\gamma_i \wedge t=T) \Rightarrow \Diamond(\delta=\nu_i \wedge t\leq T+2Z))] \quad [4]$$

Teorema T7 aplicado a [3]. E temos a equação MF7 $\wedge$ MF8.

c.q.d.

É interessante notar a facilidade da prova da assertivas MF7 e MF8 utilizando o conceito de cota superior.

CAPÍTULO 7

ASPECTOS DE SIMULAÇÃO

7.1. INTRODUÇÃO

A GRTTL é um formalismo desenvolvido para a verificação e desenvolvimento de controladores para Sistemas Dinâmicos a Eventos Discretos (DEDS). Esta metodologia constrasta com a aplicação da simulação no estudo de DEDSs em alguns aspectos. No estudo baseado em simulação, existe uma grande flexibilidade na adaptação dos controladores aos sistemas a serem controlados. A solução é encontrada num processo de tentativa-e-erro, onde o comportamento do sistema é simulado um grande número de vezes até que se tenha observado o mesmo na maioria dos casos relevantes. Este tipo de solução tem o inconveniente de ser não sistematizável, pela falta de um arcabouço teórico para a mesma.

Para a utilização da GRTTL, ou um método analítico-formal de solução, a vantagem está justamente na maneira formal como os resultados são obtidos. Uma vez provado que um determinado controlador satisfaz as especificações de malha-fechada de um DEDS, não há mais necessidade de maiores investigações. Por outro lado, não raras vezes, os sistemas a serem controlados são tão complexos, e sua descrição em GRTTL tão intrincada, que a prova formal destes resultados fica difícil, senão impossível. Além disto, não se tem uma maneira clara de como passar de uma descrição em fórmulas de um controlador, em GRTTL, para um programa que implemente suas funções. Objetivando uma solução conciliatória, apresenta-se neste capítulo um estudo de uma associação entre a GRTTL e o "Sistema de Simulação Baseado em Conhecimento" (SSBC) desenvolvido no DCA/FEE/UNICAMP [39]. A utilização da GRTTL e do SSBC em conjunto visa fornecer um ambiente de simulação que tenha as vantagens da GRTTL, enquanto ferramenta formal, e as facilidades do SSBC, enquanto ferramenta de simulação.

É importante observar que existem **requisitos** para que uma modelagem numa lógica temporal genérica seja compatibilizada com um simulador também genérico. Mostra-se que a GRTTL e o SSBC cumprem tais requisitos. Cumpridos os requisitos, faz-se um **mapeamento** das características da GRTTL para aquelas do SSBC. Por fim, apresenta-se uma **metodologia** para, de uma maneira clara, transformar as especificações em GRTTL para um modelo de simulação compatível com o SSBC. Antes de apresentar os requisitos, o mapeamento e a metodologia envolvidos nesta transformação, o Sistema de Simulação Baseado em Conhecimento (SSBC) é apresentado com algum detalhe. Ao final, alguns exemplos são dados para ilustrar o método.

7.2. SISTEMA DE SIMULAÇÃO BASEADO EM CONHECIMENTO (SSBC)

O Sistema de Simulação Baseado em Conhecimento (SSBC) é uma ferramenta responsável por oferecer serviços de modelagem, execução e análise de resultados de simulação, para sistemas Dinâmicos a eventos discretos. Sua concepção funcional é baseada no paradigma de objetos, privilegiando características como modularidade, flexibilidade, hierarquização funcional e configurabilidade [40].

O domínio de aplicação do SSBC compreende os sistemas passíveis de descrição por eventos discretos. A adoção de técnicas de inteligência artificial, junto com a utilização de uma linguagem de processamento simbólico (PROLOG) permite um alto grau de flexibilidade na concepção tanto do modelo de informação adotado como da configuração do simulador [40].

O modelo de informação adotado caracteriza-se por adotar uma representação do conhecimento através de uma tripla formada por **Frames**, **Regras de Produção** e **Procedimentos**.

Frames são estruturas organizadas como uma coleção de slots. Cada slot corresponde a um tipo de informação relacionada com a entidade em descrição. A informação associada a cada slot é interna à entidade e bloqueada ao conhecimento de outras entidades, sendo que, se necessário, outros frames podem ser mencionados através de indicação explícita.

Regras de Produção são usadas de forma a representar condicionantes tanto do processo de simulação quanto do comportamento de entidades a serem simuladas.

Procedimentos são usados para definir comportamentos específicos, ou métodos da entidade à qual se encontram associados.

A flexibilidade advinda das técnicas de inteligência artificial e do processamento simbólico reflete-se na composição do modelo de simulação de cada entidade a ser considerada na simulação e na composição do sistema a ser simulado.

Cada entidade a ser simulada pode ser modelada a partir de um conjunto de informações genéricas (default) que a descrevem, permitindo-se

anexar novas informações, configuráveis, pelo usuário. As regras de produção e os procedimentos relacionados com cada entidade são compostos pelo usuário a partir de uma biblioteca básica de operadores e procedimentos, podendo-se também definir novos operadores de procedimentos.

O sistema completo a ser simulado é composto pela indicação de todas as entidades que o compõe e pela explicitação de todas as interações existentes entre estas entidades. Deve-se também explicitar todas as regras que condicionam a operação conjunta das entidades.

O modelo genérico (default) usado para a descrição de cada entidade a ser simulada, é composto pelos seguintes atributos característicos: eventos, sequenciamento de eventos, temporalidade das transições, condicionantes e elementos de comunicação física e de controle. O conjunto de eventos corresponde à indicação e descrição de todos os eventos passíveis de ocorrerem internamente à entidade. É importante notar que durante o processo de modelagem, uma atividade pode ser descrita através dos eventos início e fim de atividade. A cada evento indicado corresponde um número identificador do mesmo.

EVENTO - Como descrição do evento entende-se a composição de todos os procedimentos a ele associados. A execução de um evento corresponde ao disparo de todos os procedimentos a ele associados, provocando todas as alterações de características da entidade, correspondendo à mudança de estado referente ao evento.

SEQUENCIAMENTO DE EVENTOS - Corresponde à indicação da sequência em que os eventos da entidade ocorrem. A este sequenciamento dá-se o nome de rotina, podendo ser descritas várias rotinas para uma mesma entidade. Trechos de sequenciamento compulsório de eventos são simbolizados pelo operador ">" entre os indicadores de eventos subsequentes. Quando após a ocorrência de um dado evento podem haver vários eventos subsequentes, esta multiplicidade é indicada pelo operador "/". A ocorrência deste operador remete a lógica de simulação para um conjunto de regras que , avaliada pela máquina de inferência do sistema, indicará o evento a ser assumido como subsequente.

TEMPORALIDADE DAS TRANSIÇÕES - Corresponde à indicação das condicionantes de tempo entre a ocorrência de eventos subsequentes. A temporalidade é expressa na descrição de cada evento, indicando-se o tempo que deve decorrer entre a execução de um evento e de seu subsequente. Sua

indicação pode seguir uma dentre as seguintes distribuições: constante, aleatória, normal e exponencial. Quando da não explicitação de temporalidade na ocorrência do próximo evento, assume-se a situação default de possibilidade de ocorrência imediata para o próximo evento.

CONDICIONANTES - São os conjuntos de regras a serem avaliados pela máquina de inferência. Esta regras tomam o formato de regras de produção (se <condições> então <ações>). Podem existir conjuntos de regras que condicionem a ocorrência de cada evento, o sequenciamento de eventos (decisão de fluxo) e o interrelacionamento entre entidades. Tanto as condições como as ações são compostas a partir do agrupamento de procedimentos, pertinentes à biblioteca ou definidos pelo usuário, atuando sobre variáveis a serem definidas pelo usuário, durante o processo de modelagem.

ELEMENTOS DE COMUNICAÇÃO FÍSICA E DE CONTROLE - São elementos de modelagem que representam a comunicação física ou comunicação de controle entre entidades. São geralmente representados por portas, flags, contadores e filas.

ESTADO INICIAL - No SSBC o estado inicial é representado num frame especial, onde são armazenados os valores iniciais de variáveis, filas, portas e contadores referentes ao sistema.

A composição do sistema a ser simulado é feita indicando-se quais as entidades que o formam e o conjunto de regras que condicionam a ocorrência dos eventos das entidades frente o estado global do sistema. A execução da simulação [40] é efetuada através dos métodos pertinentes a cada bloco lógico do SSBC, podendo ser classificada como adotando a estratégia de interação de processos.

Os blocos lógicos que compõem a execução da simulação são: **Controlador de Simulação, Ordenadores Lógicos, Monitor de Entidades** e **Manipulador de Conhecimento.**

O **Controlador de Simulação** controla o relógio de simulação e os procedimentos da lógica de simulação, como a execução dos eventos.

Os Ordenadores Lógicos escolhem, para cada entidade, qual o próximo evento a ser executado. Esta escolha é feita através de inferência sobre o

sequenciamento de eventos da entidade e sobre seus condicionantes (regras). Note-se que a este nível avalia-se cada entidade de forma independente do sistema.

O Monitor de Entidades escolhe quais os eventos que podem ser executados imediatamente, dentre aqueles apresentados pelas entidades. Note-se que para esta escolha são avaliadas as regras de interrelacionamento entre entidades e o estado global do sistema. Após esta escolha o monitor apresenta estes eventos para serem executados pelo controlador de simulação.

O Manipulador de Conhecimento [40] é o responsável por todo o processo de manipulação de Frames, regras e procedimentos. Esta manipulação é feita após o recebimento de mensagens provenientes dos outros blocos lógicos que compõem o SSBC, sendo também o resultado desta manipulação enviado através de mensagens. Toda a manipulação de regras é feita por um sub-módulo que consiste em uma máquina de inferência executando encadeamento reverso.

O modelo funcional da execução da simulação é representado pela figura abaixo:

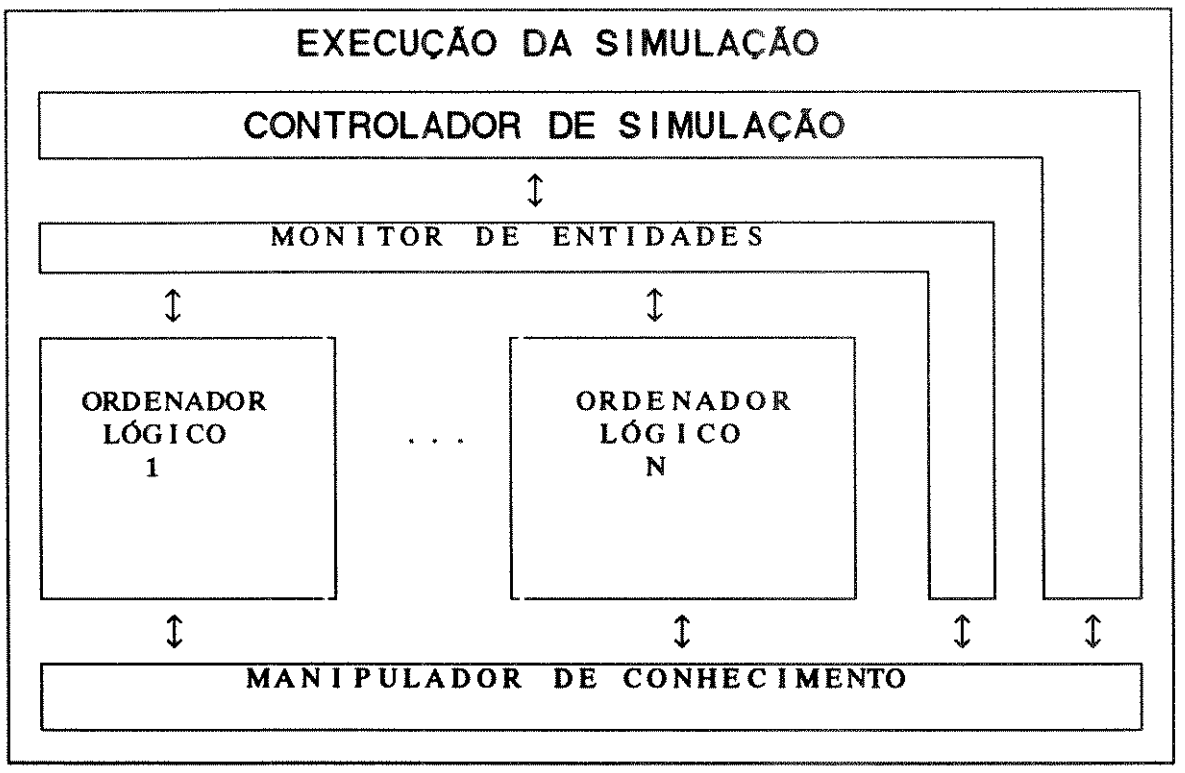


figura 12

7.3. REQUISITOS PARA TRANSFORMAÇÃO DE GRTTL PARA SSBC

A transformação de especificações em lógica temporal (L.T.) para modelos de simulação deve obedecer a alguns requisitos para que se garanta uma relação clara entre estes. Estes requisitos dependem tanto da L.T. usada quanto do simulador-alvo. Dependendo do tipo de L.T. e do simulador escolhidos, esta transformação se processa com um grau variável de complexidade. A princípio, toda especificação realizada em uma lógica temporal linear pode ser manipulada de forma a gerar um modelo de simulação orientado a evento. Neste trabalho restringe-se a discussão ao relacionamento entre especificações em uma GRTTL, descrita no capítulo 5, e o SSBC, descrito na seção 7.2.

