



Universidade Estadual de Campinas  
Faculdade de Engenharia Elétrica e Computação  
Departamento de Engenharia de Computação e Automação Industrial

# **Estratégias de Decisão para o Planejamento de Circulação de Trens em Tempo Real**

**Autor: Alexandre Tazoniero**

**Orientador: Prof. Dr. Fernando Antonio Campos Gomide**

**Dissertação de Mestrado** apresentada à Faculdade de Engenharia Elétrica e de Computação da Universidade Estadual de Campinas como parte dos requisitos para obtenção do título de Mestre em Engenharia Elétrica, área de concentração Automação

## **Banca Examindora**

Prof. Dr. Fernando Antonio Campos Gomide..... DCA/FEEC/Unicamp  
Prof. Dr. José Ricardo Pelaquim Mendes..... DEP/FEM/Unicamp  
Prof. Dr. Takaaki Ohishi..... DENSIS/FEEC/Unicamp  
Dr. Rodrigo Almeida Gonçalves..... CFlex/Campinas/SP

Campinas, SP

Junho/2007

FICHA CATALOGRÁFICA ELABORADA PELA  
BIBLIOTECA DA ÁREA DE ENGENHARIA E ARQUITETURA - BAE -  
UNICAMP

T219e      Tazoniero, Alexandre  
Estratégias de decisão para o planejamento de circulação  
de trens em tempo real. / Alexandre Tazoniero. --Campinas,  
SP: [s.n.], 2007.

Orientador: Fernando Antonio Campos Gomide  
Dissertação (Mestrado) - Universidade Estadual de  
Campinas, Faculdade de Engenharia Elétrica e de  
Computação.

1. Ferrovias – Tráfego. 2. Conjuntos difusos. 3.  
Heurística. I. Gomide, Fernando Antonio Campos. II.  
Universidade Estadual de Campinas. Faculdade de  
Engenharia Elétrica e de Computação. III. Título.

Título em Inglês: Decision making strategies for real-time trains movement  
planning

Palavras-chave em Inglês: Trains movement planning, Train dispatch, Fuzzy  
control, Heuristic search, Heuristics

Área de concentração: Automação

Titulação: Mestre em Engenharia Elétrica

Banca examinadora: José Ricardo Pelaquim Mendes, Takaaki Ohishi e  
Rodrigo Almeida Gonçalves

Data da defesa: 29/06/2007

Programa de Pós-Graduação: Engenharia Elétrica

## COMISSÃO JULGADORA - TESE DE MESTRADO

Candidato: Alexandre Tazoniero

Data da Defesa: 29 de junho de 2007

Título da Tese: "Estratégias de Decisão para o Planejamento de Circulação de Trens em Tempo Real"

Prof. Dr. Fernando Antônio Campos Gomide (Presidente): \_\_\_\_\_

Dr. Rodrigo Almeida Gonçalves: \_\_\_\_\_

Prof. Dr. José Ricardo Pelaquim Mendes: \_\_\_\_\_

Prof. Dr. Takaaki Ohishi: \_\_\_\_\_

## Resumo

O transporte ferroviário tem grande participação no transporte de cargas e passageiros em todo o mundo. No Brasil a malha ferroviária sofreu um processo de abandono e deterioração no período de 1960 a 1990. A partir de 1990 a privatização da rede ferroviária nacional iniciou uma retomada de investimentos e nos últimos anos a demanda por transporte ferroviário vem crescendo significativamente. É necessário, então, que os recursos da ferrovia sejam utilizados de maneira eficiente para atender a crescente demanda, o que exige planejamento estratégico, tático e operacional. No nível operacional uma das principais etapas e também uma das mais carentes de ferramentas computacionais é o Planejamento de Circulação de trens.

O processo operacional de uma ferrovia é dinâmico, sujeito a inúmeras interferências imprevisíveis e uma ferramenta computacional para o apoio ao planejamento de circulação de trens deve fornecer soluções com tempo de processamento compatível com essa realidade.

Este trabalho propõe algoritmos para o planejamento de circulação de trens em tempo real, utilizando metodologias de inteligência computacional e conjuntos nebulosos. Um algoritmo objetiva decidir localmente a preferência entre trens concorrendo pelo uso de um segmento de linha singela de modo a seguir uma referência de percurso fornecida por algum algoritmo de otimização ou por um especialista. Outro algoritmo decide, além da preferência entre trens, a velocidade de percurso dos trens para manter-los o mais próximo possível de suas referências. O terceiro algoritmo usa elementos de busca em árvore para obter uma solução para o planejamento de circulação de trens. É feito um estudo comparativo dos algoritmos aqui propostos e de algoritmos existentes na literatura. O estudo comparativo é feito a partir de instâncias pequenas de problema de planejamento de circulação e uma instância que considera dados reais de uma ferrovia brasileira. Os resultados mostram que os algoritmos propostos obtêm soluções próximas às ótimas para as instâncias pequenas e soluções satisfatórias para o caso real.

**Palavras-Chave:** planejamento de circulação de trens, despacho de trens, controle nebuloso, busca, heurística.

# Abstract

Railways plays a major role in freight and passenger transportation in the whole world. The Brazilian railway system has suffered a process of abandon and deterioration from 1960 to 1990. Since 1990 the privatization of the national railways brought new investments and in the last years the demand for railway transportation has increased significantly. Railway resources must be efficiently managed to match the increasingly transportation demand. This requires efficient strategic, tactical and operational planning. One of the main tasks at the operational planning level concerns train circulation and associated tools.

Railway operation is a very dynamic process because trains are subject to many unexpected interferences. Computational tools to help trains circulation planning must provide solutions in a time range consistent with real-time needs.

This work suggests algorithms for real-time train movement planning, using computational intelligence and fuzzy set theory methodology. One of the algorithms decides the preference between trains competing for a single line track at the same moment. The aim is to drive train circulation as close as possible to reference trajectories supplied by human experts, global optimization algorithms or both. Other algorithm decides preference between trains and chose the velocity with which trains must travel to remain as close as possible to its references. The third algorithm uses depth search algorithm to obtain a solution for train circulation problems. A comparative study considering the algorithms proposed herein and algorithms suggested in the literature. The comparative study is done using small railway system instances. Data of a major Brazilian railway is adopted to illustrate how the algorithms behave to solve larger instances. Results show that the algorithms here proposed obtain near optimal solutions for small instances and satisfactory solutions for the real case;

**Keywords:** Trains movement planning, train dispatch, fuzzy control, heuristic search, heuristics.

## **Agradecimentos**

À minha família pelo amor, apoio e carinho durante toda minha vida.

À Vania, por me dar no dia-a-dia deste trabalho força, amor, dedicação e paciência.

Ao Prof. Gomide pela excelente orientação e dedicação.

Aos amigos Alexander do Valle e Francisco Osvaldo Mendes Mota Filho pelas valiosas contribuições.

Aos amigos de Curitiba, da graduação, e aos colegas da CFlex pela amizade.

À CNPq pelo apoio financeiro.

# Sumário

<b>Lista de Figuras</b> .....	xi
<b>Lista de Tabelas</b> .....	xiii
<b>Lista de Algoritmos</b> .....	xiv
<b>Notação</b> .....	xv
<b>1 Introdução</b> .....	1
1.1 Descrição do Problema .....	4
1.2 Motivação .....	5
1.3 Objetivos .....	5
1.4 Organização do Trabalho .....	6
<b>2 Planejamento em Circulação de Trens</b> .....	7
2.1 Definição do Problema .....	7
2.2 Revisão Bibliográfica .....	12
2.3 Modelo de Ferrovia .....	15
2.3.1 Configuração da linha ferroviária. ....	15
2.3.2 Atividades de manutenção. ....	16
2.3.3 Descrição de trens .....	17
2.3.4 Lógica do simulador .....	18
2.3.5 Especificação algébrica do modelo. ....	18
2.3.6 Rotina de despacho .....	20
2.4 Modelo de Programação Linear Inteira-Mista. ....	24
2.5 Algoritmo Genético .....	28
2.6 Resumo .....	30
<b>3 Algoritmos Nebulosos de Decisão.</b> .....	31
3.1 Introdução .....	31
3.2 Algoritmos Heurísticos de Decisão de Preferência entre Trens. ....	32
3.3 Decisão de Preferência de Trens .....	34
3.3.1 Princípios do algoritmo. ....	34
3.3.2 Modelagem com a teoria de conjuntos nebulosos. ....	35
3.3.3 Base de regras para decisão de preferência. ....	38
3.3.4 Decisão. ....	40
3.3.5 Algoritmo de decisão de preferência de trens. ....	43
3.4 Algoritmo de Decisão de Velocidade de Trens .....	46

3.4.1	Princípios do algoritmo. . . . .	46
3.4.2	Modelagem com a teoria de conjuntos nebulosos. . . . .	46
3.4.3	Base de regras para decisão de velocidade de trem . . . . .	48
3.4.4	Defuzzificação. . . . .	48
3.4.5	Algoritmo de decisão de velocidade de trens . . . . .	51
3.5	Exemplos . . . . .	54
3.6	Resumo . . . . .	58
<b>4</b>	<b>Algoritmo de Busca em Árvore . . . . .</b>	<b>59</b>
4.1	Introdução . . . . .	59
4.2	Busca em Árvore no Planejamento de Circulação De Trens . . . . .	60
4.2.1	Algoritmos de busca em problemas combinatoriais . . . . .	60
4.2.2	Algoritmos tradicionais de busca em árvore . . . . .	61
4.2.2.1	Algoritmos não-informados . . . . .	61
4.2.2.2	Algoritmos informados . . . . .	65
4.2.3	Representação em árvore do problema de planejamento de circulação de trens. . . . .	68
4.2.4	Referências de busca em árvore em circulação de trens. . . . .	69
4.3	Algoritmo <i>Best-First</i> para Circulação de Trens . . . . .	71
4.3.1	Princípios do algoritmo. . . . .	71
4.3.2	Algoritmo de busca para planejamento de circulação de trens. . . . .	72
4.3.3	Otimidade do algoritmo. . . . .	75
4.4	Resumo . . . . .	76
<b>5</b>	<b>Resultados e Estudo Comparativo . . . . .</b>	<b>77</b>
5.1	Introdução . . . . .	77
5.2	Resultados para Instâncias Pequenas . . . . .	78
5.2.1	Resultados com modelo ferroviário 1 . . . . .	80
5.2.2	Resultados com modelo ferroviário 2 . . . . .	88
5.2.3	Resultados com flexibilização do PAT . . . . .	95
5.2.4	Resultados com perturbações no percurso dos trens . . . . .	99
5.2.5	Resultado com heurística do menor tempo de cruzamento . . . . .	104
5.3	Resultados para Instância Real. . . . .	108
5.4	Análise dos Resultados . . . . .	113
5.5	Resumo . . . . .	116
<b>6</b>	<b>Conclusão . . . . .</b>	<b>119</b>
	<b>Referências . . . . .</b>	<b>123</b>
	<b>Apêndice A . . . . .</b>	<b>127</b>
	<b>Apêndice B . . . . .</b>	<b>131</b>



# Lista de Figuras

1.1 Evolução das Ferrovias no Brasil (fonte ANTF) . . . . .	1
1.2 Níveis de planejamento ferroviário . . . . .	2
2.1 Exemplo de configuração de uma linha ferroviária e circulação de trens . . . . .	8
2.2 Exemplo de cruzamento de trens . . . . .	8
2.3 Exemplo de ultrapassagem de trens . . . . .	9
2.4 Exemplo de ocupação indevida . . . . .	9
2.5 Exemplo de ocupação correta . . . . .	10
2.6 Exemplo de situação de bloqueio . . . . .	10
2.7 Exemplo de ordenação de circulação para evitar uma situação de bloqueio . . . . .	10
2.8 Representação de solução de planejamento de circulação de trens através de gráfico de trens . . . . .	11
2.9 Exemplo de linha ferroviária . . . . .	15
2.10 Linha ferroviária com pátios de cruzamento e segmentos singelos . . . . .	25
2.11 Algoritmo genético e o simulador . . . . .	28
3.1: Limite inferior e superior para a referência de percurso de um trem . . . . .	35
3.2: Representação dos valores associados à variável linguística Percurso . . . . .	36
3.3: Exemplo de conjuntos nebulosos para a posição do trem . . . . .	36
3.4: Classificação da posição atual do trem nos Conjuntos Nebulosos . . . . .	37
3.5: Representação da base de Regras para determinação do trem preferencial . . . . .	39
3.6: Representação da das regras e inferência nebulosa para a preferência de trens . . . . .	41
3.7: Representação da decisão para a preferência do trem . . . . .	42
3.8: Fluxograma do processo de simulação e seleção do trem preferencial . . . . .	44
3.9: Estrutura do sistema . . . . .	45
3.10: Representação dos Conjuntos Nebulosos para a velocidade do trem . . . . .	47
3.11: Representação das regras e inferência nebulosa para a velocidade do trem . . . . .	49
3.12: Representação da decisão para a velocidade do trem . . . . .	50
3.13: Tronco de triângulo . . . . .	51
3.14: Defuzzificação da velocidade do trem . . . . .	51
3.15: Fluxograma do processo de simulação, seleção do trem preferencial decisão da velocidade do trem . . . . .	53
3.16: Estrutura do sistema . . . . .	54
3.17: Referência gerada pelo Modelo de Programação Linear Inteira Mista e solução gerada pelo algoritmo de decisão de preferência de trens como critério 1 . . . . .	56
3.18: Solução gerada pelo algoritmo de algoritmo de decisão de preferência com o critério 2 . . . . .	56
3.19: Solução obtida pelo algoritmo de decisão de preferência com o critério 2 e pelo algoritmo de decisão de velocidade de trens . . . . .	57
4.1: Representação em árvore das soluções de um pequeno problema combinatorial . . . . .	60
4.2: Exemplo de busca em profundidade . . . . .	63
4.3: Exemplo de busca em largura . . . . .	63
4.4: Representação em árvore das soluções de problemas de circulação de trens . . . . .	69
4.5: Fluxograma do algoritmo de busca em árvore . . . . .	74
4.6: Estrutura do sistema com algoritmo de busca em árvore . . . . .	75
5.1: Variação do tempo de paradas (minutos) com o número de trens para o Cenário 1 . . . . .	82
5.2: Variação do tempo de paradas (minutos) com o número de trens para o Cenário 2 . . . . .	84
5.3: Solução obtida pelo PLIM (esquerda) e solução obtida pelo $APV_{PAT}$ (direita) para o Cenário 2 . . . . .	85
5.4: Variação do tempo de paradas (minutos) com o número de trens para o Cenário3 . . . . .	87
5.5: Variação do tempo de paradas (minutos) com o número de trens para o Cenário4 . . . . .	90
5.6: Variação do tempo de paradas (minutos) com o número de trens para o Cenário5 . . . . .	92
5.7: Solução obtida pelo PLMI (esquerda) e solução obtida pelo BA (direita) para o Cenário 5 . . . . .	92
5.8: Variação do tempo de paradas (minutos) com o número de trens para o Cenário 6 . . . . .	94

5.9: Flexibilização do PAT .....	96
5.10: Solução ótima .....	98
5.11: Solução obtida com a flexibilização do PAT .....	98
5.12: a. Solução do $AP_{PLIM}$ . b. Solução do $APV_{PLIM}$ .....	101
5.13: a. Solução do $AP_{PAT}$ . b. Solução do $APV_{PAT}$ .....	102
5.14: Solução ótima considerando perturbações na circulação dos trens .....	104
5.15: Solução do PLIM (esquerda) e solução do $AP_{PAT}$ (direita). ....	106
5.16: Solução do PLIM (esquerda) e solução do AHMT (direita). ....	108
5.17: Tempos de parada para cruzamento e ultrapassagem .....	111
5.18: Tempos de parada para cruzamento e ultrapassagem para o algoritmo BA .....	112
5.19: Tempos de processamento para o algoritmo BA .....	113
5.20: Solução produzida pelo algoritmo BA .....	114

# Lista de Tabelas

3.1: Base de Regras para determinação do trem preferencial . . . . .	39
3.2: Instantes de partida . . . . .	55
4.1: Análise de complexidade dos algoritmos de busca . . . . .	67
5.1: Instantes de partida de cada trem para o Cenário 1 . . . . .	81
5.2: Tempo total de paradas para cruzamento e ultrapassagem (minutos) para Cenário 1 . . . . .	81
5.3: Tempo de processamento (milisegundos) para o Cenário 1 . . . . .	82
5.4: Instantes de partida de cada trem para o Cenário 2 . . . . .	83
5.5: Tempo total de paradas para cruzamento e ultrapassagem (minutos) para o Cenário 2 . . . . .	84
5.6: Tempo de processamento (milisegundos) para Cenário 2 . . . . .	86
5.7: Instantes de partida dos trens para o Cenário 3 . . . . .	87
5.8: Tempo total de paradas para cruzamento e ultrapassagem (minutos) para o Cenário 3 . . . . .	85
5.9: Tempo de processamento (milisegundos) para o Cenário 3 . . . . .	88
5.10: Instantes de partida dos trens para o Cenário 4 . . . . .	89
5.11: Tempo total de paradas para cruzamento e ultrapassagem (minutos) para o Cenário 4 . . . . .	89
5.12: Tempo de processamento (milisegundos) para o Cenário 4 . . . . .	88
5.13: Horários de partida dos trens para o Cenário 5 . . . . .	89
5.14: Tempo total de paradas para cruzamento e ultrapassagem (minutos) trens para o Cenário 5 . . . . .	91
5.15: Tempo de processamento (milisegundos) para o Cenário 5 . . . . .	91
5.16: Instantes de partida dos trens para o Cenário 6 . . . . .	93
5.17: Tempo total de paradas para cruzamento e ultrapassagem (minutos) para o Cenário 6 . . . . .	94
5.18: Tempo de processamento (milisegundos) para o Cenário 6 . . . . .	95
5.19: Instantes de partida dos trens . . . . .	96
5.20: Tempos de parada em minutos para cruzamento e ultrapassagem para a flexibilização do PAT . . . . .	97
5.21: Instantes de partida dos trens . . . . .	100
5.22: Tempos de parada em minutos para cruzamento e ultrapassagem . . . . .	101
5.23: Instantes de partida dos trens . . . . .	103
5.24: Tempos de parada em minutos para cruzamento e ultrapassagem . . . . .	103
5.25: Instantes de partida dos trens para o Experimento 1. . . . .	105
5.26: Tempo de parada em minutos para cruzamento e ultrapassagem . . . . .	105
5.27: Instantes de partida dos trens para o Experimento 2 . . . . .	107
5.28: Tempo de parada em minutos para cruzamento e ultrapassagem . . . . .	107
5.29: Tempos de parada para cruzamento e ultrapassagem (minutos) . . . . .	110
5.30: Tempos de processamento (milisegundos) . . . . .	110
5.31: Tempos de parada para cruzamento e ultrapassagem e tempo de processamento para o algoritmo BA . . . . .	112
5.32: GAP de otimalidade para os diferentes algoritmos . . . . .	116
A1. Comprimento e número de vias dos segmentos do Modelo Ferroviário 1 . . . . .	127
A2. Velocidade (km/h) para os trens nos segmentos do Modelo Ferroviário 1. . . . .	128
A3. Comprimento e número de vias dos segmentos do Modelo Ferroviário 2 . . . . .	129
A4. Velocidade (km/h) para os trens nos segmentos do Modelo Ferroviário 2 . . . . .	130
B.1: Notações utilizadas nos algoritmos . . . . .	131

# Lista de Algoritmos

2.1: Rotina de despacho . . . . .	21
3.1: Algoritmo de decisão de preferência de trens . . . . .	43
3.2: Algoritmo de decisão de velocidade de trens . . . . .	52
4.1: Algoritmo de busca em profundidade . . . . .	62
4.2: Algoritmo de busca em largura . . . . .	64
4.3: Algoritmo de busca <i>BestFirst</i> . . . . .	66
4.4: Algoritmo de busca para planejamento de circulação de trens . . . . .	73

# Notação

$s$	segmento
$K$	Número de vias de um segmento
$k$	Via de um segmento
$S$	número de segmentos na linha ferroviária
$hdist$	distância mínima de separação que deve existir entre dois trens para garantir segurança de percurso
$K_s$	número de vias do segmento $s$
$d_s$	comprimento do segmento $s$
$v_{s,k}$	velocidade máxima permitida na via $k$ do segmento $s$
$CL_{s,k,k1}$	elemento da matriz de conexão no sentido leste-oeste
$CO_{s,k,k1}$	elemento da matriz de conexão no sentido oeste-leste
$\Psi$	Conjunto de atividades de manutenção
$r$	Atividade de manutenção
$sm_r$	segmento na qual será realizada a atividade de manutenção $r$
$km_r$	via na qual será realizada a atividade manutenção $r$
$z1_r$	instante de início da atividade manutenção $r$
$z2_r$	instante de término da atividade manutenção $r$
$sp_i$	segmento de partida (segmento inicial) do trem $i$
$N$	Número de trens a serem despachados
$i,j$	Trem ou trem fictício
$OL$	Conjunto de trens circulando no sentido oeste-leste
$LO$	Conjunto de trens circulando no sentido leste-oeste
$sp_i$	segmento de partida (segmento inicial) do trem $i$
$sd_i$	segmento de destino (segmento final) do trem $i$ ;
$kp_i$	via de partida (via inicial) do trem $i$ ;
$ip_i$	instante planejado de partida do trem $i$ do segmento $sp_i$ ;
$ic_i$	instante planejado de chegada do trem $i$ no segmento $sd_i$
$PAT_i$	programa de atividades do trem $i$
$percurso_i$	percurso do trem $i$
$t_{s,i,k}$	tempo que o trem $i$ leva para percorrer a via $k$ do segmento $s$ .
$\eta$	$OL \cup LO \cup \Psi$ .
$tf$	Trem fictício relacionado à atividade de manutenção
$x_i$	hora na qual o trem $i$ completará sua atividade atual
$a_i$	segmento onde está posicionado o trem $i$
$b_i$	via onde está posicionado o trem $i$

$e_i$	Estado do trem $i$ . 1 se esta parado e 0 caso contrário.
$\psi_{tf}$	Conjunto de trens fictícios relacionados à atividade de manutenção
$l$	Trem ou trem fictício sendo despachado
$\Delta$	intervalo tempo necessário para o trem percorrer os dois segmentos seguintes ao seu segmento atual
$\Phi$	conjunto de trens $\Phi$ que concorrem pelo uso do segmento $s_l$
$KT$	conjunto de vias que podem ser ocupadas pelo trem $l$
$\tau_1$	tempo necessário para o trem $l$ desacelerar e acelerar no segmento $s_1$
$I$	Conjunto de trens
$S1$	Conjunto de segmentos singelos
$S2$	Conjunto de pátios de cruzamento
$Sm$	conjunto de segmentos simples que estão em manutenção em determinados intervalos de tempo
$A_{i,j,s}$	1 se o trem $i, i \leq m$ , ocupa o segmento $s \in S1$ antes do trem $j, j \leq m$ ; ou 0 em caso contrário
$B_{i,j,s}$	1 se o trem $i, i \leq m$ , ocupa o segmento $s \in S1$ antes do trem $j, j > m$ ; ou 0 em caso contrário
$C_{i,j,s}$	1 se o trem $i, i > m$ , ocupa o segmento $s \in S1$ antes do trem $j, j > m$ ; ou 0 em caso contrário
$Q_{i,s}$	1 se o trem $i \in I$ ocupa o segmento $s \in Sm$ depois de um período após sua manutenção; ou 0 se o ocupa antes de a manutenção começar
$Xa_{i,s}$	instante de chegada do trem $i \in I$ no segmento $s \in S$
$Xd_{i,s}$	instante de partida do trem $i \in I$ do segmento $s \in S$
$t_{i,s} \min$	tempo mínimo para o trem $i \in I$ atravessar o segmento $s \in S$
$t_{i,s} \max$	tempo máximo para o trem $i \in I$ atravessar o segmento $s \in S$ ;
$t_{i,s}$	tempo que o trem $i \in I$ leva para atravessar o segmento $s \in S$
$K_s \min$	instante de início da manutenção no segmento $s \in Sm$
$K_s \max$	instante de termino da manutenção no segmento $s \in Sm$
$W_{i,s}$	intervalo de tempo que o trem $i \in I$ leva para executar atividades do $PAT_i$ no segmento $s \in S$
$E_{i,s}$	intervalo de tempo para o trem $i \in I$ entrar no pátio $s \in$
$s_q$	Segmento singelo
$s_p$	Pátio de cruzamento
$m$	Número de trens no sentido Oeste-Leste
$p_i$	Prioridade estática do trem $i$
$n$	Nó
$n'$	Nó filho do nó $n$

# Capítulo 1

## Introdução

Sistemas ferroviários possuem um papel importante no transporte de cargas e passageiros em todo o mundo. O transporte ferroviário caracteriza-se, especialmente, por sua capacidade de transportar grandes volumes, com elevada eficiência energética, principalmente em casos de deslocamentos a médias e grandes distâncias. Apresenta, ainda, maior segurança, em relação ao transporte rodoviário, com menor índice de acidentes e menor incidência de furtos e roubos.

A construção da malha ferroviária brasileira teve início em 1854 com a inauguração da Estrada de Ferro Mauá, inicialmente com apenas 14,5 quilômetros. Na década de 1950 a malha brasileira atingiu sua maior extensão com cerca de 38.000 quilômetros de ferrovias. Iniciou-se então um processo de redução de investimentos em ampliação e manutenção das ferrovias existentes, o que resultou na desativação de diversos trechos por todo o território levando a malha ferroviária a uma forte redução a partir de 1962, como mostra a Figura 1.1.



Figura 1.1 – Evolução das Ferrovias no Brasil (fonte: ANTF).

A partir de 1992 a RFFSA (Rede Ferroviária Federal S.A.) foi incluída no Programa Nacional de Desestatização com o qual praticamente toda a malha ferroviária foi transferida

à iniciativa privada. Após a privatização da malha nacional os investimentos foram retomados. Atualmente existe um contínuo aumento no volume de carga transportada e no uso do sistema ferroviário como um eficiente sistema de transporte. No período de 2002 a 2005 o transporte ferroviário de carga passou de 170 bilhões de TKU (tonelagem X quilometro útil) para 222 bilhões, com um crescimento médio anual de 9,2%. A participação do sistema ferroviário no transporte de cargas aumentou de 20,86% em 2000 para 24% em 2006 (fonte: ANTT – Agência Nacional Transportes Terrestres).

Os custos necessários para construção de trechos e ramais ferroviários são vultuosos, o que dificulta e, em muitos casos, inviabiliza investimentos em infra-estrutura. O aumento das exportações brasileiras nos últimos anos levou a um considerável aumento na demanda de transporte ferroviário. Como não houve investimento significativo na construção de novos trechos, o crescimento da demanda exige um complexo planejamento para ser atendida. A Figura 1.2 mostra as etapas de planejamento e operação de transporte ferroviário.

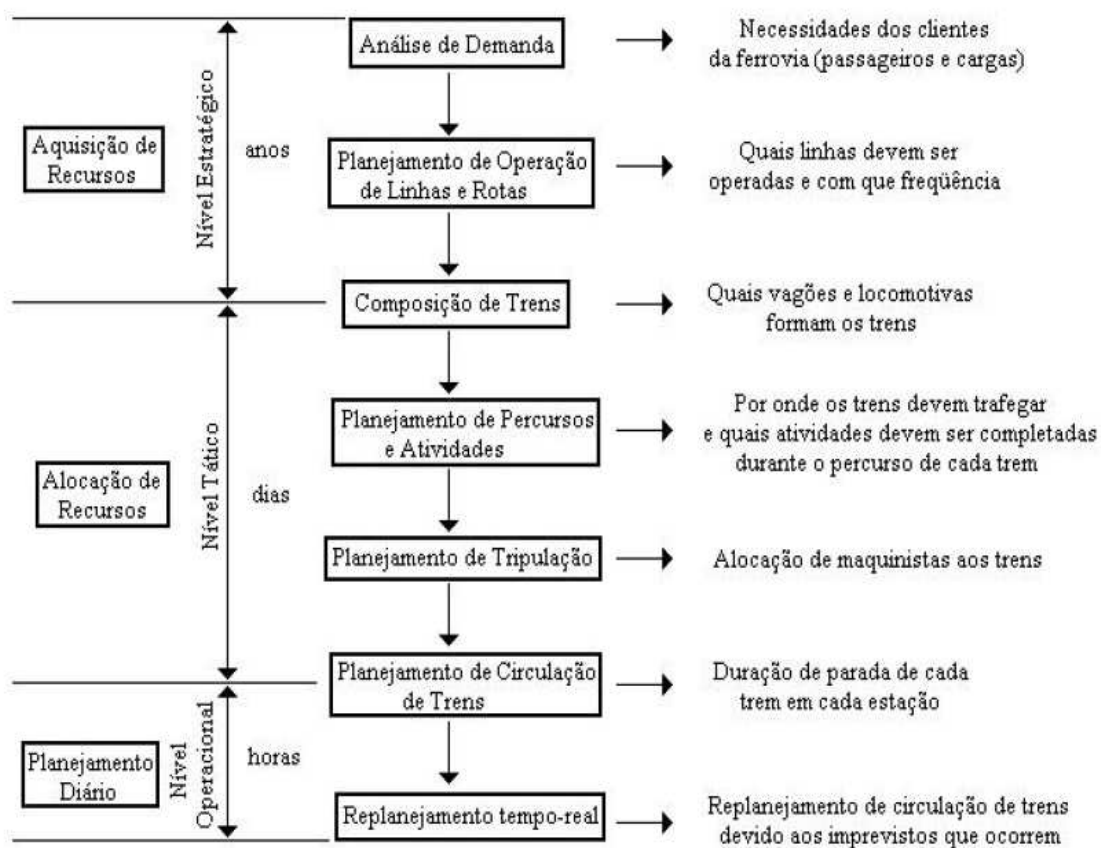


Figura 1.2 – Níveis de planejamento ferroviário.



No nível estratégico, que tem uma escala de tempo anual, se faz previsões de demanda e análise da frequência na qual alguns trechos e rotas devem ser operados. O objetivo é garantir recursos (vagões, locomotivas, etc) suficientes para atender a demanda da ferrovia.

O nível tático tem escala de tempo mensal, semanal e diária e faz a alocação dos recursos na ferrovia, distribuindo vagões e locomotivas para a formação dos trens. Neste nível são programadas também as atividades dos trens, como anexação e retirada de vagões, abastecimento, troca de tripulação, etc. O nível operacional trabalha em uma escala de tempo de horas e em tempo-real e nele faz-se o planejamento de circulação de trens, definindo-se os locais e tempos de parada de cada trem. Para atender o dinamismo da operação de uma ferrovia, o planejamento e replanejamento da circulação de trens é feito em tempo-real. Para que as metas de transporte e demanda sejam atingidas faz-se necessário utilizar ao máximo a capacidade de transporte de uma ferrovia.

Mas a utilização está muitas vezes em conflito com a infra-estrutura encontrada nas ferrovias. Isso significa que a capacidade de transporte é limitada pela capacidade da ferrovia. Sendo assim, o planejamento de circulação de trens é um importante ingrediente no aumento de produtividade de uma ferrovia.

Em grande parte das ferrovias brasileiras o planejamento da circulação de trens era, até recentemente, elaborado com lápis e papel por despachadores dos centros de controle, sendo um processo muito cansativo, dependente da experiência, sujeito a erros e limitado pela capacidade do ser humano. Como a operação de uma ferrovia é um processo dinâmico, sujeito a fatores conflitantes e imprevisíveis, essa importante etapa do planejamento de trens se torna muitas vezes limitada e pouco eficiente.

A necessidade de um melhor aproveitamento da capacidade de uma ferrovia aliada ao dinamismo de sua operação faz que sistemas de apoio à tomada decisão em tempo real sejam uma ferramenta apropriada para a questão. Existe um grande esforço em pesquisa e desenvolvimento de novas metodologias. O objetivo é a obtenção de soluções para o problema de circulação de trens que sejam adequadas ao modo de operação da ferrovia.

Ferramentas computacionais para o apoio ao planejamento de circulação de trens começaram a ser implantadas no Brasil em meados da década de 1990. Porém as pesquisas de diferentes métodos para o planejamento de circulação de trens datam da década de 70. A

aplicação de Inteligência Computacional é uma das opções que tem mostrado promissora nesta área.

### **1.1 Descrição do Problema.**

Planejamento e controle de circulação de trens são tarefas complexas e exercem uma das principais influências na produtividade de uma ferrovia. Planejamento de circulação envolve problemas de otimização combinatorial, problemas estes que são NP-Completo (Garey e Johnson, 1979). A obtenção de soluções ótimas utilizando modelos exatos de otimização não é atrativa para aplicações reais, pois o tempo necessário para obtenção de uma solução em geral é muito maior que o tempo de resposta necessário.

Obter uma solução para o planejamento de circulação de trens significa determinar os horários de chegada e partida de cada trem em cada segmento de uma linha ferroviária visando minimizar o tempo total de paradas dos trens ou otimizar um objetivo relacionado a custo ou desempenho.

A principal decisão em planejamento de circulação envolve a preferência de trens quando em situação de conflito com outros trens. A escolha de preferências determina a qualidade de uma solução. O trabalho de um despachador de trens consiste exatamente em fazer estas escolhas com objetivo de otimizar o desempenho esperado, como por exemplo, cumprir os horários de chegada estimados para cada trem.

O uso de modelos de linhas ferroviárias (Peterson e Taylor, 1982), permite simular o movimento de trens e, com isso, obter soluções para o planejamento. Como a qualidade destas soluções depende das decisões de prioridade, é muito importante utilizar uma boa estratégia de decisão.

Uma das principais limitações do processo de planejamento manual feito por controladores é a dificuldade humana de prever o impacto de cada decisão nas decisões futuras de planejamento. Dessa mesma maneira, um algoritmo para obter soluções de melhor qualidade deve ter a capacidade de tratar globalmente o problema de decisão.

## **1.2 Motivação.**

Como foi visto anteriormente, o planejamento de circulação de trens é uma etapa muito importante na operação de uma ferrovia e o uso de ferramentas computacionais de apoio à tomada de decisão é fundamental para utilizar eficientemente a capacidade da ferrovia.

Existe uma carência no Brasil no que concerne a pesquisa e desenvolvimento de metodologias adequadas à realidade nacional para o planejamento de circulação de trens, principalmente em ambientes de tempo-real. As aplicações hoje existentes utilizam abordagens que fornecem soluções que otimizam modelos, enquanto que a operação de uma ferrovia requer maior realismo e menor tempo de processamento.

Ferramentas de apoio à tomada de decisão que utilizam conceitos de Inteligência Computacional são candidatas potenciais para abordar este tipo de problema, pois fornecem ferramentas para a criação de algoritmos flexíveis que conseguem explorar o espaço de soluções otimizando algum objetivo, além de possibilitar a inserção de conhecimento ao processo de obtenção de soluções. Aplicações em problemas como Roteamento de Veículos (Homberger e Gehring, 2005), *Job-Shop* (Jennings e Bussman, 2003; Sen *et al.*, 1996; Davis, 1985), Atribuição de Tarefas (Glover e McMillan, 1986) além de muitas outras aplicações de Meta-Heurísticas (Reeves *et al.*, 1996), Busca em Árvore (Pearl, 1984; Russel e Norvig, 2003), Satisfação de Restrições (Russel e Norvig, 2003), Algoritmos Genéticos (Goldberg, 1989) em diversos problemas de otimização combinatorial confirmam o uso de Inteligência Computacional como uma estratégia atrativa.

## **1.3 Objetivos.**

Os objetivos deste trabalho são os seguintes :

- Entender e detalhar o problema de circulação de trens;
- Estudar, analisar e comparar diferentes abordagens propostas na literatura;
- Desenvolver algoritmos de planejamento de circulação de trens visando ambientes de tempo real;

- Testar e comparar diferentes propostas de planejamento de circulação de trens sob o ponto de vista da qualidade da solução e do desempenho computacional.

## **1.4 Organização do Trabalho.**

Após esta introdução, este trabalho está organizado da seguinte forma:

- Capítulo 2 define o problema de circulação, apresenta um levantamento das principais propostas existentes na literatura, e descreve o modelo de ferrovia considerado pelo simulador a eventos discretos utilizado no desenvolvimento das propostas aqui sugeridas.
- O Capítulo 3 propõe um algoritmo de decisão de prioridade e um algoritmo velocidade de trens. O primeiro algoritmo decide, baseado em uma referência de percurso, a preferência entre trens competindo pelo uso de um mesmo segmento de modo que os trens fiquem o mais próximo possível de suas referências. O segundo algoritmo decide, baseado também em uma referência de percurso, a preferência entre trens competindo pelo uso de um mesmo segmento e também a velocidade com a qual o trem preferencial deve percorrer o segmento disputado de modo a se reaproximar de sua referência;
- O Capítulo 4 desenvolve um algoritmo de Busca em Árvore para o problema de circulação de trens. Cada nó representa um conflito entre trens competindo pelo uso de um mesmo segmento e os nós são selecionados com base em uma estimativa do custo de um nó dentro de um horizonte de tempo.
- Capítulo 5 apresenta uma série de experimentos computacionais com objetivo de comparar as propostas deste trabalho com diferentes propostas existentes. Os resultados são gerados com base em instâncias simples, para as quais se conhece as soluções ótimas, e uma instância obtida de dados reais de uma ferrovia brasileira.
- O Capítulo 6 apresenta as conclusões e trabalhos futuros.

## Capítulo 2

### Planejamento em Circulação de Trens.

Esse capítulo tem o propósito de detalhar e formalizar o problema de planejamento em circulação de trens. Inicialmente é feita uma revisão de diversas propostas encontradas na literatura para a solução do problema de circulação de trens e serão mostrados os principais aspectos e notação utilizada nesse trabalho. Por fim será feita uma descrição do simulador a eventos discretos para movimentação de trens em uma malha ferroviária.

#### 2.1 Definição do Problema.

O problema de planejamento de circulação de trens requer uma discussão sobre o funcionamento de uma ferrovia.

Em uma linha ferroviária, ou seja, em um trecho de uma malha ferroviária com linha única (ou singela) a movimentação de trens ocorre nos dois sentidos da linha. Uma linha ferroviária é formada por segmentos, isto é, trechos com uma única via, pátios, ou estações intercaladas entre trechos singelos. Cruzamentos, ultrapassagens de trens e atividades, como carga, descarga, manobras, recomposição, abastecimento, etc. ocorrem em pátios e estações. Um pátio pode ter duas ou mais linhas (vias) para as atividades. A Figura 2.1 mostra um exemplo de linha com trens circulando no sentido Leste-Oeste e Oeste-Leste. Os trens são representados por flechas e o sentido da flecha indica o sentido do trem. Neste exemplo, a linha é composta por sete segmentos, sendo dois deles pátios de cruzamento ( $s_3, s_5$ ), dois pátios de atividades ( $s_1, s_7$ ) e trechos de via única (singelos) ( $s_2, s_4, s_6$ ). O segmento  $s_1$  possui quatro vias,  $s_7$  três vias e  $s_3$  e  $s_5$  duas vias cada.

Na Figura 2.1 fica evidente a principal restrição que influencia circulação de trens em uma linha: os cruzamentos e ultrapassagens de trens só podem ser feitos nos pátios de cruzamento e trechos singelos só podem ser ocupados por um trem. Naturalmente o número de trens em um pátio não pode exceder a capacidade do pátio.

