DANILO DE PAULA E SILVA

Modelagem, Análise e Controle Supervisório de Sistemas Híbridos em uma Planta Piloto

Florianópolis
2004

Universidade Federal de Santa Catarina
Programa de Pós-Graduação em Engenharia
Elétrica

Modelagem, Análise e Controle Supervisório de Sistemas Híbridos em uma Planta Piloto

Tese submetida à
Universidade Federal de Santa Catarina
como requisito para a
obtenção do grau de Mestre em Engenharia Elétrica

Danilo de Paula e Silva

Florianópolis, maio de 2004.

Modelagem, Análise e Controle Supervisório de Sistemas Híbridos em uma Planta Piloto

Danilo de Paula e Silva

Esta Tese foi julgada adequada para obtenção do Título de Mestre em Engenharia Elétrica, Área de Concentração em *Controle, Automação e Informática Industrial*, e aprovada em sua forma final pelo Programa de Pós-Graduação em Engenharia Elétrica da Universidade Federal de Santa Catarina.

Prof. José Eduardo Ribeiro Cury, Dr.
orientador

Jefferson Luiz Brum Marques, Dr
Coordenador do curso de Pós-Graduação em Engenharia Elétrica
da Universidade Federal de Santa Catarina.

Banca Examinadora

Prof. José Eduardo Ribeiro Cury, Dr.
orientador

Prof. Daniel Juan Pagano, Dr.

Prof. Rafael Santos Mendes. Dr.

Dedico este trabalho aos meus pais, a minha irmã que com seus suporte e confiança muito contribuiram proporcionando-me concluir o maior feito de minha vida até o presente momento. E ao José Vítor, meu filho, que me impulsionou a empenhar de forma decisiva a realização do mesmo.

Agradecimentos

Agradeço a meus orientadores José E. Ribeiro Cury e Daniel Juan Pagano, pela oportunidade de trabalho, além de seus ensinamentos e incentivos ao longo do mestrado.

Ao amigo Carlos Damião, que teve participação decisiva na minha formação.

Aos colegas Antônio Eduardo Cunha, Marcos Vallin, Agustinho Plucênio, André Leal, Gustavo Bouzon, Ricardo Santos que contribuíram de forma direta.

A todas as amizades que fiz na sala do Programa de Recursos Humanos nº 34 - Agência Nacional de Petróleo e durante o mestrado.

A todos que contribuíram indiretamente para o desenvolvimento desta dissertação.

Resumo da dissertação apresentada à UFSC como parte dos requisitos necessários para obtenção do grau de Mestre em Engenharia Elétrica.

Modelagem, Verificação e Controle Supervisório de Sistemas Híbridos em uma Planta Piloto

Danilo de Paula e Silva

Março/2004

Orientador : José Eduardo Ribeiro Cury
Área de Concentração : Automação e Sistemas
Palavras-chave : Sistemas Híbridos, Sistemas Condição Evento, Controle Supervisório, Planta Piloto
Número de Páginas : 113

Recentemente o DAS teve aprovado o projeto CTPETRO "Sistemas de Controle Baseados em Redes Industriais Tipo Fieldbus para o Setor de Petróleo e Gás", que visa o desenvolvimento de soluções para problemas que afetam o setor de Petróleo e Gás. No contexto deste projeto, foi adquirida uma planta piloto cujo processo consiste de várias malhas de controle de fluxo, nível e temperatura, implementadas mediante sensores e atuadores digitais industriais, que se comunicam entre si e com uma estação de supervisão e controle via rede fieldbus. O presente documento trata da resolução de um problema proposto na planta piloto através da abordagem de sistemas híbridos e controle supervisório de sistemas híbridos. A primeira parte deste documento relata uma revisão bibliográfica de conceitos sobre sistemas híbridos, verificação de sistemas híbridos e controle supervisório de sistemas híbridos. A segunda parte do documento propõe a resolução de um problema proposto na planta piloto. Para a resolução deste problema, é feito um modelo que representa todos os comportamentos possíveis da planta piloto através de um autômato híbrido. Para este modelo é aplicado a teoria de controle supervisório com o intuito de resolver o problema proposto. Resolvido o problema teoricamente, é feito na planta piloto a implementação de um supervisor através de um algoritmo constituido de um jogador de autômatos e de uma interface homem-máquina para supervisão em tempo real. Os resultados computacionais são derivados das ferramentas Chekcmate e Grail e os resultados práticos derivados do software Matlab.

Sumário

Lista de Figuras

Lista de Tabelas

Capítulo 1

Introdução

No ano de 2002 o DAS-Depto de Automação e Sistemas da Universidade Federal de Santa Catarina teve aprovado o projeto CTPETRO (Sistemas de Controle baseados em Redes Industriais Tipo Fieldbus para o Setor de Petróleo e Gás), o qual visa o desenvolvimento de soluções para problemas que afetam o setor de Petróleo e Gás. No contexto deste projeto, foi adquirida uma planta piloto composta por várias malhas de controle de fluxo, nível e temperatura, implementadas mediante sensores e atuadores digitais industriais, que se comunicam entre si e com uma estação de supervisão e controle via rede fieldbus.

O processo em questão tem aplicação direta no setor de petróleo e gás, incluindo-se, com especial ênfase, a automação das refinarias de petróleo, plataformas de extração *off-shore*, supervisão e controle de oleodutos, etc. Dentro deste projeto pretende-se contemplar o estudo de sistemas contínuos controlados por sistemas de supervisão discreta, o que caracteriza um sistema de dinâmica híbrida. Para isto pretende-se considerar a planta piloto em questão, que possui dinâmicas contínuas representadas pelas variáveis fluxo, vazão e nível e dinâmicas discretas (sinais discretos gerados por um alarme indicando temperatura alta, por exemplo) atuando em certas condições que as variáveis contínuas porventura podem alcançar, caracterizando a planta piloto como uma planta híbrida.

Tendo em vista a natureza híbrida da planta, pretende-se modelar o mesmo através do formalismo de autômatos híbridos (Alur 1993) e utilizar a ferramenta computacional *CheckMate* (Silva e Chutinan 2000) para implementar o modelo e analisar propriedades do mesmo através de especificações ACTL. Tal formalismo é um dos vários modelos que representam *sistemas híbridos*.

Após a modelagem da planta, deseja-se projetar um supervisor discreto que resolva um problema proposto. Este supervisor será projetado segundo a abordagem de con-

trole supervisório de sistemas híbridos (Cury e Niinomi 1998). Por fim, pretende-se implementar este supervisor discreto na planta piloto através de uma algoritmo feito no Matlab conectado a um servidor OPC da planta piloto. OPC é uma tecnologia capaz de unificar fonte de dados com aplicações cliente, ou seja, é uma interface de comunicação homem-máquina [1].

Sistemas híbridos (Antsaklis 2000a), (Antsaklis 2000b) em geral, são sistemas que possuem tanto dinâmicas contínuas quanto dinâmicas discretas. Tais sistemas tem sido utilizados em diversas aplicações: na aviação, através de controle de tráfego aéreo e dos sistemas de controle de aeronaves, tais como o sistema de prevenção de colisões de aeronaves; em aplicações automotivas, como por exemplo os sistemas de controle de tráfego em rodovias e os sistemas de controle em sistemas de manufatura; na robótica; na eletrônica de potência; no controle de processos químicos e nucleares em geral; entre muitas outras.

Vários modelos de sistemas híbridos tem sido propostos. Neste documento iremos considerar três modelos, denominados de *autômato híbrido* (Alur 1993), *sistema híbrido de eventos de limiar* e *sistemas condição evento*. O primeiro é uma generalização do autômato discreto finito (Hopcroft e Ullman 1979) que inclui dinâmicas contínuas em cada estado discreto denominado de locação, do autômato. Transições entre as locações são determinadas por condições chamada de guardas, nos estados contínuos. O autômato híbrido será utilizado para a modelagem de um modelo híbrido para a planta piloto.

O sistema híbrido de eventos de limiar (Chutinan 1999) consiste em três subsistemas: sistema contínuo chaveado, com entradas discretas constantes que selecionam as dinâmicas contínuas com saídas contínuas; gerador de eventos de limiar que recebe as saídas contínuas do sistema contínuo chaveado e gera eventos quando estas atravessam certos limiares e máquina de estados finito que é o sistema de transição puramente discreto com um número finito de estados. Os sistemas híbridos de eventos de limiar são atrativos para o documento em questão. Isso se deve ao fato de que a ferramenta *CheckMate* utiliza esta modelagem para o procedimento de verificação (Silva e Chutinan 2000).

Sistemas condição evento (C/E) (Gonzáles e Krogh 2001) fornecem uma forma de se definir sistemas em tempo contínuo através da interconexão de subsistemas com sinais de entrada e saída discretos. Sendo assim, os sistemas serão apresentados através de diagramas de blocos e fluxo de sinais discretos. Vale ressaltar que embora esteja sendo usado aqui para modelagem de sistemas híbridos, o formalismo de sistemas C/E

[1] É uma tecnologia de linguagem de alto nível capaz de monitorar com facilidade equipamentos conectados a uma rede de dados. Exemplo: softwares supervisórios (Process View, In Touch, etc), Painel View, entre outros

pode ser utilizado também para a modelagem de Sistemas a Eventos Discretos (SED) (Leal 2002). Através do sistema C/E iremos obter a solução de um problema proposto na planta piloto através do formalismo de controle supervisório de sistemas híbridos (Gonzáles 2000).

1.1 Objetivos do Trabalho

A principal motivação para a análise e modelagem da planta híbrida é solucionar problemas práticos sob o aspecto de controle supervisório de sistemas híbridos (Gonzáles 2000). Em particular o tema em questão ainda está pouco difundido entre os pesquisadores brasileiros. Com isso, os principais resultados obtidos estão sob enfoque teórico, estendendo metodologias e abordagens de pesquisadores pioneiros na área de controle supervisório de sistemas híbridos.

Os principais objetivos deste trabalho são:

- Fazer uma revisão bibliográfica de sistemas híbridos e de controle supervisório de sistemas híbridos
- Estudar o processo correspondente a uma planta piloto em escala de laboratório, com múltiplas malhas de controle de temperatura, nível e vazão e propor uma modelagem matemática da planta utilizando o formalismo de autômatos híbridos. Isso permite a visualização de quais são os possíveis comportamentos que a planta piloto pode realizar;
- Implementar o modelo da planta piloto na ferramenta *CheckMate*. O *CheckMate* pode gerar um autômato de estados finitos que represente o comportamento lógico aproximado do sistema híbrido (Silva e Krogh 2000). Esta construção é um ponto crítico do trabalho, pelo fato de que pode ser extremamente difícil ou até mesmo impossível obter o autômato de estados finitos devido a problemas de verificação. Tais problemas serão tratados no capítulo 3.
- Construir especificações para o comportamento desejado do sistema sob supervisão e projetar um supervisor sob o enfoque de controle Supervisório de Sistemas Híbridos.
- Implementar o supervisor na planta piloto através de um algoritmo que se comporta como um jogador de autômatos conectado às variáveis da planta através de uma conexão servidor-cliente OPC. Analisar os resultados e verificar a validação dos formalismos sistemas híbridos e controle supervisório de sistemas híbridos no experimento prático.

1.2 Organização do Documento

Este documento está organizado como segue:

O capítulo 2 apresenta uma introdução aos sistemas híbridos. Faz um breve histórico e apresenta algumas abordagens existentes nas perspectivas das comunidades de controle e de ciência da computação.

O capítulo 3 faz uma discussão sobre dois modelos de sistemas híbridos: o autômato híbrido e o sistema híbrido de eventos de limiar. Faz-se uma breve introdução sobre verificação e por fim apresentamos alguns aspectos importantes da ferramenta *CheckMate*.

O capítulo 4 trata do controle supervisório de sistemas híbridos. Nele é feito a formulação de um problema de controle supervisório de sistemas híbridos apresentando uma classe de sistemas híbridos (sistemas C/E) que será utilizada ao longo deste documento.

O capítulo 5 trata da descrição e utilização da planta piloto. Descreve suas principais características, os softwares utilizados e sua conexão com a interface Matlab. Propõe um problema para ser resolvido na planta piloto. Trata a modelagem matemática da planta piloto, descrevendo as equações utilizadas, validação dos parâmetros destas equações, linearização em certos pontos de operação, cálculo das funções de transferência e a representação destas equações no espaço de estados.

O capítulo 6 trata da síntese de um supervisor discreto para resolver um problema proposto. Para tanto, utiliza-se a abordagem de autômatos híbridos e constrói o modelo da planta no *CheckMate*, o qual é o programa utilizado para obter um autômato discreto que é uma aproximação externa do comportamento lógico do sistema híbrido. Após a obtenção do modelo discreto do sistema híbrido, apresenta-se algumas especificações discretas de controle, segundo o problema proposto e calcula-se o supervisor utilizando o formalismo de controle supervisório de sistemas híbridos descrito no capítulo 4.

O capítulo 7 trata da implementação do supervisor calculado no capítulo 6 na planta piloto. Será implementado um algoritmo capaz de interpretar o autômato discreto da planta livre sob ação de um supervisor na planta piloto e os softwares utilizados para a implementação. Por fim os resultados desta implementação

O capítulo 8 é a conclusão do documento e discussão de possíveis trabalhos futuros.

Capítulo 2

Sistemas Híbridos

Este capítulo trata de sistemas híbridos sob um aspecto geral. É de suma importância o entendimento desta abordagem, pois o trabalho que será relatado neste documento se baseia neste formalismo. Primeiramente define-se dois tipos de sistemas: os sistemas contínuos e os sistemas a eventos discretos.

Um sistema é dito ser *contínuo* se seus estados variam continuamente com o decorrer do tempo, ou seja, as variáveis do sistema dependem do tempo. De um modo geral, os sistemas contínuos pode ser representados por equações diferenciais quando o tempo é contínuo. Quando o tempo é amostrado em períodos, tais sistemas podem ser representados por equações a diferenças. Em ambos os casos, a variável tempo (t no caso contínuo e k no caso amostrado) é naturalmente uma variável independente que aparece como argumento das funções entrada, estado e saída. Exemplo de um sistema contínuo pode ser um veículo em movimento, onde se tem a variação da velocidade e distância percorrida por unidade de tempo.

Outros sistemas, de natureza bastante distinta dos sistemas contínuos, são os chamados *Sistemas a Eventos Discretos* (SED). A princípio iremos apresentar o conceito de evento. Um evento pode ser identificado com uma ação (aperto de uma tecla), uma ocorrência espontânea como um relâmpago ou o resultado de várias condições satisfeitas em um dado instante (um equipamento finaliza uma operação). Um sistema a eventos discretos (SED) é um sistema a estado discreto dirigido por eventos, ou seja, sua evolução de estado depende integralmente da ocorrência de eventos discretos assíncronos no tempo. Um exemplo de um SED é o jogo de damas. A movimentação de uma peça pode ser descrita como um evento, onde os jogadores efetuam seu lance quando querem, ou seja, de forma assíncrona com o tempo e sem intermediações, onde há uma configuração das peças antes de um movimento e outra depois de mover a peça.

Há sistemas que possuem característica puramente contínua e outros com carac-

terísticas puramente discretas. Contudo, há sistemas que possuem tanto aspectos contínuos, quanto discretos. Tais sistemas eram tratados de acordo com a característica que mais se destacava. Em certos casos, não é razoável fazer tal consideração. O sistema então deve ser tratado utilizando os conceitos de sistemas contínuos e de sistemas a eventos discretos.

Tais sistemas são chamados de *sistemas híbridos*. Genericamente, o termo híbrido se refere à mistura ou junção de dois tipos de objetos ou metodologias fundamentalmente diferentes, como um carro bicombustível. Este carro pode ser abastecido tanto por álcool como por gasolina, tendo portanto, uma propriedade híbrida em relação aos carros convencionais. Os sistemas híbridos aos quais nos referimos neste trabalho são sistemas que possuem dinâmicas contínuas e dinâmicas de eventos discretos, sendo que, além de coexistirem, também interagem entre si. Não apenas o sistema pode ter uma característica híbrida, mas a especificação sobre o comportamento desejado para o mesmo também pode ser híbrida. Um exemplo de um sistema híbrido pode ser uma panela de pressão. Se a panela for utilizada de forma correta, a pressão irá ficar em uma faixa ideal para cozinhar os alimentos. Contudo, se esta pressão for aumentado no decorrer do tempo (falha na válvula de escape por exemplo) vai chegar a um ponto que a panela irá explodir. Tal explosão não depende do tempo e sim do valor máximo de pressão que a panela pode suportar. A explosão da panela pode ser considerado um evento que ocorreu quando a pressão atingiu um certo valor.

Na próxima seção é feito um breve histórico sobre sistemas híbridos. Na seção 2.2 são apresentadas as principais abordagens existentes para o tratamento de problemas relacionados aos sistemas híbridos sob as perspectivas das comunidades de controle e ciências da computação. Na seção 2.3 são citados alguns exemplos de aplicação de sistemas híbridos e por fim é apresentada a discussão do capítulo. Este capítulo foi baseado em (Leal 2002), (Antsaklis e Koutsoukos 2002) e (Krogh 2002).

2.1 Breve histórico

Muito embora o estudo de sistemas híbridos tenha se tornado popular apenas recentemente, vários tipos de sistemas que recaem nesta categoria já foram estudados anteriormente. Sistemas dinâmicos descontínuos foram objeto de estudo sistemático na antiga União Soviética e em outros países do leste Europeu por um longo período, iniciando no final dos anos 40.

O recente interesse e atividades sobre sistemas híbridos tem sido motivado em parte por resultados de pesquisa de desenvolvimento no controle de sistemas a eventos discretos (SED) que ocorreu nos anos 80, no desenvolvimento do controle adaptativo nos

anos 70 e 80 e no interesse renovado na formulação do controle ótimo em sistemas de dados amostrados e controle digital.

Em paralelo houve um crescente interesse pelos sistemas híbridos entre os cientistas da computação e estudiosos de lógica. O advento de máquinas digitais fez com o que os sistemas híbridos se tornassem muito comuns, pois sempre que um equipamento digital interage com mundo contínuo, o comportamento envolve fenômenos híbridos que precisam ser analisados e compreendidos.

Os primeiros encontros sobre sistemas híbridos ocorreram no início dos anos 90. Organizado por Robert L. Grossas e Anil Erode, o primeiro *workshop* sobre sistemas híbridos foi realizado junto ao Instituto de Ciências Matemáticas da Universidade de Cornell, nos EUA, em 1991. Em 1992 foi realizado o segundo *workshop* sobre sistemas híbrido na Universidade Técnica de Lingy, na Dinamarca. A partir daí o tema despertou grande interesse junto à comunidade científica.

Atualmente há importantes *workshops* sobre sistemas híbridos no qual destacamos o *Hybrid Systems: Computation and Control* (HSCC) onde podemos adquirir informações a respeito de publicações dos principais trabalhos e perspectivas sobre sistemas híbridos.

2.2 Abordagens para Sistemas Híbridos

Tendo em vista a ampla área de abrangência dos sistemas híbridos, muitas foram as abordagens e os enfoques dados ao seu estudo nos temas modelagem, análise, síntese e simulação. A seguir, divide-se as diferentes abordagens nas perspectivas das comunidades de controle e ciência da computação.

2.2.1 Perspectiva da Teoria de Controle

Na comunidade de controle, abordagens de modelagem, análise e projeto de controladores para sistemas híbridos estenderam a teoria de sistemas dinâmicos para incluir modos discretos. Em realidade, pode-se estender o controle com estrutura variável, controle por modos deslizantes, controle por relês, controle por chaveamento de ganhos e até mesmo controle nebuloso como exemplos de sistemas de controle híbrido. A característica comum a todos estes esquemas de controle é sua natureza de chaveamento; com base na evolução da planta (sistema de tempo contínuo a ser controlado) e/ou no progresso do tempo, o supervisor chaveia de um regime de controle para o outro (van der Shaft e Schumacher 2000).

A estrutura mais encontrada dentre a classe de sistemas chaveados é mostrada na figura 2.1. O supervisor decide qual dos controladores deve ser usado baseado nos sinais de entrada e saída da planta.

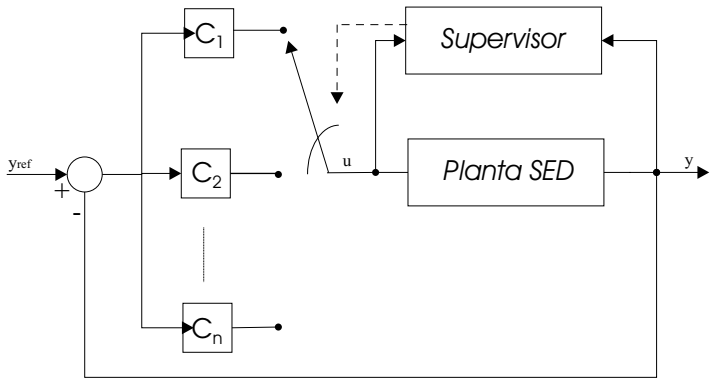


Figura 2.1: Estrutura de controle para sistemas chaveamos

Pelo fato que este tipo de estrutura tem sido utilizado com sucesso na prática há muitas décadas, vários pesquisadores voltaram a atenção para a sua modelagem e análise. Desde então, a comunidade de controle intensificou estudos sobre a extensão de definições e condições para a estabilidade de sistemas contínuos para o contexto de sistemas híbridos (Liberzon e Morse 1999).

Outro importante tema de pesquisa junto à comunidade controle tem sido o *controle supervisório de sistemas híbridos*. Inspirados na estrutura utilizada para o controle digital, alguns autores propuseram uma estrutura análoga para o controle supervisório de sistemas híbridos, na qual uma planta com características híbridas se comunica com um supervisor discreto através de uma interface formada por conversores A/D e D/A generalizados, conforme ilustrado na figura 2.2.

Nesta estrutura, a planta evolui ao longo do tempo até que uma variável de seu espaço de estados contínuo cruza determinado patamar, então a interface sinaliza a ocorrência de um evento para o supervisor. Este, por sua vez, atualiza seu estado discreto e, de acordo com o estado atual, envia um sinal discreto para a interface, a qual transforma este em um sinal contínuo a ser aplicado na planta, fazendo com o que a planta funcione em função do evento que o supervisor envia de volta para a planta.

Na metodologia de controle digital, pode-se realizar todo o projeto do controlador no domínio do tempo e então aproximar ou emular o controlador por um equivalente discreto da planta tomada em conjunto com a interface, para então realizar o projeto do controlador no domínio discreto. A analogia feita no contexto de controle supervisório de sistemas híbridos consiste em, ao invés de obter um modelo discreto, obter um modelo de eventos discretos da planta em conjunto com a interface como mostra a figura

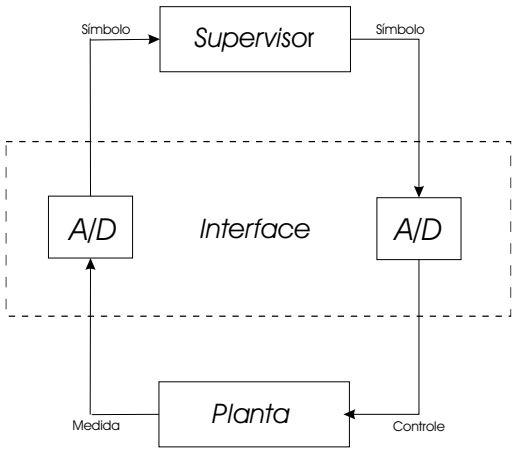


Figura 2.2: Estrutura genérica para o controle supervisório

2.3, utilizando autômatos ou redes de Petri. Assim, na perspectiva do supervisor, o conjunto planta-interface é um sistema a eventos discretos, pois recebe e envia apenas sinais discretos e então o controlador é projetado usando metodológicas de controle supervisório de SED.

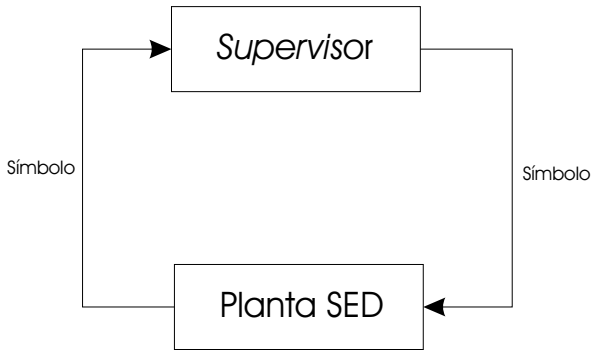


Figura 2.3: Estrutura genérica para o controle supervisório

A estrutura apresentada na figura 2.2 foi proposta no trabalho (Antsaklis e Lemmon 1993) e depois foi considerada em vários trabalhos posteriores por estes autores (Stiver e Lemmon 1995a) ,(Stiver e Lemmon 1995b). Abordagens similares baseadas em aproximações da planta por um SED foram utilizados por (Nerode 1993). Abordagens similares baseadas em aproximações da planta por um SED foram utilizadas em (Moor e Raish 2001), (Raish 1998) e (Cury e Niinomi 1998).

2.2.2 Perspectiva de Ciência da Computação

No final dos anos 80 (Wonham e Ramadge 1987), com a expansão de técnicas de controle supervisório de SED, cientistas da computação buscaram estender modelos puramente discretos de forma a incluir dinâmicas contínuas. Como resultado inicial desta extensão foram propostos os **autômatos diferenciais** (Tavernini 1987) e os **diagramas de estados híbridos** (Maler e Pnueli 1992). A seguir foram propostos os **autômatos temporizados** (Allur e Dill 1994), em cujos estados discretos foram inseridas dinâmicas do tipo relógio (na forma $\dot{x} = 1$). Assim, o comportamento contínuo resultante é crescente com uma taxa constante e o mesmo pode ser reinicializado nos instantes de ocorrência de eventos discretos. Em um passo posterior, foram introduzidos os **autômatos híbridos** (Alur 1993), os quais não possuem restrição de monotonicidade e permitem equações diferenciais do tipo linear e não linear.

A maior parte da pesquisa desenvolvida junto à comunidade de ciências da computação tem sido direcionada para o contexto de verificação formal utilizando a técnica chamada de *model checking* (Alur e Dill 1990), a qual testa exaustivamente todas as trajetórias do sistema com o intuito de testar algumas propriedades dos sistemas híbridos, quanto à técnica dedutiva de prova de teoremas (Manna e Sipma 1998), que prova a especificação através da indução sobre todas as trajetórias. Entre os avanços mais significativos estão os resultados extensivos sobre divisibilidade em problemas de verificação para várias classes de sistemas híbridos (Puri e Varaiya 1994).

Neste trabalho será utilizada uma abordagem similar ao da fig.2.2. Contudo, o sistema híbrido será modelado sob a perspectiva da ciência da computação utilizando o modelo autômato híbrido (Alur 1993) e a supervisão deste sistemas será feita utilizando a abordagem de (Cury e Niinomi 1998). Tal metodologia consiste no formalismo de sistemas condição evento (Sreenivas e Krogh 1991) que se baseia na definição formal de blocos individuais de tempo contínuo, os quais possuem entradas e saídas que são interconectadas de forma a criar modelos de sistemas híbridos através de diagrama de blocos.

2.3 Aplicações de Sistemas Híbridos

Os sistemas híbridos proporcionam um rico contexto para a definição de vários tipos de problema de controle. Enumera-se a seguir, algumas das aplicações nas quais são utilizadas abordagens de sistemas híbridos.

- aplicações automotivas, tais como em sistemas de rodovias e automóveis inteligentes (Silva e Engell 2001)

- Controle de processos químicos (Lennartson e Pettersson 1996)
- redes de distribuição de energia (Ferrari-Trecate 2002)
- sistemas de segurança crítica , tais como processos nucleares, sistemas de controle de tráfego aéreo e sistemas de controle de aeronaves (Livadas 2000)
- sistemas de manufatura (Pepyne e Cassandras 2000)
- eletrônica de potência (Miranda e Lima 2001)
- robótica (Egerstedt 2000)

2.4 Discussão

Viu-se que os sistemas híbridos são adequados para representar uma grande diversidade de aplicações, envolvendo problemas de análise e síntese e que portanto, muitos são os enfoques e as abordagens propostas para tratar cada um destes problemas. Atualmente, o tema vem despertando grande interesse junto às comunidades de controle e ciência da computação, o que tem levado a um grande fluxo de publicações sobre o mesmo. Ressalta-se que o conceito de sistemas híbridos tratado neste capítulo é utilizado ao longo do presente documento pelo fato de que a planta piloto citada no cap. 1 possui propriedades híbridas. As características da planta piloto bem como suas propriedades híbridas serão detalhadas no cap. 5.

Capítulo 3

Sistemas Híbridos: Modelagens e Verificação

Neste capítulo iremos fazer um breve estudo sobre dois modelos de sistemas híbridos que serão utilizados ao longo deste documento: o autômato híbrido e os sistemas híbridos de eventos de limiar. Na seqüencia iremos definir conceitos importantes utilizando a noção de sistemas de transição para o procedimento de verificação. Por fim, introduz-se a ferramenta *CheckMate* e uma breve discussão do capítulo.

3.1 Autômato Híbrido

Um autômato híbrido é uma generalização do *autômato discreto finito* (Hopcroft e Ullman 1979) que possui dinâmicas contínuas bem como transições discretas. Cada estado discreto é denominado *locação*. Associado com cada locação são as dinâmicas contínuas descritas por equações diferenciais e um conjunto *invariante* contínuo. O autômato híbrido pode residir em uma determinada locação tanto tempo quanto o estado contínuo permanecer dentro do seu invariante. As transições discretas entre locações ocorrem de acordo com as condições guarda e *reset* no espaço de estados contínuo. Transições discretas podem ocorrer quando a condição guarda é satisfeita e a condição *reset* deve estar satisfeita pelo estado contínuo depois que a transição discreta ocorreu. As condições guarda e *reset* podem ser representadas como um conjunto onde o estado contínuo deve pertencer a este conjunto para satisfazer cada condição. Introduziremos a notação seguinte que é usada para definir as dinâmicas contínuas para cada locação do autômato híbrido.

Seja $F_{A,B}$ o conjunto das funções que mapeiam cada membro de A nos subconjuntos de B, isto é:

$F_{A,B} = \{F : A \to 2^B\}$

A definição do autômato híbrido segue o formalismo de (Chutinan 1999).

Definição 3.1 *Um autômato híbrido é uma tupla* $H = (X, X_0, F, E, I, G, R)$ *onde:*

- $X = X_C \times X_D$ onde $X_C \subseteq R^n$ é o estado de espaço contínuo e X_D é um conjunto finito de locações discretas.
- $X_0 \subseteq X$ é o conjunto de estados iniciais.
- $F : X_D \to F_{X_C,R^n}$ associa $u \in X_D$ a uma função $F_u : X_C \to 2^{R^n}$ que descreve a inclusão diferencial[1] $\dot{x} \in F_u(x)$.
- $I : X_D \to 2^{X_C}$ associa u a um conjunto *invariante* na forma $I(u) \subseteq X_C$.
- $E \subseteq X_D \times X_D$ é um conjunto de transições discretas.
- $G : E \to 2^{X_C}$ associa $e = (u, u') \in E$ a um conjunto guarda na forma $S \subseteq I(u)$
- $R : E \times X_C \to 2^{X_C}$ designa para cada par $(u, u') \in E$ e $x \in G((u, u'))$ um conjunto de estados *resets* $S \in I(u')$, $R((u, u'), x) \subseteq I(u')$.