Um outro aspecto importante deste relacionamento corresponde ao sistema a eventos discretos a ser estudado. Uma característica desejável para o estudo de sistemas de controle é a separação clara entre as especificações de planta e controlador. Esta característica facilita a análise do comportamento da planta sob a atuação de diversos controladores. Esta **separação** também facilita a divisão de sistemas complexos em sub-sistemas, simplificando o estudo, devido a **modularidade** atingida. Tanto a GRTTL quanto o SSBC anteriormente descritos são compatíveis com as características de **separação** e **modularidade** desejadas para sistemas de controle.

Algumas características do GRTTL e do SSBC são de grande valia na transformação das especificações do primeiro para o segundo. São elas:

CARACTERÍSTICAS DA GRTTL

As características mais importantes nas especificações em GRTTL tratadas neste trabalho são:

1) **Características que permitem a descrição da evolução do sistema ao longo do tempo** - São representados por operadores temporais ($\odot, \Diamond, \Box, P, U$, etc) e sua axiomatização.

2) **Características que permitem a descrição de condicionantes de tempo real** - Neste ítem tem-se prazos. Isto é normalmente descrito via limites inferiores e superiores de tempo e a axiomatização destes conceitos.

3) **Características que permitem a descrição do não-determinismo de sistemas** - A noção de não-determinismo está associada com a idéia da não observabilidade da ocorrência de algumas (ou todas) as transições a partir de um estado, em contraste com a noção de determinismo.
É interessante notar que não se tem um tratamento estocástico (média, variância, etc), muito embora seja não-determinístico. O operador modal "∇", e seu dual "Δ", se encarregam de representar o não-determinismo nas equações em GRTTL.

4) **Características que permitem a descrição do estado inicial** - Na lógica temporal uma das assertivas fornece os estados iniciais das variáveis, filas, contadores, etc.

CARACTERÍSTICAS DO SSBC

Para simulação de sistemas modulares descreve-se as entidades que os compõem e a interação entre entidades. Na descrição de entidades as principais características são:

1 - Indicação e descrição de seus eventos.
2 - Sequenciamento dos eventos.
3 - Limitantes de tempo real.
4 - Condicionantes internos para a ocorrência de eventos.
5 - Descrição de elementos de comunicação física e de controle.
6 - Representação do estado inicial do sistema.

Na descrição da interação entre entidades as principais características são:

1 - Interligação entre elementos de comunicação física e de controle das entidades.

2 - condicionantes mútuas para a ocorrência de eventos entre duas ou mais entidades.

Uma vez respeitadas as características acima citadas para a GRTTL e para o SSBC, é possível realizar o mapeamento de características do primeiro para o segundo sistema.

7.4. MAPEAMENTO DAS CARACTERÍSTICAS DE GRTTL PARA SSBC

Uma vez feita a descrição de um sistema em GRTTL, conforme descrito nos capítulos anteriores, há a necessidade do mapeamento desta descrição para um modelo de simulação. Apresenta-se a seguir um esboço das principais relações entre os elementos usados para a modelagem nos dois formalismos:

i) **ENTIDADES** - Um modelo de simulação para o SSBC requer a caracterização de entidades físicas, como descrito na seção 3. Especificações em GRTTL geralmente não explicitam a caracterização de entidades, tornando o mapeamento não direto. No entanto, é possível reconhecer na descrição dos estados em GRTTL quais sejam os elementos físicos que compõem o sistema em análise. Desta forma distinguem-se as entidades a partir do conjunto de estados.

Um fato interessante observado é que geralmente o controlador da planta não figura como uma entidade, e sim, como um conjunto de regras que norteia a execução da simulação e a interação entre as demais entidades. Isto se deve a um detalhe de implementação do Sistema de Simulação Baseado em Conhecimento. No SSBC, tem-se uma máquina de inferência interna que, efetivamente, executa o controle. Sendo assim, só é necessário especificar o que precisa ser controlado, o que é feito através das regras.

ii) **EVENTOS** - O conjunto de eventos da especificação em GRTTL é mapeado diretamente para o conjunto de eventos que compõe os modelos das entidades. Relacionado com a ocorrência de cada evento existe um conjunto de procedimentos. Na especificação em GRTTL estes procedimentos são expressos pelos consequentes atribuidos a ocorrência de cada evento, excetuando-se a transição de estados.

iii) **SEQUENCIAMENTO DE EVENTOS** - O sequenciamento de eventos pode ser obtido das especificações em GRTTL a partir do sequenciamento das transições de estado motivadas pelos eventos. Nesta etapa são considerados os significados dos operadores temporais sobre a dinâmica da planta. O não-determinismo no sequenciamento de eventos (não observabilidade de eventos a partir

de um determinado estado) pode ser descrito por especificações em GRTTL utilizando-se os operadores ∇ e Δ já mencionados. Estas especificações podem ser diretamente mapeadas para as ramificações de sequenciamento de uma rotina no modelo de simulação. Tais ramificações são representadas na rotina pelo operador "/", descrito na seção 3.

iv) **TEMPORALIDADE DAS TRANSIÇÕES** - O conceito de temporalidade expressa os limites inferior e superior do intervalo de tempo entre a ocorrência de dois eventos determinados. Estes limites são as cotas inferior e superior de tempo expressas em GRTTL em função dos estados. Assim o mapeamento é feito diretamente.

v) **CONDICIONANTES** - Os condicionantes para habilitação de um evento no modelo de simulação podem ser obtidos do conjunto de fórmulas das especificações em GRTTL. Isto pode ser feito eliminando-se as informações de sequenciamento das fórmulas relacionadas com o evento a ser condicionado.

vi) **ELEMENTOS DE COMUNICAÇÃO FÍSICA E DE CONTROLE** - Os elementos de comunicação física são, a príncipio, as filas e os contadores. A critério do modelador e dependendo da aplicação, estes podem ser usados para apresentar informações adicionais a respeito do estado atual de uma entidade. Estes mesmos elementos, adicionados aos sinais de controle (flags), são usados como elementos de controle. As informações que caracterizam estes elementos são obtidas das fórmulas do controlador nas especificações em GRTTL.

vii) **INTERAÇÃO ENTRE ENTIDADES** - Nas especificações em GRTTL, a interação entre entidades não é identificada de forma direta. Isto se deve ao fato das entidades, conforme entendidas no SSBC, não serem explicitadas diretamente nas especificações em GRTTL. No modelo de simulação esta interação pode ser caracterizada através da 1) interligação entre os elementos de comunicação física e de controle e 2) das condicionantes mútuas entre entidades.

A interligação 1) não acrescenta novas informações às já

obtidas acima (ítem *vi*), apenas relacionando canais de comunicação inter-entidades e não intra-entidades.

As condicionantes mútuas entre entidade, 2), são obtidas das especificações do controlador, na parte onde os eventos de planta são condicionados. Normalmente trata-se de variáveis, filas, flags e portas comuns às entidades.

viii) **ESTADO INICIAL** - O mapeamento da equação que descreve o estado inicial do conjunto **planta-controlador** corresponde à criação de um Frame especial no modelo de simulação, denominado Frame de Condições Iniciais, que indica o estado inicial de cada uma das variaveis, filas, contadores e portas do modelo.

7.5. METODOLOGIA PARA TRANSFORMAÇÃO DE ESPECIFICAÇÕES EM GRTTL EM MODELOS DE SIMULAÇÃO

Para a transformação de especificações em GRTTL para modelos de simulação, considera-se que estas especificações são apresentadas como a seguir:

A) ESPECIFICAÇÕES DA PLANTA

Como especificações de planta entende-se equações que apresentem:

ESTADOS - Estes estados são apresentados como constantes. Normalmente uma só equação basta para isto.

EVENTOS - Os eventos possíveis são apresentados como constantes. Este conjunto é normalmente descrito por uma só equação.

TRANSIÇÕES - As equações de transição correspondem à maior parte das fórmulas do conjunto de especificações da planta. Neste conjunto encontram-se informações sobre as transições de estado, condicionantes de tempo, sequenciamento de eventos e as consequências de cada transição.

B) ESPECIFICAÇÕES DO CONTROLE

Como especificações de controle entende-se equações que apresentem:

* condições impostas para a ocorrência dos eventos de planta,
* descrição das ações de controle tomadas para cada evento ocorrido.

Nestas especificações, normalmente aparecem variáveis locais adicionais, usadas para implementar a lógica de controle (contadores, filas, flags, etc.). Os passos a serem seguidos para a transformação citada são:

1) O usuário deve examinar as equações de planta (e controlador) e determinar quais as entidades a serem consideradas para a simulação.

2) Determinadas as entidades, conforme o ítem acima, o usuário deve associar a cada uma os respectivos estados e eventos especificados. Vinculado à ocorrência de cada evento, existe um conjunto de procedimentos que fornece as ações a eles correspondentes. Existem procedimentos relacionados exclusivamente aos eventos ou às condicionantes (regras) de controle.

3) O sequenciamento de eventos para cada entidade pode ser feito acompanhando-se as transições de estado. Dado um evento e um estado inicial, pode-se saber qual o próximo estado e, consequentemente, o(s) próximo(s) evento(s). Deste modo, sucessivamente, obtém-se todo o sequenciamento.

4) A temporalidade das transições é obtida a partir dos limites inferiores e superiores de tempo para a transição de estado correspondente a ocorrência de cada evento. Estas informações são reconhecidas a partir da comparação dos valores da variável tempo, "t", do antecedente e consequente das fórmulas em GRTTL. Nestas fórmulas, os limites inferior e superior de tempo são obtidos das expressões ($t \geq T + lim.inf$) e ($t \leq T + lim.sup$) respectivamente, onde "T" é o valor da variável "t" no instante que a transição ocorre.

5) Os condicionantes são obtidos dos consequentes das fórmulas do controlador excetuando-se as expressões (termos) de ação. Estas expressões (termos) de ação são reconhecidas através do símbolo ⊙, next, do símbolo * , inserção de elemento em cauda de lista, e do símbolo |, retirada de elemento da cabeça de lista. Todos estes elementos operam sobre as variáveis de controle (contadores, listas, flags, etc). Também observa-se aqui o aparecimento de regras que controlam o funcionamento da planta, mas que são internas à mesma.

6) Elementos de comunicação física e de controle - A transformação destes elementos é imediata das especificações em GRTTL para o modelo de simulação, pois são as filas, contadores e flags das especificações em GRTTL. Em casos onde se deseje informações adicionais (estado atual, etc) pode-se utilizar as portas como registros.

7) A interação entre entidades é representada, no modelo de simulação, através da indicação da igualdade de conteúdo de filas, portas, contadores e flags que interligam entidades distintas.

8) As condições iniciais são obtidas da fórmula em GRTTL que traz esta informação e armazenada num Frame próprio no SSBC. A passagem é direta.

7.6. EXEMPLOS

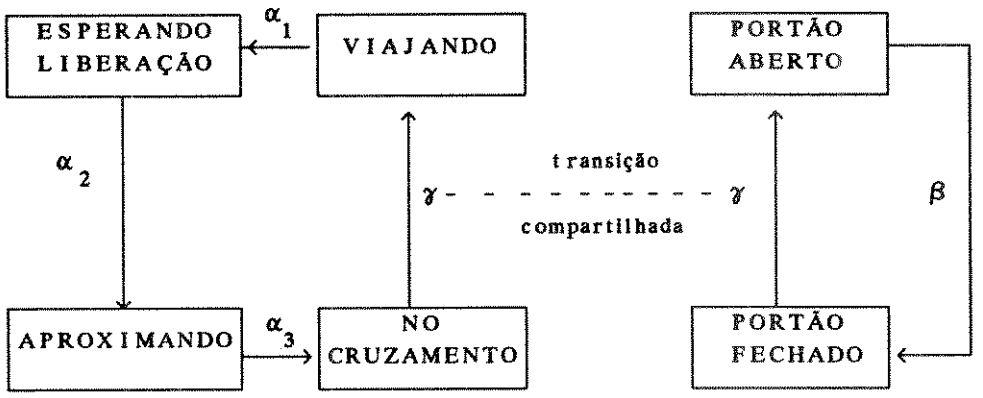
Nesta seção são estudados dois exemplos simples, porém bastante elucidativos sobre a aplicação da GRTTL na modelagem e simulação de sistemas a eventos discretos. Inicialmente, os sistemas são descritos informalmente. Em seguida, apresentam-se equações em GRTTL para: descrição da planta (sistema a ser controlado), especificações de malha fechada (condições que a planta deveria satisfazer) e controlador (aquele que faz a planta cumprir as especificações de malha fechada).

O primeiro exemplo é uma adaptação do sistema TREM-PORTÃO descrito

em [35]. O segundo exemplo é o mesmo que o apresentado no capítulo 6.

A) SISTEMA TREM-PORTÃO

Um portão num cruzamento ferroviário controla a passagem de veículos, ou pedestres, pelo mesmo. À medida que um trem se aproxima do entroncamento, a portão deve baixar em tempo hábil para prevenir acidentes. É interessante notar que neste caso a planta é composta da composição paralela de dois subsistemas: o trem e o portão. O controle elaborado deve decidir quando o portão deve baixar, em função da disposição do trem. O diagrama a seguir dá uma idéia mais clara do sistema.