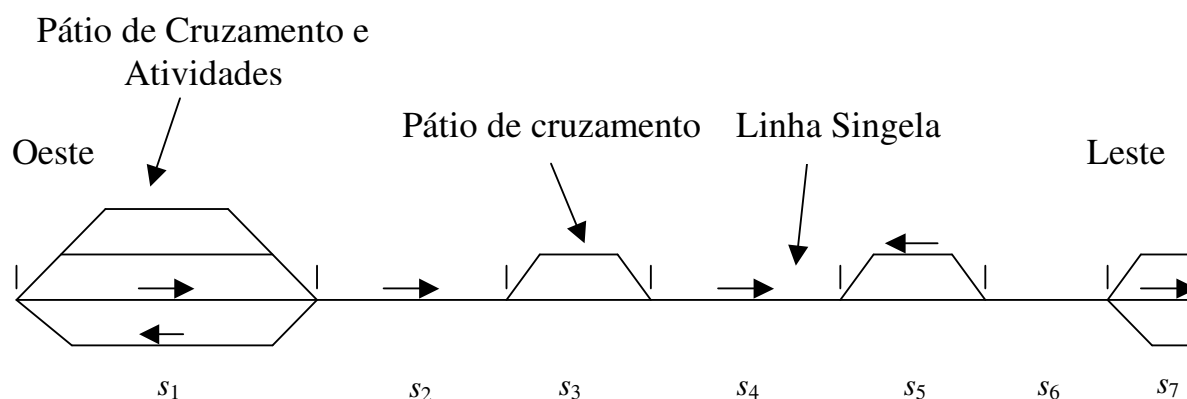


Figura 2.1: Exemplo de configuração de uma linha ferroviária e circulação de trens.

A Figura 2.2 mostra um exemplo de cruzamento de trens. No instante  $t$  o trem T1 no segmento  $s_4$  e o trem T2 em  $s_6$  competem pela ocupação do segmento  $s_5$ . Contudo, como uma via singela pode ser ocupada por um único trem, ocorre um conflito de ocupação. No instante  $t'$  o conflito é solucionado com um cruzamento no segmento  $s_4$ , sendo que o trem T2 ganha a preferência e ocupa o segmento  $s_5$  e o trem T1 aguarda o cruzamento em  $s_4$ .

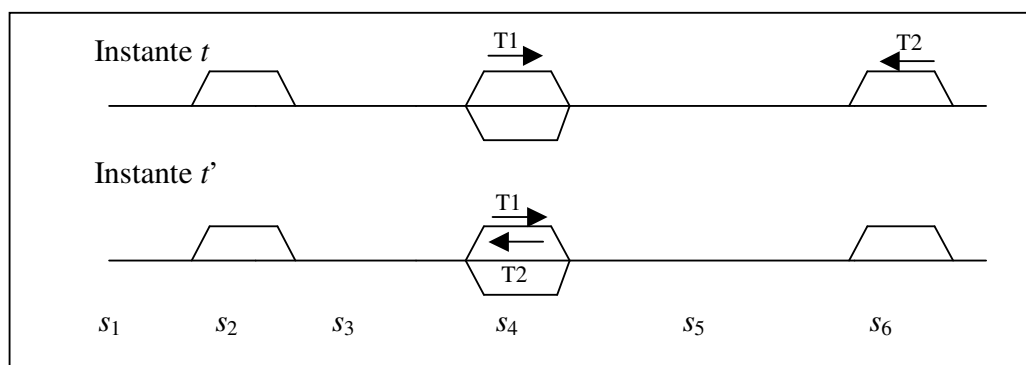


Figura 2.2: Exemplo de cruzamento de trens.

A Figura 2.3 mostra um exemplo de ultrapassagem de trens. No instante  $t$  os trens T1 e T2 no segmento  $s_4$  estão competindo pela ocupação do segmento  $s_5$  e, como no caso anterior, ocorre um conflito de ocupação. No instante  $t'$  o conflito é solucionado permitindo ultrapassagem no segmento  $s_4$ , sendo que o trem T2 ganha a preferência e o ocupa o segmento  $s_5$ .

Um exemplo de violação na restrição de ocupação de um segmento pode ser vista na Figura 2.4. No instante  $t$  o trem T1 segmento  $s_4$  deseja ocupar o segmento  $s_5$  já ocupado pelo trem T2. No instante  $t'$  o trem T1 ocupa indevidamente o segmento  $s_5$ . A Figura 2.5 mostra a ocupação correta para o segmento  $s_5$  no caso da Figura 2.3

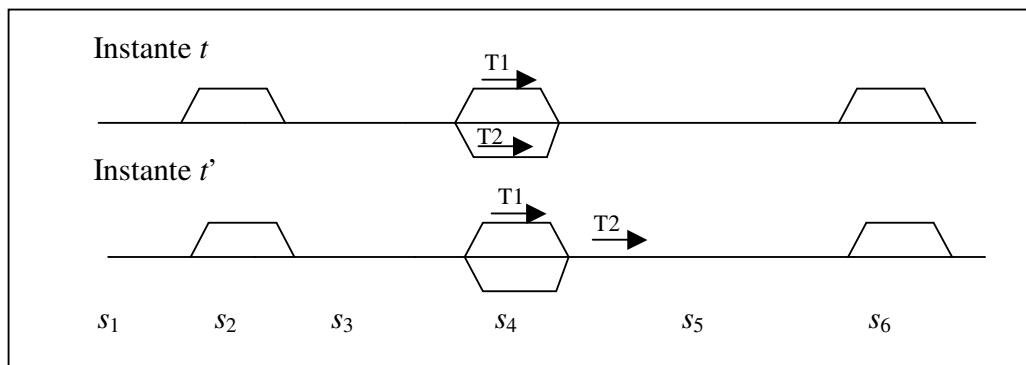


Figura 2.3: Exemplo de ultrapassagem de trens.

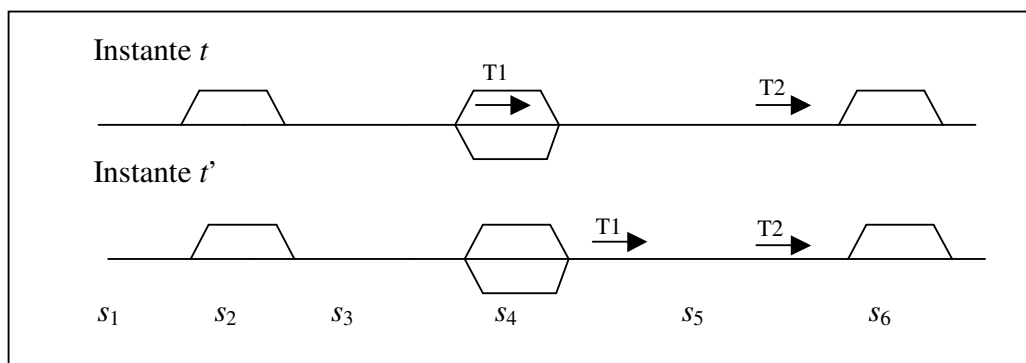


Figura 2.4: Exemplo de ocupação indevida.

Outra restrição de circulação de trens é a restrição de bloqueio. Uma situação de bloqueio em uma linha ferroviária ocorre quando trens ficam impedidos de se movimentarem, sem que algum trem seja obrigado a recuar.

A Figura 2.6 mostra como a movimentação do trem T1 para o segmento  $s_4$  em  $t'$  levou a uma situação de bloqueio, onde os quatro trens estão impedidos de se movimentarem. A Figura 2.7 mostra como esta situação de bloqueio pode ser evitada por uma ordenação apropriada do movimento dos trens.

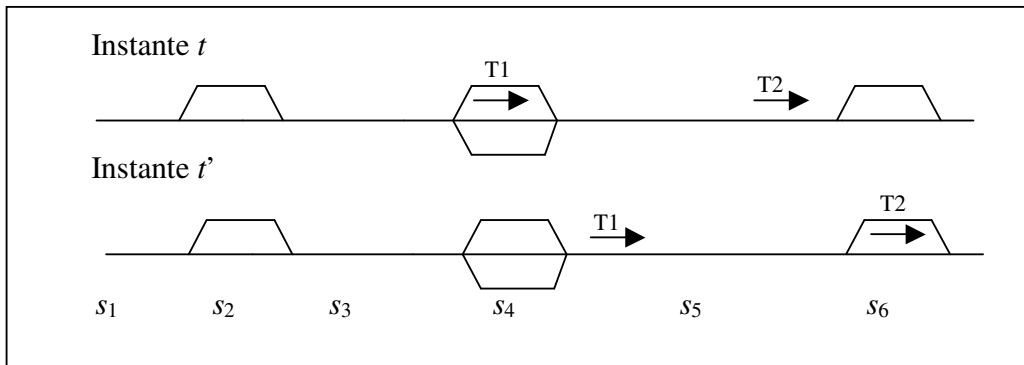


Figura 2.5: Exemplo de ocupação correta.

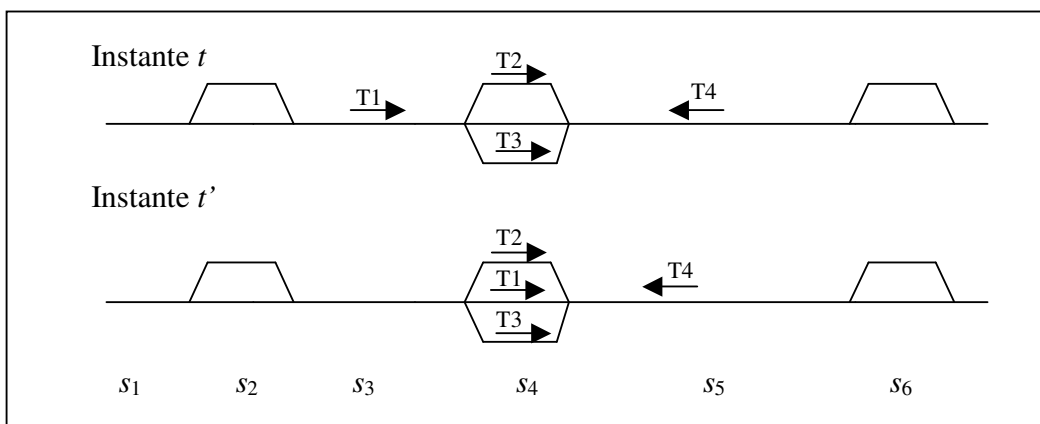


Figura 2.6: Exemplo de situação de bloqueio.

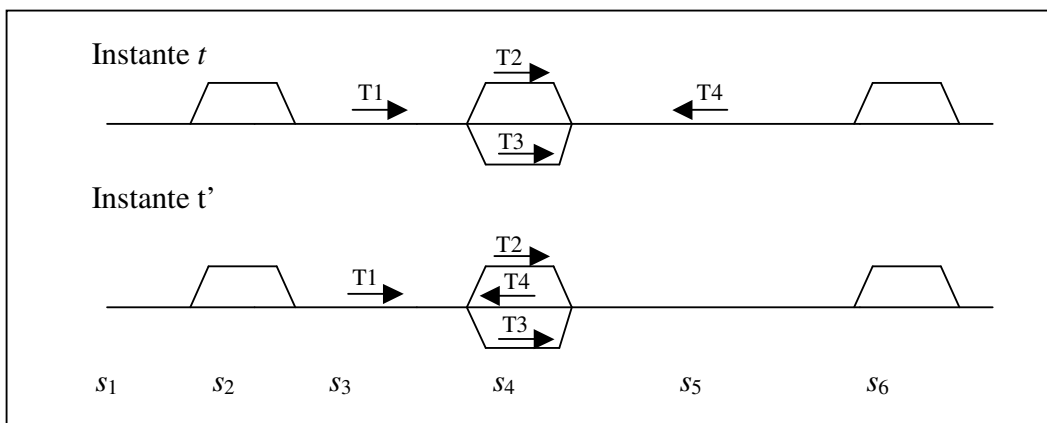


Figura 2.7: Exemplo de ordenação de circulação para evitar uma situação de bloqueio.

Obter uma solução para o problema de planejamento de circulação de trens consiste em determinar, para todos os trens, os horários de entrada e saída em cada segmento, respeitando as restrições (cruzamento, ultrapassagem, ocupação e bloqueio) e visando



minimizar o tempo total de paradas dos trens ou otimizar um objetivo relacionado a custo ou desempenho. Outras restrições de tempos e de segurança serão discutidas nas próximas seções.

A maneira tradicional e mais simples e intuitiva de se representar uma solução de planejamento de circulação de trens é através do gráfico de trens, conforme mostra a Figura 2.8.

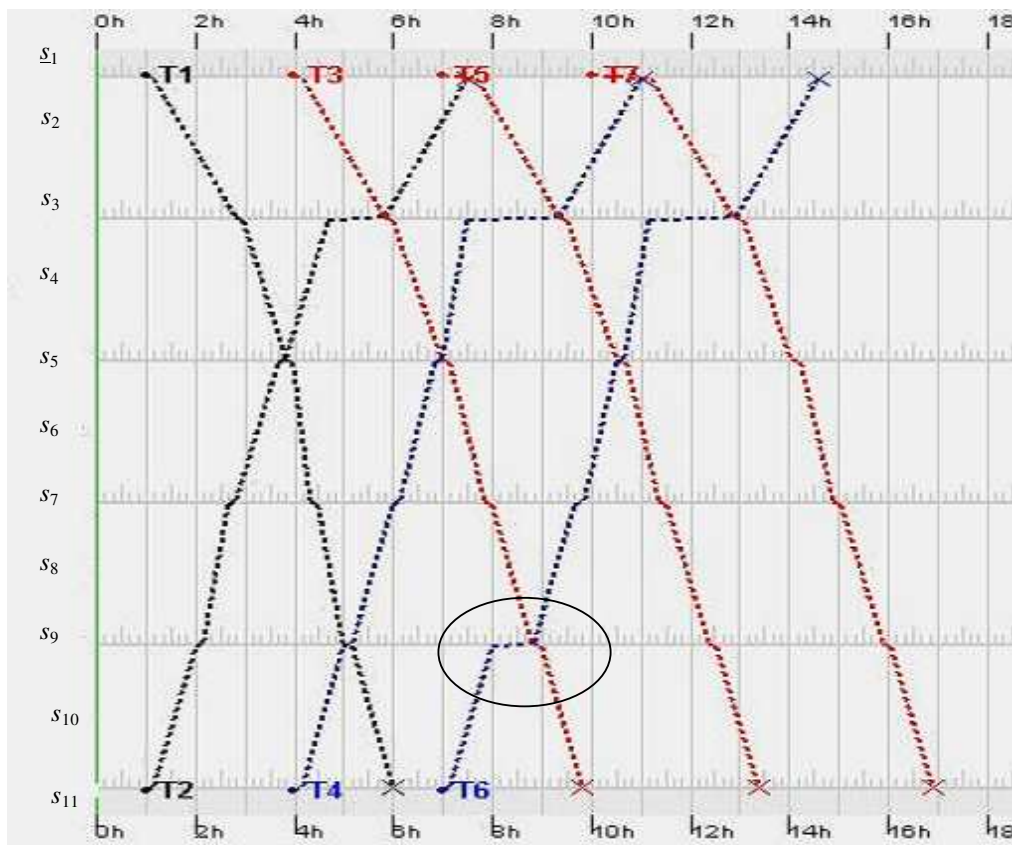


Figura 2.8: Representação de solução de planejamento de circulação de trens através de gráfico de trens.

Na Figura 2.8 o eixo horizontal representa o tempo, o eixo vertical representa a linha ferroviária por onde os trens circulam e as linhas paralelas ao eixo do tempo representam os pátios de atividade e de cruzamento. As linhas pontilhadas inclinadas representam o deslocamento dos trens pela linha ferroviária. Os trens T1, T3, T5 e T7 circulam no sentido Leste-Oeste (de cima para baixo no gráfico) e tem como origem o segmento  $s_1$  e como destino o segmento  $s_{11}$ . Os trens T2, T4 e T6 circulam no sentido Oeste-Leste e têm como origem o segmento  $s_{11}$  e como destino o segmento  $s_1$ . Um

cruzamento de trens ocorre no pátio de cruzamento  $s_9$  envolvendo os trens T6 e T3, conforme mostra a região circulada na Figura 2.8. O trem T6 permanece no segmento  $s_9$  das 8 horas às 9 horas aguardando a chegada do trem T3. Após o cruzamento os dois trens seguem seus percursos.

## 2.2 Revisão Bibliográfica.

O planejamento de circulação de trens como um problema de otimização combinatorial implica em problemas da classe NP-Completo. O número de soluções para o problema cresce exponencialmente, pois é  $O(2^n)$ , onde  $n$  é o produto do número de cruzamentos e ultrapassagens que ocorrem na circulação pelo número de pátios de cruzamento. Pode-se fazer uma analogia entre o problema de circulação de trens e o problema de *Job-Shop* (Conway *et al.*, 1967). Os trens podem ser considerados os *jobs* e os segmentos da ferrovia as máquinas. Como muitos segmentos de uma ferrovia são capazes de alocar mais de um trem ao mesmo tempo, o equivalente a alocar mais de um *job* para uma mesma máquina, o problema se torna mais complexo que o *Job-Shop*.

As primeiras propostas para encontrar uma solução para o planejamento de circulação de trens baseavam-se no uso de pesquisa operacional (Winston, 2004) com o objetivo de encontrar a solução ótima relacionada a uma determinada função objetivo. Nesta categoria podemos citar os trabalhos de Sziepel (1972), Sauder e Weterman (1983) e Jovanovic, (1989). Modelos de otimização que consideram a velocidade do trem com uma variável de decisão foram propostos por Higgins *et al.* (1996). Em Valle *et al.* (2005), conjuntos nebulosos são utilizados na flexibilização de restrições de velocidade no uso de programação inteira-mista.

A complexidade do problema se mostra como um grande desafio ao uso de modelos de otimização (Issai e Cassaigne, 2001). Para pequenas instâncias consegue-se obter soluções ótimas em um curto intervalo de tempo. Já em aplicações práticas, com grande quantidade de trens circulando, o uso de métodos exatos de otimização se mostram pouco eficientes. Além disso, o processo de despacho de trens é extremamente dinâmico e multi-objetivo. Deve-se atender a demanda de transporte e escoamento de carga com o melhor uso possível dos recursos da ferrovia (locomotivas, vagões, maquinistas, linha ferroviária, etc). Eventos como avaria de locomotivas, descarrilhamento, defeito na linha, etc., ocorrem

com frequência. Em geral, o tempo necessário para obtenção de uma solução ótima exata é muito grande com relação ao dinamismo da ferrovia. No momento que a solução for obtida, o estado do sistema é, em geral, diferente daquele considerado inicialmente e a solução obtida pode não ser mais útil.

Devido a esses problemas, diversas propostas surgiram explorando a área de inteligência computacional e simulação. O uso de heurísticas e meta-heurísticas (Reeves *et al.*, 2004) são hoje essenciais para a solução de problemas de circulação de trens, particularmente em ambientes de tempo real.

A aplicação de modelos de simulação a eventos discretos tem como principal referência o trabalho de Peterson e Taylor (1982). O modelo visa a movimentação dos trens em uma linha ferroviária com o uso do conceito de ocorrência de eventos (Cassandras e Lafortune, 1999). Uma extensão do modelo de Peterson e Taylor (1982) é proposta em Rondón (2000), onde se incorpora a noção de uma malha ferroviária e não apenas linha ou corredor.

Uma das primeiras e mais significativas aplicações de inteligência artificial para o problema é feita em Cherniavsky (1972). Este artigo apresenta uma metodologia para obtenção de soluções factíveis utilizando uma base de regras para a satisfação das restrições de circulação e um algoritmo de busca em árvore para a obtenção de melhores soluções. A modelagem do problema considera uma linha ferroviária com segmentos singelos separados por pátios de cruzamento, com trens de carga e de passageiros circulando nos dois sentidos da ferrovia.

Em Chiang *et al.* (1998), é apresentada outra proposta baseada em uma base de regras para satisfazer as restrições de circulação. Nesta abordagem os trens são despachados sem que haja prevenção de conflitos, o que produz um planejamento completo de circulação, porém infactível. O que a abordagem faz é identificar os conflitos e classificá-los de acordo com as restrições violadas. Um conjunto de regras escolhe, para cada conflito, um conjunto de operadores possíveis. Os operadores são testados até que se consiga solucionar o conflito. A solução de conflitos é repetida até se obter um planejamento factível.

Propostas com uso de Satisfação de Restrições, *Simulated Annealing* e Busca Tabu são discutidas em Issai e Singh (2001). Uma heurística baseada em satisfação de restrições

é usada na geração de uma solução inicial factível para o problema. A solução inicial obtida é, por sua vez, usada na inicialização dos algoritmos de *Simulated Annealing* e Busca Tabu. Em Issai e Singh (2001), não se define o modelo utilizado, a função de resfriamento e operadores de busca local utilizados.

O uso de algoritmos genéticos pode ser encontrado em Higgins *et al.* (1997); Mendes (2005), e Mendes *et al.* (2005). Outra abordagem de algoritmos genéticos é feita por Salim e Cai (1995). As abordagens de Mendes (2005) e Valle *et al.* (2005), serão utilizadas na comparação de resultados com as propostas apresentadas neste trabalho. Por este motivo estas propostas são resumidas nas seções seguintes.

É importante ressaltar duas grandes diferenças entre as propostas existentes. A maioria das propostas discutidas na literatura tem como objetivo obter uma programação de trens diária, semanal, etc., para ser obedecida pela ferrovia. As propostas que se enquadram nesse modelo tem como característica não o tempo de processamento necessário para obter a solução, mas sim a qualidade da solução. Outras abordagens têm como objetivo o planejamento de circulação de trens em tempo-real, visando o uso por controladores de trens (despachadores). Esta segunda categoria tem preocupação com o tempo computacional, exigindo algoritmos rápidos, isto é, algoritmos que forneçam soluções em segundos.

As propostas deste trabalho objetivam soluções em tempo-real. Assim, além da qualidade da solução, o tempo de processamento é uma preocupação em todo o trabalho. Outras propostas para planejamento de circulação de trens em tempo-real podem ser encontradas na literatura (Gomide e Vieira, 1996). Em Gomide *et al.* (1999), é discutida a proposta de um sistema baseado em conhecimento que utiliza técnicas de conjuntos nebulosos para analisar a movimentação dos trens e ajudar o despachador a guiar o trem da melhor maneira possível. Rondón (2000), apresenta um simulador a eventos discreto e um modelo de malha ferroviária para obtenção de soluções de planejamento de circulação de trens em tempo-real.

## 2.3 Modelo de Ferrovia.

Para uma melhor compreensão das propostas sugeridas neste trabalho, esta seção detalha o modelo de ferrovia adotado. Este modelo foi proposto em Rondón (2000), utilizando uma modelagem computacional orientada a objetos tanto no desenvolvimento quanto na implementação de um simulador a eventos discretos de circulação de trens. Este simulador é utilizado neste trabalho.

### 2.3.1 Configuração da linha ferroviária.

Define-se uma linha ferroviária com sendo a ligação entre duas regiões distintas por meio de trilhos. Na prática, linhas ferroviárias para o transporte de carga fazem geralmente a ligação entre grandes regiões produtoras a pontos de escoamento de produção, como portos, ou centros de consumo, como grandes cidades.

Uma linha ferroviária é subdividida em trechos menores denominados segmentos. Um segmento pode corresponder a um ponto de carga e descarga, um pátio de cruzamento e manobras, ou apenas a ligação entre dois pontos da linha. Cada segmento possui um comprimento e uma ou mais vias.

A Figura 2.9 mostra um exemplo de linha ferroviária com sete segmentos. Cada segmento  $s$  possui um número  $K$  de vias.

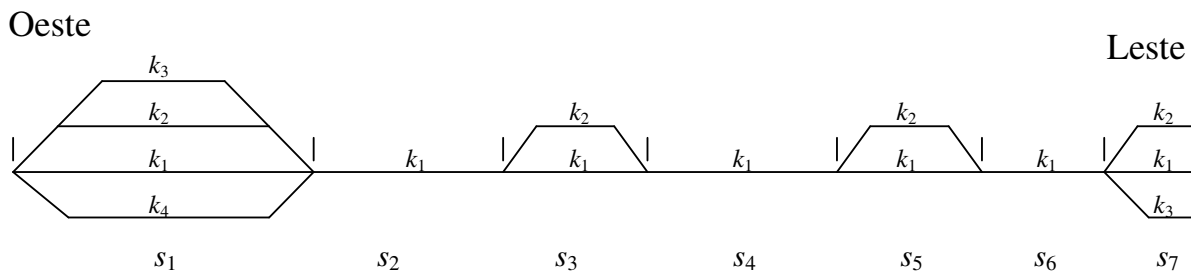


Figura 2.9: Exemplo de linha ferroviária.

Os atributos que definem uma linha ferroviária são:

- $S$ : número de segmentos na linha ferroviária;
- $hdist$ : distância mínima de separação que deve existir entre dois trens para garantir segurança de percurso.

Os atributos que cada segmento  $s$  de uma linha ferroviária são:

- $K_s$ : número de vias do segmento  $s$ ;
- $d_s$ : comprimento do segmento.

Para cada via  $k$  do segmento  $s$  define-se:

- $v_{s,k}$ : velocidade máxima permitida na via  $k$  do segmento  $s$ .

Matrizes que representam as conexões físicas entre os segmentos são definidas da seguinte forma:

$CL_{s,k,k1}$ : elemento da matriz de conexão no sentido leste-oeste. Assume o valor 1 se existe conexão física entre a via  $k$  do segmento  $s$  com a via  $k1$  do segmento  $s-1$ . Caso contrário assume valor 0;

$CO_{s,k,k1}$ : elemento da matriz de conexão no sentido oeste-leste. Assume o valor 1 se existe conexão física entre a via  $k$  do segmento  $s$  com a via  $k1$  do segmento  $s+1$ . Caso contrário assume valor 0.

### 2.3.2 Atividades de manutenção.

O modelo permite definir um conjunto  $\Psi$  de atividades de manutenção. Cada atividade de manutenção  $r \in \Psi$  a ser realizada em uma via possui os seguintes atributos:

- $sm_r$ : segmento na qual será realizada a atividade;
- $km_r$ : via na qual será realizada a atividade;
- $z1_r$ : instante de início da atividade;
- $z2_r$ : instante de término da atividade.

### 2.3.3 Descrição de trens.

O modelo original supõe a existência de  $N$  trens a serem despachados sobre a linha ferroviária. Esses trens são separados em dois conjuntos disjuntos  $OL$  e  $LO$ . Estes dois conjuntos representam trens viajando no sentido oeste-leste ( $OL$ ) e leste-oeste ( $LO$ ).

Os atributos básicos que cada trem  $i \in OL \cup LO$  possui são:

- $sp_i$ : segmento de partida (segmento inicial) do trem  $i$ ;
- $sd_i$ : segmento de destino (segmento final) do trem  $i$ ;
- $kp_i$ : via de partida (via inicial) do trem  $i$ ;
- $ip_i$ : instante planejado de partida do trem  $i$  do segmento  $sp_i$ ;
- $ic_i$ : instante planejado de chegada do trem  $i$  no segmento  $sd_i$ ;
- $percurso_i$ : percurso do trem  $i$ . Indica por quais segmento o trem  $i$  deve passar para chegar ao seu destino;
- $PAT_i$ : programa de atividades do trem  $i$ . Indica as atividades que o trem  $i$  realiza em cada segmento. As atividades podem ser de abastecimento, troca de equipagem, carga e descarga, anexação de vagões, etc.;
- $t_{s,i,k}$ : tempo que o trem  $i$  leva para percorrer a via  $k$  do segmento  $s$ .

Uma atividade de manutenção  $r \in \mathcal{P}$  é considerada como um trem fictício ocupando a via do segmento onde a atividade ocorre durante o intervalo de tempo  $z2_j - z1_j$ . Dessa forma, para todos os trens e atividades de manutenção do modelo define-se o conjunto  $\mathcal{P}_{tf}$  como o conjunto de trens fictícios relacionados a atividades de manutenção e o conjunto  $\eta = OL \cup LO \cup \mathcal{P}_{tf}$ .

### 2.3.4 Lógica do simulador.

Esta seção resume a dinâmica do modelo de simulação utilizado neste trabalho. O modelo de simulação a eventos discreto está baseado na proposta de Perterson e Taylor, 1982. Os eventos correspondem ao início e término de atividades de trem. O tempo avança de acordo com a ocorrência de eventos. Para cada tipo de evento é feito um tratamento específico de modo a evoluir o estado atual do sistema para o próximo estado.

O estado do sistema é completamente especificado conhecendo-se, para cada trem de  $\eta$  as seguintes variáveis:

- $x_i$ : instante na qual o trem  $i$  completará sua atividade atual. Caso  $i$  seja uma atividade de manutenção,  $x_i = z2_i$ ;
- $a_i$ : segmento onde está posicionado o trem  $i$ . Caso  $i$  seja uma atividade de manutenção,  $a_i = sm_i$ ;
- $b_i$ : via onde está posicionado o trem  $i$ . Caso  $i$  seja uma atividade de manutenção,  $b_i = km_i$ ;
- $e_i = 1$  se o trem está parado em uma via ou 0 caso contrário. Caso  $i$  seja uma atividade de manutenção,  $e_i = 1$ ;

### 2.3.5 Especificação algébrica do modelo.

A simulação é inicializada com a criação dos elementos físicos, como os segmentos, matrizes de conexão  $CL$  e  $CO$  e os conjuntos de trem a serem despachados,  $OL$  e  $LO$ . Cria-se também o conjunto de atividades de manutenção  $\Psi$ .

Após a inicialização dos elementos físicos do sistema seu estado inicial é determinado pelas seguintes equações:

$$x_i = \begin{cases} ip_i & i \in OL \cup LO \\ z2_i & i \in \Psi_{tf} \end{cases} \quad (2.1)$$



$$a_i = \begin{cases} sp_i & i \in OL \cup LO \\ sm_i & i \in \psi_{tf} \end{cases} \quad (2.2)$$

$$b_i = \begin{cases} kp_i & i \in OL \cup LO \\ km_i & i \in \psi_{tf} \end{cases} \quad (2.3)$$

$$e_i = 1 \quad i \in \eta \quad (2.4)$$

Após a determinação do estado inicial o passo seguinte é determinar o próximo evento a ser tratado, ou seja, aquele que ocorre no instante  $x_l$  e envolve o trem  $l$ , conforme a expressão (2.5).

$$x_l = \min_{i \in \eta} \{x_i\} \quad (2.5)$$

Caso o trem  $l$  tenha chegado ao seu destino ou seja um trem fictício responsável por uma atividade de manutenção, ele é removido do sistema e o próximo evento deve ser selecionado pela expressão (2.5).

Após a escolha do próximo evento deve verificar-se se existe uma atividade de manutenção a ser realizada neste período.

A verificação de ocorrência de atividades de manutenção é feita comparando-se o instante de tempo do próximo evento,  $x_l$ , com o intervalo de tempo no qual a via para onde o trem se destina estará interrompida, conforme expressão (2.6).

$$z1_i \leq x_l \leq z2_i, i \in \psi_{tf} \quad (2.6)$$

Para todo  $i \in \psi_{tf}$  que satisfaça a condição (2.6) cria-se um trem fictício  $tf$  referente à atividade de manutenção, conforme as expressões abaixo:

$$x_{tf} = z2_i \quad (2.7)$$

$$a_{tf} = sm_i \quad (2.8)$$

$$b_{tf} = km_i \quad (2.9)$$

$$e_{tf} = 1 \quad (2.10)$$

O trem fictício  $tf$  criado é inserido no conjunto  $\eta$  e a manutenção  $r$  é retirada do conjunto  $\Psi$ .

$$\eta = \eta \cup \{tf\} \quad (2.11)$$

$$\Psi = \Psi - r \quad (2.12)$$

Após a criação do trem  $tf$  o próximo evento deve ser escolhido pela expressão (2.5). Caso nenhum trem de manutenção tenha sido criado o tratamento do evento selecionado é feito pela rotina de despacho.

### 2.3.6 Rotina de despacho.

A rotina de despacho decide se o trem  $l$  pode seguir o seu trajeto ou se deve ser mantido em sua via atual. Para que um trem possa seguir seu trajeto, deve-se encontrar uma via  $k_1$  para onde ele possa ser deslocado sem ocasionar bloqueio na linha.

O Algoritmo 2.1 resume a rotina de despacho. Detalhes sobre a sintaxe utilizada para descrever os algoritmos estão no Apêndice B.

Pode-se verificar no Algoritmo 2.1 que o fato de uma via estar disponível não significa que o trem possa ocupá-la. Deve-se garantir que o deslocamento do trem para a via não ocasionará bloqueio na linha. Um algoritmo para prevenção de bloqueio é apresentado em Rondón (2000).

No planejamento de circulação de trens a principal decisão é a escolha do trem preferencial entre o conjunto de trens que pretendem ocupar um mesmo segmento. É essa decisão o maior fator de influência na qualidade do planejamento obtido.

Em ferrovias com grande fluxo de trens, a concorrência entre trens ocorre com uma frequência muito alta. A rotina de despacho deve tomar as decisões apropriadas para a obtenção de um planejamento que atenda os objetivos da ferrovia

Cada vez que um trem deseja movimentar para outro segmento é necessário verificar se existem trens concorrentes, isto é, trens que competem pela ocupação de um mesmo segmento  $s_1$  em um mesmo intervalo de tempo. Esse intervalo de tempo pode ser

definido como  $[x_l, x_l + \Delta]$ , onde  $\Delta$  é o intervalo tempo necessário para o trem percorrer o segmento seguinte ao seu segmento atual:

$$\Delta = t_{l,s_1,k_1} \quad (2.14)$$

```

procedimento rotinaDeDespacho(l) retorna sucesso ou falha

entradas:
l //trem sendo despachado
variáveis locais:
s1 //próximo segmento do trem l
concorrentes //lista dos trens concorrendo pelo segmento s1 no mesmo
                instante de tempo
tremPreferencial //trem selecionado como preferencial
KT //lista das vias de s1 que podem ser acessadas pelo trem l
k1 //via pertencente ao conjunto KT

inicio
    s1 ← proximoSegmentoDoTrem (l):
    concorrentes ← encontraConcorrendo (s1)
    tremPreferencial ← escolheTremPreferencial(concorrentes, l)
    KT ← encontraViasDoSegmentoQuePodemSerOcupadasPeloTrem(s1, l)
    enquanto (KT ≠ ∅):
        k1 ← selecionaViaDoConjunto(KT)
        Se deslocamentoDoTremCausaBloqueio(k1, tremPreferencial) então
            KT = KT - {k1}
            k1 ← null
        Caso contrário
            despachaTremParaVia(k1, tremPreferencial)
            retorna sucesso

    Se k1 = null então retorna falha.
fim

```

Algoritmo 2.1: Rotina de despacho.

onde  $t_{l,s_1,k_1}$  é o tempo necessário para o trem  $l$  percorrer a via  $k_1$  do segmento  $s_1$  e  $t_{l,s_2,k_1}$  é o tempo necessário para o trem  $l$  percorrer a via  $k_1$  do segmento  $s_2$ .

O conjunto de trens  $\Phi$  que concorrem pelo uso do segmento  $s_1$  no intervalo de tempo requisitado pelo trem  $l$  é:

$$\Phi_{s_1,x_l,\Delta} = \{i \mid x_l \leq Xa_{i,s_1} \leq x_l + \Delta, i \in OL \cup LO\} \quad (2.15)$$

onde  $Xa_{i,s_1}$  é o instante no qual o trem  $i$  chegará ao segmento  $s_1$ .

A escolha do trem preferencial entre os trens concorrentes é, como foi dito antes, uma etapa fundamental para a qualidade do planejamento. A base das propostas deste trabalho está em desenvolver algoritmos para seleção do trem preferencial de maneira a obter um planejamento com qualidade, como será visto nos próximos capítulos.

Para que o deslocamento do trem  $l$  para a via  $k_1$  do segmento  $s_1$  seja possível, algumas restrições devem ser satisfeitas.

- existência de conexão física entre  $b_l$  e  $k_1$ ;
- a via está desocupada;
- a via não está reservada para outro trem.

Nos casos em que o trem  $l$ , o que está sendo despachado, é o trem selecionado dentre os trens que concorrem pelo uso do segmento  $s_1$ , este se desloca para a via  $k_1$  e o estado do sistema é atualizado:

$$a_l = s_1 \quad (2.16)$$

$$b_l = k_1 \quad (2.17)$$

$$e_l = 0 \quad (2.18)$$

$$x_l = x_l + t_{l,s_1,k_1} + \tau_{l,s_1} \quad (2.19)$$

Onde  $t_{l,s_1,k_1}$  é o tempo necessário para o trem  $l$  percorrer a via  $k_1$  do segmento  $s_1$ ;  $\tau_{l,s_1}$  é o tempo necessário para o trem  $l$  desacelerar e acelerar ao entrar e partir do segmento  $s_1$ . Quando um trem entra ou parte de um segmento é necessário que ele reduza sua velocidade por questões de segurança.

Quando a rotina de despacho não encontra uma via disponível para o trem se deslocar, este é mantido na sua via atual. As seguintes situações justificam por que uma via pode não ser encontrada para o trem:

- a. O deslocamento do trem  $l$  ocasiona bloqueio: nesse caso é necessário encontrar o primeiro trem viajando no sentido contrário ao do trem  $l$ . Se este trem pode ser despachado sem ocasionar bloqueio, o trem  $l$  não desloca até o instante do próximo evento deste trem. Caso contrário, procura-se o próximo trem viajando no sentido contrário até encontrar um trem que eventualmente bloqueia o trem  $l$ .
- b. Existem trens ocupando e reservando vias: nesse caso, o trem  $l$  não é movimentado até que um trem libere uma via, ou seja, obtém-se o menor instante no qual os trens que reservam vias liberam suas vias e o menor instante no qual os trens que ocupam as vias as liberam. O menor desses dois instantes é o instante correspondente do próximo evento do trem  $l$ .
- c. Existem apenas trens ocupando vias: nesse caso o trem  $l$  não se move até que um dos trens desocupe uma das vias.
- d. Existem apenas trens reservando vias: nesse caso o trem  $l$  não se move até que um dos trens libere a via correspondente.

## 2.4 Modelo de Programação Linear Inteira-Mista.

Esta seção resume uma formulação de um modelo de programação linear e mista para resolver o problema planejamento de circulação proposto por Valle *et al.* (2005). Este modelo estende o proposto por Higgins *et al.* (1996)

Considere o seguinte conjunto de trens  $I = \{1, 2, \dots, m, m+1, \dots, N\}$  no qual trens em sentido oeste são representados pelos índices de 1 a  $m$  e em sentido leste de  $m+1$  a  $N$ .

Seja  $S = S1 \cup S2$  o conjunto de pátios e de segmentos simples; onde  $S1$  é o conjunto de segmentos singelos e  $S2$  é o conjunto de pátios de cruzamento.

O conjunto  $Sm \subset S$  representa o conjunto de segmentos simples que estão em manutenção em determinados intervalos de tempo.

As variáveis inteiras utilizadas para decidir qual trem ocupa primeiramente um segmento ou para definir quando um segmento está em manutenção são:

$A_{i,j,s} = 1$  se o trem  $i$ ,  $i \leq m$ , ocupa o segmento  $s \in S1$  antes do trem  $j$ ,  $j \leq m$ ; ou 0 em caso contrário;

$B_{i,j,s} = 1$  se o trem  $i$ ,  $i \leq m$ , ocupa o segmento  $s \in S1$  antes do trem  $j$ ,  $j > m$ ; ou 0 em caso contrário;

$C_{i,j,s} = 1$  se o trem  $i$ ,  $i > m$ , ocupa o segmento  $s \in S1$  antes do trem  $j$ ,  $j > m$ ; ou 0 em caso contrário;

$Q_{i,s} = 1$  se o trem  $i \in I$  ocupa o segmento  $s \in Sm$  depois de um período após sua manutenção; ou 0 se o atravessa antes de a manutenção começar.

