Análise, Verificação e Síntese de Segmentos de Via de uma Malha Metroviária

Adilson Luiz Bonifácio *Arnaldo Vieira Moura*
João B. Camargo Jr. *Jorge R. de Almeida Jr.*

Relatório Técnico IC–99-18

Agosto de 1999

Análise, Verificação e Síntese de Segmentos de Via de uma Malha Metroviária

Adilson Luiz Bonifácio* Arnaldo Vieira Moura João B. Camargo Jr.†
Jorge R. de Almeida Jr.‡

Resumo

O objetivo desse trabalho é a aplicação de técnicas de especificação formal para modelar sistemas distribuídos realistas. Os modelos formais são baseados em autômatos híbridos. O sistema alvo desse trabalho são segmentos de via de uma malha metroviária. A análise e a verificação do modelo considerado, bem como a síntese de certos parâmetros importantes do modelo, são auxiliadas por ferramentas semi-automáticas. Dessa forma, obteve-se de maneira automática a validação do comportamento do sistema descrito, bem como a síntese de parâmetros críticos para a operação da malha.

Palavras-Chave: Análise, Síntese, Sistemas Híbridos, Sistema Metroviário, Verificação.

1 Introdução

O uso de especificações formais no processo de desenvolvimento de sistemas tem se tornado cada vez mais indispensável no contexto atual da computação, principalmente no desenvolvimento de sistemas distribuídos e de sistemas que envolvem aplicações críticas. Os chamados sistemas críticos englobam sistemas reativos e sistemas de tempo real, onde uma falha na operação do sistema pode causar enormes prejuízos ou danos irreparáveis. As especificações descritas através de uma linguagem formal, ou através de um modelo matemático, possuem características bastante interessantes para utilização nessas aplicações, tal como a eliminação de inconsistências e ambigüidades que, potencialmente, são encontradas em especificações de projetos [BG98]. Dessa forma, o número de erros na fase de implantação e implementação dos sistemas analisados é sensivelmente diminuído, uma vez que o número de erros gerados na fase de especificação tende a diminuir drasticamente.

Certas aplicações críticas apresentam características de sistemas distribuídos híbridos. Sistemas distribuídos híbridos, de um modo geral, são sistemas resultantes da interconexão de sistemas de dinâmica contínua com sistemas de dinâmica discreta. Ou seja, são sistemas que apresentam características de dinâmica contínua regulada pela intervenção de eventos discretos. Uma das formas de se especificar tais sistemas é através da utilização de

*Suporte Parcial da CAPES, DS - 44/97

†Escola Politécnica, Universidade de São Paulo, Av. Prof. Luciano Gualberto, Trav. 3, N. 158, São Paulo, SP, 05508-900

‡Escola Politécnica, Universidade de São Paulo, Av. Prof. Luciano Gualberto, Trav. 3, N. 158, São Paulo, SP, 05508-900, USP

autômatos híbridos [AHH93]. Os autômatos híbridos, resumidamente, são autômatos finitos cujos estados determinam o comportamento dinâmico do sistema e cujas transições entre estados provocam uma alteração no perfil dinâmico do sistema. Diferentes componentes do sistema distribuído são modelados por autômatos independentes. A comunicação entre os vários componentes do sistema é regulada por mensagens trocadas entre os autômatos independentes, e também por uma memória compartilhada à qual os autômatos componentes têm acesso. O comportamento global do sistema é obtido da cooperação entre esses autômatos independentes. Normalmente, teorias clássicas de controle de sistemas discretos lidam, se tanto, com poucos componentes distribuídos cujas características dinâmicas são complexas. Sistemas modelados usando a noção de autômatos híbridos, por outro lado, apresentam vários componentes distribuídos cujas características dinâmicas são mais simples, porém determinantes.

Nesse trabalho, o sistema modelado abrange regiões de uma malha metroviária. Na sua totalidade, o sistema metroviário é bastante complexo. Embora de topologia relativamente simples, o sistema apresenta uma elevada quantidade de subcomponentes e sensores, espalhados por toda a extensão da malha metroviária. Além disso, há intensa troca de mensagens e uma complexa atividade de coordenação entre esses componentes físicos. Para contornar essa complexidade, o sistema global foi dividido em segmentos mais simples, os quais, então, foram modelados. Os segmentos selecionados enfocam propriedades distintas, e a conjunção dos resultados da análise dos vários segmentos contribuem para a validação e a verificação do sistema total. Um dos segmentos críticos da malha metroviária é a região situada no final da via. Essa parte da malha é utilizada como um pátio de manobras para permitir o retorno dos trens que alcançam o fim da linha.

Na maioria das vezes, modelos construídos para sistemas realistas resultam num autômato de tal complexidade que se torna imprescindível a cooperação de ferramentas automáticas para análise da especificação produzida. Nesse trabalho, para analisar os modelos aqui desenvolvidos, foi utilizada a ferramenta HyTech [HH94], que é um verificador de autômatos híbridos lineares.

As próximas seções estão organizadas da seguinte forma. A seção 2 apresenta a noção de autômatos híbridos como um mecanismo para a especificação de sistemas híbridos realistas. São apresentadas a definição desses autômatos, suas características e uma ferramenta de apoio utilizada para a análise dos modelos produzidos. A seção 3 contém a descrição da malha metroviária analisada, apresentando suas funcionalidades e sua operação. A seção 4 descreve os modelos desenvolvidos, apresenta a análise e a verificação de suas propriedades, bem como descreve a síntese de certos parâmetros do sistema modelado. A última seção conclui com sugestões para trabalhos futuros.

2 Autômatos Híbridos

Os autômatos híbridos vêm se tornando um dos principais formalismos utilizados para se especificar sistemas híbridos, isto é, sistemas cujo comportamento apresenta uma combinação de características discretas e contínuas. Basicamente, autômatos híbridos são autômatos finitos cujos estados descrevem o comportamento dinâmico do sistema modelado. Transições entre estados caracterizam uma mudança de perfil dinâmico do sistema alvo. Os vários componentes que fazem parte do sistema são modelados por autômatos independentes, e o comportamento do sistema como um todo é obtido da interação e integração dos autômatos independentes. Transições de estado num dos componentes são sinalizadas por eventos, ou mensagens, recebidos de outros componentes ou do ambiente externo. O sistema global é modelado pelo autômato resultante do produto dos autômatos componentes. As próximas subseções apresentam o formalismo em maiores detalhes.

2.1 Um Exemplo Simples: O Controle de um Aquecedor

Um exemplo bastante simples é mostrado na Figura 1 [HHWT97]. O modelo é um sistema dinâmico euclidiano, descrito por um conjunto de variáveis reais evoluindo no tempo deterministicamente. O sistema é caracterizado por um ponto no R^n, onde n é o número

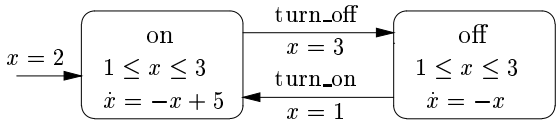

Figura 1: Autômato do Aquecedor

de variáveis reais usadas no modelo. No caso do exemplo, n vale 1. Uma trajetória do sistema é uma curva no R^n. O modelo é constituído por modos de operação. Cada modo de operação recebe um rótulo e descreve um certo perfil dinâmico do sistema através de uma equação diferencial. No autômato híbrido da Figura 1, existem dois modos de operação, o modo ligado, cujo rótulo é *on*, e o modo desligado, cujo rótulo é *off*. No modo *on* a temperatura cresce à taxa de $-x+5$ graus por minuto. Quando o sistema está no modo *off*, a temperatura cai à taxa de x graus por minuto. A cada modo de operação está também associada uma condição inicial. A condição inicial no modo *on* é $x = 2$ graus. No modo *off*, a condição inicial é *false* e não está ilustrada na Figura 1.

Para forçar uma mudança de perfil dinâmico, atribui-se à cada modo de operação uma condição invariante. O sistema só pode se manter num certo modo de operação enquanto sua condição invariante for satisfeita. Uma mudança de modo de operação pode estar condicionada à verificação de uma propriedade e pode, também, atribuir novos valores às variáveis reais. No exemplo, a condição invariante para ambos os modos de operação é $1 \leq x \leq 3$, ou seja, a mudança de modo deve ocorrer antes da temperatura deixar o intervalo de operação de [1,3] graus. O aquecedor só é desligado quando a temperatura alcança 3 graus, no modo *on*. E o aquecedor só é ligado quando a temperatura cai para 1 grau, no modo *off*. A evolução das variáveis reais é descrita a partir da condição inicial, de equações diferenciais presentes nos modos, além das transições entre modos.

2.2 Definições

Nesta subseção são formalizados os conceitos apresentados na subseção anterior. Um *autômato híbrido* é um sistema $A = (X, V, \mathit{flow}, \mathit{init}, \mathit{inv}, E, \mathit{jump}, \Sigma, \mathit{syn})$, constituído dos seguintes componentes:

Variáveis: Um conjunto finito $X = \{x_1, x_2, \ldots, x_n\}$ de variáveis. O autômato aquecedor da Figura 1 utiliza a variável x para modelar a temperatura. No caso do exemplo, $n = 1$ e $X = \{x\}$.

Modos de Operação: Um conjunto finito V de modos de operação. No exemplo, *on* e *off* são os dois modos de operação existentes, ou seja, $V = \{\mathit{on}, \mathit{off}\}$.

Condições de Atividade Contínua: É dada pelo componente *flow*. Para todo modo $v \in V$, *flow(v)* é um predicado sobre o conjunto de variáveis $X \cup \dot{X}$, onde $\dot{X} = \{\dot{x}_1, \ldots, \dot{x}_n\}$. A variável $\dot{x}_i$, para $1 \leq i \leq n$, é a derivada primeira de x_i com relação ao tempo, ou seja, $\dot{x}_i = dx_i/dt$. Enquanto o modo de operação de A é v, as variáveis em X evoluem de acordo com uma curva diferencial onde os valores das variáveis reais e suas derivadas primeiras satisfazem a condição de atividade contínua *flow(v)*. No exemplo da Figura 1, o modo de operação *on* tem uma condição de atividade

contínua dada pelo predicado $\dot{x} = -x + 5$, e no modo *off* essa condição é $\dot{x} = -x$. Já na Figura 5, a atividade contínua *flow(v)* é dada por $\dot{x} \in [2, 4]$, no modo *on*. Essa atividade contínua, não-determinística, representa o predicado $2 \leq \dot{x} \leq 4$. Isso significa que o valor de $\dot{x}_i$ pode ser fixado, não-deterministicamente, em qualquer valor no intervalo $[2, 4]$.

Condições Invariantes: É dada pelo componente *inv*. Para todo modo $v \in V$, *inv(v)* é um predicado sobre as variáveis em X. Enquanto o modo de operação do autômato é v, as variáveis em X devem satisfazer a condição invariante *inv(v)*. No exemplo da Figura 1, em ambos os modos de operação do aquecedor a condição invariante é $1 \leq x \leq 3$.

Condições Iniciais: É dada pelo componente *init*. Para todo modo $v \in V$, *init(v)* é um predicado sobre as variáveis em X. O autômato A pode começar no modo v quando a condição inicial *init(v)* é verdadeira. Na representação gráfica, as condições iniciais aparecem como rótulos em arcos de entrada sem pontos de origem. As condições falsas não são ilustradas. No modo *on*, no exemplo da Figura 1, a condição inicial é $x = 2$, e no modo *off* a condição inicial é *false*.

Chaves de Controle ou Transições: É formado por um multi-conjunto E de arestas (v, v') onde $v, v' \in V$ são modos de operação. Na Figura 1, existem duas transições, (*on*, *off*) e (*off*, *on*). Ou seja, existem duas arestas, uma de *on* para *off* e outra de *off* para *on*.

Condições de Mudança de Fase: É dada pelo componente *jump*. Para toda transição $e \in E$, *jump(e)* é um predicado sobre as variáveis em $X \cup X'$, onde $X' = \{x'_1, \ldots, x'_n\}$. O símbolo primitivo x_i, para $1 \leq x \leq n$, refere-se ao valor da variável x_i antes da mudança do modo de operação, e o símbolo derivado x'_i refere-se ao valor atribuído à variável x_i após a mudança do modo de operação. Desta forma, a condição *jump* relaciona o valor das variáveis reais antes da mudança do modo de operação com os possíveis valores dessas variáveis após a mudança do modo de operação. Na representação gráfica do autômato, são usados "guardas" para representar as condições de mudança de fase. Por exemplo, o guarda $(x_1 = x_2) \rightarrow (x_1 := 2x_2)$ significa a condição de mudança de fase dada pelo predicado $x_1 = x_2 \wedge x'_1 = 2x_2 \wedge x'_2 = x_2$. No exemplo do aquecedor, a transição (*on*, *off*) tem uma condição de mudança de fase dada pelo predicado $x = 3 \wedge x' = x$, e a transição (*off*, *on*) tem a condição de mudança de fase dada pelo predicado $x = 1 \wedge x' = x$. No caso do exemplo, portanto, o valor das variáveis reais não muda quando há uma transição entre modos.

Eventos ou Mensagens de Sincronização: É um conjunto Σ de eventos, junto com uma função *syn* que associa um evento de Σ a cada transição $e \in E$. No exemplo, tem-se $\Sigma = \{$*turn_on*, *turn_off*$\}$. À transição (*on*, *off*) corresponde o evento *turn_off* e à transição (*off*, *on*) corresponde o evento *turn_on*. Os eventos permitem a sincronização entre autômatos híbridos distribuídos, como será visto ainda nessa seção.

De ora em diante, exceto se houver menção explícita em contrário, um autômato híbrido A denotará sempre um sistema na forma $(X, V, \mathit{flow}, \mathit{init}, \mathit{inv}, E, \mathit{jump}, \Sigma, \mathit{syn})$. Uma referência a um autômato híbrido A_i denotará um sistema semelhante, cujos componentes são referenciados pelo mesmo índice i.

2.3 Configurações e Trajetórias

Uma *configuração* de um autômato híbrido A é um par (v, a) consistindo de um modo de operação $v \in V$ e de um vetor $a = (a_1, \ldots, a_n)$. Esse último contém um valor $a_i \in$ R

para cada variável $x_i \in X$. A configuração (v, a) de A é *admissível* se o predicado *inv(v)* é verdadeiro quando a variável x_i é substituída pelo valor a_i, para $i = 1, \ldots, n$. Observe, no exemplo da Figura 1, que a configuração $(on, 1.5)$ é admissível. Já a configuração $(on, 0.5)$ não é admissível. A configuração (v, a) é *inicial* se o predicado *init(v)* for verdadeiro quando todo x_i for substituído pelo valor a_i. No mesmo exemplo, é fácil ver que a configuração $(on, 2)$ é inicial.

Considere um par (q, q') formado por duas configurações admissíveis, $q = (v, a)$ e $q' = (v', a')$. O par (q, q') é uma *mudança de fase* de A se existe uma transição $e \in E$ com origem no modo v e destino no modo v' e tal que o predicado *jump(e)* seja verdadeiro quando toda variável x_i é substituída pelo valor a_i, e toda variável x_i' é substituída pelo valor a_i'. No exemplo da Figura 1, existem duas mudanças de fase, ((*on*,3),(*off*,3)) e ((*off*,1),(*on*,1)). Por outro lado, o par (q, q') é uma *atividade contínua* de A se $v = v'$, se existe um real não negativo $\delta \in \mathrm{R}$ (a duração da atividade contínua) e se existe também uma função diferenciável $\rho : [0, \delta] \to \mathrm{R}^n$ (a curva da atividade contínua), tais que os três requisitos que se seguem sejam satisfeitos:

1. Pontos inicial e final: $\rho(0) = a$ e $\rho(\delta) = a'$.
2. Condição invariante: Para todo instante de tempo $t \in [0, \delta]$, a configuração $(v, \rho(t))$ é admissível.
3. Condição de atividade contínua: Tome $\dot{\rho} : [0, \delta] \to \mathrm{R}^n$ sendo a derivada primeira de ρ. Para todo instante de tempo $t \in [0, \delta]$, o predicado *flow(v)* é verdadeiro quando cada variável x_i é substituída pela i-ésima coordenada do vetor $\rho(t)$, e toda variável $\dot{x}_i$ é substituída pela i-ésima coordenada do vetor $\dot{\rho}(t)$.

No exemplo da Figura 1, os pares $((off, 3), (off, 2))$ e $((off, 3), (off, 2.5))$ são atividades contínuas. Quando ocorre uma mudança de fase (q, q'), diz-se que v' é um modo de operação sucessor de v. Quando (q, q') é uma atividade contínua, a' é dito um ponto sucessor de a. Note que um ponto é sucessor dele mesmo, visto que sempre existe a atividade contínua com duração 0 (zero).

Uma *trajetória* de um autômato híbrido A é uma seqüência finita $q_0, q_1, \ldots, q_k$, de configurações admissíveis onde q_0 é uma configuração inicial de A, e onde todo par (q_j, q_{j+1}) de configurações consecutivas da seqüência é uma mudança de fase ou uma atividade contínua de A. Uma configuração q_n de A é *alcançável* se q_n é a última configuração de alguma trajetória de A. No exemplo da Figura 1, todas as configurações admissíveis são alcançáveis.

2.4 O Autômato Produto

Um sistema distribuído, normalmente, é formado por vários componentes operando concorrentemente e comunicando-se uns com os outros. O sincronismo do sistema global é capturado de duas maneiras: (i) forçando com que transições ocorram no mesmo evento, e (ii) usando variáveis compartilhadas. Para se impor o sincronismo do tipo (i), calcula-se o autômato produto dos autômatos distribuídos que fazem parte do sistema. Aquelas transições, nos autômatos participantes, às quais está associado um mesmo evento, correspondem a uma única transição, simultânea, no autômato produto.