Um exemplo de um autômato híbrido, adaptado de (Chutinan 1999) é mostrado na fig.3.1 O autômato híbrido modela um termostato. As locações *on* e *off* correspondem aos estados on e off respectivamente. O estado contínuo x é a temperatura. O sistema inicia com a temperatura entre 29º C e 30º C. A transição de off para on pode ocorrer em qualquer ponto que satisfaça o guarda. O não determinismo irá permanecer enquanto a trajetória contínua de x estiver satisfazendo o invariante da locação off . No instante que chega no limite do invariante de off, a transição e1 ocorre obrigatoriamente e o termostato liga (estado on). Quando a temperatura passar de 40º, a transição e2 pode ocorrer em qualquer ponto pertencente a este guarda , isso até atingir 42º, que é a fronteira do invariante da locação on, então o termostato volta necessariamente a ficar desligado (estado off) e o processo se repete. O não determinismo existe porque há uma infinidade de pontos em que a condição guarda é satisfeita. Por exemplo, e1 pode ocorrer quando $x = 29º$, mas em outro momento nada impede que e1 ocorra em x=29.5º, pois este ponto também pertence ao guarda de e1. No entanto, se a trajetória x atingir a fronteira do invariante, obrigatoriamente e1 deverá ocorrer (se este não ocorreu antes), pois a temperatura x não poderá permanecer fora do invariante.

[1]São equações diferenciais definidas por intervalos. Ex: $\dot{x} \in [1, 8]$. O diferencial de x pode assumir qualquer valor compreendido entre os valores 1 e 8

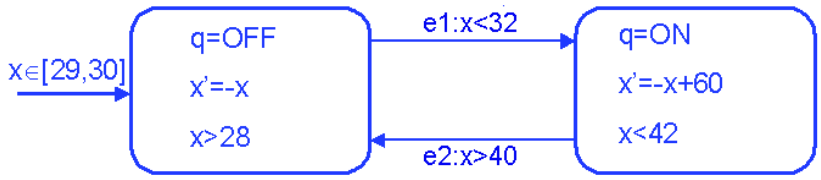


Figura 3.1: termostato

3.1.1 Autômato Híbrido Invariante-Poliedral

Definição 3.2 *Um Autômato Híbrido Invariante-Poliedral (PIHA)*[2] *é um autômato híbrido* $H = (X, X_0, F, E, I, G, R)$ *com as seguintes restrições*

- Para cada $u \in X_D$, F associa a para u um vetor de campo determinístico $f_u : X_C \to X_C$.
- Para cada $u \in X_D$, $I(u)$ é um poliedro convexo.
- Para cada $e = (u, u') \in E$, o conjunto guarda $G(e)$ é a união das faces de $I(u)$.
- Para cada $e = (u, u') \in E$ e $x \in G(e)$, $R(e, x) = \{x\}$, o mapa do *reset* é sempre um mapa identidade.
- O conjunto de estados iniciais deve ser da forma $X_0 = \cup_i (P_i, u_i)$ onde cada $P_i \subseteq I(u)$ é um politopo e $u_i \in U$. Aqui, a notação (P, u) significa o conjunto $\{(x, u) \in X | x \in P\}$
- I, G, e E deve satisfazer a seguinte condição.

$$\forall e = (u, u') \in E, G(e) \subseteq I(u').$$

O PIHA é equivalente a chaveamento de dinâmicas contínuas no qual não há "*jumps*" na trajetória de estados contínuos.

3.2 Sistema Híbrido de eventos de limiar

Sistema Híbrido de eventos de limiar (TEDHS)[3] consiste de três tipos de subsistemas, o *sistema contínuo chaveado* (SCS), o *gerador de eventos de limiar* (TEG) e

[2] A abreviatura utilizada é do nome em inglês Polyedral-Invariant Hybrid Automata

[3] As abreviaturas utilizadas são dos nomes em inglês: threshold event-driven hybrid systems, switched contínuos system threshold event generator e finite state machine respectivamente.

a *máquina de estados finito* (FSM). Sem perda de generalidade assumimos que há somente um subsistema como mostra a fig.3.2. Subsistemas múltiplos do mesmo tipo podem ser reduzido a um subsistema simples por representação dos sinais de entrada e saída em vetores. O sinal de entrada *u(.)* para o SCS é um sinal constante por partes, contínuo à direita e com limites à esquerda , assumindo valores sobre um conjunto finito de condições U. Este sinal é gerado pelo estado do FSM, de onde as transições entre os estados são forçadas por sinal evento *v(.)*, um sinal que assume valores não nulos apenas em instantes isolados do tempo no conjunto $\{0,1\}^{n_v}$ gerado pelo TEG[4]. A função de transição de estados w da FSM é não determinístico (por causa das incertezas do sistema). O conjunto de eventos que representa as transições na FSM possui valores não nulos para *v(.)*, isto é, valores pertencentes ao conjunto de eventos de limiar $V = \{0,1\}^{n_v} - \{0\}^{n_v}$.

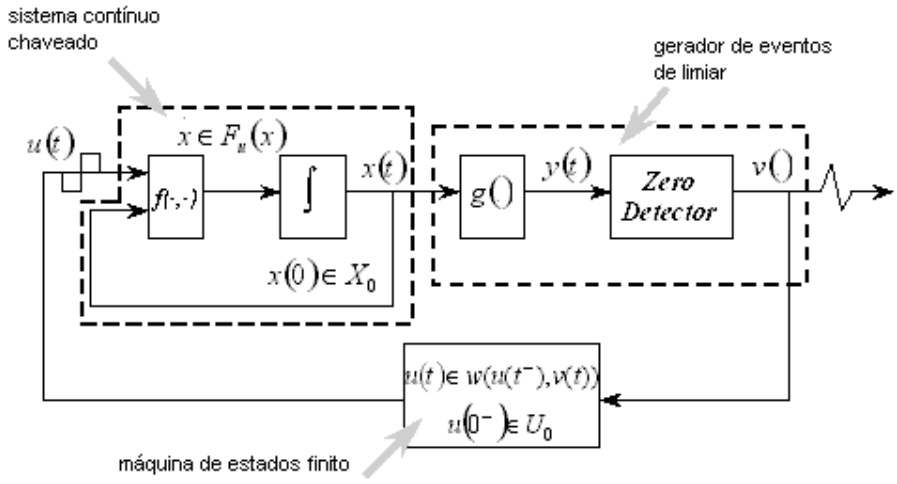


Figura 3.2: Diagrama de blocos ilustrando um Sistema híbrido de eventos de limiar

As dinâmicas contínuas do SCS são definidas por uma trajetória de estados $x(.)$, que evolui em $X = \mathbb{R}^{n_x}$ e que satisfaz uma equação de estados da forma $\dot{x}(t) = f_{u(t)}x(t)$, onde o sinal discreto $u(.)$ seleciona a equação de estados para a dinâmica contínua em cada instante. Similar à função de transição w não determinística para a FSM, a equação de estados é usada para modelar as dinâmicas contínuas. Os estados iniciais do SCS e da FSM são dados por $x(0) \in X_0 \subseteq \mathbb{R}^{n_x}$ e $u(0^-) \in U_0 \subseteq U$, respectivamente.

Os eventos de limiar que constituem os valores não nulos de *v(.)*, são gerados a seguir. Cada função de saída g_i define um evento de superfície $M_i = \{x | g_i(x) = 0\}$ onde i é o número dos diversos eventos de superfície. A função $g(.)$ é aplicado no sinal $x(.)$ produzindo o sinal de saída $y(.)$. O sinal resultante $y(.) = g(x(.))$ é a entrada para o detector de zeros que gera o sinal *v(.)* de acordo com:

[4]Significa que o conjunto $\{0,1\}$ possui n_v dimensões, onde n_v é o número de eventos ocorridos.

$$v_i(t) = \begin{cases} 1 & \text{se } y_i(t) = 0 \wedge (\exists \Delta > 0)(\forall \delta \in (0, \Delta)) : y_i(t - \delta) < 0 \\ 0 & \text{de outra forma} \end{cases}$$

Note que o detector de zeros é direcional: um evento é gerado somente quando uma componente de $y(t)$ se aproxima de zero por um dos lados. Note também que um evento distinto é gerado quando mais de um limiar é atingido em um instante particular. Isto reflete o fato que quando limiares múltiplos são alcançados simultaneamente pelas componentes de $y(t)$, a informação sobre a localização do estado contínuo do sistema é conhecida com maior precisão quando qualquer um dos patamares é alcançado individualmente. Sendo assim, não existe a ocorrência simultânea de eventos de limiar.

O cenário de interesse é quando o sinal discreto $u(.)$ muda de valor em resposta aos eventos de limiar observados em $v(.)$. Então, um par de sinais $(u(.), v(.))$ é admissível somente se descontinuidades em $u(.)$ ocorrem somente quando $v(.)$ é não nulo. Isto poderia representar uma situação, por exemplo, onde o SCS está operando sob um controle de malha fechada com a FSM sendo o controlador e as transições de estados no FSM são guiados pelos eventos de limiar.

Assegura-se que $u(.)$ é um sinal constante pode ter número finito de chaveamento em um intervalo de tempo finito qualquer, assumimos que $v(.)$ é *não-explosivo* (o termo usado em processos literários de Markov), ou não nula.

3.3 Bissimulação e Sistema de Transição Quociente

Esta seção introduz algumas definições formais de *sistemas de transição*, *sistemas de transição quociente e bissimulação*, que são a base para o procedimento de verificação que iremos considerar. Mais detalhes estão em (Chutinan 1999).

Definição 3.3 *(Sistema de transição) Um sistema de transição T é uma tripla $T = (Q \rightarrow Q_0)$ onde Q é o conjunto de estados, $\rightarrow \subseteq Q \times Q$ é o conjunto de transições e $Q_0 \subseteq Q$ é o conjunto de estados iniciais.*

Dado q e $q' \in Q$, a notação $q \rightarrow q'$ indica que $(q, q') \in \rightarrow$. Um *caminho* do sistema de transição é uma seqüencia de estados de tamanho finito ou tamanho infinito com cada par de estados consecutivos satisfazendo a relação de transição.

Definição 3.4 *(Caminho) Dado um sistema de transição $T = (Q, \rightarrow, Q_0)$, uma*

seqüência de estados $\pi = q_0 q_1 \ldots \in Q^* \cup q^\omega$ *é denominado um* caminho *de* T *se* $q_i \rightarrow q_{i+1}$ *para todo* $i \geq 0$. *O caminho* π *é denominado um "*run*" de* T *se* $q_0 \in Q_0$.

Dado um conjunto de estados $P \subseteq Q$, os conjuntos de estados que podem alcançar P ou pode ser alcançado por P em um passo são denominados *pré-condição* e *pós condição* de P respectivamente. Estes conjuntos serão definidos formalmente a seguir.

Definição 3.5 *(Conjuntos Pré/pós-condição) Dado um sistemas de transição* $T = (Q, \rightarrow, Q_0)$ *e um conjunto* $P \subseteq Q$, *a pré-condição de* P, *denotado por* $Pre(P)$, *é definido como*

$$Pre(P) = \{q \in Q | \ \exists p \in P, q \rightarrow p\}.$$

e a pós-condição de P, denotado por $Post(P)$, *definido como*

$$Post(P) = \{q \in Q | \ \exists p \in P, p \rightarrow q\}.$$

Para a proposta de verificação, introduziremos uma função etiqueta que designa uma etiqueta para cada estado no sistema de transição. O sistema de transição com a função etiqueta é denominada de *sistema de transição etiquetado.*

Definição 3.6 *(sistema de transição etiquetado) Um* sistema de transição etiquetado *é uma tupla* $T = (Q, \rightarrow, Q_0, \pounds, L)$ *onde* $(Q, \rightarrow, Q_0)$ *é um sistema de transição,* $\pounds$ *é um conjunto contável de etiquetas e* $L : Q \rightarrow \pounds$ *é a função etiqueta.*

Dado dois sistemas de transição etiquetados T_1 e T_2 com o mesmo conjunto de etiquetas, T_2 é dito simular T_1 se para todo caminho de T_1, existir um caminho correspondente em T_2. Este conceito de simulação será formalizado a seguir.

Definição 3.7 *(Simulação) Seja* $T_1 = (Q_1, \rightarrow, Q_{01}, \pounds, L_1)$ *e* $T_2 = (Q_2, \rightarrow, Q_{02}, \pounds, L_2)$ *sistemas de transição etiquetados. Uma relação de simulação de* T_1 *para* T_2 *é uma relação binária* $\psi \subseteq Q_1 \times Q_2$ *tal que:*

1. Para todo $q_1 \in Q_1$, deve existir $q_2 \in Q_2$ tal que $(q_1, q_2) \in \psi$;
2. Se $(q_1, q_2) \in \psi$, e $q_1 \rightarrow_1 q_1'$, então existe q_2' tal que $q_2 \rightarrow_2 q_2'$ e $(q_1', q_2') \in \psi$;
3. Se $(q_1, q_2) \in \psi$, então $q_1 \in Q_{01} \Leftrightarrow q_2 \in Q_{02}$; e

4. (Simulação com etiqueta) Se $(q_1, q_2) \in \psi$, então $L_1(q_1) = L_2(q_2)$.

Dizemos que T_2 *simula* T_1 por ψ, utiliza-se a notação $T_1 \preceq_\psi T_2$, se ψ é uma relação de simulação de T_1 por T_2. T_2 é também chamado de uma ψ-simulação de T_1.

Definição 3.8 *(Bissimulação) Seja* $T_1 = (Q_1, \rightarrow_1, Q_{01}, \pounds, L_1)$ *e* $T_2 = (Q_2, \rightarrow_2, Q_{02}, \pounds, L_2)$ *sistemas de transição etiquetados. Uma relação de bissimulação entre* T_1 *e* T_2 *é uma relação binária* $\psi \subseteq Q_1 \times Q_2$ *tal que* ψ *é uma relação de simulação de* T_1 *por* T_2 *e* $\psi^{-1} \subseteq Q_2 \times Q_1$ *é uma relação de simulação de* T_2 *por* T_1.

Dizemos que os sistemas de transição T_1 e T_2 bissimulam mutuamente por ψ, utiliza-se a notação $T_1 \equiv_\psi T_2$, se ψ é uma relação de bissimulação entre T_1 e T_2.

Uma abordagem para encontrar uma bissimulação do sistema de transição etiquetado é o *sistema de transição quociente* (QTS). Um sistema de transição quociente é construído através de uma partição do espaço de estados sob o sistema de transição. A partição deve ser consistente com a função etiquetada em que não há dois estados no mesmo conjunto na partição que são permitidos ter diferentes etiquetas.

Relações de bissimulação são importantes para a verificação porque qualquer dois sistemas de transição que possuem uma relação de bissimulação são equivalentes em termos de alcançabilidade de estados. A verificação pode ser executada em qualquer um dos dois sistemas de transição e o resultado será idêntico. Isto tem uma implicação significante para a verificação de um sistema de transição de estados infinitos em que se pudermos encontrar uma bissimulação de estados finita, podemos executar a verificação na bissimulação de estados finitos e o procedimento de verificação está garantido que termina.

3.4 Verificação de Sistemas Híbridos

Sistemas de transição nos proporcionam a base para a formulação para o problema de verificação. Nesta seção, faremos uma breve introdução sobre a árvore de computação lógica (CTL)[5] (Chutinan 1999). Este formalismo é utilizado para expressar as especificações de sistemas de transição, em especial, para o QTS. Também nesta seção será tratada a verificação de sistemas híbridos utilizando bissimulação e QTS.

[5]A abreviatura utilizada é do nome em inglês computation tree logic

3.4.1 Árvore de computação lógica

Árvore de computação lógica é uma linguagem que é usada para especificar a evolução do sistema em termos de valores verdadeiros através de proposições atômicas ao longo dos caminho da CTL. Uma fórmula CTL consiste de operadores temporais e de quantificadores de caminhos . Operadores temporais especificam a evolução do sistema ao longo de um único caminho da árvore computacional. Há quatro operadores temporais básicos: **X**, **F**, **G**, e **U**. O operador temporal **X** ("próximo passo") assegura que a propriedade é garantida no próximo estado do caminho. **F** ("no futuro") assegura que a propriedade é garantida em algum estado no futuro (incluindo o estado atual). **G** ("Globalmente") assegura que a propriedade é garantida globalmente, isto é, em todos os estados ao longo caminho. O operador **U** envolve duas propriedades. A inserção $f\mathbf{U}g$ requer que g é garantido em algum estado no futuro (incluindo o estado atual) e que f é garantido em todo estado ao longo do caminho antes da ocorrência de g. Quantificadores de caminho **A** e **E** se é para todos os caminhos ou para alguns caminhos respectivamente. A tabela 3.1 resume os operadores CTL.

Quantificadores de caminho		**Operadores temporais**	
A	Para todos os caminhos	**F**	No futuro
E	Para algum caminho	**G**	Globalmente
		X	Próximo passo
		U	Até que

Tabela 3.1: Operadores básicos da CTL

Formalmente, a sintaxe da fórmula CTL f é dado pela semântica:

$$\textbf{Semântica CTL:} f \dashrightarrow ap, \sim ap,\ f\&f,\ f|f, \mathbf{AX}f, \mathbf{AF}f, \mathbf{AG}f, \mathbf{A}f\mathbf{U}f, \mathbf{EX}f, \mathbf{EF}f, \mathbf{EG}f, \mathbf{E}f\mathbf{U} \tag{3.1}$$

onde ap denota uma proposição atômica e $\sim$, & e | são os operadores lógicos *not*, *and* e *or* respectivamente. Há uma subclasse de CTL que é a ACTL, a qual é utilizada pela ferramenta *CheckMate*. Na seção 2.5 discutiremos sobre a ACTL.

3.4.2 Verificação utilizando bissimulação

O conceito para especificação de transições de estados finitos pode ser estendido para transições de estados infinito, tal que o traço discreto do sistema de transição seja T_H, onde H é um TEDHS ou um autômato híbrido. Proposições atômicas podem ser

designadas para cada estado de T_H para formar um sistema de transição etiquetado e uma árvore computacional para T_H (com incontável número de nós) pode ser obtida por desdobramento de seu grafo de transições. Uma especificação CTL pode ser interpretada como antes no caso de sistemas de transição de estados finitos. O problema aqui é que a computação feita pela fórmula CTL não terminará porque o espaço de estado de T_H é infinito.

Para retificar este problema, muitos autores consideram o problema encontrando uma bissimulação finita de T_H. Verificando propriedades da bissimulação é equivalente verificar as mesmas propriedades do sistema original. Então, se uma bissimulação finita for encontrada, a verificação pode ser executada na bissimulação, garantindo seu término. A figura 3.3a mostra a abordagem de verificação utilizando bissimulação. No entanto, uma bissimulação de estados finitos existe somente para algumas classes de sistemas híbridos (Chutinan 1999). Sabendo que uma bissimulação de estados finitos pode não existir para a maioria dos sistemas híbridos, o procedimento de calcular a bissimulação pode não terminar e o passo de verificação pode não ser executado. Mesmo que uma bissimulação de estados finitos exista, pode levar um longo tempo para o procedimento de calcular a bissimulação convergir para a solução. Na próxima seção, discutiremos uma alternativa para contornar este problema em certos casos.

3.4.3 Verificação utilizando sistemas de transição quociente

Nesta seção, observamos que os sistemas de transição quociente são, em geral, simulações sob o sistema de transição. Poderíamos então executar a verificação no sistema de transição quociente em qualquer interação do procedimento de bissimulação mais rápido que esperar pelo procedimento de bissimulação terminar. Se a propriedade é verificada, não precisaríamos refinar o sistema de transição quociente. Senão, outra interação de refinamento pode ser executado e a verificação pode ser atendida no novo QTS. Continuando este processo, em algumas vezes se é possível concluir ou não que o sistema híbrido satisfaz determinada propriedade antes que a bissimulação fosse encontrada. A figura 3.3b mostra a alternativa de abordagem de verificação usando QTS. A restrição aqui é que como o QTS são aproximações conservativas sob o sistema de transição, somente propriedades *universais* podem ser verificadas. No contexto de CTL, uma propriedade universal é uma propriedade que pode ser especificada como uma fórmula ACTL.

Resumindo, se pudermos verificar que todas a trajetórias no QTS satisfazem alguma propriedade, podemos concluir que todas as trajetórias do sistema híbrido também satisfazem as mesmas propriedades, ou seja, encontramos uma bissimulação para o QTS.

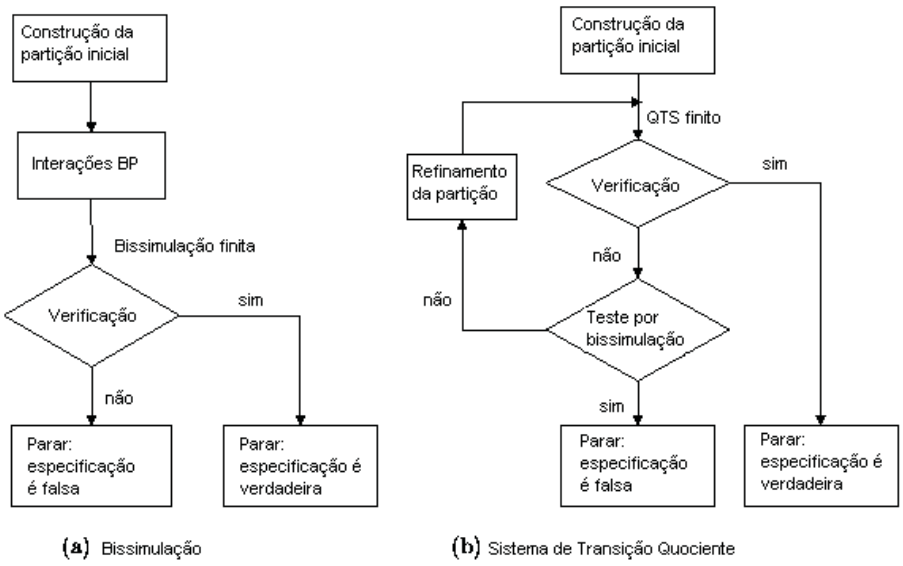


Figura 3.3: Abordagens de verificação

Para encontrar uma condição de bissimulação que termina a verificação, talvez seja necessário calcular um *refinamento da partição*. Na próxima seção, será feito uma introdução da ferramenta *CheckMate* e introduziremos o refinamento da partição que o *CheckMate* usa.

3.5 CheckMate

O *CheckMate* é uma ferramenta de verificação de sistemas híbridos. O *CheckMate* implementa uma técnica de verificação que é aproximar conservativamente o sistema híbrido original para um modelo discreto puro (Alur 1993). O processo de verificação no *CheckMate* é mostrado na figura 3.4.

A classe de sistemas híbridos que pode ser modelada no *CheckMate* é o TEHDS visto na seção 2.2. O procedimento de verificação inicia com a conversão do modelo TEHDS feito no ambiente Simulink em um modelo equivalente, que é o PIHA, tratado na seção 2.1. O PIHA resultante é discretizado na forma de um QTS, por uma partição inicial de acordo com as especificações do usuário. As transições feitas entre os estado no QTS são calculadas utilizando a técnica de análise de alcançabilidade contínua *Aproximações Flow Pipe* (Chutinan e Krogh 1999b). O QTS é então verificado pelas especificações ACTL utilizando a técnica padrão *Model Checking* (Alur e Dill 1990) para um sistema de transição de estados finitos. A verificação pode falhar devido ao fato do sistema híbrido aproximado pelo QTS ser modelado de uma maneira mais grosseira. A partição para o QTS pode ser refinado gerando uma aproximação mais exigente. Após o refinamento da partição, as aproximações *flow pipe* são utilizadas

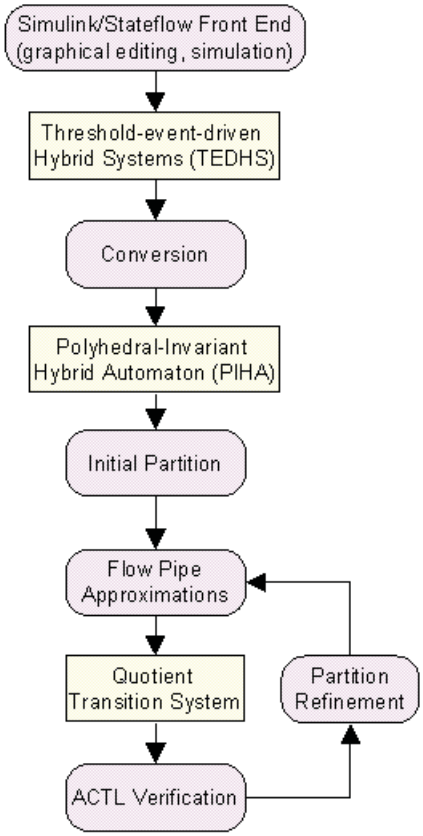


Figura 3.4: Procedimento de verificação do *CheckMate*

para redefinir as transições no QTS. O processo se repete até que o QTS satisfaça a especificação ou o usuário escolha encerrar a verificação.

Mais detalhes do *CheckMate* serão discutidos nas subseções subseqüentes.

3.5.1 Principais Blocos do *CheckMate*

Modelos do *CheckMate* são construídos sob a interface Matlab/Simulink utilizando duas máscara de blocos do simulink, além dos vários blocos padrão do Simulink. A seguir a descrição de cada bloco do Checkmate:

3.5.1.1 *Switched Continuous System* (SCS)

- Parâmetro: Função de Mudanças f

- Entrada: Sinal Discreto u

- Saída: Vetor de estados Contínuos x

- Descrição:Dinâmicas contínuas selecionadas pelo sinal de entrada discreto

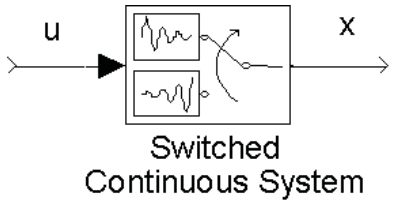


Figura 3.5: Bloco SCS

As dinâmicas contínuas do bloco SCS são da forma:

$$\dot{x} = f_u(x) \tag{3.2}$$

O SCS aceita três tipos de dinâmicas contínuas, que podem ser especificadas por cada valor da entrada discreta u. A tabela 3.2 ilustra os tipos de dinâmicas contínuas:

'clock'	Dinâmicas contínuas da forma $\dot{x} = c$ onde c é um vetor constante
'linear'	Dinâmicas contínuas da forma $\dot{x} = Ax + b$ onde A e b são vetores constantes $n \times n$ e $n \times 1$ respectivamente
'nolinear'	Dinâmicas contínuas da forma $\dot{x} = f(x)$

Tabela 3.2: Tipos de dinâmicas contínuas

3.5.1.2 *Polyhedral Threshold* (PTHB)

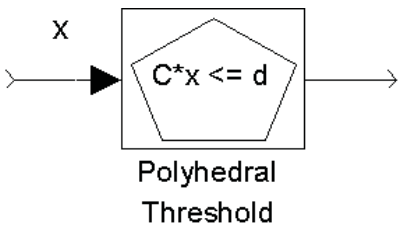


Figura 3.6: Bloco PTHB

- Parâmetro: C,d

- Entrada: Vetor de estados Contínuos x

- Saída: Sinal Booleano

- 1 se $Cx \leq d$, 0 se não

- Descrição: As saídas indicam quando x está dentro da região (poliedro $Cx \leq d$) ou não.

3.5.1.3 Bloco Máquina de Estados Finito (FSM)

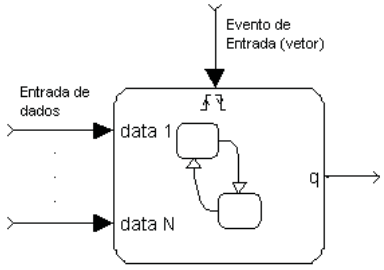


Figura 3.7: Bloco FSM

Dinâmicas discretas são modeladas usando um FSM. O FSM possui as seguintes propriedades:

- Entradas: podem ser eventos (vetores) ou dados escalares

- Dados de entrada devem ser funções booleanas do PTHB e FSMB produz saídas somente

- Eventos de entrada devem ser funções booleanas de saídas do PTHB somente

- Saídas: Sinal discreto (Inteiro) indicando estado ativo do FSM

Outros blocos do Simulink suportados pelo *CheckMate* inclui multiplexadores e blocos lógicos (AND, OR, NOT, etc.) para criar combinações lógicas de PTHB e saídas FSM.

3.5.2 Aproximações *Flow Pipe*

Para calcular um QTS para um PIHA, é necessário calcular o conjunto de estados alcançáveis sob as dinâmicas contínuas para uma locação dada do PIHA, denominado de aproximações *flow pipe*.

Para a dinâmica clock, as equações de todas as locações são da forma $\dot{x} \in \mathcal{F}$ onde $\mathcal{F}$ é um poliedro constante. Neste caso os conjuntos alcançáveis de qualquer poliedro inicial são obtidos pela projeção do poliedro inicial em todas as possíveis direções que as trajetórias do poliedro inicial podem alcançar. A fig. 3.8 mostra como funciona as aproximações flow pipe para a dinâmica clock.

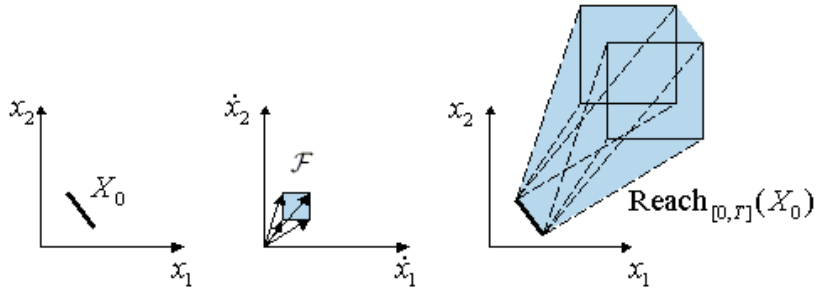


Figura 3.8: Método aproximações *flow pipe* para a dinâmica clock

No caso de dinâmicas não-lineares e lineares da forma $\dot{x} = Ax + b$ a técnica de aproximações flow pipe aproxima os conjuntos alcançáveis por uma seqüencia de poliedros convexos. O método inicia dividindo os conjuntos alcançáveis em *time segments* e aproxima cada conjunto alcançável de cada segmento por um encapsulamento de um poliedro convexo, denominado de politopo, ilustrado na fig. 3.9.