As variáveis de decisão quantos aos instantes de chegada e de partida dos trens são as seguintes:

$Xa_{i,s}$  = instante de chegada do trem  $i \in I$  no segmento  $s \in S$ ;

$Xd_{i,s}$  = instante de partida do trem  $i \in I$  do segmento  $s \in S$ ;

Os parâmetros de entrada são os seguintes:

$t_{i,s} \min$  = tempo mínimo para o trem  $i \in I$  atravessar o segmento  $s \in S$ ;

$t_{i,s} \max$  = tempo máximo para o trem  $i \in I$  atravessar o segmento  $s \in S$ ;

$t_{i,s}$  = tempo que o trem  $i \in I$  leva para atravessar o segmento  $s \in S$ ;

$K_s \min$  = instante de início da manutenção no segmento  $s \in S$ ;

$K_s \max$  = instante de termino da manutenção no segmento  $s \in S$ ;

$W_{i,s}$  = intervalo de tempo que o trem  $i \in I$  leva para executar atividades no segmento  $s \in S$ ;

$ip_i$  = instante planejado de partida do trem  $i \in I$  de seu segmento de origem;

$ic_i$  = horário planejado de chegada do  $i \in I$  trem no seu segmento de destino;

$E_{i,s}$  = intervalo de tempo para o trem  $i \in I$  entrar no pátio  $s \in S$ ;

O objetivo adotado nesta abordagem do problema é minimizar o tempo total de paradas dos trens. A função objetivo para este problema possui a seguinte forma:

$$\min \sum (Xa_{i,sp_i} - Xa_{i,sd_i}) \quad (2.20)$$

Onde  $sd_i$  é o segmento de destino do trem  $i \in I$ .

Para modelar as restrições, consideremos a linha ferroviária da Figura 2.10. Note que os índices dos pátios e dos segmentos aumentam em sentido leste.

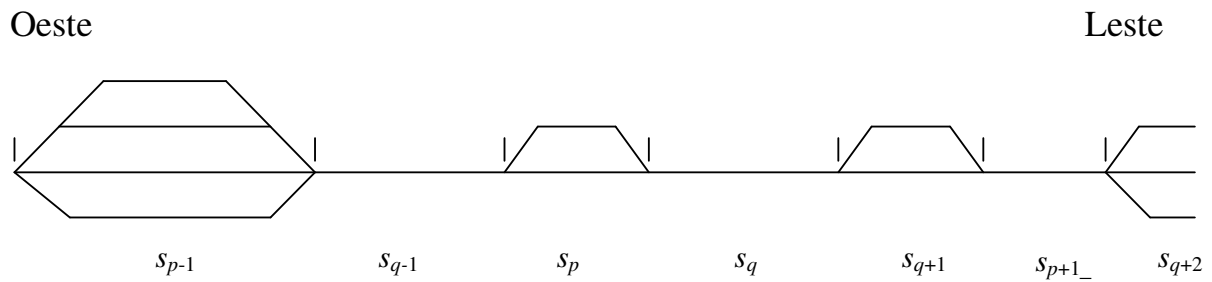


Figura 2.10: Linha ferroviária com pátios de cruzamento e segmentos singelos.  $s_{p-1}, s_p, s_{p+1} \in S1$ ;  $s_{q-1}, s_q, s_{q+1}, s_{q+2} \in S2$

Restrições para o caso em que trens que se movem em sentidos opostos se cruzam nos pátios de cruzamento e ultrapassagem são as seguintes:

$$Xd_{i,s_{q+1}} + M * B_{i,j,s_p} \geq Xa_{j,s_{q+1}} + E_{j,s_{q+1}} \quad (2.21)$$

$$Xd_{i,s_q} + M * (1 - B_{i,j,s_p}) \geq Xa_{j,s_q} + E_{j,s_q} \quad (2.22)$$

$$\forall s \in S1; j > m, i \leq m$$

A restrição 2.21 indica que se o trem  $j \in I, j > m$ , parte antes, o trem  $i \in I, i \leq m$ , só pode partir do pátio  $s_{q+1} \in P2$  depois que o trem  $j \in I$  chegar e entrar no pátio  $s_{q+1} \in P2$ . A restrição (2.22) é similar, considerando que o trem  $i \in I$  parte antes.

As restrições que interrompem o tráfego para trens no sentido leste em segmentos sob manutenções são:

$$Xd_{i,s_q} + M * (1 - Q_{i,s_p}) \geq Xa_{j,s_{q+1}} \quad (2.23)$$

$$Xd_{i,s_q} + M * (1 - Q_{i,s_p}) \geq Xa_{j,s_{q+1}} \quad (2.24)$$

$$\forall s_p \in Sm, i > m$$

E para trens no sentido oeste:

$$Xd_{i,s_{q+1}} + M * (1 - Q_{i,s_p}) \geq K_{s_p} \max \quad (2.25)$$

$$Xa_{i,s_q} + M * Q_{i,s_p} \leq K_{s_p} \min \quad (2.26)$$

$$\forall s_p \in Sm, i \leq m$$

As restrições (2.23) e (2.24) indicam que para atravessar um segmento  $s \in Sm1$ , ou o trem  $i \in I, i > m$ , parte do pátio  $s \in S2$  depois do término da manutenção de  $s \in Sm1$ , ou o ele chega ao pátio  $s_{q+1} \in S2$  antes que comece a manutenção  $s \in Sm1$ . De maneira análoga, as restrições (2.25) e (2.26) são definidas restrições para trens em sentido oeste.

Para as restrições de (2.21) a (2.24), a constante  $M$  é escolhida grande o suficiente para que todas as restrições sejam satisfeitas.

As restrições de tempo para trens no sentido leste percorrer os segmentos são as seguintes:

$$Xa_{i,s_{q+1}} - Xd_{i,s_q} \geq T_{i,s_p} \min \quad (2.27)$$

$$Xa_{i,s_{q+1}} - Xd_{i,s_q} \leq T_{i,s_p} \max \quad (2.28)$$



$$\forall s_p \in S1, i > m$$

Para trens no sentido oeste:

$$Xa_{i,s_q} - Xd_{i,s_{q+1}} \geq T_{i,s_p} \min \quad (2.29)$$

$$Xa_{i,s_q} - Xd_{i,s_{q+1}} \leq T_{i,s_p} \max \quad (2.30)$$

$$\forall s_p \in S1, i \leq m$$

A restrição (2.27) indica o tempo mínimo para o trem  $i \in I, i > m$ , percorrer o segmento  $s_p \in S1$ ; enquanto que a restrição (2.27) indica o tempo máximo para o trem  $i \in I$  percorrer o segmento  $s_p \in S1$ . De maneira análoga, as restrições (2.29) e (2.30) são definidas restrições para trens em sentido oeste.

As restrições que modelam ultrapassagens e garantem que elas ocorrerão nos pátios de cruzamento e ultrapassagem para trens no sentido leste são:

$$Xd_{i,s_q} + M * C_{i,j,s_p} \geq Xa_{j,s_{q+1}} \quad (2.31)$$

$$Xd_{j,s_q} + M * (1 - C_{i,j,s_p}) \geq Xa_{i,s_{q+1}} \quad (2.32)$$

$$\forall s_p \in S1, i, j > m$$

Para trens no sentido oeste:

$$Xd_{i,s_{q+1}} + M * A_{i,j,s_p} \geq Xa_{j,s_q} \quad (2.33)$$

$$Xd_{j,s_{q+1}} + M * (1 - C_{i,j,s_p}) \geq Xa_{i,s_q} \quad (2.34)$$

$$\forall s_p \in S1, i, j \leq m$$

A restrição (2.32) indica que se o trem  $j \in I$  parte primeiro, então o trem  $i \in I$  só pode partir do pátio  $s_q \in S2$  depois que o trem  $j \in I$  chegar no pátio  $s_{q+1} \in S2$ . Isto garante a ocorrência de no máximo um trem na mesma direção em cada segmento. A restrição (2.33) é similar exceto que considera o trem  $i \in I$  indo primeiramente. Análise similar é válida para as restrições criadas para trens em sentido oeste (2.33) e (2.34).

As restrições de tempo para os trens percorrer os pátios de cruzamento e ultrapassagem são as seguintes:

$$Xd_{i,s_q} - Xa_{i,s_q} \geq E_{i,s_q} + S_{i,s_q} + T_{i,s_q} \quad (2.35)$$

$$\forall s_p \in S2, i \in I$$

A restrição (2.35) indica que o instante de partida do trem  $i \in I$  do pátio  $s_q \in S2$  é no mínimo igual ao instante de chegada neste pátio mais o tempo que o trem necessita para entrar no pátio, mais o tempo necessário para realizar atividades, se houverem, neste pátio, mais o tempo que o trem necessita para atravessá-lo.

A garantia de que trens não partem antes do horário planejado, requer as seguintes restrições:

$$Xa_{i,sp_i} \geq ip_i \quad (2.36)$$

Onde  $sp_i \in S2$  é o segmento de origem do trem  $i \in I$ .

## 2.5 Algoritmo Genético.

O algoritmo genético apresentado por Mendes, 2005, utiliza o modelo do simulador baseado em eventos discretos apresentado na seção 2.3. A Figura 2.11 mostra onde o código genético é utilizado no simulador.

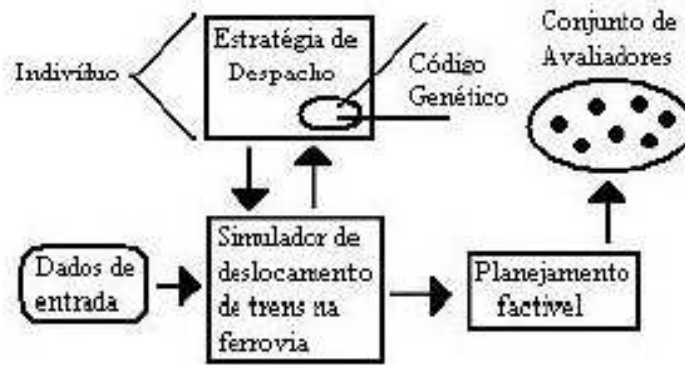


Figura 2.11: Algoritmo genético e o simulador.

Basicamente, os dados de entrada para o simulador são: topologia da ferrovia, horário e ponto de partida de todos os trens, percurso de cada trem na ferrovia, estratégia de despacho, programa de atividades de cada trem e eventos externos ao simulador (perda de potência, quebra de trens, etc.).

O modelo baseado em eventos discretos simula o processo de despacho e movimentação dos trens na ferrovia de forma detalhada. Quando um trem está para ser despachado, todos os conflitos com outros trens competindo pelo uso do mesmo segmento são resolvidos. A Estratégia de Despacho decide qual trem possui maior preferência quando esses conflitos ocorrem. Diferentes estratégias implicam diferentes decisões de despacho e, portanto, diferentes planejamentos. Os planejamentos são avaliados de acordo com as funções objetivo. A idéia do algoritmo é evoluir uma população de indivíduos, cada um deles representando uma estratégia de despacho, até se obter um planejamento com decisões de despacho que sejam satisfatórias, em um intervalo de tempo aceitável, definido pelo planejador.

Cada indivíduo é representado por de um cromossomo composto por  $2N$  vetores, onde  $N$  é o número de trens presentes na simulação da ferrovia no horizonte de planejamento, 2 vetores por trem. O tamanho de cada vetor é o número de segmentos  $S$  que compõe o percurso do trem. No primeiro vetor, denominado vetor de velocidades, cada elemento define a velocidade com a qual o respectivo trem percorre o correspondente segmento em seu percurso. No segundo vetor, denominado vetor de prioridades, cada elemento define a prioridade que o trem possui para percorrer o correspondente segmento quando competir pelo uso dele com outros trens. Os valores dos componentes de ambos vetores são números reais positivos. Portanto, durante a simulação, se conflitos ocorrerem, a estratégia de despacho escolhe os trens utilizando seus vetores de prioridades e decide a velocidade com a qual o trem a ser despachado percorrerá o segmento desejado.

O custo computacional para gerar e avaliar cada planejamento cresce quando a complexidade do planejamento cresce, i.e., quando as linhas crescem assim como o número de trens e conflitos. A complexidade influencia o tamanho da população, caso cada indivíduo deva ser avaliado diretamente. O uso de técnicas de agrupamentos é uma alternativa para reduzir o número de avaliações diretas e, conseqüentemente, o tempo de

execução do algoritmo genético, sem comprometer significativamente a qualidade da solução do problema de otimização.

A idéia básica é realizar a etapa de avaliação da população na seguinte sequência: agrupa-se os indivíduos da população usando, por exemplo, o algoritmo *Fuzzy Cmeans*. É importante notar que o agrupamento é feito no espaço genotípico, antes de avaliar os indivíduos. Após, escolhem-se os indivíduos representativos de cada grupo, por exemplo, os indivíduos mais próximos dos centros de cada grupo, para serem avaliados diretamente. Os valores da função objetivo correspondente aos indivíduos restantes são estimados a partir dos valores da função objetivo correspondente aos elementos representativos. Um estudo comparativo sobre o custo computacional e da qualidade da solução do algoritmo genético utilizando *Fuzzy Cmeans* e do algoritmo genético clássico é feito em Mendes, 2005.

O algoritmo genético proposto por Mendes, 2005, utiliza a idéia de agrupamento e se mostrou mais rápido que um algoritmo genético convencional. Mantendo diversidade e obtendo, em média, planejamentos com qualidade similar ou superior que o algoritmo genético convencional.

## **2.6 Resumo.**

Este capítulo apresentou o problema de planejamento de circulação de trens, com suas restrições e objetivos. Foi feita uma revisão dos principais trabalhos existentes na literatura para o problema de circulação de trens. Por fim foi apresentada a modelagem dos elementos de uma malha ferroviária e foi descrito o funcionamento do simulador a eventos discretos utilizado neste trabalho.

O próximo capítulo apresenta propostas para decidir sobre a preferência entre trens.

## Capítulo 3

### Algoritmos Nebulosos de Decisão

Esse capítulo tem o propósito de apresentar dois algoritmos para o planejamento de circulação de trens. Um algoritmo decide a preferência de trens com uso de conjuntos nebulosos, o objetivo é manter os trens próximos a uma referência de percurso. Um segundo algoritmo decide a preferência entre trens e também a velocidade dos trens para que estes se reaproximem de suas referências. Por fim, exemplos ilustrativos são apresentados.

#### 3.1 Introdução

No controle de tráfego ferroviário, despachadores e controladores planejam a movimentação dos trens na linha e tomam as decisões quanto à preferência dos trens nas situações de cruzamentos e ultrapassagens.

Ferramentas de planejamento e auxílio à tomada de decisão, como o simulador a eventos discretos apresentado no capítulo anterior, precisam utilizar critérios que expressem de maneira condizente a realidade do processo operacional ferroviário para produzir planejamentos de circulação que atendam da melhor forma possível os objetivos da ferrovia.

Como foi explicado no capítulo anterior, obter uma solução para o problema de planejamento de circulação de trens consiste em determinar, para todos os trens, os instantes de chegada e partida em cada segmento de seu percurso de maneira que as restrições de cruzamento, ultrapassagem, ocupação e bloqueio sejam respeitadas.

A principal decisão envolvida nesse processo é a de determinar qual é o trem preferencial, isto é, qual trem tem a preferência para ocupar uma via quando mais de um trem deseja ocupá-la no mesmo intervalo de tempo. São essas decisões de planejamento da circulação que determinam a sua qualidade. Além disso, uma estratégia de tomada de decisão de despacho rápida é fundamental para o planejamento em tempo real,

Qualidade de planejamento é um termo que deve ser tratado com cautela. Em situações reais o planejamento é feito conciliando objetivos conflitantes como reduzir o tempo de percurso dos trens, reduzir gasto de combustível, atender as demandas de carga e descarga, respeitar as escalas dos maquinistas, manutenções da linha, etc., além da reação à ocorrência de perturbações. Em geral não se observa uma função objetivo específica de uso geral para medir a qualidade do planejamento.

Uma medida muito usada para avaliar um planejamento é o tempo total de paradas dos trens devido a cruzamentos e ultrapassagens. Por ser uma grandeza fácil de ser medida e de grande importância prática, ela é a mais presente em publicações acadêmicas sobre o tema e será também adotada neste trabalho.

Este capítulo sugere dois algoritmos heurísticos para a decisão de preferência entre trens em tempo real. O primeiro algoritmo tem como objetivo decidir apenas a preferência entre trens que disputam a ocupação de um mesmo segmento simultaneamente. O segundo algoritmo objetiva decidir a preferência de ocupação dos trens e a velocidade com a qual o trem preferencial deve percorrer o segmento desejado.

### 3.2 Algoritmos Heurísticos de Decisão de Preferência entre Trens.

Como não existe uma regra universal que oriente a decisão sobre a preferência entre trens e que resulte no melhor planejamento para dado objetivo, as propostas de algoritmos de despacho de trens são, em sua maioria, heurísticas orientadas pelo objetivo.

Peterson e Taylor (1982), propõe uma rotina de despacho baseada na prioridade estática<sup>1</sup> de um trem, no instante que o trem chega a uma determinada via e no tempo que o trem levará para percorrer esta via. Com base nisso é definida a seguinte expressão de escolha de trem.

$$y_j = \min_j \{ Xa_{j,s} + t_{s_1,j,k_1} + \alpha p_j \} \quad (3.1)$$

O trem  $j$  escolhido para ocupar a via  $k_1$  é o trem com menor valor de  $y_j$ , Na expressão (3.1)  $Xa_{j,s_1}$  é o instante que o trem  $j$  chega ao segmento  $s_1$ ,  $t_{j,s_1,k_1}$  é o tempo

---

<sup>1</sup> Prioridade estática é uma prioridade estabelecida *a priori* de acordo com a importância estratégica ou o valor econômico que o trem representa.

que o trem  $j$  leva para atravessar a via  $k_1$  do segmento  $s_1$ ,  $p_j$  é a prioridade estática do trem  $j$  e  $\alpha$  é uma constante positiva.

Em Rondón (2000) é proposta uma solução baseada na Teoria de Conjuntos Nebulosos (Yuan e Klir, 1995; Yen e Reza, 1998; Pedrycz e Gomide, 1998; Passino e Yurkovich, 1998). Uma prioridade é dada a um trem de acordo com o seu atraso (tempo de paradas para cruzamento e ultrapassagem). A variável lingüística Atraso tem os conjuntos nebulosos, *Pequeno*, *Médio* e *Alto*, e a variável Prioridade tem os conjuntos *Baixa*, *Média* e *Alta* associados aos seus termos. O trem com maior prioridade é o preferencial para percorrer uma via.

Assim temos:

Atraso = (*Pequeno*, *Médio*, *Baixo*);

Prioridade = (*Baixa*, *Média*, *Alta*);

As seguintes regras nebulosas são sugeridas para avaliar a circulação de cada trem:

Se o Atraso é *Pequeno* então a Prioridade é *Baixa*.

Se o Atraso é *Médio* então a Prioridade é *Média*.

Se o Atraso é *Alto* então a Prioridade é *Alta*.

Em Gomide *et al*, 1999, é apresentada uma solução similar, baseada também na Teoria De Conjuntos Nebulosos. Cada trem possui, baseado em suas características, uma referência de tempo de percurso, sendo que o trem pode estar adiantado ou atrasado em relação a essa referência. Existe também um limite superior e um limite inferior para a posição do trem em relação a sua referência. Os atrasos e avanços são classificados com uso de conjuntos nebulosos, sendo estes ND = *no delay*, SD = *small delay*, BD = *big delay*, SA = *small advance*, BA = *big advance*. Quando dois trens, A e B, competem por uma via é construída uma tabela de decisão com três regiões, a de prioridade do trem A, a de prioridade do trem B e uma região duvidosa. A decisão sobre essa região duvidosa é feita baseada em outras regras como tempo de parada e regularidade de tráfego.

### **3.3 Decisão de Preferência de Trens.**

#### **3.3.1 Princípios do algoritmo.**

Quando é feita a programação de um trem, define-se sua carga, o número de vagões, as locomotivas e também os horários de partida e chegada que o trem deverá cumprir para que sua carga chegue no destino dentro do prazo estipulado. Após a partida do trem, o controlador deve planejar os cruzamentos e ultrapassagens de trens com o objetivo de cumprir os horários desejados de chegada de cada trem de forma eficiente.

O algoritmo sugerido nesta seção considera que cada trem possui uma referência de percurso. Essa referência pode ser determinada pelos horários desejados de chegada e partida de um trem em cada segmento não levando em consideração as paradas para cruzamentos e ultrapassagens, o que é denominado o Programa de Atividades de Trem, e também pode ser determinada por algum algoritmo de otimização como os derivados do modelo de programação linear inteira mista ou algoritmo genético.

A idéia principal do algoritmo é orientar, durante a simulação, a decisão local de qual trem tem preferência em relação aos outros se baseando nas referências globais de percurso. O objetivo é manter o percurso dos trens o mais próximo possível de sua referência (Tazoniero *et al.*, 2005, Valle *et al.*, 2005a).

Para cada trem e sua referência é definido um limite superior e um limite inferior de tempo de percurso, que determinam uma região na qual o percurso do trem está condizente com sua referência. Se o percurso do trem está fora dos limites, sua situação não é condizente com sua referência. A Figura 3.1 ilustra estes princípios.

O limite inferior é otimista, pois representa o menor tempo de percurso que o trem pode realizar e o limite superior representa o maior atraso que o trem pode sofrer na sua partida para que se mantenha consistente com a referência, quando o trem faz o percurso com o menor tempo de percurso. Quando o percurso de um trem está contido na região entre seu limite inferior e a referência, a sua situação é de avanço com relação a sua referência, isto é, o trem está adiantado. Por outro lado, quando o percurso de um trem está contido na região entre a referência e o limite superior, o trem está atrasado, isto é, está com



perda de percurso, mas recuperável. Um trem com percurso à direita do limite superior está definitivamente prejudicado.

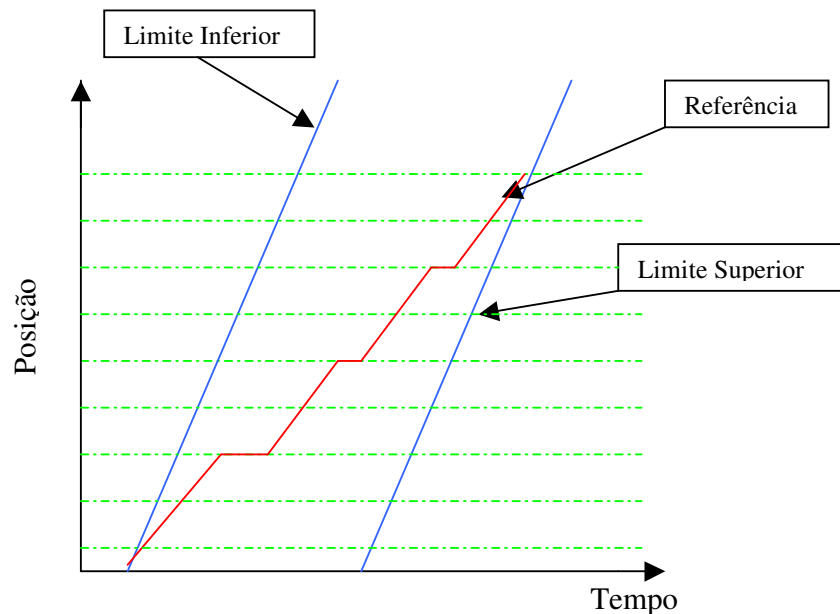


Figura 3.1: Limite inferior e superior para a referência de percurso de um trem.

### 3.3.2 Modelagem com a teoria de conjuntos nebulosos.

Em certo instante de seu percurso um trem está em certa posição na linha ferroviária. Comparando esta posição do trem com sua referência de percurso o trem pode estar em três situações diferentes: atrasado, adiantado e na referência.

A teoria de conjuntos nebulosos se apresenta como uma ferramenta muito apropriada para tratar este problema. Podemos modelar a posição do trem em relação a sua referência com o uso de variáveis lingüísticas (Yuan e Klir, 1995; Yen e Reza, 1998; Pedrycz e Gomide, 1998; Passino e Yurkovich, 1998), como Percurso. Os seguintes conjuntos podem ser considerados como valores da variável lingüística Percurso: *Muito Adiantamento* (MD), *Pouco Adiantamento*(PD), *Sem Atraso*(SA), *Pouco Atraso*(PS) e *Muito Atraso*(MS). A Figura 3.2 mostra um exemplo de como estes conjuntos nebulosos são representados.

Por exemplo, suponha um trem  $i$  cujo percurso é do segmento  $s_a$  para o segmento  $s_b$ , conforme Figura 3.3. O trem deve partir de  $s_a$  às 3 horas e chegar a  $s_b$  às 12 horas. Durante

o percurso o trem passa pelo segmento  $s_c$ , entre  $s_a$  e  $s_b$ . Para que o trem cumpra seu horário desejado de chegada ele deve passar por  $s_c$  às 9 horas, com limite inferior às 8 horas e limite superior às 10 horas.

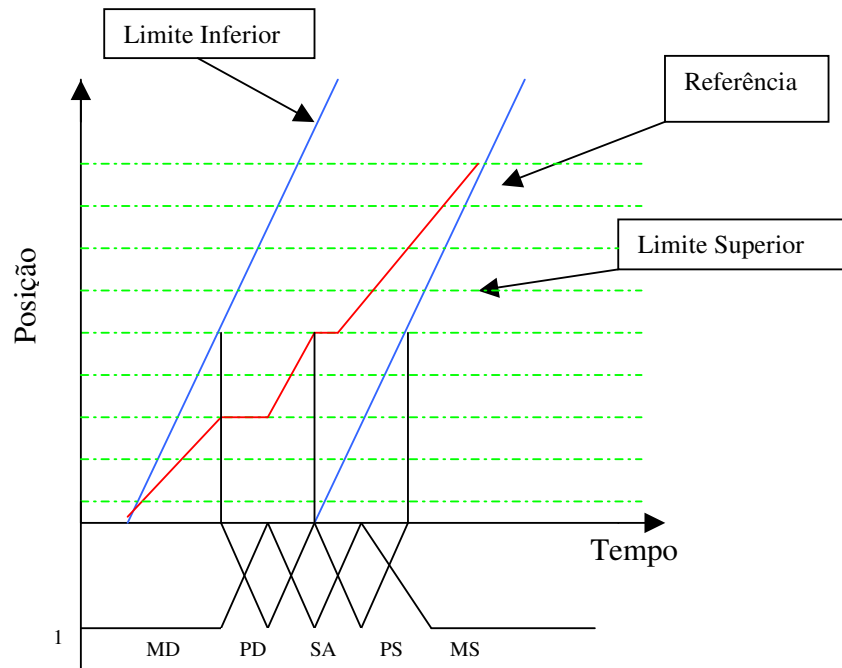


Figura 3.2: Representação dos valores associados à variável lingüística Percurso.

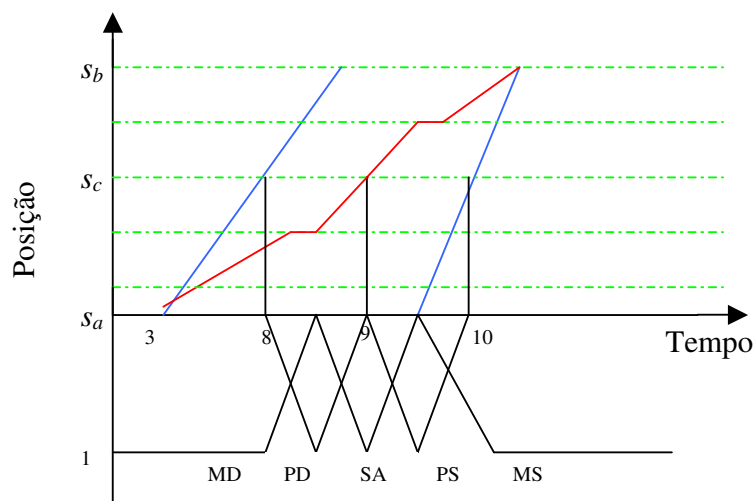


Figura 3.3: Exemplo de conjuntos nebulosos para a posição do trem.

Os conjuntos *Muito Adiantamento* (MD) e *Muito Atraso* (MS) são representados por Funções de Pertinência Trapezoidais, enquanto os conjuntos *Pouco Adiantamento* (PD), *Sem Atraso* (SA) e *Pouco Atraso* (PS) são representados por Funções de Pertinência Triangulares.

Supondo que o trem chegue no segmento  $s_c$ , 9.33h, a posição atual com relação à sua referência é compatível com o conjunto *Sem Atraso* com grau 0.33 e ao conjunto *Pouco Atraso* com grau 0.67, conforme Figura 3.4. As funções de pertinência dos conjuntos nebulosos são denotadas por  $\mu(.) (Xa_{i,s}) \rightarrow [0,1]$ , onde (.) é MD, PD, SA, PS ou MS. No exemplo da Figura 3.4:

$$\mu(SA)(Xa_{i,s_c}) = 0.33 \quad (3.2)$$

$$\mu(PS)(Xa_{i,s_c}) = 0.67 \quad (3.3)$$

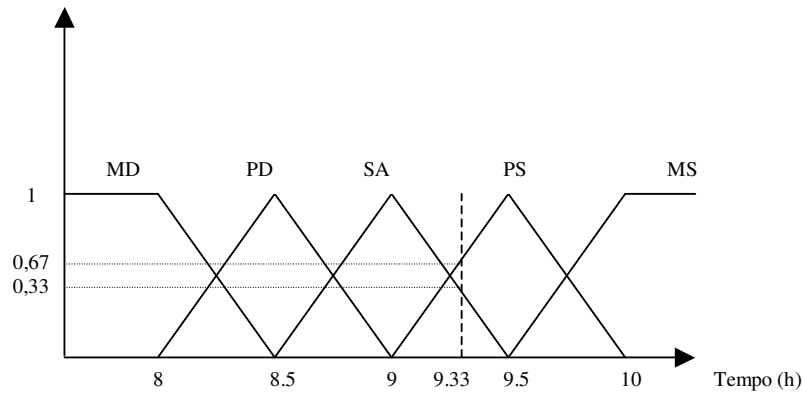


Figura 3.4: Classificação da posição atual do trem nos Conjuntos Nebulosos.

Dessa mesma maneira podemos, para qualquer posição do trem em qualquer instante, determinar o grau de compatibilidade de seu percurso com a sua referência. A próxima seção mostra como estes conceitos são utilizados para criar uma base regras para a decisão de preferência entre trens.

### 3.3.3 Base de regras para decisão de prioridade.

Durante o deslocamento do trem pela ferrovia vai haver situações nas quais dois ou mais trens competem pela ocupação simultânea pelo mesmo segmento. Nestas situações deve-se decidir qual trem tem a preferência sobre o outro.

Como foi visto na seção anterior, podemos caracterizar o percurso de um trem em relação a sua referência em qualquer momento do seu movimento de acordo com a variável lingüística Percurso e seus valores *Muito Adiantamento*, *Pouco Adiantamento*, *Sem Atraso*, *Pouco Atraso* e *Muito Atraso*.

A estratégia de decisão para estabelecer a preferência entre trens competindo pela ocupação de um segmento pode ser representada por regras. Essas regras são construídas como objetivo de manter os trens o mais próximo possível de suas referências, pois esta é a circunstância em que se minimiza os atrasos.

Por exemplo, considere um trem, TREM1, adiantado em relação a sua referência concorrendo pela ocupação de um segmento  $s_a$  com um trem, TREM2, que está atrasado. Temos duas possibilidades de decisão. Em uma delas o TREM1 tem preferência, ocupa o segmento  $s_a$  e o trem atrasado permanece em seu segmento atual aguardando o cruzamento com o TREM2, por exemplo. A consequência dessa decisão seria que o TREM2, que já está atrasado em relação à sua referência, atrasaria ainda mais seu percurso enquanto o trem TREM1 continuaria adiantado. Na outra possibilidade, o trem TREM2 tem a preferência e ocupa o segmento  $s_a$  enquanto TREM1 atrasaria seu percurso. Dessa maneira TREM1 se aproximaria de sua referência enquanto o outro trem recuperaria o atraso, podendo até melhorar seu percurso em relação à referência.

A partir do objetivo, que é minimizar o atraso, e utilizando a variável lingüística Percurso e seus valores podemos construir a seguinte regra.

Se o TREM1 está *Muito Atraso* e o TREM2 está *Muito Adiantamento* então a Preferência é do TREM1.

As outras regras são construídas utilizando a mesma idéia. Com isso tem-se uma base de regras para a determinação da prioridade entre os trens e fazendo o mapeamento da posição dos trens nessa base de regras determina-se a preferência entre os trens para todas

as situações. A Tabela 3.1 mostra as regras e a Figura 3.5 mostra, esquematicamente, o mapeamento.

TREM1 \ TREM2	MD	PD	SA	PS	MS
MD	EMPATE	TREM1	TREM1	TREM1	TREM1
PD	TREM2	EMPATE	TREM1	TREM1	TREM1
SA	TREM2	TREM2	EMPATE	TREM1	TREM1
PS	TREM2	TREM2	TREM2	EMPATE	TREM1
MS	TREM2	TREM2	TREM2	TREM2	EMPATE

Tabela 3.1: Base de Regras para determinação do trem preferencial.

Pode-se observar na Figura 3.5 que existe uma região de empate, onde os percursos dos trens estão em circunstâncias idênticas, isto é, na mesma posição em relação às suas referências. Nestas situações a decisão deve ser tomada com base em outros critérios.

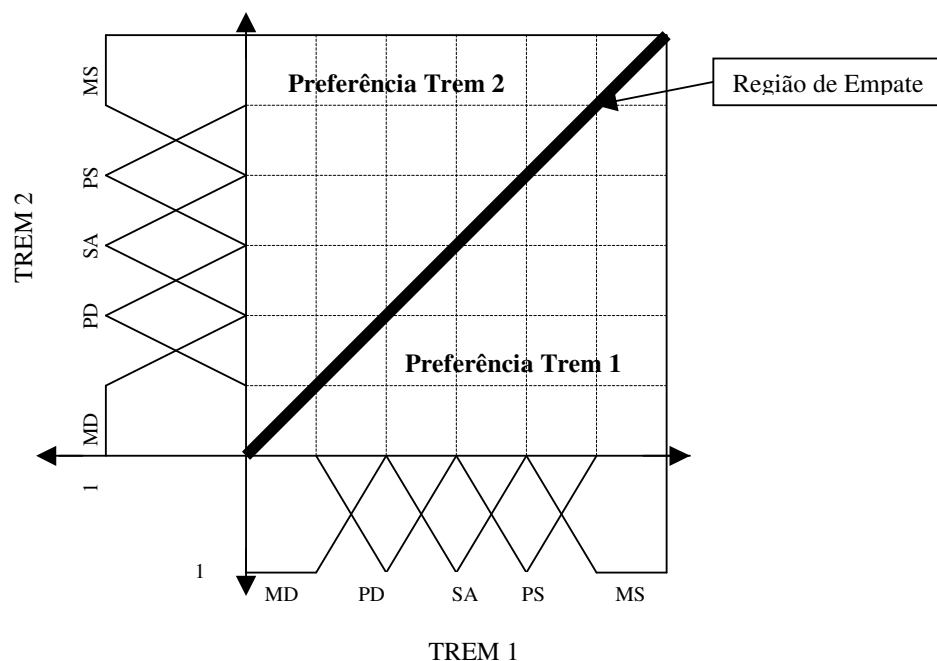


Figura 3.5: Representação da base de Regras para determinação do trem preferencial.

Neste trabalho o critério de desempate dá preferência para o trem cujo deslocamento manterá os trens em concorrência mais próximos às respectivas referências. O critério de desempate tem papel fundamental na qualidade do planejamento gerado pelo algoritmo.

Nos casos que não há perturbações no deslocamento dos trens, as decisões serão sempre das regras de desempate. Isso acontece porque os trens seguirão seus percursos na mesma velocidade que o percurso de referência e sempre chegarão aos segmentos nos mesmos instantes de suas referências. Uma decisão de desempate inadequada pode levar muitos trens a se afastarem de suas referências. O critério adotado neste trabalho garante que, não havendo perturbações no deslocamento dos trens e utilizando uma referência que considera as paradas obrigatórias e as paradas para cruzamento e ultrapassagem, a decisão sempre manterá o percurso dos trens sobre suas referências e com isso o algoritmo sempre obterá uma solução idêntica à referência.

### 3.3.4 Decisão.

Na seção anterior foi mostrada a base de regras para determinação do trem prioritário e as regras de desempate. Veremos agora como determinar as regras ativas a partir da caracterização do percurso dos trens quanto aos conjuntos nebulosos apresentados anteriormente e como fazer a decisão para determinar o trem preferencial.

Por exemplo, suponha que um trem  $i$  e um trem  $j$  concorrem pela ocupação de um mesmo segmento  $s_a$  e que os respectivos percursos são tais que:

$$\mu_{(PS)}(X_{a_i, s_a}) = 0.2$$

$$\mu_{(MS)}(X_{a_i, s_a}) = 0.8$$

$$\mu_{(PS)}(X_{a_j, s_a}) = 1$$

Segundo a base de regras, as seguintes regras estão ativas:

1. “Se o trem  $i$  está *Pouco Atraso* e o trem  $j$  está *Pouco Atraso* então a preferência é do trem  $j$ ”.
2. “Se o trem  $i$  está *Muito Atraso* e o trem  $j$  está *Pouco Atraso* então a preferência é do trem  $i$ ”.

Considerando que a conjunção da premissa é a *t-norma min*, o grau de ativação de cada regra (Yuan e Klir, 1995; Yen e Reza, 1998; Pedrycz e Gomide, 1998; Passino e Yurkovich, 1998), é dado por:

$$1. \lambda_1 = \min\{\mu_{(PS)}(Xa_{i,s_a}), \mu_{(PS)}(Xa_{j,s_a})\} = \min\{0.2, 1\} = 0.2$$

$$2. \lambda_2 = \min\{\mu_{(MS)}(Xa_{i,s_a}), \mu_{(PS)}(Xa_{j,s_a})\} = \min\{0.8, 1\} = 0.8$$

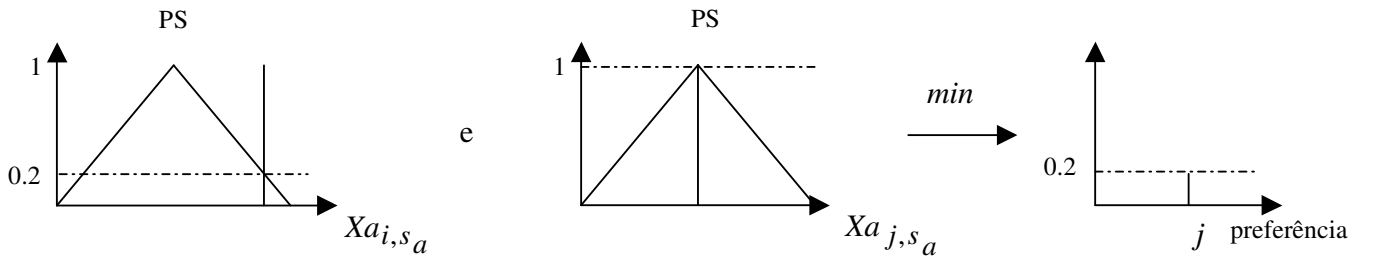
A Figura 3.6 mostra a representação das regras e a inferência nebulosa.

Utilizando o critério do máximo para defuzzificação (Passino e Yurkovich, 1998), expressão (3.2), o trem preferencial  $l$  será determinado pela regra com maior grau de ativação.

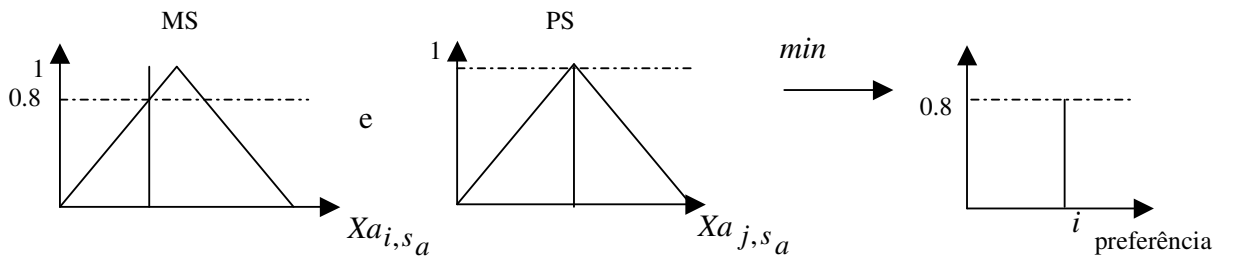
$$l = \{\arg \max_k \lambda_k\} \quad (3.2)$$

Esta é dada pela regra 2, pois:

$$\max\{\lambda_1, \lambda_2\} = \lambda_2 \quad (3.3)$$



Se  $Xa_{i,s_a}$  é *PD* e  $Xa_{j,s_a}$  é *SA* então a Preferência é trem  $j$ .