Em geral, se A_1 e A_2 são autômatos híbridos, então o *autômato produto* de A_1 e A_2 é um sistema

$$A = (X_1 \cup X_2, V_1 \times V_2, \mathit{flow}, \mathit{init}, \mathit{inv}, E, \mathit{jump}, \Sigma_1 \cup \Sigma_2, \mathit{syn})$$

onde
$$\begin{aligned} \mathit{flow}((v_1, v_2)) &= \mathit{flow}(v_1) \wedge \mathit{flow}(v_2), \\ \mathit{inv}((v_1, v_2)) &= \mathit{inv}(v_1) \wedge \mathit{inv}(v_2), \\ \mathit{init}((v_1, v_2)) &= \mathit{init}(v_1) \wedge \mathit{init}(v_2). \end{aligned}$$

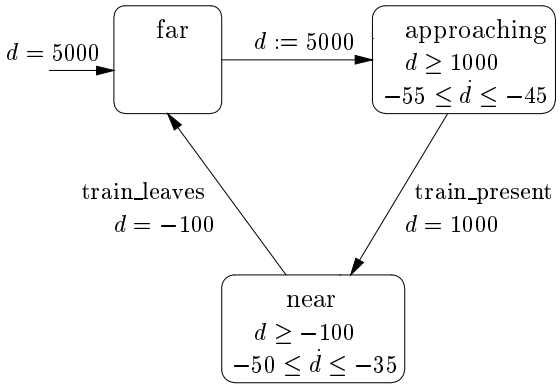

Figura 2: Autômato do trem na ferrovia

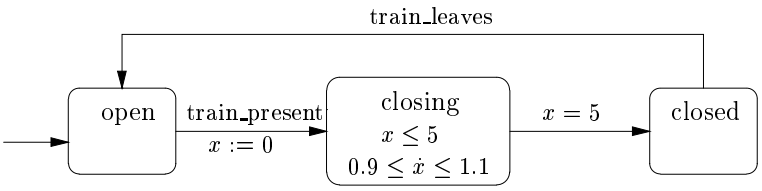

Figura 3: Autômato do controle da cancela

E, ainda, $e = ((v_1, v_1'), (v_2, v_2')) \in E$ se e somente se uma das três condições abaixo é verificada:

1. $v_1 = v_1'$, $e_2 = (v_2, v_2') \in E_2$, $syn_2(e_2) \notin \Sigma_1$.
   Daí, $jump(e) = jump_2(e_2)$ e $syn(e) = syn_2(e_2)$.

2. $v_2 = v_2'$, $e_1 = (v_1, v_1') \in E_1$ e $syn_1(e_1) \notin \Sigma_2$.
   Daí, $jump(e) = jump_1(e_1)$ e $syn(e) = syn_1(e_1)$.

3. $e_1 = (v_1, v_1') \in E_1$, $e_2 = (v_2, v_2') \in E_2$, $syn_1(e_1) = syn_2(e_2)$.
   Daí, $jump(e) = jump_1(e_1) \wedge jump_2(e_2)$ e $syn(e) = syn_1(e_1)$.

Portanto, as transições e_1 e e_2 são executadas simultaneamente no autômato produto quando $syn_1(e_1) = syn_2(e_2)$, isto é, quando a ambas está associado um mesmo evento. Por outro lado, essas transições são intercaladas no autômato produto se o evento $syn_1(e_1)$ não ocorre no conjunto de eventos de A_2, e nem $syn_2(e_2)$ ocorre no conjunto de eventos de A_1.

Como exemplo, imagine uma ferrovia com uma passagem de nível onde foi instalada uma cancela. Um trem trafega pela via e se aproxima da passagem de nível. Deseja-se verificar se o trem passa em segurança pelo trecho da cancela. Ou seja, a cancela deve estar fechada quando o trem atravessar esse trecho da via. Inicialmente, o trem se aproxima da passagem de nível com uma velocidade no intervalo de $[-55, -45]$ metros por segundo. Um sensor está localizado a uma distância de 1000 metros da cancela. Em um certo momento, o trem passa pelo sensor que, então, comanda o início de fechamento da cancela. A cancela começa a se fechar, concluindo o movimento em 5 unidades de tempo. No entanto, o

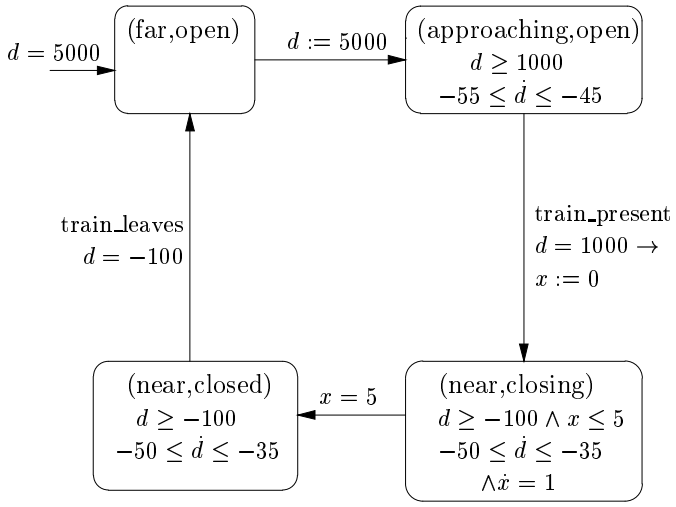

Figura 4: Autômato produto do trem com o controlador

mecanismo que opera a cancela introduz uma imprecisão de até 10%, para mais ou para menos, no tempo de movimentação da cancela. Assim, a cancela pode completar a operação de fechamento num ponto qualquer entre 4.5 e 5.5 unidades de tempo após iniciar seu movimento. Quanto ao trem, após a passagem pelo sensor, este diminui sua velocidade para um valor no intervalo $[-50, -35]$ metros por segundo, e mantém sua velocidade neste intervalo até cruzar e ultrapassar a região da cancela. À uma distância de 100 metros, do outro lado da cancela, o trem passa por um segundo sensor. Este segundo sensor comanda o início do movimento para abrir a cancela.

A Figura 2 [HHWT95b] ilustra o autômato do trem e a Figura 3 mostra o autômato que modela o controle de movimentação da cancela. O autômato produto pode ser visualizado na Figura 4. Observe como eventos com o mesmo rótulo no autômato do trem e no autômato do controlador forçam o sincronismo no autômato produto. O evento *train_present* sinaliza a passagem do trem pelo primeiro sensor, e o evento *train_leaves* sincroniza os autômatos quando o trem sai da região de perigo. Isso garante que o autômato da cancela e o autômato do trem sincronizam suas transições quando o trem passa pelos sensores. No autômato produto, na Figura 4, foram descartados os modos que não são alcançáveis. Por exemplo, o modo $(far, closed)$ foi descartado. A ferramenta HyTech é capaz de construir esse autômato produto automaticamente.

2.5 Computação de Configurações Alcançáveis

O conjunto de configurações para as quais um predicado φ é satisfeito é denominado φ-região, ou região φ. Veja que a região *inv* é formada por todas as configurações admissíveis. Da mesma forma, a região *init* agrupa todas as configurações iniciais.

Dada uma região φ, $Post(\varphi)$ designa a *região alcançável* a partir da região φ. Em geral, $Post(\varphi)$ reúne todas aquelas configurações q' para as quais existe uma configuração q em φ tal que a configuração q' é alcançável a partir de q através de uma trajetória de A constituída de uma atividade contínua, seguida ou não de uma mudança de fase. Fazendo $\varphi_0 = \varphi$, a iteração desse processo, computará as regiões $\varphi_{k+1} = Post(\varphi_k)$, para $k = 1, 2, \ldots$. Se atingirmos uma região φ_k tal que $\varphi_k = \varphi_{k+1}$, então φ_k caracteriza todas as configurações alcançáveis por trajetórias de A partindo de φ_0.

Para se aplicar esse processo é preciso, primeiramente, calcular a região $Post(\varphi)$ de maneira eficiente, dada a região φ. Também é necessário que a computação convirja após um número finito de aplicações de *Post*. É possível satisfazer a primeira restrição considerando-se uma subclasse restrita dos autômatos híbridos, os autômatos híbridos lineares [HHWT97, Ho95, Hen96], apresentados na Subseção 2.6. A segunda restrição pode ser satisfeita tomando-se uma subclasse dos autômatos híbridos lineares, os *timed automata*, onde as variáveis são todas relógios. A ferramenta HyTech trabalha com autômatos híbridos lineares, englobando a primeira restrição. A segunda restrição não é contemplada pela ferramenta. Isto é, pode não haver convergência no processo iterativo.

Outra forma de se calcular regiões alcançáveis é através do operador Pre. Dada uma região φ, $Pre(\varphi)$ é verdadeira para a configuração q se existe uma configuração q' em φ tal que q' é alcançável a partir de q através de uma trajetória de A formada por uma atividade contínua, seguida ou não de uma mudança de fase. Iniciando com uma região ψ_0 e iterando-se esse processo, pode-se obter as regiões ψ_{k+1}, onde $\psi_{k+1} = Pre(\psi_k)$. Se o processo atingir uma região ψ_k tal que $\psi_{k+1} = \psi_k$, então ψ_k representa a região a partir da qual pode-se alcançar a região original, ψ_0.

2.6 Autômatos Híbridos Lineares

A não-linearidade das características dinâmicas de um sistema real pode ser um fator complicante para a análise do modelo. Nessas situações, são usados métodos de linearização do modelo que transformam o modelo original do sistema num autômato híbrido linear [Hen96, HHWT97], mais conveniente para análise, e usualmente não-determinístico. A necessidade de linearização também é imposta pelo fato de que as ferramentas (semi) automáticas disponíveis para a análise das especificações lidam, via de regra, apenas com autômatos híbridos lineares [HH94].

Um *predicado atômico linear* é uma inequação entre constantes racionais e combinações lineares de variáveis com coeficientes racionais, como por exemplo $2x + 4y - 7z/2 \leq -10$. Um *predicado linear convexo* é uma conjunção finita de predicados atômicos lineares, e um *predicado linear* é uma disjunção finita de predicados lineares convexos. Um autômato híbrido A é um autômato híbrido linear, se satisfaz os seguintes requisitos:

1. **Linearidade**: Para todo modo de operação $v \in V$, a condição de atividade contínua *flow(v)*, a condição invariante *inv(v)* e a condição inicial *init(v)*, são predicados lineares convexos. E também, para toda transição $e \in E$, a condição de mudança de fase *jump(e)* é um predicado linear convexo;

2. **Atividade contínua independente**: Para todo modo de operação $v \in V$, a condição de atividade contínua *flow(v)* é um predicado sobre as variáveis que estão no conjunto $\dot{X}$ somente, não contendo variáveis do conjunto X.

O segundo requisito garante que a razão de variação das variáveis contínuas não depende do valor dessas variáveis, mas depende somente do modo de operação. É um tanto limitante para a modelagem proibir atividades contínuas, tais como $\dot{x} = x$. Por outro lado, é permitido o uso de variáveis tais como relógios, cronômetros e relógios com derivas. É possível, por exemplo, especificar $\dot{x} \in [1 - \epsilon, 1 + \epsilon]$, para alguma constante ϵ. Nesse caso $\dot{x} = 1 - \epsilon$ representa um relógio com atraso ϵ, e $\dot{x} = 1 + \epsilon$ representa um relógio com avanço ϵ. Nesses dois casos a condição de linearidade é atendida.

Como um exemplo, observe o autômato ilustrado na Figura 5 ignorando, por ora, a presença das variáveis y e z. Este autômato resulta de uma linearização simples do autômato original, mostrado na Figura 1. No autômato da Figura 1, no modo *on* o valor de x está restrito ao intervalo $[1, 3]$. Como, nesse modo, o valor de $\dot{x}$ é $\dot{x} = -x+5$, conclui-se que, nesse mesmo modo, o valor de $\dot{x}$ está restrito ao intervalo $[2, 4]$, que é a especificação atribuída a variável $\dot{x}$ no modo *on* no autômato ilustrado na Figura 5. De forma similar, no modo

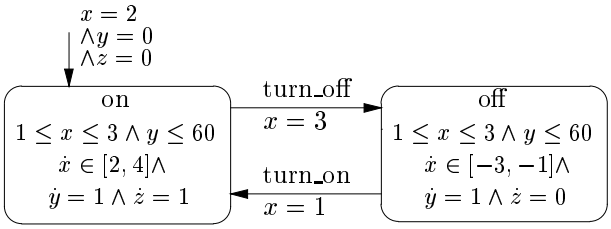

Figura 5: Autômato Aquecedor com verificação de segurança

off a função $-x$ é relaxada para o intervalo $[-3, -1]$. Neste processo de linearização, cada atividade contínua não-linear é "relaxada" para um intervalo de valores que compreende a solução da equação diferencial original. Observe como as trajetórias no autômato linear cobrem todas as trajetórias possíveis no autômato original.

Pode-se mostrar o seguinte teorema: se A é um autômato híbrido linear, e se φ é uma região de configurações para A, então o cálculo de $Post(\varphi)$ converge e o resultado será uma nova região de configurações para A [AHH93]. Esse teorema permite automatizar a análise dos autômatos híbridos lineares.

2.7 Requisitos de Segurança

Um requisito de segurança é um conjunto de restrições e condições impostas às configurações do sistema. Uma configuração é segura se satisfaz à todas as condições de segurança que estão associadas a si. Caso contrário a configuração é dita insegura. Os requisitos de segurança são satisfeitos pelo sistema como um todo se e somente se todas as configurações alcançáveis são seguras. Ou seja, dentre todas as configurações alcançáveis não devem existir configurações inseguras. Geralmente, um requisito de segurança é especificado pela descrição de combinações de valores desejáveis e combinações de valores indesejáveis para as variáveis do sistema.

Para um autômato híbrido, os requisitos de segurança são especificados usando-se asserções sobre os modos de operação. Uma asserção φ sobre modos de operação de um autômato híbrido A é uma função que atribui a todo modo $v \in V$ um predicado $\varphi(v)$ sobre as variáveis em X. A asserção sobre um modo de operação é verdadeira ou falsa para uma configuração (v, a) de A, se o predicado $\varphi(v)$ é verdadeiro ou falso quando toda variável x_i é substituída pelo valor a_i. Um requisito de segurança também pode ser especificado criando-se um predicado *unsafe* sobre as configurações de A. Esse predicado descreve as condições que violam a segurança do sistema. Assim, o sistema modelado é seguro se a região *unsafe* de configurações de A não incluir nenhuma configuração alcançável de A. Para verificar se um autômato híbrido A satisfaz um requisito de segurança especificado por uma asserção *unsafe*, especificada sobre suas configurações, calcula-se uma outra asserção sobre configurações, *reach*, a qual caracteriza todas as configurações alcançáveis de A. A região *reach* é computada calculando-se a região $Post(\varphi_0)$, onde $\varphi_0 = init$. Logo, *reach* é a região formada por todas as configurações de A alcançáveis a partir da região *init*. Em seguida, verifica-se se as regiões *reach* e *unsafe* têm um ponto em comum, ou seja, se a intersecção entre essas regiões não é vazia. Em caso afirmativo, o requisito de segurança é violado. Esse sendo o caso, a ferramenta HyTech é capaz de indicar uma trajetória de A que leva à região *unsafe*.

Para o autômato exemplo da Figura 1, um requisito de segurança poderia ser o seguinte: o aquecedor não deve ficar ligado mais de 2/3 do tempo nos primeiros 60 minutos de operação. Pode-se usar uma variável y para modelar o tempo total de operação do aquecedor e uma variável z para representar o tempo total gasto apenas enquanto o aquecedor está

ligado. Observe que essas variáveis auxiliares não alteram o comportamento do autômato. No exemplo é possível especificar a condição insegura *unsafe*, através do predicado $y = 60 \wedge z > 2y/3$, válido para os dois modos de operação. Na Figura 5 é apresentado o autômato modificado da Figura 1 com as variáveis auxiliares, y e z, adicionadas. A variável y é um *relógio* ($\dot{y} = 1$ para todos os modos de operação) e z é um *cronômetro* ($\dot{z} = 0$ ou $\dot{z} = 1$ em todo modo de operação). A atividade contínua no modo de operação *on* é descrita por $\dot{x} \in [2, 4]$ e no modo de operação *off* é descrita por $\dot{x} \in [-3, -1]$. Essa modificação foi introduzida para "linearizar" o modelo (veja a Subseção 2.6) e torná-lo capaz de ser analisado pela ferramenta HyTech (veja a Subseção 2.10).

2.8 Verificação de Propriedades

De um modo geral, as propriedades a serem verificadas são expressas na forma de sentenças lógicas escritas numa linguagem formal apropriada, envolvendo componentes da especificação. Então, a verificação da propriedade eqüivale à demonstração da veracidade da sentença descritiva.

No autômato da Figura 5, o objetivo é verificar que o aquecedor não permanece ligado mais que 2/3 do tempo na primeira hora de operação. Acrescentando-se a invariante $y \leq 60$ garante-se a computação das configurações alcançáveis apenas para as trajetórias tomadas nos primeiros 60 minutos. A região de configurações especificada como *unsafe*, para ambos os modos de operação, é $y = 60 \wedge z \geq 2/3$. A computação das configurações alcançáveis começa na região $\varphi_0 = init = \{(on, x = 2 \wedge y = 0 \wedge z = 0), (off, false)\}$. A configuração inicial está no modo de operação *on* e x vale inicialmente 2 graus. Em seguida, calcula-se a região $\varphi_1 = Post(\varphi_0)$ em dois passos. Num primeiro passo calcula-se a região alcançável através de atividades contínuas a partir da região φ_0. Num segundo passo calcula-se a região alcançável através de uma mudança de fase a partir dessa região intermediária. O predicado obtido para φ_1 é $x \leq 3 \wedge 2z+2 \leq x \leq 4z+2 \wedge y = z$. A ferramenta HyTech realiza a computação automaticamente até que as regiões não cresçam mais. Com 73 iterações a região *reach* é calculada e determina-se que o predicado $(reach(v) \wedge unsafe(v))$ é falso para os dois modos de operação. Ou seja, não existe nenhuma configuração para as quais *reach* e *unsafe* são verdadeiras ao mesmo tempo. Isso satisfaz o requisito de segurança do aquecedor. Pelo fato do autômato híbrido linear cobrir todas as trajetórias do autômato híbrido não-linear original, é suficiente provar que os requisitos de segurança são satisfeitos pelo autômato linear.

2.9 Síntese de Parâmetros e Análise Paramétrica

Parâmetros são constantes simbólicas que assumem valores fixos. Uma análise paramétrica do modelo é produzida quando se determinam intervalos de valores para as constantes simbólicas do modelo e de forma que a segurança do sistema seja mantida. Desse modo, uma análise paramétrica determina que restrições sobre os valores dos parâmetros são necessárias e suficientes para que nenhum dos requisitos de segurança do sistema seja violado. Com isso, podem ser sintetizados valores máximos e valores mínimos para os parâmetros.

Em um autômato híbrido, um parâmetro α pode ser representado por uma variável que nunca muda de valor, isto é, em todo modo de operação inclui-se a condição $\dot{\alpha} = 0$. Além disso, em todas as transições especifica-se a condição $\alpha' = \alpha$. Logo, o parâmetro α mantém o mesmo valor em todas as configurações ao longo de uma trajetória do autômato. Um valor $a \in \mathrm{R}$ é dito seguro para um parâmetro α se, quando a restrição $\alpha = a$ for adicionada à todas as demais condições do modelo, nenhuma configuração insegura se torna alcançável. O intervalo de valores para α é obtido de tal forma que o predicado $(reach(v) \wedge unsafe(v))$ nunca seja verdadeiro, para todo modo de operação v.