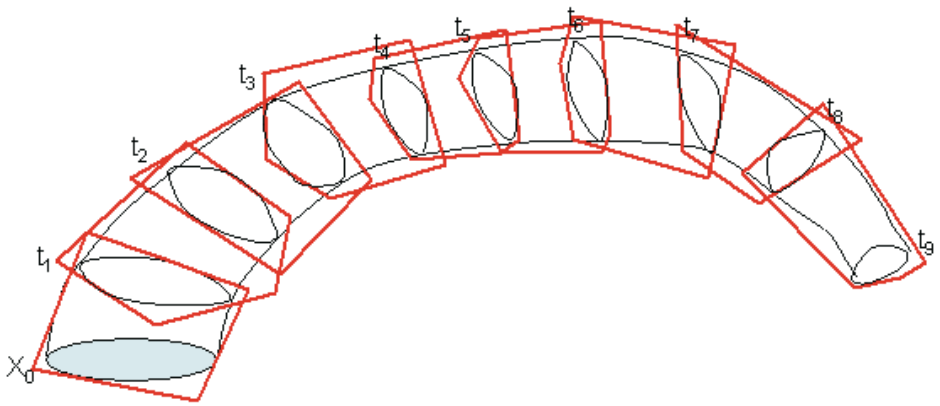


Figura 3.9: Método aproximações *flow pipe* para as dinâmicas linear e não linear

Mais detalhes sobre esta técnica podem ser encontrados em (Chutinan e Krogh 1999b).

3.5.3 Verificação ACTL

ACTL é uma subclasse do CTL. A sintaxe da fórmula ACTL f é dado pela semântica:

$$\textbf{Semântica CTL:} f \dashrightarrow ap, \sim ap,\ f\&f,\ f|f, \mathbf{AX}f, \mathbf{AF}f, \mathbf{AG}f, \mathbf{A}f\mathbf{U}f \tag{3.3}$$

Note que somente o quantificador universal **A** está presente na fórmula ACTL. Isto é devido ao fato que o *CheckMate* utiliza aproximações conservativas (QTS) para verificar propriedades do sistema híbrido. Concluímos que o sistema híbrido original satisfaz a especificação das aproximações conservativas somente quando uma especificação universal (uma especificação no qual requer que uma certa propriedade é garantida para todos os comportamentos do sistema) é usada. A seguir alguns exemplos de expressões ACTL.

- AG seguro: o sistema é seguro em todos os caminhos
- AG (AF reset): o sistema é resetado infinitamente ao longo de todo caminho computacional

Dado uma especificação CTL e um sistema de transição de estados finitos , o problema de verificação, também chamado de problema *model checking* é formalizado como:

Problema Model Checking: encontrar o conjunto de valores onde a fórmula CTL dada é verdadeira.

3.5.4 Refinamento da Partição

Neste tópico será discutido o refinamento da partição que é usado pelo *CheckMate* (Chutinan e Krogh 2001).

Se a verificação do QTS falhar devido uma discretização grosseira, o usuário tem a opção de refinar a partição, construir o próximo QTS, e uma nova tentativa de verificação. Para evitar que haja uma explosão de estados no refinamento, o *CheckMate* somente refina os estados que satisfazem os seguintes critérios:

- Estados com mais de um estado sucessor

- Estados iniciais que não satisfazem a especificação ACTL e todos os seus descendentes

Os critérios foram adotados pelas razões seguintes. Para o primeiro critério, não é preciso refinar um estado se este somente tem um único comportamento, sempre direcionado a um único caminho, de tal forma que não há ganho de nova informação. Para o segundo critério, recai no conceito que a especificação ACTL é universal. Então, todos os comportamentos dos estados iniciais que satisfazem a especificação estão no QTS são garantidos para satisfazer a especificação no sistema híbrido. Então, a verificação resultante não pode ser melhorada por refinamento desses estados iniciais e seus descendentes.

O esquema seguinte de refinamento utilizado pelo *CheckMate* para cada estado que foi refinado.

- Calcular os estados fonte, ou seja, o conjunto de estados que possuem transição para o estado corrente.
- Super aproximar o conjunto alcançável para cada estado fonte que está contido no mesmo invariante como os politopos do estado corrente através de um hiperretângulo (Isto é feito para reduzir o número de politopos que podemos processar enquanto mantém conservativamente a aproximação).
- Subtrai e interseciona cada hiperretângulo com o politopo associado a cada estado corrente. Isto separa o politopo para o estado corrente na parte que pode ser alcançada e a parte que é definitivamente inalcançável por cada estado fonte.

O esquema de refinamento é designado para dividir o politopo de cada estado em partes que podem ser alcançadas por diferentes combinações dos estados fontes.

3.6 Discussão

Neste capítulo apresentou-se uma introdução de conceitos importantes sobre o formalismo de sistemas híbridos. Estes conceitos estão implícitos no procedimento de verificação do *CheckMate* , outros são importantes para uma modelagem proposta na planta piloto através de autômatos híbridos. Tal modelagem será discutida no cap. 5. O importante para o leitor é fixar este formalismo para então facilitar o entendimento de como modelar uma planta híbrida e utilização do *CheckMate* .

Discutiu-se, ainda neste capítulo, a ferramenta *CheckMate* sob um contexto teórico. A utilização do *CheckMate* para o presente trabalho é gerar um autômato discreto

que represente todos os comportamentos possíveis da planta de acordo com o modelo autômato híbrido. Tal autômato híbrido foi modelado conforme sistema híbrido que será proposto no cap. 5.

Vale ressaltar que alguns conceitos como refinamento da partição do QTS foram citados de forma específica. Este procedimento é importante para o uso do *CheckMate* , não sendo o foco deste documento detalhar o amplo formalismo dos procedimentos de verificação de sistemas híbridos.

Capítulo 4

Controle Supervisório de Sistemas Híbridos

4.1 Sistema Condição Evento

Neste capítulo será feita a modelagem da planta híbrida sob o formalismo de sistema C/E. O experimento que será feito nos capítulos subseqüentes será baseado nesta abordagem. Sistema C/E pode ser aplicado tanto para modelagem de sistemas híbridos como para modelagem de SED (Leal 2002).

4.2 Formulação do Problema

Considera-se a classe de sistemas híbridos de tempo contínuo, $\mathcal{H}$, composta pela interconexão de um subsistema de dinâmica discreta, $\mathcal{H}_d$ e outro de dinâmica contínua, $\mathcal{H}_c$, conforme ilustrado na fig.4.1. Os aspectos discretos da planta híbrida são caracterizados por *sinais* condição e *sinais* evento (Sreenivas e Krogh 1991).

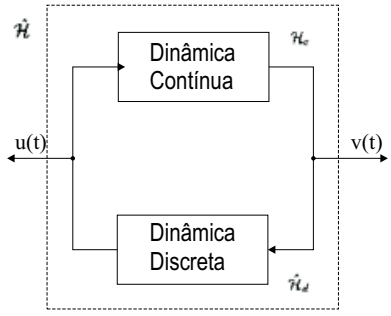


Figura 4.1: Planta híbrida $\hat{\mathcal{H}}$

O sinal de entrada para o subsistema contínuo, $\mathcal{H}_c$, é um sinal condição, *u(.)*, um sinal constante por partes, contínuo à direita e com limites à esquerda (figura 4.2(a)), assumindo valores sobre um conjunto finito de condições U. O espaço de todos os sinais condição *u(.)* em $[0, \infty)$ é denotado por $\mathcal{U}$. O sinal de saída do subsistema contínuo é um sinal evento, *v(.)*, um sinal que assume valores não nulos apenas em instantes isolados do tempo (ver figura 4.2 (b)). O espaço de todos os sinais evento *v(.)* em $[0, \infty)$ é denotado por $\mathcal{V}$.

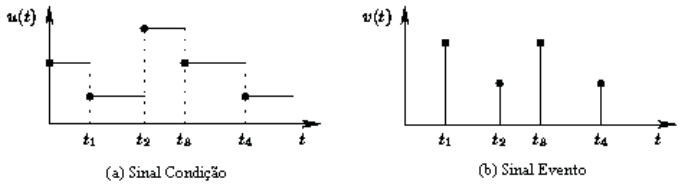


Figura 4.2: Representação dos sinais condição evento

O subsistema contínuo pode ser divido ainda em dois blocos funcionais: um sistema contínuo chaveado e um gerador de eventos de limiar conforme mostrado na figura 4.3.

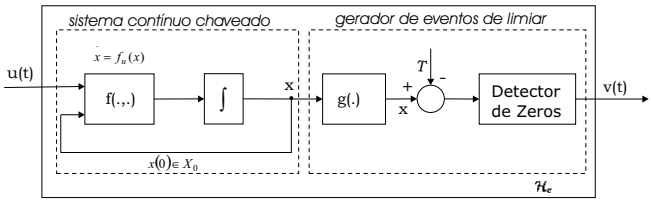


Figura 4.3: Subsistema contínuo $\mathcal{H}_c$

As dinâmicas contínuas da planta híbrida $\hat{\mathcal{H}}$ são definidas por uma trajetória de estados $x(.)$, que evolui em $X = \mathbb{R}^n$ e que satisfaz uma equação de estados da forma $\dot{x}(t) = f_{u(t)}x(t)$, onde $f_u : \mathbb{R}^n \to \mathbb{R}^n$ para todo $u \in U$. Assim, a seleção da dinâmica contínua(representada pela equação de estados) é feita a cada instante pelo sinal condição $u(t)$. O valor inicial da trajetória de estados, $x(0)$, pertence ao conjunto de estados iniciais $X_0 \subseteq \mathbb{R}^n$. O conjunto de todas as possíveis trajetórias de estado para um dado sinal de entrada $u(.) \in U$, iniciando em qualquer estado do conjunto $X' \subset X_0$, é denotado por $X_{u(.)}(X')$.

A função $g : X \to \mathbb{R}^m$ gera o sinal contínuo de saída, $y \in \mathbb{R}^m$, da trajetória de estados, ou seja, $y(.) = g(x(.))$. Cada componente do sinal contínuo de saída, y_i, é comparado a um patamar, T_i, definido pelos componentes de um vetor de patamares, $T \in \mathbb{R}^m$ e o sinal de saída evento é gerado por um *detector de zeros*, definido para $t > 0$ e $i = 1, 2, ..., m$ como:

$$= \begin{cases} 1 & \text{se } y_i(t) - T_i = 0 \wedge (\exists \Delta > 0) \\ & (\forall \delta \in (0, \delta)) : y_i(t-\delta) - T_i < 0 \\ 0 & \text{de outra forma} \end{cases}$$

Um evento de limiar ocorre nos instantes em que $v_i \neq 0$. Note que os sinais evento $v(.)$ assumem valores não nulos, isto é, valores pertencentes ao conjunto de eventos de limiar $V = \{0,1\}^{n_v} - \{0\}^{n_v}$.

Observe que na equação de $v_i(t)$ a condição dada por $y_i(t-\delta) - T_i < 0$ faz com o que um evento de limiar seja gerado somente quando a superfície limite for alcançada por um dos lados, ou seja, a sinalização do evento é unidirecional. Note que a inequação $y_i(t-\delta) - T_i < 0$ deve ser satisfeita para um $\delta > 0$, ou seja, para que um evento v seja gerado no instante t é necessário que $y_i - T_i < 0$ durante algum tempo Δ antes da superfície $y_i - T_i$ ser atingida. Essa restrição impede que eventos sejam "continuamente" gerados quando y_i desliza sobre a superfície por um certo instante de tempo.

O subsistema discreto $\hat{\mathcal{H}}_d$ é um sistema de dinâmica puramente discreta que mapeia, de forma não determinística, sinais evento $v(.) \in \mathcal{V}$ em sinais condição $u(.) \in \mathcal{U}$. Assume-se que $\hat{\mathcal{H}}_d$ pode mudar o sinal condição de entrada para $\mathcal{H}_c$ se e somente se um evento de limiar é observado. O subsistema discreto tem uma característica de possibilitar a adição de informações sobre as possibilidades de chaveamento, ou seja, $\hat{\mathcal{H}}_d$ permite modelar restrições sobre as possíveis entradas a serem aplicadas a $\mathcal{H}_c$. A dinâmica discreta é expressa por autômatos de Moore, sendo que as transições de estados representam os eventos e as saídas dos estados são as condições u (Cury e Niinomi 1998).

A seguir dá-se continuidade à modelagem da planta híbrida, utilizando-se para tal o formalismo de sistemas sistema C/E introduzido por (Sreenivas e Krogh 1991). Antes, porém, define-se o *produto cartesiano síncrono* de $\mathcal{V}$ e $\mathcal{U}$, denotado por $\mathcal{V} \otimes \mathcal{U}$, como o conjunto de todos os pares $(v(.), u(.)) \in \mathcal{V} \times \mathcal{U}$, tal que descontinuidades em $u(.)$ ocorrem apenas em instantes em que $v(.)$ é não nulo. Define-se ainda o conjunto $\mathcal{V}_m \subseteq \mathcal{V}$ com a classe de sinais evento $v(t)$ que têm a propriedade de possuir um número finito de descontinuidades em $t \in [0, \infty)$.

O subsistema contínuo $\mathcal{H}_c$ é definido como um subconjunto de $\mathcal{V} \otimes \mathcal{U}$, sendo que o par $(v(.), u(.)) \in \mathcal{H}_c$ se e somente se existe uma trajetória de estados $x(.) \in \mathcal{X}_{u(.)}(X_0)$ tal que o sinal evento de saída resultante é $v(.)$.

Descreve-se o comportamento discreto do sistema C/E $\mathcal{H}_c$ pelo seu modelo estado discreto. Para tanto, introduz-se a *representação de traço discreto* para $\mathcal{H}_c$ como uma 4 - tupla (W, β, ρ, W_0), descrita com segue. Considere o sinal condição $w(.)$, constante

por partes e contínuo à direita , que registra o valor correspondente da trajetória de estados $x(.)$ apenas nos instantes de descontinuidades em $(v(.), u(.)) \in \mathcal{H}_c$. O conjunto W é o conjunto de todos os valores que o sinal $w(.)$ pode assumir e W_0 é o conjunto dos possíveis valores iniciais para $w(.)$. A diferença entre $w(.)$ e $x(.)$ é que somente o segundo registra a trajetória entre as descontinuidades na entrada e saída do sistemas . Assim, W é o mesmo conjunto original de estados X, ou seja, $W = \mathbb{R}^n$ e $W_0 = X_0$. A função de transição $\beta : w \times u \to W$ para $w(.)$ possui a forma $w(t) = \beta(w(t^-), u(t^-))$ e é tal que:

$$\omega(t) = \begin{cases} \Phi_{u(t^-)}(t, \omega(t^-)) & \text{se para algum } i = 1, 2, ..., m, g_i(\Phi_{u(t^-)}(t, \omega(t^-))) - T_i = 0 \\ & \bigwedge(\exists \Delta > 0)(\forall \delta \in (0, \Delta)) : g_i(\Phi_{u(t^-)}(t - \delta, \omega(t^-))) - T_i < 0 \\ \omega(t^-) & \text{de outra forma} \end{cases}$$

onde $\Phi_u(t, x(t_0))$ é a solução da equação diferencial $\dot{x} = f_u(x)$ para $u \in U$, $t \geq t_0$ e valor inicial $x(t_0)$. Utiliza-se a notação t^- para simbolizar o limite pela esquerda ao instante t. Observe que as transições de estado (mudanças no valor de w) ocorrem apenas nos instantes em que um hiperplano é alcançada. A função de saída evento $\rho : W \times W \to V$ gera o evento de limiar correspondente no instante de transição do estado $w(.)$ e é nula em qualquer outro instante. Ela é definida como:

$$v(t) = \rho(w(t^-), w(t)) \tag{4.1}$$

De forma semelhante, o subsistema discreto é definido como $\hat{\mathcal{H}}_d \subseteq \mathcal{V} \otimes \mathcal{U}$. O modelo de estado discreto para $\hat{\mathcal{H}}_d$ é dado pela 4 - tupla $(Q, \hat{\delta}, \phi, q_0)$ onde q é conjunto discreto de estados, enumerável e possivelmente infinito, $q_0 = q(0^-)$ é o estado inicial e as funções de transição $\hat{\delta} : Q \times V \to 2^Q$ e de saída condição $\phi : Q \to U$ são definidas para $t \in [0, \infty)$ por:

$$q(t) \in \hat{\delta}(q(t^-), v(t)) \tag{4.2}$$

$$u(t) = \phi(q(t)) \tag{4.3}$$

Note que a ocorrência de um evento $v(.)$ a transição do estado $q(.)$ é feita de forma não determinística e que o sinal de saída ŭdo subsistema discreto $\hat{\mathcal{H}}_d$ depende apenas deste estado. Portanto, a escolha do sinal condição a ser aplicado ao subsistema contínuo, em resposta ao evento observado, é feita de forma não determinística por $\hat{\mathcal{H}}_d$. Desta forma, ao observar um evento $v(t)$, $\hat{\mathcal{H}}_d$ transita para o estado $q(t)$ e um sinal condição $u(t)$ é aplicado de forma instantânea.

4.2.1 Supervisão da planta Híbrida

Considere agora o esquema de controle supervisório da fig.4.4 para a classe de sistemas híbridos apresentada anteriormente nesta capítulo. Observe que neste modelo foi acrescida uma entrada m ao subsistema discreto $\hat{\mathcal{H}}_d$ e conseqüentemente, à planta híbrida $\hat{\mathcal{H}}$. Assim, denotaremos por $\mathcal{H}_d$ e $\mathcal{H}$ o subsistema discreto em malha fechada e a planta híbrida em malha fechada respectivamente, de forma a diferenciá-los dos seu correspondentes sem ação de controle. O supervisor $\mathcal{F}$ observa os sinais condição $u(.)$ e evento $v(.)$ da planta híbrida $\mathcal{H}$, em resposta, aplica uma entrada de controle $m(.)$ ao subsistema discreto $\mathcal{H}_d$ de forma a restringir a gama de possíveis condições $u(.)$ aplicadas ao subsistema contínuo $\mathcal{H}_d$. A entrada de controle é um sinal evento $m(.) \in \mathcal{M}$ tomando valores não-nulos no conjunto $M = 2^U$ e é interpretada como o conjunto das condições que $\mathcal{H}_d$ pode aplicar à $\mathcal{H}_c$. O cenário de interesse é quando $m(.)$ muda de valor somente quando ocorre uma descontinuidade no sinal evento $v(.)$. O supervisor pode gerar um sinal evento $m(t)$ se e somente se ele observar um evento de limiar $v(t)$, o que faz com o que as descontinuidades em $m(.)$ e em $v(.)$ e conseqüentemente em $u(.)$ sejam síncronas. No momento de ocorrência de um evento $v(t) \in V$ uma entrada de controle $m(t) \subseteq U$ é aplicada no supervisor e, estando $\mathcal{H}_d$ no estado $q(t^-)$, o conjunto das possíveis condições de entrada a serem aplicadas à $\mathcal{H}_c$ fica restrito à $m(t) \cap \phi(q(t)) \in U : q(t) \in \hat{\delta}(q(t^-)), v(t))$.

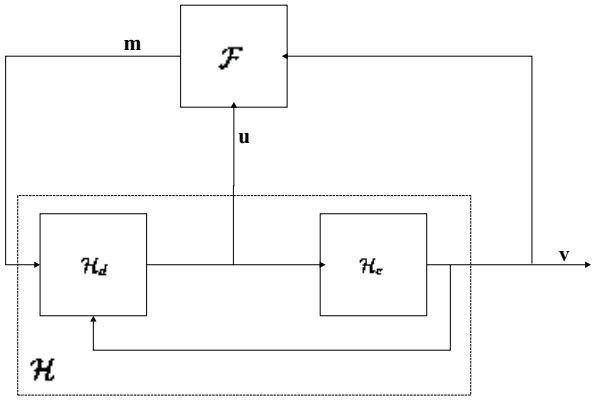


Figura 4.4: Esquema de controle supervisório para a planta híbrida.

O subsistema discreto com entrada de controle é definido como $\mathcal{H}_d \subseteq \mathcal{V} \otimes \mathcal{M} \otimes \mathcal{U}$, sendo que a única diferença entre o modelo de estado discreto de $\mathcal{H}_d$ em relação ao modelo para a versão em malha aberta reside na função de transição, definida agora na forma: $\delta : Q \times V \times M \rightarrow 2^Q$ para $t \in [0, \infty)$ como:

$$\delta(q(t^-), v(t), m(t)) = q(t) \in \hat{\delta}(q(t^-), v(t)) : \phi(q(t)) \in m(t) \tag{4.4}$$

Modelando a restrição imposta pela entrada de controle $m(t)$ às possíveis condições

que o subsistema discreto pode escolher no instante t de ocorrência do evento $v(t)$. A função de saída condição é a mesma que a do modelo sem entrada de controle, ou seja, $u(t) = \phi(q(t))$.

O sistema híbrido sob ação de um supervisor é dado por $\mathcal{H} \subseteq \mathcal{V} \otimes \mathcal{M} \otimes \mathcal{U}$, sendo que o mesmo apresenta uma característica não determinística oriunda do subsistema $\mathcal{H}_d$.

O supervisor $\mathcal{F}$ é definido como um subconjunto de $\mathcal{V} \otimes \mathcal{M} \otimes \mathcal{U}$ sendo que para $(v(.), u(.), m(.)), (v'(.), m'(.), u'(.)) \in \mathcal{F}$, tem-se que:

$$(v(.), u(.)) = (v'(.), u'(.))(\forall t \in [t_0, t_1) \wedge (v(t_1) = v'(t_1))) \Rightarrow m(t_1) = m'(t_1 \qquad (4.5)$$

Descreve-se o comportamento discreto do supervisor C/E $\mathcal{F} \subseteq \mathcal{V} \otimes \mathcal{M} \otimes \mathcal{U}$ pelo seu modelo discreto dado pela 4-tupla Z, ξ, ψ, z_0, onde Z é o conjunto de estados, $z_0 = z(0^-)$ é o estado inicial, $\xi : Z \times U \times V \rightarrow Z$ é a função de transição de estado e $\psi : Z \times Z \rightarrow M$ é a função saída evento, definidas para $t \in [0, \infty)$ como:

$$z(t) = \xi(z(t_-), u(t), v(t)) \qquad (4.6)$$

$$m(t) = \psi(z(t_-), z(t_-)) \qquad (4.7)$$

Observe que a transição de estado do supervisor ocorre de forma determinística e que é forçada pelo sinal evento $v(t)$. Observe ainda que a função de saída depende do estado alcançado nesta transição. Assim, no instante em que o supervisor observa o evento $v(t)$ proveniente da planta híbrida, ele muda seu estado discreto e aplica o sinal $m(t)$ ao sistema híbrido.

Finalmente, o fechamento da malha resulta em um sistema C/E autônomo $\mathcal{F}/\mathcal{H} \subseteq \mathcal{V} \otimes \mathcal{U}$ e seu modelo de estado discreto é obtido pela conexão cascata e realimentação (Sreenivas e Krogh 1991) dos modelos do supervisor $\mathcal{F}$ e do sistema híbrido $\mathcal{H}$.

A seguir, expressaremos o comportamento lógico dos sistemas em termos de linguagens.

4.2.2 Comportamento Lógico

Uma vez que o sistema C/E possui sinais de entrada e saída que assumem valores discretos no tempo, é possível associar a este um comportamento lógico, o qual é representado aqui por linguagens de palavras de comprimento finito para sistema C/E (Sreenivas e Krogh 1991).

Seja um sistema C/E $\mathcal{G} \subseteq \mathcal{V} \otimes \mathcal{U}$, por exemplo. A linguagem gerada de $\mathcal{G}$, denotada por $\mathcal{L}(\mathcal{G})$, é a linguagem definida sobre palavras de comprimento finito obtidas pelo prefixo-fechamento de todas as cadeias formadas pelas seqüencias dos valores que os pares de sinais (v(.),u(.)) $\in \mathcal{V} \otimes \mathcal{U}$ assumem em seus pontos de descontinuidade (sem registrar os instantes de ocorrência de tais descontinuidades). Vale lembrar que os sinais $v(.)$ $u(.)$ assume valores sobre os conjuntos discretos V e U, respectivamente. Portanto, para o sistema $\mathcal{G}$ tem-se que $\mathcal{L}(\mathcal{G}) \subseteq (V \times U)^*$. Na eq. 4.8 apresenta-se uma definição formal da linguagem $\mathcal{L}(\mathcal{G})$.

$$\begin{aligned} \mathcal{L}(\mathcal{G}) \quad = \quad & \{\omega = \omega^1 \circ \omega^2 \circ ... \circ \omega^n \in (V \times U)* : \exists (v(.), u(.)) \in \mathcal{V} \otimes \mathcal{U} \bigwedge \\ & t \geq 0 \bigwedge t_1 < t_2 < ...t_n < t, (v(t_i), u(t_i)) = \omega^i \bigwedge \\ & (v(t'), u(t')) = (0, u(t'))\forall t' \neq t_1, t_2, ...t_n, t' \leq t\} \end{aligned} \tag{4.8}$$

Conforme visto anteriormente, sinais condição são utilizados somente nos instantes de ocorrência de eventos. Desta forma, para possibilitar a representação da aplicação de um sinal condição inicial (antes que tenha ocorrido qualquer evento) associa-se a este um evento fictício denominado *evento de inicialização*, denotado por η. Portanto, η é um evento fictício que é associado à escolha não determinística do estado inicial $x(0)$ e que não está associado a transições de estado na planta, ou seja, $\eta \notin V$. O conjunto $V^+ = \{\eta\} \cup V$ denota a inclusão do evento de inicialização em V. Considerando-se ainda o sistema $\mathcal{G}$, tem-se então que $\mathcal{L}(\mathcal{G}) \subseteq (V^+ \times U)^*$, ou mais especificamente $\mathcal{L}(\mathcal{G}) \subseteq (\{\eta\} \times U)(V \times U)^*$, uma vez que eventos η são considerados apenas de inicialização da planta. O evento de inicialização η não só tem a função de inicialização, mas também é importante para sistema C/E que possuem vários estados iniciais. A inserção do evento η faz com o que estes sistemas tenham apenas um estado inicial.

Ao longo deste trabalho são tratadas duas classes de linguagens, a saber: linguagens em V^* e $(V^+ \times U)^*$. A seguir comenta-se sobre a utilização de cada uma destas linguagens.

- As especificações sobre os comportamentos desejados para o sistema híbrido sob

supervisão (em malha fechada), são expressas por linguagens $E \subseteq V^*$, ou seja, em termos de seqüencias de patamares atingidos no espaço de estados contínuo da planta híbrida. Assim, os controles u e U sob os quais os eventos $v \in V$ são gerados não fazem parte da especificação.

- Para descrever os comportamentos lógicos dos sistemas sem entrada de controle $(\mathcal{L}(\hat{\mathcal{H}}_d))$, $(\mathcal{L}(\mathcal{H}_c))$ e $(\mathcal{L}(\hat{\mathcal{H}}))$ e também para descrever o comportamento lógico do sistema híbrido sob supervisão $(\mathcal{L}((\mathcal{F}/\mathcal{H}))$ são utilizadas linguagens em $(V \times M \times U)^*$. A semântica de um evento $\sigma = vu \in \mathcal{L}(\hat{\mathcal{H}})$, por exemplo, é a seguinte: quando uma variável do espaço de estados contínuo do sistema híbrido $\hat{\mathcal{H}}$ atinge um certo patamar, o subsistema contínuo gera um evento de limiar $v \in V$, em resposta, o subsistema discreto aplica uma condição $u \in U$.

O comportamento lógico do sistema híbrido sob ação do supervisor é dado pela linguagem $(\mathcal{L}((\mathcal{F}/\mathcal{H})) \subseteq (V \times U)^*$. No entanto, as especificações sobre o comportamentos lógicos desejados para o sistema híbrido em malha fechada são feitas em termos de eventos, ou seja, por linguagens $E \subseteq V^*$. Sendo assim, é necessário definir a projeção $P_V : (V^+ \times U)^* \rightarrow V^*$, a fim de que se possa relacionar o comportamento real com o desejado para o sistema sob supervisão.

$$P_V(\sigma) = \begin{cases} \varepsilon & \text{para } \sigma = \eta u \in (\{\eta\} \times U) \\ v & \text{para } \sigma = vu \in (V \times U); \end{cases}$$

A projeção definida anteriormente também pode ser aplicada para linguagens $(V^+ \times U)^*$. Assim, a projeção de uma linguagem em $K \subseteq (V^+ \times U)^*$ é definida como:

$$P_V(K) = \{t \in V^* : (\exists s \in K) P_V(s) = t\} \tag{4.9}$$

Desta forma, a projeção $P_V : (V^+ \times U)^* \rightarrow V^*$ apaga os símbolos η e $u \in U$ de palavras em $(V^+ \times U)^*$.

Considere novamente um sistema $\mathcal{G} \subseteq (V^+ \times U)^*$. Define-se agora a linguagem marcada de $\mathcal{G}$, denotada por $\mathcal{L}_m(\mathcal{G})$, como o conjuntos das cadeias $\omega \in (V^+ \times U)^*$ formadas pelas seqüencias dos valores que os pares de sinais $(v(.), u(.)) \in \mathcal{V}_m \otimes \mathcal{U}$ assumem em seus pontos de descontinuidades e tais que $P_V(\omega)$ representa uma seqüencia de eventos de limiar que corresponde ao completamente de uma tarefa por parte do sistema híbrido .

Por fim apresentar-se o problema de síntese de supervisores para sistema híbrido .

Problema 4.1 *(Síntese de Supervisores para Sistemas Híbridos - SSSH) Seja H o*

modelo de C/E da planta híbrida com entrada de controle e dada as especificações $A, E \subseteq V^*$ *para o comportamento da planta em malha fechada, encontrar um supervisor C/E* $\mathcal{F}$ *consistente para* $\mathcal{H}$ *tal que:*

$$A \subseteq P_V[\mathcal{L}_m(\mathcal{F}/\mathcal{H})] \subseteq E \tag{4.10}$$

Onde A é a especificação do comportamento mínimo da planta em malha fechada. Na seção seguinte utiliza-se da teoria de controle supervisório de SED de modo a propor uma solução formal para o problema supra apresentado.

4.3 Abordagem por Controle de Sistemas a Eventos Discretos

Nesta seção, o problema de síntese de supervisores para sistema híbrido (SSSH) é traduzido para uma abordagem de controle puramente discreta. Mostra-se então como este problema pode ser solucionado através de um problema equivalente no domínio de SED. Esta seção descreve de forma sucinta algumas definições e teoremas de controle de SED. Maiores detalhes sobre esta abordagem estão em (Leal 2002), (Cury e Niinomi 1998), (Gonzáles 2000).

Antes de apresentar um modelo de SED para o sistema híbrido é necessário definir dois novos conjuntos. Seja uma linguagem $K \subseteq (V^+ \times U)^*$.

- O conjunto ativo de eventos em K após $\omega \in \overline{K}$ é dado por $V_K(\omega) = \{v \in V^+ : (\exists u \in U)\omega \circ vu \in \overline{K}\}$
- O conjunto ativo de condições em K após $\omega \in \overline{K}$ para um dado $v \in V_K(\omega)$ é definido como $U_K(\omega, v) = \{u \in U : \omega \circ vu \in \overline{K}\}$.