Este exemplo é praticamente o mesmo que o encontrado em [35]. A primeira alteração feita corresponde à criação de um estado intermediário entre os estados de "viajando" e "aproximando" que representa a retenção do trem num certo trecho da ferrovia até que haja permissão para o mesmo seguir seu caminho até o cruzamento. A outra alteração corresponde à ferramenta de modelagem utilizada para a planta e o controlador. Em [35], usa-se o formalismo TTM/RTTL, enquanto que neste trabalho só a GRTTL basta para a modelagem.

Por razões de simplicidade, as tarefas de sincronizar a abertura do portão e a passagem do trem fazem parte da dinâmica da planta. A verificação da existência ou não de trem no cruzamento também. Sendo assim, a única tarefa pertinente ao controlador conrresponde realmente à sincronização do fechamento do portão com a chegada de um trem. Isto foi feito com a finalidade de concentrar toda a atenção na questões de tempo real envolvidas neste exemplo.

Modelagem em GRTTL

A modelagem em lógica temporal segue como já visto no capítulo 5. Primando pela simplicidade das expressões em GRTTL, as seguintes abreviações são feitas:

ESTADOS

Viajando	V
Esperando liberação	E
Aproximando	A
No cruzamento	C
Portão aberto	PA
Portão fechado	PF

AXIOMAS DA PLANTA

$$P1)\Box[\delta=\alpha_1 \vee \delta=\alpha_2 \vee \delta=\alpha_3 \vee \delta=\gamma \vee \delta=\beta]$$

Esta fórmula descreve o conjunto de estados possíveis.

$$P2)\ \Box[\delta=\alpha_1 \Rightarrow ((x1=V) \wedge (\odot x1=E) \wedge (\odot x2=x2))]$$

Esta equação diz que se ocorre o evento α_1 o trem passa do estado "V" para o estado "E". É interessante notar que a variável local correspondente ao portão, permanece com o mesmo valor.

$$P3)\Box[\delta=\alpha_2 \Rightarrow ((y=0) \wedge (x1=E) \wedge (\odot x1=A) \wedge (\odot x2=x2))]$$

O trem só passa de "E" para "A" se não houver trem no cruzamento (y=0). O estado do portão não é alterado. Note que a própria planta controla a transição do estado de espera ao estado de Aproximação. É transição espontânea.

$$P4)\Box[\delta=\alpha_3 \Rightarrow (x1=A \wedge \odot x1=C \wedge y=y+1 \wedge \odot x2=x2 \wedge l_A=10 \wedge u_A=\infty)]$$

Esta assertiva informa que se α_3 ocorre, o trem passa de "A" para "C"; o contador y é incrementado de 1 e o estado do portão não se altera.

Tem-se aqui a expressão dos limites de tempo inferior e superior. Eles informam que o intervalo de tempo entre a imediata chegada do trem no estado "A" (ou a ocorrência do evento α_2) e a imediata chegada do trem no estado C (ou a ocorrência do evento α_3) varia de 10 a ∞ unidades de tempo.

$$P5)\Box[\delta=\gamma \Rightarrow (y>0 \wedge x1=C \wedge \odot x1=V \wedge x2=PF \wedge \odot x2=PA \wedge \odot y=y-1)]$$

Esta equação simplesmente informa que a ocorrência de γ só se dá se o valor do contador for maior que zero; que o estado do trem passa de "C" para "V"; que o estado do portão passa de "PF" para "PA" e que o contador é decrementado de 1. Vale salientar que o evento γ é um evento partilhado pelos dois subsistemas: o trem e o portão.

$$P6)\ \Box[\delta=\beta \Rightarrow (x2=PA \wedge \odot x2=PF \wedge \odot x1=x1)]$$

Se o evento β ocorre, o portão passa do estado "PA" para o estado "PF". A variável de estado do trem permanece inalterada.

$$P7)\Box[V\neq E \wedge V\neq A \wedge V\neq C \wedge V\neq PA \wedge V\neq PF \wedge E\neq A \wedge E\neq C \wedge E\neq PA \wedge E\neq PF \wedge A\neq C \wedge A\neq PA \wedge A\neq PF \wedge C\neq PA \wedge C\neq PF \wedge PA\neq PF]$$

Esta equação fornece uma lista de todos os estados da planta e afirma que todos são distintos entre si.

$$P8)\ x1=V \wedge x2=PA \wedge y=0$$

O estado inicial determina que o trem começa viajando, o portão começa aberto e o contador do número de trens no cruzamento começa com o valor zero.

ESPECIFICAÇÕES DE MALHA FECHADA EM GRTTL

Como especificações de malha fechada para este sistema tem-se o seguinte:

$$MF1)\ \Box \neg[x1=C \wedge x2=PA]$$

Nunca deve ocorrer a situação na qual o trem esteja no cruzamento e o portão aberto.

MF2) $\Box[(\delta=\beta)=>\odot((\delta=\alpha_2)\mathbb{P}(\delta=\beta))]$

Sempre que o portão fechar, seu próximo fechamento é precedido da aproximação de um trem. Em outras palavras, o portão não fecha à toa.

MF3) $\Box[(\delta=\alpha_2 \wedge t=T)=> \wedge (\delta=\beta \wedge (t\leq T+10))]$

Se o trem chegar no estado aproximando, o portão deve ser baixado em, no máximo, 10 unidades de tempo.

DESCRIÇÃO DO CONTROLADOR

Este exemplo foi descrito em várias referências [35], [23] e [26]. Na referência [23] o autor apresenta uma metodologia para construção do controlador, utilizando o formalismo TTM/RTTL. O controle gerado em GRTTL é similar ao ali encontrado.

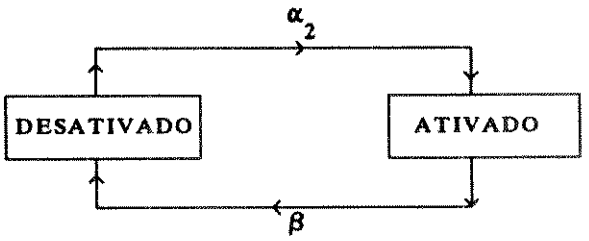
Existem duas maneiras de controlar o sistema em estudo. São elas:

1) Providenciar o intertravamento entre o trem e o portão. Isto dar-se-ia colocando como condição para a liberação do trem do estado de espera o fechamento do portão. Observa-se que todas as três exigências de malha fechada são satisfeitas com esta medida. Em outras palavras, o evento α_2 passa a ser um evento partilhado com β. Em termos de equações, tem-se:

   C1) $\Box[\delta=\alpha_2 => ((x2=PA)\ (\odot x2=PF))]$

2) Aproveitando o limite inferior de tempo na transição de "A" para "C", tem-se condições de elaborar um outro tipo de controle que satisfaça as especificações. Neste caso necessita-se de um comando que, após a ocorrência de α_2, dispararia a ocorrência de um evento compartilhado β com limite superior de 10 unidades de tempo. Isto garante que o portão está fechado quando da passagem

do trem pelo cruzamento. Para esta abordagem, o controlador teria o seguinte diagrama de estado:



Em termos de expressões em GRTTL tem-se:

ATIVADO	1
DESATIVADO	0

C1] $\square[\delta=\alpha_2 \Rightarrow (x3=0 \wedge (\odot x3=1))]$

Esta equação diz que se o evento α_2 ocorrer, o estado do controlador muda de desativado para ativado. Note que o evento α_2 é um evento partilhado.

C2) $\square[\delta=\beta \Rightarrow (x3=1 \wedge (\odot x3=0) \wedge l_{\alpha 3\beta}=0 \wedge u_{\alpha 3\beta}=10)]$

Esta assertiva informa que a ocorrência de β sinaliza a mudança de estado do controlador de ativado para desativado. O tempo associado a esta transição é de, no máximo, 10 unidades de tempo.

C3) x3=0

Esta assertiva informa que o estado inicial da variável x3 é igual a zero (desativado).

Por serem todas as duas alternativas de controladores por demais intuitivas e já documentadas em [23], não será apresentada a prova de que o controlador satisfaz as condições em malha-fechada.

Modelo de Simulação

Para a descrição do modelo de simulação para o sistema trem-portão, Segue-se a metodologia proposta anteriormente. É importante notar que condicionantes de tempo real já são incorporadas neste exemplo.

Neste caso há a necessidade de se criar uma entidade especial que faça o sincronismo do fechamento do portão. Ela faz parte do controlador e é chamada de ativador. É interessante notar que o ativador terá seus eventos partilhados com eventos das outras duas entidades.

1) ENTIDADES -

a) TREM
b) PORTÃO
c) ATIVADOR

2) EVENTOS -

a) TREM

* EVENTOS

PEDE_LIBERAÇÃO(0,ALEAT,1000)	(α1)	[1]
APROXIMA(0,ALEAT,1000)	(α2)	[2]
INC_CRUZAMENTO(10,ALEAT,1000)	(α3)	[3]
FIM_CRUZAMENTO(0,ALEAT,1000)	(α4)	[4]

* PROCEDIMENTOS

Relativos à dinâmica interna da planta

PEDE_LIBERAÇÃO: ALTERA_PORTA(EESTADO,W,E)[P2]
APROXIMA:ALTERA_PORTA(EESTADO,W, E) [P3]
INC_CRUZAMENTO:ALTERA_PORTA(EESTADO,W,C)[P4] ALTERA_PORTA(EY,+)
FIM_CRUZAMENTO:ALTERA_PORTA(EESTADO,W,Y)[P2] ALTERA_PORTA(EY,+)

c) ATIVADOR -

* EVENTOS

ATIVA (0,UNI,10) [1]

DESATIVA(0,UNI,1000) [2]

* PROCEDIMENTOS

ATIVA: ALTERA_FLAG(FATIVA, SET)
DESATIVA: ALTERA_FLAG(FATIVA, RESET)

3) SEQUENCIAMENTO DE EVENTOS

a) TREM

* ROTINA

ROT1 : {/>1>2>3>4>/}

b) PORTÃO

* ROTINA

ROT1 : {/>1>2>/}

c) ATIVADOR

* ROTINA

ROT1 : {/>1>2>/}

4) TEMPORALIDADE DAS TRANSIÇÕES

Neste exemplo aparecem limitantes de tempo real. Após a ocorrência do evento α_2, exige-se que o evento α_3 ocorra em, no mínimo, 10 unidades de tempo. Só o trecho no qual o trem se aproxima do cruzamento é que merece atenção, pois os outros são livres de condicionantes. Isto é expresso no SSBC indicando-se: o valor do intervalo de tempo até a ocorrência do próximo evento (10) e sua distribuição e dispersão estatística como é visto a seguir:

INC_CRUZAMENTO(10,ALEAT,1000) (α3) [3]

Esta expressão indica que a ocorrência do próximo evento se dá após 10 unidades de tempo com uma dispersão aleatória de 1000 (pode ocorrer a qualquer momento após as 10 unidades de tempo). Quando não há indicação, assume-se

a situação default de execução sem limitantes inferiores ou superiores de tempo. Outra maneira de representar a condição default é a seguinte:

PEDE_LIBERAÇÃO(0,ALEAT,1000) (α1) [1]

Para o ativador, tem-se também considerações de tempo real. Neste caso o evento ATIVA tem a seguinte expressão:

ATIVA (0,UNI,10) [1]

5) CONDICIONANTES -

a) TREM

REGRAS

SE VERIFICA_VALOR(EESTADO,VALOR)
E VALOR EH_IGUAL V
ENTÃO EXECUTA EVENTO 1 - [P2]

SE VERIFICA_VALOR(EESTADO,VALOR)
E VALOR EH_IGUAL E
E VERIFICA_VALOR(EY,VALOR)
E VALOR EH_IGUAL 0
ENTÃO EXECUTA EVENTO 2 - [P3]

SE VERIFICA_VALOR(EESTADO,VALOR)
E VALOR EH_IGUAL A
ENTÃO EXECUTA EVENTO 3 - [P4]

(REGRAS ASSOCIADAS AO CONTROLADOR)

SE SVERIFICA_FLAG(FATIVA/ATIVADOR)
E FATIVA ESTA RESET
ENTÃO SEXECUTA EVENTO 2/TREM - [C1]

SE SVERIFICA_FLAG(FATIVA/ATIVADOR)
E FATIVA ESTA SET
ENTÃO SEXECUTA EVENTO 1/PORTÃO - [C2]

SE VERIFICA_VALOR(EY,VALOR)
E VALOR EH_IGUAL 0
SE VERIFICA_VALOR(EESTADO,VALOR)
E VALOR EH_IGUAL C
SE VERIFICA_VALOR(EPORTÃO,VALOR)
E VALOR EH_IGUAL PF
ENTÃO EXECUTA EVENTO 4 - [P5]

REGRAS DE DECISÃO DE FLUXO

SE DECISÃO INICIAL
ENTÃO EXECUTA EVENTO 1

SE ULTIMO EVENTO FOI 4
ENTÃO EXECUTA EVENTO 1

b) PORTÃO

REGRAS

SE VERIFICA_VALOR(SPORTÃO,VALOR)
E VALOR EH IGUAL PF
ENTÃO EXECUTA EVENTO 1 - [P6]

SEVERIFICA_VALOR(SPORTÃO,VALOR)
E VALOR EH IGUAL PF
ENTÃO EXECUTA EVENTO1

REGRAS DE DECISÃO DE FLUXO

SE DECISÃO INICIAL
ENTÃO EXECUTA EVENTO 1

SE ULTIMO EVENTO FOI 2
ENTÃO EXECUTA EVENTO 1

c)ATIVADOR

REGRAS

SE VERIFICA_VALOR(EESTADO1,VALOR)
E VALOR EH IGUAL A
ENTÃO EXECUTA EVENTO 1

SE VERIFICA_VALOR(EESTADO1,VALOR)
E VALOR EH IGUAL PF
ENTÃO EXECUTA EVENTO 2
REGRAS DE DECISÃO DE FLUXO

SE ULTIMO EVENTO FOI 2
ENTÃO EXECUTA EVENTO 1

6) ELEMENTOS DE COMUNICAÇÃO FÍSICA E DE CONTROLE

a) TREM

EESTADO
EY

b) PORTÃO

SPORTÃO
SESTADO

c) ATIVADOR

FATIVA (flag)
EESTADO1
EPORTÃO1
} - PORTAS

7) INTERAÇÃO ENTRE ENTIDADES

SPORTÃO (portão) = EPORTÃO (trem) = EPORTÃO1 (ativador)
SESTADO (portão) = EESTADO (trem) = EESTADO1 (ativador)

8) ESTADO INICIAL

EESTADO (SESTADO) (EESTADO1) = V
SPORTÃO(EPORTÃO) (EPORTÃO1)= PA
EY = 0
FATIVA = RESET

B) SISTEMA DE PROCESSAMENTO COM DUAS MÁQUINAS

Este exemplo é uma extensão do exemplo apresentado em [36] onde algumas adaptações foram feitas para ressaltar questões de tempo real. Ele já foi descrito no capítulo 6, de modo que seu funcionamento e modelagem em GRTTL não é apresentado aqui. Passa-se direto para a modelagem para o simulador.