Se  $Xa_{i,s_a}$  é *PD* e  $Xa_{j,s_a}$  é *PS* então a Preferência é trem  $i$ .

Figura 3.6: Representação da das regras e inferência nebulosa para a preferência de trens.

Portanto, sendo a regra 2 a escolhida, então que o trem preferencial é o trem  $i$ . A escolha deste trem como sendo preferencial está de acordo com o raciocínio utilizado na construção das regras o qual, por sua vez, reflete o objetivo de minimizar o atraso. A Figura 3.7 mostra a decisão sugerida por cada regra.

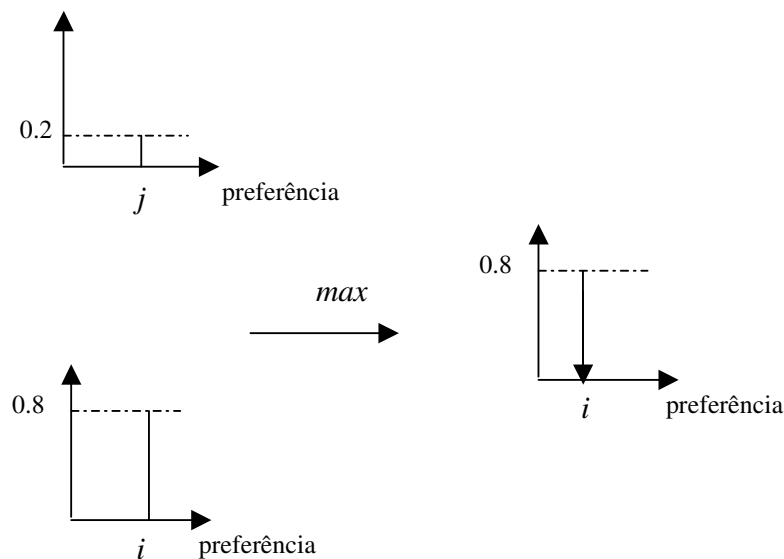


Figura 3.7: Representação da decisão para a preferência do trem.

Poderia se pensar em escolher o trem preferencial baseando-se apenas no valor absoluto do seu atraso ou adiantamento. Nesse caso quanto maior o atraso de um trem maior seria a sua preferência. A diferença do uso desse método para o método aqui apresentado, que utiliza conjuntos nebulosos, é que a preferência de um trem é ponderada pela sua referência e por seu atraso. Em situações práticas temos muitos casos que um pequeno atraso para certo trem tem uma importância maior que um grande atraso para outro trem. O uso de conjuntos nebulosos para a classificação do trem em sua referência consegue tratar esse tipo de situação e proporcionar uma solução mais adequada à realidade operacional de uma ferrovia.



### 3.3.5 Algoritmo de decisão de preferência de trens.

As seções anteriores apresentaram os princípios de funcionamento do algoritmo de decisão de preferência de trens, a caracterização do percurso do trem na variável *Percurso* assim como a determinação das regras ativas e defuzzificação para obtenção do trem preferencial. O Algoritmo 3.1 mostra o algoritmo de decisão de preferência de trens.

```
procedimento escolheTremPreferencial(trem1, trem2, s) retorna tremPreferencial
entradas:
trem1 e trem2 //trem
s //segmento disputado
variáveis locais:
referencial, referencia2 //referencias de percurso dos trens
pertinencias1[], pertinencias2[] //vetor com os valores das funções de pertinência para
a variável lingüística Percurso
listaRegrasAtivas: //lista com as regras ativas
uniaoDosConjuntosInferidos: //união dos conjuntos inferidos pelas regras ativas
baseDeRegras //base de regras para a preferência entre
chegadaSegmento1 //instante de chegada do trem no segmento s
percurso1 //conjuntos nebulosos da variável lingüística Percurso
chegadaSegmento2 //instante de chegada do trem no segmento s
conjuntosPercurso2 //conjuntos nebulosos da variável lingüística Percurso
regrasAtivas //lista com as regras ativas
preferencias //preferências inferidas pelas regras ativas

inicio
referencial ← trem1.referencia
referencia2 ← trem2.referencia
chegadaSegmento1 ← trem1.instanteQueOTremChegouNoSegmento(s)
conjuntosPercurso1 ← defineConjuntosNebulsosPercurso (referencial, s)
pertinencias1[] ← caracterizaPercurso (conjuntosPercurso1, chegadaSegmento1)
chegadaSegmento2 ← trem2.instanteQueOTremChegouNoSegmento(s)
conjuntosPercurso2 ← defineConjuntosNebulsosPercurso (referencia2, s)
pertinencias2[] ← caracterizaPercurso (conjuntosPercurso2, chegadaSegmento2)
regrasAtivas ← determinaRegrasAtivas (conjuntosPercurso1, conjuntosPercurso2)
preferencias ← fazInferencia (regrasAtivas, pertinencias1[], pertinencias2[])
tremPreferencial ← fazDefuzzificacao (preferencias)
retorna tremPreferencial

fim
```

Algoritmo 3.1: Algoritmo de decisão de preferência de trens.

O algoritmo de decisão de preferência de trens é chamado nas situações de conflitos entre trens, ou seja, quando dois ou mais trens competem pela ocupação de um mesmo segmento em um mesmo instante de tempo.

O simulador a eventos discreto e modelo de ferrovia apresentados no capítulo anterior simulam a movimentação dos trens na ferrovia. A Figura 3.8 mostra o fluxograma do processo de simulação.

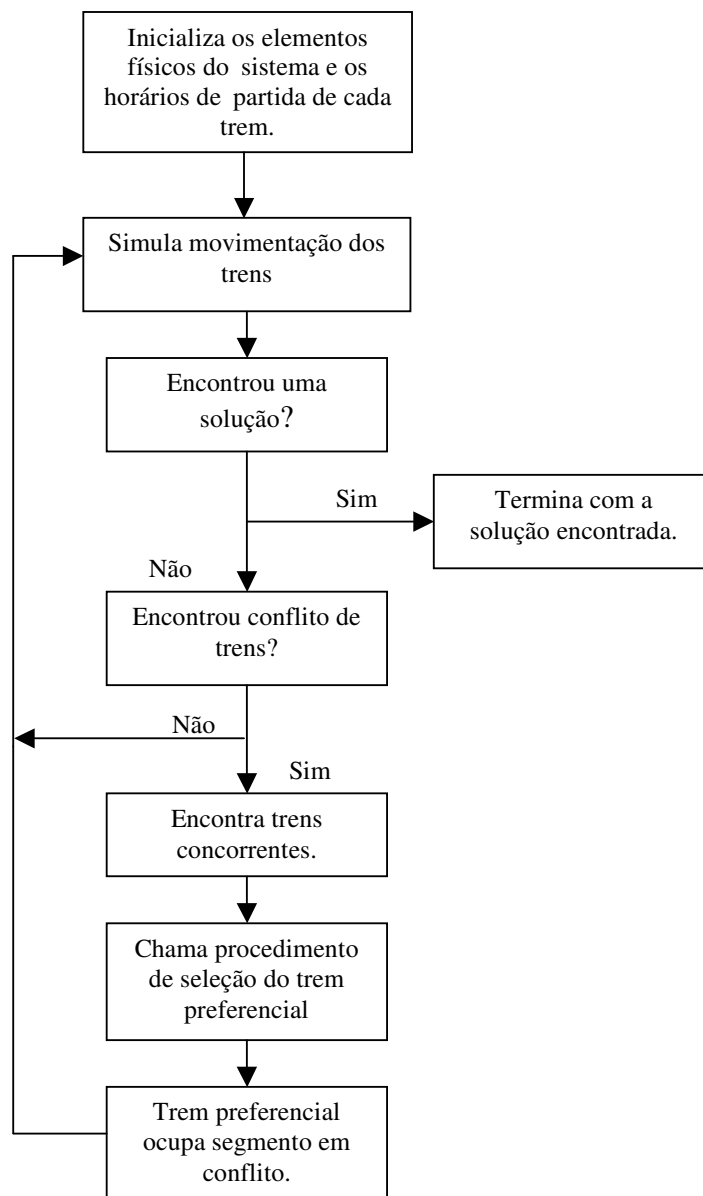


Figura 3.8: Processo de simulação e seleção do trem preferencial.

O processo é iniciado com a criação dos elementos físicos do sistema e com a inicialização dos horários de partida de cada trem. Os trens são movimentados em direção aos seus destinos até que um conflito de trens ocorra ou uma solução seja encontrada. No caso de uma solução ter sido encontrada o processo é finalizado com esta solução. Quando um conflito é encontrado o simulador utiliza o Algoritmo 3.1 para decidir dentre os trens concorrendo pelo uso de um segmento no mesmo instante de tempo qual é o trem preferencial. O trem escolhido como preferencial ocupa o segmento disputado e o processo de simulação continua com a movimentação dos trens.

É importante ressaltar que o algoritmo de decisão de preferência de trens utiliza para sua decisão a referência de percurso de cada trem. Estas referências podem ser fornecidas por algum especialista de circulação de trens ou por algum algoritmo de otimização global, como os de Mendes (2005) e Valle *et al.* (2005). A Figura 3.9 mostra a interação entre os diversos elementos do sistema.

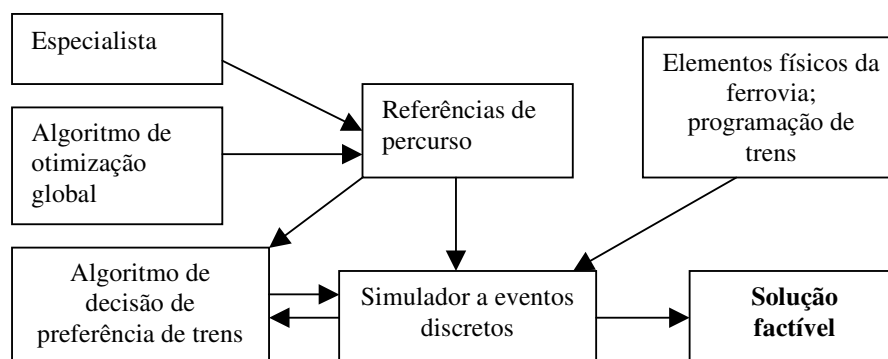


Figura 3.9: Estrutura do sistema

O simulador a eventos discretos recebe como entrada os dados sobre os elementos físicos do sistema e a programação de trens, que contém os horários de partida, o segmento de origem, o segmento de destino e o programa de atividades de cada trem. As referências de percurso de cada trem também são entradas do sistema. O objetivo é gerar um solução factível e para isso os conflitos entre trens são resolvidos pelo algoritmo de decisão de preferência.

### **3.4 Algoritmo de Decisão de Velocidade de Trens.**

#### **3.4.1 Princípios do algoritmo.**

Foi visto na seção anterior um algoritmo que determina a preferência entre trens concorrendo pelo uso de um mesmo segmento no mesmo instante de tempo através da classificação destes em conjuntos nebulosos determinados pelas suas referências de percurso. Essa seção descreve um algoritmo que juntamente com o algoritmo da seção anterior determina a preferência entre os trens e a velocidade que o trem preferencial deve percorrer o segmento concorrido para se aproximar de sua referência.

A escolha de um trem como preferencial não permite, no caso que este esteja atrasado, a recuperação deste atraso. Consegue apenas garantir que este trem não tenha mais atrasos. O atraso de um trem cresce monotonicamente em modelos nos quais os trens possuem velocidade de percurso constante, como é no caso do algoritmo de decisão de prioridades.

Considerando que a velocidade não seja mais constante e que seja uma variável de decisão do problema, podemos determinar as velocidades dos trens para que estes tenham a possibilidade de reduzir ou aumentar esta velocidade e, com isso, diminuir seu adiantamento ou seu atraso em relação à referência.

Neste algoritmo calcula-se, para cada trem escolhido como preferencial pelo algoritmo de decisão de preferência, qual a velocidade que ele deve percorrer o segmento concorrido para se aproximar de sua referência (Tazoniero *et al.*, 2007).

#### **3.4.2 Modelagem com teoria de conjuntos nebulosos.**

Quando um trem está atrasado ou adiantado em relação a sua referência de percurso, duas ações podem ser feitas para que o trem se aproxime de sua referência, que são aumentar ou reduzir sua velocidade de percurso.

Podemos considerar que cada trem possui uma velocidade mínima e máxima de percurso e que o aumento ou redução de velocidade podem ser modelados pela variável lingüística Velocidade (Yuan e Klir, 1995; Yen e Reza, 1998; Pedrycz e Gomide, 1998;

Passino e Yurkovich, 1998). Os seguintes conjuntos podem ser considerados como valores da variável lingüística Velocidade: *Muita Redução (MR)*, *Pouca Redução (PR)*, *Sem Aumento (SA)*, *Pouco Aumento (PA)* e *Muito Aumento (MA)*.

Estes conjuntos podem ser representados por funções de pertinência triangulares, conforme a Figura 3.10.

Todo trem possui um limite máximo para sua velocidade, seja por capacidade ou segurança, e também um limite mínimo desejado de velocidade. Estes valores limitam o aumento e redução de velocidade para cada trem. Devido a características de um trem, como peso, número e tipo de locomotivas, tipo de vagões, etc. e a características da linha ferroviária, como inclinação, bitola, etc, cada trem possui também valores de velocidade que otimizam o gasto de combustível. Essas velocidades também podem ser utilizadas como limitantes de velocidade.

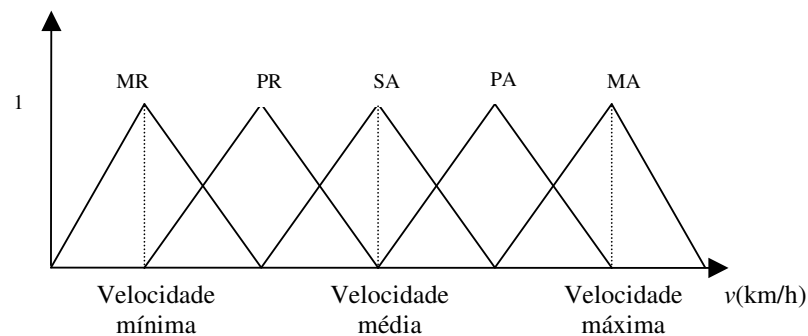


Figura 3.10: Representação dos conjuntos nebulosos para a velocidade do trem.

Utilizando a caracterização do percurso dos trens quanto a variável lingüística Percurso apresentada nas seções anteriores podemos relacionar o aumento ou redução de velocidade com a posição relativa do trem em relação a sua referência. Com isso uma base de regras pode ser construída, o que será visto na seção seguinte.

### 3.4.3 Base de regras para determinação de velocidade de trem.

Considerando, por exemplo, um trem que esteja atrasado em relação a sua referência. Para que este trem se reaproxime de sua referência é necessário que sua velocidade seja aumentada e que o tempo de percurso em um determinado segmento seja reduzido. Se esse aumento de velocidade for superior ao necessário a consequência seria que o trem iria se adiantar em relação a referência. O correto seria, então, aumentar a velocidade na quantia adequada.

Utilizando os mesmo conjuntos nebulosos que caracterizam o percurso do trem em relação a sua referência, *Muito Adiantamento*, *Pouco Adiantamento*, *Sem Atraso*, *Pouco Atraso*, *Muito Atraso* e também os conjuntos que caracterizam a velocidade do trem, *Muita Redução*, *Pouca Redução*, *Sem Aumento*, *Pouco Aumento* e *Muito Aumento*, pode-se estabelecer um conjunto de regras que possibilitem a trens atrasados e adiantados que se reaproximem de suas referências, respectivamente, através do aumento e redução de suas velocidades. As regras são mostradas a seguir:

Se o trem está *Muito Adiantamento* então o ganho de velocidade é *Muita Redução*.

Se o trem está *Pouco Adiantamento* então o ganho de velocidade é *Pouca Redução*.

Se o trem está *Sem Atraso* então o ganho de velocidade é *Sem Aumento*.

Se o trem está *Pouco Atraso* então o ganho de velocidade é *Pouco Aumento*.

Se o trem está *Muito Atraso* então o ganho de velocidade é *Muito Aumento*.

### 3.4.4 Defuzzificação.

Esta seção mostra como determinar as regras ativas a partir da caracterização do percurso dos trens quanto aos conjuntos nebulosos relativos a variável lingüística Percurso, mostra também como é feita a inferência da variável lingüística Velocidade, agregação e defuzzificação para determinar a velocidade do trem preferencial.

Supondo por exemplo que um trem  $i$  foi escolhido pelo algoritmo de decisão de preferência de trens como preferencial para ocupar um segmento  $s_a$  e que a caracterização do trem na variável linguística Percurso é:

$$\mu(PS)(Xa_{i,s_a}) = 0.2$$

$$\mu(MS)(Xa_{i,s_a}) = 0.8$$

As regras ativas são:

Se o trem está *Pouco Atraso* então o ganho de velocidade é *Pouco Aumento*.

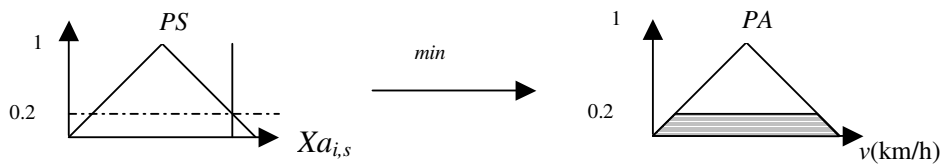
Se o trem está *Muito Atraso* então o ganho de velocidade é *Muito Aumento*.

Considerando que a conjunção da premissa é a  $t$ -norma  $\min$ , o grau de ativação de cada regra é dado por:

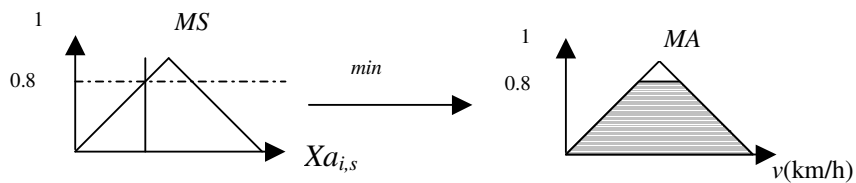
$$\lambda_1 = \min\{\mu(PS)(Xa_{i,s_a})\} = \min\{0.2\} = 0.2$$

$$\lambda_2 = \min\{\mu(MS)(Xa_{i,s_a})\} = \min\{0.8\} = 0.8$$

A Figura 3.11 mostra a decisão sugerida por cada regra:



Se  $Xa_{i,s_a}$  é *PS* então a *velocidade* é *PA*



Se  $Xa_{i,s_a}$  é *MS* então a *velocidade* é *MA*

Figura 3.11: Representação das regras e inferência nebulosa para a velocidade do trem.

A Figura 3.12 mostra agregação utilizando a *s-norma max* o dos conjuntos nebulosos referentes à decisão sugerida por cada regra. O valor final defuzzificado será obtido a partir dessa união.

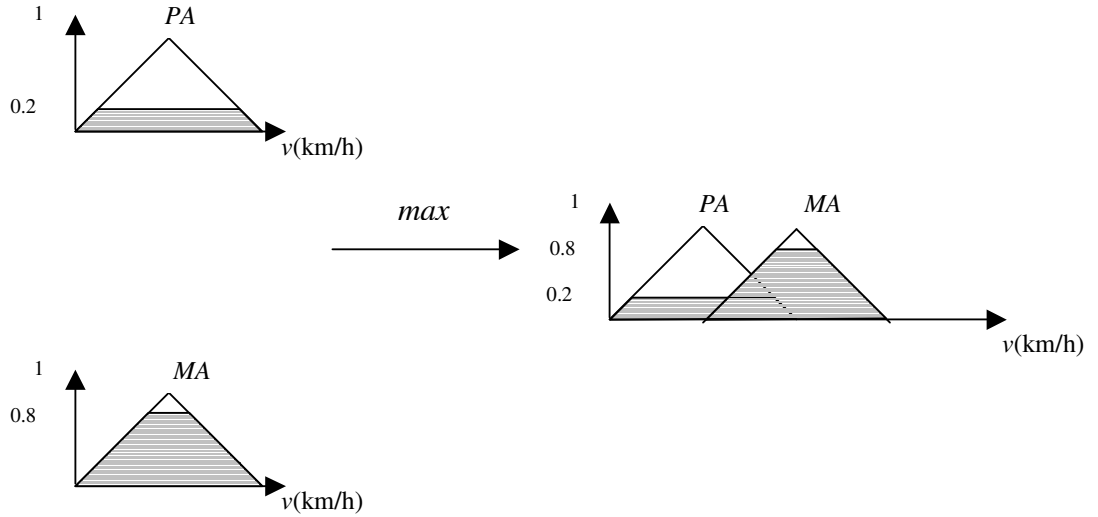


Figura 3.12: Representação da decisão para a velocidade do trem.

A defuzzificação será feita utilizando o método da Média das Áreas (Passino e Yurkovich, 1998), pelo qual o valor defuzzificado é dado por:

$$velocidade = \frac{\sum_{ra} b_{ra} \int_v \mu_{(.)}(v_{ra}) dv}{\sum_{ra} \int_x \mu_{(.)}(v_{ra}) dv} \quad (3.6)$$

Onde  $b_{ra}$  é o centro de cada conjunto nebuloso inferido pela regra  $ra$  e  $\int_v \mu_{(.)}(v_{ra}) dv$  é a área cada conjunto nebuloso inferido pela regra  $ra$ .

Em comparação com o método de defuzzificação do Centro de Gravidade o método da Média das Áreas é mais eficiente computacionalmente. O cálculo da área de cada conjunto nebuloso inferido é, geralmente, menos custoso que o cálculo do centro de gravidade da união dos conjuntos nebulosos inferidos. No caso deste trabalho, devido ao uso de funções de pertinência triangulares, o cálculo dessa área é o calculo dá área de um tronco de triângulo:



$$ÁreaTroncoTriângulo = (b - a) \left( h - \frac{h^2}{2} \right) \quad (3.7)$$

Onde  $a$  e  $b$  formam as bases do triângulo e  $h$  é a altura do tronco, ou seja, o mínimo dos graus de pertinência dos conjuntos da premissa. A Figura 3.13 mostra um exemplo de tronco de triângulo e os valores  $a, b$  e  $h$ .

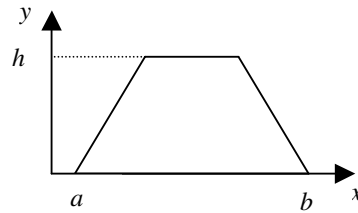


Figura 3.13: Tronco de triângulo.

Supondo que para o trem em questão a velocidade média seja de 30 km/h e que a velocidade máxima seja de 50 km/h e a mínima de 10 km/h, efetuando os cálculos temos que  $velocidade = 47.3$  km/h

Esse valor significa que o trem teria que percorrer o segmento  $s_a$  com velocidade de 47.3 km/h para recuperar sua condição de atraso em relação a sua referência. A Figura 3.14 mostra estes valores para o exemplo apresentado acima.

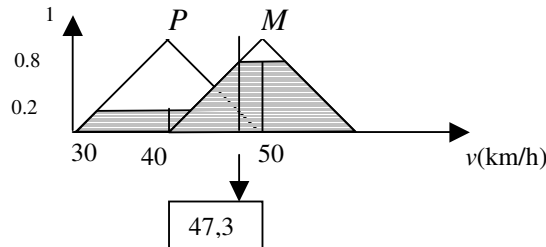


Figura 3.14: Defuzzificação da velocidade do trem.

### 3.4.5 Algoritmo de decisão de velocidade de trens.

As seções anteriores apresentaram os princípios do algoritmo de decisão de velocidade de trens. O Algoritmo 3.2 resume o algoritmo e a Figura 3.15 mostra o fluxograma do processo.

O algoritmo de decisão de velocidade de trem se comunica, durante o processo de simulação, com o simulador a eventos discreto, do qual recebe as informações sobre o trem

que foi escolhido como preferencial e sobre o segmento sendo disputado, conforme a Figura 3.16. O Algoritmo 3.2 calcula então a velocidade para o trem percorrer o segmento disputado e fornece essa velocidade para o simulador, que a utiliza para movimentar o trem na linha.

```

procedimento determinaVelocidadeTrem(trem,s) retorna velocidade

entradas:
trem //trem
S //segmento
variáveis locais:
vMaxima, vMinima, vMédia //velocidades do trem
baseDeRegras //base de regras para determinar velocidade do trem
vetorPertinencias[] //vetor com os valores das funções de pertinência para a variável
    lingüística Percurso
chegadaSegmento //instante de chegada do trem no segmento s
referencia //referencia de percurso do trem
conjuntosPercurso // lista com os conjuntos nebulosos da variável lingüística Percurso
conjuntosVelocidade //lista com os conjuntos nebulosos da variável lingüística Velocidade
regrasAtivas //lista com as regras ativas
conjuntosInferidos //lista com os conjuntos nebulosos para a velocidade do trem inferidos
    pelas regras ativas
uniaoDosConjuntosInferidos //união dos conjuntos inferidos

inicio

    vMaxima ← trem.velocidadeMaxima
    vMinima ← trem.velocidadeMinima
    vMédia ← trem.VelocidadeMédia
    referencia ← trem.referencia
    chegadaSegmento ← trem.instanteQueOTremChegouNoSegmento(s)
    conjuntosPercurso ← defineConjuntosNebulososPercurso (referencia,s)
    pertinencias[] ← caracterizaPercurso(conjuntosPercurso,chegadaSegmento)
    conjuntosVelocidade ← defineConjuntosVelocidade(vMaxima,vMinima,vMédia)
    regrasAtivas ← determinaRegrasAtivas(pertinencias[],conjuntosVelocidade)
    conjuntosInferidos ← fazInferenciaRegrasAtivas
        (regrasAtivas,pertinencias[],conjuntosVelocidade)
    uniaoDosConjuntosInferidos ← fazUniaoRegras(conjuntosInferidos)
    velocidade ← fazDefuzzificacao(uniaoDosConjuntosInferidos)

    retorna velocidade

fim

```

Algoritmo 3.2: Algoritmo de decisão de velocidade de trens.

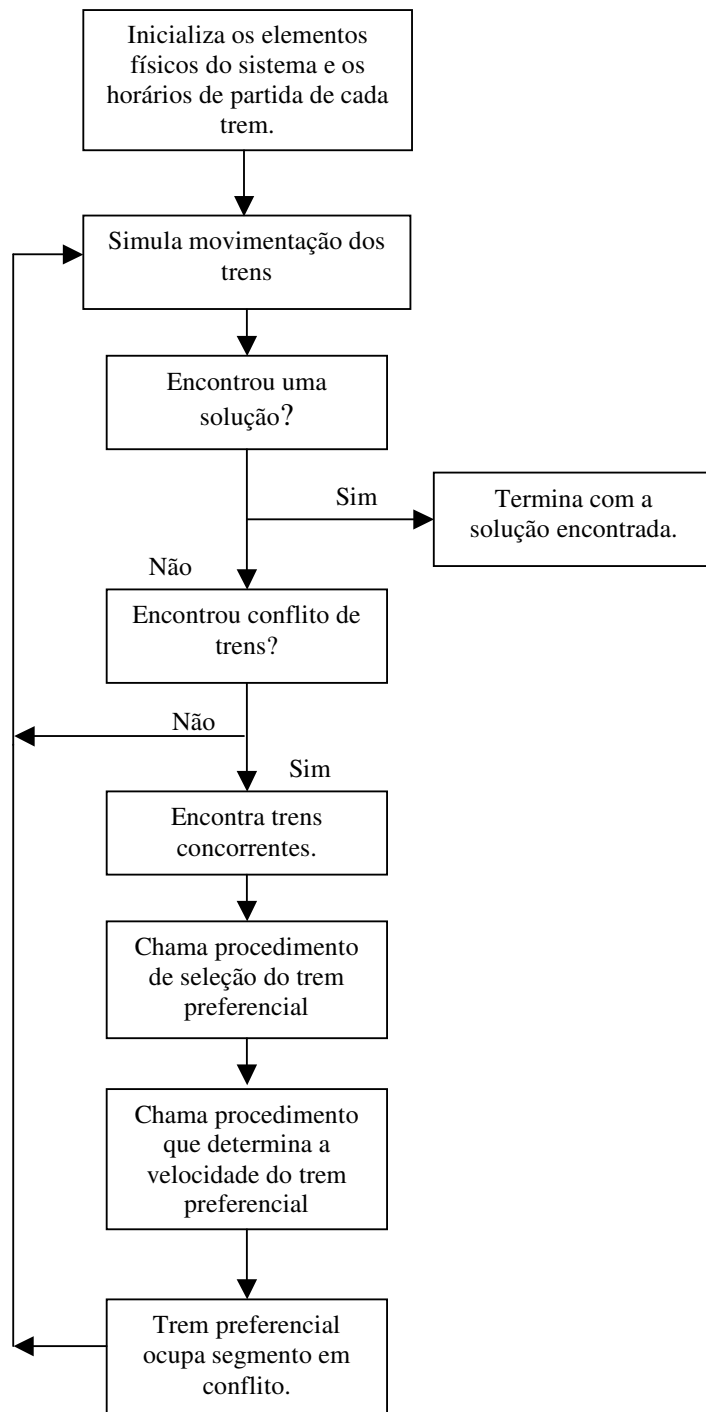


Figura 3.15: Processo de simulação, seleção do trem preferencial e decisão da velocidade do trem.

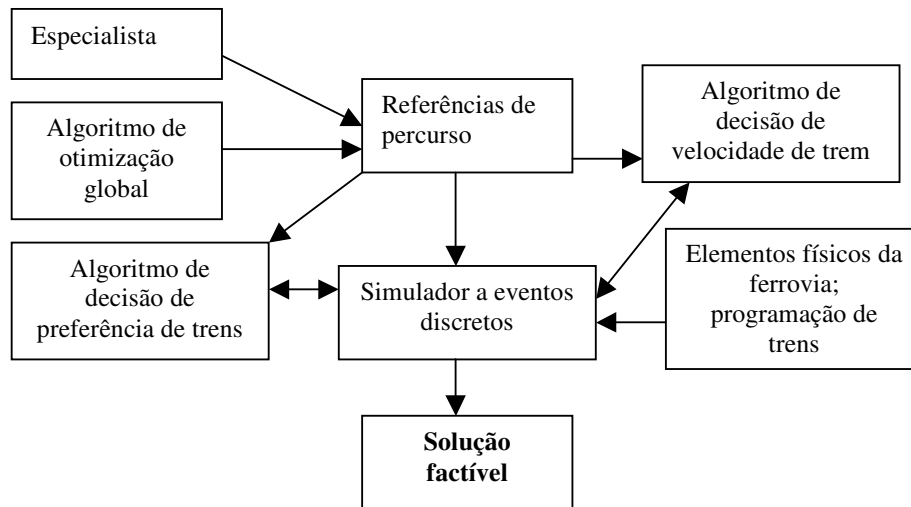


Figura 3.16: Estrutura do sistema

### 3.5 Exemplos.

Esta seção apresenta exemplos simples para ilustrar o funcionamento dos algoritmos de decisão de preferência de trens e decisão de velocidade de trem. Para o algoritmo de decisão de preferência de trens foram considerados dois critérios de desempate.

Critério 1: preferência para o trem cujo deslocamento manterá os trens concorrentes mais próximos de suas referências.

Critério 2: preferência para o trem que terminará de percorrer o segmento disputado no menor instante de tempo.

O modelo de linha considerado para este pequeno estudo possui seis segmentos que são pátios de cruzamento ou ultrapassagem e cinco segmentos com linha singela. Os detalhes sobre o comprimento de cada segmento e sobre as velocidades dos trens para cada segmento estão no Apêndice A seção A.1, Modelo Ferroviário 1.

A linha tem como limite oeste o segmento  $s_0$  e como limite leste o segmento  $s_{10}$ . Foram considerados cinco trens circulando, sendo três no sentido oeste-leste e dois no sentido leste-oeste. Os horários de partida de cada trem estão na Tabela 3.2. As referências de percurso para cada trem foram obtidas com o modelo de programação matemática de Valle *et al.*,(2005).

	Instante de partida
TREM1	01:00
TREM2	01:00
TREM3	03:00
TREM4	03:30
TREM5	07:30

Tabela 3.2: Instantes de partida.

A Figura 3.17 mostra a referência gerada pelo modelo de programação matemática. A solução gerada utilizando o algoritmo de preferência de trens com o critério 1 obteve a mesma solução que sua referência, o modelo de programação matemática. A Figura 3.18 mostra a solução gerada utilizando o algoritmo de preferência de trens com o critério 2 e a Figura 3.19 mostra a solução gerada pelo algoritmo de preferência de trens com o critério 2 e pelo algoritmo de decisão de velocidade .

Pode-se ver na Figura 3.17 que o cruzamento entre os trens TREM4 e TREM1 ocorreu no segmento  $s_6$  enquanto que na Figura 3.18 ocorreu em  $s_8$ . Essa diferente decisão feita pelo algoritmo de preferência de trens é decorrente do critério de desempate adotado. Como tanto o TREM1 e o TREM4 estavam seguindo exatamente suas referências no momento que disputaram o  $s_7$  e logo estavam em uma situação de empate, a decisão foi feita pelo critério de desempate. O critério 2 optou por dar preferência para o trem que cruzaria antes este segmento, no caso o TREM1. Essa decisão levou o TREM4 a se atrasar em relação à referência e ganhar preferência nos cruzamentos seguintes. Já o critério 1 deu preferência ao cruzamento que manteria os trens mais próximos de suas referências, no caso dando preferência para o TREM4, o que levou a manter os dois trens em suas referências. Essa diferença de decisão pelos critérios de desempate mostram a importância de um critério adequado. Como o critério 1 é o que consegue cumprir melhor o objetivo de manter os trens o mais próximos possível de suas referências, ele será utilizado no restante do trabalho

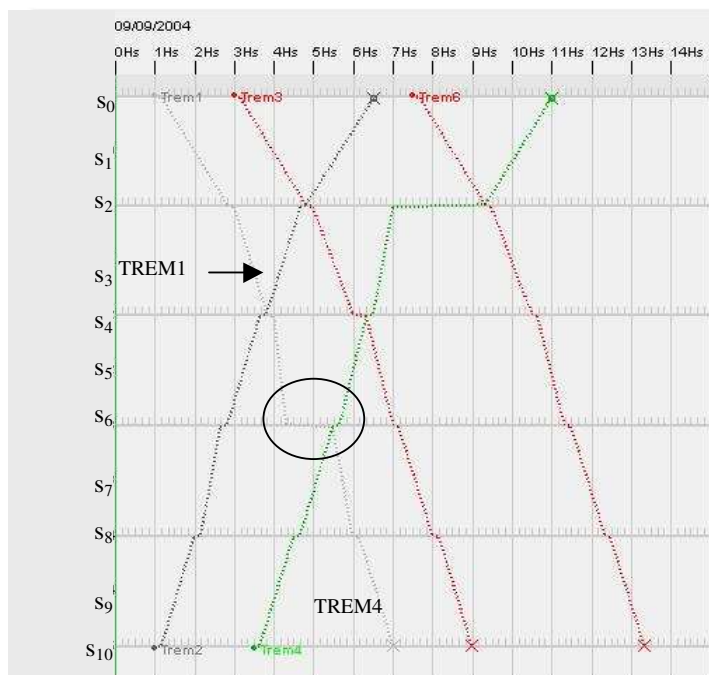


Figura 3.17: Referência gerada pelo Modelo de Programação Linear Inteira Mista e solução gerada pelo algoritmo de decisão de preferência de trens como critério 1.

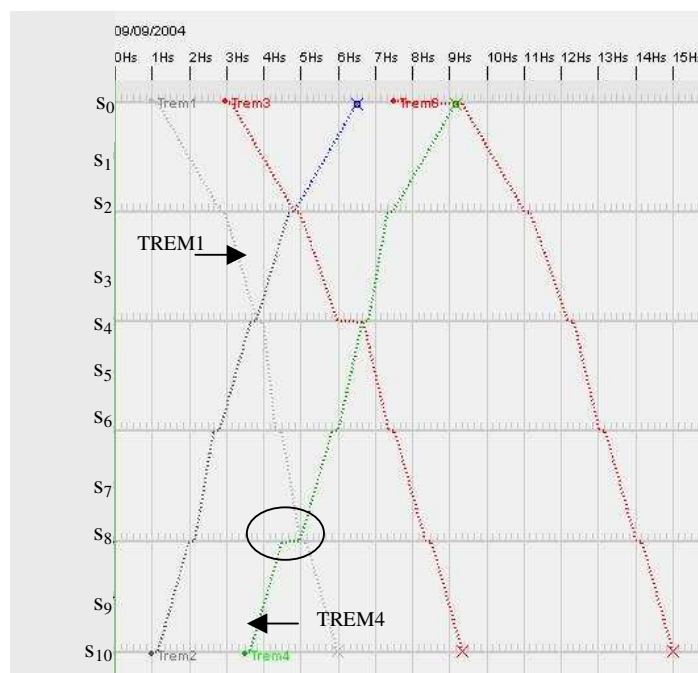


Figura 3.18: Solução gerada pelo algoritmo de decisão de preferência com o critério 2

A Figura 3.19 mostra a solução obtida pelo algoritmo de decisão de preferência com o critério 2 e pelo algoritmo de decisão de velocidade de trens.

Pode-se ver na Figura 3.19 que o cruzamento entre os trens TREM1 e TREM4 também ocorreu no segmento  $s_8$ , como na Figura 3.18. A diferença para esta solução da Figura 3.19 está no fato de que quando o TREM4 tem a preferência no seu próximo cruzamento ele percorre  $s_5$  com uma velocidade superior a da Figura 3.18, conforme está destacado na Figura 3.18. Isso faz com que ele se reaproxime de sua referência e mantenha o seu último cruzamento conforme a referência.

A possibilidade de alterar a velocidade do trem fez com que a solução da Figura 3.19 fosse mais próxima que a solução da Figura 3.17 do que a solução da Figura 3.18. Logo se tem a possibilidade de corrigir a posição dos trens em relação as suas referências durante a simulação, o que não é possível no algoritmo que apenas decide a preferência entre trens.

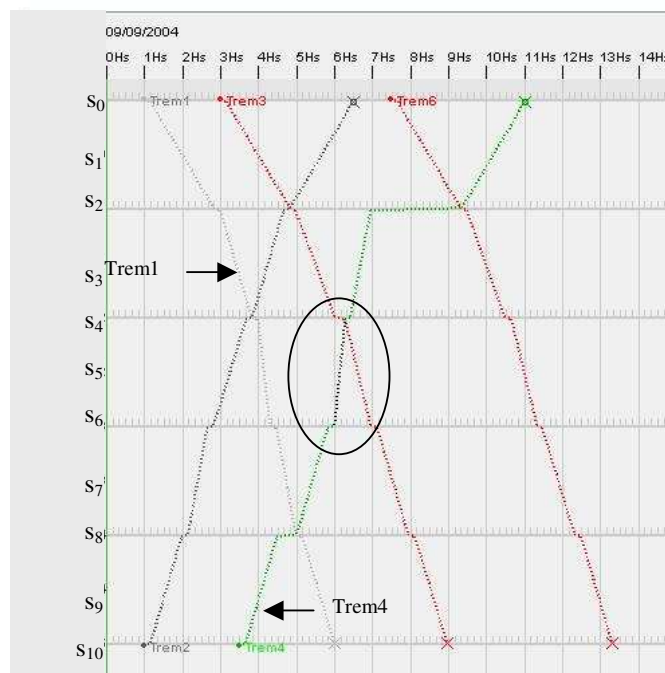


Figura 3.19: Solução obtida pelo algoritmo de decisão de preferência com o critério 2 e pelo algoritmo de velocidade de trens.

