DENISE MARIA VECINO SATO

I-DCOP: CLASSIFICAÇÃO DE TRENS BASEADA EM UM PROCESSO ITERATIVO COM OTIMIZAÇÃO DISTRIBUÍDA DE RESTRIÇÕES

Dissertação de Mestrado apresentada ao Programa de Pós-Graduação em Informática da Pontifícia Universidade Católica do Paraná como requisito parcial para obtenção do título de Mestre em Informática.

CURITIBA

2014

DENISE MARIA VECINO SATO

I-DCOP: CLASSIFICAÇÃO DE TRENS BASEADA EM UM PROCESSO ITERATIVO COM OTIMIZAÇÃO DISTRIBUÍDA DE RESTRIÇÕES

Dissertação de Mestrado apresentada ao Programa de Pós-Graduação em Informática da Pontifícia Universidade Católica do Paraná como requisito parcial para obtenção do título de Mestre em Informática.

Área de Concentração: *Ciência da Computação*

Orientador: Prof. Dr. Edson Emílio Scalabrin

CURITIBA

2014

Dados da Catalogação na Publicação
Pontifícia Universidade Católica do Paraná
Sistema Integrado de Bibliotecas – SIBI/PUCPR
Biblioteca Central

S253i
2014

Sato, Denise Maria Vecino
I-DCOP : classificação de trens baseada em um processo iterativo com otimização distribuída de restrições / Denise Maria Vecino Sato ; orientador, Edson Emílio Scalabrin. – 2014.
xv, 111 f. : il. ; 30 cm

Dissertação (mestrado) – Pontifícia Universidade Católica do Paraná, Curitiba, 2014
Bibliografia: f. [91]-95

1. Informática. 2. Algoritmos. 3. Trens de ferro. 4. Inteligência artificial distribuída. 5. Restrições (Inteligência artificial). I. Scalabrin, Edson Emilio. II. Pontifícia Universidade Católica do Paraná. Programa de Pós-Graduação em Informática. III. Título.

CDD 20. ed. – 004

Pontifícia Universidade Católica do Paraná

ATA DE DEFESA DE DISSERTAÇÃO DE MESTRADO
PROGRAMA DE PÓS-GRADUAÇÃO EM INFORMÁTICA

DEFESA DE DISSERTAÇÃO DE MESTRADO Nº 03/2014

Aos 14 dias do mês de Agosto de 2014 realizou-se a sessão pública de Defesa da Dissertação **"I-DCOP: Classificação de Trens Baseada em um Processo Interativo com Otimização Distribuída de Restrições"** apresentado pela aluna **Denise Maria Vecino Sato,** como requisito parcial para a obtenção do título de Mestre em Informática, perante uma Banca Examinadora composta pelos seguintes membros:

Prof. Dr. Edson Emílio Scalabrin
PUCPR (Orientador) _____ Aprov.
(assinatura) (Aprov/Reprov)

Prof. Dr. Bráulio Coelho Àvila
PUCPR _____ Aprovada
(assinatura) (Aprov/Reprov)

Prof. Dr. Gilson Yukio Sato
UTFPR _____ Aprovada
(assinatura) (Aprov/Reprov)

Conforme as normas regimentais do PPGIa e da PUCPR, o trabalho apresentado foi considerado aprovado (*aprovado/reprovado*), segundo avaliação da maioria dos membros desta Banca Examinadora. Este resultado está condicionado ao cumprimento integral das solicitações da Banca Examinadora registradas no Livro de Defesas do programa.

Prof.ª Dr.ª Andreia Malucelli
Coordenadora do Programa de Pós-Graduação em Informática

Programa de Pós-Graduação em Informática PPGIa PUCPR

Dedico esse trabalho ao meu marido Koji e a
minha filha Lara que são parte de todas
as minhas realizações.

Agradecimentos

Nesta página muito especial deste trabalho, gostaria de agradecer a algumas pessoas, dentre as muitas que me ajudaram a realizá-lo.

Em especial a minha família, sempre presente na minha vida.

Ao meu orientador, prof. Edson Scalabrin, por todo o suporte e apoio e principalmente por sempre confiar no trabalho desenvolvido.

A CAPES, por tornar esse trabalho possível.

A *Swiss Railways* (SBB), representada pelo Sr. Stephan Leber, por ceder os dados do pátio de Lausanne, utilizados nos experimentos dessa pesquisa.

Ao Sr. Peter Márton, meu contato sobre os dados utilizados nos experimentos, por todo auxílio despendido.

Ao prof. Fabrício Enembreck, que sempre esteve disposto a ajudar com diferentes olhares sobre as soluções apresentadas aos problemas.

Ao Sr. Thomas Léauté, um dos criadores do *framework* FRODO, pelas valiosas sugestões para uso da ferramenta e também nas contribuições ao modelo proposto para o problema.

A todos os amigos e colegas que, de alguma forma, participaram dessa caminhada e contribuíram para que esse trabalho fosse realizado.

Resumo

No presente trabalho, o problema de classificação de trens foi modelado como um Problema de Otimização Distribuída de Restrições (DCOP). Essa modelagem, denominada Modelo para Otimização da Classificação de Trens (*Optimization Model for Train Classification* - OMTC), gera um plano de classificação para solucionar um cenário do problema. Ela foi incluída em um processo iterativo, denominado DCOP Iterativo (*Iteractive DCOP* - I-DCOP), que permite solucionar cenários mais complexos do problema. Um plano de classificação descreve como reordenar vagões recebidos em um pátio formando novos trens e foi representado no OMTC e no I-DCOP utilizando a codificação apresentada por Maue (2011). Essa codificação utiliza uma sequência de números binários, chamada *bitstring*, que descreve as ações que os vagões devem executar no pátio. A partir dela é possível derivar um plano de classificação viável e ótimo para um problema, desde que as *bitstrings* de vagões pertencentes ao mesmo trem atendam algumas restrições. Um plano viável resulta na formação correta[1] dos trens e um plano ótimo minimiza seu tempo de execução. O OMTC contém essas restrições e uma função que otimiza a quantidade total de *roll-ins*[2] dos vagões. O I-DCOP estende o OMTC incluindo restrições de capacidade e quantidade máximas para as linhas de ordenação do pátio. Cada iteração do processo inclui ou altera os valores possíveis para as *bitstrings* (domínio) do OMTC, para os vagões que violaram alguma restrição. O OMTC e o I-DCOP foram avaliados com cenários fictícios baseados em dados reais. O OMTC gerou planos de classificação viáveis e ótimos para os cenários, minimizando a quantidade total de *roll-ins* dos vagões. O I-DCOP resolveu cenários envolvendo restrições mais complexas, obtendo soluções sub-ótimas. Ele mostrou ainda como um problema de otimização distribuída de restrições pode incluir restrições adicionais a partir de domínios definidos de forma iterativa.

Palavras-Chave: Classificação de Trens, Planos de Classificação, Pátios de Classificação, DCOP.

---

[1] "Correto" nesse contexto significa de acordo com critérios pré-estabelecidos. No caso do problema da classificação de trens, o critério identifica para qual trem de saída o vagão é destinado e sua ordem no mesmo.

[2] *Roll-in* é uma operação quer permite direcionar um vagão à uma linha de ordenação específica

Abstract

In this research the train classification problem was modeled as a Distributed Constraint Optimization Problem (DCOP). This model, named Optimization Model for Train Classification (OMTC), generates a classification schedule to solve a specific classification problem. The OMTC was included into an iterative process, named Iterative DCOP (I-DCOP), which solves the problem for more complex scenarios. A classification schedule describes how to arrange incoming cars in the yard into different train formations and it was presented on OMTC and I-DCOP using the notation proposed by Maue (2011). This notation is composed by sequences of bits, named "bitstrings", each one representing the actions a car should take on the classification yard. It is possible to derive feasible and optimal classification schedules, based on constraints between "bitstrings" of cars related to the same output train. A feasible schedule results on the correct[3] formation of the outgoing trains and an optimal schedule minimizes its execution time. The OMTC includes the constraints needed for an optimal and feasible schedule and a function to optimize the total number of roll-ins[4]. The I-DCOP extends the OMTC including the constraints of limited sorting tracks capacity and amount. Each iteration includes or changes the "bitstrings" possible values (domain) on the OMTC, only for the cars which violate some constraint. Both OMTC and I-DCOP have been measured using fictitious scenarios based on real data. The OMTC has generated optimal and feasible schedules to the problem scenarios, optimizing the total number of roll-ins. The I-DCOP solved more complex scenarios, providing sub-optimal solutions. It also has showed that distributed constraint optimization problems can include additional constraints based on interactively defined domain.

Keywords: Train Classification, Classification Schedules, Classification yards, DCOP.

---

[3] "Correct" on this context means: according to pre-defined criteria. In the case of the train classification problem, the criteria defines the output train of each car and this position on it.

[4] Roll-in identifies an operation where a car should be switched to a specific sorting track.

Lista de Figuras

Lista de Tabelas

Lista de Abreviaturas

ADOPT	*Asynchronous Distributed OPTimization*
ANTF	Agência Nacional de Transportes Ferroviários
CNT	Confederação Nacional de Transportes
DCOP	*Distributed Constraint Optiomization Problem* ou Problema de Otimização Distribuída de Restrições
DPOP	*Distributed Pseudotree Optimization Procedure*
DTREE	*Distributed Tree Optimization*
FRODO	*FRamework for Open/Distributed constraint Optimization*
IBGE	Instituto Brasileiro de Geografia e Estatística
I-DCOP	DCOP Iterativo (*Iterative DCOP*)
Ipea	Instituto de Pesquisa Econômica Aplicada
OMTC	Modelo para Otimização da Classificação de Trens (*Optimization Model for Train Classification*)
SBB	*Swiss Railways*
XML	*Extensible Markup Language*

Lista de Símbolos

b	*Bitstring* utilizada na representação do plano de classificação.
B	Plano de classificação.
C	Capacidade de uma linha de classificação.
c_{pull}	Tempo para realização de um *pull-out*.
c_{push}	Tempo para separar um vagão e direcioná-lo para uma linha de classificação.
D	Domínio finito e discreto para uma variável de um DCOP.
d	Valor atribuído a uma variável em um DCOP.
F	Custo global de um DCOP.
g_m	Quantidade de grupos do trem de saída m.
g_{max}	Grupo com maior número de vagões.
g_{min}	Grupo com menor número de vagões.
G	Maior tipo da sequência de vagões de entrada.
h	Tamanho do plano de classificação, ou seja, a quantidade de passos de ordenação.
i	Passo de ordenação de um plano de classificação.
k	Número de máximo de cadeias de uma decomposição de cadeia dada de acordo com o algoritmo GCD.
l	Quantidade de trens de entrada.
m	Quantidade de trens de saída de um cenário de classificação.
n	Quantidade de vagões da sequência de entrada do problema de classificação de trens.
n_m	Quantidade de vagões do trem de saída m.
r	Quantidade de operações de *roll-in*.
s	Tamanho do trem, obtido pela quantidade de vagões do mesmo.

t	Tipo de um vagão, que determina seu grupo.
v	Variável de um DCOP.
W	Quantidade total de linhas de ordenação.
$w(i)$	Peso de uma linha de ordenação, que representa a quantidade de vagões que ocuparam essa linha antes i-ésimo passo de ordenação
x	Agente de um DCOP.
τ	Representação de um trem, composta pelos vagões.
τ_x	Identifica um vagão dentro de um trem, onde o índice x indica a sua posição relativa aos demais vagões do trem.

Sumário

Capítulo 1

Introdução

A classificação de trens em um pátio ferroviário consiste em receber trens de diferentes origens com destinos diversos e organizar seus vagões de acordo com restrições como, por exemplo plano de viagem, linhas disponíveis no pátio, etc. No contexto desse trabalho, o termo "classificação" refere-se, portanto, ao processo de ordenação e montagem de novas composições de trens. Esse processo de "desmontar" e "montar" novas composições de trens é essencial ao transporte ferroviário, caracterizado por possuir um custo fixo elevado, devido ao arrendamento da malha, terminais e materiais rodantes, e um custo variável (mão de obra, combustível e energia) relativamente baixo (REIS, 2007). Essa característica faz com que a escala seja um fator fundamental para diluir os custos fixos e aumentar a margem de lucro da ferrovia (REIS, 2007). Como nem sempre o número de vagões para um mesmo destino é suficiente, trens com diferentes destinos são agrupados para compartilhar determinado trecho da viagem. Tal compartilhamento permite reduzir os custos do transporte, mas gera a necessidade dos pátios de classificação, responsáveis por redistribuir a composição original em novas composições por meio da classificação de trens.

Além do contexto prático, no qual a otimização de um pátio ferroviário pode trazer ganhos substanciais a esse modal de transporte, o problema de classificação de trens é complexo, devido ao grande número de variáveis e restrições que devem ser consideradas. A geração automática de um plano capaz de otimizar os recursos e reduzir o tempo de permanência dos trens nesses pátios é um campo interessante de investigação, permitindo a avaliação de diferentes estratégias para otimizar esse problema do mundo real.

O presente trabalho propôs-se, portanto, em gerar um plano que resolve um problema de classificação de trens em pátios ferroviários, otimizando seu tempo de execução para

melhorar a operação de pátios ferroviários. O problema de classificação de trens e o escopo considerado nessa pesquisa estão descritos em maiores detalhes no Capítulo 2.

1.1. Motivação

Como o modal ferroviário caracteriza-se por um custo fixo elevado (material rodante), mas com custo de mão de obra, combustível e energia relativamente baixos (REIS, 2007), a escala de operações influencia diretamente na diluição desses custos fixos, aumentando a margem de lucro das ferrovias. Os pátios de classificação são utilizados como uma das formas de minimizar esse custo, à medida que permitem que composições de trens com diferentes destinos compartilhem trechos de viagem.

Fleury (2007) realizou um estudo sobre os problemas do modal ferroviário nacional no qual constatou que 65% dos empresários indicam a indisponibilidade de rotas como um dos principais fatores para o não uso desse modal. Essa indisponibilidade de rotas pode estar associada a dois fatores: infraestrutura e estratégias operacionais. Os problemas de infraestrutura incluem a falta de terminais de transbordo, linhas férreas na rota desejada ou mesmo falta de capacidade da linha. Independente da necessidade de investimentos em infraestrutura (para melhorar a disponibilidade dessas rotas) existem melhorias operacionais que podem minimizar esses problemas. Portanto, há espaço para otimização de processos operacionais, como por exemplo, a classificação de trens que ocorre nos pátios, trazendo assim uma motivação vinculada ao estudo de um problema complexo com aplicação no mundo real.

A geração de planos para a realização da classificação de trens nos pátios pode ser considerada uma tarefa complexa devido ao grande número de variáveis e restrições envolvidas como, por exemplo plano de viagem dos trens, quantidade de linhas do pátio, capacidade das linhas de classificação, etc. Com isso, além de uma aplicação prática significativa, o problema abordado apresenta uma complexidade a ser suplantada do ponto de vista computacional, gerando a necessidade de estudo sobre a modelagem desse problema, para que o mesmo seja resolvido com um esforço computacional adequado.

1.2. Problema de Pesquisa

O problema de pesquisa abordado nesse trabalho é basicamente um problema de ordenação aplicado a otimização de pátios ferroviários, denominado de classificação de trens. A classificação de trens é o processo de reordenar vagões em diferentes composições de acordo com planos previamente definidos, denominados planos de classificação (do inglês: *classification shedule*), de acordo com critérios que regem a formação de novas composições (MAUE, 2011).

De forma pragmática, o problema pode ser definido como um problema de geração de **planos de classificação viáveis e ótimos**. Um **plano de classificação** descreve as operações que devem ser aplicadas aos vagões que chegaram ao pátio para formar corretamente os trens de saída. O plano é dito **viável** se para uma determinada sequência de vagões de entrada é possível formar corretamente os trens de saída. O termo "corretamente" indica que os vagões foram direcionados aos trens de saída previamente definidos e que sua posição relativa aos demais vagões do mesmo trem seguem os requisitos de ordenação especificados. O plano é considerado **ótimo** quando possui a menor quantidade de passos de ordenação necessários, quantidade essa que está diretamente relacionada ao tempo de execução total do plano para formação dos trens de saída.

Um passo de ordenação está associado à operação de pátios ferroviários e significa, de forma simplificada, uma distribuição dos vagões vindos de uma linha de ordenação entre as demais linhas de ordenação disponíveis, sabendo previamente para qual linha cada vagão deve ser destinado. Esse processo operacional será descrito em maiores detalhes na seção 2.4.

1.3. Objetivos

O objetivo geral da pesquisa foi modelar o problema da classificação de trens como um Problema de Otimização Distribuída de Restrições (DCOP). Esse modelo de otimização busca encontrar planos de classificação viáveis e ótimos, que resolvam um cenário específico do problema. Para alcançar esse objetivo foi necessário atingir os seguintes objetivos específicos:

- Formalizar o problema de classificação de trens, definindo o critério utilizado para ordenação dos vagões no pátio e demais restrições a serem consideradas;

- Implementar o algoritmo *Greedy Chain Decomposition* (GCD), que gera um plano de classificação viável e ótimo para um determinado cenário do problema;
- Definir o problema de classificação de trens como um DCOP, criando um Modelo para Otimização da Classificação de Trens (OMTC), capaz de gerar planos de classificação viáveis e ótimos;
- Avaliar o OMTC, comparando seu resultado com o algoritmo GCD e escolher um algoritmo DCOP que melhor se adequa a esse modelo;
- Estender o OMTC para incluir as restrições que determinam quantidade e capacidade máximas para as linhas de ordenação do pátio;
- Avaliar o desempenho OMTC e sua extensão, utilizando cenários fictícios baseados em dados reais de um pátio ferroviário.