Modelo de Simulação

O modelo de simulação para o SSBC foi gerado a partir do conjunto de equações da planta (P1 a P11) e do controlador (C1 a C9).

1) ENTIDADES - O exame das equações de planta revela a existência das seguintes entidades:

a) GERADOR DE EVENTOS -

b) PLANTA - Cada peça é uma entidade passiva a ser processada pelas máquinas m1 e m2. Como única entidade ativa tem-se a própria planta como um todo.

OBS.: O controlador é apresentado através de um conjunto de regras que condicionam a operação da planta.

2) EVENTOS - Para cada entidade tem-se os respectivos eventos, e procedimentos associados.

a) GERADOR DE EVENTOS

A existência do gerador de eventos é necessária para que se proceda a uma indicação aleatória de um evento a ser avaliado e executado pela planta. Seu único evento é:

GERA_VALOR [1]

Aqui indica-se o único evento ocorrido no gerador de tipos de eventos, descrito com mais detalhe em seu procedimento a seguir.

* PROCEDIMENTOS:

GERA_VALOR{ALEAT,INT,1,8,STIPO}

A ocorrencia do evento GERA_VALOR dispara um procedimento que gera aleatoriamente um valor inteiro entre 1 e 8, correspondem aos eventos da planta. Este valor é usado como indicador de um evento a ser analisado e executado pela planta.

b) PLANTA

CHEGA_PEÇA	(α)	[1]
ESCOLHE_MÁQUINA(Z,UNI,0)	(β)	[2] (1)
INIC_ESP_2PROC_M2(Z,UNI,Z)	(ρ)	[3]
INIC_ESP_2PROC_M1(Z,UNI,Z)	(γ)	[4]
INIC_2PROC_M2(Z,UNI,0)	(μ)	[5]
INIC_2PROC_M1(Z,UNI,0)	(υ)	[6]
FIM_2PROC_M2	(ω)	[7]
FIM_2PROC_M1	(λ)	[8]

Aqui são indicados todos os eventos passíveis de ocorrer na planta.

* PROCEDIMENTOS:

Os procedimentos abaixo descritos são os associados com a dinâmica interna da planta:

CHEGA_PEÇA: ALTERA_PORTA{ENTRADA,+}

Este procedimento determina que o contador ENTRADA seja acrescido de 1 unidade quando da execução do evento chega_peça.

---

(1) Vide temporalidade das transições.

ESCOLHE_MAQ:GERA_VALOR(ALEAT,INT,1,2,STIPO)
VERIFICA_VALOR(STIPO,VALOR)
CONCAT(E,V,EMAQ)
ALTERA_VALOR(EMAQ,+)
ALTERA_PORTA(ENTRADA,-)

Este procedimento determina que seja gerado um valor inteiro aleatorio entre 1 e 2, que corresponde à escolha da máquina 1 ou 2. Este valor é concatenado com a indicação de porta "E", e esta porta (E1 ou E2) tem seu valor acrescido de uma unidade, representando que uma peça se encontra em processamento por aquela máquina (M1 ou M2). Por fim decrementa-se de uma unidade a porta ENTRADA, indicando que a peça não mais se encontra esperando o processamento.

INIC_ESP_2PROC_M2:ALTERA_VALOR(E1,-)
ALTERA_VALOR(ESPERA2,+)

Este procedimento indica que quando uma peça termina seu processamento pela máquina M1 e inicia a espera para ser processada pela máquina M2, decrementa-se o número de peças em processamento pela máquina 1 e incrementa-se de 1 o número de peças na espera para ser processado pela máquina M2.

INIC_ESP_2PROC_M1: ALTERA_VALOR(E2,-)
ALTERA_VALOR(ESPERA1,+)

Este procedimento indica que quando uma peça termina seu processamento pela máquina M2 e inicia a espera para ser processada pela máquina M1, decrementa-se o número de peças em processamento pela máquina 2 e incrementa-se de 1 o número de peças na espera para ser processado pela máquina M1.

INIC_2PROC_M1: ALTERA_PORTA(ESPERA1,-)
ALTERA_PORTA(E2PROC1,+)

Este procedimento indica que quando a peça tem iniciado seu 2º processamento pela máquina M1, decrementa-se de 1 unidade o número de peças na espera para serem processadas pela máquina 1 e incrementa-se de 1 unidade o número de peças em 2º processamento pela máquina M1.

INIC_2PROC_M2: ALTERA_PORTA{ESPERA2,-}
ALTERA_PORTA{E2PROC2,+}

Este procedimento indica que quando a peça tem iniciado seu 2º processamento pela máquina M2, decrementa-se de 1 unidade o número de peças na espera para serem processadas pela máquina 2 e incrementa-se de 1 unidade o número de peças em 2º processamento pela máquina M2.

FIM_2PROC_M1:ALTERA_PORTA{E2PROC1,-}
ALTERA_PORTA{SAIDA1,+}

Este procedimento indica que quando a peça tem terminado seu 2º processamento pela máquina M1, decrementa-se de 1 unidade o número de peças em 2º processamento pela máquina M1 e incrementa-se de 1 unidade o número de peças saidas da máquina M1.

FIM_2PROC_M2:ALTERA_PORTA{E2PROC2,-}
ALTERA_PORTA{SAIDA2,+}

Este procedimento indica que quando a peça tem terminado seu 2º processamento pela máquina M2, decrementa-se de 1 unidade o número de peças em 2º processamento pela máquina M2 e incrementa-se de 1 unidade o número de peças saidas da máquina M2.

Associados com as assertivas de controle tem-se os seguintes procedimentos:

CHEGA_PEÇA:ALTERA_PORTA{CONTADOR,+}
ALTERA_PORTA{CONTROLE,+}

Em termos de controle, a chegada de uma peça para processamento acrescenta 1 unidade ao contador "CONTADOR", que indica o número de peças que estão na espera para terem seu primeiro processamento. Também faz com que seja acrescido de 1 unidade o contador de pedidos "CONTROLE", que representa a individualidade de cada pedido, correspondente ao índice i nas especificações em GRTTL.

ESCOLHE_MÁQUINA:

A ocorrencia do evento em que se escolhe a máquina a processar a peça não implica em qualquer procedimento de controle.

INIC_ESP_2PROC_M2: VERIFICA_VALOR{CONTROLE,VALOR}
CONCATENA{QP,VALOR}
ALTERA_PORTA{CONTADOR,-}

O início de espera de uma peça saida da máquina M1, para ter seu 2º processamento na máquina M2, faz com que seu indicador de pedido (valor de CONTROLE) seja concatenado à fila QP que indica as peças esperando para serem processadas pela máquina M2. Também decrementa-se de 1 unidade o contador "CONTADOR", já que a peça foi encaminhada para seu primeiro processamento.

INIC_ESP_2PROC_M1: VERIFICA_VALOR{CONTROLE,VALOR}
CONCATENA{QQ,VALOR}
ALTERA_PORTA{CONTADOR,-}

O início de espera de uma peça saida da máquina M2, para ter seu 2º processamento na máquina M1, faz com que seu indicador de pedido (valor de CONTROLE) seja concatenado à fila QQ que indica as peças esperando para serem processadas pela máquina M1. Também decrementa-se de 1 unidade o contador "CONTADOR", já que a peça foi encaminhada para seu primeiro processamento.

INIC_2PROC_M2: AVALIA_CABEÇA{QP,CABEÇA}
ALTERA_PORTA{PROC2M2,W,CABEÇA}

O início do 2º processamento da peça pela máquina M2 faz com que a cabeça da fila de espera QP (indicador da peça) seja reconhecida e seu valor escrito no registrador que indica qual o indicador da peça em 2º processamento na máquina M2.

INIC_2PROC_M1: AVALIA_CABEÇA{QQ,CABEÇA}
ALTERA_PORTA{PROC2M1,W,CABEÇA}

O início de 2º processamento da peça pela máquina M1 faz com que a cabeça da fila de espera QQ (indicador da peça) seja reconhecida e seu valor escrito no registrador que indica qual o indicador da peça em 2º processamento na máquina M1.

FIM_2PROC_M2: ELIMINA_CABEÇA(QP)

A ocorrência do fim de 2º processamento pela máquina M2 faz com que seja eliminada da fila QP o indicador da peça que teve seu término de processamento (2º processamento).

FIM_2PROC_M1: ELIMINA_CABEÇA(QQ)

A ocorrência do fim do 2º processamento pela máquina M2 faz com que seja eliminada da fila QP o indicador da peça que teve seu término de processamento (2º processamento).

Obs.: É interessante notar que o evento β refere-se somente à escolha não determinística; não dando, a priori, nenhuma informação sobre qual máquina é escolhida.

3) SEQUENCIAMENTO DE EVENTOS

Aqui são indicados os sequenciamentos dos eventos em cada entidade do sistema.

a) GERADOR DE EVENTOS

* ROTINA:

ROT1 : { 1 >/ }

Esta indicação leva à ocorrência do único evento desta entidade que é a geração do número aleatório entre 1 e 8. A barra indica a decisão de fluxo a ser tomada quando se atinge o fim da rotina. Neste caso sempre o fluxo é remetido novamente ao evento 1, vide regra de decisão de fluxo abaixo.

b) PLANTA:

* ROTINA:

ROT1 : { /> 1 >/> 2 >/> 3 >/> 4 >/> 5 >/> 6 >/> 7 >/ 8/}

Aqui mostra-se que a rotina é completamente flexível, podendo-se executar qualquer evento, com base nas regras de decisão de fluxo abaixo. Isto decorre do fato de remeter-se a uma decisão de fluxo toda vez que qualquer evento seja executado. Nas regras de decisão de fluxo procede-se à escolha de qual evento deva ser executado.

4) TEMPORALIDADE DAS TRANSIÇÕES

Neste exemplo tem-se a ocorrência de transições com limitantes de tempo real. Por exemplo, Após a ocorrência do evento β, exige-se que o evento ρ (ou γ) ocorra num prazo de Z unidades de tempo. Isto é expresso no SSBC indicando-se: o valor do intervalo de tempo até a ocorrência do próximo evento (Z) e sua distribuição e dispersão estatística como tem-se a seguir:

ESCOLHE_MÁQUINA(Z,UNI,0) (β) [2]

Esta expressão indica que a ocorrência do próximo evento ocorre após Z unidades de tempo sem dispersão (Distribuição Uniforme com variação nula). Quando não há indicação, assume-se a situação default de execução sem limitantes inferiores ou superiores de tempo.

É interessante notar que os limitantes de tempo introduzidos pelo controlador nas equações em malha fechada, são incorporados às especificações dos eventos. Isto ocorre pela maneira única de se escrever as características de cada transição.

5) CONDICIONANTES

a) GERADOR DE EVENTOS

REGRAS DE DECISÃO DE FLUXO
SE ULTIMO_EV FOI 1
ENTÃO ESCREVE EVENTO 1

Esta regra indica a repetição contínua do evento gerador de um número aleatório.

b) PLANTA

REGRAS

Estas regras indicam a lógica de controle interno da própria planta, sem levar em conta o controlador.

SE VERIFICA_PORTA ENTRADA
E EESPERA EH_MAIOR 0
ENTÃO EXECUTA EVENTO 2

É necessário que haja peças esperando processamento para que o evento "escolha de máquina para processamento" ocorra.

SE VERIFICA_PORTA E1
E E2 EH_MAIOR 0
ENTAO EXECUTA EVENTO 3

É necessário que haja peças sendo processadas pela máquina M1 para que o evento "entrar em espera da máquina M2" ocorra.

SE VERIFICA_PORTA E2
E E1 EH_MAIOR 0
ENTAO EXECUTA EVENTO 4

É necessário que haja peças sendo processadas pela máquina M2 para que o evento "entar em espera da máquina M1" ocorra.