### **3.6 Resumo.**

Neste capítulo foram apresentados dois algoritmos para decisão de preferência entre trens. Foi definido que cada trem possui uma referência de percurso e o objetivo dos algoritmos é manter os trens o mais próximo possível de suas referências. O primeiro algoritmo decide apenas a prioridade dos trens. Já o segundo algoritmo decide a preferência e a velocidade dos trens.

As decisões de preferência do algoritmo de decisão de preferência são tomadas localmente, analisando apenas os trens em conflito. No próximo capítulo será apresentado um algoritmo que analisa para todos os trens que estão circulando o peso de cada decisão de cruzamento ou ultrapassagem.



## Capítulo 4

### Algoritmo de Busca em Árvore

Esse capítulo apresenta uma revisão bibliográfica dos principais algoritmos de busca em árvore. Em seguida é apresentada a utilização de busca em árvore em problemas de otimização combinatória. Por fim, um algoritmo de busca em árvore para o planejamento de circulação de trens é proposto.

#### 4.1 Introdução.

No capítulo anterior foram apresentados dois algoritmos de decisão de preferência entre trens para desenvolver uma solução de planejamento de circulação de trens. Baseado em uma referência de percurso, o primeiro algoritmo busca tomar as decisões de preferência de modo a manter os trens o mais próximo possível de suas referências. O segundo algoritmo determina da mesma maneira que o primeiro a preferência entre trens, mas também determina a velocidade com a qual o trem deve seguir seu trajeto de forma a manter-se na referência.

Ambos os algoritmos do capítulo anterior são utilizados durante a simulação dos movimentos dos trens na linha ferroviária utilizando o modelo de simulação do Capítulo 2. É importante notar que a decisão de preferência entre trens é feita localmente, pois apenas os trens que concorrem pelo uso do mesmo segmento no mesmo instante de tempo são analisados para a tomada de decisão.

Decisões locais de preferência entre trens tornam o processo de simulação simples, efetivo e computacionalmente muito rápido. Entretanto, na maioria dos casos soluções ótimas locais não são necessariamente globalmente ótimas. Portanto, as soluções obtidas pelos algoritmos nebulosos de decisão são sub-ótimas.

A utilização de algoritmos que possibilitem ao modelo analisar o impacto de cada decisão local para todos os trens circulando e para os futuros cruzamentos, se mostra como uma alternativa para dar maior visão ao processo de obtenção de solução.

Algoritmos de busca em árvore, busca local e meta heurísticas são abordagens apropriadas para analisar obter decisões de preferência.

Este capítulo apresenta um algoritmo de busca em árvore para a solução do problema de planejamento de circulação de trens em tempo-real.

## 4.2 Busca em Árvore no Planejamento de Circulação De Trens.

### 4.2.1 Algoritmos de busca em problemas combinatoriais.

Problemas de otimização combinatorial como planejamento de circulação de trens, *Job-Shop*, caixeiro viajante, roteamento de veículos, etc., consideram um conjunto de variáveis de decisão, sendo que para cada uma deve ser atribuído um valor dentro de um conjunto discreto de possibilidades (Pearl, 1984).

Representando um problema de decisão através de uma árvore, cada folha da árvore é solução do problema. Cada nó da árvore representa uma variável de decisão e cada arco que segue até o nó representa o valor atribuído a ela. A Figura 4.1 mostra um exemplo de árvore para um pequeno problema de decisão.

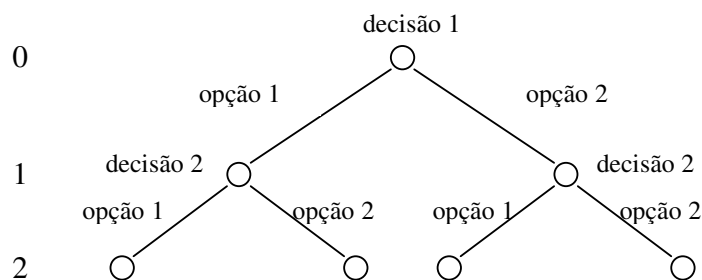


Figura 4.1: Representação em árvore das soluções de um pequeno problema combinatorial.

O nó inicial tem profundidade zero e os nós gerados a partir de dele, chamados de nós filhos, tem profundidade 1. Os nós filhos dos nós de profundidade 1 tem profundidade 2 e assim por diante.

Como o objetivo é obter a melhor solução, em geral é inviável computacionalmente a enumeração de todas as soluções para problemas grandes. Tenta-se então, obter a melhor

solução possível em um determinado espaço de tempo. Para isso, diversos tipos de algoritmos de busca em árvore foram desenvolvidos.

#### **4.2.2 Algoritmos tradicionais de busca em árvore.**

Como foi dito anteriormente, enumerar todas as soluções para a escolha da melhor solução é um procedimento inviável para problemas complexos. As metodologias existentes para a busca de uma solução ou da melhor solução tentam conciliar três fatores: tempo de processamento, quantidade de memória e qualidade da solução.

Os algoritmos de busca são divididos em dois tipos, os não-informados e os informados (Pearl, 1984; Russel e Norvig, 2003).

Estes algoritmos podem ser avaliados segundo quatro critérios.

1. completeza: o algoritmo garante encontrar uma solução se ela existir;
2. otimalidade: o algoritmo garante encontrar a solução ótima;
3. complexidade temporal: quanto tempo leva para achar uma solução;
4. complexidade espacial: quanta memória é necessária para o algoritmo.

A avaliação de complexidade temporal e complexidade espacial é feita com base em três valores:  $b$ , o fator de ramificação ou número máximo de sucessores de qualquer nó;  $d$ , a profundidade da solução menos profunda; e  $cm$ , o comprimento máximo de qualquer caminho na busca.

##### **4.2.2.1 Algoritmos não-informados.**

Os algoritmos não-informados são assim chamados por não utilizarem informações específicas sobre o problema e sobre a qualidade e natureza da solução durante o processo de busca. O principal objetivo desses métodos é encontrar uma solução factível.

Os dois principais algoritmos de busca não-informada são os algoritmos de Busca em Profundidade e Busca em Largura.

**Busca em profundidade:** Na busca em profundidade a busca por uma solução é feita dando preferência para os nós com maior profundidade. Partindo do nó inicial todos os nós filhos deste são gerados e um destes filhos é escolhido para ser explorado e tem todos

seus filhos gerados. Esse procedimento é repetido até que uma solução seja encontrada ou até que um caminho sem solução seja detectado. No caso de uma solução ter sido encontrada, o processo de busca termina com essa solução. Se um caminho sem solução for encontrado, o processo de busca retorna para o nó não explorado mais próximo e continua a busca.

A busca em profundidade é uma estratégia *last-in-first-out*, LIFO. Os nós expandidos são colocados no início de uma fila, que são os primeiros a serem expandidos. Isso garante que os nós com maior profundidade sejam expandidos primeiro. O algoritmo é resumido no Algoritmo 4.1 (Pearl, 1984; Russel e Norvig, 2003).

```
procedimento buscaEmProfundidade(estadoInicial) retorna solução ou falha

entradas:
estadoInicial //o estado inicial representa o estado inicial do problema

variáveis locais:
n //representa o nó atual do processo de busca
n' //representa um nó filho do nó n
abertos //lista que contém os nós que estão abertos, ou seja, não foram expandidos
fechados //lista que contém os nós que estão fechados, ou seja, já foram expandidos
listaNós //lista auxiliar de nós

inicio
    n ← inicializaNóDaBuscaComEstadoInicial(estadoInicial)
    insereNóFinalDaLista(abertos, n)
    repita
        Se Abertos =  $\emptyset$  então retorna falha
        n ← removePrimeiroNó(abertos)
        insereNó(fechados, n)
        listaNós ← expandeFilhosDoNó(n)
        insereNósNoInicioDaLista(abertos, listaNós)
        para cada n' de listaNós
            Se n' é solução então retorna solução
            Se n' é caminhoSemSolução então removeNó(abertos, n')
        limpaLista(fechados)

fim
```

Algoritmo 4.1: Algoritmo de busca em profundidade

A Figura 4.2 mostra a idéia de busca em profundidade. Os círculos representam nós, os círculos escuros são caminhos sem solução e o quadrado representa a primeira solução encontrada. Os números ao lado de cada nó representam a ordem na qual os nós foram expandidos.

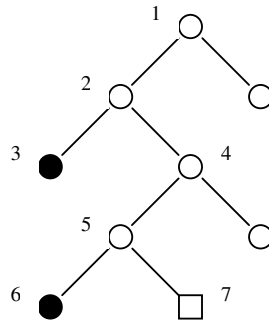


Figura 4.2: Exemplo de busca em profundidade.

**Busca em largura:** Na busca em largura o processo de busca é feito dando preferência a nós de menor profundidade. Partindo do nó inicial todos seus filhos de profundidade 1 são expandidos. A partir dos nós de profundidade 1 todos nós de profundidade 2 são também expandidos. Dessa maneira a busca prossegue até que se encontre uma solução.

Esse algoritmo tem uma estratégia *first-in-first-out*, FIFO. Os nós expandidos são colocados no final de uma lista e com isso os nós de menor profundidade são expandidos primeiro. O algoritmo é mostrado no Algoritmo 4.2 (Pearl, 1984; Russel e Norvig, 2003).

A Figura 4.3 mostra a idéia de busca em largura. Pode-se ver que a busca em largura teve que explorar um número maior de nós que a busca em profundidade da Figura 4.3.

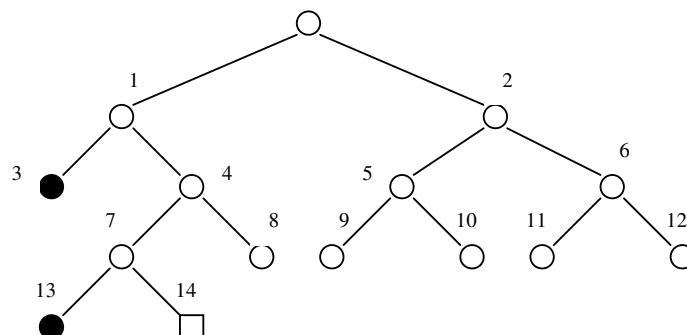


Figura 4.3: Exemplo de busca em largura.

**procedimento** buscaEmLargura(*estadoInicial*) **retorna** *solução* ou *falha*

**entradas:**

*estadoInicial* //o estado inicial representa o estado inicial do problema

**variáveis locais:**

*n* //representa o nó atual do processo de busca

*n'* //representa um nó filho do nó *n*

*abertos* //lista que contém os nós que estão abertos, ou seja, não foram expandidos

*fechados* //lista que contém os nós que estão fechados, ou seja, já foram expandidos

*listaNós* //lista auxiliar de nós

**inicio**

*n* ← inicializaNóDaBuscaComEstadoInicial(*estadoInicial*)

insereNóFinalDaLista(*abertos*, *n*)

**repita**

**Se** *Abertos* =  $\emptyset$  **então retorna** *falha*

*n* ← removePrimeiroNó(*abertos*)

insereNó(*fechados*, *n*)

*listaNós* ← expandeFilhosDoNó(*n*)

insereNósNoFinalDaLista(*abertos*, *listaNós*)

**para cada** *n'* **de** *listaNós*

**Se** *n'* é *solução* **então retorna** *solução*

**Se** *n'* é *caminhoSemSolução* **então** removeNó(*abertos*, *n'*) **e**  
limpaLista(*fechados*)

**fim**

Algoritmo 4.2: Algoritmo de busca em largura

O exemplo da Figura 4.2 e 4.3 é um caso no qual a busca em profundidade é vantajosa na busca pela solução, pois teve que explorar um menor número de nós e também teve que manter um menor número de nós em memória. Para casos em que a solução se encontra em uma profundidade baixa, a busca em largura se torna mais vantajosa.

A aplicação dos algoritmos de busca em profundidade e busca em largura é inviável para problemas onde a solução esteja em uma profundidade muito grande ou com largura

muito grande. Para estes problemas o número de combinação das variáveis de decisão é muito grande e estas estratégias são muito ineficazes.

#### 4.2.2.2 Algoritmos informados.

Para um problema com muitas soluções e cujo objetivo é encontrar a melhor solução, os métodos de busca não-informados não são a melhor opção. Além de demandarem muito tempo de processamento podem levar à soluções insatisfatórias.

Os métodos de busca informados são mais apropriados para estes problemas. Estes métodos se caracterizam por utilizar informações heurísticas na decisão de quais nós devem ser expandidos. Heurísticas são utilizadas para estimar o valor da função objetivo a ser otimizada de um nó até uma solução do problema. Elas podem ser obtidas a partir de regras práticas do problema, experiência de especialistas, etc. O uso de informações heurísticas consegue, geralmente, levar o processo de busca a obter soluções de boa qualidade, mas podem também levar o processo a soluções insatisfatórias.

O principal algoritmo de busca informada, que serve de base para outros importantes algoritmos, é o *Best-First* (Pearl, 1984; Russel e Norvig, 2003).

**Busca *Best-First*:** A principal característica deste algoritmo é a maneira de selecionar o próximo nó a ser expandido. Esta seleção é feita a partir de uma estimativa do valor da função objetivo do nó até se chegar à solução para cada nó candidato à expansão. O nó com melhor estimativa é o nó selecionado.

A estimativa pode ser feita com base em diferentes aspectos e depende da estrutura de cada problema e das informações disponíveis durante o processo de busca.

Pode-se representar a estimativa de cada nó  $n$  através de uma função heurística  $h(n)$ . O valor atribuído a um nó  $n$  é:

$$f(n) = h(n) \quad (4.1)$$

Onde  $h(n)$  é a estimativa do custo do nó até uma solução. O algoritmo do melhor em profundidade, *best-first*, é resumido no Algoritmo 4.3:

**procedimento** buscaBestFirst(*estadoInicial*) **retorna** *solução* ou *falha*

**entradas:**

*estadoInicial* //o estado inicial representa o estado inicial do problema

**variáveis locais:**

*n* //representa o nó atual do processo de busca

*n'* //representa um nó filho do nó *n*

*abertos* //lista que contém os nós que estão abertos, ou seja, não foram expandidos

*fechados* //lista que contém os nós que estão fechados, ou seja, já foram expandidos

*listaNós* //lista auxiliar de nós

**inicio**

*n* ← inicializaNóDaBuscaComEstadoInicial(*estadoInicial*)

insereNóFinalDaLista(*abertos*, *n*)

**repita**

**Se** *Abertos* =  $\emptyset$  **então retorna** *falha*

*n* ← removeNóComMenorValorDaFunçãoObjetivoF(*abertos*)

insereNó(*fechados*, *n*)

*listaNós* ← expandeFilhosDoNó(*n*)

**para cada** *n'* **de** *listaNós*

    calculaValorDaFunçãoObjetivoF(*n'*)

**Se** *n'* é *solução* **então retorna** *solução*

**Se** *n'*  $\notin$  *abertos* e *n'*  $\notin$  *fechados* **então**

        insereNó(*abertos*, *n*)

**Se** *n'*  $\in$  *abertos* ou *n'*  $\in$  *fechados* **então**

        atribuiAoNóOMenorValorDaFuncaoObjetivoF(*n'*)

**Se** *n'*  $\in$  *fechados* **então**

        insereNó(*abertos*, *n'*)

        removeNó(*fechados*, *n'*)

**fim**

Algoritmo 4.3: Algoritmo de busca *BestFirst*



O algoritmo *best-first* tem complexidade computacional  $O(b^{cm})$ , mas uma função heurística adequada pode contribuir consideravelmente para seu desempenho (Pearl, 1984; Russel e Norvig, 2003).

Se algumas propriedades forem verificadas pela função heurística  $h(n)$  o algoritmo *best-first* garante a otimalidade da solução encontrada. Este algoritmo é conhecido como  $A^*$ .

**Busca  $A^*$ :** Este é o algoritmo do tipo *best-first* mais conhecido. A grande diferença entre este algoritmo e o *best-first* tradicional está na função de avaliação do nó  $n$ ,  $f(n)$ . A avaliação é feita agregando o custo para alcançar cada nó,  $g(n)$ , com o custo estimado do nó até uma solução do problema,  $h(n)$ .

$$f(n) = g(n) + h(n) \quad (4.2)$$

Para garantir otimalidade a função  $h(n)$  deve possuir duas propriedades, ser admissível (sempre subestimar o custo de um nó até a solução) e ser monotônica crescente.

Com isso temos que:

$$h(n) \leq C^* \quad (4.3)$$

Onde  $C^*$  é o custo, isto é, o valor da função  $f(.)$  para a solução ótima.

Este algoritmo é também eficiente pois expande o menor número de nós para obter a solução ótima, se ela existir.

Um problema do algoritmo  $A^*$  está na quantidade de memória necessária e na dificuldade para se encontrar uma função  $h(n)$  admissível (Pearl, 1984; Russel e Norvig, 2003). A Tabela 4.1 resume as principais características dos algoritmos de busca apresentados (Pearl, 1984; Russel e Norvig, 2003).

	Busca em Profundidade	Busca em Largura	<i>Best-First</i>	$A^*$
Completo	N	N	S	S
Otimalidade	N	N	N	S
Complexidade temporal	$O(b^{cm})$	$O(b^{d+1})$	$O(b^{cm})$	$O(b^{cm})$
Complexidade espacial	$O(bm)$	$O(b^{d+1})$	$O(b^{cm})$	$O(b^{cm})$

Tabela 4.1: Análise de complexidade dos algoritmos de busca .

As próximas seções desenvolvem um algoritmo *best-first* modificado para o problema de planejamento de circulação de trens.

#### 4.2.3 Representação em árvore do problema de planejamento de circulação de trens.

Sendo o problema de planejamento de circulação de trens um problema de otimização combinatorial podemos representar suas soluções através de uma árvore.

A variável de decisão no problema de trens é a decisão da preferência entre trens quando há a concorrência pelo uso de um mesmo segmento no mesmo instante. Cada nó representa a concorrência entre dois ou mais trens por um segmento e cada caminho que segue de um nó a preferência atribuída a cada um dos trens concorrentes. Cada decisão de preferência leva a uma diferente solução diferente.

A Figura 4.4 mostra a expansão de um nó para um pequeno problema com apenas dois trens circulando,  $t_1$  e  $t_2$ , e sete segmentos:  $s_1, s_2, s_3, s_4, s_5, s_6$  e  $s_7$ . Os segmentos  $s_1, s_3, s_5$  e  $s_7$  são pátios de cruzamento e os segmentos  $s_2, s_4$  e  $s_6$  são linhas singelas. O trem  $t_1$  inicia seu percurso no segmento  $s_1$  e tem como destino o segmento  $s_7$ , já o trem  $t_2$  tem como origem o segmento  $s_7$  e como destino o segmento  $s_1$ .

No nó raiz, indicado pelo número zero, o trem  $t_1$  alocado no segmento  $s_3$  compete com o trem  $t_2$  alocado no segmento  $s_5$  pela ocupação do segmento singelo  $s_4$  entre  $s_3$  e  $s_5$ . Os nós filhos 1 e 2 representam as duas hipóteses para o conflito no nó raiz. No nó 1 o trem  $t_2$  tem a preferência para ocupar o segmento  $s_4$  enquanto o trem  $t_1$  permanece em  $s_3$  aguardando o cruzamento com  $t_2$ . Na outra hipótese o trem  $t_1$  tem a preferência para ocupar o segmento  $s_4$  enquanto o trem  $t_2$  é atrasado em  $s_5$ .

Neste exemplo os nós 1 e 2 são também soluções ou folhas do problema. Para problemas maiores com maior número de trens e segmentos o procedimento para a geração dos nós filhos e folhas é mesmo. Para cada conflito geram-se nós referentes a cada possibilidade de resolução do conflito. Esse procedimento se repete até que se encontre uma solução para o problema.

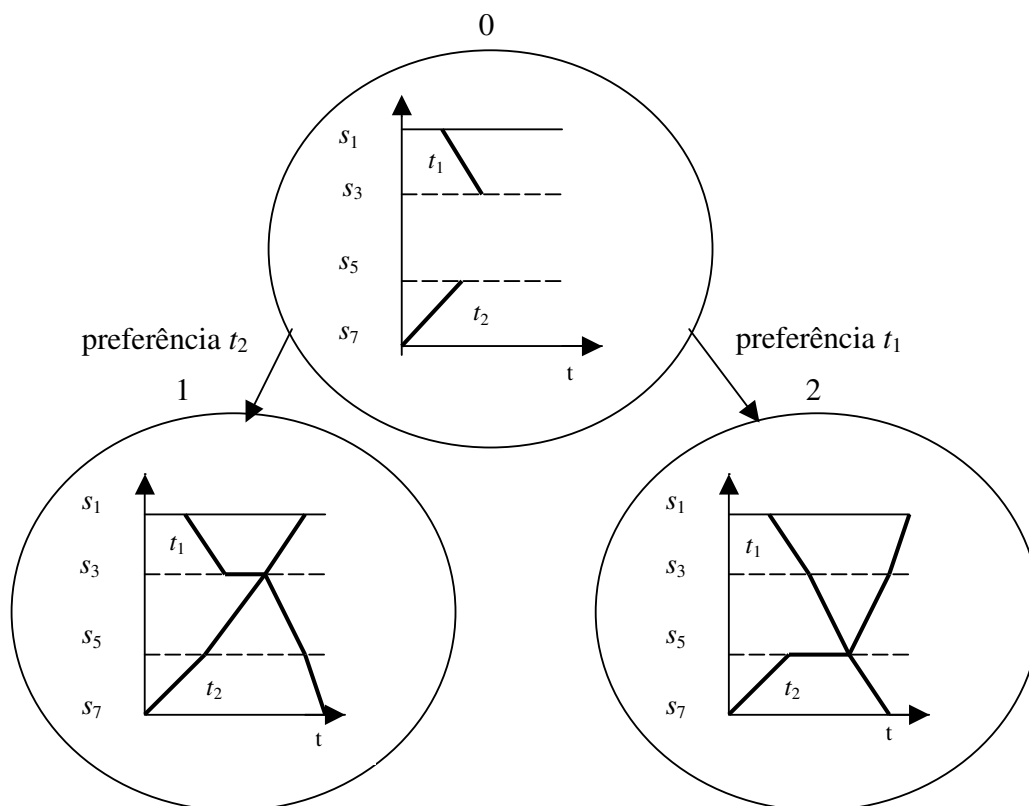


Figura 4.4: Representação em árvore das soluções de problemas de circulação de trens.

#### 4.2.4 Referências de busca em árvore em circulação de trens.

A primeira aplicação de algoritmos de busca em árvore ao problema de circulação de trens foi proposta por Cherniavsky, 1972. Cherniavsky propõe uma metodologia para obtenção de soluções factíveis utilizando uma base de regras para satisfação das restrições de cruzamento e bloqueio, e um algoritmo de busca em árvore para obtenção de melhores soluções. A modelagem do problema considera uma linha ferroviária com segmentos singelos, separados por pátios de cruzamento, e trens de carga e de passageiros circulando nos dois sentidos da ferrovia.

O algoritmo é inicializado determinando-se os tempos de chegada e partida de cada trem em cada segmento. As restrições de circulação são consideradas, nessa primeira fase,

apenas para os trens de passageiros, que são prioritários e não devem ter seus percursos alterados. Como as restrições não são verificadas para os trens de carga, o planejamento obtido inicialmente contém uma série de violações de restrições, isto é, conflitos.

O planejamento inicial com conflitos é o nó raiz para o algoritmo de busca. Seguindo uma ordem de segmentos e de tempo, os conflitos são identificados. Para cada conflito identificado existe um conjunto de operadores elementares que podem ser utilizados para a resolução do conflito. A aplicação de um operador a um conflito gera um novo nó com novos conflitos, que é considerado o nó filho do nó raiz. Quando uma situação de bloqueio é atingida com a aplicação dos operadores elementares, um outro operador, chamado de operador de bloqueio, é aplicado para que a situação de bloqueio seja evitada. Partindo do nó raiz, são gerados nós até a profundidade máxima definida. Cada nó com profundidade máxima é avaliado e tem o custo da função objetivo estimado. A estimativa é feita com o cálculo de caminhos de custo mínimo. Para o nó pai é calculada a hora de chegada de cada trem em cada segmento através de projeções de percurso de custo mínimo. O mesmo cálculo é feito para cada nó filho com profundidade máxima e é calculada a soma das diferenças entre as novas horas de chegadas nos segmentos com as horas calculadas para o nó pai.

O próximo nó a ser explorado é o nó com profundidade 1 que gerou o nó com profundidade máxima que teve menor custo da função objetivo estimado. O procedimento de identificação de conflitos, geração de nós e estimativas de custo se repete até que uma solução factível para o problema seja encontrada.

#### **4.3 Algoritmo *Best-First* para Circulação de Trens.**

O modelo de ferrovia apresentado no Capítulo 2 simula a movimentação dos trens em uma linha considerando as restrições de bloqueio, cruzamento, ultrapassagem e ocupação. Nas situações em que dois ou mais trens concorrem pelo uso de um mesmo segmento no mesmo instante de tempo um procedimento de decisão de preferência é chamado para escolher qual trem ocupará o segmento.

Podemos considerar que cada vez que existe uma concorrência por um segmento temos um novo nó na árvore de busca. Assim cada decisão tomada em um nó leva a um

diferente ramo da árvore e a novos nós. Com isso a simulação pode ser vista como uma busca que percorre a árvore do problema e encontra a uma solução segundo sua estratégia de despacho. A seção seguinte apresenta os princípios do algoritmo.

#### 4.3.1 Princípios do algoritmo.

No exemplo da Figura 4.4 temos duas possíveis soluções para o conflito no nó raiz. Neste caso a escolha da melhor solução é simples: basta calcular o valor da função objetivo para cada nó e selecionar o nó com menor valor. A solução escolhida seria a solução ótima do problema. Para instâncias maiores, com maior número de trens e segmentos, a escolha do próximo nó não pode ser feita através da expansão de todos nós até que se encontre o melhor nó, pois o número de conflitos é, em geral, muito alto. Uma estratégia alternativa seria simular a movimentação dos trens com o simulador a eventos discreto em um horizonte de tempo a partir de um nó e analisar o impacto do atual conflito nos cruzamentos e ultrapassagens que ocorrerão nesse horizonte. Essa simulação permite que a busca em árvore selecione o próximo nó considerando o que já aconteceu até o conflito e o que acontecerá para cada alternativa de decisão. Naturalmente, durante a simulação novos conflitos vão ocorrer e devem também ser resolvidos. Uma maneira rápida para resolver conflitos é utilizar uma estratégia gulosa, como selecionar dentre os trens concorrentes o trem que cruzará o segmento disputado primeiro (Tazoniero *et al.*, 2007).

Considerando a função objetivo deste trabalho, temos que o custo do estado inicial até um determinado nó  $n$  é dado pelo total de tempo de paradas para cruzamento e ultrapassagens de todos os trens.

Utilizando a notação de busca em árvore (Pearl, 1984; Russel e Norvig, 2003):

$$g(n) = \sum_{i=0}^N \text{TempoParadaTrem}_{n,i} \quad (4.4)$$

Onde  $\text{TempoParadaTrem}_{n,i}$  é o tempo de parada para cruzamento e ultrapassagens do trem  $i$  do nó raiz até o nó  $n$ .

O valor da estimativa do custo do nó até uma solução,  $h(n)$  é dado por:

$$h(n) = \sum_{i=0}^N \text{TempoParadaEstimadoTrem}_{n,i} \quad (4.5)$$

Onde o  $\text{TempoParadaEstimadoTrem}_{n,i}$  é o tempo de parada para cruzamento e ultrapassagens para cada trem  $i$  obtido com a simulação da movimentação dos trens dentro de um horizonte de tempo a partir do nó  $n$ .

A avaliação de um nó passa a ser dada por:

$$f(n) = g(n) + h(n) \quad (4.6)$$

ou

$$f(n) = \sum_{i=0}^N \text{TempoParadaTrem}_{n,i} + \sum_{i=0}^N \text{TempoParadaEstimadoTrem}_{n,i}$$

Considerando uma estratégia de busca em árvore do tipo *Best-First*, o nó selecionado é o nó com menor valor de  $f(.)$ .

#### 4.3.2 Algoritmo de busca para planejamento de circulação de trens.

Os passos do algoritmo de planejamento de circulação seguem os do algoritmo *best-first* tradicional resumido na seção 4.2.2. A principal diferença está no fato de que no algoritmo desenvolvido neste trabalho os nós não selecionados para serem explorados não são armazenados. Essa característica do algoritmo permite a ele uma considerável redução no consumo de memória. O Algoritmo 4.4 resume algoritmo de planejamento de circulação.

A Figura 4.5 mostra o fluxograma do algoritmo de busca em árvore contextualizado no processo de simulação.

O processo de busca é inicializado com os elementos físicos da ferrovia e os horários de partida da origem de cada trem. A partir disso é feita a simulação da movimentação dos trens até o momento que se encontre um nó (conflito), onde dois ou mais trens competem pela ocupação de um mesmo segmento em um mesmo instante de tempo. Para o nó encontrado, seus filhos são gerados e para cada filho é calculado o valor de  $f(.)$ . A simulação da movimentação dos trens é reiniciada a partir do nó que teve menor

valor de  $f(.)$  e é feita até que se encontre o próximo nó. O processo é repetido até que uma solução do problema seja encontrada.

**procedimento** buscaEmÁrvore(*estadoInicial*) **retorna** *solução* ou *falha*

**entradas:**

*estadoInicial* //o estado inicial representa o estado inicial do problema

**variáveis locais:**

*n* //representa o nó atual do processo de busca

*n'* //representa um nó filho do nó *n*

*abertos* //lista que contém os nós que estão abertos, ou seja, não foram expandidos

*fechados* //lista que contém os nós que estão fechados, ou seja, já foram expandidos

*listaNós* //lista auxiliar de nós

**inicio**

$n \leftarrow \text{inicializaNóDaBuscaComEstadoInicial}(\text{estadoInicial})$

$\text{insereNóFinalDaLista}(\text{abertos}, n)$

**repita**

**Se**  $\text{Abertos} = \emptyset$  **então retorna** *falha*

$n \leftarrow \text{removeNóComMenorValorDaFunçãoObjetivoF}(\text{abertos})$

$\text{limpaLista}(\text{abertos})$

$\text{listaNós} \leftarrow \text{expandeFilhosDoNó}(n)$

$\text{calculaValorDaFunçãoObjetivoFParaTodosNós}(\text{listaNós})$

$\text{ordenaListaPelaOrdemCrescenteDaFunçãoObjetivo}(\text{listaNós})$

**para cada** *n'* **de** *listaNós*

**Se** *n'* **é** *solução* **então retorna** *solução*

**Se**  $n' \notin \text{abertos}$  **então**

$\text{insereNó}(\text{abertos}, n)$

**Se**  $n' \in \text{abertos}$  **então**

$\text{atribuiAoNóOMenorValorDaFuncaoObjetivoF}(n')$

**fim**

Algoritmo 4.4: Algoritmo de busca para planejamento de circulação de trens.

É importante ressaltar que o cálculo de  $f(.)$  envolve um novo processo de simulação, feito a partir de cada nó. A diferença na simulação feita no calculo de  $f(.)$  é que para está não é utilizada uma estratégia global para resolução dos conflitos, mas sim uma estratégia

local. A estratégia local utilizada para a busca em árvore seleciona como trem preferencial o trem que atravessará o segmento sendo disputado no menor instante de tempo.

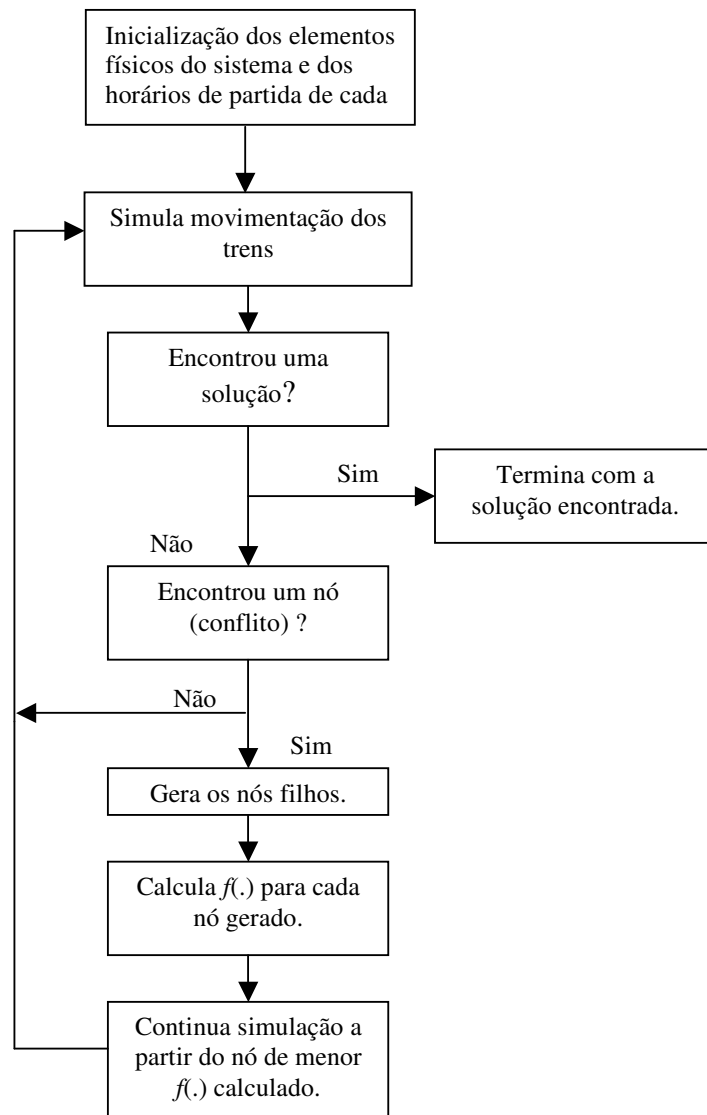


Figura 4.5: Fluxograma do algoritmo de busca em árvore.

A Figura 4.6 mostra a estrutura de funcionamento do sistema para obtenção de soluções para o planejamento de circulação de trens factíveis e mostra também aonde o algoritmo de busca em árvore está dentro do sistema.



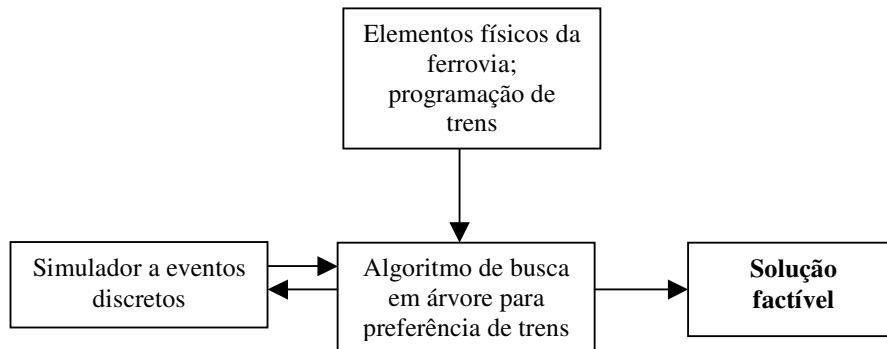


Figura 4.6: Estrutura do sistema com algoritmo de busca em árvore.

#### 4.3.3 Otimalidade do algoritmo.

O algoritmo de busca em árvore utiliza o mesmo princípio do  $A^*$  para o cálculo da função  $f(n)$  do custo de um nó até uma solução. A otimalidade do  $A^*$  é garantida se  $h(n)$  for monotônica crescente e for uma função otimista, ou seja:

$$h(n) \leq C^* \quad (4.7)$$

Onde  $C^*$  é o custo, isto é, o valor da função  $f(.)$  para a solução ótima.

A função  $h(n)$  proposta neste algoritmo é calculada a partir da simulação de um nó  $n$  em um horizonte de tempo. O valor de  $h(n)$  é, então, calculado sobre o custo de uma solução factível. Com isso temos que necessariamente:

$$h(n) \geq C^* \quad (4.8)$$

A equação (4.8) se explica pelo fato de não ser possível encontrar uma solução factível com custo menor que o ótimo. O algoritmo então não é ótimo, pois a função de estimativa não é otimista, o que não garante que a solução ótima seja encontrada. Portanto, em geral a solução obtida é sub-ótima.

Considerando as restrições de tempo impostas por problemas de planejamento em tempo-real, o objetivo do algoritmo é obter soluções de boa qualidade com um pequeno tempo de processamento.

#### **4.4 Resumo.**

Este capítulo apresentou os conceitos de busca em árvore para problemas de decisão e os principais algoritmos. A seguir desenvolveu-se um algoritmo de busca em árvore para a solução do problema de circulação de trens.

No próximo capítulo é feito um estudo comparativo das diferentes propostas consideradas neste trabalho e outras propostas existentes na literatura.

## Capítulo 5

### Resultados e Estudo Comparativo

Este capítulo apresenta e discute os resultados obtidos pelos algoritmos para o planejamento em circulação de trens propostos nos capítulos 3 e 4 e compara com os resultados obtidos com algoritmos existentes na literatura. Os resultados são obtidos a partir de instância simples, geradas com a variação dos principais parâmetros do sistema, e uma instância real, obtida com dados reais de uma ferrovia brasileira. Por fim é feito um resumo dos resultados.

#### 5.1 Introdução.

Cada algoritmo possui características e objetivos que influenciam diretamente o seu desempenho. Para realizar a análise e comparações entre resultados obtidos pelo uso de cada modelo, é necessário considerar uma metodologia de testes consistente. Em Rardin e Uzsoy, 2001, é feita uma discussão sobre as diferentes maneiras de gerar e analisar resultados de algoritmos heurísticos para problemas de otimização. O artigo apresenta uma proposta baseada no teste de quatro diferentes algoritmos para o problema de *Job-Shop*. Esta proposta considera diferentes instâncias variando os principais parâmetros do problema. Estas instâncias são solucionadas por todos os algoritmos e os resultados são comparados com as melhores soluções conhecidas para as instâncias. O artigo também é sugere o uso de dados reais, se disponíveis, para geração de resultados. Esta abordagem é utilizada na geração, análise e comparação dos resultados dos algoritmos propostos neste trabalho.

Devido às limitações de tempo impostas pela obtenção da solução ótima pelo algoritmo de programação matemática serão considerados dois tipos de instâncias: instâncias pequenas, para as quais se tem a solução ótima, e instâncias grandes com dados reais de ferrovias brasileiras. Os algoritmos foram implementados com a linguagem de programação Java e os resultados foram obtidos em um computador Pentium 4 2.8GHz com 1024Mb de memória RAM.

## 5.2 Resultados para Instâncias Pequenas.

O objetivo do estudo de instâncias pequenas é analisar o comportamento dos algoritmos propostos para casos simples e comparar os resultados dos diversos algoritmos quanto a qualidade da solução supondo que o modelo de programação linear inteira-mista fornece a solução ótima. Outro ponto importante deste estudo é analisar o efeito da variação de cada parâmetros sobre os resultados. Os parâmetros considerados são os seguintes:

- número de trens circulando;
- intervalo de tempo entre os horários de partida dos trens;
- número de segmentos da linha ferroviária.

A variação destes parâmetros permite a geração de instâncias com maior ou menor tráfego de trens. Quanto maior o número de trens, menor o intervalo entre os instantes de partida e quanto menor o número de segmentos da linha, maior a densidade de trens do problema.

Os resultados serão analisados sobre dois aspectos: tempo total de paradas devido a cruzamentos e/ou ultrapassagens e tempo de processamento dos algoritmos. Estes valores permitem analisar a qualidade da solução obtida para a função objetivo adotada neste trabalho e analisar o tempo de processamento necessário para que o algoritmo obtenha a solução.