Na Figura 6 o autômato do trem foi parametrizado com a variável α no lugar da constante 1000, que indicava a distância da cancela onde se encontrava o primeiro sensor. Portanto,

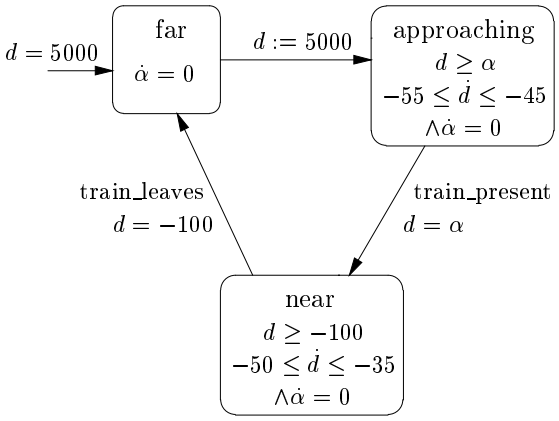

Figura 6: Autômato do trem parametrizado

o controlador deve detectar a presença do trem quando esse se encontra a α metros da travessia da cancela. Usando essa nova descrição do modelo, a ferramenta HyTech informa que se deve impor $\alpha > 287\frac{7}{9}$ metros para que o requisito de segurança seja satisfeito. O primeiro sensor, portanto, deve estar localizado a mais de $287\frac{7}{9}$ metros da cancela para que o sistema opere de forma segura.

2.10 O Analisador HyTech

O software HyTech [HHWT95a, HHWT97, HHWT95b, HH94] é uma ferramenta automática que pode ser usada na análise, verificação de propriedades e síntese de parâmetros para sistemas híbridos realistas. Normalmente, a especificação desses sistemas resulta num autômato produto que apresenta uma complexidade de tal ordem a tornar imprescindível a cooperação de ferramentas automáticas para análise da especificação produzida. O HyTech é um verificador de autômatos híbridos lineares, e que pode ser usado para analisar um modelo híbrido após sua linearização.

O arquivo de entrada da ferramenta possui duas partes. A primeira parte descreve os autômatos híbridos lineares participantes do modelo. O autômato produto é calculado automaticamente. A análise prossegue a partir desse autômato produto. A segunda parte possui uma seqüência de comandos que direcionam a computação por parte da ferramenta. A linguagem de especificação desses comandos é muito semelhante a uma linguagem de programação, com tipos de dados primitivos, e com operações primitivas tais como *Post*, além de operadores binários e quantificação existencial. Para maior facilidade, existem macros já construídas que permitem a análise de alcançabilidade, a análise paramétrica e a geração de trajetórias de erro.

Veja a seguir a seqüência de comandos utilizada no sistema do aquecedor, onde α é um parâmetro que representa o tempo em que o aquecedor se mantém ligado na primeira hora de operação:

```
unsafe := y=60 & z >= alpha;                                    (1)
reachable :=                                                    (2)
        reach forward from init_states endreach;                (3)
bad_alpha_values := omit all locations                          (4)
        hide non_parameters in reachable & unsafe endhide;      (5)
```

```
good_alpha_values := ~bad_alpha_values;                    (6)
    prints ‘‘Bad values:’’;                                (7)
        print omit all locations bad_alpha_values;         (8)
    prints ‘‘Good values:’’;                               (9)
        print omit all locations good_alpha_values;        (10)
```

Na linha (1) é especificada a configuração insegura. Nas linhas (2) e (3) são computadas as configurações alcançáveis a partir das configurações iniciais, iterando-se o operador *Post*. As linhas (4) e (5) especificam o predicado que caracteriza as configurações alcançáveis inseguras. O símbolo $\sim$, na linha (6), significa negação. As linhas de (7) a (10) produzem a saída. O resultado é:

```
Bad values for spec.:
      alpha <= 36
Good values for spec.:
      alpha > 36
```

Portanto, conclui-se que $\alpha > 36$ é suficiente para satisfazer os requisitos de segurança do aquecedor. Ou seja, o aquecedor nunca fica ligado mais de 36 minutos durante a primeira hora de operação.

Uma outra vantagem da ferramenta é a geração de trajetórias de erro quando o sistema falha para um determinado requisito de segurança. Geralmente, isto é usado para se depurar o modelo e encontrar os erros de especificação e mal funcionamento do modelo, dados certos parâmetros.

Inicialmente, o HyTech foi usado para analisar sistemas de controle, como é o caso do exemplo do aquecedor e do exemplo do trem. Hoje, a ferramenta também está sendo usada, inclusive, na modelagem e análise de circuitos temporizados. Inúmeros outros exemplos podem ser encontrados em [HWT95, HH94, HHWT95b, HWT96, Ho95, AHH93].

3 Descrição do Segmento de Malha Metroviária

As malhas metroviárias, atualmente, apresentam uma grande complexidade devido a crescente demanda por transporte coletivo. A automação das malhas ocorreu em função da necessidade de se implantar melhorias e aumentar sua capacidade de transporte com níveis compatíveis de disponibilidade. Um exemplo do avanço tecnológico implantado no sistema é o uso de circuitos microprocessados, os quais são vantajosos no que diz respeito a aspectos de controle e supervisão. Esta automação, por sua vez, demanda uma verificação cuidadosa dos aspectos de segurança do sistema. A ênfase desse trabalho está voltada para os aspectos de segurança de uma malha metroviária. O objetivo é, de um lado, modelar o sistema, validando seu funcionamento operacional e verificando diferentes propriedades do modelo. Por outro lado, deseja-se também sintetizar valores mais apropriados para alguns parâmetros críticos do sistema, de forma a obter-se uma maior eficiência e uma melhor eficácia na sua operação.

Para descrever uma malha metroviária é preciso, antes de tudo, ressaltar dois pontos importantes. Em primeiro lugar, o sistema modelado apresenta o requisito básico de um sistema híbrido, que é a mesclagem de sistemas analógicos com sistemas digitais. A parte analógica é composta pela operação mecânica do trem e seus componentes, tais como circuitos de via e máquinas de chave. Já a parte digital é formada pelo conjunto de sistemas discretos que coordenam e regulam o funcionamento da parte analógica, como a comunicação entre sensores, via software. Em segundo lugar, o sistema modelado é relativamente complexo, mas ainda com características que permitem uma modelagem usando autômatos híbridos, quando dividido em estudos de caso separados.

O funcionamento e a operação da malha metroviária, de um modo geral, estão apoiados em vários componentes e em diferentes equipamentos. Um dos principais aspectos da modelagem é capturar como e quando acontece a cooperação entre as partes que compõem o sistema. Primeiramente, serão descritos os equipamentos de via, os quais estão distribuídos ao longo da malha metroviária. Não serão apresentados todos os equipamentos que fazem parte do sistema. Serão descritos somente os equipamentos relevantes para a modelagem e a análise desenvolvidas na próxima seção. Em seguida são apresentados algumas das funções envolvidas na automação da malha metroviária.

3.1 Conceitos Gerais de uma Malha Metroviária

Alguns dos componentes e conceitos relacionados à operação de uma malha metroviária são sucintamente descritos a seguir:

- Circuitos de via: Circuitos de via são trechos da via onde é possível a detecção da presença de trens, fornecendo indicações de ocupação e de desocupação desses trechos da via. Essas informações são processadas pelo sistema de sinalização. Outra função do circuito de via, ou simplesmente *cv*, é transmitir códigos de velocidade ao trem, de forma que este trafegue a uma velocidade segura, de acordo com sua posição na via, e de acordo com sua posição relativa a outros trens que estão em movimento na via. Os comandos de restrição de velocidade servem para impor certos limites à velocidade dos trens, em determinados pontos da via. Isso ocorre porque em certas condições operacionais é necessário que os trens trafeguem a velocidades mais baixas do que a velocidade padrão do perfil programado para a região. Esses comandos impõem ou removem condições de restrição de velocidade num determinado trecho da via, como por exemplo, entre duas estações ou entre uma estação e o final de via. Os circuitos de via são delimitados por um mecanismo que une dois trilhos eletricamente. Um trem, quando ocupa um circuito de via, impede a recepção de um sinal transmitido, fazendo com que seja detectada sua presença naquele trecho de via.

- Máquinas de chave: As máquinas de chave dos Aparelhos de Mudança de Via (AMVs) são controladas por intertravamento. Essas máquinas são operadas eletricamente (ou manualmente em condições de emergência), podendo assumir dois estados: (i) normal; e (ii) reverso. As máquinas de chave também podem, em cada posição, assumir os estados de travado ou destravado eletricamente. Uma máquina de chave está posicionada corretamente quando existir uma correspondência lógica entre o comando de posicionamento e a informação física de posição da máquina de chave, dada pela ponta da agulha. Ou seja, a informação física deve estar coerente com a informação presente no sistema automático de controle. A máquina de chave está travada quando da não alimentação do respectivo motor da máquina de chave (travamento elétrico).

- Zonas Terminais: são trechos de via, compostos por um ou mais circuitos de via, usados para manobras de retorno dos trens.

- Regiões de Intertravamento: são as regiões onde se encontram os AMVs. É nessas regiões que se efetuam as mudanças de rota dos trens. São chamadas também de "regiões de AMVs".

- Intertravamento Microprocessado: faz parte do sistema automático que controla a operação do sistema global, e exerce funções de controle na movimentação dos trens. Este equipamento é encarregado de controlar a movimentação dos trens de forma automática e segura, comunicando-se com as máquinas de chave e com os circuitos de via.

As funções básicas do sistema de controle global são: (i) comandar e receber indicações de máquinas de chave; (ii) gerar e verificar códigos de velocidade em circuitos de via; (iii) determinar as indicações de estado (ocupado/desocupado) dos circuitos de via; e (iv) comunicar-se com o intertravamento para verificação de sentido de tráfego, ocupação/desocupação dos circuitos de via adjacentes, e a posição das máquinas de chave.

O sistema global comunica-se com as interfaces dos circuitos de via através de Caixas à Margem da Via (CMV). O código de velocidade determina a velocidade máxima que um trem pode atingir num trecho. Existem diferentes códigos de velocidade que podem ser distribuídos de acordo com o segmento de via focalizado. Esses códigos podem variar entre 0 e 100 km/h. A transmissão dos códigos de velocidade num circuito de via é realizada através de um par de CMVs, uma CMV transmissora e outra CMV receptora de códigos de velocidade. Baseado nas informações recebidas do controle de movimentação dos trens, determina-se que velocidade deve ser imposta a um circuito de via em particular. O código é transmitido ao CMV daquele circuito que, por sua vez, lhe impõe o código de velocidade adequado.

As chamadas funções de supervisão servem para alimentar o sistema com informações através da leitura dos estados dos equipamentos da via. Essas informações são utilizadas pelas funções binárias que implementam o intertravamento. As informações trocadas têm como função o estabelecimento dos perfis de velocidade dos trens, curvas de frenagem quando um trem se aproxima de outro trem ou de um circuito contendo um AMV, entre outras funções. Os estados dos equipamentos de via, obtidos por leitura, correspondem ao posicionamento físico dos AMVs. Os equipamentos que estão na via são comandados através das funções de atuação.

A função de intertravamento é um conjunto de regras que permitem ao sistema realizar suas principais atividades, como movimentação e proteção das máquinas de chave e aplicação de códigos de velocidade. O modo de controle do intertravamento indica o modo de operação que deve ser utilizado numa certa região de intertravamento. Existem três modos de controle: o modo central, o modo local e o modo automático. Este último é o modo que interessa para fins de modelagem. No modo automático os comandos sobre os equipamentos de via são gerados automaticamente pelo sistema. As funções de comando sobre as máquinas de chave permitem uma atuação direta no posicionamento das máquinas nas regiões de AMVs. As máquinas de chave são movimentadas para as posições de normal ou reverso, desde que as condições de intertravamento o permitam. Os AMVs de uma determinada região de intertravamento podem assumir dois modos de controle mutuamente exclusivos, o manual e o automático. O modo de controle manual só é usado em casos de emergência e será desconsiderado na modelagem. O modo de controle automático indica que o controle do AMV está sendo efetuado pelo sistema. Os sinais enviados aos equipamentos de via são responsáveis pelo acionamento das máquinas de chave (normal/reversa) e pelo travamento das mesmas.

Entre as funções de controle da movimentação dos trens estão: (i) seleção do modo de controle do intertravamento; (ii) comando sobre máquinas de chave; (iii) comando de restrição de velocidade; e (iv) inversão no sentido do tráfego. O comando de inversão no sentido do tráfego permite a inversão no sentido normalmente utilizado entre regiões de AMVs. A inversão é utilizada quando um ou mais trens ficam bloqueados num trecho de via ou quando um trem chega a uma zona terminal.

3.2 Os Requisitos de Segurança de uma Malha Metroviária

Nesta subseção são descritos os requisitos de segurança relevantes para a modelagem, e que envolvem a parte de automação do sistema. Do ponto de vista de segurança, os requisitos básicos são: evitar colisões e descarrilhamentos de trens. Estas situações podem ocorrer, por exemplo, por uma movimentação errada de um AMV. Por isso, o sistema deve executar de maneira segura as funções de controle de movimentação dos trens, através das funções

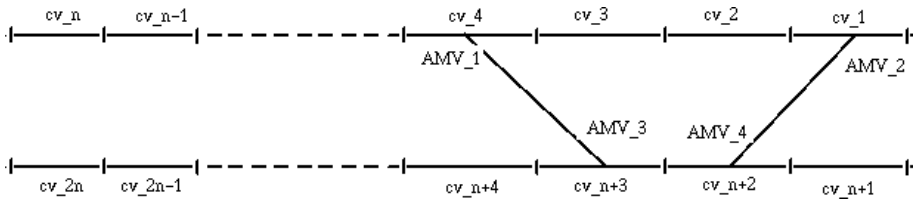

Figura 7: Região de vias da malha metroviária

de proteção dos AMVs, controle de tráfego e controle de seleção dos códigos de velocidade. São apresentados a seguir os requisitos de segurança que devem ser aplicados ao sistema:

1. Só pode haver a ocupação de um circuito de via em condições normais do sistema se não houver tráfego estabelecido no sentido oposto;

2. Só pode haver a ocupação de um circuito de via em condições normais do sistema se o circuito de via já não estiver ocupado por nenhum outro trem;

3. Só pode haver a indicação de desocupação de um circuito de via em condições normais do sistema se o circuito de via já estiver desocupado pelo próprio trem;

4. Só pode haver destravamento de uma máquina de chave se o circuito de via a que ela pertence estiver desocupado;

5. O perfil seguro de velocidade em circuitos de via vizinhos a uma ocupação deve ser respeitado pelo intertravamento;

6. Na ocupação seqüencial dos circuitos de via, os códigos de velocidade devem obedecer ao sentido do tráfego estabelecido naquela região e aos perfis de velocidade impostos pelas condições da via;

7. Só pode haver inversão de sentido se o trecho para onde o trem está se dirigindo não estiver sendo utilizado como saída para outro trem e se o circuito de via a ser invadido não estiver ocupado.

3.3 Funcionamento do Modelo Adotado de uma Malha Metroviária

Para oferecer garantias de segurança, sistemas de controle de tráfego metroviário exigem, em função do estado da tecnologia atualmente utilizada, o posicionamento e travamento de um AMV, antes que o trem invada o *cv* onde este AMV está localizado. *O modelo tratado neste trabalho desvia dessa concepção, conduzindo um exercício de modelagem da dinâmica do sistema onde se permite que o trem adentre ao* cv *antes que o AMV esteja posicionado e travado. Experimentos desse tipo revelam até que ponto o requisito de segurança básico pode ser relaxado e de forma que o sistema ainda ofereça garantias de uma operação segura.*

No sistema global abordado podem existir vários trens trafegando pela malha metroviária. Um mapa de um conjunto de circuitos de via formando uma região da malha metroviária pode ser visualizado na Figura 7. Os circuitos de via possuem comprimento padrão de cerca de 200 metros. Cada circuito pode ser ocupado para dar passagem a um determinado trem. A ocupação do circuito ocorre quando este está livre, respeitando o perfil que lhe foi imposto.

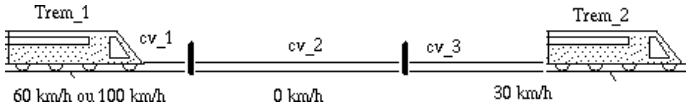

Figura 8: Tomada de perfis evitando colisões

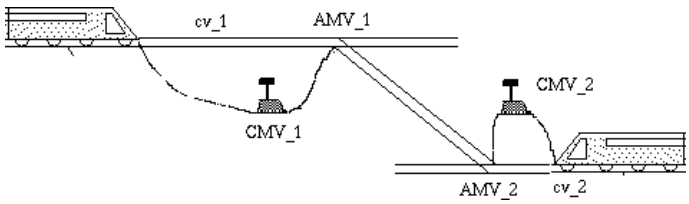

Figura 9: Mudança de vias com trens em sentidos opostos

A ocupação do circuito é provocada através de um pedido do trem em questão, o qual precisa passar pelo circuito para chegar ao seu destino. A ocupação é concedida impondo perfis de velocidade adequados. Perfis de velocidade são enviados aos trens através de transmissores adaptados aos *cv*s, permitindo que os trens trafeguem de forma segura. Podem existir diferentes perfis de velocidade atribuídos aos circuitos de via. Os perfis básicos utilizados no sistema são: 0 km/h, 30 km/h, 60 km/h, 80km/h e 100 km/h. A desocupação se dá simplesmente quando o trem deixa o circuito de via no qual estava presente. Quando o trem deixa um circuito, este fica livre para uma nova ocupação pelo mesmo trem ou para a ocupação por outros trens que realizam aquela mesma rota. Na verdade, os perfis de velocidade são impostos de acordo com a posição dos trens na via. Existe uma seqüência de perfis de velocidade atribuídos ao longo da via, antecipadamente, tido como perfis padrão. Isso quer dizer que um circuito de via tem um perfil pré-determinado. Por exemplo, um circuito que contém um aparelho de mudança de via possui um perfil padrão de 30 km/h. Circuitos de via distantes das regiões de intertravamento podem ter perfis de até 100 km/h. O código de 0 km/h é aplicado a um circuito que, na realidade, não pode ser ocupado. Este circuito pode estar desocupado, mas o circuito seguinte a ele pode estar ocupado. O perfil zero evita que o trem invada o próximo circuito, como no caso de frenagens, por exemplo.