SE VERIFICA_PORTA ESPERA2
E ESPERA2 EH_MAIOR 0
ENTAO EXECUTA EVENTO 5

É necessário que haja peças esperando para serem processadas pela máquina M2 para que o evento "começar o processamento na máquina M2" ocorra.

SE VERIFICA_PORTA ESPERA1
E ESPERA1 EH_MAIOR 0
ENTAO EXECUTA EVENTO 6

É necessário que haja peças esperando para serem processadas pela máquina M1 para que o evento "começar o processamento na máquina M1" ocorra.

SE VERIFICA_PORTA E2PROC2
E E2PROC2 EH_MAIOR 0
ENTAO EXECUTA EVENTO 7

É necessário que haja peça sendo processada na máquina M2 para que o evento "termino do processamento 2 na máquina M2 " ocorra.

SE VERIFICA_PORTA E2PROC1
E E2PROC1 EH_MAIOR 0
ENTAO EXECUTA EVENTO 8

É necessário que haja peça sendo processada na máquina M1 para que o evento "término do processamento 2 na máquina M1 " ocorra.

REGRAS DE DECISÃO DE FLUXO

Estas regras indicam a decisão de fluxo a ser tomada pela planta após a execução de cada evento.

SE VERIFICA_VALOR (ETIPO,VALOR)
E VALOR EH_IGUAL 1 (C8)
ENTÃO EXECUTA EVENTO 1

SE VERIFICA_VALOR (ETIPO,VALOR)
E VALOR EH_IGUAL 2 (C7)
ENTÃO EXECUTA EVENTO 2

SE VERIFICA_VALOR
E VALOR EH_IGUAL 3 (C1)
ENTÃO EXECUTA EVEN

SE VERIFICA_VALOR (ETIPO,VALOR)
E VALOR EH_IGUAL 4 (C2)
ENTÃO EXECUTA EVENTO 4

SE VERIFICA_VALOR (ETIPO,VALOR)
E VALOR EH_IGUAL 5 (C3)
ENTÃO EXECUTA EVENTO 5

SE VERIFICA_VALOR (ETIPO,VALOR)
E VALOR EH_IGUAL 6 (C4)
ENTÃO EXECUTA EVENTO 6

SE VERIFICA_VALOR (ETIPO,VALOR)
E VALOR EH_IGUAL 7 (C5)
ENTÃO EXECUTA EVENTO 7

SE VERIFICA_VALOR (ETIPO,VALOR)
E VALOR EH_IGUAL 8 (C6)
ENTÃO EXECUTA EVENTO 8

Todas estas regras apenas verificam o tipo de evento indicado e o escalonam para execução.

6) ELEMENTOS DE COMUNICAÇÃO FÍSICA E DE CONTROLE

1) GERADOR DE EVENTOS
PORTAS
STIPO

b) PLANTA

PORTAS
ETIPO
ENTRADA
E1
E2
ESPERA1
ESPERA2
SAIDA1
SAIDA2
- Internas à planta.

CONTROLE
CONTADOR } - Relativas ao Controlador
PROC2M2

FILAS

QP

QQ

7) INTERAÇÃO ENTRE ENTIDADES

ETIPO (planta) = STIPO (gerador de eventos)

Aqui apenas indica-se que a porta ETIPO da planta deve ser automaticamente alterada junto com a porta STIPO do gerador de eventos. Isto é executado como o envio de uma mensagem com o valor do tipo de evento a ser executado, do gerador de eventos para a planta.

ESTADO INICIAL DO SISTEMA

O estado inicial dos sistema é indicado através do valor de cada porta, fila ou contador.

a) planta

ENTRADA = 0 STIPO = 0 E1=0 E2=0 ESPERA1=0
ESPERA2 = 0 SAIDA1 = 0 SAIDA2 = 0 ETIPO = 0

b) controle

CONTADOR = 0 CONTROLE = 0 PROC2M2 = 0
PROC2M1 = 0 QP = ø QQ = ø

7.7. CONCLUSÃO

Ao final deste capítulo confirma-se a possibilidade de transformação de especificações em GRTTL para modelos de simulação em SSBC. Além disto verifica-se que o modelo de simulação guarda semelhanças com a especificação inicialmente proposta, em GRTTL. Isto facilita o trabalho conjunto de pesquisadores das duas áreas envolvidas. Apresentou-se neste capítulo a proposição e discussão de uma nova abordagem de controle de sistemas a eventos discretos baseada na interação da lógica temporal com a simulação.

CAPÍTULO 8

RESULTADOS DE SIMULAÇÃO

Este capítulo dedica-se à apresentação de simulações feitas no SSBC do sistema Trem-Portão descrito e modelado no capítulo 7. Este sistema foi escolhido pela sua simplicidade e consequente facilidade na modelagem.

O SSBC fornece como saída, no seu estado atual de desenvolvimento, uma lista contendo os eventos ocorridos e os respectivos tempos. Considerando que esta interface homem-máquina não é de fácil visualização, estes resultados são depois comentados de uma maneira mais clara para o leitor.

Algumas das simulações a seguir poderão parecer óbvias. No entanto, deve-se observar sua função de confirmação da correção das sentenças em lógica temporal. Caso as últimas não estivessem corretamente escritas a possibilidade de ocorrência de situações díspares aumentaria com o nº de casos simulados.

Para a análise dos resultados, algumas considerações quanto à notação precisam ser feitas, em particular, quanto ao registro dos resultados e à escolha dos tempos e intervalos.

Registro dos resultados

Para o registro dos eventos ocorridos na simulação, o SSBC fornece um relatório (Trace) onde cada linha é composta pelo seguinte conjunto de informações:

[Entidade,[Nome_do_evento,Instante_em_que_evento_ocorre,Número_do_evento].

Quando dois ou mais eventos ocorrem ao mesmo tempo, o simulador declara isto através da coincidência dos tempos de ocorrência, enfileirando-os no registro de maneira irrelevante.

Escolha dos tempos e intervalos

Algumas variáveis do sistema proposto têm caráter não-determinístico, como o tempo que o trem leva para pedir liberação, o tempo que o controlador (ativador) leva para responder a este pedido, o tempo que o trem leva para passar pelo cruzamento, etc. Para esta simulação, os tempos foram escolhidos arbitrariamente, de modo que os tempos se repetem ao longo de todo o processo. Simulações com os tempos acima citados variando conforme uma geração aleatória são previstas em futuros trabalhos, onde a implementação das mesmas seja mais imediata no simulador escolhido.

Em algumas simulações alguns ciclos serão subtraídos para que o texto não se alongue em demasia, desde que a compreensão dos resultados não seja alterada.

RESULTADOS DO SSBC

SIMULAÇÃO 1: TREM-PORTÃO FUNCIONANDO CORRETAMENTE - ARQUIVO TRACE01.DAT

```
[trem,[pede_lib,0,1]].
[trem,[aprox,10,2]].
[ativ,[ativa,10,1]].
[canc,[fecha,10,1]].
[trem,[inic_cruzamento,20,3]].
[trem,[fim_cruzamento,40,4]].
[canc,[abre,40,2]].
[ativ,[desativa,40,2]].                                  1

[trem,[pede_lib,90,1]].
[trem,[aprox,100,2]].
[ativ,[ativa,100,1]].
[canc,[fecha,100,1]].
[trem,[inic_cruzamento,110,3]].
[trem,[fim_cruzamento,130,4]].
[canc,[abre,130,2]].
[ativ,[desativa,130,2]].                                 2

[trem,[pede_lib,180,1]].
[trem,[aprox,190,2]].
[ativ,[ativa,190,1]].
[canc,[fecha,190,1]].
[trem,[inic_cruzamento,200,3]].
[trem,[fim_cruzamento,220,4]].
[canc,[abre,220,2]].
[ativ,[desativa,220,2]].                                 3

[trem,[pede_lib,270,1]].
[trem,[aprox,280,2]].
[ativ,[ativa,280,1]].
[canc,[fecha,280,1]].
[trem,[inic_cruzamento,290,3]].
[trem,[fim_cruzamento,310,4]].
[canc,[abre,310,2]].
[ativ,[desativa,310,2]].                                 4

[trem,[pede_lib,540,1]].
[trem,[aprox,550,2]].
[ativ,[ativa,550,1]].
[canc,[fecha,550,1]].
[trem,[inic_cruzamento,560,3]].
[trem,[fim_cruzamento,580,4]].
[canc,[abre,580,2]].
[ativ,[desativa,580,2]].                                 7

[trem,[pede_lib,630,1]].
[trem,[aprox,640,2]].
[ativ,[ativa,640,1]].
[canc,[fecha,640,1]].
[trem,[inic_cruzamento,650,3]].
[trem,[fim_cruzamento,670,4]].
[canc,[abre,670,2]].
[ativ,[desativa,670,2]].                                 8

[trem,[pede_lib,720,1]].
[trem,[aprox,730,2]].
[ativ,[ativa,730,1]].
[canc,[fecha,730,1]].
[trem,[inic_cruzamento,740,3]].
[trem,[fim_cruzamento,760,4]].
[canc,[abre,760,2]].
[ativ,[desativa,760,2]].                                 9

[trem,[pede_lib,810,1]].
[trem,[aprox,820,2]].
[ativ,[ativa,820,1]].
[canc,[fecha,820,1]].
[trem,[inic_cruzamento,830,3]].
[trem,[fim_cruzamento,850,4]].
[canc,[abre,850,2]].
[ativ,[desativa,850,2]].                                 10
```

```
[trem,[pede_lib,360,1]].
[trem,[aprox,370,2]].
[ativ,[ativa,370,1]].
[canc,[fecha,370,1]].
[trem,[inic_cruzamento,380,3]].
[trem,[fim_cruzamento,400,4]].
[canc,[abre,400,2]].
[ativ,[desativa,400,2]].                    } 5

[trem,[pede_lib,450,1]].
[trem,[aprox,460,2]].
[ativ,[ativa,460,1]].
[canc,[fecha,460,1]].
[trem,[inic_cruzamento,470,3]].
[trem,[fim_cruzamento,490,4]].
[canc,[abre,490,2]].
[ativ,[desativa,490,2]].                    } 6

[trem,[pede_lib,900,1]].
[trem,[aprox,910,2]].
[ativ,[ativa,910,1]].
[canc,[fecha,910,1]].
[trem,[inic_cruzamento,920,3]].
[trem,[fim_cruzamento,940,4]].
[canc,[abre,940,2]].
[ativ,[desativa,940,2]].                    } 11

[trem,[pede_lib,990,1]].
[trem,[aprox,1000,2]].
[ativ,[ativa,1000,1]].
[canc,[fecha,1000,1]].
[trem,[inic_cruzamento,1010,3].
[trem,[fim_cruzamento,1030,4]].
[canc,[abre,1030,2]].
[ativ,[desativa,1030,2]].                   } 12
```

Neste caso a simulação funcionou perfeitamente. Note que o portão fecha sempre antes do trem estar no cruzamento, e abre assim que o trem sai do mesmo. O tempo no cruzamento é de 20 unidades de tempo. Cada grupo de oito instruções corresponde a um ciclo completo do sistema. O tempo total de simulação foi de 1030 unidades de tempo.

SIMULAÇÃO 2: TREM-PORTÃO OPERANDO DE MANEIRA IRREGULAR - ARQUIVO TRACE03.DAT

```
[trem,[pede_lib,0,1]].
[trem,[aprox,10,2]].
[ativ,[ativa,10,1]].
[canc,[fecha,10,1]].
[trem,[inic_cruzamento,20,3]].
[trem,[fim_cruzamento,40,4]].
[canc,[abre,40,2]].                         } 1

[trem,[pede_lib,80,1]].
[ativ,[desativa,80,2]].
[trem,[aprox,90,2]].
[trem,[inic_cruzamento,100,3]].
[ativ,[ativa,100,1]].
[canc,[fecha,100,1]].
[trem,[fim_cruzamento,120,4]].
[canc,[abre,120,2]].                        } 3
```

[trem,[pede_lib,40,1]].
[ativ,[desativa,40,2]].
[trem,[aprox,50,2]].
[trem,[inic_cruzamento,60,3]].
[ativ,[ativa,60,1]].
[canc,[fecha,60,1]].
[trem,[fim_cruzamento,80,4]].
[canc,[abre,80,2]].

2

[trem,[pede_lib,120,1]].
[ativ,[desativa,120,2]].
[trem,[aprox,130,2]].
[trem,[inic_cruzamento,140,3]].
[ativ,[ativa,140,1]].
[canc,[fecha,140,1]].
[trem,[fim_cruzamento,140,4]].
[canc,[abre,140,2]].

4

Nesta simulação, foi retirada a regra que dá o controle do fechamento do portão ao ativador. Foi estipulado um tempo arbitrário para o fechamento do portão de 20 unidades de tempo, após o fechamento do mesmo. Como os tempos envolvidos são múltiplos de 20, coincidentemente o trem só passa no cruzamento no momento em que o portão está fechando, não ocasionando maiores desastres, mas já não se tem uma situação cômoda de fechamento, como no exemplo anterior. O tempo total de simulação foi de 140 unidades de tempo.

SIMULAÇÃO 3: TREM-PORTÃO OPERANDO DE MANEIRA IRREGULAR - ARQUIVO TRACE04.DAT

[trem,[pede_lib,0,1]].
[canc,[fecha,0,1]].
[trem,[aprox,10,2]].
[ativ,[ativa,10,1]].
[trem,[inic_cruzamento,20,3]].
[trem,[fim_cruzamento,40,4]].
[canc,[abre,40,2]].