1.4. Contribuições

Esse trabalho traz como principais contribuições: (i) concepção de um modelo DCOP para um problema complexo do mundo real, denominado OMTC, (ii) concepção de um processo iterativo que utiliza o OMTC e inclui restrições adicionais por meio de domínios definidos de forma iterativa e (iii) lições aprendidas sobre diferentes abordagens para concepção do OMTC.

1.5. Estrutura do trabalho

O trabalho está organizado da seguinte maneira: o Capítulo 1 fornece uma visão sobre o problema de pesquisa e sua contextualização, seus objetivos e a motivação para desenvolvimento da pesquisa, o Capítulo 2 descreve em maiores detalhes o problema real tratado na pesquisa e os trabalhos relacionados, o Capítulo 3 descreve o embasamento teórico para modelar um DCOP, o Capítulo 4 descreve o OMTC definido para o problema, sua extensão pelo uso do I-DCOP e o processo criado para geração e verificação dos planos de classificação obtidos e o Capítulo 5 descreve os resultados da avaliação do OMTC e do I-DCOP, utilizando cenários fictícios baseados em dados reais.

Capítulo 2

Classificação de Trens

O problema de classificação de trens está relacionado diretamente à característica do modal de transporte ferroviário. O modal de transporte ferroviário é dependente da malha ferroviária que o suporta e possui um custo fixo elevado devido ao arrendamento da malha, terminais e materiais rodantes (REIS, 2007). Portanto, para aumentar a lucratividade desse modal é necessário ampliar ao máximo a escala de transporte. Para reduzir os custos operacionais, composições de trens são agregadas para compartilhar trechos de viagem, caso a quantidade de vagões para um determinado destino não seja suficiente para justificar o custo operacional. Essas composições precisam ser posteriormente classificadas e reorganizadas nos pátios de classificação. Essa prática ilustra o problema de classificação de trens que será descrito nesse capítulo. As seções do Capítulo abordarão o problema de classificação de trens, delimitando e formalizando como o mesmo será tratado nessa pesquisa.

2.1. Classificação de Trens

Uma necessidade que emerge da busca por sustentabilidade no transporte ferroviário é a classificação de trens. Isso ocorre devido ao fato de que, em alguns casos, composições de trens são agregadas com o intuito de compartilhar trechos da viagem. Com isso, em determinado momento da viagem, essas composições precisam ser desagrupadas para seguir aos seus destinos. Esse processo é também denominado classificação de trens.

A classificação de trens consiste em receber trens vindos de diferentes origens com destinos diversos e ordenar os vagões para gerar novas composições (trens de saída), considerando restrições como: plano de viagem, quantidade de linhas disponíveis, tamanho das linhas, destino, tipos de vagões, etc. Esse processo ocorre em instalações específicas,

existentes na malha ferroviária, denominadas pátios. Para entender o problema de classificação de trens é necessário entender o funcionamento básico dessas instalações.

2.2. Pátios Ferroviários

O transporte ferroviário de carga depende diretamente da malha ferroviária existente e de como a mesma é operacionalizada. Segundo a pesquisa realizada pela Confederação Nacional de Transportes (CNT) em 2011, o Brasil possui 30.051 km de malha ferroviária, dos quais 28.614 km são ferrovias concedidas à iniciativa privada (CNT, 2011). Comparado a outros países, percebe-se que a densidade da malha ferroviária ainda é baixa, conforme pode ser observado na Figura 1.

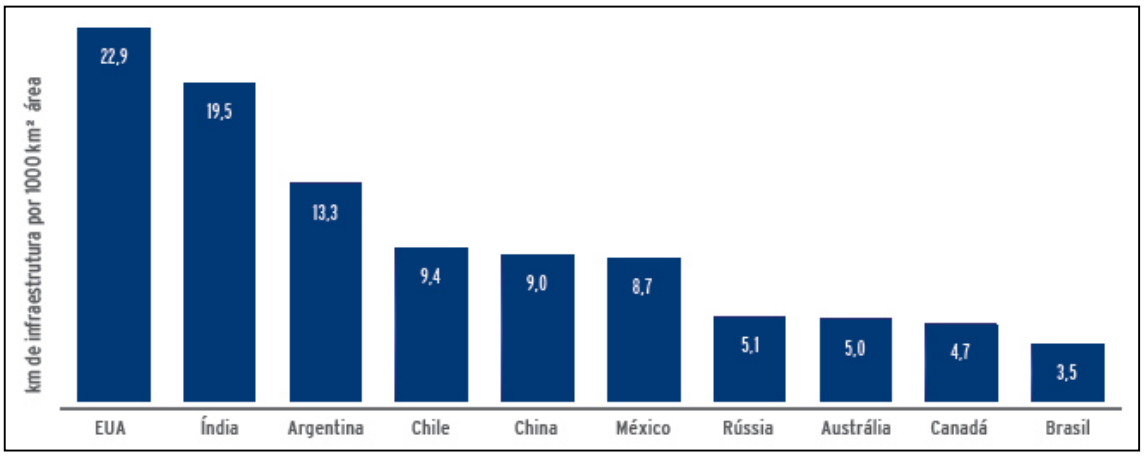

Figura 1 – Densidade do Transporte Ferroviário

Fonte: Pesquisa CNT de Ferrovias 2011

Apesar dos investimentos efetuados no setor ferroviário após o processo de concessão das ferrovias brasileiras, a malha ferroviária ainda não comporta uma movimentação significativa de transporte de carga. Ainda, segundo a pesquisa CNT 2011, a extensão atual destinada ao transporte de carga é composta por 12 malhas concedidas, que equivalem a 28.614 km, representando 94,4% do sistema ferroviário nacional.

As ferrovias são mais indicadas para o transporte de grandes volumes de carga em longas distâncias devido as suas características técnicas e econômicas. Porém, a economia desse modal de transporte só é realmente obtida quando a malha ferroviária utiliza uma

grande capacidade operacional, devido ao alto custo de construção da malha e da aquisição de material rodante.

Um aspecto operacional que pode ser otimizado para melhorar a utilização da malha ferroviária atual são os terminais. Segundo mesma pesquisa "a eficiência do transporte ferroviário está diretamente relacionada com as condições de infraestrutura oferecidas pelos terminais, respeitando os diversos tipos de cargas, pois envolvem agilidade, informação e segurança para o cliente no transbordo de sua carga". O grau de importância e sofisticação de um pátio está condicionado à frequência de operações que precisam ser executadas no mesmo, tornando o investimento em ferrovias rentável (FALAVINHA, 1982). Para Wright e Ashford (1997), os terminais são um dos componentes mais importantes em um sistema de transporte, já que seu custo compreende uma parte significativa do custo total do transporte. Um trem de carga viaja em média 92km/dia, uma viagem que dura aproximadamente 1 hora. Durante as demais 23 horas do dia, os vagões permanecem em um pátio ou outro tipo de terminal. Portanto, a redução do tempo gasto nesses pátios ou terminais é algo que afeta diretamente o custo operacional do transporte ferroviário.

2.3. Pátios de Classificação

Um pátio, segundo o glossário da Agência Nacional de Transporte Ferroviário (ANTF), é uma "área de esplanada em que um conjunto de vias é preparado para formação de trens, manobras e estacionamento de veículos ferroviários e outros fins" (ANTF, 2013). Para Wright e Ashford (1997), a principal função de um pátio ferroviário é a classificação de trens. Os pátios que possuem essa como principal função são denominados na literatura como **Pátios de Triagem** (CHANDRA, 2007) (FALAVINHA, 1982) ou **Pátios de Classificação** (JACOB, MARTON, *et al.*, 2011) (MAUE, 2011). Nessa dissertação é utilizada a denominação Pátio de Classificação, tratada como sinônimo de Pátio de Triagem. Os pátios de classificação podem ainda ser separados em três categorias principais (CHANDRA, 2007):

- **Pátio plano:** as linhas estão praticamente no mesmo nível de elevação e os vagões são realocados para o processo de classificação com o auxílio de um motor.
- **Pátio por gravidade:** o nível de elevação natural do solo permite a colocação de linhas com inclinação. As linhas permitem que os vagões se movimentem no processo de classificação pela ação da gravidade.

- **Pátio com rampa:** uma rampa artificial é criada para gerar a inclinação necessária para o deslocamento dos vagões. Nesse tipo de pátio, os vagões são puxados por um motor até a rampa, de onde então são direcionados pela ação da gravidade para as linhas de ordenação.

Os pátios com rampa são mais eficientes e estão gradualmente substituindo os pátios planos (DAGANZO, DOWLING e HALL, 1983). As considerações efetuadas nessa dissertação referem-se em geral aos pátios com rampa, mas podem ser generalizadas para os demais tipos de pátio, à medida que o problema de classificação de trens apresenta-se da mesma forma nos três tipos citados.

2.4. *Layout* e Operação dos Pátios de Classificação

O *layout* dos pátios de classificação busca minimizar o tempo de permanência dos vagões e permitir que o processo de classificação ocorra de forma rápida (CHANDRA, 2007). Um *layout* típico contém áreas para as operações básicas de recepção, classificação e expedição posicionadas em linha, conforme Figura 2:

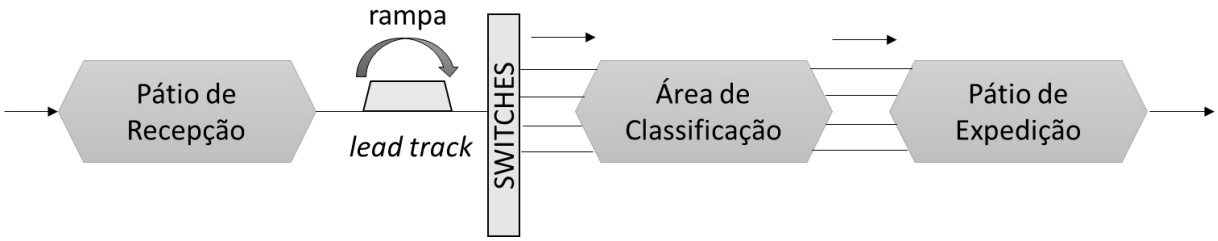

Figura 2 – Layout Básico de Pátio de Classificação

Fonte: Adaptado a partir do esquema apresentado por Maue (2011)

Cada área consiste basicamente de um conjunto de linhas férreas paralelas, reservadas para um tipo específico de operação. A conexão entre o pátio de recepção e a área de classificação é chamada de *lead track*. Entre o final da *lead track* e a área de classificação existe uma árvore de *switches* capaz de direcionar cada vagão para uma linha específica. As linhas da área de classificação são conectadas ao pátio de expedição via outra árvore de *switches.* A rampa, pode ou não existir, caracterizando os pátios com rampa, conforme já mencionado. Podem existir variações dessa estrutura básica, normalmente efetuadas por falta espaço, para instalação das linhas férreas.

O *layout* típico para uma área de classificação apresenta um acesso às linhas da área de classificação pelos dois lados, conforme mostrado na Figura 3 (a). Há ainda opções de *layout* nos quais os trens de saída partem diretamente da área de classificação, como mostrado na Figura 3 (b). Alguns pátios contêm *layouts* mais avançados, como o mostrado na Figura 3 (b), que inclui uma rampa secundária. Independentemente do *layout*, os pátios de classificação contêm a subestrutura mostrada na Figura 3 (c), essencial ao processo de ordenação que será descrito nessa dissertação.

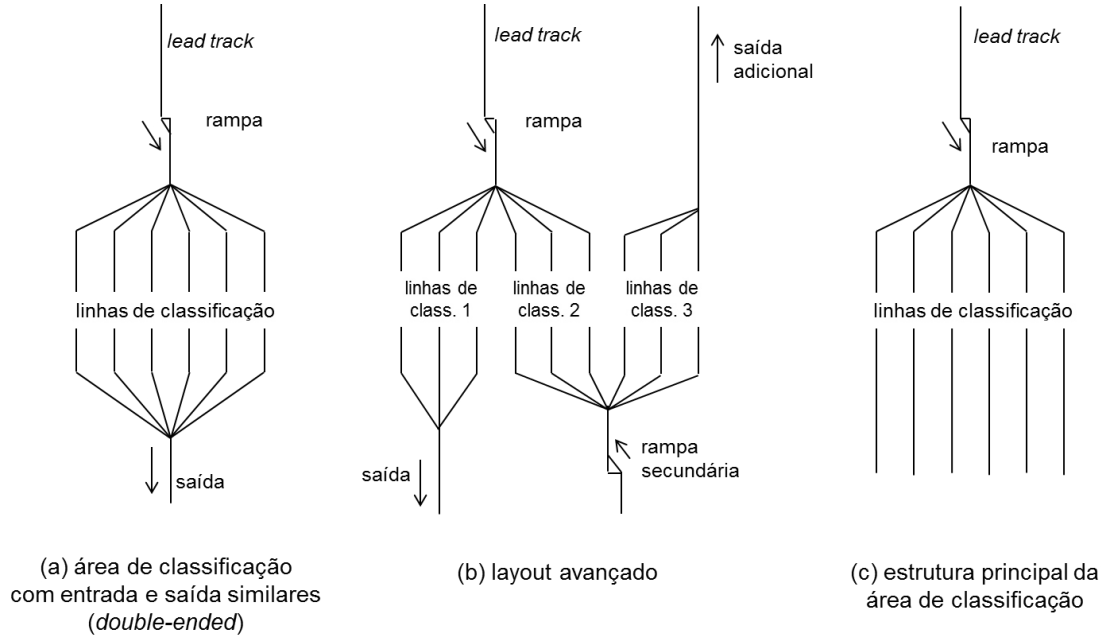

Figura 3 – Tipos de *layout* para Área de Classificação

Fonte: Adaptado do esquema apresentado por Maue (2011)

A operação de pátios ferroviários utiliza uma série de recursos materiais e humanos, tais como: locomotivas, vagões, inspetores de pátios, controladores de pátio, maquinistas, equipe mecânica para reparos, entre outros. Com isso, o desafio na operação de um pátio ferroviário está relacionado ao processo de coordenação eficiente do movimento dos trens, considerando uma quantidade limitada de recursos, além do plano de viagem e as prioridades de cada trem.

Diferentes estratégias para alocação dos recursos, definição dos planos de viagem, ou mesmo para a tomada de decisões em situações de conflito podem afetar a eficiência operacional do pátio gerando atrasos nas viagens planejadas. Portanto, a operação de um pátio ferroviário pode ser considerada uma tarefa complexa, à medida que envolve diferentes

variáveis que se relacionam de forma dinâmica para gerar informação ao processo de coordenação. Em ambientes dinâmicos, como o descrito acima, o uso de modelos matemáticos exatos torna-se complexo devido ao fato de não ser possível definir as restrições para cada uma das variáveis.

Vários autores definem as principais operações de um pátio de classificação, entre eles, Wright e Ashford (1997), Chandra e Agarwal (2007), Falavinha (1982) e Maue (2011). Esses autores descrevem três funções consideradas básicas em pátios ferroviários de classificação:

- **Recepção:** recebimento de trens de entrada vindos da linha principal, suportada por um conjunto de linhas.
- **Classificação:** onde os vagões são ordenados e classificados de acordo com algum critério, por exemplo, em blocos de destino comum.
- **Expedição:** onde os blocos ordenados são agrupados em trens de saída, aguardando a expedição.

O diagrama da Figura 4 mostra a operação básica do processo de classificação de trens. Esse processo é composto basicamente de duas operações realizadas com os vagões, denominadas *roll-in* e *pull-out*:

- ***Roll-in*:** consiste em separar os vagões vindos da *lead track,* de forma individual, empurrando-os sobre a rampa, destinando cada vagão para uma linha específica da área de classificação, de acordo com a posição atual da árvore de *switches*. Essa operação necessita, portanto, que seja definida a linha de destino para cada vagão que chega a rampa.
- ***Pull-out*:** consiste em puxar todos os vagões que estão dispostos em uma linha da área de classificação e direcioná-los novamente à *lead track*, para que um novo *roll-in* seja executado.

O processo de classificação de trens é, portanto, composto de diversas operações de *roll-in* e *pull-out*, com o objetivo de formar "corretamente" os trens de saída, conforme mostrado no diagrama. A próxima seção descreve o requisito de ordenação que define como formar os trens de saída.