Quando se tem a ocupação de um circuito de via os perfis dos circuitos adjacentes são atribuídos conforme sua distância do circuito base. Esse esquema, chamado de *sombra*, garante a segurança da malha, com a imposição de velocidades altas para situações onde não é possível a colisão e velocidades mais baixas para evitar colisões. Veja a ilustração na Figura 8. Como os trens estão muito próximos, o perfil atribuído deve ser seguro o bastante para que, no pior caso, um trem possa parar antes de atingir o trem que está à frente. A uma velocidade de 60 km/h o Trem_1 consegue parar antes de atingir o Trem_2, que vai a 30 km/h, desde que seja aplicado um código de velocidade de 0 km/h no circuito que vai ser invadido até o Trem_1 parar. Se o Trem_1 estiver a 100 km/h, ao invadir o cv_2 não conseguirá parar antes de atingir o cv_3, podendo colidir com o Trem_2. Outro tipo de situação que pode ocorrer é ilustrado na Figura 9. Quando dois trens trafegam em sentidos inversos, ou seja, quando um trem está percorrendo uma determinada rota e outro trem está efetuando uma manobra na mesma linha em sentido contrário, existe o perigo, na região de AMV, dos trens trocarem de rota e se chocarem, ocasionando um acidente.

Nas regiões dos AMVs, além do pedido de ocupação do circuito é necessário que uma requisição de movimentação dos AMVs seja atendida. Essa movimentação é efetuada conforme a necessidade da ocasião. Observe, na Figura 10, que o trem se aproxima e ocupa o

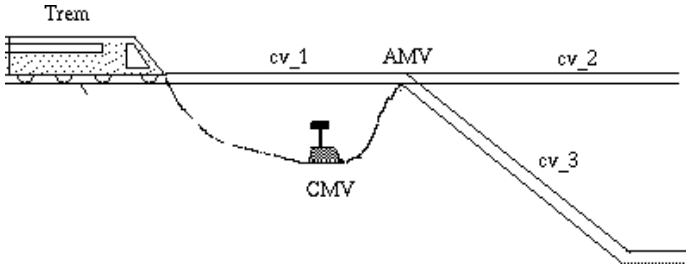

Figura 10: Mudança de vias na região de AMV

circuito de via cv_1. Ao ocupar o circuito cv_1 o trem já deve fazer o pedido de requisição de movimentação do AMV. Se for seguir em frente para o circuito cv_2, o trem deve pedir o posicionamento para deixar o AMV no estado *normal*. Caso queira mudar de direção indo para o circuito cv_3, o pedido de posicionamento deve colocar o AMV no estado *reverso*. Quando um trem faz o pedido de movimentação para um AMV, este deve se destravar e começar a movimentar-se para a posição correta. Essa movimentação leva algum tempo, até que a ponta da agulha esteja no lugar correto. O tempo de movimentação do AMV deve ser adequado para que no momento do trem atravessá-lo, o AMV já esteja travado e a travessia ocorra com segurança.

Um outro ponto interessante do funcionamento do sistema é a parte que compreende o fim de via. Essa parte da malha metroviária é ilustrada na Figura 7, onde os circuitos de via mais à direita são uma espécie de pátio de manobras para o retorno dos trens. É tomado como padrão o tráfego na via superior no sentido da esquerda para a direita, e na parte inferior no sentido inverso. O trem que ocupa o circuito de via cv_4 pode optar por seguir tanto para o circuito cv_3, como pode optar por trocar de via, indo para o circuito cv_{n+3}. Escolhendo seguir na via superior, o trem deve chegar ao cv_1 e em seguida retornar, no sentido inverso, pedindo o posicionamento do AMV_2 em *reverso*, e partir em direção ao cv_{n+2}. Desse ponto, com o AMV_3 posicionado em *normal*, o trem continua em linha reta, retornando pela linha inferior. Outra possibilidade é o trem que está no circuito cv_4 optar por mudar de via. Nesse caso, o trem teria solicitado a movimentação do AMV_1 e do AMV_3 para *reverso*. Em seguida, o trem ocuparia o circuito cv_{n+3}, prosseguiria até o cv_{n+1}, para então solicitar o posicionamento do AMV_3 e do AMV_4 em *normal*, e partiria em direção ao circuito de via cv_{2n}.

Atualmente parte do tráfego metroviário é controlado por sistemas eletrônicos de sinalização, implementados através do uso de microprocessadores. Estes sistemas oferecem facilidades para a diminuição do intervalo entre trens consecutivos na malha, aumentando a capacidade de transporte com níveis compatíveis de disponibilidade, segurança, presteza e confiabilidade, a um custo adequado.

Existem várias outras características do controle do sistema metroviário que podem sofrer mudanças para melhorar o seu funcionamento e sua operação. Uma das opções é a utilização de estações rádio-base [FAM98], onde informações podem ser passadas diretamente para o trem via rádio. Também pode-se considerar a operação automática do trem (*driveless operation*), via rádio, para o envio e recebimento de dados diretamente para os equipamentos de via. Com a sinalização via rádio, a distância entre os trens não fica limitada por trechos de via e zonas de proteção. Mais trens podem, então, estar sobre a mesma linha e a freqüência dos trens pode ser maior. Os trens também teriam a responsabilidade de determinar, eles mesmos, suas próprias velocidades seguras, tomando em consideração as condições da via, além da presença e da distância de outros trens.

Outra possibilidade é que os circuitos de via poderiam ser ocupados dinamicamente, na realização de uma determinada rota. Não seria necessária a ocupação de um conjunto de

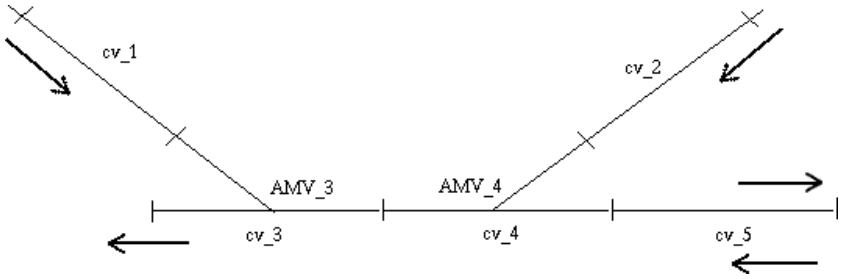

Figura 11: Segmento de via utilizado para manobras de retorno dos trens

circuitos de via (alinhamento de rota) para a movimentação numa linha, o que aumenta em muito a disponibilidade, o rendimento e a capacidade do sistema. Define-se apenas um intervalo de circuitos de via para a operação segura do sistema. Como por exemplo, quando da ocupação de circuitos de via concorridos.

4 Análise, Verificação e Síntese de Propriedades

As características dos equipamentos, dos componentes, do funcionamento e da operação do sistema metroviário mostram sua grande complexidade. São inúmeros circuitos de via e vários AMVs presentes numa malha espalhada por quilômetros. Além disso, é preciso lidar também com a presença de vários trens trafegando simultaneamente por toda a malha metroviária. Ademais, a ferramenta HyTech não suporta a totalidade da complexidade do modelo para um sistema do porte do sistema metroviário completo. A solução, dessa forma, é a divisão da malha metroviária em segmentos independentes, para um posterior estudo de caso em separado. Seguindo essa estratégia, é possível exercitar e modelar as principais características do sistema, observando até que ponto a segurança destas características é garantida.

4.1 O Segmento Modelado

Para efeitos de modelagem e análise, foi selecionado um segmento de via de uma região da malha metroviária. O segmento modelado é ilustrado na Figura 11. Este segmento é composto de cinco circuitos de via e representa uma região utilizada para manobras de retorno dos trens. Estão presentes também dois AMVs que são responsáveis pelo controle das mudanças de via, de acordo com o sentido do tráfego e a presença dos trens nos circuitos de via adjacentes. O modelo inclui ainda dois trens trafegando no segmento em questão, respeitando as regras de tráfego da região. As características e complexidade deste segmento de via permitem que propriedades interessantes do sistema possam ser verificadas e sintetizadas.

A complexidade do modelo e suas características intrínsecas são de ordem tal que a ferramenta HyTech ainda conseguiu dar suporte automático na fase de análise e síntese de propriedades. A experiência prática revelou, entretanto, que qualquer ampliação na complexidade do sistema modelado passaria a exigir recursos de memória além da capacidade total disponível na máquina usada para executar a ferramenta. No caso, o equipamento usado para exercitar os modelos foi um PC comum, com um processador Pentium II de 350 MHz, dispondo de 320 MB de memória RAM, e mais 420 MB de espaço para *swap* em disco.

Para a verificação de propriedades dessa parte da via, os dois trens trafegam conforme as regras de sentido, velocidade e posição na via que lhes foram impostas. Ambos começam seu percurso no mesmo ponto inicial, no circuito de via cv_1. A partir do circuito de via cv_1 o trem deve ocupar o circuito de via cv_3. Ao ocupar o circuito cv_3, o pedido de posicionamento do AMV_3 para *reverso* deve ser efetuado. Quando estiver atravessando do cv_3 para o cv_4, outro pedido é colocado para que o AMV_4 seja colocado em *normal*. Na seqüência, o trem ocupa o circuito de via cv_5 e se move até o final deste circuito. Neste ponto, o sentido do tráfego se inverte e o trem retorna pelo mesmo circuito de via cv_5. Novamente, o pedido de posicionamento do AMV_4 para *normal* é efetuado quando da ocupação do cv_4. Já na segunda ocupação do circuito de via cv_3, o AMV_3 deve estar em *normal* e o trem, dessa forma, continua na mesma linha de onde veio, mas agora trafegando em sentido contrário. O trecho todo é ilustrado de maneira mais abrangente na Figura 7. Quando o trem sai do circuito de via cv_3 indo para a esquerda, o modelo adotado o faz retornar, de maneira não-determinística, tanto pelo circuito de via cv_1 como pelo circuito de via cv_2. Se partir do circuito de via cv_2 o trem ocupa o circuito de via cv_4, passando a trafegar na linha inferior, no sentido inverso. Para isso, é preciso solicitar o posicionamento do AMV_4 para *reverso*. Na seqüência, o trem passa para o cv_3, após requisitar o posicionamento do AMV_3 para *normal*. Após passar pelo cv_3 a simulação recomeça no cv_1 ou no cv_2, novamente.

4.2 Os Modelos Adotados

O modelo completo do segmento de malha metroviária em questão é formado por nove autômatos híbridos. São dois autômatos, um para cada trem que trafega pelo segmento, mais dois autômatos, um para cada AMV, e ainda mais cinco autômatos, um para cada um dos circuitos de via.

Os autômatos dos trens modelam a movimentação dos mesmos pelo segmento de via, respeitando a ocupação correta dos circuitos. Para tal, é preciso respeitar as regras de movimentação citadas na subseção anterior. Nos autômatos dos trens também são especificados os perfis de velocidade, incluindo os perfis negativos, que indicam a inversão no sentido do tráfego. Os autômatos que capturam o comportamento dos trens são os componentes mais complexos do modelo, pois uma boa parte das decisões tomadas pelo sistema estão localizadas nos sistemas de bordo dos trens. Os dois autômatos para os AMVs modelam o controle e a movimentação física dos AMVs, situados nos circuitos de via cv_3 e cv_4. Os AMVs podem assumir quatro estados. Podem estar se movimentando, para *reverso* ou *normal*, ou ainda podem estar estacionados na posição *normal* ou *reversa*. A cada pedido de movimentação, os AMVs levam um determinado tempo para completar a movimentação. Os autômatos dos circuitos de via são simples. Eles modelam apenas a ocupação e a liberação dos trens dentro desses trechos de via.

A seguir são discutidos e detalhados cada um dos autômatos utilizados na modelagem. Devido à similaridade dos modelos, são descritos somente um dos autômatos dos trens, um dos autômatos dos AMVs e também apenas um dos autômatos para os circuitos de via. A especificação do modelo completo, na linguagem própria da ferramenta HyTech, é mostrada no Apêndice A.

4.2.1 O Autômato do Trem

O autômato do trem possui oito estados, usados para representar a circulação do trem pelos circuitos de via e a passagem do mesmo pelos AMVs. Veja a ilustração na Figura 12. As variáveis usadas nos autômatos dos trens são $aux1$ e k, para um dos trens, e $aux2$ e w, para o outro trem. As variáveis $aux1$ e $aux2$, indicam a distância dos trens dentro dos circuitos ocupados. As variáveis k e w regulam a ocupação dos circuitos de acordo com as regras de manobras estipuladas para o retorno dos trens. Essas variáveis controlam a ocupação dos circuitos de via, de acordo com o sentido do tráfego e também de acordo com a ocupação do circuito pelo outro trem, respeitando as normas de segurança.

O autômato inicialmente se encontra no estado "Ocupar", indicando a iminência do trem ocupar um dos circuitos de via. No caso, a requisição de ocupação é feita para o primeiro circuito de via, o cv_1. Se não existir nenhum outro trem no circuito de via cv_1, a ocupação é permitida. Porém, o trem só pode começar a se locomover pelo circuito de via cv_1 se não existir ocupação cedida a outro trem no circuito de via cv_3. Isso porque o sistema opera com uma margem de segurança de um circuito de via adjacente, no mesmo sentido de tráfego. Observe que sem a margem de segurança, o trem pode trafegar pelo circuito cv_1 e colidir com outro trem que esteja no início do circuito cv_3 e trafegando no mesmo sentido. Ou poderia haver um outro trem trafegando no circuito cv_3 no sentido contrário, da direita para a esquerda, uma situação que ainda implicaria em risco de colisão. Após a ocupação efetiva do circuito de via cv_1, o trem começa a se locomover a uma velocidade que varia de 7 a 9 metros por segundo. O trem deve se manter nesse intervalo de velocidade até atravessar toda extensão do circuito de via cv_1. Quando atravessar os 200 metros do circuito de via, o trem deve desocupar o circuito cv_1 e ocupar o próximo circuito de via, o cv_3. Se este último estiver ocupado o trem deve esperar até que esse circuito seja liberado.

Quando lhe é cedida a ocupação do circuito cv_3, o trem continua a mover-se com uma velocidade no intervalo de 7 a 9 metros por segundo. Essa velocidade é exercida em regiões que possuem AMVs. A efetivação da ocupação do circuito de via cv_3 só ocorre se não existir nenhum outro trem no circuito de via cv_4, em qualquer sentido, ou no circuito de via cv_5, também em qualquer sentido. Assim, evita-se o bloqueio dos trens durante o tráfego nessa região. No momento em que o trem começa a se movimentar no cv_3, um pedido de posicionamento é feito ao AMV_3. A máquina de chave do AMV_3 movimenta o mecanismo para a posição correta, ou seja, a posição *reversa*. O trem atravessa o AMV_3 e continua sua viagem pelo circuito de via cv_3. Antes de deixar o circuito cv_3, o trem faz a requisição para a ocupação do circuito de via cv_4.

Continuando, ao entrar no circuito de via cv_4 depois da requisição ser aceita, o trem deve pedir o posicionamento do AMV_4 situado neste circuito. A máquina de chave movimenta o AMV_4 para o estado *normal*. A travessia é completada e a liberação do circuito de via cv_4 é efetivada.

No circuito de via cv_5 a operação é a mesma. A ocupação é efetivada e o trem percorre todo o circuito de via cv_5. Quando o trem chega ao fim da via, no circuito cv_5, ele pára e inverte o sentido do percurso. A ocupação do circuito de via cv_5 é novamente realizada, porém, agora, com o trem em movimento no sentido contrário. O perfil de velocidade é tomado como um valor no intervalo de -9 a -7 metros por segundo. A velocidade negativa indica a movimentação no sentido da direita para a esquerda, ou seja, no sentido contrário ao sentido da rota original. Se houver um outro trem ocupando o cv_4, o trem ocupa o cv_5, porém não se movimenta até que o outro trem passe pelo cv_4. Observe que na ocupação do circuito de via cv_4, mencionado, o trem está trafegando no sentido inverso ao original, já que de outra forma poderia ocorrer um bloqueio.

Ao deixar o circuito de via cv_5, o trem deve ocupar o circuito de via cv_4. Isso, porém, só ocorrerá se nenhum outro trem estiver no circuito cv_4, vindo do circuito cv_2, percorrendo a rota superior. Quando da passagem para o circuito cv_4, novamente é solicitado o posicionamento do AMV_4 para o estado *normal*. O perfil de velocidade no cv_4 continua no intervalo de -9 e -7 metros por segundo, até que o trem atravesse os 200 metros deste circuito de via. Mais uma vez, na liberação do circuito cv_4, o circuito de via cv_3 deve ser ocupado, e mais uma vez efetua-se o pedido de posicionamento do AMV_3, agora para o estado *normal*.

Após passar por todo o circuito de via cv_3, o trem desocupa este circuito, liberando-o, e retorna à posição inicial. Como é assumida uma operação circular e indefinida do sistema, o trem continuaria nessa rota, chegaria a uma outra região terminal e voltaria pela rota de sentido contrário, na linha superior. O modelo pode tanto recolocar o trem no circuito de via cv_1 como no circuito de via cv_2, de maneira não-determinística. Se o retorno se der pelo circuito cv_1, a operação se reinicia do estado inicial. Ou seja, o processo de funcionamento deve se repetir novamente.