O modelo de sistemas a eventos discretos com estrutura de controle para o sistema híbrido com entrada de controle $\mathcal{H}$ é a tripla (L, L_m, Γ), onde:

$$\begin{aligned} L &= \mathcal{L}(\hat{\mathcal{H}}) \\ L_m &= \mathcal{L}_m(\hat{\mathcal{H}}) \end{aligned} \tag{4.11}$$

isto é, L e L_m são, respectivamente, as linguagens gerada e marcada pelo sistema

híbrido *sem entrada de controle* $\hat{\mathcal{H}}$ e a estrutura de controle $\gamma \in 2^{V^+ \times U}$é um mapa tal que para todo $\omega \in L$ tem-se que:

$$\Gamma(\omega) = \{\gamma \in 2^{V^+ \times U} :: (\forall v \in V_L(\omega))(\exists u \in U_L(\omega, v)) : vu \in \gamma\} \tag{4.12}$$

Note que a estrutura de controle depende da palavra gerada pelo sistema. Conforme definida, a estrutura de controle traduz a idéia de que um dado evento v, ativo após uma cadeia $\omega \in L$, deve sempre haver pelo o menos uma condição u habilitada. Repare que a inibição de todas as condições possíveis para um dado evento significaria que o supervisor não previu uma resposta para a ocorrência de uma dada cadeia, não havendo contrapartida física para essa situação.

Seja o autômato mostrado na fig.4.5, o qual reconhece a linguagem $L \subseteq (V \times U)$. Antes de mais nada vamos interpretar a linguagem do autômato . Para este exemplo, o conjunto V é representado por $V = \{v_1, v_2, v_3, v_4\}$ e o conjunto $U = \{u_1, u_2\}$. Começamos a análise pelo estado inicial, de onde sai a transição v_1u1. Este comportamento nos diz que o estado inicial só possui um único evento ativo (v_1) e que este evento está associado a apenas uma condição, no caso a u_1. Isto significa que a planta possui restrições próprias, que fazem com o que apenas um único evento (v_1) associado a uma única condição u_1 para o estado inicial. No estado 2 há apenas um evento ativo, que é o v_2. Porém, há duas condições associadas a este evento. Já no estado 3, a regra é a seguinte: há apenas 2 eventos ativos: v_3, v_4. O evento v_3 está associado as duas condições pertencentes ao conjunto U. Mas para o evento v_4, este está associado apenas a condição u_1.

Voltando agora para estrutura de controle, para $\omega = v_1u_1 \in L$ tem-se $\Gamma(\omega) = \{\{v_2u_1\}, \{v_2u_2 \circ v_3u_1, v_2u_2 \circ v_3u_2, v_2u_2 \circ v_4u_1\}, \{v_2u_2 \circ v_3u_1, v_2u_2 \circ v_4u_1\}, \{v_2u_2 \circ v_3u_2, v_2u_2 \circ v_4u_1\}\} = \{\gamma_1, \gamma_2, \gamma_3, \gamma_4\}$

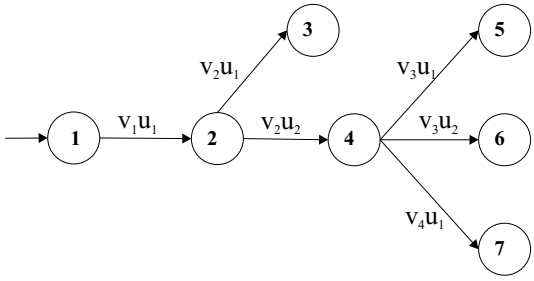


Figura 4.5: Autômato para a ilustração de Γ

Note que $\gamma = \{v_3u_1, v_3u_2\} \notin \Gamma(\omega)$ uma vez que não contempla o evento $v_4 \in V_L(\omega)$.

A proposição apresentada a seguir estabelece a equivalência lógica entre o modelo

C/E $\mathcal{H}$ e o modelo de eventos discretos $H = (L, L_m, \Gamma)$ para a planta híbrida com entrada de controle. É uma proposição adaptada por (Leal 2002) de forma a tratar de linguagens marcadas.

Proposição 4.1 *O modelo de eventos discretos* $H = (L, L_m, \Gamma)$ *e o modelo condição/evento* $\mathcal{H} \subseteq \mathcal{V} \otimes \mathcal{M} \otimes \mathcal{U}$ *para a planta híbrida com entrada de controle são logicamente equivalentes no sentido que:*

1. $(\forall \varpi \in (V^+ \times M \times U)^*), \varpi \in \mathcal{L}(\mathcal{H}) \Leftrightarrow P_{V\times U}(\varpi) \in L$;
2. $(\forall \varpi \in \mathcal{L}(\mathcal{H})(\forall vmu \in V^+ \times M \times U), \varpi \circ vmu \in \mathcal{L}(\mathcal{H}) \Leftrightarrow (\exists \gamma \in \Gamma(P_{V\times U}(\varpi))) : u \in m = \{u' \in U : vu \in \gamma\}$ *e*
3. $(\forall \mathcal{V} \in (V^+ \times M \times U)^*), \mathcal{V} \in \mathcal{L}(\mathcal{H}) \Leftrightarrow P_{V\times U}(\mathcal{V}) \in L_m$

O supervisor de eventos discretos F para a planta $H = (L, L_m, \Gamma)$ é definido pelo mapa $F : L \rightarrow 2^{V^+\times U}$ e é representado por uma máquina de estados $F = (P, V^+ \times U, \theta, p_0)$ onde P é o conjunto de estados, p_0 o estado inicial e a função de transição $\theta : V^+ \times M \times U \rightarrow P$ é tal que para $p \in P$ e $\sigma \in (V^+ \times U)$, $\theta(p, \sigma)$ é definida se e somente se para $\omega \in L$ tem-se que $\sigma \in F(\omega)$ e $\omega \in (V^+ \times U)^*$ tal que $\hat{\theta}(p_0, \omega) = p$ onde $\hat{\theta}$ consiste na extensão da função de transição para palavras em $(V^+ \times U)^*$.

A linguagem $L(F/H) \subseteq L$ representa o comportamento gerado da planta $H = (L, L_m, \Gamma)$ sob ação de um supervisor F, e é definida recursivamente como:

1. $\varepsilon \in L(F/H)$ e
2. $\omega \circ vu \in L(F/H) \Leftrightarrow \omega \in L(F/H) \wedge \omega \circ vu \in L \wedge vu \in F(\omega)$

O comportamento marcado de F/H é dado por $L_m(F/H) = L(F/H) \cap L_m$ e representa a parte da linguagem marcada da planta que sobrevive sob ação de controle.

Segundo (Leal 2002), um supervisor F é consistente para uma planta se para qualquer cadeia $\omega \in L(F/H)$ e para todo evento ativo após ω, há ao menos uma condição de entrada habilitada pelo supervisor pertencente a conjunto de condições admissíveis na planta.

A proposição seguinte sugere que o problema de síntese de supervisores para sistema híbrido (SSSH) pode ser solucionado através da resolução de um problema de controle supervisório equivalente no domínio de SED.

Problema 4.2 *(Problema do Controle Supervisório Equivalente - PCSE) Seja* $H = (L, L_m, \Gamma)$ *o modelo de SED com entrada de controle para a planta híbrida e dada as especificações* $A \subseteq E \subseteq V^*$ *para o comportamento da planta em malha fechada, encontrar um supervisor* F *consistente para* H *tal que:*

$$A \subseteq P_V[L_m(F/H)] \subseteq E. \tag{4.13}$$

No intuito de apresentar uma solução formal para este problema, apresenta-se a seguir uma série de resultados da teoria de controle supervisório de SED adaptando-so para o contexto deste trabalho (Leal 2002).

A seguir, apresentaremos algumas definições de controlabilidade de linguagens é uma adaptação ao nosso contexto da definição introduzida em (Gonzáles 2000)

Definição 4.1 *Sejam as linguagens* $K \subseteq L \subseteq (V^+ \times U)^*$. K *é dita ser vu-controlável em relação a* L *se:*

$$(\forall \omega \in \overline{K}) V_L(\omega) = V_K(\omega) \tag{4.14}$$

Assim, K é vu-controlável em relação a L se após qualquer cadeia $\omega \in \overline{K}$, todo evento possível de ocorrer em $\overline{K}$ também é possível de ocorrer em L.

Seja $\Sigma_K(\omega) = \{vu \in V^+ \times U : \omega \circ vu \in \overline{K}\}$ o conjunto ativo de condições/eventos em K após $\omega \in \overline{K}$. Seja ainda um modelo SED com estrutura de controle $H = (L, L_m, \Gamma)$. A condição apresentada na eq. 4.14 equivale à seguinte condição:

$$(\forall \omega \in \overline{K})(\exists \gamma \in \Gamma(\omega)) : \gamma \cap \Sigma_L(\omega) = \Sigma_K(\omega). \tag{4.15}$$

Ou seja, uma linguagem K é em relação a L se para toda cadeia $\omega \in \overline{K}$ existe pelo o menos um padrão de controle $\exists \gamma \in \Gamma(\omega)$ que habilite na planta o mesmo conjunto ativo de condições/eventos habilitados em K após ω.

Da mesma forma que no domínio das linguagens em $(V^+ \times U)^*$, podemos introduzir o conceito de controlabilidade para linguagens em V^*, mais detalhes sobre as definições nas linguagens em V^* estão em (Leal 2002). Entretanto, uma vez que o controle é feito através dos sinais condição, a controlabilidade de uma linguagem em V^* está diretamente associada à definição de $vu-controlabilidade$ apresentada anteriormente. Logo, a análise sobre a controlabilidade se dá no domínio $(V^+ \times U)^*$.

Segundo (Leal 2002) é possível provar que qualquer supervisor consistente é não

bloqueante para H. Segundo a mesma referência é possível provar que as propriedades de vu-controlabilidade e v-controlabilidade são fechadas para a união de conjuntos.

Logo, pode-se enunciar o seguinte lema:

Lema 4.1 *O conjunto $\mathbb{C}_{V\ U}(K)$ é não vazio, fechado para a união de conjuntos e contém um elemento supremo único, chamado "máxima linguagem vu-controlável", denotado por $Sup\mathbb{C}_{V\ U}(K)$.*

Proposição 4.2 *Sejam $H = (L, L_m, \Gamma)$ e $K \subseteq (V_+ \times U)^*$, $K \neq \emptyset$. Se $K = \overline{K} \cap L_m$ então existe um supervisor não bloqueante F para H tal que $L_m(F/H) = Sup\mathbb{C}_{V\ U}(K)$.*

A seguir iremos introduzir alguns lemas e proposições sobre o conjunto $E \subseteq V^*$

Seja o conjunto de todas as sublinguagens de E que são v-controláveis em relação L definido como:

$$\mathbb{C}_V(E) = \{E' \subseteq E : E \text{ é } v\text{-controlálvel em r a } L\} \tag{4.16}$$

Pode-se enunciar o seguinte lema:

Lema 4.2 *O conjunto $\mathbb{C}_V(E)$ é não vazio, fechado para a união de conjuntos e contém um elemento supremo único, chamado "máxima linguagem v-controlável", denotado por $Sup\mathbb{C}_V(E)$.*

Segundo (Leal 2002), é possível provar que os conjuntos $\mathrm{Sup}\mathbb{C}_{V\ U}(K)$ e $\mathrm{Sup}\mathbb{C}_V(E)$ são L_m-fechada e $P_V(L_m)$-fechada respectivamente.

Proposição 4.3 *Sejam $H = (L, L_m, \Gamma)$ e $E \subseteq V^*$, $E \neq \emptyset$.Se $E = \overline{E} \cap P_V(L_m)$ então existe um supervisor não bloqueante F para H tal que $P_V[L_m(F/H)] = Sup\mathbb{C}_V(E)$.*

Proposição 4.4 *Sejam as linguagens $L_m \subseteq L(V_+ \times U)^*$ e $E \subseteq V^*$. Seja ainda $K = P_V^{-1}(E) \cap L_m$. Se $E \subseteq P_V(L_m)$ então tem-se que $Sup\mathbb{C}_V(E) = P_V[Sup\mathbb{C}_{V\ U}(K)]$*

Desta forma, a máxima linguagem contida em E que é v-controlável em relação a L, é igual à projeção em V^* da máxima linguagem contida em K que é vu-controlável em relação a L. Note que se E consiste na especificação sobre o comportamento lógico do sistema sob supervisão feita em termos seqüencias de eventos, então K consiste na projeção de E em $(V^+ \times U)^*$ e é obtida habilitando-se todas as condições possíveis de ocorrer na planta para cada cadeia de eventos em E.

Retomando o problema de controle supervisório equivalente (PCSE) para sistema híbrido , a teoria introduzida anteriormente indica que a solução de PCSE requer a execução dos seguintes passos:

1. Dada a especificação $E \subseteq V^*$, obter a especificação equivalente $K \subseteq Lm \subseteq (V^+ \times U)^*$
2. Verificar se K é vu-controlável em relação a L. Se for, pular para o passo seguinte. Caso contrário, deve-se obter a máxima linguagem vu-controlável contida em K, denotada por $\text{Sup}\mathbb{C}_{vu}(K)$
3. O supervisor F é implementado através de um autômato tal que $L_m(F/H) = \text{Sup}\mathcal{C}_{vu}(K)$.

A seguir, ilustra-se esta metodologia através de um exemplo:

4.4 Exemplo

Seja um sistema híbrido $\mathcal{H}$ cujo comportamento lógico $L_m = \mathcal{L}_m(\hat{\mathcal{H}})$ é reconhecida pelo autômato ilustrado na fig.4.6.

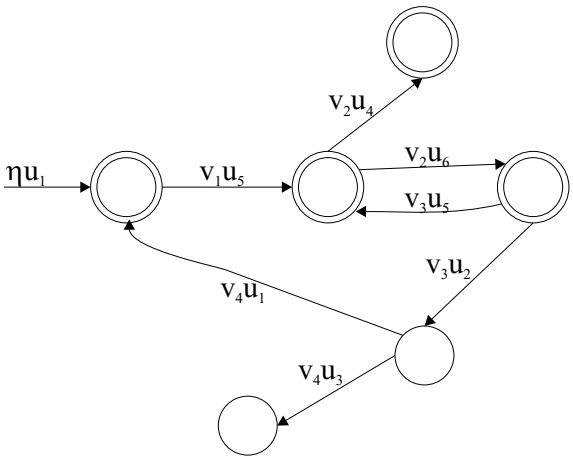


Figura 4.6: Autômato que reconhece a linguagem $L_m = \mathcal{L}_m(\hat{\mathcal{H}})$

Dada a especificação $E \subseteq P_V(L_m)$ para o comportamento desejado da planta sob supervisão como mostra a fig.4.7. Esta especificação significa que v_1 e v_2 podem ocorrer livremente, mas se ocorrer v_3, obrigatoriamente tem que ocorrer v_2.

O primeiro passo consiste em obter uma especificação equivalente que esteja contida na linguagem da planta, ou seja, $K \subseteq L_m \subseteq (V^+ \times U)^*$. Tal especificação é

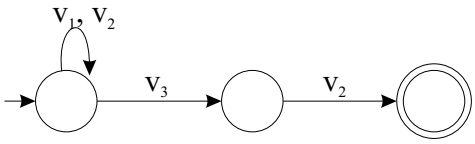


Figura 4.7: Especificação $E \subseteq P_V(L_m)$

obtida fazendo-se $K = P_V^{-1}(E) \cap L_m$ e o autômato que a reconhece é mostrado na fig.4.8.

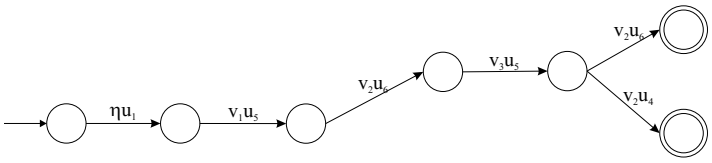


Figura 4.8: Especificação $K \subseteq L_m$

Verifica-se, entretanto, que a linguagem K não é vu-controlável em relação a L. Note, por exemplo, após a cadeia $\omega' = \eta u_1 \circ v_1 u_5 \circ v_2 u_6 \circ v_3 u_5 \in \overline{K}$, a especificação não prevê a ocorrência da transição $v_2 u_6$, sendo que as seqüencias após $\omega \circ v_2 u_6$: $v_3 u_2 \circ v_4 u_1$ ou $v_3 u_2 \circ v_4 u_3$ não seja vu-controlável em relação a K. Obtém-se então a máxima linguagem vu-controlável contida em K, a qual é reconhecida pelo autômato mostrado na fig.4.9.

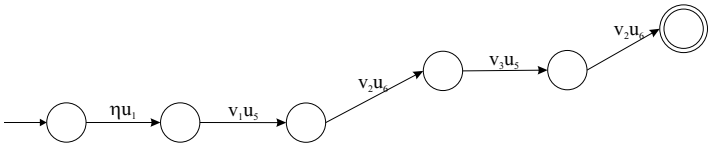


Figura 4.9: Máxima linguagem vu-controlável $Sup\mathbb{C}_{V\ U}(\mathrm{K})$

Vale lembrar que $L_m(F/H) = \mathrm{Sup}\mathbb{C}_{V\ U}(K)$. Assim, o supervisor F pode ser implementado pelo autômato da fig.4.9.

Pode-se enunciar agora o seguinte teorema (Cury e Niinomi 1998):

Teorema 4.1 *O Problema de controle Supervisório Equivalente para sistema híbrido (PCSE) possui solução se e somente se* $Sup\mathbb{C}_V(E) \supseteq A$.

Sendo assim, um supervisor F para H tal que $P_V[L_m(F/H)] = \mathrm{Sup}\mathbb{C}_V(E)$ é uma solução ótima para o PCSE no sentido de ser o menos restritivo possível.

Corolário 4.1 *O problema de Síntese de Supervisores para Sistemas Híbridos (SSSH) possui solução se e somente se o PCSE possui solução.*

4.5 Discussão

Neste capítulo, tratou-se do controle supervisório de uma classe de sistema híbrido que possui dinâmicas contínuas e discretas interagindo entre si. Neste capítulo foram introduzidos exemplos bastante acadêmicos, de forma a facilitar a introdução dos conceitos.

No próximo capítulo será iniciado ao objetivo deste trabalho, que é a implementação de um supervisor discreto numa planta piloto com características híbridas. Será feita uma breve descrição da planta piloto e de um problema proposto nesta planta que será resolvido baseado na teria descrita neste capítulo e nos capítulos anteriores.

Capítulo 5

Planta Piloto: Aspectos descritivos e modelagem matemática

Este capítulo trata da planta piloto citada no cap. 1 e de suas principais características. Ainda, este capítulo trata da formulação de um problema proposto na planta piloto. Os resultados de resolução deste problema se encontram no capítulo 6. Por fim, inicia a resolução do problema proposto com a modelagem matemática da planta piloto.

5.1 Descrição da Planta Piloto

A aquisição da planta piloto através do projeto CTPETRO tem por objetivo demonstrar de maneira didática a operação das diversas malhas de controle utilizando os mesmos equipamentos e ferramentas de configuração desenvolvidos para aplicação em controle industrial. Esta planta tem como principal característica uma rede fieldbus. Esta rede possui processamento descentralizado, onde cada dispositivo de campo possui seu próprio processamento. As principais variáveis da planta possuem dispositivos inteligentes ligados por um barramento Fieldbus.

5.1.1 Componentes da planta

A seguir uma breve descrição dos equipamentos da planta piloto.

- Moto-bombas centrífugas. São acionadas por um motor de indução e alimentadas por uma fase de 220 V e freqüência de 60 Hz. São as responsáveis pela movimentação do fluido no equipamento. A planta possui duas moto-bombas, uma para cada tanque do processo.

- LC700 - Controlador Lógico Programável. Este CLP integra a rede fieldbus da planta com a lógica discreta do CLP.

- Chave de Nível: é responsável por detectar nível baixo no tanque de água quente. O líquido, atingindo o eletrodo terra e o eletrodo de atuação, fecha o circuito pela sua própria condutividade, acionando um circuito elétrico que por sua vez comuta o relê de saída. Este sinal é enviado para o painel de controle fazendo habilitar a corrente que está indo para o conversor estático e por conseguinte que as resistências sejam ligadas.

- Termostato: está localizado no tanque de água quente e tem a função de enviar um contato para inibir o conversor estático quando a temperatura atingir um limite alto.

- Tanque reservatório: este tanque tem a função apenas de reservatório de água. As moto-bombas centrífugas alimentam os demais tanques através da água do tanque reservatório.

- Tanque de aquecimento de água: neste tanque é que se localizam os resistores, sendo possível esquentar a água. Este tanque é conectado no tanque de mistura por um cano localizado no nível máximo de água no tanque.

- Tanque de mistura: este tanque é onde obtemos os resultados finais de um processo. Este tanque é alimentado por água fria vindo de uma segunda moto-bomba e alimentado por água quente através do tanque de aquecimento.

A planta piloto possui seis dispositivos fieldbus distintos, com quantidade variada.

- LD302 - É um medidor de pressão diferencial, onde é possível converter o sinal de pressão em sinal de vazão e sinal de nível. São três LD302: um que representa o medidor de nível no tanque de aquecimento e os outro dois representam medidores de vazão de entrada dos tanques de aquecimento e mistura.

- FY302 - Posicionador de válvula. É alimentado por uma fonte externa de ar comprimido por razões de segurança intrínseca. Possui um dispositivo de retro-alimentação de posição que se utiliza do efeito Hall. Existem dois dispositivos desse tipo na planta.

- FI302 - Controlador da intensidade de corrente elétrica: Este dispositivo é responsável pelo controle de corrente que o conversor estático envia para os resistores.

- TT302 - Transmissor Indicador de Temperatura: A medição de temperatura se dá através de sensores localizados nos tanques de processo. A planta possui dois dispositivos desse tipo, um localizado no tanque de aquecimento e outro no tanque de mistura.
- DFI302 - bridge. Faz a comunicação da planta com os níveis superiores da rede fieldbus. No cap.7 este dispositivo é descrito com mais detalhes.
- FB700: Permite a integração entre o controle da planta (contínuo) e a lógica ladder do CLP (discreta).

A figura 5.1 ilustra uma representação esquemática da planta piloto com todos os seus dispositivos e equipamentos.

Figura 5.1: Foto ilustrando a planta piloto

5.1.2 Diagrama de Processo

O líquido que constitui o processo da planta é a água. O diagrama da planta é ilustrado na fig.5.2 com sua respectiva legenda na tabela 5.1. As linhas com setas indicativas representam o fluxo de água no processo. As linhas contínuas representam o fluxo principal e as linhas tracejadas são fluxos alternativos.

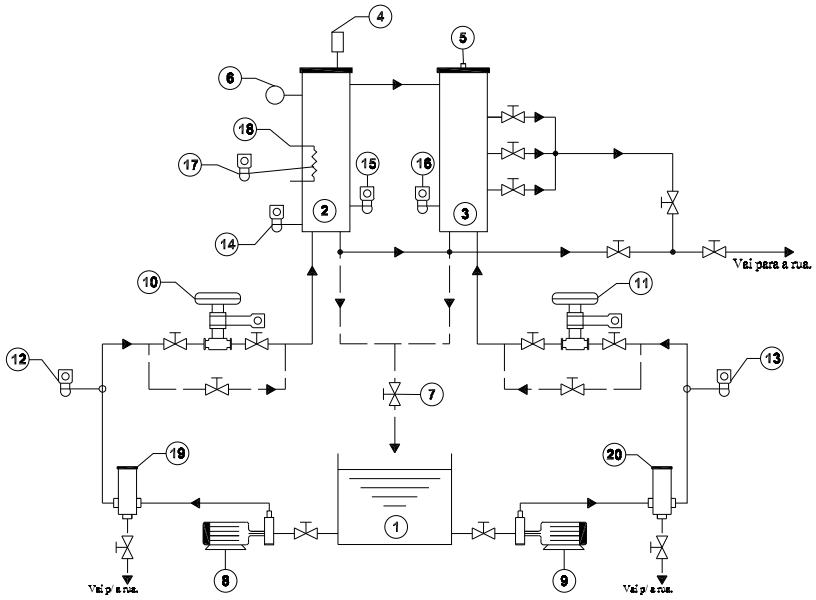


Figura 5.2: Diagrama de processo da planta piloto

ITEM	TAG	DESCRIÇÃO
1	TQ-01	Reservatório de água
2	TQ-02	Tanque de aquecimento de água
3	TQ-03	Tanque de mistura de água
4	TM-01	Termostato
5	-	Suspiro
6	CH-01	Chave de Nível
7	-	Válvulas manuais tipo esfera
8	BD-01	Bomba d'água
9	BD-02	Bomba d'água
10	FY-01	Posicionador de válvula
11	FY-02	Posicionador de válvula
12	LD-01	Transmissor de vazão
13	LD-02	Transmissor de vazão
14	LD-03	Transmissor de nível
15	TT-01	Transmissor de temperatura
16	TT-02	Transmissor de temperatura
17	FI-01	Transmissor de corrente
18	-	Resistências elétricas
19	FL-01	Filtro de água
20	FL-02	Filtro de água

Tabela 5.1: Legenda do diagrama da fig.5.2.

Seguindo o fluxo principal, a água que alimenta o tanque TQ-02 sai do TQ-01 através das bombas BD-01, passa nos filtros, passa no posicionador de válvula FY-01 e chega no TQ-01. Esta água pode ser esquentada pelas resistências elétricas, que por sua vez podem ser controladas pelo transmissor FI-01, quando o nível for alto. É possível encher o TQ-03 com água quente vindo do tanque TQ-02. É possível também misturar água quente com água fria no TQ-03 ligando a BD-02 com o TQ-01 cheio e com água quente. O TQ-02 possui um suspiro caso ambos os tanques estejam cheios. A água que sai dos tanques TQ-02 e TQ-03 podem voltar para o TQ-01 ou ir para a rua. No fluxo principal a água vai para rua evitando misturar água quente do TQ-02 com a água fria do TQ-01. No TQ-03 há mais três saídas de água para rua que podem ser habilitadas manualmente.

5.1.3 Softwares Utilizados

Tanto para a configuração dos equipamentos fieldbus quanto para o acompanhamento das dinâmicas das variáveis da rede fieldbus na planta, é necessária a utilização de softwares de configuração e monitoramento. Nesta seção, serão mostrados os principais programas utilizados na planta.

Syscon

O Syscon é responsável pela configuração dos dispositivos fieldbus da planta piloto. Podemos escolher os dispositivos que serão utilizados, criar estratégias de controle, configurar os parâmetros de cada dispositivo e depois descarregar os dados do PC para cada dispositivo na planta piloto. A fig.5.3 ilustra uma janela de trabalho do syscon.

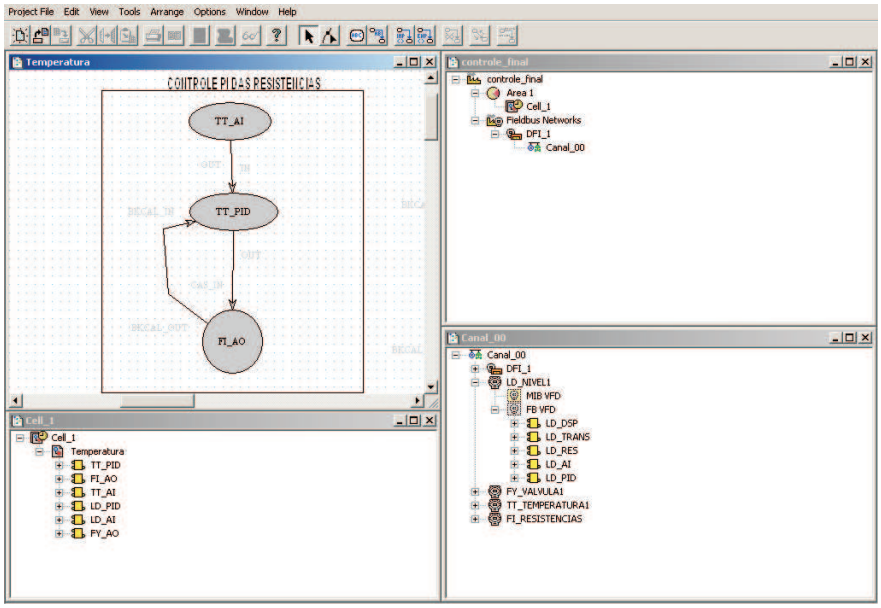


Figura 5.3: Ilustração do software Syscon

Conf700

Este software é o responsável pela configuração do CLP LC700. Através dele pode-se controlar toda a parte discreta da planta. Como exemplo de sinal discreto temos o ligamento/desligamento da bomba, o sinal da chave de nível e o sinal do termostato. A fig.5.4 ilustra uma janela de trabalho do Conf700.

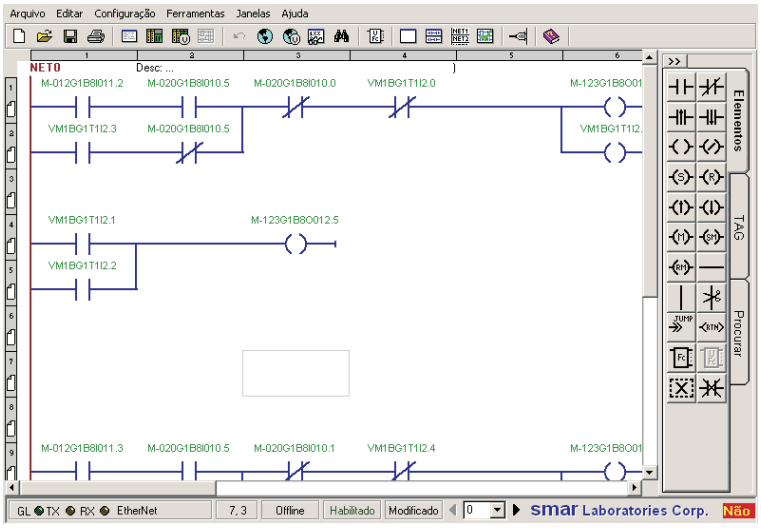


Figura 5.4: Ilustração do software Conf700

Process View

Este é o software supervisório do sistema, construído para ser a interface homem-máquina. Sua programação consiste de se fazer uma boa representação visual do sistema para o seu monitoramento e, em alguns casos, controle. O software utiliza a tecnologia OPC[1] para ter acesso às variáveis. Através dele, podemos ter acesso aos mesmos parâmetros configurados no software Syscon e no CONF700, através de servidores OPC que cada software possui.

Soft OPC Client

É um cliente OPC que tem comunicação com o MATLAB. Com isso, podemos monitorar e desenvolver algoritmos de controle com as variáveis da planta através do arquivos de extensão m do MATLAB. Mais detalhes sobre este tópico será tratado no capítulo 7.