São nove os algoritmos utilizados:

1. Modelo de Programação Linear Inteira Mista, PLIM, proposto por Valle *et al*, 2005;
2. Algoritmo Genético, GA, proposto em Mendes, 2005;
3. Algoritmo de Decisão de Preferência utilizando o resultado do PLIM como referência,  $AP_{PLMI}$ ;
4. Algoritmo de Decisão de Preferência utilizando o resultado do GA como referência,  $AP_{GA}$ ;

5. Algoritmo de Decisão de Preferência utilizando o Programa de Atividades de Trem<sup>2</sup> como referência,  $AP_{PAT}$ ;
6. Algoritmo de Decisão de Preferência e Velocidade utilizando o Programa de Atividades de Trem como referência,  $APV_{PAT}$ ;
7. Algoritmo de Decisão de Preferência e Velocidade utilizando o resultado do PLIM como referência.  $APV_{PLMI}$ ;
8. Algoritmo de Busca Em Árvore proposto neste trabalho, BA;
9. Algoritmo de Busca Em Árvore proposto por Cherniavsky, 1972, BAC.

Os parâmetros utilizados em cada algoritmos são os seguintes:

**Algoritmo genético, GA:**

- tamanho da população: 15 indivíduos;
- número de gerações: 150 gerações;
- probabilidade de mutação: 10%;
- probabilidade de *crossover* : 90%.

**Algoritmo de decisão de preferência e velocidade:**

- limites de velocidade:
  - velocidade máxima =  $1.5 * (\text{velocidade média})$ ;
  - velocidade mínima =  $0.5 * (\text{velocidade média})$ ;

**Algoritmo de Busca Em Árvore:**

- profundidade de busca: 1;
- horizonte de tempo para simular o custo de um nó: 24 horas;

**Algoritmo de Busca Em Árvore proposto por Cherniavsky, 1972:**

- profundidade de busca: 3.

Para o estudo do efeito do número de segmentos na linha ferroviária para os resultados dos algoritmos acima serão considerados dois modelos de linha ferroviária:

---

<sup>2</sup> O Programa de Atividades de Trem (PAT) contém os instantes de entrada e partida de cada segmento de um trem sem considerar as paradas para cruzamento e/ou ultrapassagens, considerando apenas as paradas para atividades obrigatórias. O conjunto dos PAT de todos os trens em uma linha ferroviária forma um planejamento de circulação de trens infactível, pois não considera as restrições de cruzamentos, ultrapassagens, ocupação e bloqueio.

- Modelo Ferroviário 1: 6 pátios de cruzamento e ultrapassagem e 5 segmentos de linha singela;
- Modelo Ferroviário 2: 9 pátios de cruzamento e ultrapassagem e 8 segmentos de linha singela.

Detalhes sobre os modelos ferroviários estão no Apêndice I.

### 5.2.1 Resultados com modelo ferroviário 1.

O Modelo Ferroviário 1 considera uma linha com 6 pátios de cruzamento e ultrapassagem e 5 segmentos de linha singela e instâncias com 3, 4, 5, 6 e 7 trens. O número de trens é um fator limitante para a obtenção da solução ótima pelo PLIM. O tempo de processamento cresce exponencialmente com o número de trens. A obtenção de uma solução com 8 trens circulando com o modelo ferroviário 1 levou mais de 40 horas de processamento.

Foram geradas 15 instâncias para o modelo ferroviário 1, classificadas pelo intervalo de tempo entre horário de partida dos trens, sendo que três intervalos foram considerados: 4 horas, 3 horas e 2 horas. Para cada intervalo de partida entre trens com mesmo segmento de origem iniciou-se com 3 trens e para cada nova instância um novo trem foi adicionado até o limite de 7 trens.

O modelo ferroviário 1 tem em sua extremidade oeste o segmento  $s_0$  e em sua extremidade leste o segmento  $s_{10}$ . Os trens com prefixo de final ímpar tem como origem o segmento  $s_0$  e como destino o segmento  $s_{10}$ , já os trens com prefixos de final par tem como origem o segmento  $s_{10}$  e como destino o segmento  $s_0$ . As velocidades dos trens para cada segmento estão no Apêndice A.

A Tabela 5.1 mostra os horários de saída de cada trem para o Cenário 1, que considera um intervalo de 4 horas entre a partida de cada trem com mesma origem.

Os tempos de parada para cruzamento e ultrapassagem e os tempos de processamento podem ser vistos nas Tabelas 5.2 e 5.3.

Analisando as soluções do AP<sub>PLMI</sub> e do AP<sub>GA</sub> na Tabela 5.2 pode-se verificar que o algoritmo de decisão de prioridade foi capaz de seguir suas referências em todos os casos. Isso mostra que para os casos onde não ocorrem perturbações no deslocamento dos trens, o

algoritmo atinge seu objetivo com grande eficácia e com baixo tempo de processamento, como pode ser visto na Tabela 5.3. O algoritmo  $AP_{PAT}$  conseguiu atingir a solução ótima apenas para o caso com 3 trens, tendo em média um tempo total de paradas 40% maior que a melhor solução.

Trem	Algoritmo				
T1	01:00	01:00	01:00	01:00	01:00
T2	01:00	01:00	01:00	01:00	01:00
T3	05:00	05:00	05:00	05:00	05:00
T4	-	05:00	05:00	05:00	05:00
T5	-	-	09:00	09:00	09:00
T6	-	-	-	09:00	09:00
T7	-	-	-	-	13:00

Tabela 5.1: Instantes de partida de cada trem para o Cenário 1.

Nº trens	Tempo de parada por algoritmo (min)								
	PLIM	GA	$AP_{PLMI}$	$AP_{GA}$	$AP_{PAT}$	$APV_{PAT}$	$APV_{PLMI}$	BA	BAC
3	113.07	113.07	113.07	113.07	113.07	113.07	113.07	113.07	113.07
4	184.07	184.07	184.07	184.07	254.07	254.07	184.07	184.07	223.03
5	223.07	223.07	223.07	223.07	277.13	264.07	223.07	223.07	234.07
6	324.1	324.1	324.1	324.1	459.1	426.07	324.1	324.1	345.1
7	416.13	425.1	416.13	425.1	583.10	517.07	416.13	416.13	436.10

Tabela 5.2: Tempo total de parada para cruzamento e ultrapassagem (minutos) para o Cenário 1.

Como o algoritmo  $AP_{PLMI}$  conseguiu seguir sua referência para todos os casos, o algoritmo  $APV_{PLMI}$  não variou a velocidade dos trens. Já o algoritmo  $APV_{PAT}$  obteve soluções melhores ou iguais as do  $AP_{PAT}$  para todos os casos. Para o caso com 7 trens essa redução foi de 12%. O aumento de velocidade consegue fazer com que os trens se aproximem de suas referências e consigam reduzir o atraso sofrido em cruzamentos anteriores. Isso levou o algoritmo  $APV_{PAT}$  a obter melhores soluções que o  $AP_{PAT}$ .

Nº trens	Tempo de processamento por algoritmo (ms)								
	PLIM	GA	AP <sub>PLMI</sub>	AP <sub>GA</sub>	AP <sub>PAT</sub>	APV <sub>PAT</sub>	APV <sub>PLMI</sub>	BA	BAC
3	328	3531	63	15	31	31	63	156	343
4	844	3754	78	47	31	31	78	125	625
5	5219	4046	94	63	47	44	63	202	1203
6	37547	12937	140	31	31	27	140	500	2047
7	646888	14656	156	31	94	97	156	594	2313

Tabela 5.3: Tempo de processamento (milisegundos) para o Cenário 1.

O algoritmo BA também atingiu para todos os casos a solução ótima, com tempos de processamento bastante inferiores aos do algoritmo PLIM. Para o caso com 7 trens este tempo foi cerca de 1000 vezes menor. O algoritmo BAC não atingiu o ótimo para todas as instâncias, mas se manteve sempre muito próximo a ela, ficando cerca de 20% acima do ótimo no pior caso.

A Figura 5.1 mostra a variação do total de tempo de paradas com o aumento do número de trens. Pode-se ver na figura que a cada trem inserido na programação, maior é o tempo de parada total. Isso se explica pelo aumento no número de cruzamentos.

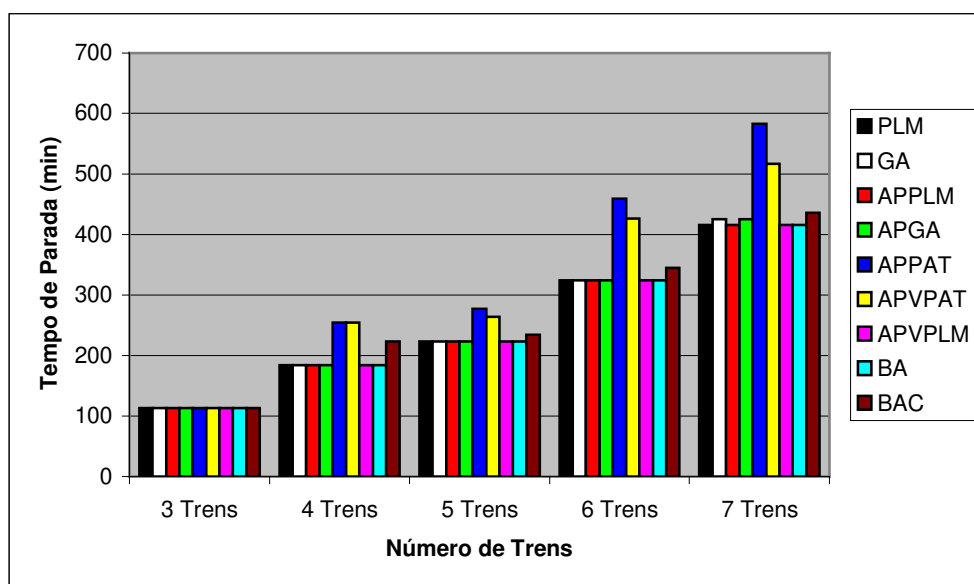


Figura 5.1: Variação do tempo de parada (minutos) com o número de trens para o Cenário 1.



Ainda na Figura 5.1 podemos ver que o algoritmo que teve pior desempenho em tempo total de paradas foi o AP<sub>PAT</sub>. A referência utilizada por este algoritmo não provém de uma solução factível otimizada, portanto o AP<sub>PAT</sub> tem poucas informações de como solucionar os conflitos. Isso pode levar este algoritmo a soluções bastante ruins.

Sobre os tempos de processamento, na Tabela 5.3, pode-se ver que, como era esperado, o PLIM aumenta exponencialmente o seu tempo de processamento com o aumento da complexidade do problema. O GA tem para os casos com 3 e 4 trens tempos de processamento bastante superior a todos os outros algoritmos, mas esse tempo não cresce exponencialmente e no caso com 7 trens é cerca de 44 vezes menor que o do PLIM. Os algoritmos de decisão de prioridade, AP<sub>PLMI</sub>, AP<sub>GA</sub>, AP<sub>PAT</sub>, e de decisão de velocidade APV<sub>PLMI</sub> e APV<sub>PAT</sub> são os que obtêm menores tempos de processamento, sendo menor que 200 milissegundos para todos os casos. Essa característica está de acordo com o objetivo destes algoritmos, que é a aplicação em tempo real.

Os algoritmos de busca, principalmente o BA, são os que têm melhor custo benefício para estes casos. Conseguem uma solução de qualidade, com baixo tempo de processamento e são independentes de outros algoritmos.

Continuando, a Tabela 5.4 mostra os horários de saída de cada trem para o Cenário 2, que considera um intervalo de 3 horas entre a partida de cada trem com mesma origem.

Trem	Algoritmo				
T1	01:00	01:00	01:00	01:00	01:00
T2	01:00	01:00	01:00	01:00	01:00
T3	04:00	04:00	04:00	04:00	04:00
T4	-	04:00	04:00	04:00	04:00
T5	-	-	07:00	07:00	07:00
T6	-	-	-	07:00	07:00
T7	-	-	-	-	10:00

Tabela 5.4: Instantes de partida de cada trem para o Cenário 2.

Os resultado de tempo de paradas para cruzamento e ultrapassagem e tempo de processamento podem ser vistos nas Tabelas 5.5 e 5.6

N° trens	Tempo de parada por algoritmo (min)								
	PLIM	GA	AP <sub>PLMI</sub>	AP <sub>GA</sub>	AP <sub>PAT</sub>	APV <sub>PAT</sub>	APV <sub>PLMI</sub>	BA	BAC
3	81.03	81.03	81.03	81.03	81.03	81.03	81.03	81.03	81.03
4	102.7	102.7	102.7	102.7	102.7	102.07	102.7	102.7	102.7
5	153.1	153.1	153.1	153.1	153.1	153.1	153.1	153.1	153.1
6	317.27	320.13	317.27	320.13	317.27	254.73	317.27	317.27	317.27
7	495.3	509.2	495.3	509.2	495.3	406.11	495.3	495.3	495.3

Tabela 5.5: Tempo de parada para cruzamento e ultrapassagem (minutos) para o Cenário 2.

A Tabela 5.5 mostra que para estas instâncias todos os algoritmos obtiveram um desempenho satisfatório em tempo de paradas. O gráfico da Figura 5.2 mostra de forma comparativa os resultados obtidos pelos algoritmos. Pode-se ver na figura que para os casos com 3, 4 e 5 trens todos algoritmos atingiram o ótimo e que para os casos com 6 e 7 trens apenas o GA e AP<sub>GA</sub> não o fizeram. O algoritmo APV<sub>PAT</sub> conseguiu, soluções com tempos de paradas para cruzamento e ultrapassagem menor que a solução ótima para 6 e 7 trens, o que só é possível devido à variação de velocidade, pois permite que os trens percorram os segmentos com velocidades maiores que as velocidades consideradas pelo PLIM. Isso não significa que o APV<sub>PAT</sub> obtém soluções melhores que o PLIM, já que ambos não utilizam as mesmas velocidades por segmento.

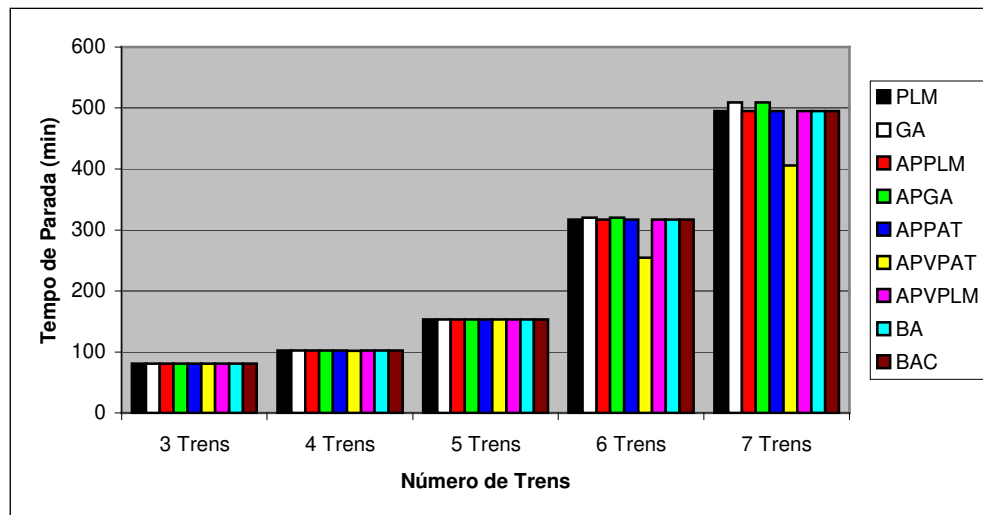


Figura 5.2: Variação do tempo de parada (minutos) com o número de trens para o Cenário 2.

A Figura 5.3 mostra as soluções obtidas pelo PLIM e pelo  $APV_{PAT}$  para o caso com 7 trens.

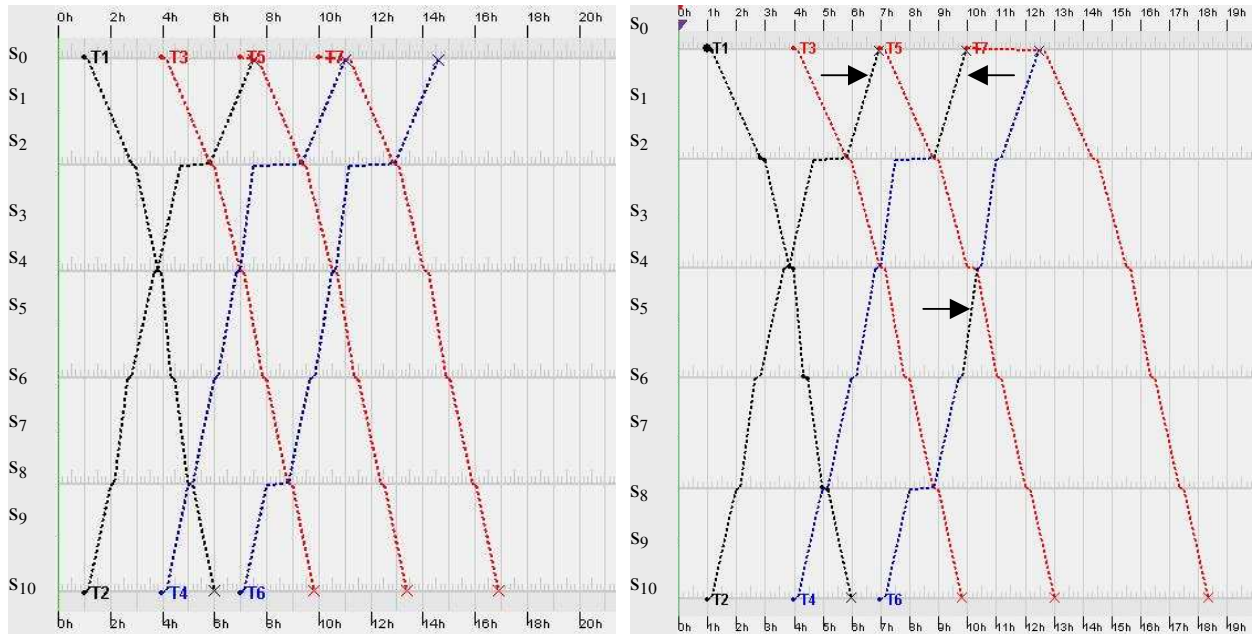


Figura 5.3: Solução obtida pelo PLIM (esquerda) e solução obtida pelo  $APV_{PAT}$  (direita) para o Cenário 2

Os cruzamentos apontados na Figura 5.3 mostram as situações nas quais os trens tiverem suas velocidades alteradas. Quando T2 decidiu a ocupação do segmento  $s_1$  com T5, T2 foi o escolhido como preferencial, pois já havia sofrido um grande atraso no segmento  $s_2$ , e teve sua velocidade aumentada. Isso possibilitou a ele chegar no segmento  $s_0$  antes da partida de T5. Pode ver na solução do PLIM que isto não ocorreu, fazendo com que T5 aguarda-se no segmento  $s_0$  a chegada de T1.

A Tabela 5.6 mostra os resultados de tempo de processamento. Os tempos de processamento para os testes com intervalo de 3 horas entre os horários de saída dos trens (Tabela 5.6) foi bastante semelhante aos resultados obtidos para um intervalo de 4 horas (Tabela 5.3). A diferença é que os resultados da Tabela 5.6 foram um pouco superiores. Isso se explica pelo aumento no número de conflitos gerados pela aproximação entre os trens.

N° trens	Tempo de processamento por algoritmo (ms)								
	PLIM	GA	AP <sub>PLMI</sub>	AP <sub>GA</sub>	AP <sub>PAT</sub>	APV <sub>PAT</sub>	APV <sub>PLMI</sub>	BA	BAC
3	562	6110	109	15	16	16	109	156	390
4	766	9391	103	15	31	32	103	187	875
5	8484	11141	109	47	31	31	109	281	1484
6	65703	13859	125	15	47	49	125	579	1609
7	730157	20547	203	43	93	94	203	656	2672

Tabela 5.6: Tempo de processamento (milisegundos) para Cenário 2.

	Instante de partida				
T1	01:00	01:00	01:00	01:00	01:00
T2	01:00	01:00	01:00	01:00	01:00
T3	03:00	03:00	03:00	03:00	03:00
T4	-	03:00	03:00	03:00	03:00
T5	-	-	05:00	05:00	05:00
T6	-	-	-	05:00	05:00
T7	-	-	-	-	07:00

Tabela 5.7: Instante de partida dos trens para o Cenário 3.

A Tabela 5.7 mostra os horários de partida de cada trem para o Cenário 3, que considera um intervalo de 2 horas entre a partida de trens. Os resultados de tempo de parada de paradas para cruzamento e ultrapassagem para a programação de trens da Tabela 5.7 estão na Tabela 5.8.

Pode-se ver na Tabela 5.8 que o GA atingiu o ótimo para todos os casos e que também os algoritmos de decisão de prioridade AP<sub>PLMI</sub>, AP<sub>GA</sub> e APV<sub>PLMI</sub> seguiram suas referências para todos os casos. O algoritmo AP<sub>PAT</sub> teve novamente os resultados com maior tempo de paradas, sendo que sua variação, APV<sub>PAT</sub>, obteve melhores soluções para 6 e 7 trens. Os algoritmos de busca BA e BAC obtiveram a solução ótima para 3, 4 e 5 trens. Para os casos com 6 e 7 obtiveram, respectivamente, no pior caso resultados 11% e 17% superiores.

Nº trens	Tempo de parada por algoritmo (min)								
	PLIM	GA	AP <sub>PLMI</sub>	AP <sub>GA</sub>	AP <sub>PAT</sub>	APV <sub>PAT</sub>	APV <sub>PLMI</sub>	BA	BAC
3	21,03	21,03	21,03	21,03	21,03	21,03	21,03	21,03	21,03
4	73,10	73,10	73,10	73,10	73,10	73,10	73,10	73,10	73,10
5	114,13	114,13	114,13	114,13	278,13	278,13	114,13	114,13	114,13
6	416,17	416,17	416,17	416,17	513,27	499,41	416,17	462,3	508,23
7	699,2	699,2	699,2	699,2	859,33	697,77	699,2	767,3	822,27

Tabela 5.8: Tempo de parada para cruzamento e ultrapassagem (minutos) para o Cenário 3.

A Figura 5.4 mostra a variação do total de tempo de paradas com o aumento do número de trens.

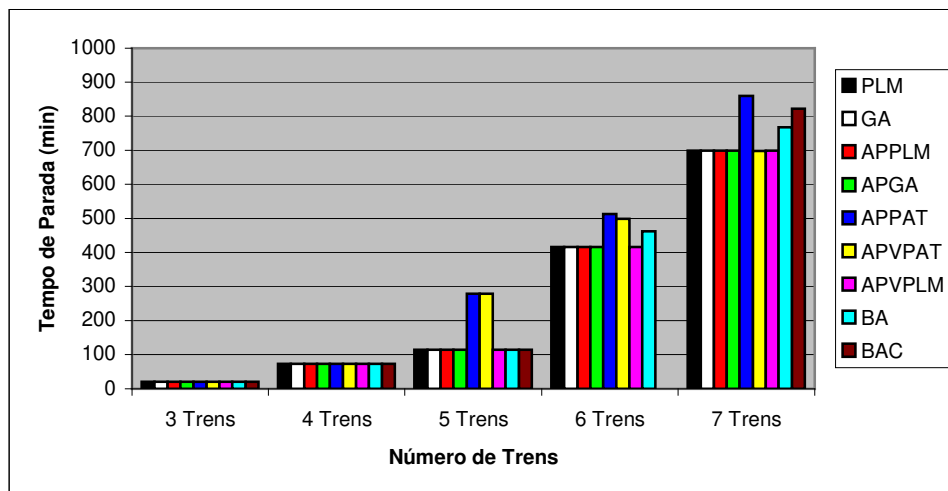


Figura 5.4: Variação do tempo de parada (minutos) com o número de trens para o Cenário3.

Comparando-se as Figuras 5.1, 5.2 e 5.4 verifica-se que com a redução do intervalo entre os horários de saída dos trens ocorre um aumento significativo no tempo de paradas. Para o caso com 7 trens a solução ótima teve tempo de paradas de 416,13 minutos, 495,3 minutos e 699,2 minutos, respectivamente. O aumento foi significativamente maior para o intervalo de 2 horas. Isso mostra que existe uma saturação da linha ferroviária com o aumento da densidade de tráfego.

A Tabela 5.9 mostra o tempo de processamento de cada algoritmo

Nº trens	Tempo de processamento por algoritmo (ms)								
	PLIM	GA	AP <sub>PLMI</sub>	AP <sub>GA</sub>	AP <sub>PAT</sub>	APV <sub>PAT</sub>	APV <sub>PLMI</sub>	BA	BAC
3	313	6719	78	15	31	31	78	188	438
4	934	9093	97	15	47	47	97	234	1047
5	6937	12000	93	15	47	48	93	344	1562
6	44765	22828	156	31	94	97	156	609	2579
7	617937	41000	266	47	94	94	266	1250	4281

Tabela 5.9: Tempo de processamento (milisegundos) para o Cenário 3.

A tendência de aumento no tempo de processamento com o aumento na densidade de tráfego se confirma para o intervalo de 2 horas, conforme a Tabela 5.9.

### 5.2.2 Resultados com modelo ferroviário 2.

O modelo ferroviário 2 considera uma linha composta por 9 pátios de cruzamento e ultrapassagem e 8 segmentos de linha singela. Seguindo o procedimento adotado na seção anterior, serão consideradas instâncias com 3, 4, 5, 6 e 7 trens para o modelo ferroviário 2. Foram geradas 15 instâncias classificadas por intervalo de tempo entre os horários de partida dos trens, considerando três intervalos: 4 horas, 3 horas e 2 horas. Para cada intervalo de partida entre trens com mesmo segmento de origem iniciou-se com 3 trens e para cada nova instância um novo trem foi adicionado até o limite de 7 trens.

O modelo ferroviário 2 tem em suas extremidade oeste o segmento  $s_0$  e em sua extremidade leste o segmento  $s_{16}$ . Os trens com prefixo de final ímpar têm como origem o segmento  $s_0$  e como destino o segmento  $s_{16}$ , já os trens com prefixos de final par tem como origem o segmento  $s_{16}$  e como destino o segmento  $s_0$ .

A Tabela 5.10 mostra os horários de partida de cada trem para o Cenário 4, que considera um intervalo de 4 horas entre a partida de cada trem com mesma origem.

Assim como nos resultados da seção anterior os algoritmos de decisão de prioridade baseado em uma referência factível,  $AP_{PLMI}$ ,  $AP_{GA}$  e  $APV_{PLMI}$  conseguiram seguir suas referências para todos os casos. Já o algoritmo baseado em uma referência não factível,  $AP_{PAT}$ , obteve os piores resultados. O  $APV_{PAT}$  superou o  $AP_{PAT}$  para todos os casos, para 7 trens essa redução foi maior que 100%. O algoritmo de busca, BA, atingiu o ótimo para 4 dos 5 casos, obtendo uma solução 4,7% acima para 6 trens. A Figura 5.5 ilustra os resultados. Pode-se ver na figura que o algoritmo BAC obteve uma solução 67% superior à ótima para o caso com 7 trens.

Trem	Instante de partida				
	01:00	01:00	01:00	01:00	01:00
T1	01:00	01:00	01:00	01:00	01:00
T2	01:00	01:00	01:00	01:00	01:00
T3	05:00	05:00	05:00	05:00	05:00
T4	-	05:00	05:00	05:00	05:00
T5	-	-	09:00	09:00	09:00
T6	-	-	-	09:00	09:00
T7	-	-	-	-	13:00

Tabela 5.10: Instantes de partida dos trens para o Cenário 4.

Os resultados de tempo de paradas para cruzamento e ultrapassagem da programação de trens da Tabela 5.10 são mostrados na Tabela 5.11.

N° trens	Tempo de parada por algoritmo (min)								
	PLIM	GA	$AP_{PLMI}$	$AP_{GA}$	$AP_{PAT}$	$APV_{PAT}$	$APV_{PLMI}$	BA	BAC
3	52,07	52,07	52,07	52,07	83,1	72,58	52,07	52,07	52,07
4	132,07	132,07	132,07	132,07	154,13	143,61	132,07	132,07	132,07
5	177,23	177,23	177,23	177,23	177,23	153,6	177,23	177,23	184,13
6	233,07	241,37	233,07	241,37	319,23	194,63	233,07	244,13	244,13
7	243,07	273,4	243,07	273,4	433,33	204,63	243,07	243,07	406,17

Tabela 5.11: Tempo de parada para cruzamento e ultrapassagem (minutos) para o Cenário 4

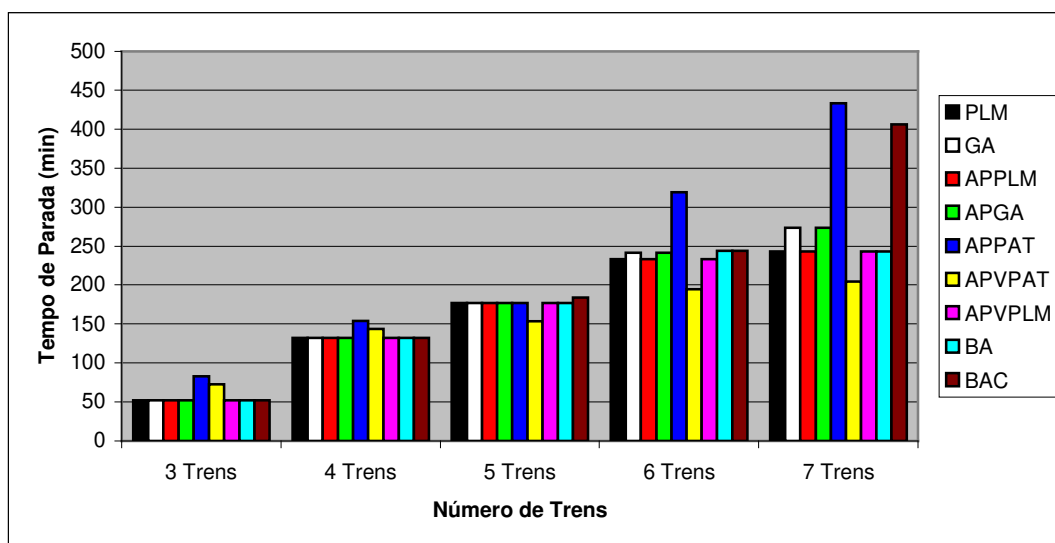


Figura 5.5: Variação do tempo de parada (minutos) com o número de trens para o Cenário 4

A Tabela 5.12 mostra os resultados de tempos de processamento.

N° Trens	Tempo de processamento por algoritmo (ms)								
	PLIM	GA	AP <sub>PLMI</sub>	AP <sub>GA</sub>	AP <sub>PAT</sub>	APV <sub>PAT</sub>	APV <sub>PLMI</sub>	BA	BAC
3	406	11406	94	16	109	109	94	266	828
4	4937	16485	172	16	47	48	172	188	1156
5	46297	21641	187	32	110	98	187	375	1985
6	508625	27953	235	31	78	75	235	390	2250
7	2466234	33141	219	47	78	82	219	550	2500

Tabela 5.12: Tempo de processamento (milissegundos) para o Cenário 4

O aumento no número de segmentos levou a uma redução do tempo de paradas, como pode ser confirmado comparando a Tabela 5.11 com a Tabela 5.2. Os tempo de processamento foram bastante próximos aos da Tabela 5.3, exceto para o PLIM que teve uma aumento significativo, sendo que a solução para 7 trens levou cerca de 40 minutos para ser obtida, conforme a Tabela 5.12. Na Tabela 5.3 pode-se ver que o caso para 7 trens levou cerca de 10 minutos.

A Tabela 5.13 mostra os horários de partida de cada para o Cenário 5, que considera um intervalo de 3 horas entre a partida de cada trem com mesma origem.



Trem	Instante de partida				
	01:00	01:00	01:00	01:00	01:00
T1	01:00	01:00	01:00	01:00	01:00
T2	01:00	01:00	01:00	01:00	01:00
T3	04:00	04:00	04:00	04:00	04:00
T4	-	04:00	04:00	04:00	04:00
T5	-	-	07:00	07:00	07:00
T6	-	-	-	07:00	07:00
T7	-	-	-	-	10:00

Tabela 5.13: Instante de partida dos trens para o Cenário 5.

Os resultados de tempo de paradas para cruzamento e ultrapassagem para a programação de trens da Tabela 5.13 são mostrados na Tabela 5.14.

Nº trens	Tempo de parada por algoritmo (min)								
	PLIM	GA	AP <sub>PLMI</sub>	AP <sub>GA</sub>	AP <sub>PAT</sub>	APV <sub>PAT</sub>	APV <sub>PLMI</sub>	BA	BAC
3	71,03	92,03	71,03	92,03	143,1	143,1	71,03	92,03	92,03
4	123,10	137,2	123,10	137,2	227,73	227,23	123,10	137,2	154,13
5	151,33	151,33	151,33	151,33	360,3	266,2	151,33	151,33	191,30
6	307,2	307,2	307,2	307,2	552,33	498,83	307,2	307,2	307,2
7	509,23	509,47	509,23	509,47	767,33	630,17	509,23	543,3	593,4

Tabela 5.14: Tempo de parada para cruzamento e ultrapassagem (minutos) trens para o Cenário 5.

Os resultados da Tabela 5.14 seguem o padrão dos resultados anteriores. Os algoritmos de decisão de prioridade mantêm suas referências. O GA e BA conseguem atingir o ótimo para os casos com 5 e 6 trens e para os outros casos conseguem soluções muito próximas às ótimas. O algoritmo BAC obteve soluções com maior tempo de paradas em relação ao ótimo para os casos com 4, 5 e 7 trens. O APV<sub>PAT</sub> obteve novamente soluções melhores que sua versão com velocidade constante, mas ainda ficando bastante acima da solução ótima. O gráfico comparativo das soluções é mostrado na Figura 5.6.

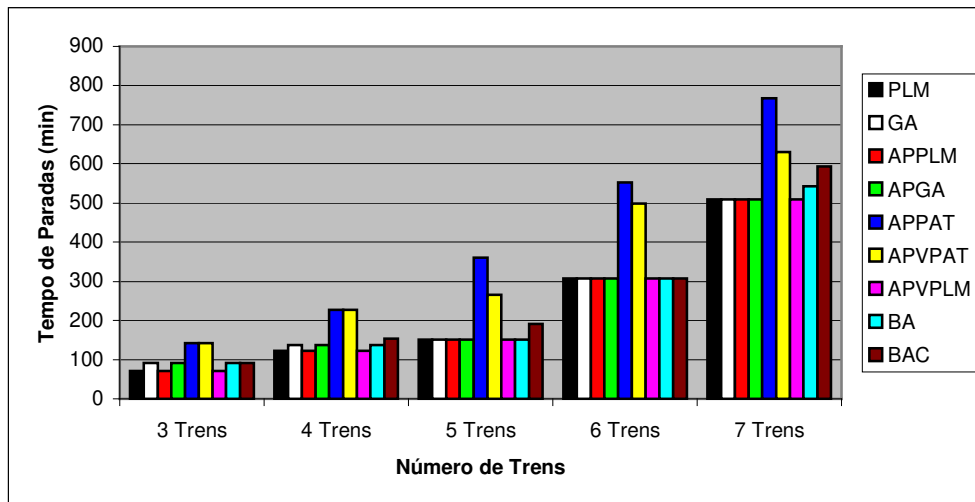


Figura 5.6: Variação do tempo de parada (minutos) com o número de trens para o Cenário 5.

A solução ótima e a solução obtida pelo BA para 7 trens podem ser vistas na Figura 5.7.

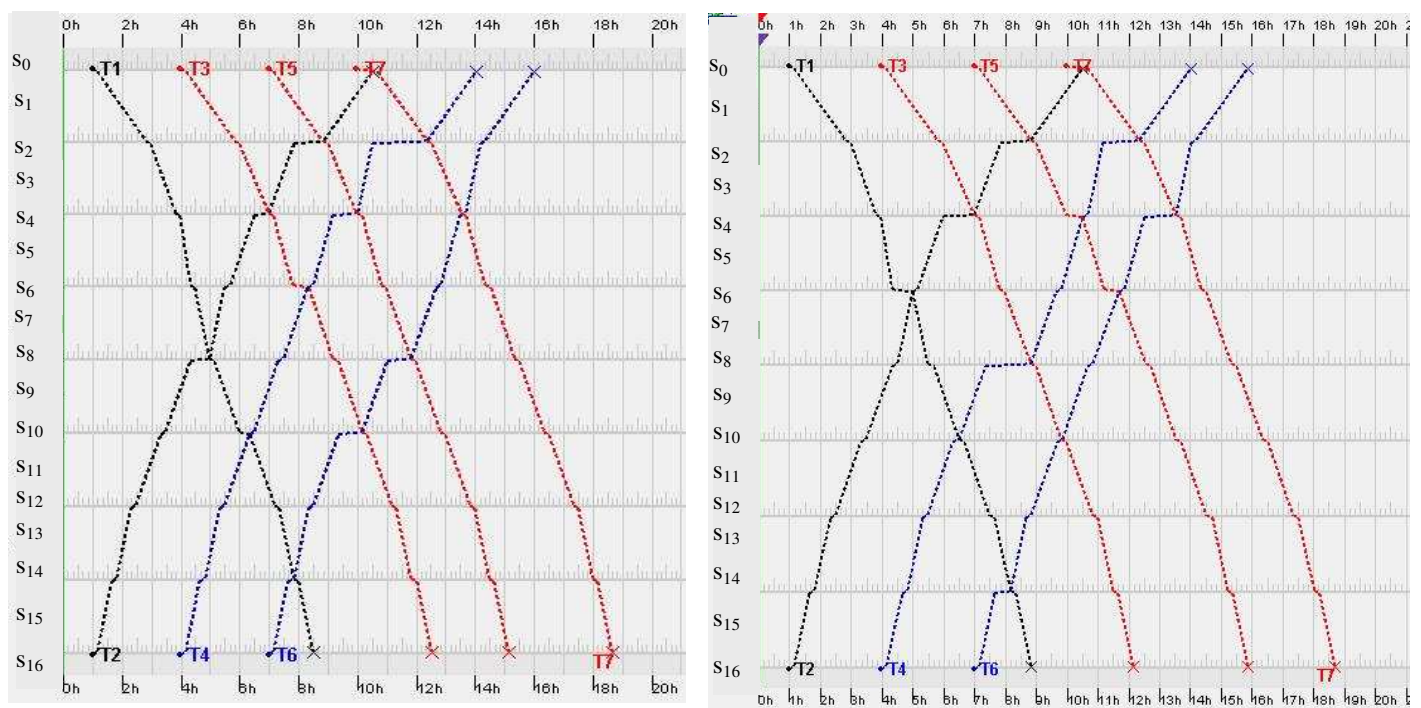


Figura 5.7: Solução obtida pelo PLIM (esquerda) e solução obtida pelo BA (direita) para o Cenário 5.

É interessante verificar na Figura 5.7 que a primeira decisão de preferência, entre os trens T1 e T2, nas soluções do PLIM e BA foram diferentes. O fato de a primeira decisão tomada ter sido diferente levou a soluções muito distintas no final do processo. A decisão

de um cruzamento tem grande influência sobre toda a solução e pode levar a soluções consideravelmente diferentes

A Tabela 5.15 mostra os tempos de processamento. Pode-se ver que em geral os algoritmos apresentaram um comportamento semelhante aos casos anteriores. Apenas o PLIM teve um grande aumento no tempo de processamento, levando cerca de 2 horas e meia para encontrar a solução para 7 trens. Os algoritmos de decisão de preferência e o algoritmo BA continuam obtendo soluções em tempos inferiores a 1 segundo.

Nº trens	Tempo de processamento por algoritmo (ms)								
	PLIM	GA	AP <sub>PLMI</sub>	AP <sub>GA</sub>	AP <sub>PAT</sub>	APV <sub>PAT</sub>	APV <sub>PLMI</sub>	BA	BAC
3	484	12859	78	31	47	49	78	125	828
4	3782	18078	141	31	78	83	141	188	1593
5	80016	21687	203	47	47	45	203	344	2031
6	1046406	30938	187	31	47	48	187	500	2531
7	9145281	35375	250	47	78	45	250	656	3343

Tabela 5.15: Tempo de processamento (milissegundos) para o Cenário 5.