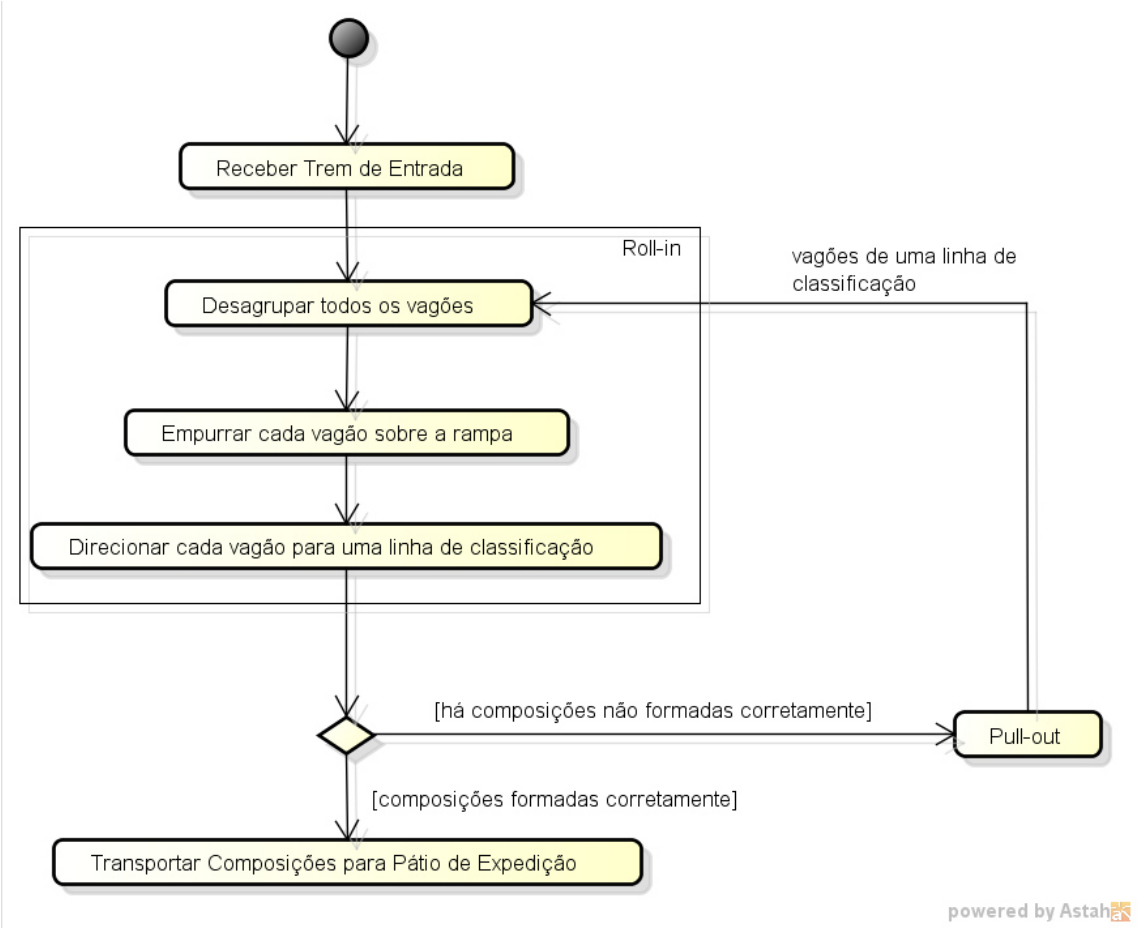

Figura 4 – Operação básica de um Pátio de Classificação

2.5. Requisito de Ordenação para Formação dos Trens de Saída

Existem diferentes requisitos de ordenação para formação de novas composições em um pátio de classificação. Dahlhaus *et al.* definiem um *framework* sistemático para requisitos de ordenação para os trens de saída (DAHLHAUS, MANNE, *et al.*, 2000). Esses requisitos foram sumarizados por Hansmann *et al.* em (HANSMANN e ZIMMERMANN, 2008), que também descreveram um amplo *framework* de procedimentos de ordenação. Mais visões do processo de classificação de trens são descritas em (STEFANO, MAUE, *et al.*, 2007) e (GATTO, MAUE, *et al.*, 2009).

Para o problema de classificação utilizado nessa pesquisa, o requisito de ordenação considerado baseia-se no conceito de grupos de vagões. Um **grupo** é um conjunto de vagões que compartilha o mesmo destino em um trem de saída. Todos os vagões de um grupo aparecem de forma consecutiva no trem de saída, mas a ordem dos mesmos dentro do grupo é

arbitrária. A composição de um trem de saída é, portanto, definida pelos seus grupos e, opcionalmente, uma ordem para esses grupos. O grupo do vagão é definido pelo seu tipo, que é um valor numérico inteiro. De forma pragmática, o requisito de ordenação utilizado segue as seguintes regras:

- Vagões do mesmo tipo devem permanecer juntos no mesmo trem de saída;
- Vagões devem ser ordenados por tipo, de forma crescente e contínua;

Esse requisito de ordenação, baseado no conceito de grupo de vagões, recebeu diferentes denominações em outros trabalhos: "*Q-node with P-children*" por Dahlhaus (DAHLHAUS, MANNE, *et al.*, 2000), "*ordered g-blocks*" por (HANSMANN e ZIMMERMANN, 2008) e "*single output string*" em (STEFANO, MAUE, *et al.*, 2007).

2.6. Definição do problema

O objetivo dessa seção é definir formalmente o problema de classificação abordado na pesquisa. As definições e suposições descritas foram previamente definidas por Maue (2011) e a seção propõe-se apenas a resumir os principais itens para que os demais capítulos possam ser mais facilmente compreendidos. A delimitação do problema tratado pela pesquisa foi efetuada utilizando a notação definida nas próximas seções.

2.6.1. Vagões e Trens

Um **trem** é representado por uma s-tupla $(\tau_1, \dots, \tau_s)$ composta de vagões $\tau_x \in \mathbb{N}$, $x = 1, \dots, s$, onde s representa também o tamanho do trem. O vagão τ_1 é o primeiro vagão do trem, e τ_s é o último. Considerando dois vagões τ_x, τ_y de um trem, define-se que τ_x está na frente de τ_y se $x < y$. Quando um trem está posicionado na *lead track*, o τ_1 é o vagão mais próximo da rampa. Quando um trem está em uma linha da área de classificação, o τ_1 é o vagão mais próximo do final dessa linha. Cada vagão τ, pertencente a um trem, terá também um tipo, que é representado por um número inteiro positivo $t \in \mathbb{N}$. A locomotiva não é incluída na representação de um trem e pode estar no início ou final do mesmo, dependendo da situação.

Cada instância de problema de classificação de trens contém uma sequência de l trens de entrada. A concatenação de todos esses trens de entrada de acordo com sua ordem de chegada ao pátio é chamada de sequência de vagões de entrada, composta por n vagões.

Os trens de saída devem ser formados de acordo com o requisito de ordenação definido anteriormente. Cada instância de problema de classificação deve receber a quantidade de trens de saída que devem ser formados, definida por m, e também o maior tipo t de cada trem, que é chamado de G_m, onde m indica o trem de saída.

2.6.2. Tipos

O **tipo** t, mencionado anteriormente, é utilizado para representar características comuns dos vagões. Pode ser utilizado, por exemplo, para especificar o destino dos vagões. Os tipos de uma sequência de entrada de vagões variam de 1 a G, que define o maior valor de t pertencente a sequência. É assumida a premissa que existe ao menos um vagão para cada tipo t. Como é também possível que exista mais de um vagão do mesmo tipo, é adicionado um índice ao tipo do vagão: $t_x \in \mathbb{N},\ x = 1, \dots, n_t$, onde n_t representa a quantidade de vagões de tipo igual a t. O índice x representa a posição relativa do vagão em relação aos demais vagões do mesmo tipo, dentro da sequência de vagões de entrada. O índice $x = 1$ representa o primeiro vagão do tipo t a chegar ao pátio, $x = 2$ o segundo, e assim por diante.

2.6.3. Linhas de Classificação

Existe uma linha de classificação reservada para a formação final de cada trem de saída. Essas linhas são também chamadas de **linhas de saída** (*output tracks*). As demais linhas da área de classificação são utilizadas para o processo de ordenação e são também chamadas de **linhas de ordenação** (*sorting tracks*). Cada linha de classificação é utilizada para saída ou ordenação, mas não ambos e, nesse trabalho foi assumido que o número de linhas de saída é igual à quantidade de trens de saída. É executada a operação de *pull-out* uma vez para cada linha de ordenação. Quando houver uma quantidade limitada de linhas de ordenação W, pode ser necessário efetuar *pull-out* da mesma linha mais de uma vez. Nesse trabalho, mesmo quando a restrição de quantidade de linhas de ordenação for incluída, cada linha de ordenação sofre um único *pull-out*.

O número máximo de vagões que uma linha pode acomodar é chamado de capacidade da linha. As linhas de ordenação possuem capacidade uniforme, e a capacidade de uma única linha é chamada de C, podendo ser outra restrição ao problema. As linhas de saída possuem a capacidade necessária para conter o maior trem de saída formado no problema.

2.6.4. Plano de Classificação

Para descrever um plano de classificação e como as operações de *roll-in* e *pull-out* são definidas no mesmo, foi utilizada a representação de um plano de classificação de acordo com a codificação proposta por Maue (MAUE, 2011), que será brevemente descrita a seguir.

Um plano de classificação descreve de forma pré-definida como o processo de ordenação será realizado. As operações de *pull-out* são especificadas indicando apenas a linha de ordenação na qual a operação será realizada, à medida que todos os vagões que ocupam essa linha no momento serão puxados. Já a operação de *roll-in* especifica a linha de destino de cada vagão envolvido no processo. O *roll-in* sempre é efetuado em todos os vagões posicionados na *lead track*. Toda operação de *pull-out* seguida de um *roll-in* é chamada de passo de ordenação (*sorting step*) ou somente passo (*step*).

Um plano de classificação é composto de um *roll-in* inicial e uma sequência de h passos de ordenação, onde o h também representa o tamanho do plano. Para um plano de tamanho h, cada passo de ordenação pode ser numerado de 1 até h e referenciado como i-ésimo passo, $i = 1, \dots, h$. Um plano de classificação B é considerado **viável** se sua aplicação em uma sequência de entrada de vagões ordena corretamente os trens de saída, cada um em uma linha de saída arbitrária.

A representação do plano de classificação é composta por um conjunto de *bitstrings.* Uma *bitstring* $b = b_h \dots b_1$ é uma sequência de números binários que define o curso de um vagão no pátio. Quando b_i for igual a 1, o vagão deve estar na linha de ordenação na qual será realizado um *pull-out* no i-ésimo passo. Conforme definido anteriormente, uma linha de ordenação sofrerá somente um *pull-out*. Nesse caso, é assumido que a linha de ordenação que sofrerá o i-ésimo *pull-out* será a linha de ordenação com índice $i - 1$.

Para um plano de classificação B, a *bitstring* definida para um vagão τ_x será referenciada como b^{τ_x}. Para executar o processo de classificação definido em B, cada vagão será inicialmente direcionado (via operação de *roll-in* inicial) para a linha de ordenação na

qual ele realizará o primeiro *pull-out*. Para encontrar qual será essa linha, deve ser verificado o primeiro passo de ordenação que possui *bit* igual a 1. Se a *bitstring* for igual a 0, τ_x será enviado diretamente para a linha de saída do trem ao qual pertence. Isso ocorre, por exemplo, com o vagão 1, onde $b^1 = 0000$ para o plano B (Figura 5). As linhas de ordenação mostradas na Figura 5 são numeradas da direita para a esquerda, iniciando em 0. As *bitstrings* assinaladas aos vagões possuem o número de *bits* igual à quantidade de passos de ordenação que serão realizados, que no caso, também é igual quantidade de linhas de ordenação disponíveis. O *bit* mais à direita indica a operação que o vagão deve realizar na linha 0, o *bit* seguinte (direita para esquerda) indica a operação que o vagão deve realizar na linha 1, e assim sucessivamente. A Figura 5 mostra a execução de um plano de classificação, seguindo essa notação, para exemplificar e esclarecer as operações indicadas pelas *bitstrings*. Cada passo de ordenação será mostrado na figura e descrito.

O plano de classificação B é viável para uma sequência de entrada de vagões com $n = 17$, ordenados em $m = 2$ trens de saída, utilizando $h = 4$ passos de ordenação. Vagões da mesma cor pertencem ao mesmo trem de saída. Os vagões são identificados pelo seu tipo (que pode definir um destino comum, por exemplo) e seu índice (13_1, 13_2), responsável por identificar a ordem de chegada ao pátio para vagões pertencentes ao mesmo tipo, conforme descrito anteriormente. As linhas de classificação dedicadas ao processo de ordenação foram denominadas com o prefixo "c" e as duas linhas de saída foram denominadas com prefixo "s".

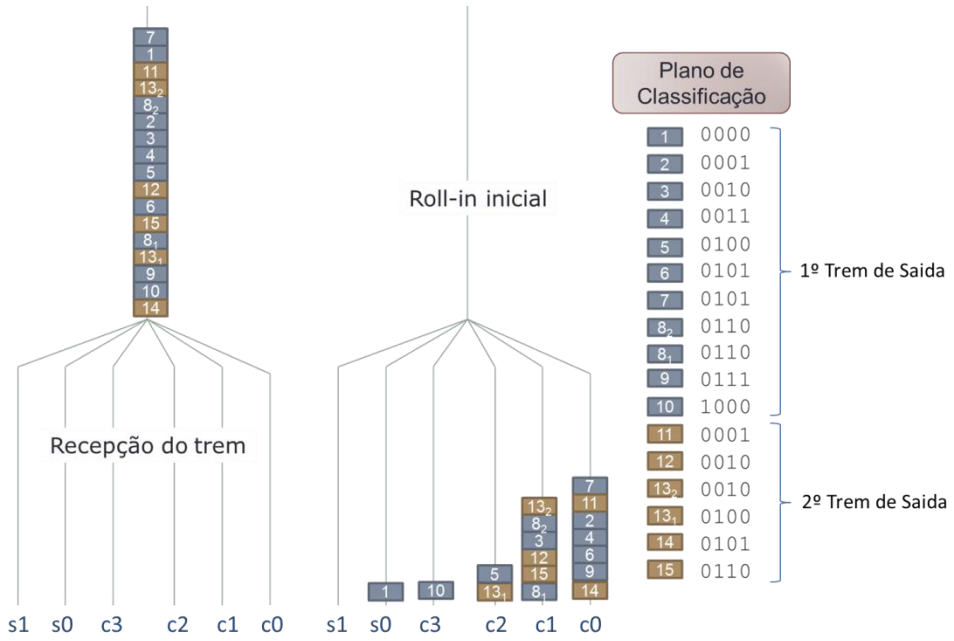

Figura 5 (a) – *Roll-in* inicial do cenário de exemplo (plano B)

A execução do *roll-in* inicial, representado na Figura 5 (a), distribui todos os vagões entre as linhas de ordenação disponíveis. Para fazer essa primeira distribuição, é verificado o primeiro *bit* 1 mais à direita de todos os vagões. O vagão é então direcionado para a linha de ordenação que sofrerá *pull-out* no passo indicado por esse *bit*. Por exemplo, o vagão 14 é direcionado para a linha ***c0*** pois sua *bitstring* é 010**1**, ou seja, o primeiro *bit* 1 da direita para a esquerda indica que esse vagão sofrerá *pull-out* no primeiro passo de ordenação, portanto na linha ***c0***. Já o vagão 5 é direcionado para ***c2***, pois tem em sua *bitstring* o valor 0**1**00. Como já mencionado, vagões que possuem a *bitstring* igual a 0, como por exemplo, o vagão 1, são direcionados diretamente para uma linha de saída disponível, no caso a ***s0***.

A partir do *roll-in* inicial, a execução do plano segue então analisando cada *bit* da *bitstring*, sempre da direita para esquerda. O primeiro passo de ordenação vai, portanto, efetuar um *pull-out* de todos os vagões que estão na linha de classificação ***c0***. A operação de *pull-out* faz com que esses vagões sejam direcionados para a *lead track* para um novo *roll-in*, que vai seguir a mesma estratégia adotada no *roll-in* inicial. Porém, agora será considerado somente o restante da *bitstring*, destacado na Figura 5 (b). Novamente, vagões que possuem 0 na parte considerada da *bitstring* serão direcionados para a linha de saída do seu trem, como ocorreu com o vagão 2 e vagão 11. Como esses dois vagões pertencem a trens de saída diferentes (informação que deve ser predefinida), serão direcionados a diferentes linhas de saída.

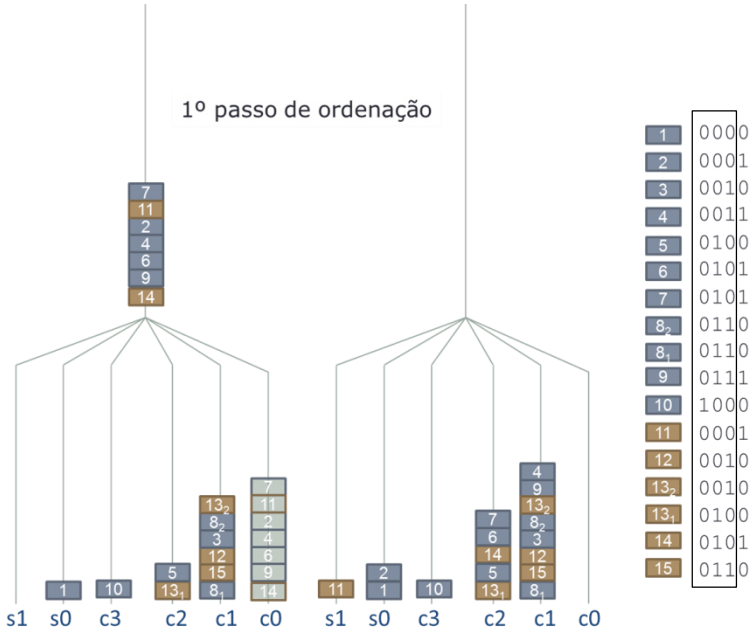

Figura 5 (b) – 1º passo do cenário de exemplo (plano *B*)

O mesmo processo ocorre até que todos os *bits* sejam avaliados, ou seja, até que sejam executados todos os passos de ordenação. Os demais passos são apresentados na Figura 5 (c), Figura 5 (d) e Figura 5 (e). De forma resumida, sempre após uma operação de *pull-out*, o vagão será direcionado para a próxima linha de ordenação definida com 1 na *bitstring*. Essa linha será onde o vagão sofrerá o próximo *pull-out*. Caso o restante da *bitstring* seja 0, o vagão será direcionado para a linha de saída do seu trem.