[1]Open Process Control

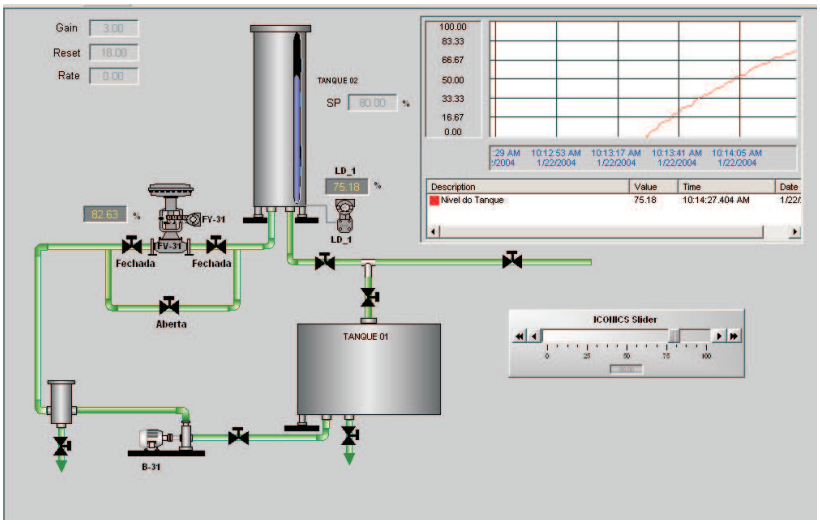


Figura 5.5: Ilustração do software Process View

5.1.4 Características híbridas da planta piloto

A planta piloto em questão é utilizada neste trabalho devido às suas características híbridas. As principais características híbridas da planta são:

- Chave de nível: é responsável por um chaveamento da dinâmica de temperatura quando o nível atinge 90% do seu valor. É também uma limitação que a planta possui.
- Tanque cheio e vazio: este dois casos provocam uma mudança na dinâmica do nível de água no tanque. Quando o tanque enche, o nível de água torna-se constante no seu valor máximo. O mesmo acontece com o tanque vazio: nível constante e igual a zero.
- Modelagem híbrida: forçam-se certos chaveamentos para que determinadas condições sejam satisfeitas. Exemplo: subdividir um sistema não-linear em equações lineares com vários pontos de operação.

5.2 Problema proposto

O problema será focado somente no tanque de aquecimento da planta. Uma justificativa é que se fosse os dois tanques, o nº de variáveis de estado iria dobrar. Com isso o problema torna-se intratável pelo fato de que a ferramenta *CheckMate* está limitada ao uso de no máximo cinco variáveis de estado contínuas. Portanto, o tanque de aquecimento de água será o sistema híbrido estudado ao longo deste documento.

Fazer com que a água que escoa para o tanque de mistura através da vazão F_s tenha uma temperatura constante e maior que a temperatura ambiente.

Tabela 5.2: Problema proposto

Deseja-se resolver o seguinte problema no tanque de aquecimento:

Onde F_s é a vazão de água do cano que liga o tanque de aquecimento ao tanque de mistura (o sentido da vazão é do tanque de aquecimento para o tanque de mistura). Uma maneira de resolver o problema proposto anteriormente é executar os seguintes passos:

- Ligar a bomba 1 e deixar que o nível do tanque de aquecimento atinja a chave do nível, situada a 90% do nível.
- Quando o nível atingir 90%, as resistências serão ligadas. Inicia-se então um controle PI do nível em um valor entre 90% e 100% do máximo. Também é feito um controle PI de temperatura para o valor desejado. A justificativa para este controle é que a constante de tempo da temperatura é muito grande comparada com a constante de tempo do nível e também porque a água só começa a esquentar quando a chave de nível atua habilitando o conversor estático a fornecer potência para as resistências.
- No instante em que a temperatura atingir o valor desejado, faz-se um controle de nível de tal forma que o tanque encha e permaneça o controle de temperatura.
- Quando o tanque estiver cheio, verificamos se o fluxo de água que escoa através do cano que interliga os dois tanques estará com agua na temperatura desejada ou não.

Quando for utilizado o termo sistema híbrido, ou planta híbrida, tais termos se referem ao tanque de aquecimento da planta.

Todo o trabalho escrito a partir deste ponto está baseado nesta seção. O tópico seguinte dá início à modelagem matemática do tanque de aquecimento com o intuito de resolver este problema.

5.3 Modelagem matemática tanque de aquecimento

A seguir será tratada a modelagem matemática do tanque de aquecimento. Consideraremos as equações de forma genérica. No próximo capítulo, quando será feita a modelagem do autômato híbrido, serão calculados os valores numéricos para cada locação. Para o cálculo das equações, considera-se que água é um líquido incompressível[2] e que as perdas de calor no tanque são desprezíveis.

A figura 5.6 ilustra o desenho do tanque de aquecimento separado da planta.

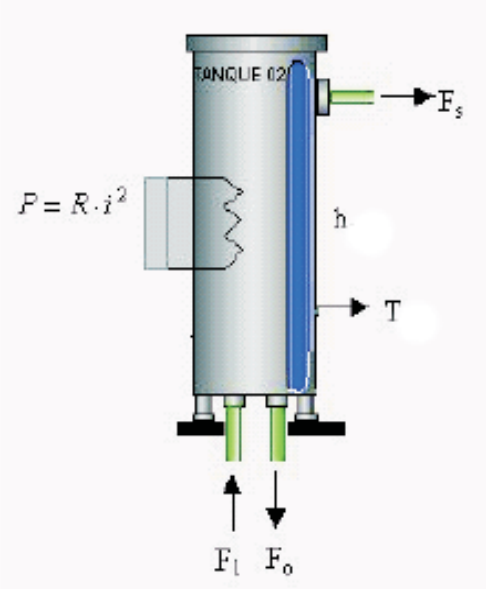


Figura 5.6: Tanque de aquecimento de água

Este tanque possui variáveis que estão conectadas aos dispositivos inteligentes. As variáveis são:

- $F_1 \rightarrow$ vazão de entrada vinda da bomba 1 em L/s.
- $h \rightarrow$ nível de água no tanque em cm.
- $T_1 \rightarrow$ temperatura da água do tanque de aquecimento em ^{o}C.
- $Q \rightarrow$ Potências dos resistores em Watts.

[2]massa específica da água não varia com o tempo

Há também variáveis que não são monitoradas pelos dispositivos diretamente. São elas:

- $F_o \rightarrow$ vazão de saída inferior do tanque de aquecimento. Denominada vazão "ladrão" do tanque. Esta vazão escoa água direto para a rua.
- $F_s \rightarrow$ vazão superior que só existe quando o nível do tanque for máximo. Ela escoa água do tanque de aquecimento para o tanque de mistura.

Por fim temos os parâmetros :

- $A \rightarrow$ área da seção transversal to tanque em cm^2.
- $A_o \rightarrow$ área da seção transversal do cano da vazão ladrão em cm^2.

Na figura 5.6, iremos aplicar o princípio de conservação de duas grandezas fundamentais: massa e energia.

Balanço de massa: Em um sistema qualquer, o balanço de massa é dado por:

$$massa_{total} = massa_{in} - massa_{out}$$

Em termos de equações para o tanque de aquecimento temos:

$$\frac{d(\rho A h)}{dt} = \rho(F_1 - F_o) \tag{5.1}$$

Assumindo a água como líquido incompressível a eq. 5.1 torna-se:

$$A\frac{dh}{dt} = F_1 - F_o \tag{5.2}$$

Pela equação de Bernoulli (Halliday e Walker 1997) obtemos a vazão ladrão F_o:

$$F_o = 2A_o k_o \sqrt{gh} \tag{5.3}$$

onde k_o é uma constante adimensional que varia entre 0,6 e 1.

Substituindo a eq. 5.3 na eq. 5.2 obtemos:

$$A\frac{dh}{dt} = F_1 - 2A_o k_o \sqrt{gh} \tag{5.4}$$

Quando o tanque está cheio, a eq. 5.4 muda para:

$$A\frac{dh}{dt} = F_1 - 2A_o k_o \sqrt{gh_{máx}} - F_s \tag{5.5}$$

onde $h_{máx}$ é o valor do nível máximo.

Balanço de energia

O balanço de energia é dado por:

$$energia_{interna} = energia_{in} - energia_{out} + energia_{fonte}$$

Em termos de equações para o tanque de aquecimento temos (Stephanopoulos 1984):

$$\frac{\mathrm{d}[\rho\ Ahc(T_1 - T_0)]}{dt} = -\rho A F_1 c(T_1 - T_0) + \rho A F_0 c(T_1 - T0) + Q \tag{5.6}$$

onde Q é a quantidade de calor fornecida pelas resistências por unidade de tempo, c é o calor específico da água e T_0 é a temperatura inicial do tanque. Reagrupando os termos da eq. 5.6 obtemos:

$$A\frac{d(hT_1)}{dt} = F_1 T_0 - F_0 T_1 + \frac{Q}{\rho c} \tag{5.7}$$

A equação 5.7 assumirá a forma de diferença de temperatura. Chamaremos de T a diferença entre a temperatura $T_1 - T_0$ onde $T_1 \geq T_0$. Desenvolvendo a derivada do produto e substituindo a eq. 5.2 na eq. 5.7 obtemos:

$$Ah\frac{dT}{dt} = -F_1 T + \frac{Q}{\rho c} \tag{5.8}$$

A eq. 5.8 terá um nível constante e igual a $h_{máx}$ caso o tanque de aquecimento esteja cheio.

Resumindo os passos da modelagem, temos que:

$$A\frac{dh}{dt} = F_1 - 2A_o k_o \sqrt{gh} \tag{5.9}$$

$$Ah\frac{dT}{dt} = -F_1 T + \frac{Q}{\rho c} \tag{5.10}$$

As variáveis nas equações 5.9 e 5.10 são classificadas como:

Variáveis de estado: h, T

Saídas: h, T

Entradas: entradas classificadas como:

Pertubações: T_0, F_0[3]

Variáveis manipuladas: Q, F_1

Apesar das equações e 5.9 e 5.10 serem não-lineares, tais equações possuem comportamento similar a uma função de transferência de primeira ordem. A fig.5.7 ilustra o comportamento da temperatura em função do tempo da eq. 5.10.

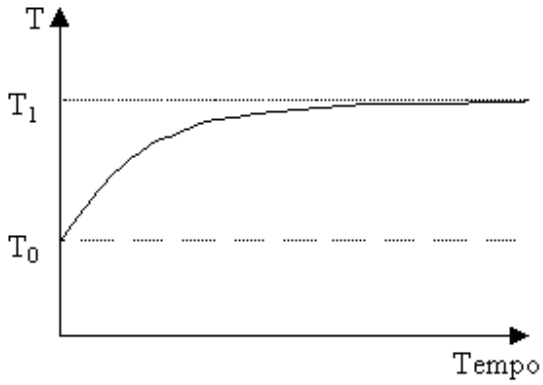


Figura 5.7: Comportamento da temperatura em função do tempo

Pode ser verificado que a temperatura possui uma faixa de variação entre T_1 e T_o.

Antes de se iniciar o próximo tópico é necessário acrescentar uma observação importante a respeito do problema proposto da seção 5.2. Para que o problema seja resolvido de forma otimizada, foram feitas as seguintes considerações:

[3] T_0 é a temperatura ambiente e como tal, pode ser distinta em instantes distintos. Já F_0 tem haver mais com certas ocorrências indesejáveis, como o cano entupir ou quebrar, alguma pessoa mexer na válvula do cano de F_0 em um dado instante, fazendo com o que esta variável seja uma pertubação.

1. Coloca a válvula do cano da vazão F_o semi-aberta. Isto se justifica na eq. 5.7 pois quanto maior for a vazão F_o, maior será a perda de energia interna pelo tanque, fazendo com o que a faixa de variação de temperatura seja menor.

2. O controle de nível com tanque cheio descrito na seção 5.2 se justifica para constatar que a vazão F_1 não precisa ser a vazão nominal da bomba para encher o tanque e contribui para que a temperatura tenha uma maior faixa de variação (vide eq.5.7) .

5.3.1 Validação e cálculo dos parâmetros

Na verdade, há dois parâmetros a serem levantados na planta.

Área do tanque

Mediu-se o diâmetro interno do tanque com uma fita métrica:

$d = 20,7cm$

Com isso:

$A = \pi r^2 = 335,885cm^2$

Área do cano da vazão F_o: calcularemos na verdade o produto $A_o k_o$.

Para isso, foi feito um teste na planta. Colocou-se a válvula do cano em questão semi-aberta em quatro pontos diferentes e mediu-se a vazão e o nível em regime permanente.

Para cada valor, aplicou-se a eq. 5.9, isolando os termos $A_o k_o$ quando a derivada do nível for nula, obtemos:

$$A_o k_o = \frac{F_1}{2\sqrt{gh}} \tag{5.11}$$

Com os valores de cada ponto calculou-se uma média aritmética deste valores.

$$A_o k_o = 0,7322e^{-4} \tag{5.12}$$

As grandes variações obtidas na tab.5.3 se devem principalmente nos dois primeiros

Abertura da Válvula *(%)*	***Vazão*** *(L/s)*	***Nível*** *(%)*	$A_o k_o$ (10^{-4})
39	0,1933	36,1	0,8808
42	0,2083	86,15	0,7911
46	0,2267	98,0	0,6132
50	0,245	100,0	0,6477

Tabela 5.3: Tabela para o cálculo de $A_0 k_0$

valores, onde as medidas são menos precisas em função do nível oscilar mesmo em regime. Estas oscilações ocorrem porque quando o tanque está com pouca água, a massa de água que escoa para dentro do tanque é significativa comparada com a massa de água dentro tanque, logo o deslocamento de água que ocorre no tanque faz com o que o nível oscile, dificultando a precisão das medidas. Em contrapartida, quando o nível está alto, a massa de água que entra é insignificante comparada com a massa de água dentro do tanque. Logo, as oscilações no nível são desprezíveis, acarretando em uma melhor precisão das medidas.

Os demais parâmetros são:

$$\rho = 1000 Kg/m^3$$
$$c = 4186 J/Kg^oC$$
$$h_{máx} = 68,0 cm$$
$$P_{máx} = 4400 W$$
$$F_{1máx} = 0,5 L/s$$
$$g = 9,81 m/s^2$$

As equações 5.9 e 5.10 são não lineares e será preciso linearizá-las em certos pontos de operação. A linearização tem por objetivo facilitar o projeto de controladores PI e proporcionais e a representação das equações em variáveis de estado. Na próxima seção iremos tratar desta linearização.

5.3.2 Linearização das equações

Balanço de massa

Para efeito de recapitulação, a equação de balanço de massa é dada por:

$$A\frac{dh}{dt} = F_1 - 2A_o k_o \sqrt{gh} \tag{5.13}$$

Temos então que linearizar o termo não linear definida por $f(h) = 2A_o k_o \sqrt{gh}$. Para isso, criaremos uma variável incremento $\bar{h}$ dada por:

$$\bar{h} = h - \tilde{h} \tag{5.14}$$

onde $\tilde{h}$ é o ponto de operação. Aplicando série de Taylor na função $f(h)$ obtemos (Stephanopoulos 1984):

$$f(\bar{h}) = A_o k_o \sqrt{\frac{g}{\tilde{h}}}\bar{h} \tag{5.15}$$

Então a eq. 5.13 torna-se:

$$A\frac{d\bar{h}}{dt} = \bar{F}_1 - 2A_o k_o \sqrt{\frac{g}{\tilde{h}}}\bar{h} \tag{5.16}$$

onde $\bar{F}_1 = F_1 - \tilde{F}_1$. Em alguns casos, quando a vazão $\bar{F}_1$ for nula, a série de Taylor terá apenas um elemento, a constante. Aplicando $h = \tilde{h}$ na função $f(h)$ e substituindo na eq. 5.13 obtemos:

$$A\frac{d\bar{h}}{dt} = -2A_o k_o \sqrt{g\tilde{h}} \tag{5.17}$$

Balanço de energia

Para a o balanço de energia, modelou-se a seguinte equação:

$$Ah\frac{dT}{dt} = -F_1 T + \frac{Q}{\rho c} \tag{5.18}$$

Considerou-se a eq. 5.18 para variações de T e F_1 somente. Com a limitação da chave de nível, o conversor estático só injeta corrente nos resistores com o nível acima de 90%. Com isso, admitiu-se a variação do nível no intervalo entre 90% e 100% é desprezível. Temos então que a eq. 5.18 possui um termo não linear:

$$f(T,\, F_1) = F_1 T \tag{5.19}$$

Usando a série de Taylor para duas ou mais variáveis para a eq 5.19 obtemos:

$$f(\bar{T},\, \bar{F_1}) = \tilde{T}\bar{F_1} + \tilde{F_1}\bar{T} \tag{5.20}$$

onde:

$$\bar{T} = T - \tilde{T} \tag{5.21}$$

e $\tilde{F_1}$ e $\tilde{T}$ são os pontos de operação para as variáveis F_1 e T respectivamente.

Substituindo a eq. 5.20 na eq. 5.18 e agrupando os termos obtemos:

$$\tilde{h}A\frac{d\bar{T}}{dt} = -\tilde{T}\bar{F_1} - \tilde{F_1}\bar{T} + \frac{Q}{\rho c} \tag{5.22}$$

Portanto, temos que a equação 5.16 possui uma entrada ($\bar{F_1}$) e uma variável apenas ($\bar{h}$). Já a equação 5.22 possui duas entradas (Q e $\bar{F_1}$). Na próxima seção calcularemos as funções de transferência para cada caso.

5.3.3 Cálculo das Funções de Transferência

Neste tópico iremos aplicar a transformada de Laplace nas equações de balanço de massa e balanço de energia linearizadas para efetuar o cálculo das funções de transferência do processo. Considera-se que as condições iniciais são nulas.

Balanço de Massa

Para facilitar o cálculo da função de transferência do balanço de massa, define-se uma constante c_o como:

$$c_o = A_o k_o \sqrt{\frac{g}{\tilde{h}}} \tag{5.23}$$

Para calcular a função de transferência de balanço de massa, aplica-se a transformada de Laplace na eq.5.16 já com a eq. 5.23 incluída:

$$As\bar{H}(s) = -c_o\bar{H}(s) + \bar{F}_1(s) \tag{5.24}$$

Após algumas manipulações algébricas, obtemos:

$$\frac{\bar{H}(s)}{\bar{F}_1(s)} = \frac{c_0^{-1}}{\frac{A}{c_0}s + 1} \tag{5.25}$$

Balanço de Energia

Seja a equação do balanço de energia linearizada:

$$\tilde{h}A\frac{d\bar{T}}{dt} = -\tilde{T}\bar{F}_1 - \tilde{F}_1\bar{T} + \frac{Q}{\rho c} \tag{5.26}$$

Aplicando a transformada de Laplace na eq. 5.26 teremos duas funções de transferência, uma para a entrada Q e outra para a entrada $\bar{F}_1$.

1. Entrada $\bar{F}_1$

   Neste caso, a função de transferência é dada por:

$$\frac{\bar{T}(s)}{F_1(s)} = -\frac{\tilde{T}(\tilde{F}_1)^{-1}}{\frac{A\tilde{h}}{\tilde{F}_1}s + 1} \tag{5.27}$$

2. Para entrada Q

   A função de transferência é dada por:

$$\frac{\bar{T}(s)}{Q(s)} = \frac{(\rho c\tilde{F}_1)^{-1}}{\frac{A\tilde{h}}{\tilde{F}_1}s + 1} \tag{5.28}$$

Apesar da temperatura possuir duas funções de transferência, utilizaremos somente a eq. 5.28. Verifica-se que todas as funções de transferência são de 1^a ordem e que este é um processo multivariável, onde a entrada F_1 acopla as variáveis nível e temperatura. No capítulo seguinte trataremos da sintonia dos controladores PI para cada função de transferência. Porém, iremos considerar os controladores para a modelagem das equações no espaço de estados contínuo tratado no tópico seguinte.

5.3.4 Modelagem das equações no espaço de estados contínuo

Iremos deduzir de forma genérica a transformação das funções de transferência e controladores para o espaço de estados.

Seja a função de transferência de 1ª ordem dada por:

$$\frac{Y(s)}{U(s)} = \frac{k}{\tau s + 1} \tag{5.29}$$

onde $U(s)$ é a entrada do sistema, $Y(s)$ é a saída, k é ganho do sistema e τ é a constante de tempo.

Para transformar esta função de transferência em uma equação de variáveis de estados, aplica-se a transformada inversa de Laplace à equação 5.29 e fazendo $y = x$:

$$\dot{x} = -\frac{1}{\tau}x(t) + \frac{k}{\tau}u(t) \tag{5.30}$$

O compensador ou controlador PI é dado por:

$$\frac{U(s)}{E(s)} = \frac{k_c s + k_i k_c}{s} \tag{5.31}$$

onde k_c é o ganho proporcional e o k_i é o ganho integral.

De acordo com (Ogata 1998), colocou-se o controlador PI na forma canônica observável . Com isso, as equações de estado para eq. 5.31 são da forma:

$$\dot{x}_c = k_i k_c e(t) \tag{5.32}$$

$$u(t) = x_c(t) + k_c e(t) \tag{5.33}$$

onde a função $e(t)$ é o erro dado por:

$$e(t) = r(t) - y(t) \tag{5.34}$$

sendo que $r(t)$ é a referência. Substituindo $e(t)$ da eq. 5.34 na eq. 5.32 e $y(t)$ por $x(t)$ obtemos:

$$\dot{x}_c = -k_i k_c x(t) + k_i k_c r(t) \tag{5.35}$$

Esta é a equação de estados para o compensador PI. Para a planta, substituindo a eq. 5.33 na eq. 5.30 a agrupando os termos obtemos:

$$\dot{x} = -\frac{(1 + kk_c)}{\tau} x(t) + \frac{k}{\tau} x_c(t) + \frac{kk_c}{\tau} r(t) \tag{5.36}$$

A figura seguinte ilustra um diagrama de blocos com o sistema processo-compensador no intuito de facilitar o entendimento ao leitor.

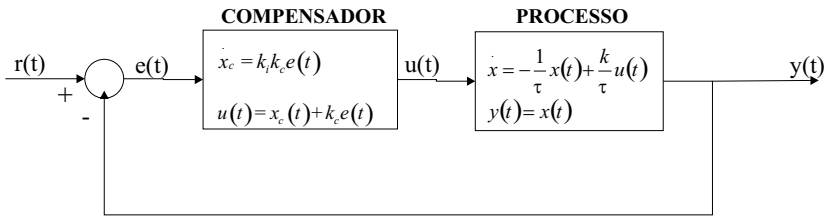


Figura 5.8: Diagrama de blocos de um sistemas processo-compensador no espaço de estados contínuo.

Colocando as equações na forma matricial:

$$\begin{bmatrix} \dot{x} \\ \dot{x_c} \end{bmatrix} \begin{bmatrix} -\frac{(1+kk_c)}{\tau} & \frac{k}{\tau} \\ -k_i k_c & 0 \end{bmatrix} \begin{bmatrix} x(t) \\ x_c(t) \end{bmatrix} + \begin{bmatrix} \frac{kk_c}{\tau} \\ k_i k_c \end{bmatrix} r(t) \tag{5.37}$$

Este modelo matricial é útil pelo fato de ser a estrutura utilizada pelo *CheckMate* de declarar as possíveis dinâmicas contínuas sob a forma de equações de estados. O capítulo seguinte trata esta questão com mais detalhes.

5.4 Discussão

Este capítulo tratou de uma breve descrição da planta piloto, onde serão feitos experimentos práticos aplicando-se a teoria de controle supervisório de sistemas híbridos. Iniciou-se a resolução do problema proposto tratando os aspectos matemáticos do tanque de aquecimento. Tal modelagem é importante não só para o modelo autômato híbrido, mas também para diversos estudos de outras áreas e abordagens. O próximo capítulo continua a resolução do problema da seção 5.2 com a modelagem do autômato híbrido de comportamento livre e o cálculo de seu respectivo autômato discreto através da ferramenta *CheckMate* , tratada no cap 3.

Capítulo 6

Resolução do problema proposto através da síntese de um supervisor discreto

O presente capítulo dá continuidade na resolução do problema proposto tratado no capítulo 5 com a modelagem de um autômato híbrido que represente o comportamento da planta livre. Em seguida trata o cálculo do autômato discreto aproximado que representa o autômato híbrido. Por fim, dada uma especificação em relação ao problema proposto, faz o projeto de um supervisor discreto de acordo com a teoria de controle supervisório de sistemas híbridos.

6.1 Modelagem do autômato híbrido

Inicia-se a modelagem destacando as variáveis nível (h) e temperatura (T). A modelagem híbrida, como cruzamento de hiperplanos e os guardas do modelo autômato híbrido do tanque de aquecimento é baseada nestas duas variáveis. As variáveis de estado utilizadas para a modelagem são 4:

- $x_1 \rightarrow$ esta variável representa o nível de água no tanque h.
- $x_2 \rightarrow$ esta variável não tem sentido físico explícito, mas surgiu com a inserção de um compensador PI para o nível.
- $x_3 \rightarrow$ esta variável representa a temperatura de água no tanque T.
- $x_4 \rightarrow$ esta variável não tem sentido físico explícito, mas surgiu com a inserção de um compensador PI para a temperatura.

A seguir a definição de cada hiperplano de h:

- $h \leq 0$: A modelagem híbrida proposta é que se a igualdade referida anteriormente for satisfeita, podem ocorrer duas possibilidades. A primeira é o tanque ficar permanentemente vazio e a segunda é o nível de água no tanque voltar a subir.
- $h \geq 0.5m$ Este hiperplano tem a função de inserir no sistema a técnica de chaveamento de controladores. Este chaveamento se justifica no intuito de atenuar o overshoot do nível de água, evitando assim que haja cruzamentos indesejáveis de hiperplanos .
- $h \geq 0.64m$: A modelagem híbrida neste caso dar-se-á pelo fato da uma restrição de funcionamento da planta piloto dada pela chave de nível. Só é possível fornecer potência aos resistores quando a condição $h >= 0.64m$ for satisfeita. Por questão de segurança, os resistores só podem estar ligados se estiverem submersos na água. Caso contrário, a falta de água acarretaria na queima dos resistores. Nesta desigualdade há várias dinâmicas possíveis de serem implementadas, que serão tratadas ainda nesta seção.
- $h \leq 0.64m$: Este hiperplano possui comportamento oposto do hiperplano anterior. Com o nível baixo, o conversor estático deixa de fornecer potência aos resistores.
- $h \geq 0.68m$: A modelagem híbrida proposta é que se a igualdade referida anteriormente for satisfeita, o tanque fica cheio e aparecerá a vazão F_s. Há também várias dinâmicas possíveis e estas serão tratadas posteriormente

Neste momento definem-se os hiperplanos para a temperatura.

- $T \geq 2$ este valor de temperatura é o valor desejado para resolver o problema proposto.
- $T \geq T_{máx}$: este hiperplano foi criado no intuito de evitar que o tanque fique muito quente.

Valores numéricos das equações:

Primeiro, reescreve-se função de transferência de 1^a ordem:

$$\frac{Y(s)}{U(s)} = \frac{k}{\tau s + 1} \tag{6.1}$$

Comparando a eq. 5.25 com a eq. 5.29 temos:

$$k = c_o^{-1} \tag{6.2}$$

e

$$\tau = \frac{A}{c_o} \tag{6.3}$$

A equação do balanço de massa será dividida em três pontos de operação, uma para cada região entre os hiperplanos do nível. A tabela 6.1 mostra os valores calculados para as funções de transferência do nível para cada ponto de operação.

Intervalos (cm)	$\tilde{h}$ (cm)	c_o	k	τ
$]0, 50]$	30,0	0,2961	3,378	113,44
$]50, 60[$	50,0	0,2293	4,36	146,48
$[60, 68[$	62,0	0,206	4,85	163,05

Tabela 6.1: Valores de k e τ para cada ponto de operação

Para o balanço de energia, comparando a eq. 5.28 com a eq. 5.29 obtém-se

$$k' = (\rho c \tilde{F}_1)^{-1} \tag{6.4}$$

e

$$\tau' = \frac{A\tilde{h}}{\tilde{F}_1} \tag{6.5}$$

Neste caso, utiliza-se dois pontos de operação para o nível e um ponto de operação para a vazão $\tilde{F}_1$. Os valores dos pontos de operação localizam-se entre a chave de nível e o nível máximo do tanque. A tabela 6.2 mostra os valores das funções de transferência da temperatura para os dois casos.

Intervalos (cm)	$\tilde{h}$ (cm)	$\tilde{F}_1$ (l/s)	k'	τ'
$[62, 68[$	62,0	0,333	1	62,54
68	68,0	0,333	1	68,60

Tabela 6.2: Valores de k' e τ' para cada ponto de operação

Nestas tabelas não se inclui os controladores. O apêndice A deste documento trata de um projeto de controlador PI feito passo a passo e os valores dos demais controladores.

Modelo Autômato Híbrido

O modelo autômato híbrido que está ilustrado na fig.6.1[1] está de acordo com a definição 3.1 e com os valores numéricos já incluídos. As equações estão no espaço de estados contínuo de acordo com a eq. 5.37.

Onde:

- r é a referência do nível igual a 64cm.
- $r_{máx}$ é a referência do nível para tanque cheio
- r_T é a referência da temperatura igual a 2^oC
- $T_{máx}$ é a diferença máxima de temperatura que o tanque pode atingir. Este valor é obtido aplicando-se o teorema do valor final na eq.5.28 com um degrau $Q = 4400\text{W}$. Logo, $T_{máx} = 3,1^oC$.

A seguir uma breve descrição de cada locação:

Locação 1: É a locação inicial do autômato híbrido. Significa que o tanque está inicialmente vazio e enchendo sob ação de um controlador proporcional. Neste momento não tem aquecimento de água.

Locação 3: sua importância se da pela modelagem de um chaveamento de controladores. O controlador proporcional dá lugar a um controlador PI. Este chaveamento tem o intuito de se evitar um sobressinal muito alto no nível de água no tanque.

Locações de 4 a 8: São locações que estão com valores de nível acima do hiperplano da chave de nível. Foi proposto aqui uma combinação de possíveis dinâmicas que podem ser configuradas na planta. As locações 4 e 5 o nível de água continua enchendo, mas na locação 4 a temperatura está em malha aberta e na locação 5 a temperatura está em malha fechada com referência igual a 2^oC. O mesmo acontece com as locações 6 e 7. Ambas estão com o nível com controlador PI com referência em $64cm$, mas na locação 6 a temperatura está em malha aberta e na locação 7 a temperatura está em malha fechada com referência igual a 2. Já a locação 8 representa uma situação em que o nível esta diminuindo. Neste caso, o nível atingirá a chave de nível fazendo cortar o aquecimento do tanque.

Locações 9 e 10: São locações que indicam que o nível de água no tanque está no máximo. A temperatura está em malha aberta na locação 9 e com um controlador PI

[1]A transição em tracejado é similar às demais transições, servindo apenas para não atrapalhar a leitura textual da figura.