Por outro lado, o trem pode retornar pelo circuito de via cv_2. Para evitar o bloqueio

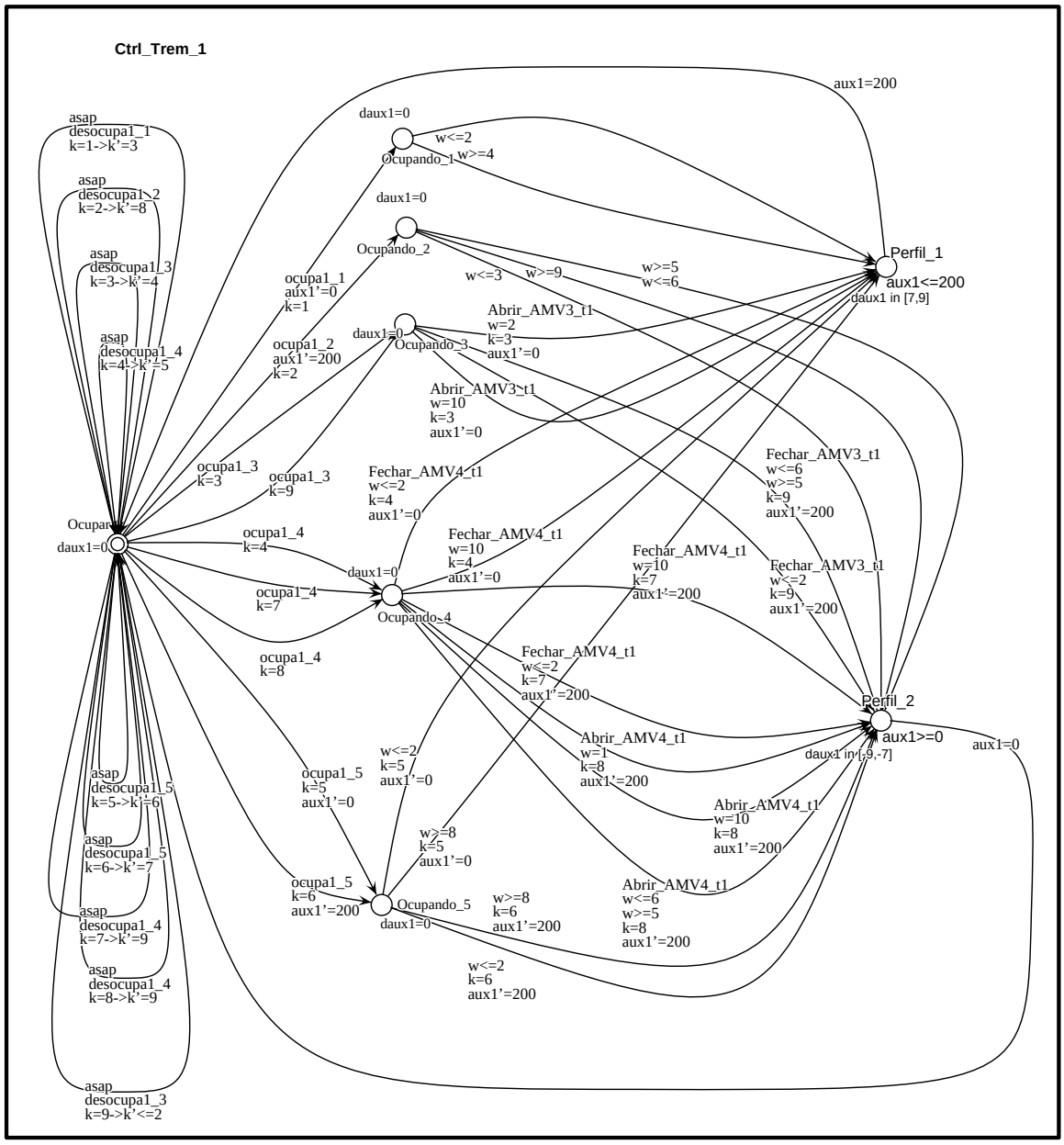

Figura 12: Autômato do trem

dos trens, o trem que vem do circuito de via cv_2 só pode ocupar o circuito de via cv_4, caso não haja nenhum outro trem no circuito de via cv_3 ou no circuito de via cv_4. Quando se movimentar para fora do circuito de via cv_2, o trem ocupa o circuito de via cv_4 e também requisita o posicionamento do AMV_4 para o estado *reverso*. Na seqüência, o trem passa pelo AMV_4, prossegue para o fim desse circuito de via e o libera. Finalmente, o trem ocupa o circuito de via cv_3 e pede o posicionamento do AMV_3, o qual se movimenta para o estado *normal*. Após o trem atravessar toda a extensão do circuito cv_3, este circuito é liberado. Por fim, o trem retorna para o circuito de via cv_1 ou para o circuito de via cv_2, recomeçando a simulação.

4.2.2 O Autômato do AMV

O autômato do AMV é constituído de quatro estados, usados para representar o controle e a movimentação do AMV. Veja a Figura 13. As variáveis usadas nos autômatos dos AMVs são $t3$ e $t4$, $t3$ para um dos AMVs e $t4$ para o outro AMV. As variáveis $t3$ e $t4$ indicam o tempo de movimentação dos AMVs para se posicionarem nos estados de *normal* ou de *reverso*.

O autômato híbrido do AMV é inicializado na posição *normal*. Quando ocorre um pedido de movimentação, o AMV começa a se deslocar, sendo seu tempo de movimentação medido por um relógio. O tempo normal de movimentação do AMV é de 15 segundos. Mas essa medida é imprecisa, podendo ocorrer um adiantamento de até 20% nessa operação. Por isso, o AMV pode concluir sua movimentação no intervalo de 12 a 15 segundos.

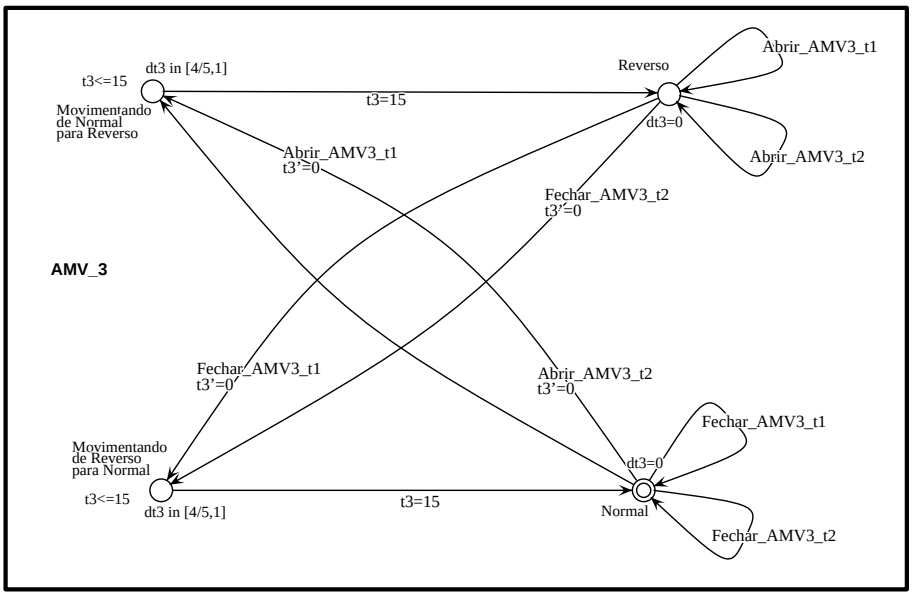

Figura 13: Autômato do AMV

A partir da posição *normal*, onde o sistema é inicializado, um pedido de movimentação do AMV pode tomar dois caminhos. Por um lado, assuma que há um pedido de movimentação para o AMV se posicionar no estado *reverso*. O AMV, então, começa a se movimentar. Em algum ponto no intervalo de 12 a 15 segundos o AMV estará na posição *reversa* e deve ser travado. Neste instante o autômato de controle do AMV passa para o estado *reverso*, terminando a movimentação do AMV. Por outro lado, o pedido pode ser para o próprio

estado *normal*. Nesse caso, como o AMV já se encontra na posição desejada, sua máquina de chave não é acionada. A movimentação do AMV no sentido contrário, a partir do estado *reverso*, é similar.

4.2.3 O Autômato do Circuito de Via

Os circuitos de via são modelados por autômatos indicando apenas a ocupação e a liberação desses circuitos pelos trens. Na transição entre esses estados são emitidas mensagens para efeito de sincronização com a posição do trem na via. Na Figura 14, é mostrado o autômato de um circuito de via, modelado dessa maneira simples.

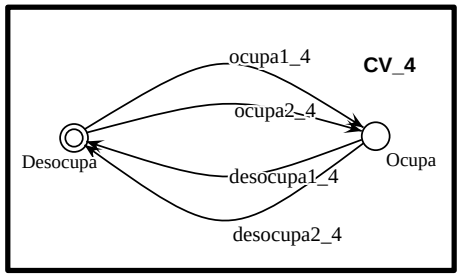

Figura 14: Autômato do circuito de via

4.3 Verificação e Síntese de Propriedades

Construídas as especificações dos trens, dos AMVs e dos circuitos de via, é possível verificar algumas propriedades do modelo. Também é possível sintetizar valores para certos parâmetros de operação, de modo a obter valores mais justos para os mesmos.

4.3.1 Verificação de Propriedades

Várias propriedades foram analisadas de modo a garantir a segurança da operação do segmento de malha selecionado. Foram analisadas propriedades que dizem respeito à posição relativa dos trens na via, e também foram consideradas propriedades relativas à movimentação dos AMVs quando da passagem dos trens pelos circuitos de via que contém esses AMVs. A especificação de uma propriedade típica é mostrada na Figura 15, no código próprio da ferramenta HyTech. Neste caso típico, é definida uma região final, *final_reg*, que caracteriza as configurações inseguras para o sistema. A análise começa de uma região inicial, *init_reg*, que contém todas as configurações iniciais do autômato produto correspondente ao modelo completo. No exemplo, a análise termina com o cálculo, no sentido *backward*, da região final que é alcançada. A região calculada é então comparada com a região que contém todas as configurações iniciais do autômato. Se não for detectado nenhum ponto em comum à essas duas regiões, a operação é segura. A diferença entre as várias análises realizadas está, principalmente, na especificação das regiões inicial e final, e na especificação da diretiva para o cálculo de alcançabilidade.

Situações que levam o sistema a uma operação insegura, são descritas a seguir:

Quanto à posição dos trens: As normas de segurança para ocupação dos circuitos de via e para o tráfego dos trens ditam quais situações são inseguras. As situações inseguras, entre os trens, são:

```
var
   final_reg, init_reg, reached: region;
init_reg:=
     loc[Ctrl_Trem_1] = Ocupar &
     loc[Ctrl_Trem_2] = Ocupar &
     loc[AMV_3] = Normal &
     loc[AMV_4] = Normal &
     k=1 & w=1 & t3=0 & t4=0;
final_reg :=
      loc[Ctrl_Trem_1] = Perfil_1 &
      loc[Ctrl_Trem_2] = Perfil_1 &
      k=3 &
      (w=3 | w=4 | w=5);
reached:=
reach backward from final_reg endreach;
if empty (reached & init_reg)
   then prints "Trem esta seguro";
   else prints "Trem violou a seguranca";
endif;
```

Figura 15: Exemplo da análise de uma propriedade

1. Um trem ocupando o circuito de via cv_3, indo em direção ao circuito cv_4, existindo outro trem na mesma rota ocupando os circuitos de via cv_4 ou cv_5, em qualquer direção. Nessa situação, os trens ficariam bloqueados;
2. Um trem ocupando o circuito de via cv_2, existindo outro trem no circuito de via cv_4, na mesma rota. Não havendo um trem no cv_4 e se houver um outro trem vindo do circuito cv_3 ou vindo do circuito cv_5, o trem do circuito de via cv_2 ainda teria um espaço de segurança para frear até alcançar o AMV;
3. Um trem ocupando o circuito de via cv_1, existindo outro trem ocupando o circuito de via cv_3 na mesma direção. Nesse caso poderia não haver a margem de segurança para a frenagem. No caso oposto, com o trem no circuito de via cv_3, em sentido contrário e se afastando do pátio de manobras, haveria espaço suficiente para a frenagem do trem que vem no circuito cv_1, antes da intersecção com a rota do trem que vinha pelo circuito de via cv_3.

Para a análise de cada uma dessas propriedades, é preciso que essas situações sejam caracterizadas por predicados sobre configurações.

Quanto à posição dos trens em relação aos AMVs: As situações inseguras na operação do sistema para esses componentes são descritas a seguir:

1. Um trem ocupando o circuito de via cv_3, vindo do circuito cv_1, à uma distância muito próxima do AMV_3 (distância insegura) e o AMV_3 está no estado *normal*, ou está em movimento;

2. Um trem ocupando o circuito de via cv_3, vindo do circuito cv_4 (sentido inverso), à uma distância muito próxima do AMV_3 (distância insegura) e o AMV_3 está em movimento;
3. Um trem ocupando o circuito de via cv_4, vindo do circuito cv_3, à uma distância muito próxima do AMV_4 (distância insegura) e o AMV_4 está em movimento;
4. Um trem ocupando o circuito de via cv_4, vindo do circuito cv_5 (no sentido inverso), à uma distância muito próxima do AMV_4 (distância insegura) e o AMV_4 está no estado *reverso*, ou está em movimento;
5. Um trem ocupando o circuito de via cv_4, vindo do circuito cv_2, à uma distância muito próxima do AMV_4 (distância insegura) e o AMV_4 está no estado *normal*, ou está em movimento;

4.3.2 Sintetização de Valores para as Variáveis *aux* e *t*, em separado

O alvo do primeiro exercício de sintetização de parâmetros foi encontrar o valor da distância máxima que o trem pode percorrer, dentro do circuito de via ocupado, até que o AMV atinja a posição correta solicitada, e ainda mantendo as condições de segurança na operação do sistema. Foi suposto, na análise, que o trem realiza o pedido de movimentação do AMV logo na entrada do circuito de via que contém o AMV. Nos autômatos dos trens, são as variáveis $aux1$ e $aux2$ que indicam as distâncias percorridas pelos trens, dentro do circuito de via que ocupam. Nessa parametrização, foi usada a variável $aux1$ como parâmetro. Devido a simetria do autômato do trem em relação às variáveis $aux1$ e $aux2$, o mesmo resultado vale para a variável $aux2$. Outro parâmetro importante na operação do sistema é o momento quando o trem solicita a movimentação do AMV, em relação a distância percorrida pelo trem dentro do circuito de via. A velocidade do trem no pátio de manobras está situada entre 7 e 9 m/s, ou entre -9 e -7 m/s, quando o trem está trafegando na linha em sentido contrário. No modelo dos AMVs, o tempo de movimentação de um AMV é medido por um relógio, isto é, por uma variável t_i cuja derivada primeira é da forma $t_i = 1 + \epsilon$. Assumindo uma imprecisão de até 20%, para menos, na medida do tempo de posicionamento do AMV, a variável t_i deve assumir um valor no intervalo $[0.8, 1.0]$ segundos. Quando o AMV termina sua movimentação ele deve ser travado e, nesse instante, o trem deve estar a uma distância ainda segura para realizar a travessia.

Em um outro estudo, o objetivo foi determinar quão mais lento os AMVs podem ser ao se movimentar, sem comprometer o funcionamento seguro da operação do sistema. Para isso, foi sintetizado o intervalo de movimentação do AMV, usando a variável t_3 como parâmetro, e de tal forma a manter o sistema operando dentro dos requisitos de segurança. Devido à simetria do autômato do AMV em relação às variáveis t_3 e t_4, o mesmo resultado vale para a variável t_4. Outro aspecto interessante da operação do sistema diz respeito à distância mínima, em relação ao AMV, que o trem deve observar quando efetua o pedido de movimentação do AMV. Se essa distância não for observada, o conjunto poderia entrar numa região insegura, onde o trem chegaria ao AMV antes deste estar posicionado e travado. Para determinar essa distância, foi usada uma variável que representa a distância entre a posição do trem e o início do circuito de via. A ferramenta HyTech, então, sintetizou um intervalo de valores seguro para essa variável.

4.3.3 A Relação entre a Posição do Trem e o Tempo de Movimentação do AMV

No item anterior foram descritos mecanismos para sintetizar valores para as variáveis que medem parâmetros críticos para certas distâncias e certos tempos, característicos da operação do sistema. Uma outra abordagem, mais rica em informação, seria medir que relações entre esses parâmetros devem ser observadas para que o sistema ainda opere de forma segura. Essas relações permitiriam o ajuste simultâneo de vários desses parâmetros. Nessa

```
var
 final_reg, init_reg, reached: region;
init_reg:=
     loc[Ctrl_Trem_1] = Ocupar &
     loc[AMV_3] = Normal &
     t3=0 & aux1=pedido;
final_reg :=
      loc[Ctrl_Trem_1] = Ocupar &
      aux1=dist_trem &
      (loc[AMV_3] = Normal |
      loc[AMV_3] = Abrindo |
      loc[AMV_3] = Fechando);
reached:== reach forward from init_reg endreach;
print omit all locations
      hide non_parameters in
           reached & final_reg
      endhide;
```

Figura 16: Exemplo da análise paramétrica

subseção são descritas as condições sob as quais foram realizados alguns estudos de parametrização múltipla.

As variáveis do tipo *aux* e *t* formaram o conjunto básico de variáveis cujos valores foram parametrizados, de forma a se obter uma relação de dependência entre ambas. Nas primeiras sintetizações, utilizou-se a marca de posição zero, dentro do circuito de via, como o ponto onde o trem efetua o pedido de movimentação do AMV. Posteriormente, além das variáveis *aux* e *t*, o ponto dentro do circuito de via onde o trem solicita a movimentação do AMV também foi colocado como parâmetro.

A Figura 16 apresenta um exemplo desse tipo de análise paramétrica. A região de configurações iniciais *init_reg* apresenta: (i) o trem_1 na iminência de ocupar o circuito de via que contém o AMV_3; (ii) o AMV_3 na posição normal; (iii) o tempo de movimentação do AMV_3 zerado; e (iv) a parametrização da distância, dentro do circuito cv_3, onde o trem solicita a movimentação do AMV_3, através do predicado $aux_1 = pedido$. As características da região final, *final_reg*, descrevem as situações inseguras, quais sejam: (i) o trem ocupando o circuito de via cv_3; (ii) o AMV_3 não está na posição *reversa*, ou seja, está numa das posições *normal*, ou se movimentando para *normal* ou para *reverso*; e (iii) a distância do trem, a partir do início do circuito cv_3, que é descrita usando-se o predicado $aux_1 = dist\_trem$, onde *dist_trem* é um parâmetro. A ferramenta HyTech determina , automaticamente, a região *reached*, que descreve todas as configurações alcançáveis a partir da região *init_reg*. Neste exemplo, foi usada a primitiva *Post*, que implementa uma busca do tipo *forward* na análise. Em seguida, a ferramenta calcula os pontos comuns entre as regiões *reached* e *final_reg*. Como a região *final_reg* descreve as configurações inseguras para operação do sistema, é preciso negar a relação entre os parâmetros *pedido* e *dist_trem* , calculada pelo sistema, para se obter a relação que deve ser observada entre esses parâmetros para que o sistema opere em segurança.