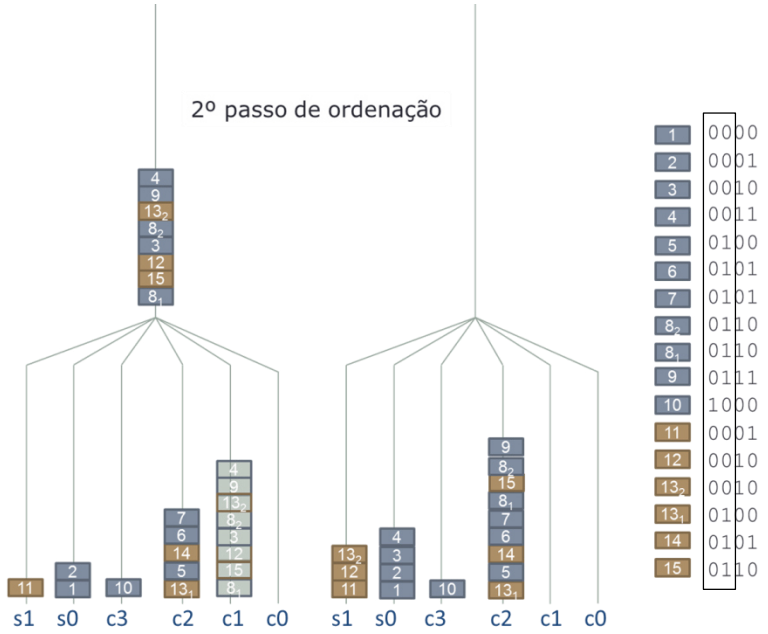

Figura 5 (c) – 2º passo do cenário de exemplo (plano *B*)

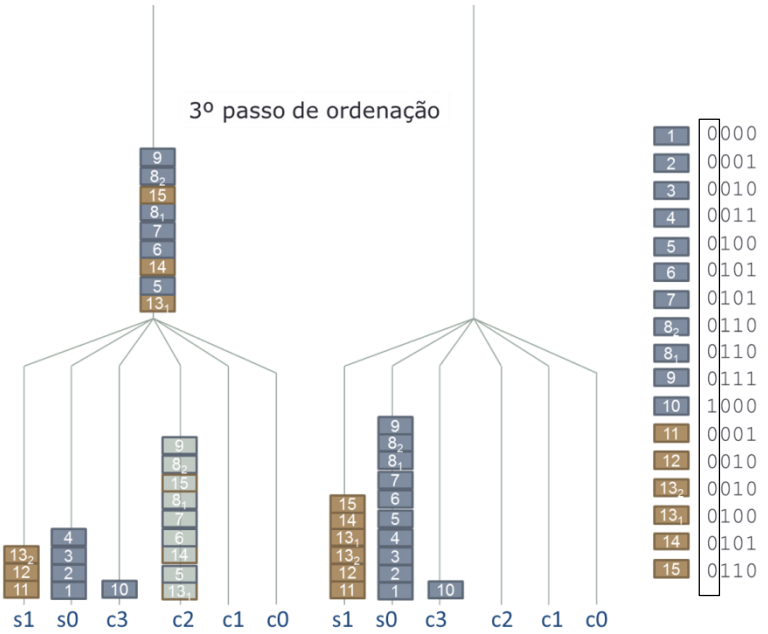

Figura 5 (d) – 3º passo do cenário de exemplo (plano *B*)

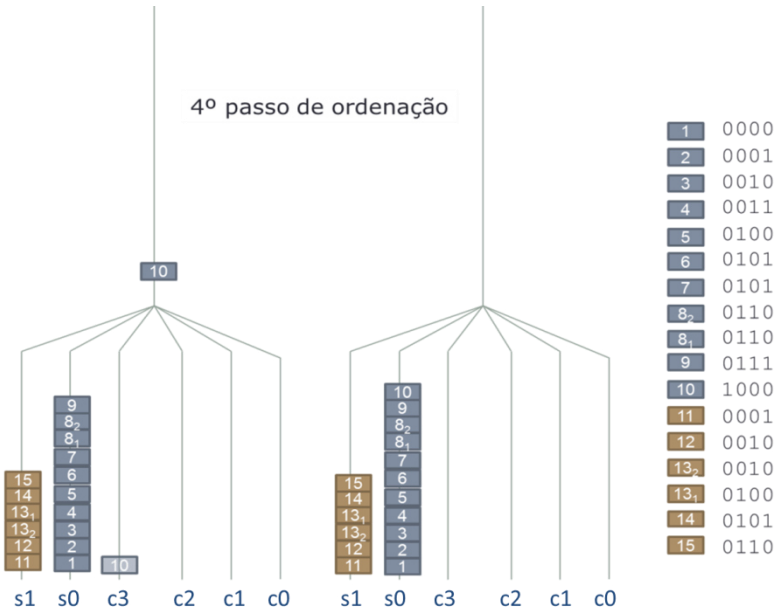

Figura 5 (e) – 4º passo do cenário de exemplo (plano B)

Fonte: Adaptado do cenário apresentado por Maue (2011)

As vantagens da representação de um plano de classificação utilizando essa codificação serão exploradas na próxima seção.

2.7. Benefícios da Representação do Plano de Classificação

A partir da representação de um plano de classificação apresentada anteriormente, é possível identificar características do processo de ordenação diretamente pelas *bitstrings*. Como um simples exemplo, se a *bitstring* contendo somente zeros é designada a qualquer vagão no plano B, esse vagão é enviado diretamente para a linha de saída do seu trem.

Além disso, é possível obter um **peso** para um plano de classificação B contando o número de bits b_i^j de B com $b_i^j = 1$, conforme Equação (1) (MAUE, 2011):

$$w(B) = \sum_{i=0}^{h-1} \sum_{j=1}^{n} b_i^j \tag{1}$$

Esse peso corresponde ao número total de *roll-ins* dos vagões, com exceção do *roll-in* final de cada vagão (que o direciona para a linha de saída). O cálculo do peso w, baseado na codificação de *bitstring*, pode também ser calculado por linha de classificação, conforme a

Equação (2). Nessa equação *n* representa a quantidade de vagões e *i* indica a linha de classificação para a qual está sendo calculado o peso. A restrição da capacidade máxima para uma linha de ordenação pode facilmente ser validada utilizando-se o peso de cada linha.

$$w(b_i) = \sum_{j=1}^{n} b_i^j \tag{2}$$

A representação utilizada também permite identificar um plano de classificação viável, utilizando o conceito de restrições. Conforme já mencionado, um plano é dito **viável** somente se o mesmo gera os trens de saída de acordo com os requisitos de ordenação especificados. Demais restrições podem ser adicionadas ao plano de classificação, porém o termo viável será sempre utilizado nesse trabalho para indicar que o plano forma os trens de saída de acordo com o critério de ordenação definido anteriormente. Relembrando, de forma resumida, o critério de ordenação adotado indica que em cada trem de saída os vagões serão organizados com tipos consecutivos e em ordem crescente, e que vagões do mesmo tipo devem permanecer no mesmo trem de saída.

Cada *bitstring* definida para os vagões é de fato um número na base 2. Isso permite definir uma relação de ordem entre as *bitstrings*, ou seja, considerar que $b < b'$, se e somente se, o valor de b na base 2 for menor que o valor de b' na base 2. A partir dessa relação de ordem, foram definidos e provados lemas e teoremas que podem ser utilizados para verificar se um plano é viável. O Teorema 1 é descrito de forma breve, pois é utilizado no OMTC.

> **Teorema 1**: Seja $\tau = \tau_1, \ldots, \tau_n$ a sequência de entrada de vagões de uma instância de problema de classificação contendo m trens de saída, e seja B um plano de classificação de tamanho h. B é um plano viável para τ se e somente se ambas as seguintes condições ocorrerem para cada par de vagões τ_x, τ_y de τ que pertencem ao mesmo trem de saída: $\tau_x > \tau_y \Rightarrow b^{\tau_x} \geq b^{\tau_y}$ (3) e $x < y \; e \; \tau_x > \tau_y \Rightarrow b^{\tau_x} > b^{\tau_y}$ (4) (MAUE, 2011, p. 26).

A partir do Teorema 1 é possível identificar a viabilidade de um plano comparando as *bitstrings* de vagões sequenciais. Esse teorema basicamente determina como definir as *bitstrings* para um par de vagões $\tau_x, \tau_y = \tau_x + 1$ de tipos consecutivos do mesmo trem de saída, a partir das seguintes regras: (i) τ_{x+1} nunca pode receber uma *bitstring* **menor** que a do vagão τ_x e (ii) se $\tau_x + 1$ chega ao pátio antes de τ_x, então $\tau_x + 1$ deve receber uma *bitstring* necessariamente **maior** que τ_x. Essas regras podem ser consideradas restrições, pois, ao

definir uma *bitstring* para um vagão, a mesma impõe uma restrição sobre o valor da *bitstring* de um vagão relacionado, que no caso é um vagão consecutivo pertencente ao mesmo trem de saída.

A primeira restrição (i) ocorre, por exemplo, no par de vagões 11 e 12 e também no par 12 e 13_2 do trem de saída direcionado para s1 (cor marrom), garantindo que a *bitstring* do vagão 12 não seja menor que a *bitstring* do vagão 11 e também que a *bitstring* do vagão 13 não seja menor que a *bitstring* do vagão 13_2 (Figura 5). Já a segunda restrição (ii), só ocorre no par de vagões 11 e 12, por isso os vagões 12 e 13_2 podem receber uma *bitstring* igual, já que os vagões chegaram ao pátio na ordem correta. A segunda restrição (ii) permite, portanto, reduzir a quantidade de passos de ordenação, desde que os vagões cheguem ao pátio pré-ordenados.

A partir da descrição de um plano de classificação utilizando a codificação de *bitstrings*, é possível descrever mais facilmente os métodos tradicionais de classificação de trens.

2.7.1. Métodos de Classificação Tradicionais

O método de classificação utilizado para resolver um problema de classificação de trens normalmente está relacionado ao requisito de ordenação que deve ser aplicado aos trens de saída. O requisito mais simples de ordenação, apresentado por um **trem unitário**, normalmente é utilizado quando há alto volume de tráfego, como no fluxo entre pátios. Um **trem unitário** é composto somente por vagões que compartilham o mesmo destino e, portanto, podem ter uma ordem arbitrária (MAUE, 2011). Esse tipo de trem é normalmente formado de maneira simples, reservando-se uma linha de classificação para o mesmo e efetuando sucessivos *roll-ins*, direcionando todos os seus vagões para a linha reservada. Ao final do processo, o trem está pronto para deixar o pátio. Porém, mesmo esse requisito mais simples de ordenação pode gerar necessidade de mais operações no pátio. Por exemplo, o trem unitário pode ser muito longo para ser formado em uma única linha de classificação. Em casos como esse, os vagões são direcionados para duas ou mais linhas de classificação, e, posteriormente coletados para formação do trem de saída no pátio de expedição. Essa técnica de formação do trem de saída, baseada na coleta de conjuntos de vagões de diferentes linhas de classificação, pode ser também aplicada a trens compostos de vários grupos, para isso

basta direcionar cada grupo para uma linha de classificação específica. Esse método de classificação é chamado de *single-stage sorting* (DAGANZO, DOWLING e HALL, 1983) (DAHLHAUS, HORAK, *et al.*, 2000) (STEFANO, MAUE, *et al.*, 2007).

O método *single-stage sorting*, em sua forma mais básica, consiste em reservar uma linha classificação para cada grupo do trem de saída, e direcionar todos os vagões para as linhas de classificação de acordo com seu grupo. Ao final, esses grupos são reagrupados, se necessário em alguma ordem especifica, formando o trem de saída. O trem unitário é somente um caso especial do *single-stage sorting*.

Já no caso de trens com muitos grupos, ou ainda trens com grupos pequenos ou compostos com somente um vagão, a aplicação do método *single-stage sorting* requer muitas linhas de classificação e deixa muito espaço ocioso nas mesmas. Com o objetivo de evitar esses problemas, o método *multistage sorting* pode ser aplicado. Esse método requer mais movimentos dos vagões, mas utiliza de forma mais eficiente o espaço disponível nas linhas de classificação (KRELL, 1962 *apud* MAUE, 2011, p.11) (SIDDIQEE, 1972 *apud* MAUE, 2011, p.11) (DAGANZO, DOWLING e HALL, 1983).

O método *multistage sorting* consiste basicamente em empurrar os vagões sobre a rampa mais de uma vez, efetuando as seguintes operações:

- **Ordenação primária:** processo inicial de separar cada vagão do trem de entrada direcionando-os para as linhas de classificação. Esse processo representa o *roll-in* inicial já descrito.
- **Ordenação secundária:** processo no qual um motor de manobras recupera os vagões de uma linha de classificação e empurra-os novamente sobre a rampa, para que seja efetuado um novo *roll-in*, direcionando cada vagão individualmente para uma nova linha de classificação. Esse processo ocorre de forma iterativa até que todos os trens de saída estejam formados corretamente. A ordenação secundária é a execução dos passos de ordenação.

Um plano de classificação pode descrever um *single-stage sorting* ou um *multistage sorting*. O problema foco analisado nesse trabalho considera o método *multistage sorting*, o qual envolve mais variáveis e restrições, gerando um problema mais complexo.

2.7.2. Estratégias de Ordenação dos Vagões para o *Multistage Sorting*

O método *multistage sorting* define que os vagões sejam direcionados mais de uma vez para as linhas de classificação, em um processo iterativo. Mas ele não define especificamente qual a estratégia utilizada para escolher qual vagão irá para determinada linha de classificação. Portanto, além da definição do método, é necessário definir a estratégia de ordenação dos vagões. Maue (2011) descreveu as seguintes estratégias utilizando a representação do plano com *bitstrings*: ordenação por trem, ordenação por grupos, ordenação triangular e geométrica. Ele apresentou cada estratégia de ordenação como uma classe de planos de classificação com características comuns. Na descrição das estratégias é considerado que algumas informações estão disponíveis. A primeira delas representa o número de trens de saída m, com suas respectivas quantidades de vagões para cada trem $n_1, \dots, n_m$, onde n_1 representa a quantidade de vagões do trem 1, n_2 referente ao segundo trem e assim sucessivamente, até a quantidade de vagões do m-ésimo trem. Também é fornecido o número de grupos de cada trem, segundo a mesma abordagem: $g_1, \dots, g_m$. A partir dessas informações podem ser definidos: g_{min}, que representa o grupo que contém o menor número de vagões e g_{max}, que representa o grupo com maior número de vagões. A seguir será fornecido um breve resumo de cada uma das estratégias.

A ordenação por trem compreende duas etapas (DAGANZO, DOWLING e HALL, 1983) (KRELL, 1962 *apud* MAUE, 2011, p.39) (SIDDIQEE, 1972 *apud* MAUE, 2011, p.39). A primeira etapa representa o *roll-in* inicial, na qual os vagões são separados de acordo com seus trens de saída, pois todos os vagões pertencentes a um trem de saída comum são enviados para uma mesma linha de ordenação. Os trens de saída resultantes desse processo, ainda desordenados, são processados sucessivamente na segunda etapa. Nessa etapa, cada trem é puxado sobre a rampa para efetuar um novo *roll-in*, ordenando os vagões de acordo com seus grupos, cada um sendo enviado para uma linha de classificação. Os grupos de vagões são então movidos das linhas de classificação na ordem necessária (caso exista) e agrupados para formar o trem de saída ordenado. Ao término desse agrupamento, o processo continua com o próximo trem.

Como os *roll-ins* realizados na segunda etapa não consideram pré-ordenação, apenas uma linha de classificação é utilizada para cada grupo. Essa estratégia ocupa m linhas de classificação após o *roll-in* inicial. Portanto, o número necessário de linhas de classificação é no mínimo $m + g_{min} - 1$ e no máximo $m + g_{max} - 1$. Esse método de classificação é

também chamado de agrupamento inicial de acordo com os trens de saída (*initial grouping according to outbound trains*) (KRELL, 1962 *apud* MAUE, 2011, p.40) (SIDDIQEE, 1972 *apud* MAUE, 2011, p.40).

A estratégia de ordenação denominada de ordenação simultânea em sua forma básica também é composta por duas etapas, mas, ao contrário da estratégia de ordenação por trem, direciona os vagões de acordo com os grupos dos trens de saída no *roll-in* inicial (FLANDORFFER, 1953 *apud* MAUE, 2011, p.40) (KRELL, 1962 *apud* MAUE, 2011, p.40) (PENTINGA, 1959 *apud* MAUE, 2011, p.40) (SIDDIQEE, 1972 *apud* MAUE, 2011, p.40). Em geral, após a primeira etapa, as linhas de classificação vão conter vagões de diferentes trens de saída, mas somente de um grupo de cada trem. Isso ocorre, à medida que a pré-ordenação de vagões não é considerada, portanto mesmo os grupos que já estejam em ordem correta acabam sendo separados. Nessa estratégia, uma operação de *pull-out* é realizada para cada grupo após a primeira etapa, gerando, assim um número g_{max} de passos de ordenação. Esse é o maior valor entre as variações de ordenação simultânea, mas mesmo assim ainda é menor que o número de passos de uma ordenação por trem, em um pátio de classificação sem restrições (MAUE, 2011).