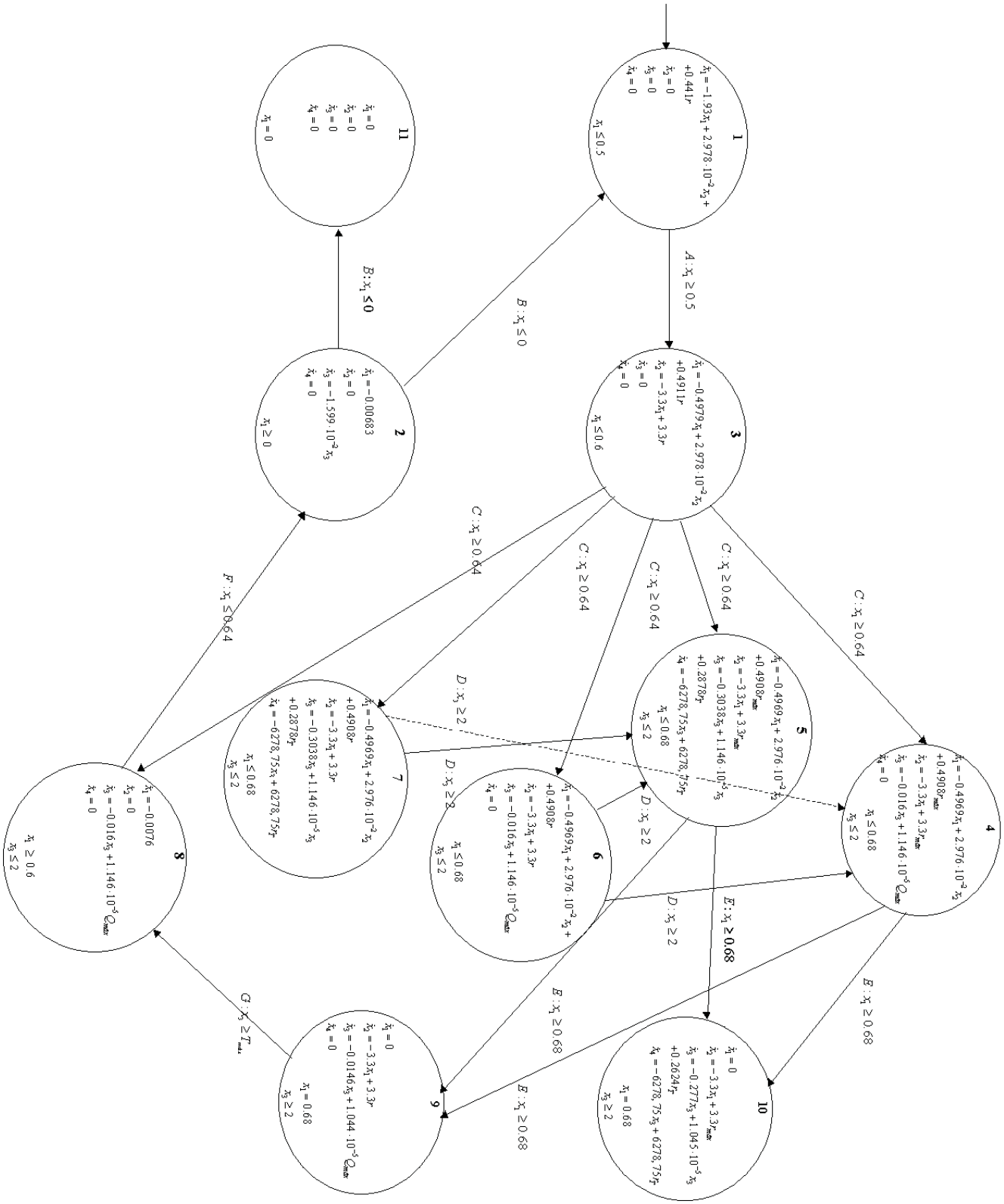


Figura 6.1: Modelo Autômato híbrido para o sistema híbrido tanque de aquecimento.

com referência igual a 2 na locação 10. Esta última é o objetivo deste trabalho. Se for possível atingir esta locação e permanecer nela, resolve-se o problema proposto.

Locação 2: Esta locação indica que a chave de nível desabilitou a temperatura e que o nível está decaindo com o tempo.

Locação 11: Significa que o tanque está vazio.

Verifica-se o autômato híbrido modelado na fig.6.1 possui transições de mesmo nome e guardas. Isto caracteriza o comportamento não determinístico deste autômato. A transição C, por exemplo, pode levar para qualquer locação de 4 a 8.

Logo, a modelagem do comportamento livre da planta é importante para que se possa formular um problema sob um aspecto geral, onde se considera no modelo todas as possibilidades de chaveamentos de dinâmicas relevantes. O modelo planta livre não é único. É possível, em um mesmo sistema híbrido, modelar uma planta livre com um modelo autômato híbrido diferente da fig.6.1. Isto depende do problema a ser resolvido, quais são as variáveis relevantes, a modelagem dos hiperplanos, escolha dos controladores, possibilidades de chaveamento, etc.

Para resolver este problema de controle, obtém-se um autômato de estados finitos que represente o comportamento lógico aproximado do sistema híbrido. Na seção seguinte é detalhada a obtenção deste modelo discreto.

6.2 Cálculo do autômato aproximação

Para o tratamento do autômato híbrido da fig.6.1, utilizaremos a ferramenta *Check-Mate* . A fig. 6.2 ilustra este modelo.

Dinâmica Contínua

O bloco SCS da fig.6.2 corresponde a um bloco de sistema contínuo chaveado modelando $\mathcal{H}_c$. Para este modelo em particular, o sinal de entrada u é um vetor (sinal multiplexado) de 4 sinais cada um assumindo o valor das variáveis x_1, x_2, x_3 e x_4 da fig.6.1.

A função de chaveamento que retorna a derivada de cada valor de u é especificada em um arquivo de extensão ".m" que associa cada valor de u uma dinâmica do autômato híbrido, onde o n^o da locação é igual a u.

Sete blocos à direita do bloco SCS correspondem a blocos de limite poliédrico (PTHB). Tais blocos equivalem aos guardas do autômato híbrido. Nota-se que o autômato híbrido da fig.6.1 possui mais que sete transições. No entanto há muitas

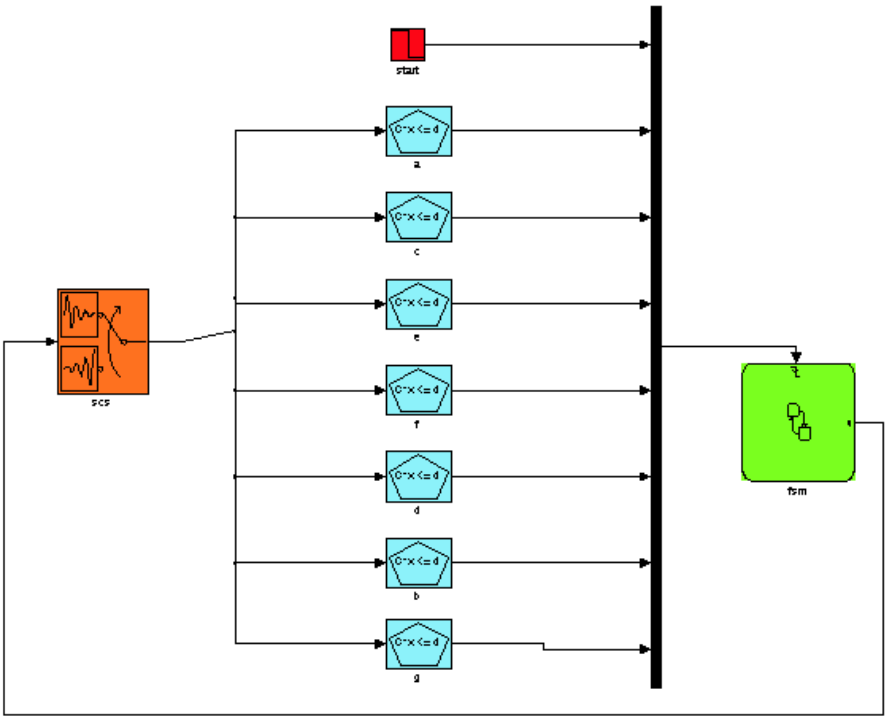


Figura 6.2: Modelo Checkmate do Sistema híbrido em malha aberta

transições iguais devido ao comportamento não-determinístico do autômato híbrido e portanto basta um PTHB para representar todas as transições iguais. Neste modelo, cada poliedro convexo limita-se apenas a definir um limite do tipo reta associado a cada transição do modelo autômato híbrido. Assim por exemplo, considerando $x = [x_1\ x_2\ x_3\ x_4]^T$, o poliedro definido pelo par (C, d), onde $C = [-1\ 0\ 0\ 0]$ e $d = [-0.5\ 0\ 0\ 0]^T$, define o limite $-x_1 \leq -0.5$ ou $x_1 \geq 0.5$. Para este poliedro, o sinal de saída só é verdadeiro para valores $x_1 \geq 0.5$.

Dinâmica Discreta

O bloco mais a direita da fig. 6.2 representa o bloco de máquina de estados finito que modela $\mathcal{H}_d$. Os *eventos de entrada* deste bloco são sinais multiplexados, ou vetores de sinais. O critério de detenção da mudança do sinal adotado para todos os eventos é do tipo *borda de descida*. Para funcionar como *borda de descida*, muda a desigualdade dos guardas. Assim, para *borda de descida*, o limite $x_1 \geq 0.5$ mudará para $x_1 \leq 0.5$.

O modelo da máquina de estado finito da fig.6.3 representa as características discretas do autômato híbrido.

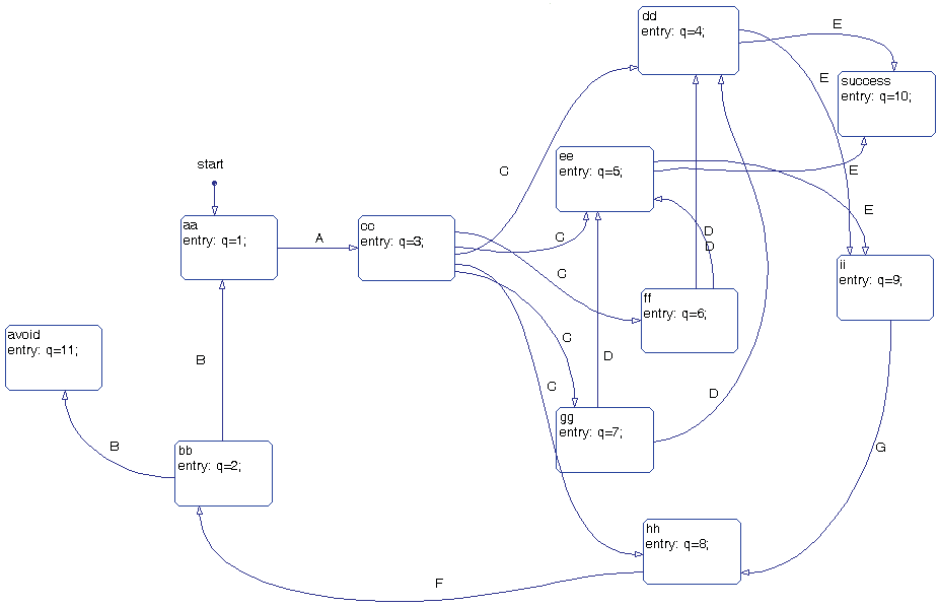


Figura 6.3: Máquina de estados finita representando a dinâmica discreta do autômato híbrido

Destacam-se as locações de nº 11 (*Avoid*) e de nº 10 (*sucesso*). A primeira significa uma locação indesejável, que é quando o tanque esvazia e permanece vazio. A segunda é a locação que resolve o problema proposto, conseguir água quente a uma temperatura constante escoando para o tanque de mistura.

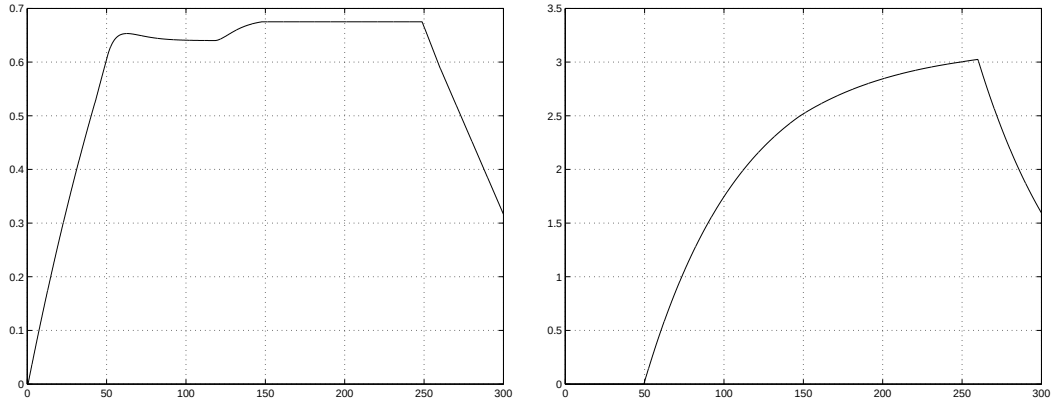
6.2.1 Simulação do Modelo Planta híbrida em malha aberta

Uma dos aspectos relevantes do *CheckMate* não só é a verificação , mas também a simulação de um sistema híbrido. A simulação engloba apenas um subconjunto de condições iniciais do sistema enquanto que a verificação considera todas as possibilidades pertencentes a um conjunto de condições iniciais. Assim, a simulação pode ser útil para conhecer a resposta do sistema sob certas condições sem a necessidade de se efetuar uma verificação, que é bem mais demorado. A simulação é feita sob o ambiente Simulink do MATLAB. A fig.6.4 ilustra um dos possíveis comportamentos que o modelo da planta híbrida em malha aberta pode realizar.

6.2.2 Verificação de propriedades do modelo autômato híbrido

A principal utilização da ferramenta *CheckMate* é a verificação de sistemas híbridos. Este tópico inicia a verificação do modelo autômato híbrido da fig.6.1. A verificação utiliza a mesma configuração de blocos da fig. 6.2.

Para a verificação, dois parâmetros são importantes no *CheckMate* : O conjunto



(a) Nível (b) temperatura

Figura 6.4: Gráficos que representam um comportamento possível do autômato híbrido

de condições iniciais e as especificações ACTL tratadas no cap. 3 para verificação de propriedades.

Para as condições iniciais temos que:

$$\begin{aligned} 0 \leq x_1 &\leq 0.3m \\ x_2 &= 0 \\ 0 \leq x_3 &\leq 0.01^oC \\ 0 \leq x_4 &\leq 0.01 \end{aligned} \tag{6.6}$$

O *CheckMate* obriga o usuário a definir um intervalo de condições iniciais para n-1 variáveis, mesmo que um intervalo de condições iniciais seja importante para apenas uma variável. Neste caso em particular, apenas o conjunto de condições iniciais de x_1 é importante analisar devido ao fato de que x_2 e x_4 não possuem sentido físico e x_3 que representa a temperatura está inativa na locação inicial e portanto sua condição inicial é sempre nula.

As especificações ACTL foram geradas em relação aos estados *avoid* e *sucesso*:

- Especificação 1 $AG(AF\ fsm == sucesso)$: Significa que todos os caminhos no futuro levam ao estado *sucesso* e vão permanecer nele.
- Especificação 2 $(AG\text{-}fsm == avoid)$: Significa que para todos os caminhos globalmente o estado *avoid* nunca será atingido.

Vale ressaltar que se a especificação 1 for satisfeita, necessariamente a especificação 2 também é satisfeita, já que no estado *sucesso* não haverá mais cruzamentos de hiperplanos. Efetuou-se a verificação no *CheckMate* através do comando *verify*. Ambas

as especificações falharam, o que significa que existe alguma condição inicial e também o comportamento não determinístico do autômato híbrido que leva ao estado *avoid* e que conseqüentemente não atinja o estado *sucesso*.

6.2.3 Obtenção do Modelo Discreto

O modelo discreto da verificação descrita anteriormente é guardado em uma variável do MATLAB/Workspace chamada de *GLOBAL AUTOMATON*. O modelo discreto é uma aproximação do modelo autômato híbrido.

Este modelo é tratado na ferramenta Grail, que é uma biblioteca de funções em C++ que permite a computação simbólica sobre expressões regulares, linguagens finitas e autômatos finitos (Gonzáles 2000). Contudo, a variável *GLOBAL AUTOMATON* não pode ser interpretada diretamente pelo Grail. Para isso, foi utilizado um algoritmo que é capaz de converter o modelo discreto calculado no *CheckMate* em uma lista de instruções que pode ser interpretada pelo Grail (Leal e Cury 2004). O resultado é um autômato C/E que descreve o comportamento da planta híbrida

A fig.6.5 mostra o resultado da verificação:

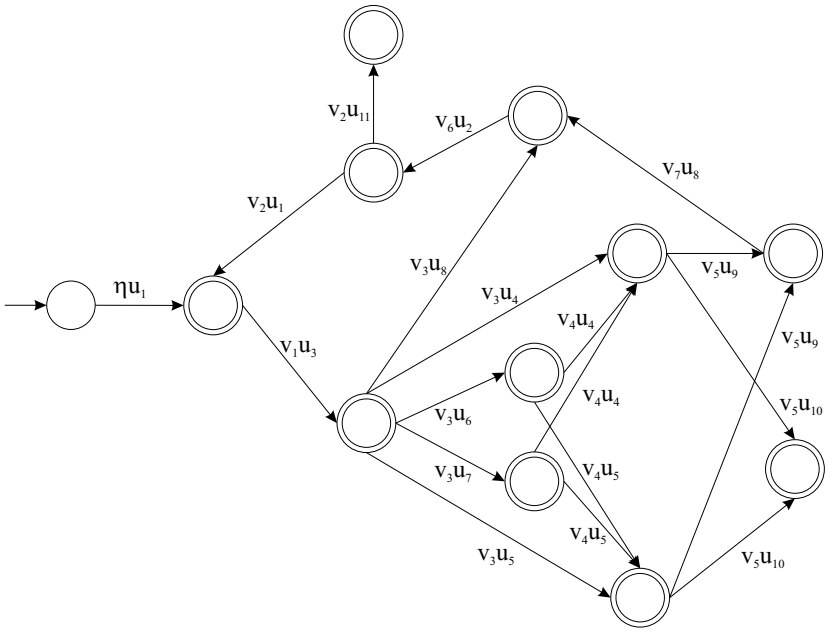


Figura 6.5: Modelo discreto correspondente ao autômato híbrido

O modelo da fig.6.5 é um autômato C/E onde v são os eventos e u são as condições. Na abordagem de sistemas híbridos, os eventos de um sistema C/E estão associados ao cruzamento de hiperplanos de um sistema híbrido. Hiperplanos diferentes e sentido de cruzamento diferentes implicam em eventos distintos. As condições são associadas as

dinâmicas de um sistema híbrido. Dinâmicas distintas são determinadas por condições distintas.

Para o modelo discreto da fig.6.5, a numeração do índices de u é a mesma que cada locação do autômato híbrido possui. As transições A, B, C, D, E, F, e G, são representados pelos eventos v_1, v_2, v_3, v_4, v_5, v_6 e v_7 respectivamente.

Analisando o modelo discreto da fig.6.5, verificamos que o evento de inicialização ocorre com a condição u_1. Repare que o estado atingido pela transição ηu_1 possui restrições de comportamento, pois há somente um evento ativo após ηu_1, que é o v_1. Para o evento v_1 há somente a condição u_3. Com a ocorrência da transição $v_1 u_3$, o estado atingido possui também apenas um evento ativo após a cadeia $\eta u_1 \circ v_1 u_3$ (v_3), contudo tal evento está associado a cinco condições: u_4, u_5, u_6, u_7 e u_8. A cadeia $\eta u_1 \circ v_1 u_3$ corresponde a inicialização do autômato híbrido na locação 1 e a ocorrência da transição A: $x_1 \geq 0,5$ levando o modelo para a locação 2. Na locação 2, há um não determinismo que faz com o que uma transição possa atingir cinco locações distintas.

Fazendo a análise completa do modelo discreto, é fácil notar que tal modelo é idêntico ao autômato híbrido (sob aspecto discreto). O que normalmente ocorre é a obtenção de uma aproximação externa e que com o refinamento desta aproximação talvez consiga atingir o modelo exato (Chutinan 1999). Neste caso em particular o modelo discreto exato do autômato híbrido foi calculado sem a necessidade de um refinamento.

O problema agora é que tal modelo não satisfaz a especificação de atingir e permanecer no estado *sucesso*. Para isso, será utilizado a teoria de controle supervisório de sistemas híbridos para resolver o problema proposto.

6.3 Projeto do supervisor discreto

Para o projeto de um supervisor, foi utilizada a ferramenta Grail. Recentes pequisas acrescentaram ao Grail novas bibliotecas de funções. Tais bibliotecas dizem respeito a teoria de controle supervisório de SED, autômato C/E e controle supervisório de sistemas C/E e outras. Foram utilizadas neste trabalho as bibliotecas de funções sobre o autômato C/E e controle supervisório de sistemas C/E. O projeto do supervisor discreto segue os mesmos passos do exemplo feito no cap.4. A planta livre é o modelo da fig.6.5

Dada a especificação $E \subseteq P_V(L_m)$ para o comportamento desejado da planta sob supervisão como mostra a fig.6.6. Esta especificação é do tipo seqüência de eventos que leva ao estado correspondente à locação 10 do autômato híbrido.

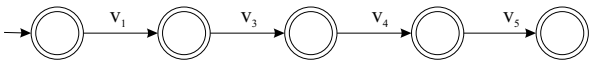


Figura 6.6: Especificação $Esp \subseteq P_V(L_m)$

O próximo passo é encontrar uma linguagem $K \subseteq (V^+ \times U)^*$. Aplica-se a projeção inversa em Esp e faz a intersecção com a linguagem marcada da planta. Com isso obtemos o autômato ilustrado na fig. 6.7.

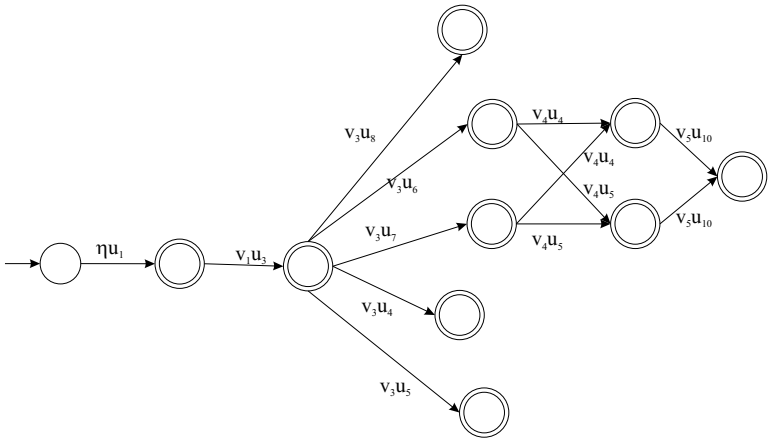


Figura 6.7: Especificação $K \subseteq L_m$

Verifica-se, entretanto, que a linguagem K não é vu-controlável em relação a L. Note, por exemplo, que após a seqüencia $\omega = \eta u_1 \circ v_1 u_2 \circ v_3 u_5 \in \overline{K}$ pode levar a seqüencias que não são vu-controláveis em relação a K. Obtém-se então a máxima linguagem vu-controlável contida em K, a qual é reconhecida pelo autômato mostrado na fig.6.8.

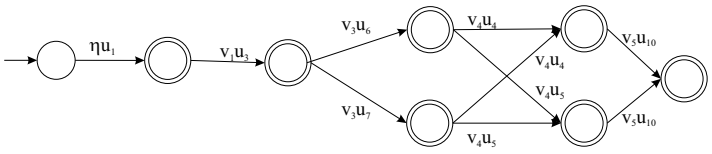


Figura 6.8: Máxima linguagem vu-controlável $Sup\mathbb{C}_{V\ U}$(K)

Com isso o supervisor para o autômato híbrido que representa o comportamento da planta híbrida foi concluído. Ressalta-se que pela fig.6.8 há quatro caminhos distintos que levam ao estado desejado. So o aspecto físico, tais caminhos combinam as variáveis nível e temperatura tendo controladores ou em malha aberta. O próximo passo é utilizar novamente o *CheckMate* para verificar as especificações da subseção 6.2.2 que falharam para o modelo discreto da planta híbrida em malha fechada.

Verificação de propriedades do modelo autômato híbrido sob ação do supervisor

Para este caso em particular, basta substituir a parte discreta da planta original pelo supervisor, pois o modelo discreto da planta é exato e não aproximado. Se o modelo discreto fosse uma aproximação externa teria que se fazer o produto síncrono do supervisor e parte discreta da planta acoplada com a parte contínua da planta (Gonzáles 2000).

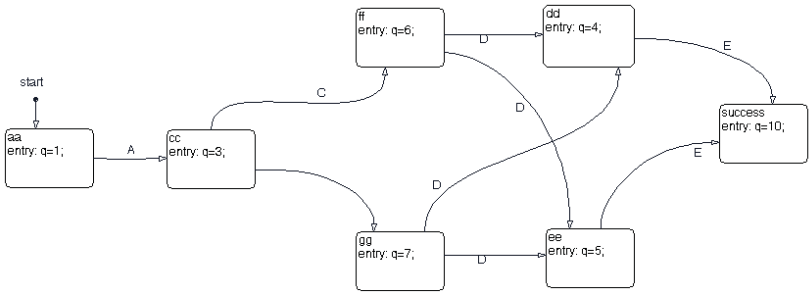
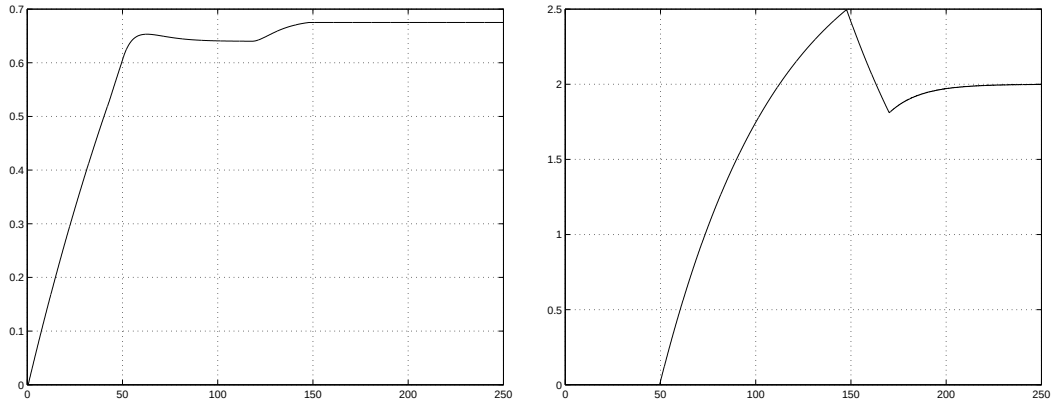


Figura 6.9: Máquina de estados finita representando a planta híbrida sob ação de um supervisor

Efetua-se agora a verificação para a 1^a especificação descrita na subseção 6.2.2. Observar-se que na fig. 6.9 o estado *avoid* não se encontra. Então não faz sentido testar a segunda especificação da subseção 6.2.2, pois o supervisor impede que tal especificação falhe.

Efetuou-se uma simulação que comprova que a especificação foi satisfeita. Pela fig. 6.10 observa-se pelo gráfico do nível que o tanque enche em torno de 150 segundos e a temperatura estabiliza em torno de 200 segundos.



(a) Nível (b) temperatura

Figura 6.10: Gráficos que representam o comportamento do autômato híbrido sob ação do supervisor

A especificação, como previsto, é satisfeita. Isso mostra que sistema híbrido modelado pelo autômato híbrido da fig.6.1 sob ação do supervisor da fig. 6.8 atende a especificação ACTL. Vale ressaltar que esta é apenas uma possibilidade das quatro

possíveis que a planta híbrida em malha fechada consegue satisfazer a especificação ACTL. Fisicamente, a possibilidade é encher o tanque água e nível com controle proporcional, chaveamento do controlador proporcional do nível para um PI com referência em 64cm. Com o nível estabilizado em 64cm, a temperatura começa a subir e está sem controle até atingir o valor de 2 graus. Com isso, o nível volta a subir e a temperatura permanece aumentando e sem controle até que o tanque enche e a temperatura passa a ser controlada com referência em 2 graus.

6.4 Discussão

Este capítulo apresentou a resolução de um problema proposto na planta piloto, que possui comportamento de um sistema híbrido guiado por eventos. Este problema aborda a síntese de supervisores discretos acoplados em uma planta híbrida.

A utilização da ferramenta *CheckMate* e uma extensão de funções sobre autômatos Grail permitiram resolver computacionalmente o problema de controle proposto. Contudo, o problema da complexidade algorítmica envolvida na verificação impõem uma severa limitação computacional para problemas mais complexos.

Os resultados obtidos até agora são teóricos, mas constituem um passo importante para uma implementação prática. O próximo capítulo trata da implementação prática do supervisor discreto na planta piloto.

Capítulo 7

Implementação de um supervisor C/E na planta piloto

Este capítulo mostra um procedimento para implementação de supervisores discretos como o projetado no cap.6. Será descrito aqui um pouco mais sobre a rede Fieldbus, em especial a rede Fieldbus da planta, denominada de Foundation Fieldbus. A arquitetura de rede, protocolos de comunicação terão uma breve descrição. Será tratado também o conceito de OPC, alguns servidores OPC disponíveis na planta e sua interface com o MATLAB. Por fim, é descrito o algoritmo de controle que introduz o supervisor discreto por um jogador de autômatos feito no MATLAB e interagindo com as variáveis contínuas da planta.

7.1 Arquitetura de rede da planta piloto

A rede Foundation Fieldbus é uma arquitetura aberta para a integração da informação. O sistema de comunicação Foundation é um sistema digital, serial e bidirecional. Esta rede se destaca das outras por ter a capacidade de distribuir o controle no campo. Isto é possível devido à inserção de processadores nos dispositivos de campo. Ela é regulamentada pela Fieldbus Foundation, uma organização internacional, sem fins lucrativos, formada pela união de mais de 185 companhias-membros que representam aproximadamente 90% do mercado mundial do mercado de instrumentação e controle de processos.

Topologia da Rede

A arquitetura de rede da planta piloto é distribuída e possui uma ponte denominada de DFI302 (Fieldbus Universal Bridge) que interliga os níveis de chão de fábrica com os níveis de usuário ou estação de trabalho. A fig.7.1 ilustra a topologia da rede

Foundation Fieldbus na planta piloto. O DFI302 é um elemento chave na arquitetura distribuída da planta. Este combina características de comunicação com acesso direto de entrada e saída (I/O) e controle para aplicações contínuas e discretas. Também provê serviços de comunicação para controle, utilizando o OPC, configuração e manutenção usando OLE.

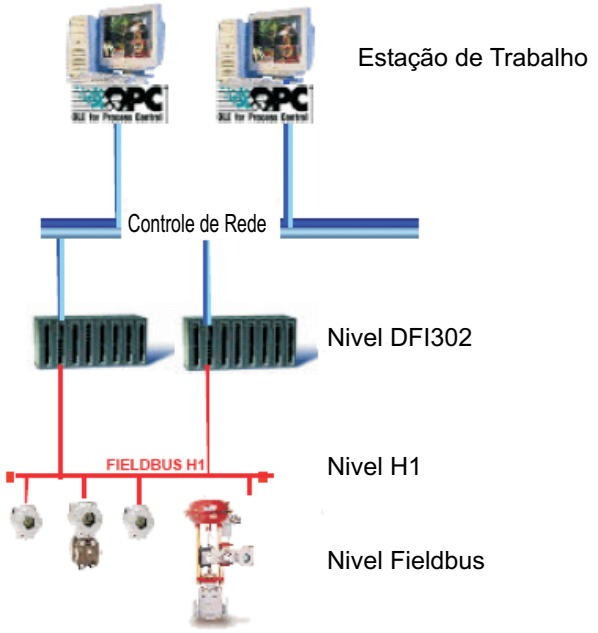


Figura 7.1: Topologia de rede da planta piloto

Os principais protocolos de comunicação do DFI são:

- Ethernet: é o protocolo que interliga os níveis mais altos, como controladores de alta velocidade, dos serviços de dados e de estações de trabalho, como mostra a fig.7.1. Este protocolo possui uma taxa de transmissão de $100 Mbits/s$ e é baseado no protocolo TCP/IP.