4.4 Os Resultados Obtidos

Os resultados obtidos a partir das análises realizadas, descritas na seção anterior, foram classificados em cinco categorias. A primeira categoria estuda propriedades de segurança relacionadas à posição dos trens na via. A segunda categoria estuda o posicionamento seguro dos AMVs quando da passagem dos trens pelos AMVs. Na terceira categoria estudam-se parâmetros seguros para a distância dos trens em relação aos AMVs, media a partir do instante em que os trens solicitam a movimentação dos AMVs. A quarta categoria estuda o mesmo problema que a categoria anterior, porém agora parametrizando o tempo de movimentação dos AMVs. A quinta categoria descreve a parametrização realizada sobre essas duas últimas variáveis, simultaneamente, procurando uma relação segura entre a distância dos trens até os AMVs versus o tempo de movimentação dos AMVs. As análises das propriedades de segurança e seus resultados são apresentados em detalhes no Apêndice B. As análises e os resultados relativos às sínteses efetuadas são descritas em detalhes no Apêndice C.

Em todos os estudos paramétricos realizados foi adotada ainda uma segunda variante de parametrização. Nessa variante, é parametrizado também o ponto onde é executado o pedido de movimentação dos AMVs pelos trens, em oposição ao ponto padrão do pedido, que é logo na tomada do circuito de via, ou seja, na posição 0 (zero) do circuito.

A Tabela 1 mostra algumas características das análises realizadas, de uma forma geral.

Posicionamento	Caracterização	Análise	Iterações	Memória (MB)	Tempo (s)
Trens	Unsafe	Backward	2 a 3	350 a 450	110 a 260
Trens	Safe	Backward	–	–	–
Trens	Unsafe	Forward	–	–	–
Trens	Safe	Forward	61	≥ 550	250 a 330
AMVs	Unsafe	Backward	7	450 a 550	280 a 300
AMVs	Safe	Backward	–	–	–
AMVs	Unsafe	Forward	82	200 a 250	150 a 160
AMVs	Safe	Forward	82	200 a 250	150 a 170

Tabela 1: Características gerais de algumas verificações

Cada linha da tabela, ilustra uma situação típica encontrada durante os exercícios de análise de propriedades do modelo. A primeira coluna identifica se a propriedade se refere à uma relação de posicionamento entre trens ou à uma relação de posicionamento entre um trem e um AMV, quando o trem se encontra no circuito de via onde se localiza o AMV. A segunda coluna destaca como a propriedade foi caracterizada para análise. Uma caracterização do tipo *unsafe* define predicados inseguros que descrevem regiões de configurações indesejadas para a operação do sistema. Já uma caracterização do tipo *safe* define predicados seguros, que mapeiam regiões de configurações seguras, ou seja, regiões que podem ser atingidas sem problemas para a operação segura do sistema. A terceira coluna apresenta o tipo de análise que foi utilizada para a verificação da propriedade. Ou seja, se foi usado o operador *Post*, para análise *forward*, ou o operador *Pre*, para análise *backward*. A quarta coluna mostra o número de iterações necessárias para que a análise convirja. Algumas dessas análises não foram bem sucedidas, sendo interrompidas por falta de recursos de máquina ou da ferramenta. A quinta coluna apresenta o intervalo aproximado da quantidade de memória que foi utilizada para se completar as análises. Em algumas análises a memória total disponível não foi suficiente para a finalização da análise. A sexta e última coluna identifica o intervalo aproximado do tempo total gasto nas análises efetuadas. Por exemplo, a linha 5 da Tabela 1 informa que um dos exercícios de verificação da distância segura entre trens

e AMVs foi modelado como uma propriedade do tipo *unsafe* e a análise evoluiu de forma *backward*, usando o operador *Pre*. Nessa execução a ferramenta usou aproximadamente 300 segundos, necessitou de 7 iterações para convergir, e usou cerca de 500 MB, entre memória RAM e memória para *swap*.

4.4.1 Verificação da distância segura entre dois trens

Foram simuladas várias situações envolvendo a movimentação de dois trens pelo segmento de malha metroviária selecionado. Cada situação foi classificada como segura (*safe*) ou insegura (*unsafe*) de acordo com as regras de segurança para operação do sistema metroviário. Nessa subseção o enfoque esteve sempre na posição relativa dos trens, bem como no sentido de percurso de cada um deles.

O resultado das propriedades analisadas e verificadas são apresentadas na Tabela 2. Nessas verificações, o pedido de movimentação dos AMVs se dá sempre na posição zero do circuito de via. A primeira e a segunda colunas indicam a posição e o sentido do primeiro trem. A terceira e a quarta colunas repetem esta informação para o segundo trem. A quinta coluna classifica a situação, caracterizada nas primeiras quatro colunas, como do tipo *safe* ou do tipo *unsafe*. Para efeitos de segurança, uma região de configurações do tipo *safe* pode ser atingida por alguma trajetória que parte da região de configurações iniciais do modelo. Uma região do tipo *unsafe*, se alcançada, representa uma violação de segurança na operação do sistema. A ferramenta HyTech calcula as configurações alcançáveis do modelo e determina se alguma delas é do tipo *unsafe*. A última coluna da tabela apresenta a indicação produzida pela ferramenta. Uma inspeção da tabela revela que não foram detectadas violações de segurança no que diz respeito às propriedades testadas.

Trem 1	Sentido	Trem 2	Sentido	Tipo	Resultado
cv_1	direto	cv_3	direto	Unsafe	Não alcançada
cv_2	direto	cv_4	inverso	Unsafe	Não alcançada
cv_3	direto	cv_4, cv_5	qualquer	Unsafe	Não alcançada
cv_3	direto	cv_3	inverso	Unsafe	Não alcançada
cv_5	qualquer	cv_3, cv_4	direto	Unsafe	Não alcançada
cv_1	direto	cv_3	inverso	Safe	Alcançada
cv_5	qualquer	cv_4	do cv_2	Safe	Alcançada
cv_3	direto	cv_2	direto	Safe	Alcançada
cv_3	inverso	cv_2, cv_5	qualquer	Safe	Alcançada

Tabela 2: Posição relativa dos trens

4.4.2 Verificação da distância segura entre o trem e o AMV

Nesta subseção, o interesse está voltado para determinar se o trem pode alcançar certos pontos do circuito de via no intervalo de tempo decorrido desde o instante que solicita a movimentação do AMV até o instante em que o AMV completa sua movimentação e se encontre travado. O instante no qual o trem solicita a movimentação do AMV sempre coincide com o momento em que o trem entra no circuito de via. Este é o marco de distância zero. Para um trem trafegando no sentido padrão da via, da esquerda para a direita, o AMV se encontra no outro extremo do circuito de via à uma distância de 200 metros. Para o outro sentido de tráfego, da direita para a esquerda, as posições se invertem. O AMV passa a estar localizado na posição zero e o trem entra no circuito de via no marco de 200 metros.

Se for adversa a relação entre a velocidade de movimentação do AMV, a posição do trem e sua velocidade, o trem pode alcançar uma distância perigosamente próxima do AMV antes que este se encontre posicionado e travado. As análises reportadas nessa subseção investigam essa possibilidade. A estratégia adotada é comum a todos os experimentos dessa subseção. Em primeiro lugar, é escolhido um dos circuitos de via que contém um AMV, junto com um sentido de percurso para o trem. Essa escolha descreve a região inicial do sistema. Em segundo lugar, é descrita a posição indesejável para o AMV, junto com uma posição final para o trem, dentro do circuito de via. Essa segunda escolha define a região final que será testada. Dependendo da posição final escolhida para o trem, essa região final pode ou não caracterizar uma região insegura para o sistema. Nos experimentos concluídos, foi definida como insegura toda região que permita o trem chegar a menos de 20 metros do AMV sem que este esteja travado na posição correta. Portanto, no sentido de tráfego padrão, à distância de 20 metros do AMV colocaria o trem na marca de 180 metros. No sentido inverso de tráfego o trem atingiria essa distância quando estivesse na posição de 20 metros.

Os resultados das situações analisadas são apresentados na Tabela 3. A primeira coluna indica o AMV escolhido. A segunda coluna indica o estado do AMV quando da passagem do trem. A terceira coluna indica de onde vem o trem. A quarta coluna representa a distância percorrida pelo trem dentro do *cv*. A quinta coluna apresenta a indicação da ferramenta HyTech, para cada situação. Novamente, uma inspeção da tabela revela que a operação do sistema é segura, nessas condições.

AMV	Posição	Trem de	Distância (m)	Resultado
AMV_3	não Reverso	cv_1	180	Não alcançada
AMV_3	não Normal	cv_4	20	Não alcançada
AMV_4	não Reverso	cv_2	20	Não alcançada
AMV_4	não Normal	cv_3	180	Não alcançada
AMV_4	não Normal	cv_5	20	Não alcançada
AMV_3	não Reverso	cv_1	160	Alcançada
AMV_3	não Normal	cv_4	40	Alcançada
AMV_4	não Reverso	cv_2	40	Alcançada
AMV_4	não Normal	cv_3	160	Alcançada
AMV_4	não Normal	cv_5	40	Alcançada

Tabela 3: Posição entre trens e AMVs

4.4.3 Sintetizando a distância percorrida pelos trens até o travamento do AMV

Foram executadas diferentes sínteses de parâmetros críticos dos quais depende a passagem segura dos trens pelos AMVs. Em particular, foi sintetizada a distância percorrida por um trem após este ativar a movimentação do AMV e até que o AMV esteja posicionado e travado. Em quatro análises executadas foi alterado o ponto onde o trem solicita a movimentação do AMV. Veja a Tabela 4. A primeira coluna indica a distância do ponto dentro do circuito de via onde o trem solicita a movimentação do AMV. A segunda coluna mostra o ponto mais avançado que o trem pode alcançar dentro do circuito de via, até o instante de travamento do AMV. No limite, se essa distância for de 200 metros (todo o comprimento do circuito de via) a passagem do trem pelo AMV ainda poderá ocorrer sem problemas. A terceira coluna indica o intervalo de tempo estipulado para que o AMV complete sua movimentação. Em todas essas simulações a velocidade do trem foi assumida

Aciona AMV (m)	Percorrido (m)	Tempo (s)	Velocidade (m/s)
0	675/4	[12,15]	[7,9]
20	755/4	[12,15]	[7,9]
125/4	800/4	[12,15]	[7,9]
40	835/4	[12,15]	[7,9]

Tabela 4: Distância percorrida pelo trem após ativação do AMV

como um valor no intervalo entre 7 e 9 metros por segundo, como mostra a quarta coluna da tabela. Observa-se dos resultados que a distância limite, contada a partir do início do circuito de via, para que o trem solicite a movimentação do AMV é de 125/4 metros, como pode ser observado da linha 3 da Tabela 4. Ou seja, se o trem aciona o AMV até o ponto distante 125/4 metros do início do *cv*, então sempre que o trem cruzar o AMV, localizado no outro extremo do *cv*, este AMV já estará travado. A saída produzida pela ferramenta HyTech, para o caso ilustrado na primeira linha da Tabela 4, é mostrada na Figura 17. A

```
Number of iterations required for reachability: 10
     dist_trem >= 0 & 4dist_trem <= 675
```

Figura 17: Parametrização da distância percorrida pelo trem até o travamento do AMV

variável *dist_trem* indica a distância percorrida pelo trem após este entrar no circuito de via. O resultado produzido pela ferramenta é a cláusula

$$(dist\_trem \geq 0) \wedge (dist\_trem \leq 675/4).$$

Neste exercício, foi usada a região insegura para o cálculo de alcançabilidade. Portanto, se o trem solicitar a movimentação do AMV na posição zero do circuito de via, a região caracterizada como insegura poderá ser atingida com o trem posicionado em qualquer ponto no intervalo de 0 a 675/4 metros, a contar do ponto inicial do *cv*. Logo, como o AMV está localizado à uma distância de $800/4 = 200$ metros do ponto inicial do *cv*, conclui-se que é seguro solicitar a movimentação do AMV no instante em que o trem adentra o *cv* onde está localizado o AMV.

Também realizou-se um outro estudo para recolher a relação segura entre a distância percorrida pelo trem e a distância do ponto onde o trem solicita a movimentação do AMV. O resultado produzido pela ferramenta é mostrado na Figura 18. A variável *pedido* indica a

```
Number of iterations required for reachability: 10
     pedido <= dist_trem & 4dist_trem <= 4pedido + 675
```

Figura 18: Relação entre o ponto de acionamento do AMV e a distância percorrida pelo trem

distância do ponto onde a movimentação do AMV é solicitada. A variável *dist_trem* indica a

distância percorrida pelo trem dentro do circuito de via. A saída produzida pela ferramenta é formada pela conjunção

$$(pedido \leq dist\_trem) \wedge (dist\_trem \leq pedido + 675/4).$$

Essa cláusula indica que relação deve vigorar entre as variáveis *pedido* e *dist_trem* para que a região insegura seja alcançada por alguma trajetória do autômato produto que modela o sistema. Portanto, para uma operação segura, deve-se observar a negação desta relação, formada pela cláusula,

$$(pedido > dist\_trem) \vee (dist\_trem > pedido + 675/4).$$

A primeira cláusula nunca ocorre, já que $dist\_trem \geq pedido$ é sempre verificada, por construção. Então, para operar o sistema de forma segura, devemos sempre manter a relação

$$dist\_trem > pedido + 675/4.$$

4.4.4 Sintetizando a operação dos AMVs quando da passagem dos trens

Nessa subseção, o objetivo é sintetizar parâmetros da operação dos AMVs. Em particular, deseja-se obter parâmetros para o tempo de movimentação dos AMVs, sempre garantindo que estes estejam travados quando da passagem dos trens. Quatro análises foram executadas alterando-se apenas o ponto onde o trem solicita o posicionamento do AMV. Dessa forma, foram sintetizados quatro diferentes intervalos de valores para o tempo de movimentação do AMV. Veja a Tabela 5. A primeira coluna indica a distância do ponto dentro do circuito de via onde o trem solicita a movimentação do AMV. A segunda coluna mostra que a distância

Ativa AMV (m)	Percorrido (m)	Tempo (s)	Velocidade (m/s)
0	200	(100/7,1120/63)	[7,9]
20	200	(90/7,16)	[7,9]
125/4	200	(12,15)	[7,9]
40	200	(80/7,128/9)	[7,9]

Tabela 5: Parametrização do tempo de operação do AMV

máxima percorrida pelo trem no exato instante de travamento do AMV foi fixada em 200 metros, ou seja, essa distância corresponde a todo o curso de um circuito de via. A terceira coluna indica o intervalo de tempo de movimentação do AMV, de modo que o sistema mantenha sua operação segura. Observe que os intervalos apresentados na terceira coluna são todos intervalos abertos nesses valores. Mais uma vez, em todas essas simulações, a velocidade do trem foi assumida no intervalo entre 7 e 9 metros por segundo, como indicado na quarta coluna da tabela. Novamente, observa-se dos resultados que a distância limite a partir do início do circuito de via para que o trem solicite a movimentação do AMV é de 125/4 metros, como está indicado na linha 3 da Tabela 5. Neste caso, o tempo de movimentação do AMV deve estar localizado no intervalo de 12 a 15 segundos. A saída produzida pela ferramenta HyTech, para o caso ilustrado na primeira linha da Tabela 5, é mostrada na Figura 19. A relação obtida indica os valores para o tempo de movimentação do AMV de tal modo que a região de configurações indesejadas seja atingida. Portanto, o predicado calculado descreve os intervalos de tempo onde a operação do sistema é insegura. Negando-se essa cláusula, obtém-se a relação que deve ser mantida para uma operação segura do sistema:

```
Number of iterations required for reachability: 7
     7tempo_amv <= 200 & 63tempo_amv >= 1120
  |
     tempo_amv >= 0 & 7tempo_amv <= 100
```

Figura 19: Parametrização do tempo de operação do AMV

$$tempo\_amv > 200/7 \ \vee \ tempo\_amv < 1120/63$$
$$\wedge$$
$$tempo\_amv < 0 \ \vee \ tempo\_amv > 100/7.$$

A parcela $tempo\_amv < 0$ nunca ocorre e pode ser descartada. A condição $tempo\_amv > 200/7$ também nunca ocorre, já que a velocidade do trem está limitada a 7 metros por segundo e a posição do AMV no circuito de via está fixada em 200 metros. Portanto, a cláusula que caracteriza a operação segura dos AMVs se reduz a

$$tempo\_amv > 100/7 \ \wedge \ tempo\_amv < 1120/63,$$

como mostrado na linha 1 da Tabela 5.

Também foi sintetizada a relação que deve prevalecer entre o tempo de movimentação do AMV e a distância do ponto onde o trem solicita a movimentação do AMV. O resultado produzido pela ferramenta é apresentado na Figura 20. A variável *pedido* indica a distância,

```
Number of iterations required for reachability: 8
     7tempo_amv + pedido <= 200 & 45tempo_amv + 4pedido >= 800
  |
     14tempo_amv + pedido <= 200 & tempo_amv >= 0
```

Figura 20: Relação entre o ponto de acionamento do AMV e seu tempo de movimentação

a contar do início do circuito de via, onde a movimentação do AMV é solicitada. A variável $tempo\_amv$ indica o intervalo de tempo de operação do AMV. A saída produzida pela ferramenta indica as relações que garantem a alcançabilidade da região de configurações inseguras para a operação deste trecho de malha. Uma vez que a condição $tempo\_amv < 0$ pode ser descartada, a negação da cláusula construída pela ferramenta se reduz a

$$(7tempo\_amv + pedido > 200) \ \vee \ (45tempo\_amv + 4pedido < 800)$$
$$\wedge$$
$$14tempo\_amv + pedido > 200.$$

A cláusula $7tempo\_amv + pedido > 200$ é impossível de se realizar, uma vez que a velocidade do trem está limitada a 7 metros por segundo e a posição do AMV no circuito

de via está fixada em 200 metros. Então, a relação de compromisso necessária para manter o sistema seguro fica reduzida a

$$(14\mathit{tempo\_amv} + \mathit{pedido} > 200) \wedge (45\mathit{tempo\_amv} + 4\mathit{pedido} < 800).$$

4.4.5 Relação entre o tempo de operação do AMV e a distância percorrida pelo trem

Nesse último estudo foi sintetizada a relação entre a distância percorrida pelo trem dentro do circuito de via até o AMV estar posicionado, e o intervalo de tempo que o AMV leva para se posicionar. Numa primeira simulação para relação entre esses dois parâmetros, utilizou-se a marca zero como o ponto onde o trem solicita a movimentação do AMV. Ou seja, o trem solicita a movimentação do AMV logo que ocupa o circuito de via. O resultado produzido pela ferramenta aparece na Figura 21. A variável *dist_trem* indica a distância

```
Number of iterations required for reachability: 10
     dist_trem >= 0 & 4dist_trem <= 45tempo_amv
```

Figura 21: Relação entre o tempo de operação do AMV e da distância percorrida pelo trem

percorrida pelo trem dentro do circuito de via. A variável *tempo_amv* indica o tempo de operação do AMV. A saída produzida pela ferramenta indica a relação que deve vigorar para que a região de configurações inseguras seja alcançada. Negando-se a cláusula construída pela ferramenta e descartando-se a condição impossível $\mathit{dist\_trem} < 0$, obtém-se a condição que deve ser observada para uma operação segura do sistema:

$$\mathit{tempo\_amv} < (4/45) \times \mathit{dist\_trem}.$$

Numa segunda análise, foi obtida a relação entre a distância percorrida pelo trem, o tempo de movimentação do AMV, e a distância do ponto onde o trem solicita o posicionamento do AMV, contada do início do *cv*. O resultado aparece na Figura 22. A relação é

```
Number of iterations required for reachability: 10
     pedido <= dist_trem & 4dist_trem <= 45tempo_amv + 4pedido
```

Figura 22: Tempo de operação do AMV, percurso do trem e ponto de acionamento do AMV

descrita pelas variáveis *dist_trem*, *tempo_amv* e *pedido*. A variável *pedido* indica a distância do ponto onde a movimentação do AMV é solicitada. A variável *tempo_amv* indica o tempo de operação do AMV e a variável *dist_trem* indica a distância percorrida pelo trem até que o AMV esteja posicionado. A saída produzida pela ferramenta indica a relação que deve ser obedecida para que a região de configurações inseguras seja alcançada, no autômato produto que modela este trecho de via. Descartando-se a cláusula $\mathit{dist\_trem} < \mathit{pedido}$, a negação daquela condição resulta em

$$\mathit{tempo\_amv} < (4/45) \times (\mathit{dist\_trem} - \mathit{pedido}).$$

Quando observada, essa condição garante uma operação segura para o trecho de malha modelado.