A Tabela 5.16 mostra os horários de partida de cada trem para o Cenário 6, que considera um intervalo de 2 horas entre a partida de cada trem com mesma origem.

Trem	Instante de partida				
T1	01:00	01:00	01:00	01:00	01:00
T2	01:00	01:00	01:00	01:00	01:00
T3	03:00	03:00	03:00	03:00	03:00
T4	-	03:00	03:00	03:00	03:00
T5	-	-	05:00	05:00	05:00
T6	-	-	-	05:00	05:00
T7	-	-	-	-	07:00

Tabela 5.16: Instante de partida dos trens para o Cenário 6.

Os resultados de tempo de paradas estão na Tabela 5.17 e os histogramas relativos estão na Figura 5.8.

Nº trens	Tempo de parada por algoritmo (min)								
	PLIM	GA	AP <sub>PLMI</sub>	AP <sub>GA</sub>	AP <sub>PAT</sub>	APV <sub>PAT</sub>	APV <sub>PLMI</sub>	BA	BAC
3	51,03	51,03	51,03	51,03	73,10	62,56	51,03	51,03	51,03
4	132,07	132,07	132,07	132,07	164,13	143,6	132,07	132,07	132,07
5	152,07	152,07	152,07	152,07	317,2	194,65	152,07	215,17	152,07
6	203,1	286,2	203,1	286,2	381,3	258,53	203,1	297,23	203,1
7	498,23	518,17	498,23	518,17	645,37	580,3	498,23	498,23	528,17

Tabela 5.17: Tempo de parada para cruzamento e ultrapassagem (minutos) para o Cenário 6.

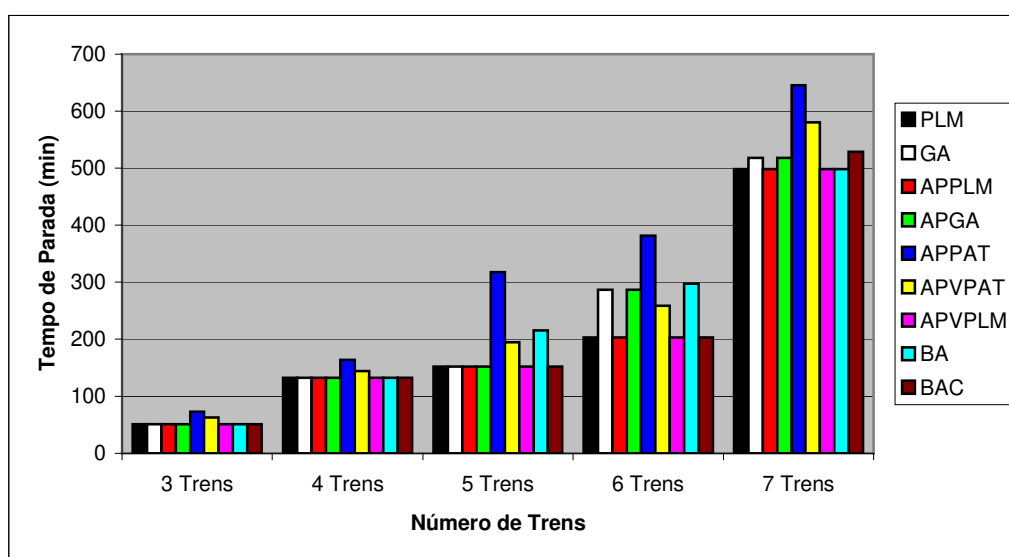


Figura 5.8: Variação do tempo de parada (minutos) com o número de trens para o Cenário 6.

Observando a Figura 5.8 pode-se verificar que o algoritmo de decisão de prioridade tendo como referência o Programa de Atividades de Trem foi novamente o com piores resultados de tempos de parada. A variação da velocidade para a mesma referência obteve melhores resultados para todos os casos, mas como nos casos anteriores superiores à solução ótima. O algoritmo BA conseguiu atingir o ótimo para os casos com 3, 4 e 7 trens, sendo que para o caso com 5 trens teve seu tempo de paradas 60% superior e para 6 trens 46%. Estes foram os casos de pior desempenho do BA. Já o algoritmo não BAC obteve a solução ótima apenas para o caso com 7 trens, sendo 6% superior neste caso. O GA obteve resultados 40% superior ao ótimo para 6 trens e 4% superior para 7 trens.

A Tabela 5.18 mostra os tempos de processamento.

Nº trens	Tempo de processamento por algoritmo (ms)								
	PLIM	GA	AP <sub>PLMI</sub>	AP <sub>GA</sub>	AP <sub>PAT</sub>	APV <sub>PAT</sub>	APV <sub>PLMI</sub>	BA	BAC
3	484	14110	110	16	47	47	110	188	938
4	5907	18219	109	16	94	97	109	360	1704
5	143266	24828	144	31	141	139	144	593	2750
6	1311187	32594	250	74	47	50	250	468	3797
7	9849672	36450	250	68	50	52	250	1100	3219

Tabela 5.18: Tempo de processamento (milissegundos) para o Cenário 6.

Os tempos de processamento seguem o mesmo padrão dos resultados anteriores. Os algoritmos de decisão de preferência e velocidade obtêm tempos menores que 1 segundo. O PLIM é o que tem maior aumento no tempo de processamento com a inserção de novos trens. O algoritmo BA obteve tempo de processamento superior a 1 segundo apenas para o caso com 7 trens.

### 5.2.3 Resultados com flexibilização do PAT

Os resultados das seções anteriores mostraram que quando o algoritmo de decisão nebuloso utiliza como referência o PAT de cada trem, as soluções obtidas apresentam, na maioria dos casos, tempos de parada para cruzamento e ultrapassagem bastante superiores às soluções ótimas. O PAT de cada trem não considera as paradas necessárias para cruzamento e ultrapassagem e com isso o PAT de todos os trens formam um planejamento de circulação de trens inactível. Quando o PAT de cada trem é utilizado como referência de percurso, é impossível que todos os trens mantenham suas referências, pois vai haver situações nas quais um ou mais trens concorrem pelo uso de um segmento em um mesmo instante de tempo e será necessário decidir qual trem ocupará o segmento e quais trens deverão permanecer em seus segmentos atuais.

Esta seção apresenta um estudo de como a flexibilização do PAT influencia os resultados. O PAT de um trem contém os tempos de entrada e partida do trem em cada segmento considerando que o trem percorre a linha ferroviária sem paradas para cruzamento e ultrapassagem. O PAT flexibilizado de um trem considera que o trem possui um instante desejado de chegada ao seu destino e os instantes de chegada e partida do trem

em cada segmento são determinados de modo a cumprir esse horário, mas também sem considerar paradas para cruzamento e ultrapassagem. A Figura 5.9 mostra como é feita a flexibilização do PAT. O instante desejado de chegada do trem é que determina o PAT flexibilizado e pode ser um parâmetro do modelo ou também pode ser fornecido por um especialista ou por algum algoritmo de otimização. Outra opção para fornecer horários desejados para os trens é utilizar dados históricos dos tempos de percurso dos trens que considerem tempos de parada para ultrapassagem e cruzamento.

Para o experimento de flexibilização do PAT será utilizada uma instância do modelo ferroviário 2 com cinco trens circulando. Os trens T1 e T3 tem origem no segmento  $s_0$  e como destino o segmento  $s_{16}$  e os trens T2, T4 e T6 tem origem no segmento  $s_{16}$  e como destino o segmento  $s_0$ . A Tabela 5.19 mostra os instantes de partida de cada trem. Os instantes desejados de chegada dos trens serão os instantes de chegada obtidos pelo modelo de programação linear inteira mista de Valle *et al*, 2005.

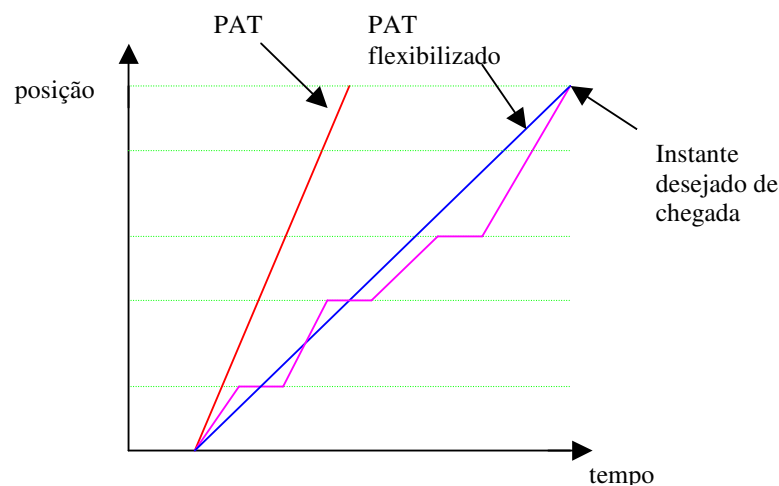


Figura 5.9: Flexibilização do PAT

Trem	Instante de partida
T1	01:00
T2	01:00
T3	05:00
T4	03:00
T6	06:00

Tabela 5.19: Instante de partida dos trens.

Os algoritmos utilizados são os seguintes:

1. Modelo de Programação Linear Inteira-Mista, PLIM, proposto por Valle *et al*, 2005;
2. Algoritmo de Decisão de Preferência utilizando o Programa de Atividades de Trem como referência,  $AP_{PAT}$ ;
3. Algoritmo de Decisão de Preferência utilizando o Programa de Atividades de Trem Flexibilizado como referência,  $AP_{PATF}$ ;
4. Algoritmo de Decisão de Preferência e Velocidade utilizando o Programa de Atividades de Trem como referência,  $APV_{PAT}$ ;

Os parâmetros para o  $APV_{PAT}$  são os seguintes:

- limites de velocidade:
  - velocidade máxima =  $1.5 * (\text{velocidade média})$ ;
  - velocidade mínima =  $0.5 * (\text{velocidade média})$ ;

A Tabela 5.20 mostra os resultados de tempos de parada em minutos para cruzamento e ultrapassagem obtidos.

Algoritmo	PLIM	$AP_{PAT}$	$AP_{PATF}$	$APV_{PAT}$
Tempo de Parada (min)	176.2	228.27	176.2	193.87

Tabela 5.20: Tempo de parada em minutos para cruzamento e ultrapassagem para a flexibilização do PAT.

Pode ver na Tabela 5, que o algoritmo de decisão de preferência utilizando o PAT flexibilizado como referência conseguiu obter a solução ótima tendo apenas as informações dos instantes de chegada de cada trem. Já os algoritmos  $AP_{PAT}$  e  $APV_{PAT}$  obtiveram soluções com tempos de parada para cruzamento e ultrapassagem superiores a solução ótima. A Figura 5.10 mostra o gráfico de trens da solução ótima e a Figura 5.11 mostra a solução obtida pelo  $AP_{PATF}$

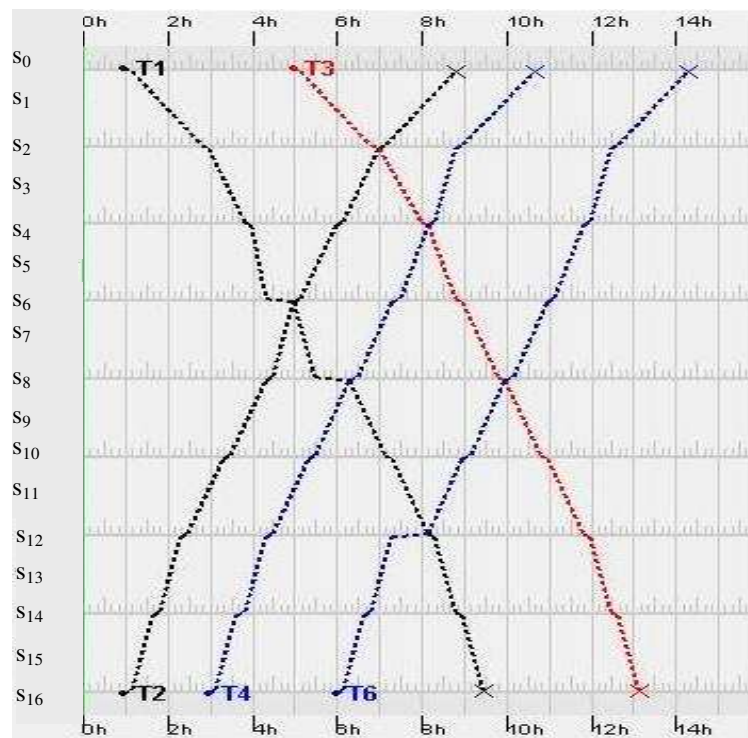


Figura 5.10: Solução ótima.

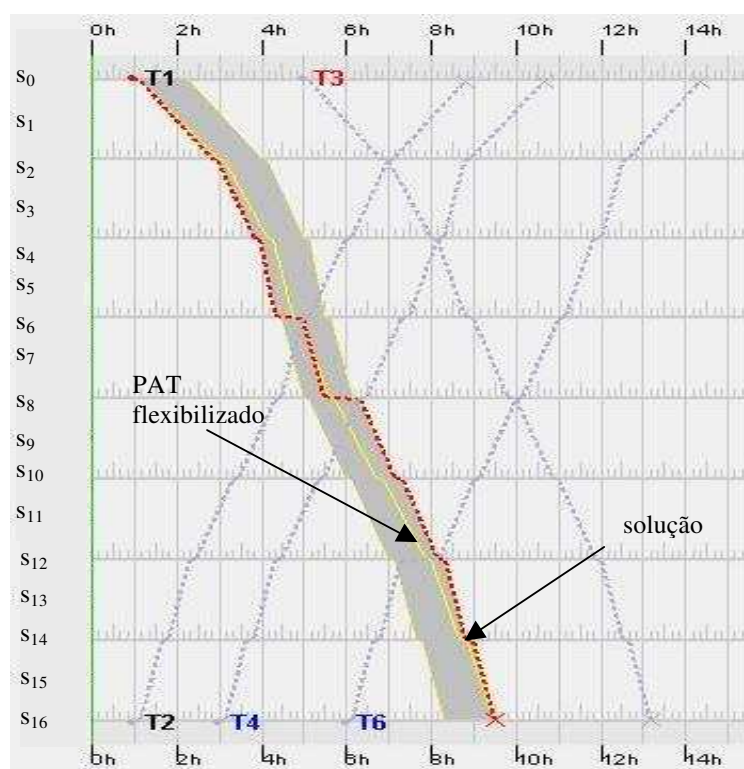


Figura 5.11: Solução obtida com a flexibilização do PAT.



A Figura 5.11 mostra em destaque o PAT flexibilizado para o trem T1 que foi utilizado como sua referência de percurso para obtenção da solução. Pode verificar que o PAT flexibilizado possui o mesmo instante de chegada no segmento  $s_{16}$  que a solução ótima da Figura 5.10, mas não considera as paradas para cruzamento e ultrapassagem.

Este experimento mostra que o uso do PAT flexibilizado pode ser uma alternativa interessante para os casos que uma referência otimizada factível não está disponível para ser utilizada como referência. Para que a efetividade dessa estratégia seja realmente comprovada seria necessário um estudo mais completo, que utilizasse um número maior de instâncias e também outras fontes dos instantes desejados de chegada dos trens. Esse estudo não está no escopo deste trabalho e faz parte das propostas de trabalhos futuros.

#### **5.2.4 Resultados com perturbações no percurso dos trens.**

As seções 5.2.1 e 5.2.2 mostraram que o algoritmo de decisão de preferência de trens consegue seguir com êxito suas referências de percurso quando estas são provenientes de um planejamento factível e quando não há perturbações no deslocamento dos trens na linha ferroviária. Esta seção apresenta um estudo da consequência da inserção de perturbações no deslocamento dos trens. Será considerada como uma perturbação no deslocamento de um trem uma situação na qual um trem é obrigado a percorrer um segmento com velocidade inferior a velocidade média do segmento. Em situações reais este tipo de comportamento dos trens é bastante freqüente, quando, por exemplo, um trem está com alguma locomotiva avariada, quando um segmento está com defeito, quando há excesso de chuvas e os trens devem manter velocidades baixas, etc.

Para o experimento de análise do efeito de perturbações será utilizada uma instância do modelo ferroviário 2 com cinco trens circulando. Os trens T1 e T3 tem origem no segmento  $s_0$  e como destino o segmento  $s_{16}$  e os trens T2, T4 e T6 tem origem no segmento  $s_{16}$  e como destino o segmento  $s_0$ . A Tabela 5.21 mostra os instantes de partida de cada trem. As referências de percurso são obtidas pelo modelo de programação linear inteira mista de Valle *et al.* (2005). É importante ressaltar que a referência fornecida ao algoritmo de decisão de preferência não considera a existência das perturbações.

Trem.	Instante de partida
T1	01:00
T2	01:00
T3	05:00
T4	03:00
T6	06:00

Tabela 5.21: Instante de partida dos trens.

Os algoritmos utilizados são os seguintes:

1. Modelo de Programação Linear Inteira-Mista, PLIM, proposto por Valle *et al*, 2005;
2. Algoritmo de Decisão de Preferência utilizando o resultado do PLIM como referência,  $AP_{PLMI}$ ;
3. Algoritmo de Decisão de Preferência e Velocidade utilizando o resultado do PLIM como referência,  $APV_{PLMI}$ ;
4. Algoritmo de Decisão de Preferência utilizando o Programa de Atividades de Trem como referência,  $AP_{PAT}$ ;
5. Algoritmo de Decisão de Preferência utilizando o Programa de Atividades de Trem Flexibilizado como referência,  $AP_{PATF}$ ;
6. Algoritmo de Decisão de Preferência e Velocidade utilizando o Programa de Atividades de Trem como referência,  $APV_{PAT}$ ;
7. Algoritmo de Decisão de Preferência e Velocidade utilizando o Programa de Atividades de Trem Flexibilizado como referência,  $APV_{PATF}$ ;

Os parâmetros para o  $APV_{PAT}$  são os seguintes:

- limites de velocidade:
  - velocidade máxima =  $1.5 * (\text{velocidade média})$ ;
  - velocidade mínima =  $0.5 * (\text{velocidade média})$ ;

A perturbação será inserida no deslocamento do trem T4, o que fará com que ele percorra o segmento  $s_{13}$  com uma velocidade de 9 km/h, enquanto que a velocidade média para este segmento é de 30 km/h.

A Tabela 5.22 mostra os resultados de tempos de parada em minutos para cruzamento e ultrapassagem obtidos.

Algoritmo	PLIM	AP <sub>PLIM</sub>	APV <sub>PLIM</sub>	AP <sub>PAT</sub>	APV <sub>PAT</sub>	AP <sub>PATF</sub>	APV <sub>PATF</sub>
Tempo de Parada (min)	176.2	364.1	337.47	275.13	208.15	364.1	337.47

Tabela 5.22: Tempo de parada em minutos para cruzamento e ultrapassagem

Pode-se verificar na Tabela 5.22 que os algoritmos obtiveram soluções com tempos de parada para cruzamento e ultrapassagem superiores à solução ótima, já que a inserção da perturbação do deslocamento do trem T4 impossibilitou os algoritmos de obterem uma solução idêntica a ótima, pois a solução ótima não considerou esta perturbação. Os algoritmos AP<sub>PLIM</sub> e APV<sub>PLIM</sub> utilizam a solução do PLIM (Figura 5.10) como referência e suas soluções podem ser vistas nas Figura 5.12a e 5.12b, respectivamente.

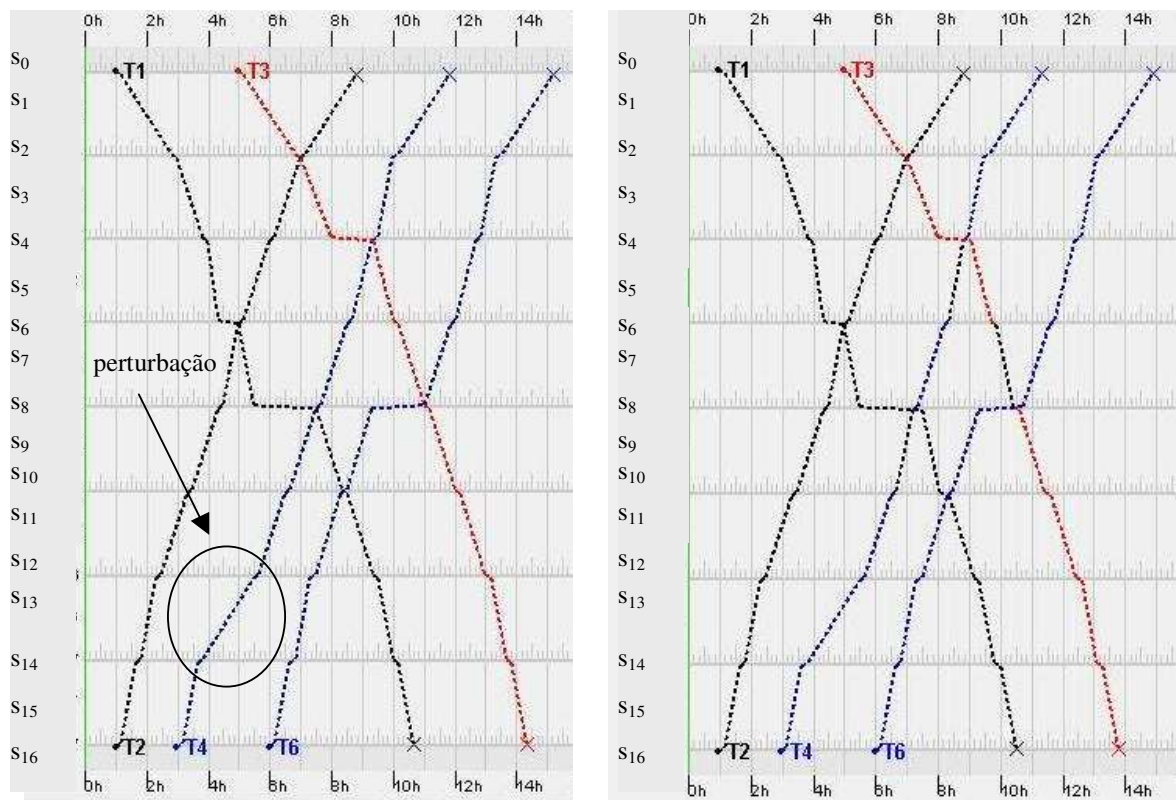


Figura 5.12: a. Solução do AP<sub>PLIM</sub>. b. Solução do APV<sub>PLIM</sub>

O círculo na Figura 5.12a mostra a perturbação no deslocamento do trem T4. Essa perturbação levou o trem T4 a ficar atrasado em relação à sua referência, o que lhe deu preferência no conflito com o trem T1 pela ocupação do segmento  $s_5$  e no conflito com o

trem T3 pela ocupação do segmento  $s_2$ . Isso fez com que os trens T1 e T3 tivessem que permanecer um grande intervalo de tempo parados aguardando a chegada do trem T4. As mesmas situações podem ser vistas na Figura 5.12b, mas neste caso a variação da velocidade dos trens permitiu reduzir os tempos de paradas para cruzamento.

As Figuras 5.13a e 5.13b mostram as soluções obtidas pelos algoritmos  $AP_{PAT}$  e  $APV_{PAT}$ . Pode ver na Figura 5.13a que a decisão de cruzamento entre os trens T1 e T2 foi diferente da decisão das Figuras 5.12a, o que fez com o que o trem T1 cruzasse com o trem T4 no segmento  $s_{10}$ , o que resultou num tempo de paradas para cruzamento menor para a solução da Figura 5.13a em relação à Figura 5.12a. Para a Figura 5.13b a variação da velocidade dos trens também possibilitou uma melhor solução que a Figura 5.13b.

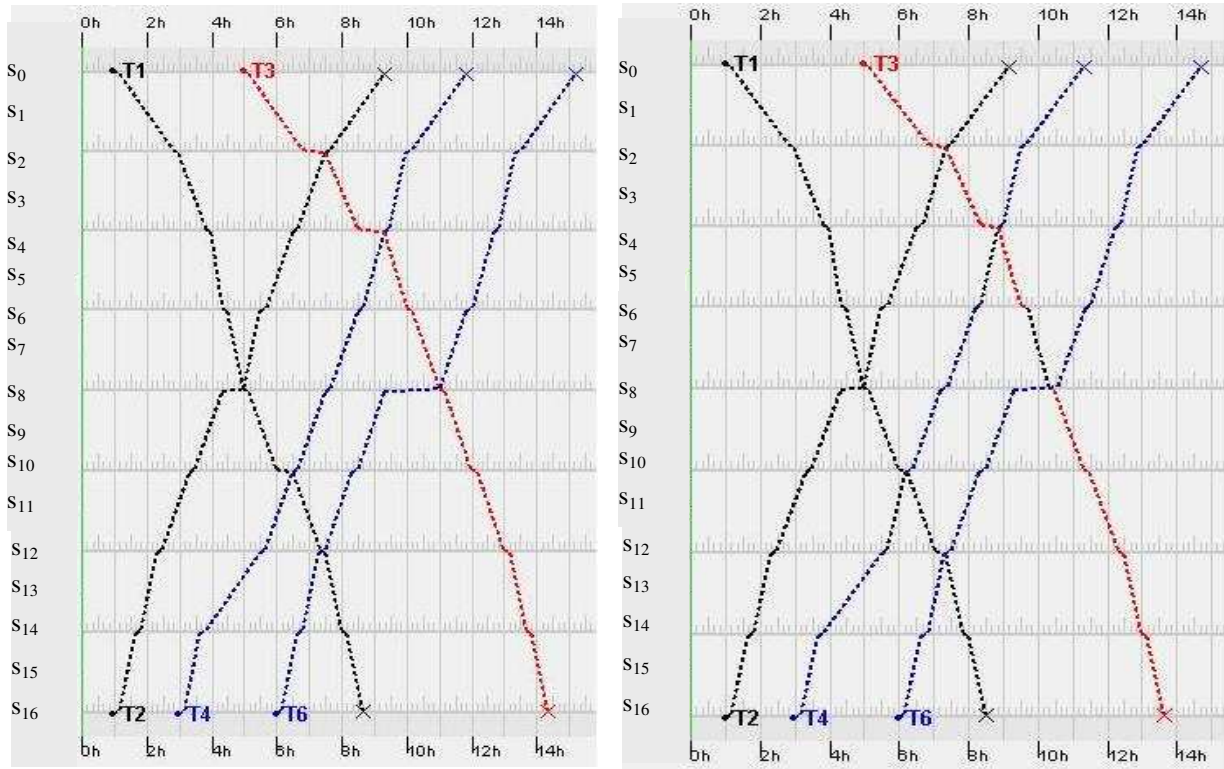


Figura 5.13: a. Solução do  $AP_{PAT}$ . b. Solução do  $APV_{PAT}$

Os algoritmos  $AP_{PATF}$  e  $APV_{PATF}$ , que utilizam o PAT flexibilizado como referência com os horários de chegada dos trens obtidos pelo PLIM, obtiveram as mesmas soluções que os algoritmos  $AP_{PLIM}$  e  $APV_{PLIM}$ , respectivamente.

Pode-se concluir desse experimento que os algoritmos de decisão de preferência e velocidade se adaptam à perturbação no deslocamento de um trem com o objetivo de manter suas referências de percurso. Nos casos apresentados acima essa estratégia levou alguns trens a sofrerem grandes atrasos devido a cruzamentos.

Um experimento interessante seria obter a nova solução ótima após o trem T4 ter reduzido sua velocidade no segmento  $s_{13}$ . Para isso foi gerada uma nova programação de trens com novos instantes de partidas e novas origens para os trens, considerando a posição dos trens quando o trem T4 terminou de percorrer o segmento  $s_{13}$ . A Tabela 5. mostra os novos horários de partida e novas origens dos trens.

Trem.	Instante de partida	Origem	Destino
T1	05:30	S8	S16
T2	06:00	S4	S0
T3	05:00	S0	S16
T4	05:30	S12	S0
T6	06:00	S16	S0

Tabela 5.23: Instante de partida dos trens.

A Tabela 5.24 mostra os resultados de tempo de paradas para cruzamento e ultrapassagem para a programação da Tabela 5.23 considerando também os cruzamentos que ocorreram nos instantes anteriores à perturbação do deslocamento do trem T4. Observe na Figura 5.10 que sem a perturbação o trem T4 chegaria no segmento  $s_{12}$  às 16:20.

Algoritmo	PLIM	AP <sub>PLIM</sub>	APV <sub>PLIM</sub>	AP <sub>PAT</sub>	APV <sub>PAT</sub>	AP <sub>PATF</sub>	APV <sub>PATF</sub>
Tempo de Parada (min)	135.17	135.17	135.17	135.17	135.17	135.17	135.17

Tabela 5.24: Tempo de parada em minutos para cruzamento e ultrapassagem

Com as novas referências de percurso todos os algoritmos conseguiram igualar a nova solução ótima. A Figura 5.14 mostra a solução obtida. O fato de que com os novos instantes de partida e com as novas referências os algoritmos conseguiram obter melhores soluções que na Tabela 5.22 indica que as referências utilizadas devem também de alguma maneira se adaptar às perturbações que possam ocorrer na circulação dos trens. Para que se chega à conclusões definitivas, uma maior quantidade de experimentos com instâncias diferentes devem ser feitos.

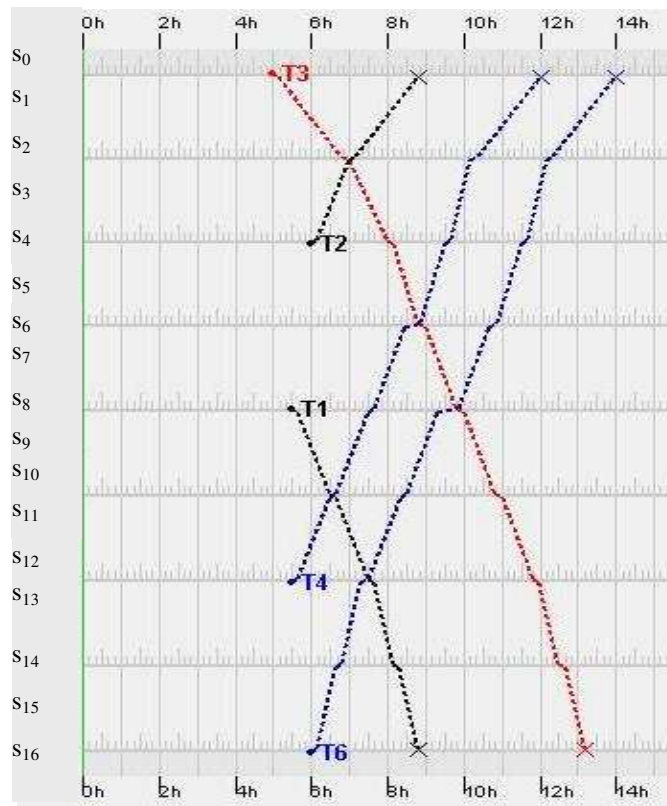


Figura 5.14: Solução ótima considerando perturbações na circulação dos trens.

### 5.2.5 Resultado com heurística do menor tempo de cruzamento

Foi visto nas seções anteriores que quando o algoritmo de decisão nebuloso utiliza uma referência não informativa, como o PAT de cada trem, as soluções obtidas têm, em geral, tempo de parada para cruzamento e ultrapassagem bastante superiores às soluções ótimas. Na seção 5.3.3 foi visto que a flexibilização do PAT pode ser uma alternativa para melhorar as soluções nos casos de não termos uma referência factível. Outra solução seria decidir a preferência entre trens competindo pela ocupação de um mesmo segmento em um mesmo instante de tempo baseado no tempo de paradas para cada possibilidade de cruzamento ou ultrapassagem. Com isso, o trem preferencial seria o trem que causasse um cruzamento com menor tempo de paradas. Essa heurística de decisão é “gulosa”, ou seja, analisa o que é melhor para cada cruzamento, sem considerar a consequência desse cruzamento para os outros trens circulando.

Para o experimento da heurística do menor tempo de cruzamento serão utilizadas duas instâncias do modelo ferroviário 2 com cinco trens circulando. Os trens T1 e T3 tem origem no segmento  $s_0$  e como destino o segmento  $s_{16}$  e os trens T2, T4 e T6 tem origem no segmento  $s_{16}$  e como destino o segmento  $s_0$ . A Tabela 5.25 mostra os instantes de partida de cada trem para o Experimento 1. As referências de percurso são obtidas pelo modelo de programação linear inteira mista de Valle *et al.* (2005) e pelo PAT de cada trem.

Trem.	Instante de partida
T1	00:00
T2	00:00
T3	03:00
T4	03:00
T6	06:00

Tabela 5.25: Instante de partida dos trens para o Experimento 1.

Com o objetivo de avaliar apenas o impacto das decisões de cruzamento na qualidade da solução, foram utilizados apenas algoritmos que assumem velocidade constante por segmento. Os algoritmos utilizados são os seguintes:

1. Modelo de Programação Linear Inteira-Mista, PLIM, proposto por Valle *et al.*, 2005;
2. Algoritmo de Decisão de Preferência utilizando o resultado do PLIM como referência,  $AP_{PLIM}$ ;
3. Algoritmo de Decisão de Preferência utilizando o Programa de Atividades de Trem como referência,  $AP_{PAT}$ ;
4. Algoritmo de Decisão de Preferência utilizando o Programa de Atividades de Trem Flexibilizado, com base nos horário de chegada fornecidos pelo PLIM, como referência,  $AP_{PATF}$ ;
5. Algoritmo de Decisão de Preferência com heurística do menor tempo de cruzamento, AHMT.

A Tabela 5.26 mostra os resultados de tempos de parada em minutos para cruzamento e ultrapassagem obtidos.

Algoritmo	PLIM	$AP_{PLIM}$	$AP_{PAT}$	$AP_{PATF}$	AHMT
Tempo de Parada (min)	185,17	185,17	275,13	185,17	185,17

Tabela 5.26: Tempo de parada em minutos para cruzamento e ultrapassagem.



Analisando a Tabela 5.26 pode-se ver que o  $AP_{PLIM}$  manteve sua referência como nas seções anteriores. O  $AP_{PAT}$  obteve tempo de paradas 48% acima da solução ótima. O algoritmo de decisão utilizando o PAT flexibilizado, com base nos horários de chegada fornecidos pelo PLIM, obteve a solução ótima e o algoritmo de decisão utilizando a heurística do menor tempo de cruzamento também obteve a solução ótima para este caso.

A Figura 5.15 mostra a solução obtida pelo PLIM e pelo  $AP_{PAT}$ . Pode-se ver na solução do  $AP_{PAT}$  que o trem T2 pára no segmento  $s_8$  para aguardar a passagem do trem T1. Essa parada do trem T2 fez com que ele tivesse preferência no cruzamento com o T3. O trem T3 parou em  $s_2$  para o T2 e com isso teve preferência no cruzamento com T4 e T6. As decisões de preferência do  $AP_{PAT}$  não levam em consideração os tempos de paradas dos trens, mas apenas tenta mantê-los próximos de suas referência, o PAT, isso pode levar ao  $AP_{PAT}$  a ter soluções com tempos de paradas consideravelmente superiores às soluções ótimas.

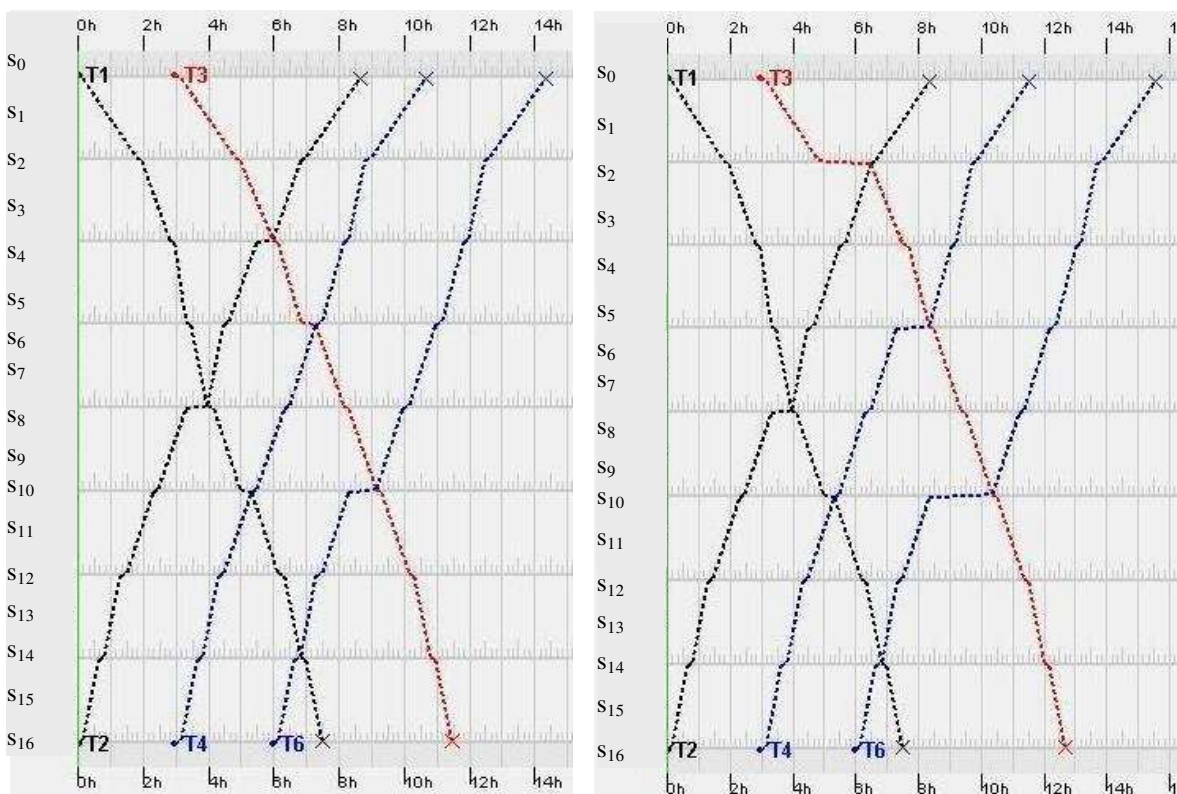


Figura 5.15 Solução do PLIM (esquerda) e solução do  $AP_{PAT}$  (direita).



A Tabela 5.27 mostra os instantes de partida de cada trem para o Experimento 2.

Trem.	Instante de partida
T1	00:00
T2	00:00
T3	01:50
T4	01:50
T6	03:30

Tabela 5.27: Instante de partida dos trens para o Experimento 2.

Os algoritmos utilizados são os mesmo utilizados no Experimento 1 acima. A Tabela 5.28 mostra os resultados de tempos de parada em minutos para cruzamento e ultrapassagem obtidos.

Algoritmo	PLIM	AP <sub>PLIM</sub>	AP <sub>PAT</sub>	AP <sub>PATF</sub>	AHMT
Tempo de Parada (min)	165,17	165,17	358,23	165,17	358,23

Tabela 5.28: Tempo de parada em minutos para cruzamento e ultrapassagem.

Para o Experimento 2 a heurística do menor tempo de cruzamento não obteve a solução ótima. Isso indica que esse algoritmo não garante a otimalidade e que pode levar a resultados com qualidade inferior à ótima. Para este caso a solução foi mais de 100% superior à ótima. A Figura 5.16 mostra a solução obtida pelo AHMT e a solução ótima.

Pode-ver na Figura 5.16 que primeira decisão de cruzamento, entre os trens T1 e T2, foi diferente entre a solução ótima e a do AHMT. Essa diferença na primeira decisão levou à solução final do AHMT ser bastante diferente da ótima. O AHMT decide a preferência entre trens através da escolha do trem que irá causar um cruzamento com menor tempo de parada. Essa decisão é feita analisando apenas os trens concorrendo pelo uso do mesmo segmento no mesmo instante de tempo, sem considerar os outros trens e os cruzamentos futuros. A Figura 5.16 mostra que essa estratégia de decisão gulosa pode levar a solução final a resultados com tempo de parada significativamente superior à solução ótima. Isso prova que a decisão gulosa local não é uma alternativa eficaz para todos os casos.