- Foundation Fieldbus: que é um dos mais avançados protocolos destinados para sistemas de automação. O nível H1 conforme mostra a fig.7.1 possui uma velocidade de $31.25 Kbit/s$ por razões de segurança intrínseca. É o nível responsável pela interconexão dos equipamentos de chão de fábrica tais como sensores, atuadores e dispositivos de I/O.

- Utilizando portas adicionais, o Protocolo Modbus conecta com os dados da rede Fieldbus virtualmente e com qualquer dispositivo da rede que esteja disponível.

7.2 OPC - Open Process Control

OPC é uma maneira unificada para conectar fontes de dados tais como dispositivos, base de dados, com aplicações cliente (interfaces homem-máquina). Esta tecnologia melhora a interface entre as aplicações cliente e servidor fazendo com que exista um mecanismo padrão para comunicação de uma fonte de dados para qualquer aplicação cliente. Em outras palavras, OPC é um mecanismo que habilita no campo a automação e as interfaces homem-máquina.

As principais características da tecnologia OPC são:

- É flexível - suporta diversas aplicações.
- Eficiência e escalável - suporta aplicações grandes e complexas.
- Facilidade de compreensão e de uso.
- Possui uma grande aceitação no mercado.

Para o caso da planta, os clientes OPC são conectados no servidor OPC através de um software backplane, ou seja, é um software que pode ser conectado diretamente por qualquer cliente OPC. Por fim, os servidores OPC se comunicam com as variáveis dos dispositivos inteligentes da planta. A fig. 7.2 ilustra a configuração OPC para a planta piloto.

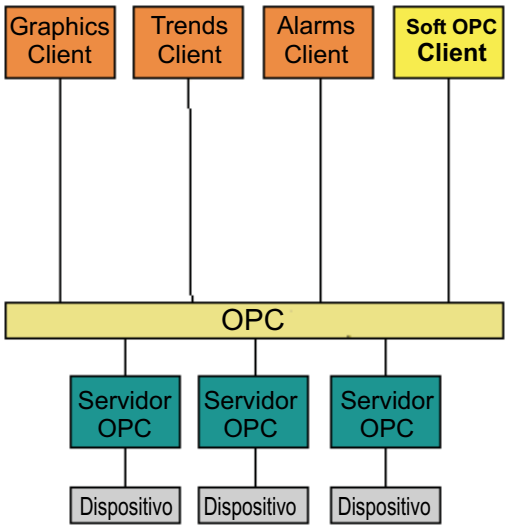


Figura 7.2: OPC na planta piloto

O servidor OPC que os softwares da planta, como o syscon, utilizam é o servidor com o nome *SMAR.DFIOLESRVER.0*. Com este servidor, é possível acessar os parâmetros de configuração que cada dispositivo inteligente da planta possui.

Recentemente um novo cliente OPC foi instalado na planta. Tal cliente possui um controle denominado de ActiveX, que possibilita a integração de dados com diferentes sistemas tais como fontes de banco de dados, linguagens de programação, etc. O controle ActiveX possibilita o software MATLAB acessar os servidores OPC da planta, em especial o servidor *SMAR.DFIOLESRVER.0*. Através de alguns comandos feitos no MATLAB, é possível acessar os parâmetros de configuração de cada dispositivo da planta e até mesmo modificá-los. O MATLAB utiliza o novo cliente OPC para se conectar aos servidores OPC da planta piloto.

7.3 Implementação do algoritmo de supervisão discreta por um servidor OPC conectado ao MATLAB

O algoritmo de implementação é um jogador de autômatos C/E conectado em um servidor OPC da planta[1]. Os eventos e condições do jogador estão associados com as variáveis T e h vindas da planta. Os controladores PIDs[2] estão embutidos na configuração da planta. Cria-se então uma configuração no Syscon que possui os blocos de nível, temperatura, controladores e atuadores. Para o nível, o atuador é o posicionador de válvula FY302 e para a temperatura é o conversor estático que está interligado com o dispositivo FI302, um medidor de corrente. Após isso, o operador descarrega esta configuração na planta, para que o experimento possa ser realizado. O mais importante é que o cliente OPC acessa as variáveis que estão no servidor OPC da última configuração descarregada do Syscon para os dispositivos da planta. Logo, tal configuração deve ser conferida para o correto funcionamento da implementação do supervisor discreto. Com relação às variáveis, o algoritmo tem capacidade de ler e setar parâmetros como ganho e integrador do PID, setpoints, etc. Para leituras e escritas das variáveis, o servidor OPC necessita de um atraso no algoritmo com o intuito de atualizar as variáveis, que varia de acordo com o n^o de dispositivos configurados na rede e se é leitura ou escrita de dados. Quanto mais dispositivos estiverem na rede, maior será o atraso. Assim ocorre com escritas, que necessitam de um atraso maior que a leitura de dados. Para este algoritmo em particular, a atualização das variáveis possui um atraso de 2 segundos. Com isso, se quisermos plotar um gráfico com 30 amostras do nível e temperatura, o tempo do experimento será de 1 minuto.

[1] A implementação do algoritmo também pode ser feita em outras linguagens de programação, como C, C++, java, etc. Para isso, é preciso obter uma biblioteca de comandos Activex para a linguagem escolhida com o intuito de se obter acesso às variáveis da planta.

[2] Os blocos de função dos dispositivos da planta são chamados de PID. Contudo, os controladores modelados são PIs, que são um caso particular dos PIDs. Logo, foi utilizado o bloco PID e configurou-se um controlador PI de acordo com a modelagem feita no capítulo 5.

O algoritmo de implementação do supervisor discreto sob abordagem do controle supervisório de sistemas híbridos possui sete passos distintos:

1. Conversão do arquivo em formato grail para uma matriz de transição. A fig.7.3 ilustra uma lista de instruções do grail representando um autômato C/E qualquer.

```
(START) |- 0
0 [0,1] 1
1 [1,5] 2
3 [3,5] 2
2 [2,6] 3
2 -| (FINAL)
```

Figura 7.3: Exemplo de uma lista de instruções de um autômato C/E

O termo *START* indica qual é o estado inicial e o termo *FINAL* são para os estados que possuem marcação. Os termos entre colchetes são os eventos e condições respectivamente. Os termos da 1^a coluna são os estados de partida das transições e os termos da 4^a coluna são os estados atingidos pelas transições. Uma função denominada de *grailce2mat* lê os números do arquivo do grail e os guarda em uma matriz, arquivada em formato *.mat*. Esta matriz possui n linhas por 4 colunas, onde a 1^a coluna significa o estado de onde sai a transição, a 2^a coluna significa os eventos pertencentes aos estados da 1^a coluna, a 3^a coluna são as condições referentes às colunas anteriores e a 4^a coluna são os estados atingidos pela transição vu. O número de linhas significa o tamanho do autômato em termos de transições. A seguir a matriz de transição equivalente à lista de instruções da fig.7.3.

$$\begin{matrix} 0 & 0 & 1 & 1 \\ 1 & 1 & 5 & 2 \\ 2 & 2 & 2 & 3 \\ 3 & 3 & 5 & 2 \end{matrix} \tag{7.1}$$

O algoritmo de conversão ordena todos as linhas pela ordem crescente dos estados de saída das transições e em seguida pela ordem crescente dos eventos servindo para facilitar o cálculo do algoritmo. No entanto, a marcação não é considerada na matriz. Isto se justifica que a marcação é um atributo teórico que serve para marcar estados que possuem tarefa completa. A marcação foi de grande importância na síntese do supervisor. Mas no caso desta implementação, o jogador irá percorrer todas as transições pertencentes a matriz de transição do autômato independente da marcação dos estados.

2. Conexão com o servidor OPC. O algoritmo configura o cliente OPC, conecta com o servidor OPC, adiciona as variáveis que se deseja utilizar, leituras e escritas das variáveis.

3. Nesta parte declaram-se alguns parâmetros importantes, como o tamanho da matriz de transição, estado inicial, variáveis auxiliares, etc.

4. Início do jogador de autômatos. Nesta parte se inicia um loop que termina de acordo com n^o de amostras das variáveis contínuas conectadas no servidor OPC, que se deseja analisar. Este loop contém o jogador de autômatos e a atualização das variáveis conectadas com o servidor OPC.

5. Eventos. Nesta parte localizam-se os eventos pertencentes ao estado inicial e através de uma função de configuração dos eventos, verifica-se qual o evento ocorreu. A função dos eventos descreve quais são os cruzamentos de patamares que podem ocorrer no estado inicial. Se não ocorreu nenhum evento, o algoritmo volta para o início do *loop* e repete-se as partes 3, 4 e 5. Se ocorreu algum evento, então ocorre o passo seguinte, que é localizar as condições associadas ao evento ocorrido.

6. Condições. Após a ocorrência de um evento, verifica-se quais são as condições associadas ao evento ocorrido e através de uma função não determinística, escolhe-se o valor da condição de maneira aleatória. Com o valor da condição, associa-se este valor uma dinâmica. Isto é feito através de uma função onde se declaram todas as dinâmicas possíveis de ocorrer na planta híbrida para cada valor da condição. Por fim, ocorrido o evento e determinado a condição, atualiza o estado do autômato e a dinâmica a ele associado e repete os passos de 3 até 6 para o estado atual.

7. Os passos de 3 a 6 ocorrem até que acabe o n^o de amostras, que então se encerra o *loop*, desconecta com o servidor OPC e fim do algoritmo

A fig.7.4 ilustra o fluxograma do algoritmo implementando o supervisor discreto na planta piloto. Cada retângulo pontilhado representa as partes do algoritmo que foram descritas anteriormente.

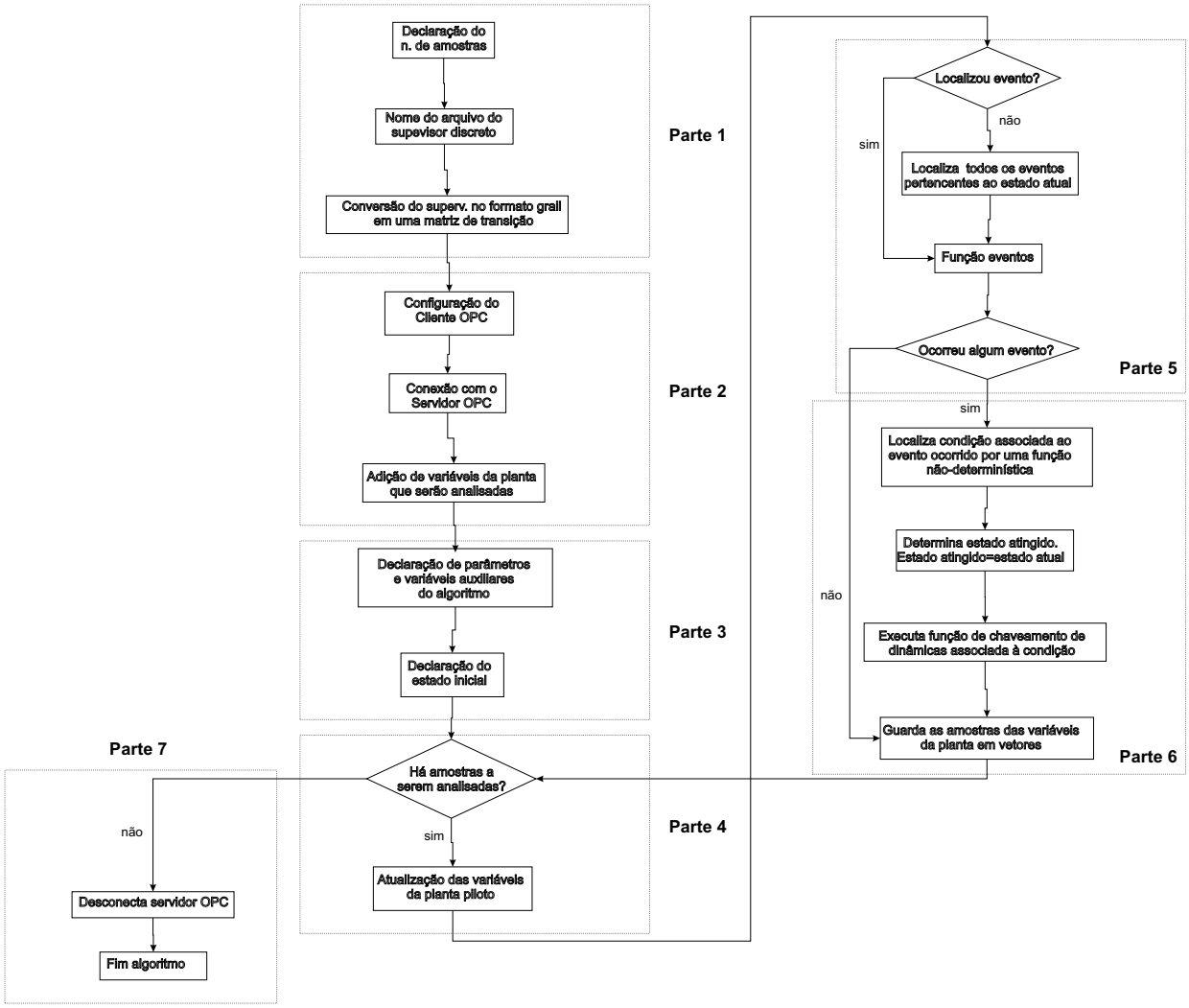


Figura 7.4: Fluxograma do algoritmo jogador de autômatos conectado à planta

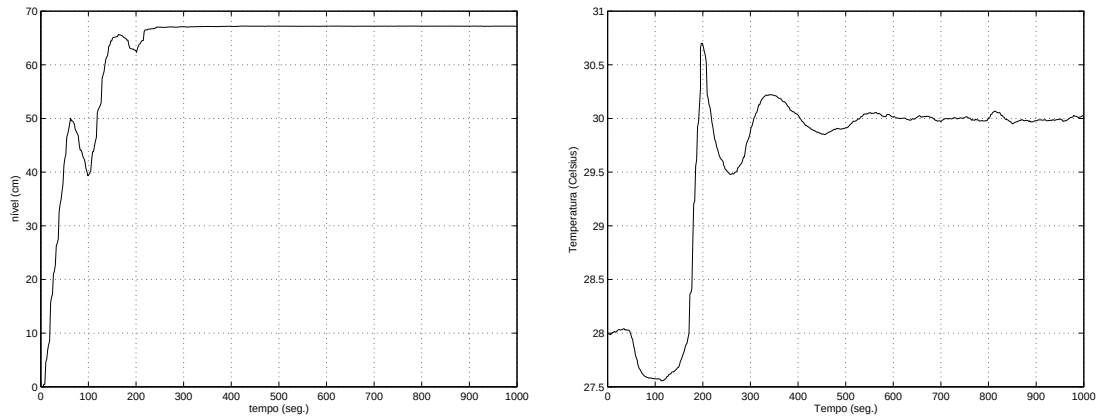
É importante ressaltar que o algoritmo descrito pelo fluxograma da fig. 7.4 é capaz de implementar supervisores para diferentes especificações. O que mudaria é o arquivo gerado pelo do grail devido a inserção de diferentes especificações e conseqüentemente a matriz de transição.

Também é possível a implementação de supervisores para outros sistemas híbridos (SH). Para isso, é necessário alterar as funções de configuração de eventos e condições do algoritmo. Isto porque a função eventos está associado com os guardas do modelo autômato híbrido de um supervisor para um SH qualquer. Basta alterar os guardas e o número de eventos de acordo com a modelagem escolhida. A função de configuração de condições está associado com as possíveis dinâmicas do sistema. Para um supervisor de um SH qualquer, basta alterar as equações, as variáveis e o número de dinâmicas possíveis de acordo com a modelagem escolhida. Deve ser checado se o supervisor em questão possui interface homem-máquina, isto é, se é possível obter acesso as suas variáveis em tempo real. Logo, cada supervisor tem seus próprios arquivos de configuração de eventos e condições.

O Apêndice B deste documento ilustra o código de instruções em MATLAB do algoritmo.

7.4 Resultados da implementação do algoritmo

A seguir é apresentado os resultados da implementação do algoritmo para o supervisor da fig.6.8.



(a) nível (b) Temperatura

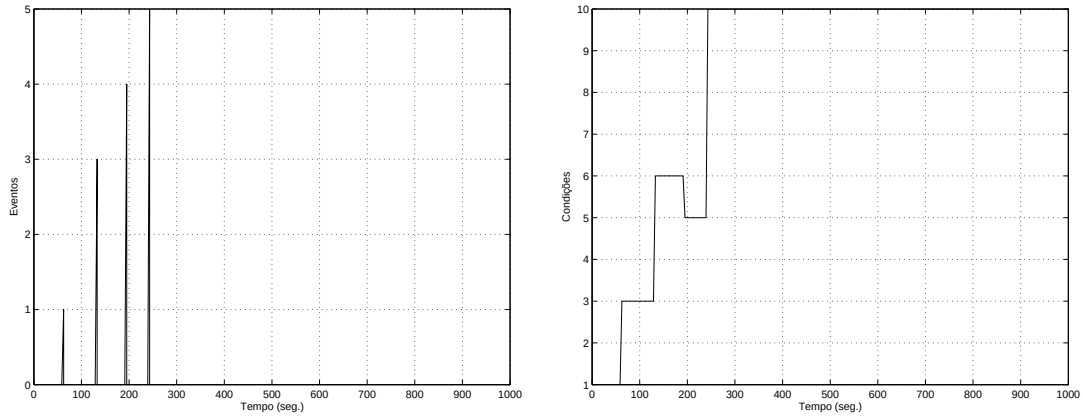
Figura 7.5: Gráficos representando o comportamento do nível e temperatura da água no tanque

Note que o nível inicia subindo, depois tem-se um queda e volta a subir. O controlador PI faz o nível atingir a referência e depois o nível sobe até encher o tanque.

A descida do nível conforme ilustra a fig.7.5(a) foi um fato inesperado e que ocorreu no instante em que o controlador, que antes era proporcional, passou a ser um PI. O controlador PID do nível, por questões de segurança, fez a válvula fechar totalmente por um certo tempo. A princípio o problema dever estar na atualização brusca do valor do T_i. O controlador PID não foi projetado para alterações bruscas nos seus parâmetros de configuração. Uma lógica de controle que varie gradativamente o valor de T_i poderia resolver este problema. Para o jogador de autômatos, o problema no PID não afetou o algoritmo porque não havia nenhum cruzamento de patamar que gerasse a ocorrência de um evento na descida do nível de água no tanque.

Pela curva da temperatura ilustrada na fig.7.5(b), nota-se que esta não foi afetada pelo chaveamento, pois ainda não havia aquecimento no tanque. Verifica-se no início do experimento uma queda de temperatura. Isto ocorreu porque foram feitas outros experimentos e provavelmente o tanque ficou com um resíduo de calor. Quando o tanque começou a encher com água fria, isto fez com o que a temperatura diminuísse e só voltou a aquecer quando a chave de nível atuou, habilitando o conversor estático a fornecer potência aos resistores.

Pela fig.7.6(a), nota-se que o supervisor segue a seqüencia de eventos dado pela especificação da fig.6.6. Pode-se observar que os eventos 1, 3, 4 e 5 ocorrem aproximadamente nos tempo de 60s, 120s 200s e 240s respectivamente. É no mesmo instante, acrescido de 2s de atraso, que o comportamento das variáveis continuas se alteram pela introdução da nova condição. Na fig.7.6(b) ilustra o comportamento das condições, que são constantes por partes e mudam de valor somente nos instantes em que ocorreu os eventos.



(a) Eventos (b) Condições

Figura 7.6: Gráficos com os eventos e condições do supervisor discreto

Foram realizadas outras duas implementações na planta, mas por coincidência, as condições associadas aos eventos foram os mesmos da fig.7.5. Coincidência pelo fato da função não determinística ser capaz de escolher outro valor de condição que pertença

ao supervisor da planta.

7.5 Discussão

Este capítulo apresentou os resultados da implementação do supervisor discreto da fig.6.8 na planta piloto através de um algoritmo que é um jogador de autômatos C/E conectado ao servidor OPC das variáveis contínuas da planta. Os resultados mostraram ser satisfatórios com relação à teoria de controle supervisório de sistemas híbridos, apesar do problema no chaveamento de controladores referente à locação 3 do autômato híbrido da fig.6.1. Logo, foi possível efetuar o controle de uma planta com características híbridas por um supervisor discreto, uma abordagem utilizada por (Cury e Niinomi 1998).

Vale ressaltar alguns pontos sobre o atraso na planta. Uma sugestão seria modelar o atraso que ocorre na atualização das variáveis do modelo como um clock entre uma dinâmica e outra, ficando assim, um modelo mais complexo e interessante. Isto é, o modelo autômato híbrido da planta teria uma variável de estado a mais e quase o dobro de locações em relação ao modelo original. No entanto, problemas de verificação com a ferramenta *CheckMate* devido ao aumento do número de locações do autômato híbrido e principalmente o aumento de variáveis de estado, tornariam a verificação intratável. Sabe-se que pesquisas recentes estão surgindo com o objetivo de resolver o problema de aspecto computacional e algorítmico que dificulta e muito a verificação de sistemas híbridos. Em um futuro não muito distante espera-se que este problema seja resolvido ou pelo o menos minimizado.

Capítulo 8

Conclusão e Perspectivas

Este capítulo reúne algumas conclusões e contribuições da pesquisa apresentada neste documento. Por fim algumas perspectivas de trabalho futuro são apresentadas.

8.1 Conclusões e Contribuições

Nesta dissertação foram estudados e implementados numa planta piloto a abordagem de controle supervisório de sistemas híbridos. Os aspectos de controle supervisório de sistemas híbridos foram abordados no cap. 4. A principal contribuição deste trabalho foi da abordagem utilizada por (Gonzáles 2000) e (Cury e Niinomi 1998) com aplicação prática. Foi feita uma modelagem sobre as características híbridas da planta piloto, modelo autômato híbrido e síntese de um supervisor discreto utilizando softwares acadêmicos, como o *CheckMate* e o Grail. Através destes resultados foi possível implementar um algoritmo que interpretasse o supervisor que controle a planta em tempo real, conforme o problema proposto do cap. 5. A implementação foi feita em um sistema híbrido relativamente simples, que possui poucas locações, mas pode mostrar, com resultados satisfatórios, a validação da teoria na prática, feita no cap. 7.

8.2 Perspectivas

Algumas sugestões de pesquisa sobre a implementação de supervisores discretos em sistemas híbridos reais são listados a seguir:

- Aperfeiçoamento de técnicas computacionais para a verificação e extração de aproximações do comportamento lógico de sistemas híbridos.

- Implementação do formalismo sistemas híbridos em plantas mais complexas. É sempre importante o início da implementação de um formalismo em sistemas reais mais simples. Contudo, isto nos leva pensar que sistemas reais mais complexos também podem ser estudados e analisados.

- A introdução da abordagem modular para o controle supervisório de sistemas híbridos (Leal 2002). A implementação do formalismo de controle supervisório de sistemas híbridos para plantas reais complexas se tornará mais atrativa com a utilização de abordagens mais apuradas, como a abordagem modular.

- Aperfeiçoamento e criação de ferramentas para verificação de sistemas híbridos. Considerando a ferramenta *CheckMate* e a dificuldade envolvida no cálculo do modelo aproximado[1] do comportamento lógico da planta híbrida é razoável considerar melhorias nesta ferramenta, como por exemplo uma melhor documentação, melhor interface homem máquina, diminuição de limitações, como a de n^o de variáveis de estado.

[1]Como descrito no cap.6 para este trabalho em particular foi obtido o modelo exato do comportamento lógico da planta híbrida.

Apêndice A

Projeto de Controladores utilizando a Teoria de Controle Clássica

A.1 Introdução

Este apêndice trata de um projeto de um controlador utilizando a teoria de controle clássico. Dadas especificações de controle, ir-se-á projetar um controlador que satisfaça as especificações. O projeto do controlador utiliza o método lugar das raízes, que é um método onde as raízes da equação característica são colocadas em um gráfico para todos os valores do ganho de malha aberta.

A.2 Projeto do controlador

Nível de água no tanque:

Esta seção dar-se início no projeto de um controlador. O processo escolhido a ser controlado é:

$$\frac{H(s)}{F(s)} = \frac{4.85}{163.05s + 1} \tag{A.1}$$

que utiliza os valores de ganho e constante de tempo ilustrados na tabela 6.1. Foi escolhido tais valores devido ao valor de referência se situar dentro do intervalo $60cm \leq h \leq 68cm$, que é o mesmo intervalo para os valores de ganho e constante de tempo do nível descritos na eq. A.1.

A.2.1 Especificações de Controle

O projeto do controlador deverá satisfazer as seguintes especificações:

- Sobre-sinal máximo[1] $SSM \leq 4\%$. Para evitar que o tanque encha.
- Tempo de acomodação[2] $t_s = 8s$. Usada para obter uma resposta rápida do sistema.

O primeiro passo é definir um controlador. O controlador será do tipo atraso de fase. Mais especificamente um PI, que possui um polo na origem.

O controlador é o mesmo da eq. A.2, mas reescrito da forma:

$$C(s) = k_c' \frac{T_i s + 1}{s} \tag{A.2}$$

onde $k_i = \frac{1}{T_i}$ e $k_c' = \frac{k_c}{T_i}$.

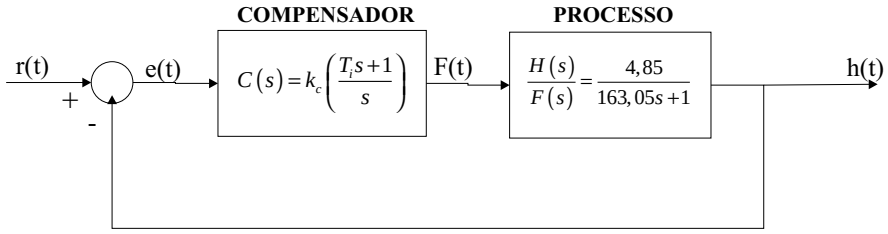


Figura A.1: Diagrama de Blocos do sistema nível-controlador

A.2.2 Cálculo dos pólos do Sistema com controlador em malha fechada

A equação característica do sistema da fig. A.1 é de segunda ordem do tipo

$$s^2 + 2\xi\omega_n + \omega_n^2 = 0 \tag{A.3}$$

que possui duas raízes do tipo:

[1] SSM é o maior desvio da saída para a entrada.

[2] t_s é o tempo que o sistema leva para entrar na região de regime permanente. Neste caso, o critério utilizado é de 2 %.

$$s_{12} = -\xi\omega_n \pm j\omega_n\sqrt{1-\xi^2} \tag{A.4}$$

Calcula-se os pólos de malha fechada (MF) da equação característica do sistema. Para isso,utiliza-se a definição do SSM que é:

$$SSM = exp(\frac{-\pi\xi}{\sqrt{1-\xi^2}}) \tag{A.5}$$

Para $SSM = 0,04$ tem-se que $\xi = 0,71564$. Utiliza-se agora a definição de t_s com o critério de 2%:

$$t_s = \frac{4}{\xi\omega_n} \tag{A.6}$$

substituindo o valor de $t_s = 8$ obtém-se $\omega_n = 0,69867$. Para calcular os pólos de MF, basta substituir os valores numéricos de ξ e ω_n na eq.A.4:

$$s_{12} = -0,5 \pm j0,34085 \tag{A.7}$$

Define-se então o ponto p_o do plano s da locação de pólos de MF. Tal ponto pode ser qualquer um dos pólos da eq. A.7, cujo módulo é:

$$|p_o| = 0,60513 \tag{A.8}$$

O compensador deverá introduzir uma fase quase nula em p_o e que o ganho em baixas freqüências permita aumentar o valor do ganho do sistema em δ vezes. Para isso o controlador terá um par pólos-zeros com a seguinte relação.

$$|p| < |z| << |p_o| \tag{A.9}$$

Para o controlador PI, $|p| = 0$. Isto significa que basta z ser diferente de zero para atender a primeira desigualdade da eq. A.9. Na segunda desigualdade da eq. A.9, a regra prática para o cálculo é escolher $|z|$ no mínimo dez vezes menor que $|p_o|$. Então o valor de $|z| = 0,060513$. Com isso, o valor de $T_i = z^{-1} = 16,53$. Para facilitar os cálculos escolheu-se $T_i = 17$.

O próximo passo é traçar o lugar das raízes para o ganho do sistema com controlador em malha aberta (MA).

A.2.3 Gráfico do lugar das raízes

Para se analisar o ganho do sistema com controlador em MA é de grande utilidade esboçar o gráfico do lugar das raízes (LR). Com este método temos todos os valores de ganho de zero até infinito e podemos analisar a posição dos pólos do sistema para cada valor deste ganho.

Dar-se início ao LR calculando a função de transferência em MA.

$$G(s) = C(s)\frac{H(s)}{F(s)} = 4,85k_c'\frac{17s+1}{163,05s^2+s} \tag{A.10}$$

Para o LR, calcula-se a FT de MA com um fator multiplicativo do tipo:

$$1 + KG'(s) = 0 \tag{A.11}$$

substitui-se a eq. A.10 na eq. A.11 e obtém-se

$$1 + K\frac{17s+1}{163,05s^2+s} = 0 \tag{A.12}$$

onde $K = 4,85k_c'$.

Pólos de $G(s)$: são os pontos onde $K = 0$.

Portanto, $p_1 = 0$ e $p_2 = -6,133 \cdot 10^{-3}$

Zeros de $G(s)$: são os pontos onde $K = \infty$.

Portanto, $z_1 = \infty$ e $z_2 = -5,882 \cdot 10^{-2}$

Determinação do n^o de ramos:

O número de ramos é o grau da equação característica. Portanto são dois ramos.

Ponto de teste sobre o eixo real $s = -1$:

A direita deste ponto há dois pólos e um zero, onde a soma total é três[3]. Portanto $s \in LR$

Intersecção das Assíntotas:

[3]O n^o ímpar da soma de pólos e zeros finitos significa que o ponto está na LR e n^o par significa LR Complementar.

Está sobre o eixo real e é único para LR.

$$\sigma_a = \frac{\Sigma\text{polos finitos} - \Sigma\text{zeros finitos}}{n - m} \tag{A.13}$$

Onde n e m são os n^o de pólos e zeros finitos respectivamente e σ_a é o ponto de intersecção das assíntotas. Logo, $\sigma_a = 0,0564$.