5 Trabalhos Futuros

Um outro aspecto da operação do sistema que poderia ser analisado usando-se autômatos híbridos é o problema da *sombra*, descrito na seção 3. A idéia seria modelar a estratégia de controle de velocidade dos trens de forma a garantir a operação segura do sistema. Essa estratégia determinaria os códigos de velocidade padrão que seriam distribuídos ao longo da via, e também os códigos de velocidade que seriam atribuídos aos circuitos de via de acordo com a posição dos trens na via. Com isso, deseja-se evitar, por exemplo, que um trem em alta velocidade colida com outro que está à sua frente, trafegando com velocidade mais baixa. Seria interessante sintetizar também os parâmetros utilizados nesse controle de velocidades, determinando os melhores ajustes para os perfis de velocidade padrão, de forma a melhorar a eficiência de operação do sistema, e ainda observando as condições de segurança.

Outro aspecto que poderia ser modelado e analisado é o *headway* do sistema. O *headway* é um parâmetro crítico na operação dos trens na via. Esse parâmetro estabelece uma distância mínima entre trens consecutivos que trafegam numa mesma linha. Existe uma distância mínima entre trens que deve ser observada para que a segurança do sistema seja garantida. Essa distância é uma função da velocidade relativa dos trens e dos seus respectivos perfis de frenagem.

Outro estudo que poderia ser realizado diz respeito à utilização de circuitos de via de comprimento variável. Isso quer dizer que a ocupação de um circuito de via estará agora sujeita à situação de tráfego da via. Caso o tráfego na via seja baixo, a extensão do circuito de via pode ser maior do que a normal. No caso oposto, quando a via suportar um tráfego mais elevado, o circuito de via deve ser curto, condizente com o tráfego naquela região. A idéia é que o desempenho e a eficiência do sistema aumentarão, já que o aproveitamento da via será melhor.

Há ainda outras características do sistema que poderiam ser modeladas. Uma dessas características diz respeito à movimentação dos trens nas estações metroviárias. Os trens devem operar de forma segura dentro das estações, como por exemplo, quando da abertura das portas automáticas para embarque e desembarque de passageiros. Existe também uma preocupação quanto à movimentação das portas dos trens nos locais corretos das estações, ou seja, as portas só devem se movimentar nos pontos demarcados como embarque ou desembarque. Outra preocupação importante diz respeito à movimentação das portas do lado correto do trem. Como os trens trafegam em ambos os sentidos da via eles podem, em um determinado momento, estar de um lado da estação e, em outro momento, podem estar do lado oposto da estação.

Referências

[ACH+94] Rajeev Alur, C. Courcoubetis, N. Halbwachs, T. A. Henzinger, Pei-Hsin Ho, X. Nicollin, A. Olivero, J. Sifakis, e S. Yovine. The algorithmic analysis of hybrid systems. Em G. Cohen e J.-P. Quadrat, editores, *Proceedings of the 11th International Conference on Analysis and Optimization of Discrete Event Systems*, volume 199 de *Lecture Notes in Control and Information Science*, pp. 331–351. Springer-Verlag, 1994.

[AHH93] Rajeev Alur, Thomas A. Henzinger, e Pei-Hsin Ho. Automatic symbolic verification of embedded systems. Em *Proceedings of the 14th Annual IEEE Real-Time Systems Symposium*, pp. 2–11. IEEE Computer Society Press, 1993.

[AHWT97] R. Alur, H. Henzinger, e H. Wong-Toi. Symbolic analysis of hybrid systems. Em *Proceedings of the 36th IEEE Conference on Decision and Control*, pp. 702–707, 1997. Invited survey.

[BG98] Adilson Luiz Bonifácio e Itana Maria de Souza Gimenes. Estudo e comparação de provadores automáticos de teoremas. *Revista Tecnológica*, (7):75–85, Outubro de 1998. Universidade Estadual de Maringá, Brasil.

[BGK+96] J. Bengtsson, W. O. D. Griffioen, K. J. Kristoffersen, K. G. Larsen, F. Larsson, P. Pettersson, e W. Yi. Verification of an audio protocol with bus collision using UPPAAL. Em Rajeev Alur e Thomas A. Henzinger, editores, *Proceedings of the Eighth International Conference on Computer Aided Verification CAV*, volume 1102 de *Lecture Notes in Computer Science*, pp. 244–256, New Brunswick, NJ, USA, julho/agosto de 1996. Springer-Verlag.

[BLL+] Johan Bengtsson, Kim Larsen, Fredrik Larsson, Paul Pettersson, Wang Yi, e Carsten Weise. New generation UPPAAL. Relatório técnico, BRICS, Dept. of Computer Science, Aalborg University, Denmark and Department of Computer Systems, Uppsala University, Sweden. Esse relatório técnico pode ser encontrado no endereço http://www.docs.uu.se/docs/rtmv/uppaal.

[BLL+96a] J. Bengtsson, K. G. Larsen, F. Larsson, P. Pettersson, e Wang Yi. UPPAAL in 1995. *Lecture Notes in Computer Science*, 1055:111–114, 1996.

[BLL+96b] Johan Bengtsson, Kim G. Larsen, Fredrik Larsson, Paul Pettersson, e Wang Yi. UPPAAL — a tool suite for automatic verification of real–time systems. Relatório Técnico RS-96-58, BRICS, Aalborg University, DENMARK and Department of Computer Systems, Uppsala University, Sweden, dezembro de 1996.

[CJAJ98] João Batista Camargo Junior e Jorge Rady Almeida Júnior. Safety analysis case in the São Paulo Metro. Relatório técnico, Digital System and Computer Engineering Department, Polytechnic School - University of São Paulo, São Paulo, Brazil, 1998.

[DKRT] P. R. D'Argenio, J.-P. Katoen, T. C. Ruys, e J. Tretmans. Modeling and verifying a bounded retransmission protocol. Relatório técnico, Faculty of Computing Science, University of Twente, AE Enschede, Netherlands. Esse relatório técnico pode ser encontrado no endereço URL http://www.docs.uu.se/docs/rtmv/uppaal.

[DKRT97] P. R. D'Argenio, J.-P. Katoen, T. C. Ruys, e J. Tretmans. The bounded retransmission protocol must be on time! Em E. Brinksma, editor, *Proceedings of the Third Workshop on Tools and Algorithms for the Construction and Analysis of Systems*, volume 1217 de *Lecture Notes in Computer Science*, pp. 416–431, Enschede, The Netherlands, abril de 1997. Springer-Verlag.

[FAM98] José Henrique Zaccardi de Freitas, Antonio Accurso, e Ivaldo Lopes Mathias. A evolução tecnológica da cmsp e o estado da arte de sistemas de sinalização baseado em comunicação. Em *Revista Engenharia*, volume 529, pp. 116–124, 1998.

[Hen96] Thomas A. Henzinger. The theory of hybrid automata. Em *Proceedings, 11th Annual IEEE Symposium on Logic in Computer Science*, pp. 278–292, New Brunswick, New Jersey, 27-30, julho de 1996.

[HH94] Thomas A. Henzinger e Pei-Hsin Ho. HyTech: The cornell hybrid technology tool. *Workshop on Hybrid Systems and Autonomous Control*, outubro de 1994.

[HHWT95a] Thomas A. Henzinger, Pei-Hsin Ho, e Howard Wong-Toi. HYTECH: the next generation. Em *Proceedings of the 16th IEEE Real-Time Systems Symposium*, pp. 56– 65, Pisa, Italy, 5-7, dezembro de 1995. IEEE Computer Society Press.

[HHWT95b] Thomas A. Henzinger, Pei-Hsin Ho, e Howard Wong-Toi. A user guide to HyTech. Em E. Brinksma, W.R. Cleaveland, K.G. Larsen, T. Margaria, e B. Steffen, editores, *TACAS 95: Proceedings of the First Workshop on Tools and Algorithms for the Construction and Analysis of Systems*, volume 1019 de *Lecture Notes in Computer Science*, pp. 41–71. Springer-Verlag, 1995.

[HHWT97] Thomas A. Henzinger, Pei-Hsin Ho, e Howard Wong-Toi. HYTECH: A model checker for hybrid systems. Em O. Grumberg, editor, *CAV'97: Proceedings of the Ninth International Conference on Computer-Aided Verification*, volume 1254 de *Lecture Notes in Computer Science*, pp. 460–463. Springer-Verlag, 1997.

[Ho95] Pei-Hsin Ho. *Automatic Analysis of Hybrid Systems.* Tese de Doutorado, Cornell University, agosto de 1995.

[HPP94] N. Halbwachs, Y.-E. Proy, e Raymond P. Verification of linear hybrid systems by means of convex approximations. Em *International Symposium on Static Analysis*, volume LNCS 864 de *Lecture Notes in Computer Science*, SAS 1994.

[HSLL97] Klaus Havelund, Arne Skou, Kim Guldstrand Larsen, e Kristian Lund. Formal modeling and analysis of an audio/video protocol: An industrial case study using UPPAAL. Relatório Técnico RS-97-31, BRICS, Aalborg University, Denmark and Bang & Olufsen, Denmark, novembro de 1997.

[HWT95] Pei-Hsin Ho e Howard Wong-Toi. Automated analysis of an audio control protocol. Em P. Wolper, editor, *Proceedings of the 7th International Conference On Computer Aided Verification*, volume 939 de *Lecture Notes in Computer Science*, pp. 381–394, Liege, Belgium, julho de 1995. Springer-Verlag.

[HWT96] Thomas A. Henzinger e Howard Wong-Toi. Using HyTech to synthesize control parameters for a steam boiler. Em J.-R. Abrial, E. Börger, e H. Langmaack, editores, *Formal Methods for Industrial Applications: Specifying and Programming the Steam Boiler Control*, volume 1165 de *Lecture Notes in Control and Information Science*, pp. 265–282. Springer-Verlag, 1996.

[JSGC96] Per Stoffer Jensen, Thomas Mark Sorensen, Jesper Gravgaard, e Palle Klaerke Christensen. *Using AUTOGRAPH to Create Input for HyTech*, setembro de 1996. Esse manual do usuário para o AUTOGRAPH pode ser encontrado no endereço http://www-cad.eecs.berkeley.edu/~tah/hytech/atg_sun/.

[LP] Henrik Lönn e P. Pettersson. Formal verification of a TDMA protocol start-up mechnism. Relatório técnico, Department. of Computer Engineering, Chalmers University of Technology, Gothenburg, Sweden and Department of Computer Systems, Uppsala University, Uppsala, Sweden. Esse relatório técnico pode ser encontrado no endereço http://www.docs.uu.se/docs/rtmv/uppaal.

[LPY] Magnus Lindahl, Paul Pettersson, e Wang Yi. Formal design and analysis of a gear controller: an industrial case study using UPPAAL. Relatório técnico, Mecel AB, Göteborg, Sweden and Department of Computer Systems, Uppsala University, Sweden. Esse relatório técnico pode ser encontrado no endereço http://www.docs.uu.se/docs/rtmv/uppaal.

[LPY96] Kim G. Larsen, Paul Pettersson, e Wang Yi. Diagnostic model-checking for real-time systems. Relatório Técnico RS-96-57, BRICS, Aalborg University, DENMARK and Department of Computer Systems, Uppsala University, Sweden, dezembro de 1996.

[LPY97] Kim G. Larsen, Paul Pettersson, e Wang Yi. UPPAAL in a NUTSHELL. *International Journal on Software Tools for Technology Transfer*, 1(1):134–152, dezembro de 1997.

[Lyn96] Nancy A. Lynch. *Distributed Algorithms*. Morgan Kaufmann, 1996.

[San88] Donald Sannella. A survey of formal software development methods. Relatório Técnico ECS-LFCS-88-56, Laboratory for Foundations of Computer Science, Department of Computer Science, University of Edinburgh, julho de 1988.

[YD] Wang Yi e Mats Daniels. Automatic verification of real time communicating systems by constraint-solving. Relatório técnico, Department of Computer Systems, Uppsala University, Uppsala, Sweden. Esse relatório técnico pode ser encontrado no endereço http://www.docs.uu.se/docs/rtmv/uppaal.