A descrição da ordenação simultânea básica indica que os vagões devem ser enviados na primeira etapa para uma linha de ordenação, para posteriormente serem enviados às linhas de saída. Isso significa que todos os vagões realizam ao menos duas operações de *roll-in*. Em pátios de classificação que permitem acesso às linhas de saída, a partir da rampa principal (Figura 3-b), o plano de classificação pode ser reduzido de um passo, enviando os grupos diretamente às linhas de saída. Essa estratégia também é denominada método simultâneo (*simultaneous method*), triagem simultânea (*simultaneous mashalling*) (PENTINGA, 1959 *apud* MAUE, 2011, p.41), ordenação por bloco (*sorting by block*) (DAGANZO, DOWLING e HALL, 1983) e agrupamento inicial de acordo com o índice (*initial grouping according to subscript*) (SIDDIQEE, 1972 *apud* MAUE, 2011, p.42).

Existem ainda duas variações da estratégia simultânea, denominadas ordenação triangular (*triangular sorting*) e ordenação geométrica (*geometric sorting*). A ordenação triangular é uma ordenação simultânea que permite no máximo três *roll-ins* para cada vagão, incluindo o *roll-in* final, que direciona os vagões para a linha de saída. A ordenação triangular foi considerada nos seguintes trabalhos: Pentinga (1959 *apud* Maue, 2011, p.42), Krell (1962 *apud* Maue, 2011, p.42), Siddiqee (1972 *apud* Maue, 2011, p.42), (DAGANZO, DOWLING e HALL, 1983) e (DAGANZO, 1986).

A ordenação geométrica não possui nenhuma restrição relacionada a quantidade de *roll-ins*. Isso, na representação de um plano de classificação, significa que não há restrição alguma referente as *bitstrings* associadas a um vagão. O desempenho desse método é igual a $g_{max} \leq 2^h$ para uma quantidade h de passos de ordenação (Krell, 1962 *apud* Maue, 2011, p.43). Essa estratégia de ordenação minimiza o tamanho do plano de classificação sem considerar a ordem de entrada dos vagões no pátio.

2.7.3. Característica Operacional de Pátios de Classificação

A aplicação dos métodos de classificação *single-stage sorting* e *multistage sorting* normalmente ocorrem de forma concorrente em um pátio, compartilhando a mesma infraestrutura. Em um período de 24 horas, trens de entrada chegam constantemente ao pátio de classificação, contendo vagões que devem formar diferentes trens de saída. É efetuado o *roll-in* desses vagões de forma contínua, direcionando-os para linhas de classificação de trens unitários ou efetuando uma ordenação primária, como parte de um processo *multistage sorting*. Esse processo de *roll-in* contínuo é interrompido em determinado momento, normalmente uma ou duas vezes ao dia, para que a rampa seja utilizada exclusivamente para a operação de ordenação secundária por algumas horas. Durante esse período, trens de saída não envolvidos na ordenação secundária deixam a área de classificação em direção ao pátio de expedição, que também mantém os trens de saída formados no processo *multistage sorting*. As linhas liberadas por esses trens são novamente incluídas no processo *multistage sorting* corrente, particularmente para formação de trens de saída, como sugerido em Krell, 1962 *apud* Maue 2011. Após o término da ordenação secundária, o modo de operação do pátio de classificação volta para ordenação primária.

Em pátios que possuem uma segunda rampa (Figura 3 b), ela pode ser utilizada na ordenação secundária. Dessa forma, é possível efetuar *roll-in* de vagões pertencentes a trens unitários para algumas linhas de classificação enquanto a ordenação secundária ocorre em outras linhas.

Existem outras variações no *layout* de um pátio de classificação com rampa que podem gerar pequenas alterações nas características operacionais, o que não muda o princípio essencial da classificação.

2.8. Objetivos e Restrições

O principal objetivo da otimização de um processo *multistage sorting* é minimizar seu tempo total. Esse tempo é composto dos tempos das operações de *roll-in* e *pull-out* executadas. Efetuar um *pull-out* de uma linha de classificação leva um tempo c_{pull} determinado pela distância que o motor de manobras deve percorrer no pátio. O tempo de *roll-in* dos vagões é proporcional ao seu número e depende de um tempo c_{push} necessário para desacoplar e empurrar um único vagão. Um processo *multistage sorting* contendo h operações de *pull-out* e um total de r operações de *roll-in* requer um tempo aproximado de $hc_{pull} + rc_{push}$. Portanto, ao aumentar o valor de h é possível reduzir r e vice-versa. Para um pátio de classificação típico, entretanto, o primeiro atributo dessa soma (hc_{pull}) domina o segundo, assim o principal objetivo do processo é reduzir a quantidade de *pull-outs* (h) (MAUE, 2011).

Outra prática comum é considerar o número total de *roll-ins* (r) como um objetivo. Em alguns pátios esse número é utilizado para determinar o valor que será cobrado pelo processo de classificação.

A principal restrição do processo de classificação busca minimizar a quantidade de linhas de classificação disponíveis. Entre as linhas de classificação existe um número de linhas reservadas para o *mutistage sorting* na prática, enquanto as demais linhas são utilizadas em outras atividades de ordenação (por exemplo, *single-stage sorting* para trens unitários). Um exemplo de como trabalhar com um número limitado de linhas de classificação foi descrito por Krell (1962 *apud* MAUE, 2011). O autor também traz informações sobre a quantidade de grupos que podem ser ordenados, sem considerar a ordem de entrada dos vagões no pátio. Requisitos de métodos *multistage sorting* referentes ao número de linhas de classificação são mencionados em (PENTINGA, 1959 *apud* MAUE, 2011, p.14) (SIDDIQEE, 1972 *apud* MAUE, 2011, p.14) (DAGANZO, DOWLING e HALL, 1983). Uma abordagem precisa para encontrar um plano *multistage sorting* considerando um número limitado de linhas pode ser encontrado em (HANSMANN e ZIMMERMANN, 2008).

Outra restrição existente é o comprimento das linhas de classificação C, dado pelo maior número de vagões que pode ser acomodados na linha. Essa restrição é tratada em (PETERSEN, 1977), na qual o congestionamento de linhas causado pela sua capacidade restrita é considerado para o *single-state sorting*. Para os métodos *multistage sorting*, existem

certos requisitos de capacidade que não são explorados de forma precisa em (BOOT, 1957 *apud* MAUE, 2011, p.14) (PENTINGA, 1959 *apud* MAUE, 2011, p.14) (SIDDIQEE, 1972 *apud* MAUE, 2011, p.14). Uma abordagem para tratar diversos trens de saída como um só, denominada de **formação de comboios,** é introduzida em (DAGANZO, 1986). Essa abordagem busca nivelar o número necessário de linhas de classificação e sua ocupação na ordenação primária e secundária.

Além dessas duas restrições supracitadas podem existir ainda mais restrições operacionais ou de uma infraestrutura específica que afetem o processo de classificação. Por exemplo, no *layout* avançado da Figura 3 (b), se a formação de um trem de saída ocorrer a partir da rampa secundária (por exemplo, em um processo *multistage sorting*), as linhas de classificação utilizadas para essa formação não estarão mais acessíveis para nenhum vagão que estiver na rampa primária.

2.9. Resumo de trabalhos relacionados

Maue (2011) proporcionou uma visão de diferentes autores sobre o problema de classificação de trens considerando aspectos algorítmicos. O quadro a seguir mostra uma visão consolidada dos trabalhos referenciados, relacionados ao método *multistage sorting*.

Quadro 1–Visão consolidada dos trabalhos relacionados

Referência	**Título**	**Contribuição**
(DAGANZO, 1986)	(Static Blocking at Railyards: Sorting Implications and Track Requirements)	Descreveu os métodos de ordenação por trem, ordenação por bloco (ordenação simultânea), ordenação triangular e definiu uma abordagem para formação de comboios.
(DAGANZO, DOWLING e HALL, 1983)	(Railroad classification yard throughput: The case of multistage triangular sorting)	Analisou três estratégias para o *multistage sorting*: ordenação por bloco (ou simultânea), ordenação por trem e ordenação triangular com foco no tempo de serviço (utilizando equações exatas ou aproximações). Concluiu que a estratégia de ordenação triangular, que permite mais classificações em um conjunto de linhas,

		não requer um tempo de serviço significativamente maior em pátios planos.
(DAHLHAUS, HORAK, *et al.*, 2000)	(The train marshalling problem)	Mostrou que o problema de ordenação de trens (*train marshalling*) é NP-completo considerando que a ordenação dos trens é dada por uma sequência de vagões com mesmo destino e que a informação disponível é o número de vagões n.
(DAHLHAUS, MANNE, *et al.*, 2000)	(Algorithms for Combinatorial Problems Related to Train Marshalling)	Analisou o problema de classificação de trens em um pátio com rampa baseado no método de ordenação *radix sort*. O intuito foi minimizar a quantidade de passos de ordenação (*roll-in e pull-out*) e para isso analisou como definir os requisitos de ordenação para os trens de saída utilizando uma árvore P-Q. Em geral, o problema é NP-completo. São discutidos dois casos que podem ser resolvidos de forma eficiente: abordagens: caso de um nó Q com filhos nós P e o caso de nó P com filhos nós Q.
(GATTO, MAUE, *et al.*, 2009)	(Shunting for Dummies: An Introductory Algorithmic Survey)	Definiu um algoritmo eficiente com tempo de execução linear para resolver cenários reais de ordenação em um pátio com rampa utilizando método *multistage sorting* com um número limitado de linhas.
(MÁRTON, MAUE e NUNKESSER, 2009)	(An Improved Train Classification Procedure for the Hump Yard Lausanne Triage)	Combinou programação inteira com uma ferramenta de simulação para validar a codificação de planos de classificação em cenários reais do pátio de Lausanne, na Suíça.
(MAUE, 2011)	(On the Problem of Sorting Railway Freight Cars: An Algorithmic	Apresentou os principais métodos utilizados para solucionar o problema de classificação de trens com uma abordagem algorítmica. Os

	Perspective)	métodos tradicionais não consideram a ordem de chegada dos vagões no pátio, já o método proposto considera essa ordem. O autor detalhou situações na qual essa ordem pode ser não ser cumprida e definiu um algoritmo para tratar o caso mais comum de alteração na ordem de chegada dos vagões, que está associado ao atraso dos trens.
(STEFANO, MAUE, *et al.*, 2007)	(Models for Rearranging Train Cars)	Detalhou diferentes problemas de classificação de trens do ponto de vista teórico.
BOOT, 1957 *apud* MAUE, 2011	(Zugbildung in Holland)	Descreveu as restrições operacionais quando a ordenação simultânea foi introduzida na França, Bélgica e Holanda.
KECKEISEN, 1958 *apud* MAUE, 2011	(Bau und Betrieb der Stuttgarter Hafenbahn – Construção e Operação do Pátio Ferroviário de Stuttgart)	Forneceu um exemplo de como considerar a pré-ordenação dos vagões dos trens de entrada em conjunto com a estratégia de ordenação simultânea básica. O autor mostra como reduzir a quantidade de passos de ordenação e a quantidade de linhas de classificação.
KRELL, 1962 *apud* MAUE, 2011) (KRELL, 1963 *apud* MAUE, 2011	(Grundgedanken des Simultanverfahrens e Ein Beitrag zur gemeinsamen Nutzung von Nahgüterzügen)	Descreveu e comparou as estratégias de ordenação por trem e ordenação simultânea. Além disso, incluiu dois novos métodos para a ordenação simultânea, chamados de ordenação triangular e ordenação geométrica.
SIDDIQEE, 1972 *apud* MAUE, 2011	(Investigation of sorting and train formation schemes for a railroad hump yard)	Resumiu algumas características dos quatro métodos *multistage sorting*: ordenação por trem, ordenação simultânea, ordenação triangular e ordenação geométrica, analisando qual método é mais indicado dependendo da circunstância.

Um fator comum em todos os métodos citados (*single-stage* e *multistage*) é o fato da maioria dos autores não considerarem a ordem na qual os vagões chegam ao pátio, com exceção do exemplo descrito em KECKEISEN, 1958 *apud* MAUE, 2011, p. 41 e da abordagem proposta em (MAUE, 2011). Essa prática produz um processo de ordenação no qual o curso de cada vagão independe de sua posição em relação aos demais vagões de entrada. Entretanto, essa independência gera um desperdício potencial de recursos, podendo acarretar na necessidade de um grande número de linhas de classificação ou de operações de *pull-out*.

Capítulo 3

Otimização Distribuída de Restrições

Esse capítulo apresenta uma visão geral sobre como é modelado um Problema de Otimização Distribuída de Restrições (DCOP) fornecendo uma base teórica para o OMTC concebido para solucionar o problema de classificação de trens.

3.1. Problema de Otimização Distribuída de Restrições

Um Problema de Otimização Distribuída de Restrições é um modelo geral para resolução de problemas distribuídos, que oferece soluções com um grau de qualidade ou custo, principal diferencial em relação a um DSCP (Problema de Satisfação Distribuída de Restrições) (LESSER, ORTIZ e TAMBE, 2003). Isso, permite estender um DSCP garantindo que além de encontrar uma solução considerada ou não "satisfatória", seja possível encontrar soluções que atendam a um critério definido de qualidade. Isso traz uma vantagem quando se procura uma solução para um problema do mundo real, à medida que nesses casos encontrar uma solução ótima pode não ser possível devido aos recursos computacionais existentes. Para definir esse critério de qualidade da solução, os algoritmos DCOP trabalham com uma função de custo (ou objetivo) global, que é distribuída em um conjunto de restrições envolvendo agentes autônomos, e pode ser minimizada ou maximizada.

Como o DCOP trabalha de forma distribuída para encontrar a solução, traz alguns benefícios em relação a um COP (Problema de Otimização de Restrições) como: (i) custo menor de formalização, pois cada agente formula suas restrições de acordo com visão local e de seus vizinhos; (ii) facilidade em trabalhar com a característica dinâmica dos problema práticos, pois permite gerenciar alterações mais facilmente; (iii) maior tolerância a falhas por evitar um ponto centralizado de falhas (LESSER, ORTIZ e TAMBE, 2003).

Um DCOP é composto por agentes, que são responsáveis por determinar um valor para uma ou mais variáveis. Os agentes, além de determinar um valor para suas variáveis, devem comunicar-se entre si para garantir uma solução global ótima para o problema. A solução global ótima é obtida por meio de uma função de custo global, que é a soma dos custos individuais de cada atribuição de variável, mais os custos das relações entre variáveis (que podem ser associadas a diferentes agentes).

Formalmente, um DCOP é composto por n variáveis $V = \{v_1, v_2, \dots, v_n\}$ e cada variável é relacionada a um agente x_i. Uma variável tem um domínio finito e discreto $D_1, D_2, \dots, D_n$ respectivamente. Somente o agente x_i conhece seu domínio D_i e pode atribuir um valor para sua variável v_i. Cada agente é, portanto, responsável por definir um valor d_i para todas as variáveis associadas ao mesmo, onde $d_i \in D_i$, e a escolha do valor da variável deve buscar minimizar seu custo unário. Além desse custo unário, a escolha dos valores das variáveis deve ser coordenada entre os agentes para que o valor da função custo (ou objetivo) global seja minimizado (MODI, SHEN, *et al.*, 2003).

A definição da função de custo global leva em consideração as restrições que um agente impõe em seu agente vizinho. Essa relação de restrição entre dois agentes é definida pela função de custo binária, dada por $f_{ij}: D_i x\ D_j \rightarrow N$, para um par de variáveis v_i, v_j. Os dois agentes x_i e x_j são considerados vizinhos quando existir alguma restrição entre as suas variáveis.

O DCOP procura um conjunto de atribuições de variáveis $A^* = \{d_1, d_2, \dots, d_n \,|d_1 \in D_1,\ d_2 \in D_2, \dots, d_n \in D_n\}$ que minimiza o custo acumulado F, como descrito em (MAILLER e LESSER, 2004). A função de custo global é dada pela Equação (5):

$$F(A) = \sum_{x_i, x_j \in V} f_{ij}(d_i, d_j)\,, onde\ x_i \coloneqq d_i, x_j \coloneqq d_j\ em\ A \tag{5}$$

Um DCOP é interessante quando pretende-se representar problemas reais que não podem ou não devem ser resolvidos de maneira centralizada.

3.2. Algoritmos para DCOP

Diversos algoritmos distribuídos já foram propostos para solucionar um DCOP, podendo ser classificados como síncronos, assíncronos ou parcialmente síncronos, de acordo com o modelo de tempo utilizado, seguindo a classificação proposta por (LYNCH, 1996).

No modelo síncrono, os componentes, no caso os agentes, executam ações de forma simultânea. Portanto, algoritmos DCOP síncronos trabalham de forma determinística, o que não permite explorar o paralelismo possível com a distribuição do problema. Já no modelo assíncrono, os agentes executam seus ciclos em ordens e velocidades arbitrárias. Não há uma sincronização entre o ciclo de execução de cada agente, portanto esses algoritmos que seguem esse modelo trabalham de forma não determinística. Segundo Lynch (1996), o modelo assíncrono pode não resolver os problemas de maneira eficiente e inclusive pode não conseguir chegar a uma solução para o problema. Além disso, como nesse modelo os agentes devem tomar decisões com base em uma visão local do problema, o mecanismo de busca se torna mais complexo. No modelo parcialmente síncrono existe um controle na execução dos ciclos dos agentes, porém o mesmo não é tão rígido como no modelo síncrono (LYNCH, 1996).