Ângulos da assíntotas:

É o ângulo entre $2(n - m)$ assíntotas. O cálculo do ângulo é feito pela seguinte equação:

$$\varphi_l = \pm\frac{180(2l + 1)}{n - m}(l = 0, 1, 2, ...|n - m|) \tag{A.14}$$

Logo, $\varphi_0 = \varphi_1 = 180$. As duas assíntotas possuem um ângulo de 180 graus entre si no ponto σ_a.

Ângulo de partida dos pólos e de chegada dos zeros:

Todos os pólos e zeros finitos são reais. Portanto o ângulo de partida para os pólos e de chegada para os zeros é de ± 180.

Intersecção com o eixo imaginário:

No pólo $s = j\omega = 0$ é a intersecção com o eixo imaginário. Logo $K = 0$ e $\omega = 0$.

Pontos de ramificação ou pontos sela:

Correspondem às raízes de ordem múltipla. Se n ramos se aproximam, n ramos deixam com o ponto de ramificação com espaçamento $\frac{180}{n}$. Logo , são dois ramos que deixam o(s) ponto(s) de ramificação com espaçamento de 90^o.

Condições para o cálculo do ponto de ramificação.

- condição necessária mas não suficiente: $d\frac{G'(s)}{ds} = 0$
- as soluções desta equação devem também satisfazer a eq. A.11 quando a solução for raízes complexas. Para raízes reais sempre existe um K que satisfaz a eq.A.11, bastando apenas verificar se os pontos pertencem ao LR ou ao lugar das raízes complementar (LRC).

Então faz-se

$$\frac{d}{ds}\left(\frac{17s+1}{163,05s^2+s}\right)=0 \tag{A.15}$$

após algumas manipulações algébricas, obtém-se a seguinte equação:

$$2771,85s^2+326,1s+1=0 \tag{A.16}$$

A solução da eq. A.16 é $s_1=-3,1509\cdot 10^{-3}$ e $s_2=-0,1145$. São raízes reais, portanto são pontos de ramificação. Ambos do LR, pois s_1 está a direita dos dois pólos e do zero, sendo portanto três pontos e o s_2 está à direita somente do pólo na origem, caracterizando assim, um ponto.

Na figura A.2 mostra o esboço gráfico do lugar das raízes com a região $\xi > 0,715$ onde os pólos de MF do sistema devem se situar na região cônica formada pelas duas retas.

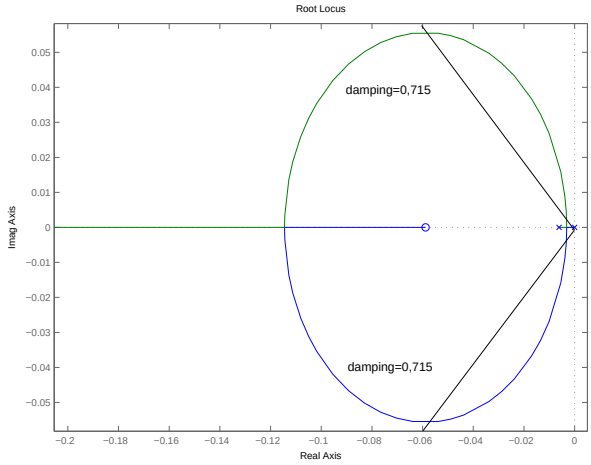


Figura A.2: Gráfico do lugar das raízes

Cálculo do módulo de K para o ponto $s_2=-0,1145$:

$$|K|=\frac{1}{|G'(s)|} \tag{A.17}$$

Substituindo s_2 na eq. A.17 obtemos $K=2,14$.

Cálculo do módulo de K para o ponto $s_{\xi\omega_n}=-\xi\omega_n=-0,5$, que é o valor da parte real de p_o. Substituindo s_{ω_n} na eq. A.17 obtemos $K=5,368$. Com isso, $k_c'=1,107$.

O gráfico da fig. A.3 ilustra o comportamento do sistema com o controlador PI.

Pode-se observar que as especificações do sistema não foram atingidas. Isto se deve ao fato de que o zero do controlador é muito pequeno, próximo dos pólos dominantes

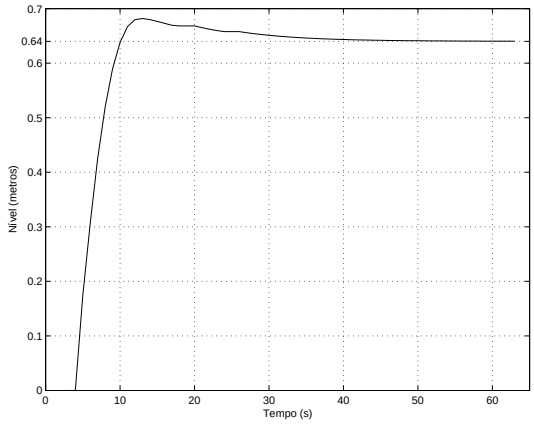


Figura A.3: Resposta do sistema com controlador

e também porque o cálculo dos pólos em MF feito na eq. A.4 é para uma função de 2^a ordem pura, não levando em conta o zero. Observa-se pela fig.A.2 que aumentando o valor de K, o SSM cai. Com isso, colocando o valor de em torno de 14,55 , teremos $k'_c = 3,0$. O gráfico da fig.A.4 ilustra a nova resposta do sistema:

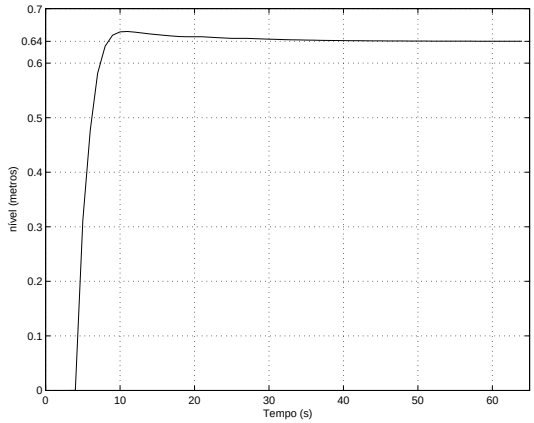


Figura A.4: Resposta do sistema com $k'_c = 3,0$.

Nota-se que o pico máximo foi de aproximadamente $0,66$ metros que corresponde a um SSM de $3,1\%$, satisfazendo a especificação. No caso do tempo de acomodação, tal especificação pode ser relaxada, pelo fato de que a especificação importante para este sistema seja o SSM, para que se evite que o tanque encha. Um fator importante não considerado aqui, mas considerado no problema tratado neste documento é a saturação da válvula. A vazão máxima é em torno de $0,5$ litros por segundo. Com isso, a resposta do sistema com saturação será bem mais lenta que a resposta sem saturação, que é o que ocorre na prática.

Aqui é ilustra-se um gráfico da planta piloto com o controlador de nível.[4]

[4]O atraso inicial do resposta é em virtude da demora do operador em ligar o processo com relação a plotagem do gráfico.

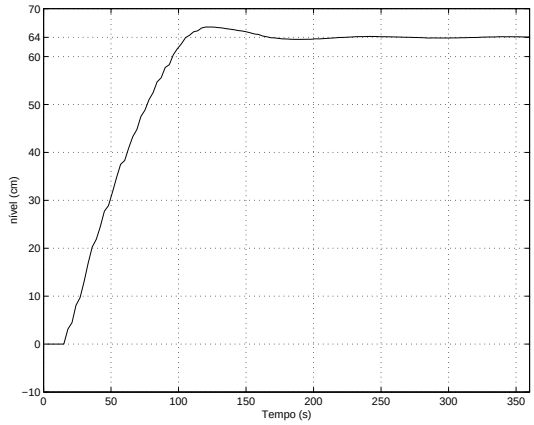


Figura A.5: Resposta do nível na planta piloto

Temperatura:

Para a temperatura, foi projetado um controlador PI para a função de transferência da eq. 5.28 com os valores da última linha da tab. 6.2. Utilizou-se a mesma metodologia feita para o nível de água e os resultados foram:

$$\begin{aligned} k_c &= 29,3 \\ T_i &= 18 \end{aligned} \tag{A.18}$$

A.3 Discussão

Este apêndice tratou de um projeto de um controlador PI feito passo a passo utilizando a teoria de controle clássica. Este projeto tem como intuito mostrar uma metodologia de projeto e não uma técnica de controle. Parâmetros de controle como as pertubações, filtros de referência e atraso de transporte não foram tratados, apesar de serem bastante relevantes para a teoria de controle, devido ao fato de que esta não é a proposta de trabalho deste documento e sim o controle de um sistema híbrido por um supervisor discreto sob a abordagem de Sistemas Híbridos e Sistemas a Eventos Discretos. Os controladores do ponto de vista de um sistema híbrido, são apenas um subconjunto do conjunto de todas as dinâmicas contínuas deste sistema. Por fim, ressalta-se que as especificações foram um meio utilizado para ser uma referência de projeto, mas o ajuste final se baseou em atender apenas uma especificação (o SSM para o nível e o t_s para a temperatura) e não foi levado em conta a saturação. Contudo, resultados práticos mostraram que a resposta do sistema nível com controlador foram satisfatórias, mesmo com a saturação real existente.

Apêndice B

Algoritmo de implementação no MATLAB

Este apêndice ilustra o código de instruções do algoritmo jogador de autômatos C/E conectado ao servidor OPC da planta piloto descrito no cap.7.

```
clear all

k=input('Informe o n. de amostras ')

To=input('Informe a temperatura ambiente ')

s=input('Informe o nome completo do supervisor ')

% Parte 1:
% Funcao que converte o arquivo do grail em uma matriz contendo
% a funcao de transicao do supervisor C/E

 [X, V, U, n, Mark, name] = grailce2mat(s);

 sortrows(X,[1 2]); % Ordena a matriz X de forma crescente em
 % relacao as colunas 1 e 2

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
 % Parte 2: Criacao dos ervidor OPC

OPC = actxserver('Softing.SOAXC.OPCDataControl.1') ;

% Selecao do servidor OPC
```

```
set(OPC,'ServerName', 'Smar.DfiOleServer.0');
set(OPC,'UpdateRate', 1000) ;

% Selecao das variaveis do processo
I = get(OPC,'Items') ;
invoke(I, 'AddItem', 'LD_AI.PV.VALUE') ;
% Nivel de agua no tanque
invoke(I, 'AddItem', 'FY_AO.OUT.VALUE') ;
% Saida do posicionador de valvula
invoke(I, 'AddItem', 'LD_PID.SP.VALUE') ;
% Setpoint do nivel
invoke(I, 'AddItem', 'LD_PID.GAIN') ;
% Ganho kp
invoke(I, 'AddItem', 'LD_PID.RESET') ;
% Ganho Ti

invoke(I, 'AddItem', 'TT_AI.PV.VALUE') ;
% Valor da temperatura
invoke(I, 'AddItem', 'FI_AO.OUT.VALUE') ;
% Saida do conversor de potencia
invoke(I, 'AddItem', 'TT_PID.SP.VALUE') ;
% Setpoint da temperatura
invoke(I, 'AddItem', 'TT_PID.GAIN') ;
% Ganho kpt
invoke(I, 'AddItem', 'TT_PID.RESET') ;
% Ganho Tit

% Obter acesso individual a cada uma das variaveis
ItemH      = get (I, 'Item',0);
ItemFY_AO  = get (I, 'Item',1);
ItemPID_SP = get (I, 'Item',2);
ItemPID_kp = get (I, 'Item',3);
ItemPID_Ti = get (I, 'Item',4);

ItemT        = get (I, 'Item',5);
ItemFI_AO    = get (I, 'Item',6);
ItemTTPID_SP = get (I, 'Item',7);
ItemTTPID_kp = get (I, 'Item',8);
ItemTTPID_Ti = get (I, 'Item',9);

% Conexao ao servidor OPC e atualizacao das variaveis
invoke(OPC, 'Connect')
invoke(OPC, 'Update');
```

```
get (ItemH, 'Value');
get (ItemT, 'Value');

pause(2)

invoke(OPC, 'Update');
get (ItemT, 'Value')

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%   Parte 3: Declaracao de parametros utilizados ao longo
% do algoritmo

[m,n]=size(X);  % Informa as m linhas e n colunas de X
UU=[];          % Vetor Sinal condicao
P=[];           % Vetor Sinal Evento
state=X(1,1);   % Informa o estado inicial
                % Sempre sera o primeiro elemento da matriz por
                % causa do comando "sortrows"
q=100;
tic
cont = 1;
Time = [];
fq=0;
ff=1;
flag=0;
fv=0;
To=26;          % Temperatura ambiente

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% Parte 4:
% Testa  o automato atraves de um jogador de automatos C/E

while cont<=k     % Numero de amostras que se deseja analisar

if fq==0          %   Teste para evitar que na ocorrencia de um
                  % evento o comando pause seja executado 2 vezes

   invoke(OPC, 'Update'); % Atualizacao das variaveis
   pause(2)     % Pausa a execucao do algoritmo em 2 seg
end
```

```
% Principais variaveis da planta piloto
   h(cont) = get (ItemH, 'Value');       % Nivel
   T(cont) = get (ItemT, 'Value');       % Temperatura
   Q(cont) = get (ItemFI_AO, 'Value');   % Potencia
   v(cont) = get (ItemFY_AO, 'Value');   % Valvula

fq=0;

%  Parte:5 Associacao dos eventos com os cruzamentos de
% patamares.
if ff == 1  %  Teste para que a rotina seguinte ocorra uma
            % unica vez apos encontrar o estado.
for i=1:m % Localiza o estado atual na matriz X

    if state == X(i,1)
        ff=0;
        break;
    end
end
end

j=i;

while state==X(j,1) %  Testa todos os eventos possiveis de
                    % ocorrer pertencentes a um estado
    p=X(j,2);
    [fv]= eventos(cont, p ,h, T, To); % Funcao eventos
    if fv==1
        disp (p)
        break;
    end
    j=j+1;

    if j>m
       break;
    end

end

 P(cont)=fv*p; % Valor do sinal evento

%   Parte 5: Verifica-se a condicao associada e estado
```

```
% atingido apos uma ocorrencia  de um evento

if  fv==1 %   Indica que ocorreu um evento no estado atual

                [u, state] = nao_determinismo_temp(j, m, p, X)
                %   Funcao  que fornece o valor de u nao
                % deterministico associado ao evento corrente

                flag=1; % Habilita a funcao condicoes
                fq=1;   % Desabilita atualizacao inicial
                        % de variaveis
                ff=1;   % Habilita localizacao de estado
                fv=0;   % Indica que nao ocorreu nenhum evento
                        % no estado atual
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

UU(cont)= u; % Sinal condicao

    % Parte 6:

    if flag==1      % teste para que a condicao seja aplicada
                    % somente na ocorrencia de um evento
                    % condicao aplicada

q = condicoes_temperatura(OPC, To, ItemPID_SP, ItemPID_kp,
ItemPID_Ti, ItemTTPID_SP, u)

fq=1;
    end

    Time(cont)=toc;
   [h(cont) T(cont) P(cont) UU(cont) Time(cont)];

    cont=cont+1; % Atualiza contador de amostras
    flag=0; %   Zera o flag para que se aplique as condicoes
            % somente quando a condicao fv=1 (ocorreu um evento)
            % for satisfeita
end

save data h T P UU  Time %  Salva as amostras das variaveis em
                         % um arquivo de extensao .mat
```

```
invoke(OPC, 'Disconnect') % Fim do algoritmo
```

Algoritmo B.1: Corpo principal do Jogador de Autômatos.

```
function [X, V, U, n, Mark, name] = grailce2mat(file)

% Converte o arquivo .do para .mat

if ~exist(file),
    error(['File ' char(file) ' not found!']);
end
[path,name,ext] = fileparts(file);
ini = 0;
Mark = [];
X = [];
V=[];  % vetor de eventos
U=[];  % vetor de condicoes
fid = fopen(file);
s=fgetl(fid);
while ischar(s)
    if ~isempty(strfind(s,'|- '))
        ini = sscanf(s,'(START) |- %u ');
    elseif ~isempty(strfind(s,' -|'))
        Mark = [Mark sscanf(s,'%u -| (FINAL)')];
    else
        Trans = sscanf(s,'%i [%i,%i] %i');
        if length(Trans)==4
            X = [X; Trans'];
            if ~any(V==Trans(2))
                V = [V Trans(2)];
            end
            if ~any(U==Trans(3))
                U = [U Trans(3)];
            end
        end
    end
    s=fgetl(fid);
end

V=sort(V);
U=sort(U);
```

```
X=sortrows(X,[1 2]);

n=max([X(:,1);X(:,4)])+1;
fclose(fid);

disp(X)
```

Algoritmo B.2: Conversão do arquivo no formato Grail para uma Matriz.

```
function[fv]= eventos(cont, p ,h, T)

% Funcao que verifica a ocorrencia ou nao de eventos
% pertencentes ao estado atual
% Funcao especifica para o automato hibrido da fig. 6.1

switch p                    % Variavel evento

case 0                      % Patamar çInicializaao da planta

    if h(cont)>=-1

        fv=1;               % Ocorreu o evento 0
    else
        fv=0;               % Nao ocorreu eventos
    end

case 1

    if h(cont)>=52          % Patamar chaveamento de controladores

        fv=1;               % Ocorreu o evento 1
    else
        fv=0;               % Nao ocorreu eventos
    end

case 3
    if h(cont)>=60          % Patamar Indica resistencias ligadas

        fv=1;
    else
        fv=0;
```

```
    end


case 4

    if T(cont)>=2+To         %  Patamar Encher o tanque

        fv=1;
    else
        fv=0;
    end

case 5

    if h(cont)>=67          % Patamar Tanque esta cheio

        fv=1;
    else
        fv=0;
    end


case 7

     if T(cont)>=3+To    % Patamar Temperatura muito alta

        fv=1;
     else
        fv=0;
     end

case 6
    if h(cont)<=60       % Patamar Desliga resistencias

        fv=1;
    else
        fv=0;
    end

case 2
    if h(cont)<=1        %  Patamar Tanque vazio

        fv=1;
```

```
    else
        fv=0;
    end

end

return
```

Algoritmo B.3: Função de Configuração de Eventos.

```
function q = condicoes_temperatura(OPC, To, ItemPID_SP,
             ItemPID_kp, ItemPID_Ti, ItemTTPID_SP,  u)

% Associando as dinamicas possiveis a cada estado atraves da
% condicao u

switch u, %Variavel condicao

case 1 % Estado inicial - tanque enchendo

    set(ItemPID_Ti, 'Value',100000);% Ti
    pause(2)
    invoke(OPC, 'Update');
    set(ItemPID_kp, 'Value',50); %   Nivel com Controle
    pause(2)                     % Proporcional
    invoke(OPC, 'Update');
    set(ItemPID_SP, 'Value',60); % Setpoint do nivel em 60cm
    invoke(OPC, 'Update');
    pause(2)

    q= get (ItemPID_SP, 'Value');% Artificio para que a presente
        % funcao seja aceita. Variavel sem funcao no algoritmo.

case 3 % Chaveia o controle de nivel proporcional - PI
   set(ItemPID_kp, 'Value',3);  % Controle PI Kp
   set(ItemPID_Ti, 'Value',18); % Ganho Ti
   invoke(OPC, 'Update');       % Atualizacao das variaveis
   pause(2)                     % Pause para atualizacao

   q= get (ItemPID_SP, 'Value');
```

```
case 2 % Tanque esvaziando com temp. desligada
    set (ItemPID_SP, 'Value', 0);
    invoke(OPC, 'Update');
    pause(2)
    q= get (ItemPID_SP, 'Value');

case 4  %  Liga o conversor, temp em malha aberta e tanque
        % enchendo

   set(ItemPID_SP, 'Value',67); %  Setpoint do nivel para tanque
                                % cheio
   set(ItemTTPID_SP, 'Value',10);
   invoke(OPC, 'Update');       % Atualizacao das variaveis
   pause(2)                     % Pause para atualizacao
   q= get (ItemPID_SP, 'Value');

case 5  %  Liga o conversor, temp com controle PI e tanque
        % enchendo

   set(ItemPID_SP, 'Value',67); %   Setpoint do nivel para
                                % tanque cheio
   set(ItemTTPID_SP, 'Value',To+2); % Referencia da temperatura
                                    % igual a 2
   invoke(OPC, 'Update');       % Atualizacao das variaveis
   pause(2)                     % Pause para atualizacao
   q= get (ItemPID_SP, 'Value');

case 6 %  Liga o conversor, temp em malha aberta com PI em 64cm
       % no nivel

    set(ItemTTPID_SP, 'Value',10); % malha aberta
    set(ItemPID_SP, 'Value',64); % Setpoint do nivel em 64cm
    invoke(OPC, 'Update');       % Atualizacao das variaveis
    pause(2)                     % Pause para atualizacao
    q= get (ItemPID_SP, 'Value');

case 7  %  Liga o conversor, temp com controle PI e com
        % controle PI em 64cm no nivel
   set(ItemTTPID_SP, 'Value',To+2);
   set(ItemPID_SP, 'Value',64); % Setpoint do nivel em 64cm
   invoke(OPC, 'Update');
   pause(2)
   q= get (ItemPID_SP, 'Value');
```

```
case 8  % O tanque comeca a esvaziar

    set (ItemPID_SP, 'Value', 0); %Valvula em 0%
    invoke(OPC, 'Update');
    pause(2)
    q= get (ItemPID_SP, 'Value');

case 10 % Controle de temperatura com tanque cheio: SUCESSO

    set(ItemTTPID_SP, 'Value',To+2);
    invoke(OPC, 'Update');
    pause(2)
    q= get (ItemPID_SP, 'Value');

case 9 % Temperatura em malha aberta e tanque cheio

    set(ItemTTPID_SP, 'Value',10);
    invoke(OPC, 'Update');
    pause(2)
    q= get (ItemPID_SP, 'Value');

case 11 % Estado indesejado, tanque vazio

    invoke(OPC, 'Update');
    pause(2)
    q= get (ItemPID_SP, 'Value');
end
return
```

Algoritmo B.4: Função de Configuração de Condições.

```
function [u, state] = nao_determinismo_temp(j, m, p, X)

% Funcao para gerar uma condicao u nao-deterministica
B=[];
b=j;                % Guarda alinha da matriz x
c=1;                % Declara o indice de B

while X(b,2)==p     %  Testa quantas çcondioes estao
                    % associadas ao evento p
    B(c)=X(b,3);    % Guarda o valor do sinal u no vetor linha B
    c=c+1;          % Atualiza contadores
    b=b+1;
    if b>m          % Evita erro de dimensao de X
    break;
    end
end

z=length(B);        % Calcula o tamanho do vetor B e guarda em z

d=randperm(z);      %   Gera o vetor d de tamanho igual ao B
                    % mas com os indices de B como elemetos
                    % permutados aleatoriamente no vetor d

u=X(d(1)+b-c,3); %   u recebe o valor de X com o indice contendo
                 % o primeiro elemento de d, que eh
                 % um indice de B aleatorio, mais a defazagem
                 % dos contadores de X e B

state=X(d(1)+b-c,4); % state recebe o valor do estado a ser
                     % atingido associado a condicao u
                     % anterior.
return
```

Algoritmo B.5: Função de escolha da condição de forma não-determinística.

Referências Bibliográficas

Allur, R. e Dill, D. L. (1994). A theory of timed automata,, *Theorycal Computer Science* **126(2)**: 183–235.

Alur, R. Courcoubetis, C. e Dill, D. (1990). Model checking for a real-time systems, 5^{th} *IEEE Symposium on Logic in Computer Science, pp. 414-425.*

Alur, R. Courcoubetis, C. H. T. A. e. H. P. H. (1993). Hybrid automata: an algoritmic approach to the specification and verification of hybrid systems, *in Grossman et al.* p. 209–229.

Antsaklis, P. J. e Koutsoukos, X. D. (2002). Hybrid systems: Review and recent progress., *Chapter in Software-Enabled Control: Information Technologies for Dynamic systems.* **88(7)**: 879–557.

Antsaklis, P. J. Stiver, J. A. e Lemmon, M. D. (1993). Hybrid systems modeling and autonomous control systems, *in Grossman et al* p. 366–392.

Antsaklis, P. J. (2000a). A brief introducion to the teory and aplications of hybrid systems, *Proceedings of the IEEE, Special Issue on Hybrid Systems, Theory and applications* .

Antsaklis, P. J. (2000b). Special issue on hybrid systems, theory and applications, *Proceedings of the IEEE* .

Chutinan, A. e Krogh, B. H. (1999a). Computing approximating automata for a class of linear hybrid systems., *In Hybrid Systems V, LNCS 1567, Springer-Verlag* p. 16–37.

Chutinan, A. e Krogh, B. H. (1999b). Verification of polyhedral-invariant hybrid automata using polygonal flow pipe approximations, *In Hybrid systems: Computation and Control, Second International Workshop, Springer-Verlag* p. 76–90.

Chutinan, A. e Krogh, B. H. (2001). Verification of infinite-state dynamic systems using approximate quotient transition systems, *IEEE Trans. on Automatic Control* **46(9)**: 1401–1410.

Chutinan, A. (1999). *Hybrid system verification using discrete model approximations*, Ph.d. in electrical engeneering, Carnegie Mellon University, Pittsburgh, USA.

Cury, J. E. R. Krogh, B. H. e Niinomi, T. (1998). Synthesis of supervisory controllers for hybrid systems based on approximating automata, *IEE Transations on Automatic Control, Special issue on Hybrid Control Systems* **43(4)**: 564–568.

Egerstedt, M. (2000). Behavior besed robotics using hybrid automata, *Third International Workshop, HSCC 2000* .

Ferrari-Trecate, e. a. (2002). Modeling and control of co-generation power plants:, 5^{th} *International Workshop, HSCC 2002* p. 209–224.

Garcia, T. R. (2002). *Controle supervisório de sistemas a eventos discretos: Uma abordagem por modelo condição/evento*, Dissertação de mestrado, Universidade Federal de Santa Catarina, Florianópolis, Brasil.

Gonzáles, J. M. E. da Cunha, A. E. C. C. J. E. R. e Krogh, B. H. (2001). Surpevision of event-driven hybrid systems: Modeling and syntesis, 4^{th} *International Workshop, HSCC*, Rome, Italy.

Gonzáles, J. M. E. (2000). *Aspectos de síntese de surpevisores para sistemas a eventos discretos e sistemas híbridos*, Tese (doutorado), Universidade Federal de Santa Catarina, Florianópolis, Brasil.

Halliday, D. Resnick, R. e Walker, J. (1997). *Fundamentos de Física 2,* 4^a *ed.*, Livros Técnicos e Científicos Editora S.A.

Hopcroft, J. E. e Ullman, J. D. (1979). Introducion to automata theory, languages and computation, *Addison-Wesley* .

Khalil, H. K. (1996). *Nonlinear systems*, Prentice Hall.

Krogh, B. H. (2002). Recent advances in discrete analysis and control of hybrid systems, 6^o *International Workshop on Discrete Event Systems*, Zaragoza, Spain.

Leal, A. B. E. e Cury, J. E. R. (2004). Modular supervision of hybrid systems: A des approach, 7^{th} *Workshop on Discrete Event Systems.*

Leal, A. B. (2002). *Controle supervisório modular de sistemas híbridos*, Exame de qualificação (doutor), Universidade Federal de Santa Catarina, Florianópolis, Brasil.

Lennartson, B. Tittus, M. E. B. e Pettersson, S. (1996). Hybrid systems in process control, *IEEE*, Control Engeneering Lab. Chalmers University of Thecnology, Gothenburg, Sweden.

Liberzon, D. e Morse, A. S. (1999). Basic problems in stability and design of switched systems,, *IEEE Control Systems Magazine* **19(5)**: 59–70.

Livadas, e. a. (2000). High-level modeling and analysis of the traffic alert and collision avoidance system, *IEEE, Special Issue of Hybrid Systems: Theory and Aplications* **88(7)**: 926–948.

Maler, O. Manna, Z. e Pnueli, A. (1992). From timed to hybrid systems, *in the Bakker et al.* p. 474–484.

Manna, Z. e Sipma, H. (1998). Deductive verification of hybrid systems usin step, *First International Workshop, HSCC 98, Berkeley, California, USA* **11(6)**: 665–683.

Miranda, M. V. C. e Lima, A. M. N. (2001). Using hyibrid automata to model power eletronic circuits,, *VI Congresso brasileiro de Eletrônica de Potência - COPEB 2001* .

Moor, T. Davoren, J. M. e Raish, J. (2001). Modular supervisory control of a class of hybrid systems in behavior framework, *Proc. euproean Control Conference ECC2001, Porto Portugal* p. 870–875.

Nerode, A. e. K. (1993). Models of hybrid systems: Automata, topologies, controllability, observability,, *in Grossman et al* p. 317–356.

Ogata, K. (1998). *Engenharia de Controle Moderno* 3^a *ed.*

Pepyne, D. L. e Cassandras, C. G. (2000). Optimal control of hybrid systems in manfacturing, *IEEE, Special Issue of Hybrid Systems: Theory and Aplications* **88(7)**: 1108–1123.

Puri, A. e Varaiya, P. (1994). Decidability of hybrid systems with rectangular differential inclusions, **818**: 95–104.

Raish, J. e O'Young, S. D. (1998). Discrete approximation and supervisory control of continuous systems, *IEEE Transacitons on Automatic Control, Special issue on Hybrid Control Systems* **43(4)**: 569–573.

Rico, J. E. N. (2000). *Apostila de Controle Clássico.*

Silva, B. I. e Krogh, B. H. (2000). Formal verification of hybrid system using checkmate: A case study, *American Control Conference.*

Silva, B. I. Richeson, K. K. B. H. e Chutinan, A. (2000). Modeling and verification of hybrid dynamical system using checkmate, *ADPM 2000.*

Silva, B. I. Stursberg, O. K. B. H. e Engell, S. (2001). An assessment of the current status of algorithmic, approaches to the verification of hybrid systems, *Conference on Decision and Control*, Carnegie Mellon University, Pittsburgh, USA.

Sreenivas, R. S. e Krogh, B. H. (1991). On condition/event systems with discrete state realizations., *Discrete Event Dynamic Systems: Theory and Applications* p. 1:209–236.

Stephanopoulos, G. (1984). *Chemical Process Control An introducion to theory and practice*, Prentice Hall.

Stiver, J. A. Antsaklis, P. J. e Lemmon, M. D. (1995a). Hybrid control system design for hybrid control systems, 34^{th} *IEEE Conference on Decision and control,* .

Stiver, J. A. Antsaklis, P. J. e Lemmon, M. D. (1995b). Interface and controler designs for hybrid control systems, *Hybrid Systems II, Spribger-Velag* .

Tavernini, L. (1987). Differencial automata and their discrete simulators, *Nonlinear Analysis, Theory, Methods and Aplications* **11(6)**: 665–683.

van der Shaft, A. e Schumacher (2000). *An Introducion to Hybrid Dynamics Systems, Vol. 251 of Lecture Notes in Control and Informtion Sciences.*

Wonham, W. M. e Ramadge, P. J. (1987). On the supremal controllable sublanguage of given language, *SIAM, Journal of Control and Optimization* **25(3)**: 637–659.