A Os Autômatos do Modelo

```
var aux1, aux2, -- distância do trem
    t3, t4      -- tempo de movimentação do AMV
     : analog;
    k, w        -- controle da ocupação dos circuitos
     : discrete;

-- ------------------------------------------------

automaton Ctrl_Trem_1
synclabs: ocupa1_1, ocupa1_2, ocupa1_3, ocupa1_4, Fechar_AMV3_t1,
          Abrir_AMV3_t1, desocupa1_2, desocupa1_5, desocupa1_4,
          desocupa1_3, desocupa1_1, Fechar_AMV4_t1, Abrir_AMV4_t1,
          ocupa1_5;
initially Ocupar;
loc Perfil_1: while aux1<=200 wait { daux1 in [7,9] }
        when aux1=200 goto Ocupar;
loc Perfil_2: while aux1>=0 wait { daux1 in [-9,-7] }
        when aux1=0 goto Ocupar;
loc Ocupando_1: while True wait { daux1=0 }
        when w<=2 goto Perfil_1;
        when w>=4 goto Perfil_1;
loc Ocupando_2: while True wait { daux1=0 }
        when w<=6 & w>=5 goto Perfil_2;
        when w<=3 goto Perfil_2;
        when w>=9 goto Perfil_2;
loc Ocupando_3: while True wait { daux1=0 }
        when k=9 & w>=5 & w<=6 sync Fechar_AMV3_t1 do { aux1'=200 } goto Perfil_2;
        when k=3 & w=10 sync Abrir_AMV3_t1 do { aux1'=0 } goto Perfil_1;
```

```
        when k=3 & w=2 sync Abrir_AMV3_t1 do { aux1'=0 } goto Perfil_1;
        when k=9 & w<=2 sync Fechar_AMV3_t1 do { aux1'=200 } goto Perfil_2;
loc Ocupando_4: while True wait { daux1=0 }
        when k=8 & w>=5 & w<=6 sync Abrir_AMV4_t1 do { aux1'=200 } goto Perfil_2;
        when k=8 & w=10 sync Abrir_AMV4_t1 do { aux1'=200 } goto Perfil_2;
        when k=7 & w=10 sync Fechar_AMV4_t1 do { aux1'=200 } goto Perfil_2;
        when k=4 & w=10 sync Fechar_AMV4_t1 do { aux1'=0 } goto Perfil_1;
        when k=8 & w=1 sync Abrir_AMV4_t1 do { aux1'=200 } goto Perfil_2;
        when k=4 & w<=2 sync Fechar_AMV4_t1 do { aux1'=0 } goto Perfil_1;
        when k=7 & w<=2 sync Fechar_AMV4_t1 do { aux1'=200 } goto Perfil_2;
loc Ocupar: while True wait { daux1=0 }
        when k=4 & asap sync desocupa1_4 do { k'=5 } goto Ocupar;
        when k=3 & asap sync desocupa1_3 do { k'=4 } goto Ocupar;
        when k=6 sync ocupa1_5 do { aux1'=200 } goto Ocupando_5;
        when k=5 sync ocupa1_5 do { aux1'=0 } goto Ocupando_5;
        when k=7 & asap sync desocupa1_4 do { k'=9 } goto Ocupar;
        when k=8 sync ocupa1_4 goto Ocupando_4;
        when k=1 & asap sync desocupa1_1 do { k'=3 } goto Ocupar;
        when k=9 & asap sync desocupa1_3 do { k'<=2 } goto Ocupar;
        when k=8 & asap sync desocupa1_4 do { k'=9 } goto Ocupar;
        when k=6 & asap sync desocupa1_5 do { k'=7 } goto Ocupar;
        when k=5 & asap sync desocupa1_5 do { k'=6 } goto Ocupar;
        when k=2 & asap sync desocupa1_2 do { k'=8 } goto Ocupar;
        when k=7 sync ocupa1_4 goto Ocupando_4;
        when k=9 sync ocupa1_3 goto Ocupando_3;
        when k=4 sync ocupa1_4 goto Ocupando_4;
        when k=3 sync ocupa1_3 goto Ocupando_3;
        when k=2 sync ocupa1_2 do { aux1'=200 } goto Ocupando_2;
        when k=1 sync ocupa1_1 do { aux1'=0 } goto Ocupando_1;
loc Ocupando_5: while True wait { daux1=0 }
        when k=6 & w<=2 do { aux1'=200 } goto Perfil_2;
        when k=5 & w<=2 do { aux1'=0 } goto Perfil_1;
        when k=5 & w>=8 do { aux1'=0 } goto Perfil_1;
        when k=6 & w>=8 do { aux1'=200 } goto Perfil_2;
end -- Ctrl_Trem_1

automaton Ctrl_Trem_2
synclabs: Abrir_AMV4_t2, Fechar_AMV4_t2, desocupa2_4, ocupa2_4,
          desocupa2_1, desocupa2_5, desocupa2_2, ocupa2_3,
          ocupa2_2, ocupa2_1, Abrir_AMV3_t2, Fechar_AMV3_t2,
          Perfil_1, desocupa2_3, ocupa2_5;
initially Ocupar;
loc Perfil_2: while aux2>=0 wait { daux2 in [-9,-7] }
        when aux2=0 goto Ocupar;
loc Ocupando_4: while True wait { daux2=0 }
        when w=8 & k>=5 & k<=6 sync Abrir_AMV4_t2 do { aux2'=200 } goto Perfil_2;
        when w=8 & k=10 sync Abrir_AMV4_t2 do { aux2'=200 } goto Perfil_2;
        when w=7 & k=10 sync Fechar_AMV4_t2 do { aux2'=200 } goto Perfil_2;
        when w=4 & k=10 sync Fechar_AMV4_t2 do { aux2'=0 } goto Perfil_1;
        when w=7 & k<=2 sync Fechar_AMV4_t2 do { aux2'=200 } goto Perfil_2;
```

```
        when w=4 & k<=2 sync Fechar_AMV4_t2 do { aux2'=0 } goto Perfil_1;
        when w=8 & k=1 sync Abrir_AMV4_t2 do { aux2'=200 } goto Perfil_2;
loc Ocupar: while True wait { daux2=0 }
        when w=5 sync ocupa2_5 do { aux2'=0 } goto Ocupando_5;
        when w=6 sync ocupa2_5 do { aux2'=200 } goto Ocupando_5;
        when w=9 & asap sync desocupa2_3 do { w'<=2 } goto Ocupar;
        when w=4 & asap sync desocupa2_4 do { w'=5 } goto Ocupar;
        when w=3 & asap sync desocupa2_3 do { w'=4 } goto Ocupar;
        when w=1 sync ocupa2_1 do { aux2'=0 } goto Ocupando_1;
        when w=2 sync ocupa2_2 do { aux2'=200 } goto Ocupando_2;
        when w=3 sync ocupa2_3 goto Ocupando_3;
        when w=4 sync ocupa2_4 goto Ocupando_4;
        when w=9 sync ocupa2_3 goto Ocupando_3;
        when w=7 sync ocupa2_4 goto Ocupando_4;
        when w=2 & asap sync desocupa2_2 do { w'=8 } goto Ocupar;
        when w=5 & asap sync desocupa2_5 do { w'=6 } goto Ocupar;
        when w=6 & asap sync desocupa2_5 do { w'=7 } goto Ocupar;
        when w=8 & asap sync desocupa2_4 do { w'=9 } goto Ocupar;
        when w=1 & asap sync desocupa2_1 do { w'=3 } goto Ocupar;
        when w=8 sync ocupa2_4 goto Ocupando_4;
        when w=7 & asap sync desocupa2_4 do { w'=9 } goto Ocupar;

loc Ocupando_3: while True wait { daux2=0 }
        when w=9 & k>=5 & k<=6 sync Fechar_AMV3_t2 do { aux2'=200 } goto Perfil_2;
        when w=3 & k=10 sync Abrir_AMV3_t2 do { aux2'=0 } goto Perfil_1;
        when w=9 & k<=2 sync Fechar_AMV3_t2 do { aux2'=200 } goto Perfil_2;
        when w=3 & k=2 sync Abrir_AMV3_t2 do { aux2'=0 } goto Perfil_1;
loc Perfil_1: while aux2<=200 wait { daux2 in [7,9] }
        when aux2=200 sync Perfil_1 goto Ocupar;
loc Ocupando_2: while True wait { daux2=0 }
        when k<=6 & k>=5 goto Perfil_2;
        when k<=3 goto Perfil_2;
        when k>=9 goto Perfil_2;
loc Ocupando_1: while True wait { daux2=0 }
        when k<=2 goto Perfil_1;
        when k>=4 goto Perfil_1;
loc Ocupando_5: while True wait { daux2=0 }
        when w=6 & k<=2 do { aux2'=200 } goto Perfil_2;
        when w=6 & k>=8 do { aux2'=200 } goto Perfil_2;
        when w=5 & k>=8 do { aux2'=0 } goto Perfil_1;
        when w=5 & k<=2 do { aux2'=0 } goto Perfil_1;
end -- Ctrl_Trem_2

automaton AMV_3
synclabs: Abrir_AMV3_t2, Abrir_AMV3_t1, Fechar_AMV3_t2, Fechar_AMV3_t1;
initially Normal;
loc Reverso: while True wait { dt3=0 }
        when True sync Fechar_AMV3_t1 do { t3'=0 } goto Fechando;
        when True sync Fechar_AMV3_t2 do { t3'=0 } goto Fechando;
        when True sync Abrir_AMV3_t1 goto Reverso;
```

```
        when True sync Abrir_AMV3_t2 goto Reverso;
loc Normal: while True wait { dt3=0 }
        when True sync Abrir_AMV3_t1 do { t3'=0 } goto Abrindo;
        when True sync Abrir_AMV3_t2 do { t3'=0 } goto Abrindo;
        when True sync Fechar_AMV3_t1 goto Normal;
        when True sync Fechar_AMV3_t2 goto Normal;
loc Abrindo: while t3<=15 wait { dt3 in [4/5,1] }
        when t3=15 goto Reverso;
loc Fechando: while t3<=15 wait { dt3 in [4/5,1] }
        when t3=15 goto Normal;
end -- AMV_3

automaton AMV_4
synclabs: Abrir_AMV4_t2, Fechar_AMV4_t1, Fechar_AMV4_t2, Abrir_AMV4_t1;
initially Normal;
loc Reverso: while True wait { dt4=0 }
        when True sync Fechar_AMV4_t2 do { t4'=0 } goto Fechando;
        when True sync Fechar_AMV4_t1 do { t4'=0 } goto Fechando;
        when True sync Abrir_AMV4_t1 goto Reverso;
        when True sync Abrir_AMV4_t2 goto Reverso;
loc Normal: while True wait { dt4=0 }
        when True sync Abrir_AMV4_t1 do { t4'=0 } goto Abrindo;
        when True sync Abrir_AMV4_t2 do { t4'=0 } goto Abrindo;
        when True sync Fechar_AMV4_t2 goto Normal;
        when True sync Fechar_AMV4_t1 goto Normal;
loc Abrindo: while t4<=15 wait { dt4 in [4/5,1] }
        when t4=15 goto Reverso;
loc Fechando: while t4<=15 wait { dt4 in [4/5,1] }
        when t4=15 goto Normal;
end -- AMV_4

automaton CV_1
synclabs: ocupa2_1, ocupa1_1, desocupa2_1, desocupa1_1;
initially Desocupa;
loc Desocupa: while True wait {  }
        when True sync ocupa1_1 goto Ocupa;
        when True sync ocupa2_1 goto Ocupa;
loc Ocupa: while True wait {  }
        when True sync desocupa1_1 goto Desocupa;
        when True sync desocupa2_1 goto Desocupa;
end -- CV_1

automaton CV_2
synclabs: ocupa2_2, ocupa1_2, desocupa2_2, desocupa1_2;
initially Desocupa;
loc Desocupa: while True wait {  }
        when True sync ocupa1_2 goto Ocupa;
        when True sync ocupa2_2 goto Ocupa;
loc Ocupa: while True wait {  }
        when True sync desocupa1_2 goto Desocupa;
```

```
        when True sync desocupa2_2 goto Desocupa;
end -- CV_2

automaton CV_3
synclabs: ocupa2_3, ocupa1_3, desocupa2_3, desocupa1_3;
initially Desocupa;
loc Desocupa: while True wait {  }
        when True sync ocupa1_3 goto Ocupa;
        when True sync ocupa2_3 goto Ocupa;
loc Ocupa: while True wait {  }
        when True sync desocupa1_3 goto Desocupa;
        when True sync desocupa2_3 goto Desocupa;
end -- CV_3

automaton CV_4
synclabs: ocupa2_4, ocupa1_4, desocupa2_4, desocupa1_4;
initially Desocupa;
loc Desocupa: while True wait {  }
        when True sync ocupa1_4 goto Ocupa;
        when True sync ocupa2_4 goto Ocupa;
loc Ocupa: while True wait {  }
        when True sync desocupa1_4 goto Desocupa;
        when True sync desocupa2_4 goto Desocupa;
end -- CV_4

automaton CV_5
synclabs: ocupa1_5, ocupa2_5, desocupa1_5, desocupa2_5;
initially Desocupa;
loc Desocupa: while True wait {  }
        when True sync ocupa2_5 goto Ocupa;
        when True sync ocupa1_5 goto Ocupa;
loc Ocupa: while True wait {  }
        when True sync desocupa2_5 goto Desocupa;
        when True sync desocupa1_5 goto Desocupa;
end -- CV_5
```

B Verificação de Propriedades

B.1 Posicionamento entre Trens: Usando-se Predicados *Unsafe*

B.1.1 Análise: O trem no circuito cv_5 e outro trem no circuito cv_3 ou no circuito cv_4, no sentido padrão da via

```
var
   final_reg, init_reg, reached: region;
init_reg:=
     loc[Ctrl_Trem_1] = Ocupar &
     loc[Ctrl_Trem_2] = Ocupar &
```

```
     k=1 & w=1;
final_reg :=
      loc[Ctrl_Trem_1] = Perfil_1 &
      (loc[Ctrl_Trem_2] = Perfil_1 |
      loc[Ctrl_Trem_2] = Perfil_2) &
      k=5 &
      (w>=3 & w<=7);
reached:=
      reach forward from init_reg endreach;
if empty (reached & final_reg)
   then prints "Trem esta seguro";
   else prints "Trem violou a seguranca";
endif;
```

B.1.2 Resultado

```
Number of iterations required for reachability: 61
Trem esta seguro

============================================================
Max memory used =     0 pages =        0 bytes =   0.00 MB
Time spent      =     266.99u +     58.44s =     325.43 sec total
============================================================
```

B.2 Posicionamento entre Trens: Usando-se Predicados *Safe*

B.2.1 Análise: O trem no circuito cv_5 e outro trem no circuito cv_3 no sentido contrário da via

```
var
   final_reg, init_reg, reached: region;
init_reg:=
     loc[Ctrl_Trem_1] = Ocupar &
     loc[Ctrl_Trem_2] = Ocupar &
     k=1 & w=1;
final_reg :=
      loc[Ctrl_Trem_1] = Perfil_2 &
      (loc[Ctrl_Trem_2] = Perfil_1 |
       loc[Ctrl_Trem_2] = Perfil_2) &
      k=9 &
      (w=5 | w=6);
reached:=
      reach forward from init_reg endreach;
if empty (reached & final_reg)
   then prints "Trem esta seguro";
   else prints "Trem violou a seguranca";
```

```
endif;
```

B.2.2 Resultado

```
Number of iterations required for reachability: 61
Trem violou a seguranca

===============================================================
Max memory used =     0 pages =        0 bytes =   0.00 MB
Time spent      =     214.01u +     53.82s =     267.83 sec total
===============================================================
```

B.3 Posicionamento dos Trens em Relação aos AMVs: Usando-se Predicados *Unsafe*

B.3.1 Análise: O trem vindo do circuito cv_2 e o AMV_4 está em reverso após 180 metros

```
var
   final_reg, init_reg, reached: region;
init_reg:=
     loc[Ctrl_Trem_1] = Ocupar &
     loc[Ctrl_Trem_2] = Ocupar &
     loc[AMV_3] = Normal &
     loc[AMV_4] = Normal &
     k=1 & w=1 & t3=0 & t4=0;
final_reg :=
      loc[Ctrl_Trem_1] = Perfil_2 &
      k=8 & aux1<=20 &
      (loc[AMV_4] = Normal | loc[AMV_4] = Abrindo |
       loc[AMV_4] = Fechando);
reached:=
      reach forward from init_reg endreach;
if empty(reached & final_reg)
   then prints "O trem passa com seguranca pelo AMV";
   else prints "O AMV pode nao estar na posicao correta";
endif;
```

B.3.2 Resultado

```
Number of iterations required for reachability: 82
O trem passa com seguranca pelo AMV

===============================================================
```

```
Max memory used =     0 pages =        0 bytes =   0.00 MB
Time spent      =     126.90u +     33.05s =     159.95 sec total
============================================================
```

B.4 Posicionamento dos Trens em Relação aos AMVs: Usando-se Predicados *Safe*

B.4.1 Análise: O trem vindo do circuito cv_4 e o AMV_3 não está em normal após 160 metros

```
var
   final_reg, init_reg, reached: region;
init_reg:=
     loc[Ctrl_Trem_1] = Ocupar &
     loc[Ctrl_Trem_2] = Ocupar &
     loc[AMV_3] = Normal &
     loc[AMV_4] = Normal &
     k=1 & w=1 & t3=0 & t4=0;
final_reg :=
      loc[Ctrl_Trem_1] = Perfil_2 &
      k=9 & aux1<=40 &
      (loc[AMV_3] = Reverso | loc[AMV_3] = Abrindo |
       loc[AMV_3] = Fechando);
reached:=
      reach forward from init_reg endreach;
if empty(reached & final_reg)
   then prints "O trem passa com seguranca pelo AMV";
   else prints "O AMV pode nao estar na posicao correta";
endif;
```

B.4.2 Resultado

```
Number of iterations required for reachability: 82
O AMV pode nao estar na posicao correta

============================================================
Max memory used =     0 pages =        0 bytes =   0.00 MB
Time spent      =     127.78u +     32.47s =     160.25 sec total
============================================================
```

C Síntese de Parâmetros

C.1 Parametrizando a distância segura do trem

C.1.1 Análise

```
var
 final_reg, init_reg, reached: region;
init_reg:=
     loc[Ctrl_Trem_1] = Ocupar &
     loc[AMV_3] = Normal &
     t3=0 & aux1=0;
final_reg :=
      loc[Ctrl_Trem_1] = Ocupar &
      aux1=dist_trem &
      (loc[AMV_3] = Normal | loc[AMV_3] = Abrindo |
      loc[AMV_3] = Fechando);
reached:=
      reach forward from init_reg endreach;
print omit all locations
      hide non_parameters in
           reached & final_reg
      endhide;
```

C.1.2 Resultado

```
Number of iterations required for reachability: 10
     dist_trem >= 0   & 4dist_trem <= 675
```

C.2 Parametrizando a distância do trem e a solicitação de movimentação do amv

C.2.1 Análise

```
var
 final_reg, init_reg, reached: region;
init_reg:=
     loc[Ctrl_Trem_1] = Ocupar &
     loc[AMV_3] = Normal &
     t3=0 & aux1=pedido;
final_reg :=
      loc[Ctrl_Trem_1] = Ocupar &
      aux1=dist_trem &
      (loc[AMV_3] = Normal | loc[AMV_3] = Abrindo |
      loc[AMV_3] = Fechando);
reached:=
```

```
      reach forward from init_reg endreach;
print omit all locations
      hide non_parameters in
           reached & final_reg
      endhide;
```

C.2.2 Resultado

```
Number of iterations required for reachability: 10
     pedido <= dist_trem   & 4dist_trem <= 4pedido + 675
```

C.3 Parametrizando a operação do amv

C.3.1 Análise

```
var
   final_reg, init_reg, reached: region;
init_reg:=
     loc[Ctrl_Trem_1] = Ocupar &
     loc[AMV_3] = Normal &
     t3=0 & aux1=0;
final_reg :=
      loc[Ctrl_Trem_1] = Perfil_1 &
      aux1=200 &
      t3=tempo_amv &
      (loc[AMV_3] = Normal | loc[AMV_3] = Abrindo |
      loc[AMV_3] = Fechando);
reached:=
      reach forward from init_reg endreach;
print omit all locations
      hide non_parameters in
           reached & final_reg
      endhide;
```

C.3.2 Resultado

```
Number of iterations required for reachability: 7
     7tempo_amv <= 200   & 63tempo_amv >= 1120
  |
     tempo_amv >= 0   & 7tempo_amv <= 100
```

C.4 Parametrizando a operação do amv e a solicitação de movimentação deste

C.4.1 Análise

```
var
   final_reg, init_reg, reached: region;
init_reg:=
     loc[Ctrl_Trem_1] = Ocupar &
     loc[AMV_3] = Normal &
     t3=0 & aux1=pedido;
final_reg :=
      loc[Ctrl_Trem_1] = Perfil_1 &
      aux1=200 &
      t3=tempo_amv &
      (loc[AMV_3] = Normal | loc[AMV_3] = Abrindo |
      loc[AMV_3] = Fechando);
reached:=
      reach forward from init_reg endreach;
print omit all locations
      hide non_parameters in
           reached & final_reg
      endhide;
```

C.4.2 Resultado

```
Number of iterations required for reachability: 8
     7tempo_amv + pedido <= 200   & 45tempo_amv + 4pedido >= 800
  |
     14tempo_amv + pedido <= 200   & tempo_amv >= 0
```

C.5 Parametrizando a distância segura do trem e a operação do amv

C.5.1 Análise

```
var
   final_reg, init_reg, reached: region;
init_reg:=
     loc[Ctrl_Trem_1] = Ocupar &
     loc[AMV_3] = Normal &
     t3=0 & aux1=0;
final_reg :=
      loc[Ctrl_Trem_1] = Ocupar &
      aux1=dist_trem &
      t3=tempo_amv &
      (loc[AMV_3] = Normal | loc[AMV_3] = Abrindo |
```

```
      loc[AMV_3] = Fechando);
reached:=
      reach forward from init_reg endreach;
print omit all locations
      hide non_parameters in
           reached & final_reg
      endhide;
```

C.5.2 Resultado

```
Number of iterations required for reachability: 10
     dist_trem >= 0   & 4dist_trem <= 45tempo_amv
```

C.6 Parametrizando a distância segura do trem e a operação do amv, bem como a solicitação de movimentação do amv

C.6.1 Análise

```
var
   final_reg, init_reg, reached: region;
init_reg:=
     loc[Ctrl_Trem_1] = Ocupar &
     loc[AMV_3] = Normal &
     t3=0 & aux1=pedido;
final_reg :=
      loc[Ctrl_Trem_1] = Ocupar &
      aux1=dist_trem &
      t3=tempo_amv &
      (loc[AMV_3] = Normal | loc[AMV_3] = Abrindo |
      loc[AMV_3] = Fechando);
reached:=
      reach forward from init_reg endreach;
print omit all locations
      hide non_parameters in
           reached & final_reg
      endhide;
```

C.6.2 Resultado

```
Number of iterations required for reachability: 10
     pedido <= dist_trem   & 4dist_trem <= 45tempo_amv + 4pedido
```