Entre os algoritmos assíncronos, o ADOPT (*Asynchronous Distributed Optimization*), proposto por Modi *et. al*. 2003, destaca-se por ser o primeiro método assíncrono completo, conhecido por oferecer garantia de qualidade. Ele é considerado completo pois possui garantia de encontrar uma solução com grau de qualidade caso a mesma exista. Apesar disso, a busca por essa solução pode gerar um custo computacional e de comunicação muito alto, impossibilitando atingir a mesma com recursos computacionais limitados. Uma das desvantagens desse algoritmo é o alto número de ciclos e mensagens trocadas em algumas situações. Ele é considerado correto, segundo a definição de (TSANG, 1993), que considera um algoritmo correto se os resultados gerados são realmente soluções para o problema.

O algoritmo ADOPT utiliza uma pseudo-árvore para priorizar os agentes em relação a definição de sua variável. Essa pseudo-árvore define, uma ordem de prioridade a partir da qual o algoritmo efetua a busca, utilizando uma estratégia oportunista. Cada agente mantém a escolha do melhor valor baseado em sua visão local. Essa visão é posteriormente propagada aos demais agentes, de acordo com a prioridade, para que seja então obtida uma solução

global com menor custo. O mecanismo de busca do ADOPT utiliza uma busca assíncrona em profundidade com um conceito de reconstrução de soluções abandonadas, além de um mecanismo de detecção de término que oferece garantia de qualidade para as soluções encontradas (MODI, SHEN, *et al.*, 2003). Abaixo é apresentado o pseudo-código do algoritmo, conforme apresentado em (MODI, SHEN, *et al.*, 2003). Uma extensão desse trabalho foi apresentada em (MODI, SHEN, *et al.*, 2005), contendo exemplos e experiências adicionais.

Algoritmo 1: ADOPT

Entrada: valores de domínio para o agente D_i, funções de custo δ, pseudo-árvore

Saída: valores de variáveis definidas para cada agente d_i, minimizando a solução global em relação as funções de custo fornecidas

```
#Currentvw: informação sobre o valor das variáveis do
agente em relação ao seu pai
#xi: agente
#di: valor atual da(s) variável(eis) de xi
#c(d): limite inferior atual de custo para a sub-árvore do
xi
```

```
1   Inialize: Currentvw ← {}; di ← null;
2     ∀d ∈ Di:
3       c(d) ← {}
4       hill_climb;
4   when received (VALUE, (xj, dj))
5     add(xj, dj) to Currentvw;
      # contexto alterado
6     if Currentvw changed then
7     ∀d ∈ Di:
8       c(d) ← 0
9     end if;
10    hill_climb;
11  when received (VIEW, vw, cost)
12    d ← value of xi in vw
13    if vw is compatible with Currentvw ∪ {(xi, d)} then
14        c(d) ← max(c(d), cost);
15        if c(d) changed then
16          hill_climb;
17        end if;
18    end if;
```

```
19  procedure hill_climb
20    ∀d ∈ D_i:
        # e(d) é o custo estimado de x_i se ele escolher d
21      e(d) ← δ(x_i, Currentvw ∪ {(x_i, d)}) + c(d);
22    choose d that minimizes e(d)
23      prefer current value d_i for tie;
24      d_i ← d;
25    SEND (VALUE, (x_i, d_i), childLimit) para todos os filhos
      # envia VIEW para o melhor valor d
26    SEND (VIEW, viewContext, e(d)) para pai
```

Fonte: Adaptado de (MODI, SHEN, *et al.*, 2003)

Uma visão mais detalhada do funcionamento do algoritmo e todas as mensagens e variáveis utilizadas é fornecida em (MODI, SHEN, *et al.*, 2003) e (MODI, SHEN, *et al.*, 2005).

O algoritmo DPOP, também correto e completo, é baseado em uma abordagem de programação dinâmica. Ele é uma evolução do algoritmo DTREE (*Distributed Tree Optimization*), proposto em (PETCU e FALTINGS, 2005). Ele utiliza uma pseudo-árvore para representar as restrições do problema, assim como o ADOPT. O DPOP requer uma quantidade linear de mensagens trocadas pelos agentes, que está diretamente relacionada a largura da pseudo-árvore. A complexidade desse algoritmo está relacionada ao tamanho das mensagens trocadas, que, segundo Petcu e Faltings (2005), pode gerar uma mensagem muito grande no pior caso, exigindo grande quantidade de memória.

O algoritmo DPOP é iniciado com a priorização dos agentes em uma pseudo-árvore. A partir dessa árvore, os agentes folha propagam mensagens UTIL para seus pais, até chegar ao nó raiz, ou seja, um processo *bottom-up*. Um agente só propaga essa mensagem após receber as mensagens UTIL de todos os seus filhos. Sempre que um agente envia essa mensagem para seu pai, a mesma contém o cálculo da utilidade (ou custo) obtido na sub-árvore do agente. Portanto, essa mensagem contém todas as utilidades ótimas que podem ser alcançadas para os agentes da sub-árvore, calculadas para os valores possíveis do agente que está propagando a mensagem. Com isso, temos uma mensagem que pode se tornar muito grande de acordo com a largura da pseudo-árvore.

Após receber a mensagem UTIL, o nó raiz é responsável por calcular o valor ótimo para si própria, a partir da mensagem UTIL recebida e também do cálculo do custo unário

para seus valores possíveis. Assim que determinar esse valor, o agente do nó raiz inicia um processo *top-down* que envia mensagens VALUE para seus descendentes. Abaixo segue o pseudo-código do DPOP.

Algoritmo 2: DPOP

Entrada: agentes *X*, valores de domínio para os agentes *D*, função de utilidade e as restrições entre agentes *R*

Saída: valores de variáveis definidas para cada agente d_i, minimizando a solução global em relação as funções de custo fornecidas

$P(X)$ - pai do nó X
$C(X)$ - filhos do nó X
$PP(X)$ - pseudo-pais do nó X
$PC(X)$ - pseudo-filhos do nó X

```
1   DPOP (X, D, R)
    Cada agente X_i executa:
2
3   Fase 1: geração da pseudo-árvore
4   escolha um líder para todo X_j ∈ X
5   o líder escolhido inicia a geração da pseudo-árvore
6   X_i conhece P(X_i), PP(X_i), C(X_i) e PC(X_i)
7   Fase 2: propagação da mensagem UTIL
8   if |Children(X_i)| == 0 (ex.: X_i é nó folha) then
9     UTIL_{X_i}(P(X_i)) ← Compute_utils(P(X_i), PP(X_i))
10    Send_message (P(X_i), UTIL_{X_i}(P(X_i)))
11  activate UTIL_Message_handler()
12  Fase 3: propagação da mensagem VALUE
13  activate VALUE_Message_handler()
14  END ALGORITHM
15
16  UTIL_Message_handler(X_k, UTIL_{X_k}(X_i))
17  store UTIL_{X_k}(X_i)
18  if UTIL messages from all children arrived then
19    if Parent(X_i) == null (X_i é a raiz) then
20      v_i^* ← Choose_optimal(null)
21      Send VALUE (X_i, v_i^*) to all C(X_i)
22    else
23      UTIL_{X_i}(P(X_i)) ← Compute_utils(P(X_i), PP(X_i))
24      Send_message(P(X_i), UTIL(P(X_i)))
25  return
```

```
26
27 VALUE_Message_handler(VALUE^{X_i}_{P(X_i)})
28 add all X_k ← v*_k ∈ VALUE^{X_i}_{P(X_i)} to agente_view
29 X_i ← v*_i = Choose_{optimal(agent_view)}
30 Send VALUE^{X_l}_{X_i} to all X_l ∈ C(X_i)
31
32 Choose_optimal(agent_view)
33   v*_i ← argmax_{v_i} Σ_{X_l∈C(X_i)} UTIL_{X_l}(v_i, agent_view)
34 return v*_i
35
36 Compute_utils(P(X_i), PP(X_i))
37 for all combinations of values of X_k ∈ PP(X_i) do
38   let X_j be Parent(X_i)
39   similarly to DTREE, compute a vector UTIL_{X_i}(X_j)
     of all {Util_{X_i}(v*_i(v_j), v_j) | v_j ∈ Dom(X_j)}
40 assemble a hypercube UTIL_{X_i}(X_j) out of all these
   vectors (totalling |PP(X_i)| + 1 dimensions).
41 return UTIL_{X_i}(X_j)
```

Fonte: Adaptado de (PETCU e FALTINGS, 2005)

Uma descrição completa sobre o algoritmo DPOP pode ser obtida em (PETCU e FALTINGS, 2005).

3.3. Considerações do Capítulo

A modelagem de um problema do mundo real como DCOP consiste principalmente na definição dos valores possíveis para as variáveis de cada agente, suas restrições e nas funções de custo. Portanto, a complexidade do modelo está associada em como escolher esses agentes, suas variáveis, como definir a relação entre os agentes e também a função de custo a utilizar. Uma outra característica de um DCOP que deve ser considerada é o grau de interdependência entre os agentes, já que os algoritmos têm uma abordagem distribuída.

Utilizar uma abordagem DCOP é especialmente apropriado quando pretende-se representar problemas reais que não podem ou não precisam ser resolvidos de maneira centralizada. Dentro do contexto do problema proposto, que é a classificação de trens em um pátio ferroviário, foi possível identificar uma divisão do problema entre os trens de saída

formados. Conforme descrito na seção 2.7, os vagões só possuem relação com demais vagões pertencentes ao mesmo trem de saída. Portanto, nesse caso, utilizando um DCOP, será possível designar um agente por trem de saída, garantindo uma solução distribuída.

Optou-se por efetuar a validação inicial do modelo utilizando somente dois algoritmos: ADOPT e DPOP. O algoritmo ADOPT usa uma abordagem de busca gulosa e requer uma quantidade menor de memória, porém maior tempo de processamento. O algoritmo DPOP usa uma abordagem baseada em programação dinâmica e gera um maior consumo de memória, porém requer um menor tempo de processamento. Portanto, temos dois algoritmos que se comportam de forma diferenciada em relação ao consumo de memória e tempo de processamento. Com isso, pretende-se verificar como esses algoritmos atuam em cenários fictícios construídos para determinar o algoritmo mais indicado à execução dos experimentos baseados em dados reais. Ambos os algoritmos são conhecidos e utilizados na comunidade de pesquisa.

Capítulo 4

Modelo para Otimização da Classificação de Trens

O presente trabalho modelou o problema de classificação de trens como um DCOP, que, conforme descrito anteriormente, utiliza o conceito de programação por restrições aliado a uma abordagem distribuída baseada em agentes autônomos. Essa modelagem foi denominada Modelo para Otimização da Classificação de Trens (OMTC), utilizou as restrições de viabilidade de plano de classificação obtidas a partir da codificação já apresentada, adequando-as a um modelo de restrições que possa ser utilizado por um algoritmo de resolução de um DCOP.

As seções desse capítulo apresentam o OMTC. Para isso, inicialmente é fornecida uma visão do algoritmo *Greedy Chain Decomposition* (GCD), definido por de Maue (2011), responsável por fornecer a decomposição de cadeia para um cenário de classificação de trens e que permite derivar um plano de classificação viável e ótimo. A seguir é apresentado como essa decomposição de cadeia pode facilitar a definição das restrições que garantem a viabilidade de um plano de classificação. Por último, é descrito uma extensão do OMTC para incluir as restrições de capacidade e quantidade de linhas de classificação. Essa extensão levou a um processo iterativo para geração do plano de classificação, chamado de DCOP Iterativo (I-DCOP).

4.1. Algoritmo *Greedy Chain Decomposition*

Para geração automática de um plano de classificação ótimo, foi proposto o conceito de cadeia e de decomposição de cadeia de vagões. De forma simplificada, uma cadeia abrange os vagões de tipos consecutivos (ordem crescente) pertencentes a um mesmo trem de saída, que chegaram ao pátio já na ordem correta. Portanto, identificando essas cadeias, que na

verdade representam vagões que não precisam "trocar" de lugar para serem designados a seus respectivos trens de saída, é possível gerar um plano de classificação viável. A seguir, será descrito brevemente o funcionamento de uma decomposição de cadeia.

A representação de uma cadeia é feita utilizando a notação $[\tau_1, \tau_{11}]$. Essa notação representa uma sequência de vagões, composta somente por vagões que chegaram ao pátio após τ_1 e que possuem o mesmo tipo que τ_1 ou um tipo maior que o tipo do vagão τ_1 e menor ou igual ao tipo de τ_2, excluindo da sequência somente o próprio vagão τ_2. Para uma sequência de vagões ser considerada uma cadeia, ela deve conter somente vagões de tipos consecutivos em ordem crescente. Considerando a sequência de vagões de entrada da Figura 5, é possível identificar a sequência definida por $[6, 8_2]$ como uma cadeia que contém todos os vagões dos tipos 6 e 7, porém nenhum vagão do tipo 8. Como outro exemplo, a sequência $[12, 13_1] = (12,13_2)$ apresenta uma cadeia contendo apenas o vagão 13_2, mas não 13_1 do tipo maior do intervalo. Já a sequência $[6, 8_1] = (6, 8_2, 7)$, não pode ser considerada uma cadeia, pois os vagões não estão na ordem crescente de tipo. É importante destacar que uma cadeia nunca é composta de vagões pertencentes a mais de um trem de saída, pois para vagões de trens de saída diferentes não existe nenhuma dependência na geração de planos de classificação viáveis. Uma cadeia representa portanto, uma sequência de vagões que já chegaram na ordem correta ao pátio, e se esses vagões seguirem as mesmas operações dentro do pátio, eles permaneceram ordenados em seus trens de saída.

A quantidade gerada de cadeias para uma determinada sequência de vagões de entrada está relacionada ao grau de pré-ordenação dos vagões. Portanto, uma sequência de entrada que já está previamente ordenada irá gerar menos cadeias e as mesmas em contrapartida, irão conter mais vagões. Consequentemente, um plano de classificação para uma sequência de vagões de entrada "mais pré-ordenada" necessitará de menos passos de ordenação.

Uma sequência de índices (índice 1 representa o primeiro vagão da sequência de entrada, índice 2 o segundo e assim sucessivamente) é chamada de decomposição de cadeia se gerar sequências de trens que são cadeias e se todo vagão da sequência de entrada pertencer a uma sequência. No exemplo da Figura 5, que contém $n = 17$ vagões na sequência de entrada e $m = 2$ trens de saída, uma decomposição da cadeia é dada pela seguinte sequência de índices: $16, 12, 11, 10, 9, 7, 13, 3, 2, 15, 8, 4, 1, 18$. Esses índices correspondem a vagões que pertencem a diferentes trens de saída e o índice 18 foi incluído para gerar a cadeia final, mas

não pertence a nenhuma cadeia, e pode ser considerado como se um vagão "*fake*" para um trem de saída adicional (m_3). A partir desses índices é possível gerar as cadeias descritas abaixo:

Cadeias do trem de saída m_1 ($k_1 = 9$)

$[\tau_{16}, \tau_{12}] = [1,2] = (1)$
$[\tau_{12}, \tau_{11}] = [2,3] = (2)$
$[\tau_{11}, \tau_{10}] = [3,4] = (3)$
$[\tau_{10}, \tau_{9}] = [4,5] = (4)$
$[\tau_{9}, \tau_{7}] = [5,6] = (5)$
$[\tau_{7}, \tau_{13}] = [6,8_2] = (6,7)$
$[\tau_{13}, \tau_{3}] = [8_2, 9] = (8_1, 8_2)$
$[\tau_{3}, \tau_{2}] = [9,10] = (9)$
$[\tau_{2}, \tau_{15}] = [10,11] = (10)$

Cadeias do trem de saída m_2 ($k_2 = 4$)

$[\tau_{15}, \tau_{8}] = [11,12] = (11)$
$[\tau_{8}, \tau_{4}] = [12,13_1] = (12, 13_2)$
$[\tau_{4}, \tau_{1}] = [13_1, 14] = (13_1)$
$[\tau_{1}, \tau_{18}] = [14,16] = (14,15)$

Na decomposição de cadeia, k_i indica a quantidade de cadeias gerada a partir da decomposição para o trem de saída m_i. Portanto, para a sequência de exemplo temos $k_1 = 9$ e $k_2 = 4$. O tamanho da decomposição de cadeia, chamado de k, é dado por $\max(k_1, k_2) = \max(9,4) = 9$.

Porém, gerar uma decomposição de cadeia válida não garante ainda um plano de classificação ótimo. Em sua pesquisa, Maue (2011) comprovou por teoremas matemáticos, que a partir da menor decomposição de cadeia (ou seja, menor k) para uma sequência de vagões de entrada é possível obter um plano de classificação viável e ótimo, ou seja, que possui a menor quantidade *h* de passos de ordenação. O Lema 3, que é base para essas conclusões, define basicamente que vagões da mesma cadeia podem receber a mesma *bitstring*, ou seja, eles podem executar a mesma sequência de operações dentro do pátio. Isso pode ser intuitivamente verificado, pois indica que os vagões que já chegaram em ordem

correta ao pátio, seguindo o mesmo "caminho", ficarão ordenados corretamente no trem de saída. Já para cadeias diferentes, os vagões da cadeia seguinte da decomposição devem receber uma *bitstring* de valor maior, exceto quando a cadeia contiver vagões de outro trem de saída. Essa restrição garante que o plano gerado seja viável. Aplicando essa regra na decomposição da cadeia mostrada anteriormente é gerado um plano viável para a sequência de entrada de vagões.