1

[trem,[pede_lib,120,1]].
[ativ,[desativa,120,2]].
[trem,[aprox,130,2]].
[trem,[inic_cruzamento,140,3]].
[ativ,[ativa,140,1]].
[canc,[fecha,147,1]].
[trem,[fim_cruzamento,160,4]].
[canc,[abre,160,2]].

4

[trem,[pede_lib,40,1]].
[ativ,[desativa,40,2]].
[trem,[aprox,50,2]].
[trem,[inic_cruzamento,60,3]].
[ativ,[ativa,60,1]].
[canc,[fecha,67,1]].
[trem,[fim_cruzamento,80,4]].
[canc,[abre,80,2]].

2

[trem,[pede_lib,160,1]].
[ativ,[desativa,160,2]].
[trem,[aprox,170,2]].
[trem,[inic_cruzamento,180,3]].
[ativ,[ativa,180,1]].
[canc,[fecha,187,1]].
[trem,[fim_cruzamento,200,4]].
[canc,[abre,200,2]].

5

```
[trem,[pede_lib,80,1]].
[ativ,[desativa,80,2]].
[trem,[aprox,90,2]].
[trem,[inic_cruzamento,100,3]]
[ativ,[ativa,100,1]].
[canc,[fecha,107,1]].
[trem,[fim_cruzamento,120,4]].
[canc,[abre,120,2]].
```

3

```
[trem,[pede_lib,200,1]].
[ativ,[desativa,200,2]].
[trem,[aprox,210,2]].
[trem,[inic_cruzamento,220,3]].
[ativ,[ativa,220,1]].
[canc,[fecha,227,1]].
[trem,[fim_cruzamento,240,4]].
[canc,[abre,240,2]].
```

6

Esta simulação é similar à anterior, com o agravante de que agora o tempo entre a abertura e o fechamento do portão é de 27 unidades de tempo. Com isto, já é possível ver as consequências deletérias da ausência de um controle do fechamento do portão. Num exemplo real, isto demonstraria que a regra retirada é de vital importância para o bom funcionamento do sistema, auxiliando no desenvolvimento do controlador. O tempo total de simulação é igual a 240 unidades de tempo.

SIMULAÇÃO 4: TREM-PORTÃO OPERANDO DE MANEIRA IRREGULAR - ARQUIVO TRACE05.DAT

```
[trem,[pede_lib,0,1]].
[canc,[fecha,0,1]].
[trem,[aprox,10,2]].
[ativ,[ativa,10,1]].
[trem,[inic_cruzamento,20,3]].
[trem,[fim_cruzamento,40,4]].
```

1

```
[trem,[pede_lib,80,1]].
[trem,[aprox,90,2]].
[trem,[inic_cruzamento,100,3]].
[trem,[fim_cruzamento,120,4]].
[trem,[pede_lib,120,1]].
[trem,[aprox,130,2]].
[trem,[inic_cruzamento,140,3]].
[trem,[fim_cruzamento,160,4]].
```

3

```
[trem,[pede_lib,40,1]].
[canc,[abre,40,2]].
[ativ,[desativa,40,2]].
[trem,[aprox,50,2]].
[trem,[inic_cruzamento,60,3]].
[ativ,[ativa,60,1]].
[trem,[fim_cruzamento,80,4]].
```

2

Nesta simulação, além do que já acontecia nas simulações 2 e 3, o tempo entre a abertura e o fechamento do portão é infinito (na prática, maior que o tempo de simulação). Isto acarretou a passagem do trem várias vezes pelo cruzamento sem a correspondente resposta do portão.

CAPÍTULO 9

CONCLUSÃO

A apresentação da análise e controle de DEDS através da GRTTL foi o objetivo principal deste trabalho. Acredita-se que a GRTTL é um formalismo inédito, até onde se tem notícia, combinando aspectos de indeterminismo e considerações de tempo real numa abordagem primal. Seu conjunto de símbolos, axiomas e regras de inferências foi composto por elementos de formalismos diferentes, ganhando como capacidade de descrição de sistemas a soma das capacidades de cada um deles separadamente. A regra de inferência derivada cuja prova foi esboçada é a primeira de um grupo de regras que podem ser adaptadas do formalismo apresentado por Ostroff [203] para a GRTTL.

A aplicação da lógica na simulação de DEDS foi feita através da associação da GRTTL com o SSBC fornecendo um conjunto simulador com características formais bem acentuadas, provindas da GRTTL, não negligenciando, contudo, os aspectos mais voltados para a simulação, supridos pelo SSBC. De uma maneira mais geral, a princípio, uma lógica genérica poderia ser associada com um simulador também genérico, desde que os dois satisfaçam os requisitos descritos no capítulo 7. Neste caso, haveria a necessidade de se esboçar um mapeamento das características e gerar uma metodologia de transformação entre a lógica e o simulador escolhidos. Acredita-se que esta associação entre a GRTTL e o SSBC pode ser um valioso resultado tanto em simulação quanto em controle de sistemas.

Com respeito a perspectivas futuras, vislumbra-se vários caminhos que podem ser trilhados no sentido de aprimorar o presente trabalho. Ampliar o sistema de prova da GRTTL é, com certeza, uma tarefa interessante, visto que possibilita uma maior versatilidade na prova de asserções. Uma tarefa não menos interessante é a associação da GRTTL com outros formalismos, na tentativa de se conseguir sistemas de representação e análise de DEDS cada vez mais concisos e poderosos. Sugere-se aqui a associação de lógicas não-clássicas com a álgebra minimax [15], pois esta junção pode produzir uma base teórica para a proposição de um sistema bastante geral e poderoso para a descrição irrestrita de DEDS. Um objetivo bastante citado pela literatura tem sido a criação de um sintetizador automático de controladores, a partir da equações da planta e das especificações em malha-fechada, para o caso infinito. Ainda que não se tenha resultados nesta direção, a GRTTL pode se prestar para este estudo.

Em termos da aplicação da GRTTL junto com o SSBC, várias perspectivas são apresentadas. A prova rigorosa de que a passagem das expressões em GRTTL para o SSBC não compromete o conteúdo semântico é uma boa direção que pode ser

trilhada. Para tanto, pode-se definir um grupo de características que as fórmulas em GRTTL possuem e mostrar que o algoritmo de transformação para o SSBC não as altera, e vice-versa. Um conversor semi-automático do formato GRTTL para o SSBC é um tema de estudo interessante a ser desenvolvido.

REFERÊNCIAS

[1] Ho,Y.C.; *"Dynamics of Discrete Event Systems"*, Proceedings of the IEEE, JAN, 1989, pp. 3-6.

[2] Cao,X.R. e Ho,Y.C.; *"Models of Discrete Event Dynamics Systems"*, Special Section - IEEE Control Systems Magazine, JUL, 1990, pp. 69-76.

[3] Cinlar,E.;*"Introduction to Stochastic Process"*, Prentice Hall, Inc, 1975.

[4] Chrétienne,P.H. e Faure,R.; *"Processus stochastiques, leurs graphes, leurs usages"*, Recherche Opérationelle Appliquée 2, 1974.

[5] Ross,S.M.; *"Stochastic Processes"*, Wiley series in probability and mathematical Statitistics, 1983.

[6] Kleinrock,L.; *"Queueing Systems"*, Vol.I - Theory, New York: Wiley, 1975.

[7] Panico,J.A.; *"Queueing Theory"*, Prentice Hall Inc., 1963.

[8] Gelenbe,E.; *"Introduction to Queueing Networks"*, John Wiley & Sons, 1987.

[9] Haas,P.J. e Shedler,G.S.; *"Regenerative Generalized Semi-Markov Processes"*, Stochastic Models, Vol 3, pp. 409-438, 1987.

[10] Ramadge,P.J. e Wonham,W.M.; *"Supervisory Control of a Class of Discrete Event Processes"*, SIAM Journal of Control and Optmization, vol.25, pp 206-230, 1987.

[11] Wonham,W.M.; *"On the Control of Discrete-Event Systems. Three Decades of Mathematical Systems Theory"*. Lecture Notes On Computer, Information end Science, Vol.135, pp.542-562, 1989.

[12] Lin,F e Wonham,W.M.; *"Descentralized Supervisory Control of Discrete-event systems"*, Inf. Sci., Vol.44, pp. 199-224, 1988.

[13] Cohen,G., Dubois,D., Quadrat,J.P. e Viot,M.; *"A linear system theoretic view of discrete event processes and its use for performance evaluation in manufacturing"*, IEEE Transactions on Automatic Control, Vol.30, pp. 210-220, 1985.

[14] Cohen,G., Moller,P, Quadrat,J.P. e Viot,M.;*"Algebraic tools for the performance evaluation of Discrete Event Systems"*, Proceedings of the IEEE,Vol.77, nº1, pp. 39-58, JAN, 1989.

[15] Cunninghame-Green,R.A.; *"Minimax Algebra"*, Berlin, W. Germany: Springer-Verlag, 1979.

[16] Cassandras,C., Cao,X.R. e Ho,Y.C.; *"Discrete Event Dynamic Systems: Modeling, Analysis, and Application"*, American control conference, Workshop notes, 1988.

[17] Peterson,J.L.; *"Petri Net Theory and the Modeling of Systems"*, Englewood Cliffs, NJ: Prentice Hall, 1986.

[18] Heuser, C.A.; *"Modelagem Conceitual de Sistemas"*, EBAI - Kapelusz, 1988.

[19] Marton,M.; "*Analisador automático de rede de Petri temporizada para validadção de protocolos de comunicação*", Tese de Mestrado 918 - UNICAMP, 1989.

[20] Inan,K. e Varaya, P.; "Finitely recursive process models for discrete event systems, IEEE, Transactions Automatica Control, vol.33, pp. 626-639, 1988.

[21] Hughes,G.E. e CressWell,M.J.; "*An Introduction to Modal Logic*", Methuen an Co. Ltda, 1977.

[22] Rescher,N. e Urquhart,A.;"*Temporal Logic*", Springer-Verlag, Library of Exact Philosophy, 1971.

[23] Ostroff,J.S.;"*Temporal Logic for Real-Time Systems*", John Wiley & Sons Inc., 1989.

[24] Lehmann,D. e Shelar,S.;"*Reasoning with time and Chance*", Information and Control, 53, pp. 165-198,1982.

[25] Manna,Z. e Pnueli,A.;"*Verification of Concurrent Programs: A Temporal Proof System*", Technical Report, STAN-CS-83-967, Dept. of Computer Science, Stanford University, JUN-1983.

[26] Ostroff,J.S.;"*Synthesis of Controllers for Real-time Discrete Event Systems*", IEEE CDC, Tampa, Florida, DEC, pp.138-144.

[27] Enderton,H.B.;"*A Mathematical Introduction to Logic*", Academic Press, New York, 1972.

[28] Thistle,J.G. e Wonham, W.M.;"*Control Problems in A temporal Logic Framework*", Int. Journal of control, Vol.44, Nº 04, 943-976, 1986.

[29] Manna,Z e Wolper,P;"*Synthesis of Communicating Processes from Temporal Logic Specifications*", ACM Transactions on Programming Languages and systems, 6(1), pp. 68-93, JAN, 1984.

[30] Moszkowski,B.;"*A temporal Logic for Multilevel Reasoning about Hardware*", Computer, 18(2), pp. 10-19, FEV, 1985.

[31] Knight.J.F. e Passino,K.M., 1990, "*Decidability for a Temporal Logic Used in Discrete-Event System Analysis*", International Journal of Control, Vol.52, Nº 6, 1489-1506, 1990.

[32] Buchi,J.R.;"*On a decision Method in restricted second order arithimetic*", Proceedings of the 1960 International congress of Logic, Methodology en Philosophy of Science, Ed. E. Nagel, 1-11, 1962.

[33] Passino,K.M. e Antsaklis,P.J.;"*Branching time temporal logic for discrete event system analysis*", Proceeding 26º Allerton Conference on communication, Control and computing, University of Illinois at Urbana-Champaign, pp. 1160-1169, 1988.

[34] Ostroff,J.S.;"*Real-Time Computer Control of Discrete Systems Modeled by extended State Machines: a Temporal Logic Approach*", PhD Thesis, University of Toronto, 1987.

[35] Ostroff,J.S. e Wonham,W.M.;"*A framework for Real-Time Discrete Event Control*", IEEE Transactions on Automatic Control, Vol.35, Nº 04, ABR, 1990.

[36] Lin.J.Y. e Ionescu,D;"*A generalized Temporal Logic Approach for Control Problems of a Class of Nondeterministic Discrete Event Systems*", Proceedings of the 29º CDC, Honolulu, Hawaii, DEZ, 1990.

[37] Kemeny,J.G., Shell,J.L. e Knapp,A.W.;"*Denumerable Markow Chains*", Van Nostrand, Princeton, NJ, 1966.

[38] Feller,W.; "*An introduction to Probability Theory and its Applications*", John Wiley, New York, 1957.

[39] Loyolla, W. e Signoretti, A., (1991). "*Um ambiente de simulação baseado em conhecimento*", RT-DCA 004/90, FEE-UNICAMP, Campinas.

[40] Loyolla,W.; Gomide,F.; Signoretti,A.; Mendes, M.J. e Bingulac,S., (1991). "*A knowledge-Based Simulation Environment*" 5th IFAC/IMACS ,JUL, SWANSEA, UK.