## 5.4 Resultado para o caso real

A seção 5.2 mostrou os resultados obtidos pelos algoritmos para instâncias pequenas. O objetivo foi o de analisar os valores da função objetivo, isto é, o tempo total de paradas para cruzamento e ultrapassagens em relação às soluções ótimas. Com esse estudo foi possível verificar o comportamento dos algoritmos.

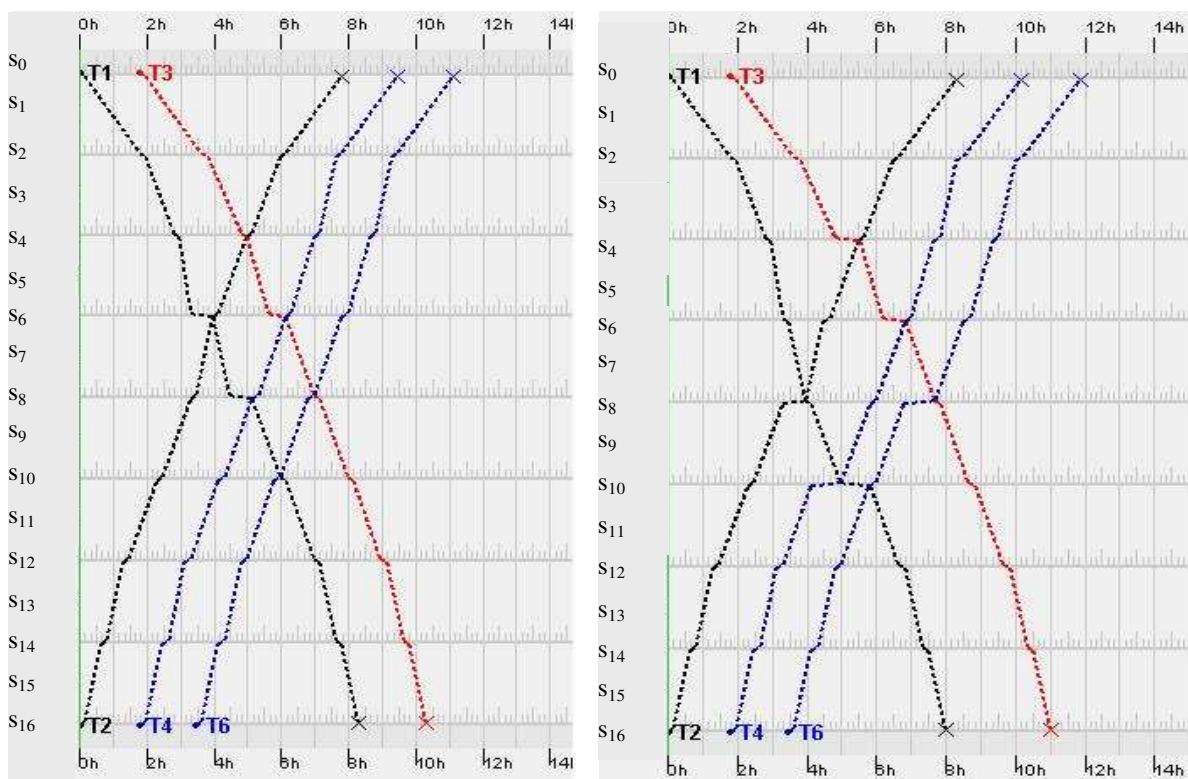


Figura 5.16 Solução do PLIM (esquerda) e solução do AHMT (direita).

No entanto as instâncias da seção anterior não representam situações reais onde, além da qualidade, o tempo de processamento é um fator limitante. Em geral, não se pode concluir que o comportamento dos algoritmos para resolver os casos pequenos é o mesmo em casos reais de maior porte. O número de soluções e possibilidades é significativamente maior e os algoritmos podem seguir caminhos bastante diferentes.

O problema é que, para casos reais, não se tem as soluções ótimas exatas. Foi visto que, mesmo para um problema simples com apenas 7 trens, 9 pátios de cruzamento e 8 segmentos de linha singela, o PLIM levou cerca de 3 horas e meia para fornecer a solução.

O objetivo desta seção é, portanto, analisar a relação entre tempo de processamento e qualidade da solução pelos algoritmos fornecidos neste trabalho, exceto o PLIM.

Os algoritmos utilizados serão os seguintes:

1. Algoritmo Genético, GA, proposto em Mendes (2005);
2. Algoritmo de Decisão de Prioridade utilizando o resultado do GA como referência,  $AP_{GA}$ ;
3. Algoritmo de Decisão de Prioridade utilizando o Programa de Atividades de Trem como referência,  $AP_{PAT}$ ;
4. Algoritmo de Decisão de Prioridade e Velocidade utilizando o Programa de Atividades de Trem como referência,  $APV_{PAT}$ ;
5. Algoritmo de Busca Em Árvore proposto neste trabalho, BA;
6. Algoritmo de Busca Em Árvore proposto em Cherniavsky, 1972, BAC.

Os parâmetros utilizados em cada algoritmo são:

**Algoritmo Genético, GA:**

- tamanho da população: 30 indivíduos;
- número de gerações: 30 gerações;
- probabilidade de mutação: 10%;
- probabilidade de *crossover* : 90%.

**Algoritmo de Decisão de Prioridade e Velocidade:**

- limites de velocidade:
  - velocidade máxima =  $1.5 * (\text{velocidade média})$ ;
  - velocidade mínima =  $0.5 * (\text{velocidade média})$ ;

**Algoritmo de Busca Em Árvore:**

- profundidade de busca: 1;
- horizonte de tempo para simular o custo de um nó: 0 a 20 horas;

**Algoritmo de Busca Em Árvore proposto em Cherniavsky, 1972:**

- profundidade de busca: 2.

Será considerada uma instância com dados reais de uma ferrovia brasileira. O modelo ferroviário tem 25 segmentos tipo pátios de cruzamento e 24 segmentos de linha singela e 35 trens. Foi considerado um horizonte de 36 horas de planejamento.

Os horários de partida dos trens são os horários de partida típicos dos trens no dia que a instância foi criada.

A Tabela 5.29 mostra os tempos de parada para cruzamento e ultrapassagem e a Tabela 5.30 mostra os tempos de processamento.

Algoritmo	GA	APGA	APPAT	APVPAT	BA	BAC
Tempo de parada (min)	3798	3798	6218	4802	3365	3516

Tabela 5.29: Tempo de parada para cruzamento e ultrapassagem (minutos).

Algoritmo	GA	APGA	APPAT	APVPAT	BA	BAC
Tempo de processamento (ms)	720000	453	453	547	42515	220953

Tabela 5.30: Tempo de processamento (milissegundos).

Pode-se ver na Tabela 5.29 que os algoritmos de decisão de preferência e velocidade necessitam de um tempo de processamento muito baixo, em torno de 500 milissegundos. Isso mostra que esses algoritmos são apropriados para aplicações em tempo real, nas quais o tempo de resposta é uma característica fundamental devido ao dinamismo da operação ferroviária. O algoritmo  $AP_{PAT}$ , que não utiliza referência factível, obteve tempo de parada muito superior aos dos outros algoritmos. Já o  $APV_{PAT}$ , que decide a velocidade dos trens, conseguiu resultados melhores para a mesma referência. Utilizando uma referência factível o algoritmo de preferência,  $AP_{GA}$ , obteve solução idêntica a sua referência: a solução do GA. O algoritmo BA obteve o melhor resultado e também um tempo de processamento que, apesar de ser maior que os dos algoritmos de preferência, ainda é adequado para aplicações em tempo-real, visto que consegue uma boa solução em um período menor que 1 minuto. A Figura 5.17 mostra os resultados de tempos de parada para cruzamento e ultrapassagem.

O resultado do o algoritmo de busca em árvore, mostrado na Tabela 5.25, é o melhor resultado, obtido com um horizonte de simulação de 8 horas na estimativa de custo de um nó. Foi feito um estudo para analisar o efeito da variação do horizonte de simulação

na estimativa do custo de um nó no tempo de resposta e qualidade da solução do algoritmo de busca. Foram feitas 10 simulações utilizando diferentes horizontes de tempo. A Tabela 5.31 mostra os tempos de parada para cruzamento e ultrapassagem e tempos de processamento obtidos.

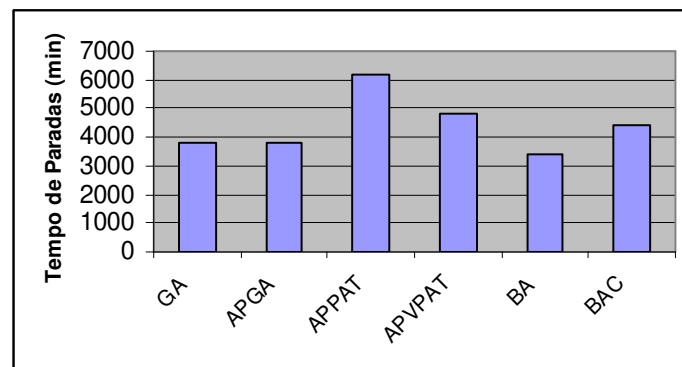


Figura 5.17: Tempo de parada para cruzamento e ultrapassagem.

Os dados da Tabela 5.31 estão representados graficamente nas Figuras 5.18 e 5.19. Pode-se ver na Figura 5.18 que o aumento do horizonte de tempo da busca leva inicialmente para a obtenção de melhores soluções. O aumento do horizonte de planejamento significa aumentar a “visão” da estimativa, fazendo com que o algoritmo enxergue mais longe a consequência de cada cruzamento. Entretanto, a partir de certo valor o aumento do horizonte de planejamento não leva a melhores soluções. Isso acontece porque o efeito de um cruzamento sobre os outros cruzamentos é reduzido depois de algumas horas que ele ocorre. Com isso, o uso de um horizonte muito grande não traz vantagens para o algoritmo e aumenta o tempo de processamento. Como era esperado, o aumento do limitante leva a um contínuo aumento no tempo de processamento, como pode ser confirmado pela Figura 5.19. A solução obtida com o horizonte de 8 horas pode ser visualizado na Figura 5.20.

Horizonte (horas)	Tempo de Parada (minutos)	Tempo de Processamento (milisegundos)
0	4435	26340
2	3614	30250
4	3372	35110
6	3478	38093
8	3365	42515
10	3516	43921
12	3472	44594
14	3516	46859
16	3516	48016
18	3516	48078

Tabela 5.31: Tempo de parada para cruzamento e ultrapassagem e tempo de processamento para o algoritmo BA.

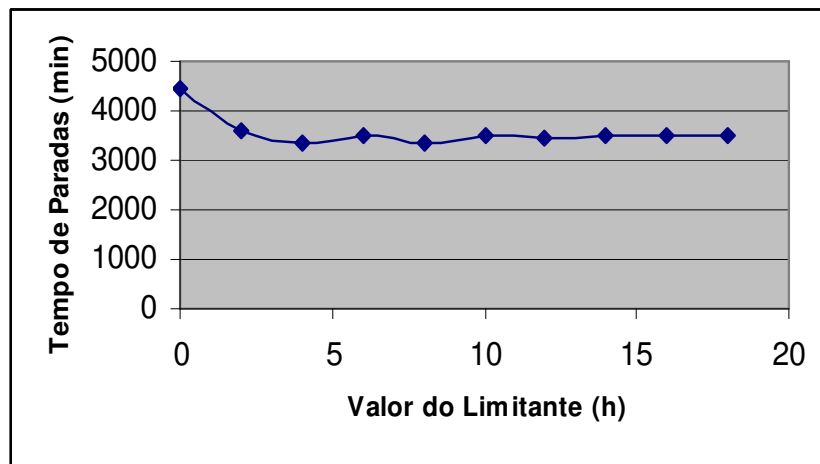


Figura 5.18: Tempo de parada para cruzamento e ultrapassagem para o algoritmo BA.

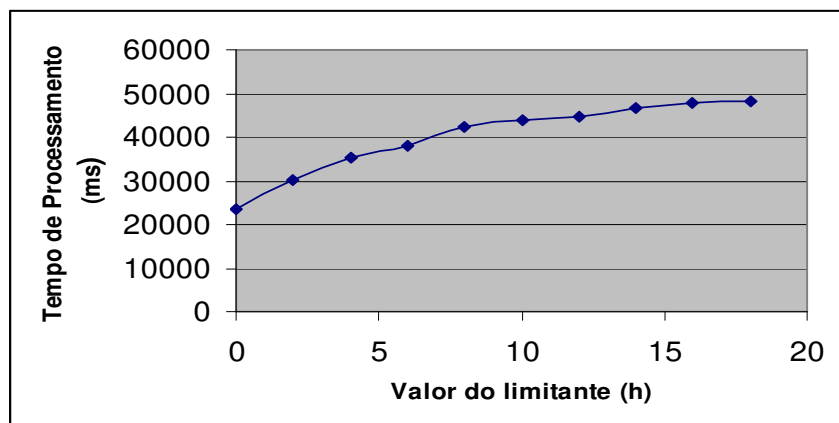


Figura 5.19: Tempo de processamento para o algoritmo BA.

#### 5.4 Análise dos Resultados.

Este capítulo apresenta estudo do comportamento dos algoritmos para dois tipos de instâncias: instâncias pequenas para as quais se conhece a solução ótima, mas que não são significativas para situações reais; instâncias grandes obtidas com dados reais de ferrovias brasileiras.

Os experimentos com instâncias pequenas mostram aspectos importantes dos algoritmos. Os algoritmos de decisão de preferência com velocidade constante e variável que utilizam uma referência factível conseguem seguir suas referências com sucesso. Como não há distúrbios no percurso dos trens, estes sempre obtêm os mesmos partida de saída e tempos de percurso que suas referências e com isso para decidir a preferência entre os trens basta verificar qual a prioridade utilizada na referência e utilizar esta mesma decisão.

Quando os algoritmos de preferência e velocidade utilizam referências não factíveis, como o Programa de Atividades de Trem, estes já não conseguem obter soluções ótimas para a maioria dos casos, sendo que algumas soluções têm tempos de parada significativamente maiores que o ótimo. É importante ressaltar que a comparação do algoritmo de preferência, utilizando o PAT como referência, com a solução ótima é desfavorável ao algoritmo de preferência, pois este tenta seguir para todos os trens uma referência infactível, impossível de ser cumprida, sem dispor de qualquer outra informação sobre o problema.

O algoritmo que tem a possibilidade de variar a velocidade dos trens consegue resultados melhores que sua versão com velocidade constante para a maioria dos casos. Em algumas situações o algoritmo com velocidade variável obteve soluções melhores que as ótimas, o que só é possível porque o modelo de programação inteira mista supõe velocidades constante por segmento. Pode-se concluir disso que a variação da velocidade dos trens é uma estratégia interessante para se obter soluções de melhor qualidade, caso consumo de combustível e outras considerações (e.g. capacidade dos terminais, trechos da linha) não tornem esta estratégia proibitiva.

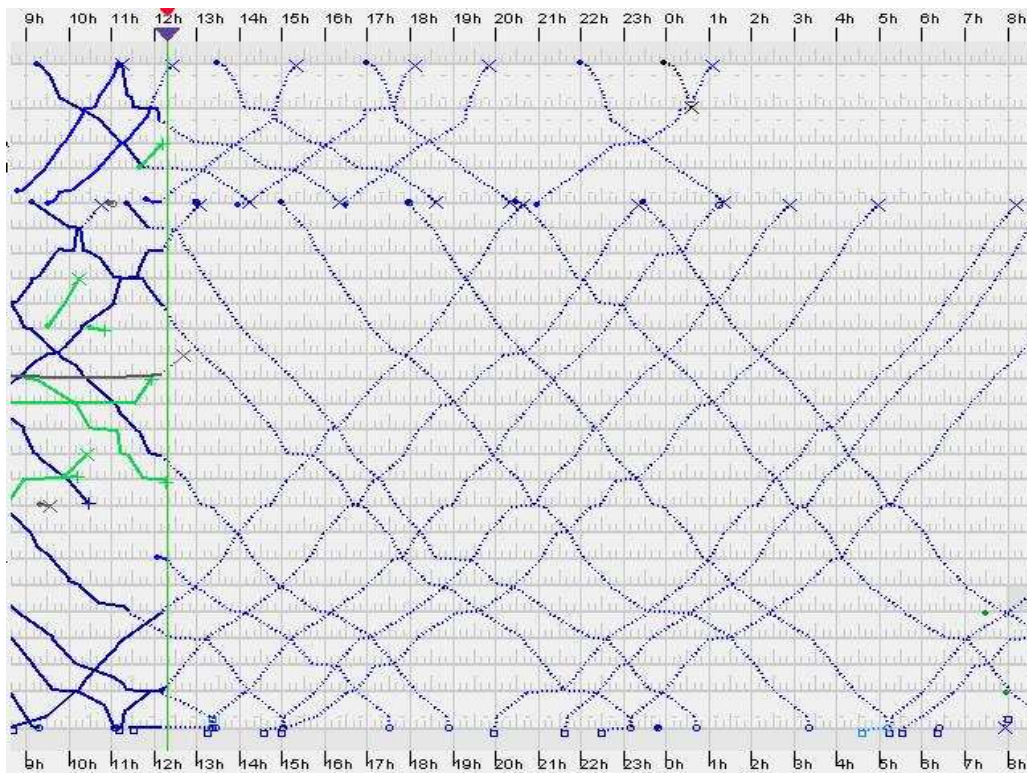


Figura 5.20: Solução produzida pelo algoritmo BA.

Os algoritmos de busca não utilizam referências e por isso é uma ferramenta importante. O algoritmo de busca em árvore BA consegue, para muitos casos, obter soluções ótimas e se aproxima muito destas para os outros casos. O algoritmo BAC utiliza uma estratégia diferente do BA. O BAC expande vários nós até uma profundidade máxima definida, que para os testes pequenos foi definida como 3. A partir dos nós de profundidade máxima uma estimativa é feita para cada nó e é escolhido no nó inicial o trem que levou ao



nó com menor estimativa. O fato de vários nós serem expandidos dá ao BAC uma visão maior que o BA, fazendo com que em alguns casos aquele tenha melhores soluções que este. Entretanto, a estimativa feita pelo BA é mais informativa que a feita pelo BAC, pois considera um planejamento factível para a escolha do próximo nó. Isso faz com que o BA obtenha soluções de qualidade com um menor tempo de processamento que o BAC.

Como os casos testados na seção 5.3.1 não são significativos, os tempos de processamento dos algoritmos de preferência e de busca são bastante baixos e não podem ser utilizados como padrão dos algoritmos. Pode-se ver que o crescimento do tempo de processamento do PLIM é exponencial com o crescimento da complexidade da instância. O GA parte inicialmente com tempos maiores de processamento que o PLIM, mas tem um crescimento linear no tempo de processamento e para instâncias com 6 e 7 trens seus tempos de processamento são muito inferiores que o PLIM.

A Tabela 5.32 mostra para as 30 instâncias pequenas (seções 5.2.1 e 5.2.2) o *GAP* de otimalidade.

$$GAP = \left( \frac{\text{soluçãoAlgoritmo} - \text{soluçãoÓtima}}{\text{soluçãoÓtima}} \right) * 100 \quad (5.1)$$

Segundo a Tabela 5.323 o algoritmo que se manteve mais próxima da solução ótima foi o GA. O algoritmo BA manteve-se em média 5,3% acima do ótimo, tendo no pior caso um resultado 43% acima. Esses resultados com instâncias pequenas mostram que o BA é bastante promissor, pois consegue um bom equilíbrio entre tempo de processamento e qualidade da solução. O BAC obtém resultados um pouco acima do BA, com média de 8,5%. Como dito anteriormente o AP<sub>PAT</sub> obteve a pior média com 42% acima do ótimo.

A seção 5.2.3 mostrou que a flexibilização do PAT pode ser uma alternativa promissora para os casos em que uma referência factível não está disponível, o que ocorre principalmente em aplicações reais. Utilizando como referência o PAT flexibilizado, o algoritmo de decisão de preferência conseguiu obter a solução ótima, o que não foi possível utilizando o PAT como referência. A seção 5.2.5 apresentou outra alternativa para os casos nos quais uma referência factível não está disponível, que é a heurística gulosa do menor tempo de cruzamento. Esta seção mostrou que essa heurística pode obter bons resultados para alguns casos, mas que em outros casos pode levar a soluções com qualidade significativamente inferior às ótimas.

Foi feito na seção 5.2.4 um estudo com perturbações no deslocamento dos trens. Esse estudo indicou que as referências utilizadas pelo algoritmo de decisão de preferência devem considerar as perturbações ocorridas. Ao tentar manter as referências de percurso antigas o algoritmo de decisão obteve solução com maior tempo de parada que a solução obtida com a referência que considerou a perturbação ocorrida.

Para o estudo com a instância obtida a partir de dados reais não foi possível utilizar o PLIM e apenas os outros algoritmos foram utilizados.

Os resultados para a instância real foram análogos aos resultados para instâncias pequenas. O algoritmo  $AP_{GA}$  consegue manter sua referência com sucesso. Já o  $AP_{PAT}$  obtém a solução com maior tempo de parada para cruzamento e ultrapassagem e o  $APV_{PAT}$  consegue reduzir em cerca de 77% este resultado. Os algoritmos de busca conseguem obter boas soluções respeitando as restrições de tempo de processamento impostas por aplicações em tempo real. O GA consegue também obter soluções de boa qualidade, mas com um tempo de processamento que não é apropriado para aplicações nas quais se espera uma solução em poucos segundos.

	GAP otimalidade (%)	
	Média	Pior Caso
$AP_{PLMI}$	0	0
GA	3,5	40
$AP_{GA}$	3,5	40
$AP_{PAT}$	42	243
$APV_{PAT}$	23	243
$APV_{PLMI}$	0	0
BA	5,3	43
BAC	8,5	67

Tabela 5.32: GAP de otimalidade para os diferentes algoritmos.

## **5.5 Resumo.**

Neste capítulo foi feito um estudo comparativo entre os algoritmos de decisão de preferência e velocidade e o algoritmo de busca em árvore apresentado neste trabalho com outras três diferentes abordagens: Programação Linear Inteira Mista, Algoritmo Genético e o algoritmo de Busca em Árvore proposto por Cherniavsky, 1972.

Foram considerados dois tipos de instâncias: instâncias pequenas e uma instância complexa obtida com dados reais de uma grande ferrovia brasileira.

Os resultados para as instâncias pequenas e grandes mostram que os algoritmos de decisão de prioridade e velocidade cumprem seu objetivo quando utilizam referências factíveis. Para o caso que não existe uma referência factível o algoritmo de decisão de velocidade se mostrou bastante apropriado na obtenção de soluções de melhor qualidade. A grande vantagem destes algoritmos é que conseguem soluções, mesmo para o caso complexo, em tempos menores que 1 segundo.

Os algoritmos de busca, principalmente o proposto neste trabalho, conseguem soluções de boa qualidade em um tempo de processamento menor que 1 minuto.

O próximo capítulo conclui esta dissertação e sugere trabalhos futuros.

## Capítulo 6

### Conclusão.

O planejamento de circulação de trens é uma das mais importantes e também uma das mais complexas etapas no processo operacional de uma ferrovia. Isso se deve ao dinamismo inerente ao problema: situações imprevistas ocorrem a todo instante, existem conflitos entre a necessidade de cumprir as demandas dos diferentes pontos de uma ferrovia, entre as escalas de maquinistas, entre as manutenções da linha, etc.

Cabe aos despachadores e controladores de trens a função de tomarem decisões de planejamento de circulação para cumprir da melhor maneira possível todas as restrições da operação. Isso envolve a priorização de alguns objetivos em detrimento de outros.

Ferramentas computacionais de apoio ao processo de despacho são escassas no Brasil e no mundo. As poucas existentes partem da idéia de fornecer uma solução ótima para uma determinada função-objetivo. A prática operacional já mostrou que ferramentas de apoio à tomada de decisão em tempo-real são mais adequadas à complexidade do problema. Porém, existe um debate entre tempo de processamento das ferramentas e qualidade da solução obtida.

Este trabalho teve como objetivo o desenvolvimento de algoritmos para o planejamento de circulação de trens em tempo real, com o apoio de ferramentas da Inteligência Computacional.

O primeiro algoritmo objetiva decidir a preferência entre trens concorrendo pelo uso de um segmento e manter uma referência de percurso fornecida por algum especialista ou por algum algoritmo de otimização global. Devido a perturbações na circulação dos trens ou a erros de decisão, muitos trens acabam não cumprindo suas referências. Para isso foi desenvolvido um segundo algoritmo de decisão de preferência que também determina a velocidade entre trens. A variação da velocidade dos trens permite uma maior flexibilidade ao planejamento, pois dá a possibilidade de trens atrasados ou adiantados se aproximarem de suas referências.

Uma das restrições dos algoritmos de decisão de preferência reside no fato de que a decisão é tomada localmente, baseando-se apenas nos trens em conflito. Com base nesta

restrição foi desenvolvido um algoritmo de busca em árvore do tipo *Best-First* que permite que cada decisão seja tomada com base em uma análise global de toda a circulação.

Este trabalho também incluiu um estudo comparativo entre os algoritmos propostos neste trabalho e outras três importantes algoritmos:

1. Modelo de Programação Linear Inteira-Mista;
2. Algoritmo Genético com agrupamento;
3. Algoritmo de busca em árvore proposto por Cherniavsky, 1972.

O estudo foi feito sobre duas classes de instâncias: as pequenas, que servem como ilustração dos algoritmos e verificação da otimalidade e uma instância retirada do principal trecho de uma das principais ferrovias do Brasil.

Os resultados mostram que o modelo matemático não consegue fornecer soluções ótimas exatas para a instância real, ou seja, não consegue fornecer a solução em um tempo adequado. O algoritmo genético obtém soluções de boa qualidade para as duas classes de instâncias, mas também não é consistente com as restrições de tempo de processamento para aplicação em tempo real. Os algoritmos de decisão de preferência conseguem, com êxito, manter suas referências quando não há perturbações no deslocamento dos trens. Porém, para o caso que a referência fornecida não é factível, o algoritmo de decisão de preferência produz soluções de baixa qualidade. O que se explica pelo fato do algoritmo tentar manter uma referência impossível de ser cumprida pelos trens, fazendo com que a comparação de seus resultados com a solução ótima seja bastante desfavorável ao algoritmo de preferência, como foi visto na seção 5.4. Nestes casos, a versão do algoritmo que possibilita a variação de velocidade obtém resultados significativamente melhores. O tempo de processamento dos algoritmos de decisão de prioridade é menor que 500ms para todas as instâncias consideradas neste trabalho. Foi visto também que a flexibilização do PAT e a heurística gulosa de decisão podem ser estratégias que possibilitem a obtenção de boas soluções quando uma referência factível não está disponível.

O algoritmo de busca aqui sugerido consegue um compromisso aceitável entre tempo de processamento e qualidade da solução. Conseguindo soluções muito próximas às ótimas para as instâncias pequenas e para a instância real, forneceu a melhor solução dentre os outros algoritmos, levando para isso um tempo menor que 1 minuto.

A proposta de busca em árvore de Cherniavsky, 1972, expande um número maior de nós na escolha do próximo nó a ser expandido, o que permite soluções de boa qualidade, mas com um tempo de processamento superior ao algoritmo aqui proposto.

A análise dos algoritmos sugere algumas melhorias, além de novos algoritmos que podem ser desenvolvidos a partir deste trabalho:

- Melhoria do algoritmo de decisão de preferência que permita soluções de melhor qualidade na ausência de uma referência factível;
- Fazer um número maior de testes com diferentes instâncias para verificar a eficácia do uso do PAT flexibilizado como referência;
- Desenvolvimento de uma função de estimativa analítica para o algoritmo de busca que permita soluções de melhor qualidade ou que seja otimista e garanta otimalidade ao algoritmo;
- Como o algoritmo de busca é do tipo *Best-First*, as escolhas feitas durante a busca podem levar a mínimos locais. A aplicação de operadores de busca local ou mesmo o desenvolvimento de uma meta-heurística, como busca-tabu, pode ser uma linha de pesquisa promissora.

Atualmente, uma ferramenta computacional de apoio à tomada de decisão desenvolvida com uso dos algoritmos e princípios apresentados neste trabalho se encontra em funcionamento em duas ferrovias brasileiras: ALL (América Latina Logística) e CFN (Companhia Ferroviária do Nordeste). Na principal linha da ALL o sistema conseguiu reduzir em 17% o tempo total de parada para cruzamento e ultrapassagem.

## Referências

- CASSANDRAS, C G., LAFORTUNE, S. *Introduction to Discrete Event System*. Kluwer Academic Publishers, Boston,USA: 1999.
- CHIANG, H., CHIANG, T., HAU, H., HSIEH, C. e KO, S. Knowledge-based system for railway scheduling. *Data & Knowledge Engineering*, vol.27, pp. 289-312: 1998.
- CHERNIAVSKY, A.L. A program for timetable compilation by a look-ahead method. *Artificial Intelligence*, vol. 3, pp. 61-67: 1972.
- CONWAY *et al.* *Theory of Scheduling*. Addison-Wesley, London, UK, 1967.
- DAVIS, L. Job-shop scheduling with genetic algorithms. *Proceedings of 1st International Conference on Genetic Algorithms*, Pittsburgh, PA, USA, pp.136–140: 1985.
- GAREY, M. R. e JOHNSON, D.S. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W.H. Freeman and Company, New York: 1979.
- GLOVER, F. e McMILLAN, C. The general employee scheduling problem: An integration of MS and AI. *Computers and Operational Research* Vol. 13, No.5, pp. 563-573: 1986.
- GOLDBERG, D. E. *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison-Wesley, Menlo Park, California, USA: 1989.
- GOMIDE, F., BESSA, E., VIEIRA, P. e NETO, L. Railway Dispatch Planning and Control. In: *Proc. of NAFIPS99*, New York, USA, pp. 134-138: 1999.
- HIGGINS, A., KOZAN, E. e FERREIRA, L. Optimal scheduling of trains on a single line track. *Transportation. Research*, 30B, pp.147-161: 1996.
- HIGGINS, A., KOZAN, E. e FERREIRA, L. Heuristics techniques for single line train scheduling. *Journal of Heuristics*, 3(1), pp. 43–62: 1997.
- HOMBERGER, J. e GEHRING, H. A two-phase hybrid metaheuristic for the vehicle routing problem with time windows. *European Journal of Operational Research*, vol.162, pp.220-238: 2005.
- ISSAI, M. T. e CASSAIGNE. P. Predictive and reactive approaches to the train scheduling problem: A knowledge management perspective. *IEEE Transactions on System, Man and Cybernetics*, 31(4),pp.:476-484: 2001.

- ISSAI, M. T. e SINGH, G. M. Hybrid Applications of Constraint Satisfaction and Meta-Heuristics to Railway Timetabling: A Comparative Study. *IEEE Transactions on System, Man, and Cybernetics*: v.31, n.1, pp. 87–95, 2001
- JENNINGS, N. R. e BUSSMANN, S. Agent-based control systems - why are they suited to engineering complex systems? *IEEE Control Systems Magazine*, vol.23, pp.61-73: 2003.
- JOVANOVIĆ D. *Improving railroad on-time performance: models, algorithms and applications*. PhD thesis, University of Pennsylvania: 1989.
- MENDES, F. O. M. F. *Aplicação de Modelos de Estimação de Fitness em Algoritmos Genéticos*. Tese de Mestrado. Universidade Estadual de Campinas, 2005.
- MENDES, F., GONÇALVES, R. e GOMIDE, F. Genetic Algorithms, fuzzy clustering and discrete event systems: An application in scheduling. *Proceedings of the I Workshop on Genetic Fuzzy Systems*, Granada, Spain, pp. 83-88: 2005.
- PEDRYCZ, W., GOMIDE, F. *An Introduction to Fuzzy Sets: Analysis and Design*, MIT Press, Cambridge, Massachusetts, USA: 1998.
- PASSINO, K. M. e YURKOVICH, S. *Fuzzy Control*. Addison-Wesley, Menlo Park, California, USA: 1998.
- PEARL, J., *Heuristics: Intelligent Search Strategies for Computer Problem Solving*. Addison-Wesley Publishing Company, Los Angeles: 1984.
- PETERSEN E. e TAYLOR A. Line block prevention in rail line simulation and optimization. *Transportation Science* 16(2), pp.192-205: 1982.
- REEVES, C., RAYWARD, V., OSMAN, I. e SMITH, G. *Modern Heuristic Search Methods*. John Wiley and Sons, New York, USA: 1996.
- RARDIN, R. e L., UZSOY, R. Experimental evaluation of heuristic optimization Algorithms: A tutorial. *Journal of Heuristics*, vol.7, pp. 261–304: 2001.
- RONDÓN, M. *Modelagem estruturada e sistemas inteligentes: uma aplicação a sistemas de transporte ferroviário*. Tese de Mestrado, FEEC, Universidade Estadual de Campinas: 2000.
- RUSSEL, S. e NORVIG, P. *Artificial Intelligence: A Modern Approach*. Prentice Hall, New Jersey, USA, 2ed: 2003.



- SALIM V. e CAI, X. Cai. Scheduling cargo trains using genetic algorithms. *IEEE International Conference on Evolutionary Computation*, Perth, WA, Australia, vol.1, pp. 224-227: 1995.
- SEN, A. K., BAGCHI, A. e RAMASWAMY, R. Searching Graphs with A\*: Applications to Job Sequencing. *IEEE Transactions on System, Man and Cybernetics – Part A: System and Humans*, vol.26, no.1, pp.168-174: 1996.
- SZIPEL, B. Optimal train scheduling on single track railway. *Operations Research*, vol.20, pp.343-352: 1972.
- SAUDER, R. L. e WETERMAN, W. Computer-aided train dispatching: Decision support through optimization. *Interfaces*, vol.13, pp.24–37: 1983.
- TAZONIERO, A., GONÇALVES, R e GOMIDE, F. Fuzzy algorithm for real-time train dispatch and control. *Proceedings of NAFIPS 2005*, Ann Arbor, Michigan, USA, pp.332-336: 2005.
- TAZONIERO, A., GONÇALVES, R e GOMIDE, F. Decision Making Strategies for Real-Time Train Dispatch and Control in PEDRYCS, W. *et al.*, *Analysis and Design of Intelligent Systems Using Soft Computing Techniques*, Springer-Verlag, Berlin, Alemanha, pp.195-204:2007.
- VIEIRA P. e GOMIDE, F. Computer-aided train dispatch. *IEEE Spectrum*, 33(7), pp.51–52: 1996.
- VALLE, A., TAZONIERO, A., MENDES, F. Intelligent optimization in train circulation planning and control. *International Heavy Haul Association Congress*, Best Student Paper, Rio de Janeiro: 2005a.
- VALLE, A., GONÇALVEZ, R e GOMIDE, F. Fuzzy optimization model for train dispatch systems. *Proc. of the 11th International Fuzzy Systems Association World Congress*, Beijing, China, vol.3, pp.1788–1793, July 2005.
- WINSTON, W. L. *Operations Research: Applications and Algorithms*. Thomson Brooks/Cole. Belmont, Canada. 4 ed: 2004.
- YEN, J. e REZA, L.. *Fuzzy Logic: Intelligence, Control, and Information*. Prentice Hall, New Jersey, USA: 1998.
- YUAN, B. e KLIR, G. *Fuzzy Sets and Fuzzy Logic*. Prentice Hall, New Jersey, USA: 1995.

## Apêndice A

Este apêndice contém os dados dos modelos ferroviários 1 e 2 utilizados na geração dos resultados deste trabalho. As Tabelas A1 e A3 mostram os comprimentos e o número de vias de cada segmento para cada modelo ferroviário. As Tabelas A2 e A4 mostram a velocidade em quilômetros por hora para cada trem em cada segmento dos modelos ferroviários.

### A.1 Modelo Ferroviário 1.

Segmento	Comprimento (km)	Número de vias
s <sub>0</sub>	3	2
s <sub>1</sub>	30	1
s <sub>2</sub>	3	2
s <sub>3</sub>	30	1
s <sub>4</sub>	3	2
s <sub>5</sub>	20	1
s <sub>6</sub>	3	2
s <sub>7</sub>	15	1
s <sub>8</sub>	3	2
s <sub>9</sub>	15	1
s <sub>10</sub>	3	2

Tabela A1. Comprimento e número de vias dos segmentos do Modelo Ferroviário 1.

	Velocidade por trem (km/h)			
Segmento	T3, T5 e T7.	T4 e T6.	T1	T2
s <sub>0</sub>	18	18	18	18
s <sub>1</sub>	18	18	18	18
s <sub>2</sub>	18	18	18	18
s <sub>3</sub>	30	60	36	36
s <sub>4</sub>	18	18	18	18
s <sub>5</sub>	30	30	60	24
s <sub>6</sub>	18	18	18	18
s <sub>7</sub>	18	18	30	30
s <sub>8</sub>	18	18	18	18
s <sub>9</sub>	18	18	18	18
s <sub>10</sub>	18	18	18	18

Tabela A2. Velocidade (km/h) para os trens nos segmentos do Modelo Ferroviário 1.

## A.2 Modelo Ferroviário 2.

Segmento	Comprimento (km)	Número de vias
s <sub>0</sub>	3	2
s <sub>1</sub>	30	1
s <sub>2</sub>	3	2
s <sub>3</sub>	30	1
s <sub>4</sub>	3	2
s <sub>5</sub>	20	1
s <sub>6</sub>	3	2
s <sub>7</sub>	15	1
s <sub>8</sub>	3	2
s <sub>9</sub>	15	1
s <sub>10</sub>	3	2
s <sub>11</sub>	15	1
s <sub>12</sub>	3	2
s <sub>13</sub>	15	1
s <sub>14</sub>	3	2
s <sub>15</sub>	15	1
s <sub>16</sub>	3	3

Tabela A3. Comprimento e número de vias dos segmentos do Modelo Ferroviário 2.

	Velocidade por trem (km/h)			
Segmento	T3, T5 e T7.	T4 e T6.	T1	T2
s <sub>0</sub>	18	18	18	18
s <sub>1</sub>	18	18	18	18
s <sub>2</sub>	18	18	18	18
s <sub>3</sub>	30	60	36	36
s <sub>4</sub>	18	18	18	18
s <sub>5</sub>	30	30	60	24
s <sub>6</sub>	18	18	18	18
s <sub>7</sub>	18	18	30	30
s <sub>8</sub>	18	18	18	18
s <sub>9</sub>	18	18	18	18
s <sub>10</sub>	18	18	18	18
s <sub>11</sub>	18	18	18	18
s <sub>12</sub>	18	18	18	18
s <sub>13</sub>	30	30	30	30
s <sub>14</sub>	18	18	18	18
s <sub>15</sub>	30	30	30	30
s <sub>16</sub>	18	18	18	18

Tabela A4. Velocidade (km/h) para os trens nos segmentos do Modelo Ferroviário 2.

## Apêndice B

### Notações utilizadas nos algoritmos.

Nome	Significado	Sintaxe
Início/fim	Início e fim do algoritmo	<b>Início</b> algoritmo <b>fim</b>
procedimento	Indica o início de um procedimento.	<b>procedimento</b>
Se/ então/ senão	Execução condicional de sentenças	<b>se</b> condição <b>então</b> sentença <b>senão</b> sentença
←	atribuição	a ← b
∅	conjunto ou lista vazia	a ← ∅
≤, ≥, >, <, =, ≠	comparadores	<b>se</b> a < b <b>então</b>
+, -, /, *	operadores matemáticos	2*4 = 8
//	comentário	//isto é um comentário
a, b, j, i, c km, s	variáveis são escritas em itálico	<i>a, b, j, i, c, km, s</i>
selecionaTrem()	nomes de procedimentos	selecionaTrem()

Tabela B.1: Notações utilizadas nos algoritmos