O autor demonstra ainda, com seu Teorema 2, que é possível derivar um plano ótimo a partir da regra proposta pelo Lema 3, considerando que ótimo é o plano que minimiza a quantidade h de passos de ordenação. O tamanho h do plano de classificação ótimo para a sequência de vagões de entrada pode ser calculado por $log_2(k-1)$. O mesmo tamanho h pode ser obtido pelo número de *bits* da maior *bitstring* designada no algoritmo proposto para gerar um plano viável e ótimo, denominado de *Greedy Chain Decomposition*. Foi mostrado que quanto menor o tamanho da decomposição da cadeia, maior o grau de pré-ordenação da sequência de entrada. De fato, uma decomposição de cadeia mais curta gera um plano ótimo.

A seguir, é apresentado o pseudo-código do algoritmo GCD, já contemplando a adaptação para geração do plano de classificação. Essa adaptação inclui a atribuição da *bitstring* para cada vagão. Foi efetuada uma alteração na 12ª linha para evitar que o algoritmo atribua *bitstring* 0 para os vagões. Isso foi necessário para evita-se que vagões sejam direcionados diretamente para a linha de saída, o que é uma característica operacional do pátio onde os dados utilizados nos experimentos foram obtidos.

É importante destacar que, para o algoritmo atribuir corretamente as *bitstrings* aos vagões, ele precisa receber como entrada as seguintes informações: (i) total de vagões da sequência de entrada, representado por n; (ii) a sequência de entrada de vagões na ordem de chegada ao pátio, representada por uma sequência de tipos $\tau = \tau_1, \dots, \tau_n$; (iii) a quantidade de trens de saída, representada por m; e, (iv) o valor do maior tipo de cada trem, representado por $G^1, \dots, G^m$. Conforme já mencionado, os tipos dos vagões indicam grupos que devem permanecer juntos nos trens de saída, representando uma abstração para identificar qualquer critério para deixar vagões agrupados. Além disso, esses tipos não se repetem em trens de saída diferentes, e é assumido que existe ao menos um vagão de cada tipo. Com isso, o algoritmo consegue somente a partir dos maiores tipos de cada trem de saída, identificar quais tipos de vagões pertencem a cada trem de saída.

Algoritmo 3: *Greedy Chain Decomposition*

Entrada: número de vagões n, sequência de entrada de vagões $\tau = \tau_1, \dots, \tau_n$, número de trens de saída m com seus respectivos maiores tipo $G^1, \dots, G^m$.

Saída: índices $y_1^1..y_{k_1}^1, y_1^2..y_{k_2}^2, \dots, y_1^m..y_{k_m}^m, y_1^{m+1}$, que representam a decomposição da cadeia.

```
     # inicializa o contador de vagões de cada
     # tipo com 0
1    for (t = 1, ..., G) do
2      set count[t] =0
     # conta quantos vagões existem para cada tipo
3    for (x = 1, ..., n) do
4      count[τ_x] ++
5    set t = 1  # inicia buscando o tipo 1
     # para cada trem de saída
6    for (k = 1, ..., m) do
7      set i = 0
       # vai procurar vagões do tipo t, que iniciou em 0,
       até chegar ao maior tipo do trem de saída atual
8      while (t ≤ G^k) do
9        set first = TRUE
10       i ++
         # para cada vagão da sequência de entrada
11       for (x = 1, ..., n) do
12         if (τ_x = t) then
             # alterado para não considerar bitstring 0
             # original → b^{τ_x} := [i − 1]_2
12a          b^{τ_x} := [i]_2
13           count[t] − −
14           if (count[t] = 0) then
15             if (first = TRUE) then
16               put y_i^k = x
17             set first = FALSE
18             t ++
19             if (t > G^k) then
20               break
21   put y_1^{m+1} = n + 1
22   return y_1^1..y_{k_1}^1, y_1^2..y_{k_2}^2, ..., y_1^m..y_{k_m}^m, y_1^{m+1}
```

Fonte: Adaptado de (MAUE, 2011)

Aplicando o Algoritmo 3, conforme definido acima, no cenário descrito pela Figura 5 é gerado o seguinte plano de classificação por tipo de vagão:

```
Trem de saída 1:
1: 0001
2: 0010
3: 0011
4: 0100
5: 0101
6: 0110
7: 0110
8: 0111
9: 1000
10: 1001

Trem de saída 2:
11: 0001
12: 0010
13: 0011
14: 0100
15: 0100
```

O plano gerado é viável e ótimo, de acordo com o conceito definido por Maue (2011), porém não considera outras restrições como: quantidade de *roll-ins*, quantidade limitada de linhas de classificação, capacidade das linhas, etc. Para considerar demais restrições Maue (2011) utilizou uma abordagem de programação inteira com aproximações.

A decomposição de cadeia, fornecida pelo algoritmo *Greedy Chain Decomposition*, foi utilizada no OMTC para garantir a geração de um plano de classificação viável, conforme detalhado na próxima seção.

4.2. Descrição do OMTC

Conforme já descrito, a representação de um plano de classificação permite identificar a viabilidade do mesmo por meio de duas restrições envolvendo as *bitstrings*. Para definição dessas restrições é necessário considerar vagões de tipos consecutivos, denominados τ_x e τ_{x+1}, pertencentes ao mesmo trem de saída:

(i) τ_{x+1} nunca pode receber uma *bitstring* **menor** que a do vagão τ_x (o que garante que os vagões permaneçam ordenados corretamente); e

(ii) se τ_{x+1} chegou ao pátio antes de τ_x, então τ_{x+1} deve receber uma *bitstring* necessariamente **maior** que τ_x (o que garante que os vagões troquem de posição).

A partir das cadeias fornecidas pelo algoritmo GCD, já é possível identificar vagões pertencentes ao mesmo trem de saída, que possuem tipos consecutivos e chegaram na ordem correta ao pátio. Esses vagões são agrupados na mesma cadeia. Os vagões da mesma cadeia devem seguir a restrição definida em (i), que garante que esses vagões permaneçam ordenados corretamente no trem de saída. Isso significa que para as cadeias contendo mais de um vagão, será necessário incluir a restrição (i) entre esses vagões no OMTC. Já a restrição (ii) deve ser aplicada aos vagões de cadeias consecutivas, ou seja, todos os vagões da cadeia 2 do trem 1, por exemplo, devem receber uma *bitstring* obrigatoriamente maior do que o último vagão da cadeia 1 do mesmo trem.

Como descrito na seção 3.1, o DCOP de forma resumida, modela o problema de acordo com um conjunto de variáveis e uma função de custo global baseada nas restrições entre essas variáveis. Além disso, as variáveis possuem a um domínio finito e discreto.

Relacionando os conceitos do DCOP com o problema apresentado no Capítulo 2 e incluindo as restrições (i) e (ii), descritas no início da seção e que garantem a viabilidade do plano, foi definido o OMTC:

- **Domínio:** o domínio de valores possível pode ser definido pela quantidade de passos de ordenação necessários para realização do processo de classificação. Utilizando o conceito de decomposição de cadeia, foi possível definir a menor quantidade de passos necessária para solução de uma instância de problema de classificação de trens. O algoritmo GCD, fornece a decomposição de cadeia para o problema, e por meio dessa decomposição foi possível obter o tamanho mínimo do plano:

$$h_{min} \coloneqq \max_{j=1\ldots m} \{k_j\} \tag{6}$$

onde m representa a quantidade de trens de saída, e k_j a quantidade de cadeias geradas para cada trem.

A quantidade de passos de ordenação, define a quantidade de *bits* necessária para representação do plano de classificação ótimo, no qual o domínio de valores varia

de 0 até a *bitstring* de tamanho h_{min} com todos os *bits* = 1. Essa informação permite, portanto, definir o domínio das variáveis do modelo pelo intervalo entre 0 e $2^{h_{min}}$. Devido a uma característica operacional do pátio a partir do qual os dados reais foram gerados, que determina que todos os vagões devem ser direcionados para uma linha de classificação no *roll-in* inicial, o valor 0 foi retirado do domínio. O domínio de valores restringe a quantidade de linhas de classificação utilizadas, que é uma das restrições existentes em um problema de classificação de trens.

- **Agentes:** cada trem de saída que deve ser formado no processo de classificação foi representado como um agente. Esse agente é o responsável por designar um valor para todos os vagões que o compõe.
- **Variáveis:** cada vagão da sequência de entrada definida por $\tau = \tau_1, \dots, \tau_n$ foi representado por uma variável. Na execução do modelo, essa variável recebe um valor decimal, que posteriormente é convertido para a *bitstring* que representa as operações que o vagão deve efetuar no pátio de classificação.
- **Função de custo individual:** a função de custo individual define um custo para todos os valores de domínio das variáveis. Para o problema de classificação de trens esse custo foi associado a quantidade de movimentos que o vagão realiza no pátio. Na codificação do plano utilizada, o *bit* com valor 1 indica que o vagão efetua um movimento de *roll-in*, assim a função de custo individual foi definida pela quantidade de *roll-ins* que cada vagão realiza no pátio, dada por:

$$w(b^{\tau_x}) = \sum_{i=1}^{h} b_i \tag{7}$$

- **Função de custo da relação (ou custo binário) e definição das restrições entre os agentes:** a função de custo binária foi modelada levando em consideração as duas restrições de viabilidade de planos de classificação e apresentadas no início dessa seção.

  A restrição (i) deve ser aplicada aos vagões pertencentes a mesma cadeia. Todo vagão da mesma cadeia (lembrando que dentro de uma cadeia os vagões já estão com os tipos ordenados) deve receber uma *bitstring* maior ou igual ao vagão

anterior a ele na cadeia. Ou seja, se uma cadeia é composta de três vagões, denominados v_1, v_2 *e* v_3, por exemplo, o vagão v_2 deve receber uma *bitstring* maior ou igual a v_1 e $v_3$deve receber uma *bitstring* maior ou igual a v_2.

A restrição (ii), se aplica "entre" cadeias, ou seja, deve ser aplicada entre vagões de cadeias consecutivas do mesmo trem de saída. Isso significa que, o primeiro vagão de uma cadeia, deve obrigatoriamente receber uma *bitstring* maior que todos os vagões pertencentes a cadeia anterior. Para a primeira cadeia do trem de saída, não é necessário definir essa restrição.

Aplicando ambas as restrições é possível garantir a geração de um plano de classificação viável. Para exemplificar e esclarecer como essa função de custo binário é definida segue novamente o cenário de exemplo já descrito na anteriormente (Figura 5), resumido agora na Figura 6, que mostra somente a situação inicial e final dos vagões.

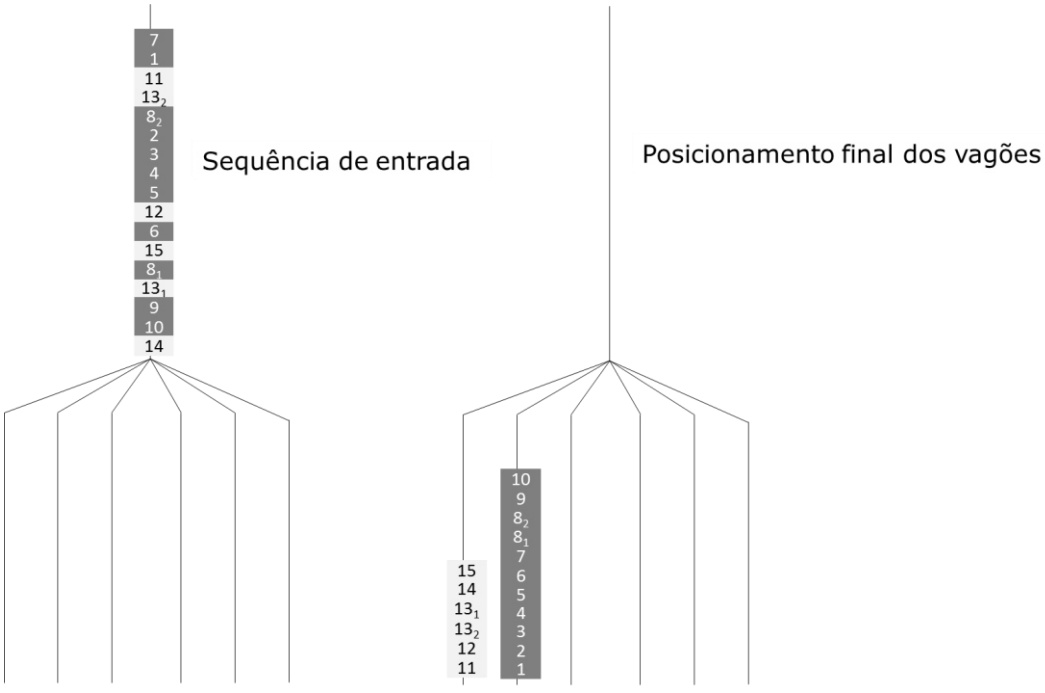

Figura 6 – Cenário exemplo de classificação de trens

Relembrando, o critério definido para correta formação dos trens de saída é que os vagões devem estar ordenados de forma crescente pelo seu tipo, e que vagões do mesmo tipo podem ter ordem arbitrária (portanto, a ordem dos vagões de tipo 8 e 13 podem ser alteradas). Existe ao menos um vagão de cada tipo e todos os

vagões do mesmo tipo devem permanecer no mesmo trem de saída. Além disso, os trens de saída estão diferenciados pelas cores na Figura 6.

Para esse cenário exemplo, temos a decomposição de cadeia fornecida pelo algoritmo GCD que gera as seguintes cadeias:

Cadeias do trem 1 (cinza escuro):

cadeia 1 = (1), cadeia 2 = (2), cadeia 3 = (3), cadeia 4 = (4), cadeia 5 = (5), cadeia 6 = $(6,7)$, cadeia 7 = $(8_1, 8_2)$, cadeia 8 = (9) e cadeia 9 = (10).

Cadeias do trem 2 (cinza claro):

cadeia 1 = (11), cadeia 2 = $(12, 13_2)$, cadeia 3 = (13_1) e cadeia 4 = $(14, 15)$.

A partir dessa decomposição de cadeia, segue as relações entre agentes pertencentes a mesma cadeia, necessárias para aplicação da restrição (i):

Tabela 1 – Relação entre agentes da mesma cadeia – cenário Figura 6

a_i	a_j
6	7
8_1	8_2
12	13_2
14	15

Nesse exemplo, muitas cadeias são compostas somente por um vagão. Esses vagões, únicos dentro de uma cadeia, só terão relação de restrição com vagões pertencentes a próxima cadeia. Para os vagões indicados na Tabela 1, a função de custo binária terá como valor padrão 0, e quando a restrição (i) não for cumprida, terá valor ∞, conforme mostrado pela relação abaixo:

$$f(a_i, a_j):$$

$$\forall\ bitstring_{a_j} < bitstring_{a_i} \rightarrow\ \infty$$

Já a restrição (ii), está relacionada a vagões pertencentes a cadeias distintas, porém vinculados ao mesmo trem de saída. Portanto, existe uma relação de restrição que deve ser definida entre o primeiro tipo de uma cadeia e o último tipo da cadeia anterior. Para o exemplo da Figura 6, os seguintes vagões devem garantir a restrição (ii):

Tabela 2 – Relação entre agentes de cadeias distintas – cenário Figura 6

a_m	a_n
2	1
3	2
4	3
5	4
6	5
8_1	6
9	8_1
9	8_2
10	9
12	11
13_1	13_2
14	13_1

Observe que, se o maior tipo de uma cadeia contiver mais de um vagão, o primeiro vagão da cadeia seguinte terá relação com todos os vagões desse tipo. Isso ocorreu com o vagão 9, que tem uma relação com 8_1 e 8_2. Para os vagões da Tabela 2, será definida função de custo com valor padrão 0, e caso não cumpram a restrição (ii), será atribuído valor ∞:

$$f(a_m, a_n):$$

$$\forall\ bitstring_{a_m} \leq bitstring_{a_n} \rightarrow\ \infty$$

O OMTC apresentado, portanto, garante que a solução fornecida pelo algoritmo DCOP gere um plano de classificação viável e ótimo para um problema de classificação de trens. O custo global da solução, quando diferente de ∞, indica a quantidade de *roll-ins* total do processo. Foi optado por otimizar-se esse valor, pois em alguns pátios de classificação esse número representa um custo financeiro no processo de classificação.

O OMTC não pôde ser estendido para incluir a restrição de capacidade das linhas de classificação. Essa restrição será tratada em maiores detalhes na próxima seção.

4.3. Extensão do OMTC para incluir restrição de capacidade de linha

A restrição de capacidade das linhas de classificação representa a restrição mais importante para o problema de classificação de trens após a otimização do tempo de execução do plano e a limitação da quantidade de linhas de classificação. Maue (2011) propôs um modelo de programação inteira para tratar essa restrição.