[41] Silva Jr., Braz I. e Mendes, Rafael S.; "*Lógica Temporal de tempo real aplicada ao controle de sistemas a eventos Discretos não determinísticos*", RT-DCA 025/91, FEE-UNICAMP, Campinas, 1991.

[42] Silva Jr., Braz I. et al; "*Aplicação de Lógica Temporal de Tempo Real Generalizada na Modelagem e Simulação de Sistemas a Eventos Discretos*", RT-DCA 026/91, FEE-UNICAMP. Campinas, 1991.

[43] Silva Jr., Braz I. e Mendes, Rafael S.; "*Lógica Temporal de Tempo Real Aplicada ao Controle de Sistemas a Eventos Discretos não Determinísticos*", 9º CBA, Vitória, ES, Brasil, SET, 1992.

[44] Loyolla, Waldomiro P.D.C. e Silva Jr., Braz I.; "*Aplicação de Lógica Temporal de Tempo Real Generalizada na Modelagem e Simulação de Sistemas a Eventos Discretos*", 9º CBA, Vitória, ES, Brasil, SET, 1992.

[45] Silva Jr., Braz I. e Mendes, Rafael S.; "*Generalized Real-Time Temporal Logic Applied on Control of non-deterministic Discrete Event Dynamic Systems*", Submetido ao 31º IEEE CDC, Tucson, Arizona, DEC, 1992.

APÊNDICE

Neste apêndice, a GRTTL é sumarizada e os principais resultados de outros autores que são utilizados neste trabalho são apresentados aqui.

1) SINTAXE

A - SÍMBOLOS

- **Símbolos lógicos:**
  Igualdade: $=$
  Conectivos: $\neg$, $\longrightarrow$
  Parênteses: (,)
  Símbolos variáveis globais: U_0, U_1, ...
  Símbolos variáveis locais: u_0, u_1, ...
  Símbolos operadores temporais: $\odot$, $\mathcal{U}$, $\Diamond$
  Símbolo operador certeza: ∇

- **Parâmetros:**
  Símbolo quantificador: $\exists$
  Símbolo constantes globais: C_0, C_1, ...
  Letras de predicado: relações n-árias, $n > 0$
  Letras de função: funções n-árias, $n > 0$

A GRTTL é uma linguagem multi-sorte, onde existe um conjunto I não-vazio, cujos membros são chamados de sortes. Cada sorte tem seus próprios símbolos variáveis (globais e locais), seus símbolos constantes globais, predicados, funções e o símbolo quantificador. Cada sorte i tem um quantificador $\exists_i$ que quantifica sobre os elementos da sorte i.

B - TERMOS

Para qualquer sorte i, os termos de sorte i da linguagem são definidos indutivamente como segue:

- Todos os símbolos constantes globais são termos.
- Todos os símbolos variáveis (globais e locais) são termos.
- Se $t_1,...,t_n$ são termos e f é uma letra de função n-ária, então $f(t_1,...,f_n)$ é um termo.

- Se t é um termo, ⊙t é um termo.
- Nenhuma outra seqüência, diferente das anteriores é um termo.

C - FÓRMULAS

As fórmulas-bem-formadas da linguagem são:

i) Se $t_1,t_2,...,t_n$ são termos de sorte $t_1,...,t_n$, respectivamente, e P é qualquer letra de predicado de sorte $i_1,i_2,...,i_n$, então $\mathcal{P}(t_1,t_2,...,t_n)$ é uma fórmula-bem-formada.

ii) Se w é uma fórmula-bem-formada, então ($\neg$w), (⊙w), ($\Diamond$w) e (∇w) são fórmulas-bem-formadas.

iii) Se w_1 e w_2 são fórmulas-bem-formadas, então $(w_1 \rightarrow w_2)$ e $(w_1 \mathcal{U} w_2)$ são fórmulas-bem-formadas.

iv) Se V é um símbolo variável global e W é uma fórmula-bem-formada, então (∃V:w) é uma fórmula-bem-formada.

v) Nenhuma outra seqüência é uma fórmula-bem-formada.

Mapeamento VAR - Define-se o mapeamento VAR da seguinte maneira:

$VAR(C) \triangleq \varnothing$, $VAR(X) \triangleq \{x\}$

$VAR(f(t_1,t_2,...,t_n)) \triangleq VAR(t_1) \cup VAR(t_2) \cup ... \cup VAR(t_n)$, onde VAR(t) é o conjunto composto de todas as variáveis que ocorrem em t.

Variáveis Livres - Seja x qualquer símbolo variável (local ou global), C qualquer símbolo constante individual, e $t_1,t_2,...,t_n$ quaisquer termos. Define-se o conjunto das variáveis livres em uma fórmula w (denotado por livre(w)) como segue:

$Livre\ (t_1 = t_2) \triangleq VAR(t_1) \cup VAR(t_2)$

$Livre\ (p(t_1,t_2,...,t_n) \triangleq VAR(t_1) \cup ... \cup \lambda\lambda R(t_n)$

$Livre\ (\neg,w_1) \triangleq Livre\ (w_1)$

$Livre\ (w_1 \rightarrow w_2) \triangleq Livre\ (w_1) \cup Livre\ (w_2)$

$Livre\ (\exists V : w_1) \triangleq Livre\ (w_1) - \{V\}$

$Livre\ (\odot w_1) \triangleq Livre\ (w_1)$

$Livre\ (\nabla w_1) \triangleq Livre\ (w_1)$

$$\text{Livre}\ (w_1, \mathcal{U} w_2) \triangleq \text{Livre}\ (w_1) \cup \text{Livre}\ (w_2)$$

$$\text{livre}(\nabla w_1) \triangleq \text{livre}(w_1).$$

Substituição simultânea - Sejam $g_1, g_2, \ldots, g_n$ termos tais que para todo i, a variável x_i pertence à mesma sorte que g_i. Então a nova fórmula $w\begin{bmatrix} x_1 \ldots x_n \\ g_1 \ldots g_n \end{bmatrix}$ pode ser obtida de w substituindo-se simultaneamente cada variável x_i pelo termo g_i. Ou seja, onde se encontra x_i, coloca-se g_i. Este procedimento ocorre da seguinte maneira:

constante $$C\begin{bmatrix} x_1 \ldots x_n \\ g_1 \ldots g_n \end{bmatrix} \triangleq C$$

variável $$X\begin{bmatrix} x_1 \ldots x_n \\ g_1 \ldots g_n \end{bmatrix} \triangleq \begin{cases} x_i \text{ se } x \neq x_i \text{ para todo i} \\ g_i \text{ se } x = x_i \end{cases}$$

função $$f(t_1, t_2, \ldots, t_n)\begin{bmatrix} x_1 \ldots x_n \\ g_1 \ldots g_n \end{bmatrix} \triangleq f\left(t_1\begin{bmatrix} x_1 \ldots x_n \\ g_1 \ldots g_n \end{bmatrix} \ldots t_n\begin{bmatrix} x_1 \ldots x_n \\ g_1 \ldots g_n \end{bmatrix}\right)$$

igualdade $$(t_1 = t_2)\begin{bmatrix} x_1 \ldots x_n \\ g_1 \ldots g_n \end{bmatrix} \triangleq t_1\begin{bmatrix} x_1 \ldots x_n \\ g_1 \ldots g_n \end{bmatrix} = t_2\begin{bmatrix} x_1 \ldots x_n \\ g_1 \ldots g_n \end{bmatrix}$$

predicados $$p(t_1, t_2, \ldots, t_n)\begin{bmatrix} x_1 \ldots x_n \\ g_1 \ldots g_n \end{bmatrix} \triangleq p\left(t_1\begin{bmatrix} x_1 \ldots x_n \\ g_1 \ldots g_n \end{bmatrix}, \ldots, t_n\begin{bmatrix} x_1 \ldots x_n \\ g_1 \ldots g_n \end{bmatrix}\right)$$

negação $$(\neg w_1)\begin{bmatrix} x_1 \ldots x_n \\ g_1 \ldots g_n \end{bmatrix} \triangleq \neg\left(w_1\begin{bmatrix} x_1 \ldots x_n \\ g_1 \ldots g_n \end{bmatrix}\right)$$

implicação $$(w_1 \rightarrow w_2)\begin{bmatrix} x_1 \ldots x_n \\ g_1 \ldots g_n \end{bmatrix} \triangleq \left(w_1\begin{bmatrix} x_1 \ldots x_n \\ g_1 \ldots g_n \end{bmatrix} \rightarrow w_2\begin{bmatrix} x_1 \ldots x_n \\ g_1 \ldots g_n \end{bmatrix}\right)$$

próximo $$(\odot w_1)\begin{bmatrix} x_1 \ldots x_n \\ g_1 \ldots g_n \end{bmatrix} \triangleq \odot\left(w_1\begin{bmatrix} x_1 \ldots x_n \\ g_1 \ldots g_n \end{bmatrix}\right)$$

eventualmente $$(\Diamond w_1)\begin{bmatrix} x_1 \ldots x_n \\ g_1 \ldots g_n \end{bmatrix} \triangleq \Diamond\left(w_1\begin{bmatrix} x_1 \ldots x_n \\ g_1 \ldots g_n \end{bmatrix}\right)$$

até que $$(w_1, \mathcal{U} w_2)\begin{bmatrix} x_1 \ldots x_n \\ g_1 \ldots g_n \end{bmatrix} \triangleq \left(w_1\begin{bmatrix} x_1 \ldots x_n \\ g_1 \ldots g_n \end{bmatrix}\ \mathcal{U}\ w_2\begin{bmatrix} x_1 \ldots x_n \\ g_1 \ldots g_n \end{bmatrix}\right)$$

Existe $$(\exists V{:}w_1)\begin{bmatrix} x_1 \ldots x_n \\ g_1 \ldots g_n \end{bmatrix} \triangleq \begin{cases} (\exists V{:}w_1)\begin{bmatrix} x_1 \ldots x_{i-1} x_{i+1} \ldots x_n \\ g_1 \ldots g_{i-1} g_{i+1} \ldots g_n \end{bmatrix} & \text{Se } x_i = V \\ & \text{para algum i} \\ (\exists V{:}w_1)\begin{bmatrix} x_1 \ldots x_n \\ g_1 \ldots g_n \end{bmatrix} & \text{Se } x_i \neq V \text{ para todo i} \end{cases}$$

certeza $(\nabla w_1)\begin{bmatrix} x_1 & \cdots & x_n \\ g_1 & \cdots & g_n \end{bmatrix} \triangleq \nabla\left(w_1\begin{bmatrix} x_1 & \cdots & x_n \\ g_1 & \cdots & g_n \end{bmatrix}\right)$

Substituibilidade - Seja x qualquer variável (local ou global). Então a relação dada por "um termo t substitui globalmente x na fórmula w" (globsub(t,x,w)), é definida indutivamente sobre as fórmulas como segue:

1) Se w é uma fórmula atômica, então globsub(t,x,w) ocorre.

2) Globsub(t,x, $\neg$w) ocorre se e somente se globsub(t,x,w) ocorre.

3) Globsub(t,x,$w_1 \rightarrow w_2$) ocorre se e somente se globsub(t,x,w_1) e globsub(t,x,w_2) ocorrem.

4) Seja * qualquer operador temporal monádico. Então globsub(t,x,*w) ocorre se e somente se:
   - x $\notin$ livre(*w) ou
   - não existem ocorrências de variáveis locais em t_1 e globsub(t,x,w) ocorre.

5) Globsub(t,x,$w_1 \mathcal{U} w_2$) se e somente se:
   - x $\notin$ livre($w_1 \mathcal{U} w_2$), ou
   - não existem ocorrências de variáveis locais em t_1 e globsub(t,x,w_1) e globsub(t,x,w_2) ocorrem.

6) Globsub(t,x,$\exists$V:w) se e somente se:
   - x $\notin$ livre($\exists$V:w), ou
   - V $\notin$ var(t) e globsub (t,x,w) ocorre

Sendo assim, se globsub(t,x,w) ocorre, pode-se fazer a substituição global da variável x pelo termo t, ou seja, $w\begin{bmatrix} x \\ z \end{bmatrix}$ não implica no aparecimento em w de ocorrências de novas variáveis globais ligadas e de novas variáveis locais dentro do escopo dos operadores modais.

2) SISTEMA DE PROVA

A - AXIOMAS (Esquemas de Axiomas)

Sejam w_1 e w_2 fórmulas-bem-formadas nas formas abaixo. Então w e $\Box w$ são axiomas (onde $\Box w = \neg \Diamond \neg w$)

A1) w, onde w é uma instância de uma tautologia proposicional.

A2) $\Box(w_1 \longrightarrow w_2) \longrightarrow (\Box w_1 \longrightarrow \Box w_2)$

A3) $\Box w_1 \longrightarrow w_1$

A4) $\odot \neg w_1 \longleftrightarrow \neg \odot w_1$

A5) $\odot((w_1 \longrightarrow w_2) \longrightarrow (\odot w_1 \longrightarrow \odot w_2)$

A6) $\Box w_1 \longrightarrow \odot w_1$

A7) $\Box w_1 \longrightarrow \odot \Box w_1$

A8) $\Box(w_1 \longrightarrow \odot w_1) \longrightarrow (w_1 \longrightarrow \Box w_1)$

A9) $w_1 \mathcal{U} w_2 \longleftrightarrow w_2 \vee (w_1 \wedge \odot (w_1 \mathcal{U} w_2))$

A10) $w_1 \mathcal{U} w_2 \longrightarrow \Diamond w_2$

A11) t = t para qualquer termo t.

A12) $t_1 = t_2 \longrightarrow \phi(t_1,t_1) \longleftrightarrow \phi(t_1,t_2))$

Onde i) ϕ é uma fórmula de estado (sem operadores temporais)

ii) globsub$(t_2,t_1,\phi(t_1,t_2))$ ocorre.

A13) $\odot \mathcal{P}(t_1,t_2,...,t_n) \Longleftrightarrow \mathcal{P}((\odot t_1),(\odot t_2),...,(\odot t_n))$, para qualquer letra de predicado n-ário e termos $t_1,...,t_n$.

A14) $\odot f(t_1,t_2,...,t_n) = f((\odot t_1),(\odot t_2),...,(\odot t_n))$, para qualquer letra de função f e termos $t_1,t_2,...,t_n$.

A15) C = (⊙C), para qualquer símbolo constante global C.

A16) V = (⊙V), para qualquer símbolo variável global V.

A17) $(\exists V{:}w) \longrightarrow w \left[\begin{smallmatrix} V \\ t \end{smallmatrix} \right]$ onde globsub(t,V,w) ocorre.

A18) $\odot(\exists V{:}w) \longrightarrow (\exists V{:}\odot w)$

A19) $\nabla(w_1 \longrightarrow w_2) \longrightarrow (\nabla w_1 \longrightarrow \nabla w_2)$;

A20) $\Delta \nabla w \longleftrightarrow w$;

A21) $\nabla w \longrightarrow w$;

A22) $\nabla \odot w \longrightarrow \odot \nabla w$

A23) $\Box \Diamond \Delta \left[\Lambda_{i=0}^{k} \odot^{(i)} \nabla w_i\right] \longrightarrow \Diamond \left[\Lambda_{i=0}^{k} \odot^{(t)} \nabla w_i\right]$

A24) Se w é uma fórmula que é verdade sob uma valoração pretendida, então w e $\Box$ w são axiomas.

A25) $\Box [\delta = \phi \longrightarrow (\odot x) = x]$, onde x é um símbolo variável local.

B - REGRAS DE INFERÊNCIA

REGRAS PRIMITIVAS

R1) Modus Ponens - MP

$$\frac{w_1, w_1 \rightarrow w_2}{w_2}$$

R2) $\forall$ - Inserção ($\forall$I)

$$\frac{w_1 \rightarrow w_2}{w_1 \rightarrow (\forall V : w_2)} \text{ onde } V \in \text{livre } (w_1)$$

R3) ∇ Inserção (∇I)

$$\frac{w}{\nabla w}$$

Dedução - w é uma dedução de um conjunto de fórmula ϕ (notação: $\phi \vdash w$), se e somente se existe uma seqüência finita de fórmulas $w_1, \ldots, w_n$ tal que $w_n = w_1$, e cada w_i é uma das possibilidades seguintes:

i) é um axioma,
ii) é uma fórmula de ϕ ou
iii) obtida de fórmulas anteriores por uma regra de inferência

Teorema - w é um teorema (notação:($\vdash w$), se e somente se $\{\} \vdash w$, onde $\{\}$ representa o conjunto vazio.

TEOREMAS E REGRAS DERIVADAS

Os teoremas e as regras derivadas aqui enunciados são os mesmos que aqueles que se encontram em [23], onde algumas adaptações são feitas para compatibilização com a GRTTL. Vale salientar que os sistemas de regras expostos em [28] e [36] são compatíveis com a GRTTL pelo fato desta última ser uma generalização daqueles dois primeiros. Pode-se usar ℓ_A e u_A como abreviatura das seguintes fórmulas, respectivamente, nas regras que se seguem:

ℓ_A = a abrevia $\quad \Box\left[\delta=\alpha \wedge t=T) \rightarrow \Diamond\ [\delta=\beta \wedge t\geq T+a)]\right]$

u_A = b abrevia $\quad \Box\left[\delta=\alpha \wedge t=T) \rightarrow \Diamond\ [\delta=\beta \wedge t\leq T+b)]\right]$

onde a e b são meta-variáveis sobre a sorte do tempo. ℓ_A e u_A representam, respectivamente, o tempo mínimo e máximo de permanência no estado A.

Teorema Lógico 1 (T1) - $\vdash w \longrightarrow \Diamond w$

Teorema Lógico 2 (T2) - $\vdash \odot w \longrightarrow \Diamond w$

Teorema Lógico 3 (T4) - $\vdash V\ w \longrightarrow \Diamond\Diamond w$

Teorema Lógico 4 (T7) - $\vdash \Box(w1 \wedge w2) \longrightarrow (\Box w1) \wedge (\Box w2)$

Teorema Lógico 5 (T44) - $\vdash (\exists V{:}w1 \wedge w2) \longleftrightarrow w1 \wedge (\exists V{:}w2)$ onde $V \notin$ livre(w1)

* Regra Derivada Limite Superior - Sejam w_1, w_2 e w_3 quaisquer fórmulas temporais e sejam d_1 e d_2 constantes quaisquer da sorte de tempo e T uma variável global da mesma sorte. Então:

$$\frac{\Box\left[(\forall T{:}(w_1 \wedge t=T) \rightarrow \Diamond\ (w_2 \wedge t\leq T+d_1))\right] \qquad \Box\left[(\forall T{:}(w_2 \wedge t=T) \rightarrow \Diamond\ (w_3 \wedge t\leq T+d_2))\right]}{\Box\left[(\forall T{:}(w_1 \wedge t=T) \rightarrow \Diamond\ (w_3 \wedge t\leq T+d_1+d_2))\right]}$$

* Regra Derivada $\Diamond Q$ -

$$\frac{w1 \longrightarrow w2 \qquad w2 \longrightarrow \Diamond w3 \qquad w3 \longrightarrow w4}{w1 \longrightarrow \Diamond w4}$$

* Regra Derivada ∃I- Seja V ∉ livre(w2), então

$$\frac{w1 \longrightarrow w2}{(\exists V{:}w1) \longrightarrow w2}$$

* Regra Derivada ER- Seja w' uma fórmula obtida de w pela substituição da ocorrência de uma subfórmula w1 em w por w2. Então:

$$\frac{w1 \longleftrightarrow w2}{w1 \longleftrightarrow w'}$$

* Regra Derivada D4-

$$\frac{w1 \longleftrightarrow w2}{\Box w1 \longleftrightarrow \Box w2}$$

3) SEMÂNTICA

As fórmulas da linguagem serão interpretadas em função de uma estrutura S composta de uma interpretação I e de uma trajetória σ, ou seja, $S=(I,\sigma)$. Seja RT-BSM $M=(S,E,s_0,\ell,p)$, então a interpretação I será dada pelo RT-BSM M. Deste modo, pode-se dizer que a estrutura S é formada por M e σ. A utilização de M como elemento que faz a interpretação é baseada na definição de $\ell{:}S \rightarrow F^*$ (onde F^* é o conjunto de todos os subconjuntos de F, o conjunto das fórmulas). Se a cada estado s está associado um conjunto de fórmulas F_s, e um estado é composto de todos os valores de constantes e variáveis (globais e locais), então o conjuntos de fórmulas pode ser associado com o conjunto de variáveis e constantes do sistema descrito pelas fórmulas.

Uma interpretação M (a interpretação I dada por M) será dada por um mapeamento cujo domínio é formado pelos parâmetros. Tem-se deste modo:

1 - M atribui um domínio D_j para cada símbolo quantificador $\exists_j$ de sorte j.

2 - M atribui para cada símbolo constante de cada sorte j um elemento em D_j.

3 - M atribui para cada símbolo variável global de cada sorte j um valor em D_j.

4 - M designa para cada símbolo predicado de sorte $(J_1,J_2,J_3,...,J_n)$ um predicado $P \subset Dj_1 \times Dj_2 \times ... \times Dj_n$.

5 - M atribui para cada símbolo de função de sorte $(J_1,J_2,...,J_n)$ uma função $f: Dj_1 \times Dj_2 \times ... \times Dj_{n-1} \rightarrow Dj_n$.

Para o modelo $M = (S,E,s_0,\ell,p)$ com seqüência $\sigma = s_0s_1s_2...$, e para qualquer inteiro $K \geq 0$, $M\sigma^{(K)}$ representa o modelo (S,E,s_{K1},ℓ,p) com seqüência $\sigma^{(K)} = s_Ks_{K+1}...$, e $M\sigma^{(0)} = M\sigma$. Na interpretação semântica, onde $S = (M,\sigma)$, denota-se a atribuição a um tempo t por uma estrutura S indutivamente, da seguinte maneira:

1 - S[C] como valor atribuído ao símbolo constante C por S.

2 - S[V] como valor atribuído ao símbolo variável global V.

3 - S[v] como valor atribuído a qualquer símbolo variável local v pelo estado inicial x_0 da trajetória σ de M.

4 - S[f] como a função atribuída ao símbolo f por S. Para qualquer símbolo de função n-ário f e quaisquer termos $t_1,...,t_n$

$$S\left[f(t_1,...,t_n)\right] = S[f]\left(S[t_1], S[t_2],...,S[t_n]\right)$$

5 - S[P] como predicado atribuído ao símbolo P por s. Para qualquer símbolo de predicado n-ário P e quaisquer termos $t_1,...,t_n$

$$S\left[P(t_1,...,t_n)\right] = S[P]\left(S[t_1], S[t_2],...,S[t_n]\right)$$

6 - Para qualquer termo da forma $(\odot t)$, onde t é um termo

$$S[\odot t] = S^{(1)}[t] = M\alpha^{(1)}[t]$$

Diz-se que uma fórmula-bem-formada w é satisfeita por uma estrutura S, representado por $\models^{s} w$, se w preeenche os seguintes requisitos:

i) $\models^{s}(t_1=t_2)$ se e somente se $S[t_1] = S[t_2]$ (t_1 e t_2 da mesma sorte).

ii) $\models^{s}\rho(t_1,t_2,...,t_n)$ se e somente se $(S[t_1],S[t_2],...,S[t_n]) \in S[P]$. Onde ρ é uma letra de predicado e $t_1,t_2,...,t_n$ são termos.

iii) $\models^{s} (\neg w)$ se e somente se não é o caso de $\models^{s} w$.

iv) $(w_1 \rightarrow w_2)$ se e somente se $\models^{s} w_2$ ou não é o caso de $\models^{s} w_1$.

v) $\models^{s}(\exists V{:}w)$ se e somente se existe, pelo uma estrutura $S'= (M',\sigma)$ tal que $\models^{s'} w$, onde M' é uma interpretação idêntica a M exceto pela atribuição que faz à variável global V, em questão.

vi) $\models^{s}(\odot w)$ se e somente se $\models^{s}{}^{(1)} w$

vii) $\models^{s}(\nabla w)$ se e somente se existe um $K \geq o$ tal que $S^{(K)} \models w$

viii) $\models^{s}(w_1 \mathcal{U} w_2)$ se e somente se para algum $K \geq 0$, $\models^{s}{}^{(K)} w_2$ e para todo $i \odot \leq C < K$, $\models^{s}{}^{(i)} w_1$

ix) $\models^{s}(\nabla w)$ se e somente se $\tilde{p}_{s0}$ $(\{\tau \in w_{s0}$ e $S \models w\}) = 1$, onde w_{s0} é o conjunto de todas as trajetórias de S^{*} que começam por s_0.

No item ix) acima vê-se que a satisfação de (∇w) não depende de instantes posteriores ao instante inicial, ou estado inicial. Isto demonstra que ∇ não é um operador temporal. Porém, dado o valor-verdade de w no instante considerado, não se pode saber o valor-verdade de (∇w) indicando assim que ∇ é um operador modal.

A interpretação intuitiva dos operadores temporais e do ∇ é a seguinte:

$\odot w$ - w será verdade no próximo instante de tempo (ou no próximo estado).

$\Diamond w$ - w será eventualmente verdade num estado futuro ou presente.

$w_1 \mathcal{U} w_2$ - w_2 será verdade em algum estado futuro, e w_1 será verdade (pelo menos) até lá.

∇w - com probabilidade 1, w será verdade no instante atual.

Para simplificar a manipulaçao de fórmulas, alguns adicionais são acrescentados ao sistema. São eles:

i)	$w_1 \vee w_2$	abrevia	$((\neg w_1) \rightarrow w_2)$	(disjunção)
ii)	$w_1 \wedge w_2$	abrevia	$\neg(w_1 \rightarrow (\neg w_2))$	(conjunção)
iii)	$w_1 \leftrightarrow w_2$	abrevia	$(w_1 \rightarrow w_2) \wedge (w_2 \rightarrow w_1)$	(bicondicional)
iv)	$(\Box w)$	abrevia	$(\neg \Diamond (\neg w))$	(daqui-por-diante)
v)	$w_1 \cup w_2$	abrevia	$(\Box w_1) \vee (w_1 \, \mathcal{U} \, w_2)$	(a-menos-que)
vi)	$w_1 \, \mathcal{P} \, w_2$	abrevia	$(\neg(\neg w_1) \, \mathcal{U} \, w_2)$	(precede)
vii)	$(\forall V{:}w)$	abrevia	$(\neg(\exists V{:}(\neg w)))$	(quantificador-existencial)
viii)	$(t_1 \neq t_2)$	abrevia	$(\neg(t_1 = t_2))$	(diferente ou não-igual)
ix)	(Δw)	abrevia	$(\neg(\nabla \neg(w)))$	(é possível)