A inclusão da restrição de capacidade limitada das linhas de classificação no OMTC gera a necessidade de inclusão de mais um conjunto de restrições ao modelo. Mais importante do que isso, um dos objetivos iniciais da pesquisa era mostrar que a utilização de um modelo baseado na otimização distribuída de restrições permitiria a inclusão de mais restrições ao problema de forma simples. Porém, a inclusão de restrição de capacidade, apesar de simples em sua concepção, resultou na prática na inclusão de uma restrição global ao OMTC.

Isso ocorreu devido a forma de modelar as operações dos vagões no pátio, que foi efetuada utilizando a codificação do plano de classificação. Com o uso dessa codificação, para obter a capacidade C ocupada em cada linha de classificação, é necessário verificar os *bits* 1 na posição da *bitstring* que referencia a linha de ordenação. Ou seja, para garantir uma restrição de capacidade de linha definida por C, seria necessário garantir que no plano de classificação de tamanho h, o peso w de todas as linhas não excedesse a capacidade máxima definida: $w(b_i) \leq C$ para $i = 1, \ldots, h$.

Isso fez com que, mesmo utilizando algoritmos DCOP, que trabalham de forma distribuída, fosse gerada uma dependência entre todas as variáveis do problema. Com isso, a designação de um valor para a variável, acabou tornando-se dependente das demais atribuições de valores de todas as outras variáveis. Na prática, a inclusão da restrição de capacidade fez com o problema fosse resolvido de forma centralizada.

Para permitir a inclusão da restrição de capacidade de linha de ordenação, restrição essencial para validação do problema utilizando cenários baseados em dados reais, foi definido um processo para execução do OMTC de forma iterativa, chamado de DCOP Iterativo (I-DCOP). Esse processo inclui a restrição de capacidade das linhas por meio de ajustes no domínio de valores dos agentes, o que será apresentado na próxima seção.

4.4. Processo para gerar e avaliar o OMTC

Para gerar o OMTC para diferentes cenários e também proporcionar uma verificação mais simples do plano de classificação obido, o processo foi automatizado e seu resultado simulado com o simulador de plano de classificação desenvolvido. Utilizando essa automatização foi possível avaliar o OMTC a partir de três processos: (i) simplificado, no qual se deseja gerar um plano de classificação viável e ótimo; (ii) parcial, no qual além da solução simplificada, é adicionada uma restrição de capacidade das linhas de classificação, considerando-se porém a existência de uma quantidade ilimitada de linhas de ordenação; e (iii) completo, no qual além da solução simplificada, são incluídas a restrição de capacidade e também de quantidade das linhas de classificação. Cada um desses processos é apresentado a seguir.

4.4.1. Processo simplificado

O processo denominado simplificado pode ser utilizado para gerar um plano de classificação viável e ótimo. Nesse processo, além das restrições para garantir viabilidade, foi incluída a otimização da quantidade total de *roll-ins* dos vagões. O plano é obtido diretamente pela execução de algoritmos DCOP para o OMTC gerado. O diagrama da Figura 7 apresenta o funcionamento desse processo:

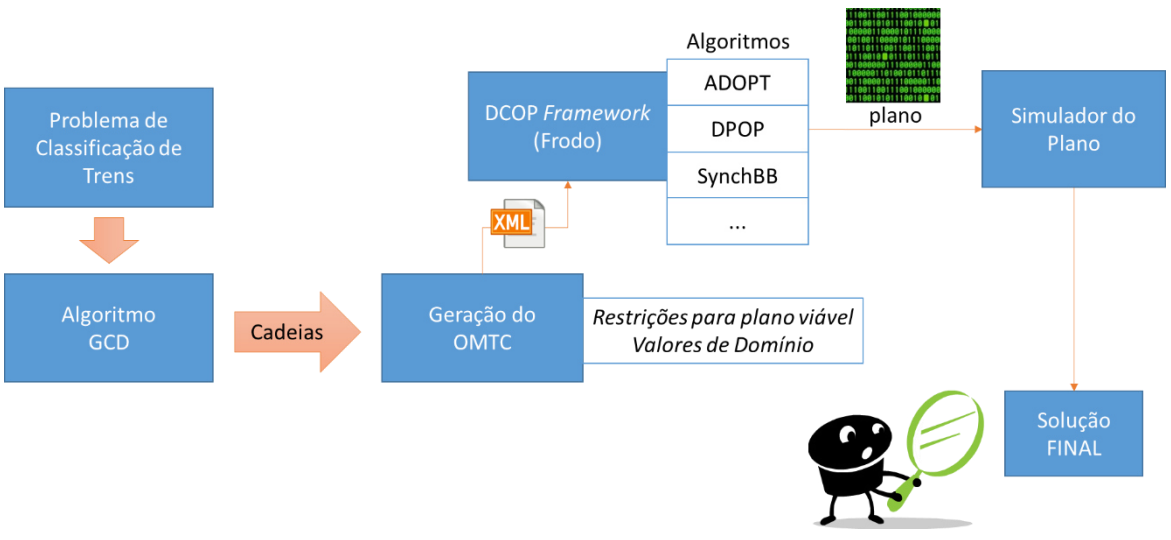

Figura 7 – Processo simplificado

O problema de classificação de trens é configurado da seguinte forma: número de vagões n, sequência de entrada de vagões $\tau = \tau_1, \dots, \tau_n$, número de trens de saída m com seus respectivos maiores tipo $G^1, \dots, G^m$. Esses dados são necessários para execução do algoritmo GCD. O problema é inicialmente encaminhado ao algoritmo GCD, que fornece todas as cadeias geradas a partir da decomposição de cadeia. Essa informação é passada ao módulo chamado de Geração do OMTC, responsável por gerar o OMTC contendo as variáveis, agentes, domínios, restrições e funções de custo em um formato XML, capaz de ser resolvido pelo *framework* FRODO, utilizando diferentes algoritmos para DCOP (LÉAUTÉ, OTTENS e SZYMANEK, 2009).

O *framework* FRODO foi adotado na pesquisa por ser uma ferramenta que permite gerar a solução para um problema formulado como um DCOP utilizando diferentes algoritmos. Além disso, esse *framework* é facilmente integrável aos demais processos por ser *open source* e baseado em na linguagem de programação Java, amplamente utilizada na área de desenvolvimento e pesquisa. Optou-se por utilizar essa ferramenta ao invés de implementar os algoritmos, pois o objetivo da pesquisa é a avaliação do modelo proposto e não dos algoritmos DCOP em si.

Após a geração de um plano de classificação para o cenário informado, esse plano é simulado, permitindo a verificação da formação dos trens de saída após a execução de todos os passos de ordenação do plano.

Para esclarecer o funcionamento desse processo, será utilizado novamente o cenário de exemplo descrito na Figura 5. Esse cenário é configurado da seguinte forma no início do processo: (i) sequência de vagões de entrada: 14, 10, 9, 13, 8, 15, 6, 12, 5, 4, 3, 2, 8, 13, 11, 1, 7; (ii) 2 trens de saída: T1 com maior tipo 10 e T2 com maior tipo 15.

A partir dessa configuração, o processo gera as cadeias para cada trem de acordo com o algoritmo GCD: **cadeias do trem 1:** cadeia 1 = (1), cadeia 2 = (2), cadeia 3 = (3), cadeia 4 = (4), cadeia 5 = (5), cadeia 6 = $(6,7)$, cadeia 7 = $(8_1, 8_2)$, cadeia 8 = (9) e cadeia 9 = (10); **cadeias do trem 2:** cadeia 1 = (11), cadeia 2 = $(12, 13_2)$, cadeia 3 = (13_1) e cadeia 4 = $(14, 15)$.

Com essas informações é montado o arquivo XML contendo: um agente por trem de saída, uma variável por vagão associada ao agente do trem de saída ao qual deve ser direcionado, uma função para cálculo do custo unário (*bits* = 1), restrições de viabilidade do plano (associadas as cadeias geradas) e restrições para cálculo do custo da solução. O domínio

de valores é determinado de acordo com o h mínimo necessário para gerar um plano viável. O arquivo XML gerado pode ser encontrado no Apêndice C. Esse arquivo é então enviado ao *framework* FRODO que gera o plano de classificação como solução, apresentado na Figura 8:

```
h: 4
Sequência de vagões de entrada com as respectivas bitstrings:

14_1: 1000
10_1: 1010
9_1 : 1001
13_1: 0100
8_1 : 1000
15_1: 1000
6_1 : 0110
12_1: 0010
5_1 : 0101
4_1 : 0100
3_1 : 0011
2_1 : 0010
8_2 : 1000
13_2: 0010
11_1: 0001
1_1 : 0001
7_1 : 0110

--- Custo da solução: 23 ---
```

Figura 8 – Plano de classificação gerado pelo OMTC

O processo recupera a solução e inicia a simulação do plano, passo a passo, seguindo as *bitstrings* fornecidas no plano gerado pelo OMTC. O Apêndice D mostra as telas dessa execução, passo a passo. A simulação do plano de classificação considera a representação do mesmo utilizando as *bitstrings*, que foram geradas pelo OMTC criado e executado pelo *framework* FRODO. O processo simplificado considera capacidade e quantidade de linhas de ordenação ilimitadas.

4.4.2. DCOP Iterativo parcial

O DCOP Iterativo (I-DCOP) parcial diferencia-se do processo simplificado, pois introduz a restrição de capacidade limitada para as linhas de ordenação. Essa informação é definida na configuração do pátio utilizada pelo simulador e é inserida dentro de um processo

iterativo que reduz o domínio das variáveis dos vagões que violam essa restrição. A Figura 9 apresenta o funcionamento desse processo.

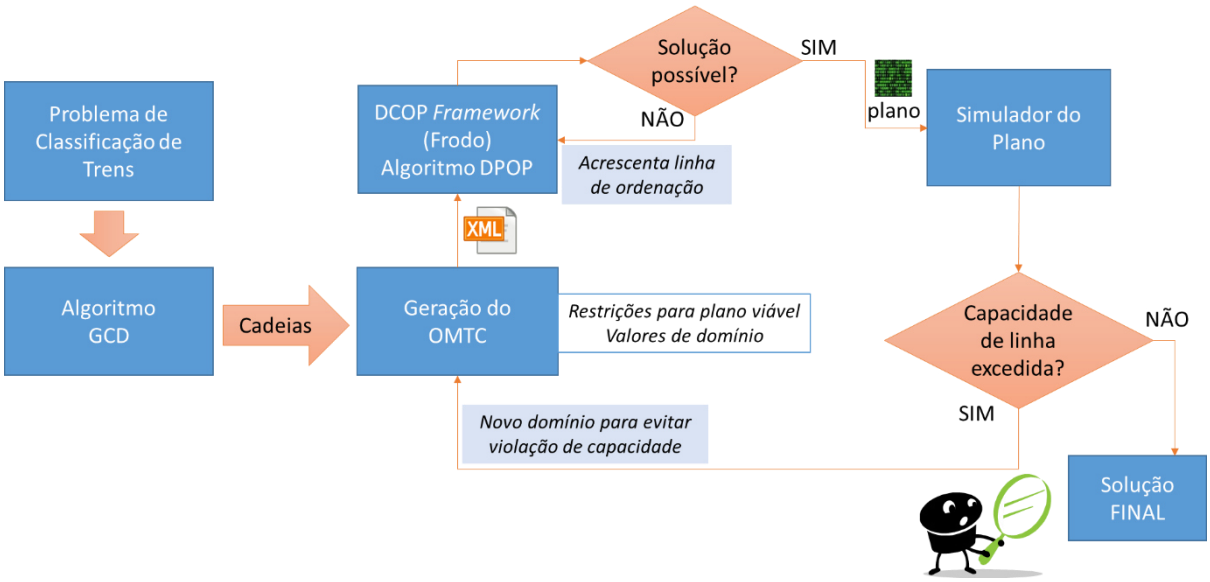

Figura 9 – DCOP Iterativo parcial

No I-DCOP parcial, o problema de classificação é inicialmente gerado somente com as restrições de viabilidade do plano baseadas na decomposição de cadeia fornecida pelo GCD. O plano de classificação gerado a partir desse modelo é simulado em um pátio configurado com a capacidade máxima das linhas de ordenação de acordo com a restrição do problema. Se, durante a execução de algum dos passos de ordenação, um ou mais vagões excederem essa capacidade, o simulador gera uma exceção indicando que a restrição de capacidade máxima da linha foi violada e o processo de simulação será interrompido ao final do passo atual de ordenação. Todos os vagões que excederam a capacidade de alguma das linhas de ordenação no passo que estava em execução, terão seu domínio reduzido. A redução do domínio consiste em retirar *bitstrings* contendo 1 no passo de ordenação que realizará *pull-out* da linha na qual o vagão excedeu a capacidade. Por exemplo, os vagões que excederam a capacidade da linha de ordenação 0, terão seu domínio reduzido, não contemplando nenhum valor que indique *bitstring* com *bit* 1 na linha 0, garantindo assim que eles não serão mais direcionados para essa linha de ordenação. O novo arquivo XML é enviado ao FRODO, que vai gerar um novo plano de classificação. O processo então é repetido até que nenhuma linha de ordenação tenha sua capacidade excedida. A definição do domínio de valores para as variáveis é portanto efetuada de forma iterativa.

Em determinado ponto do processo, pode não ser possível gerar uma solução devido a quantidade de restrições inseridas no modelo. Nesse caso, o FRODO devolverá uma solução com custo infinito. Se isso ocorrer, o processo acrescenta uma linha de classificação à configuração do pátio, o que faz com que o OMTC aumente a quantidade de valores de domínio para todos os vagões, adicionando um *bit*.

Para esclarecer o funcionamento desse processo e como o domínio de valores é reduzido será novamente utilizado o cenário de exemplo da Figura 5. O início do processo é exatamente o mesmo do processo simplificado. Assim, o cenário é configurado da mesma forma e as cadeias são geradas pelo GCD. A partir das cadeiras é gerado um XML igual ao do processo simplificado, que pode ser encontrado no Apêndice C. Porém, para que o processo iterativo seja acionado, é necessário definir uma capacidade para as linhas de ordenação menor do que a capacidade necessária para execução do plano de classificação ótimo. Para esse exemplo, a capacidade das linhas de ordenação foi reduzida para 6, valor suficiente para iniciar o processo iterativo. O processo inicia a execução do plano de classificação gerado pelo OMTC, executando o *roll-in* inicial que ocorre sem problemas, conforme apresentado na Figura 10.

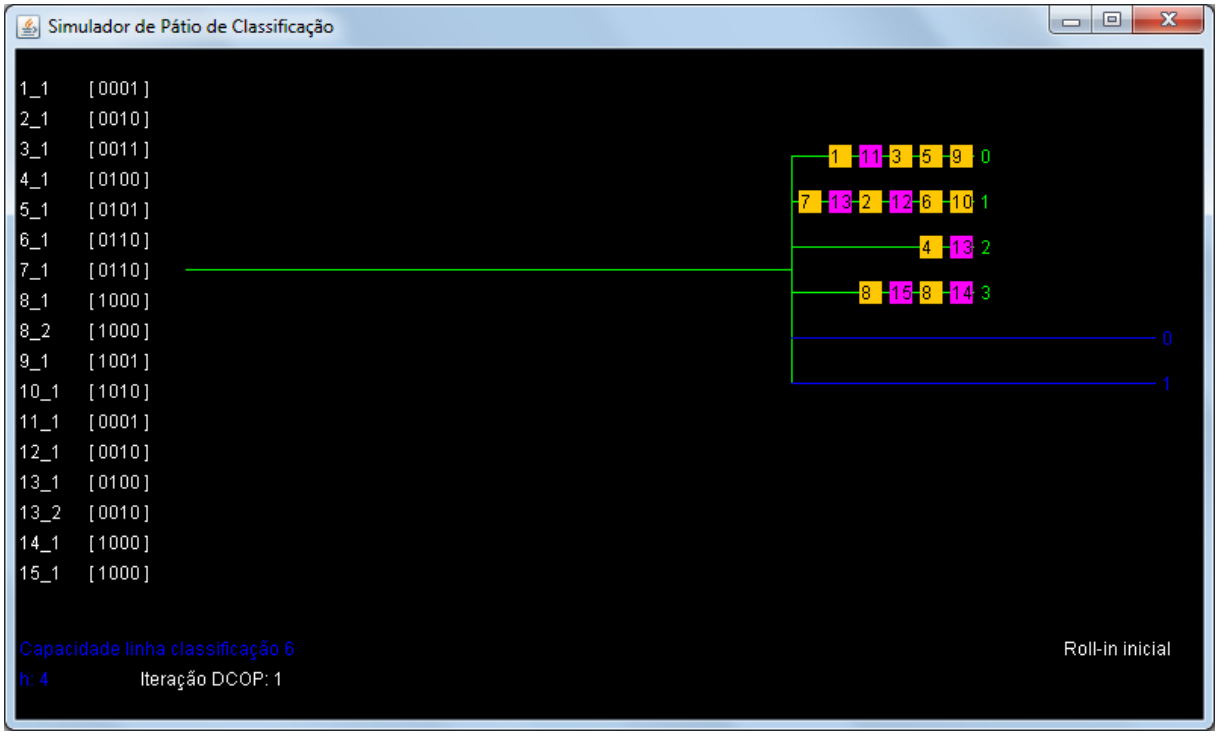

Figura 10 – *Roll-in* inicial processo iterativo

Porém, na execução do 1º passo, a linha de ordenação 1 terá sua capacidade excedida pelo vagão 3, conforme apresentado na Figura 11. Essa informação pode ser verificada pelo peso (*w*) da coluna que representa essa linha de ordenação, destacada em vermelho.

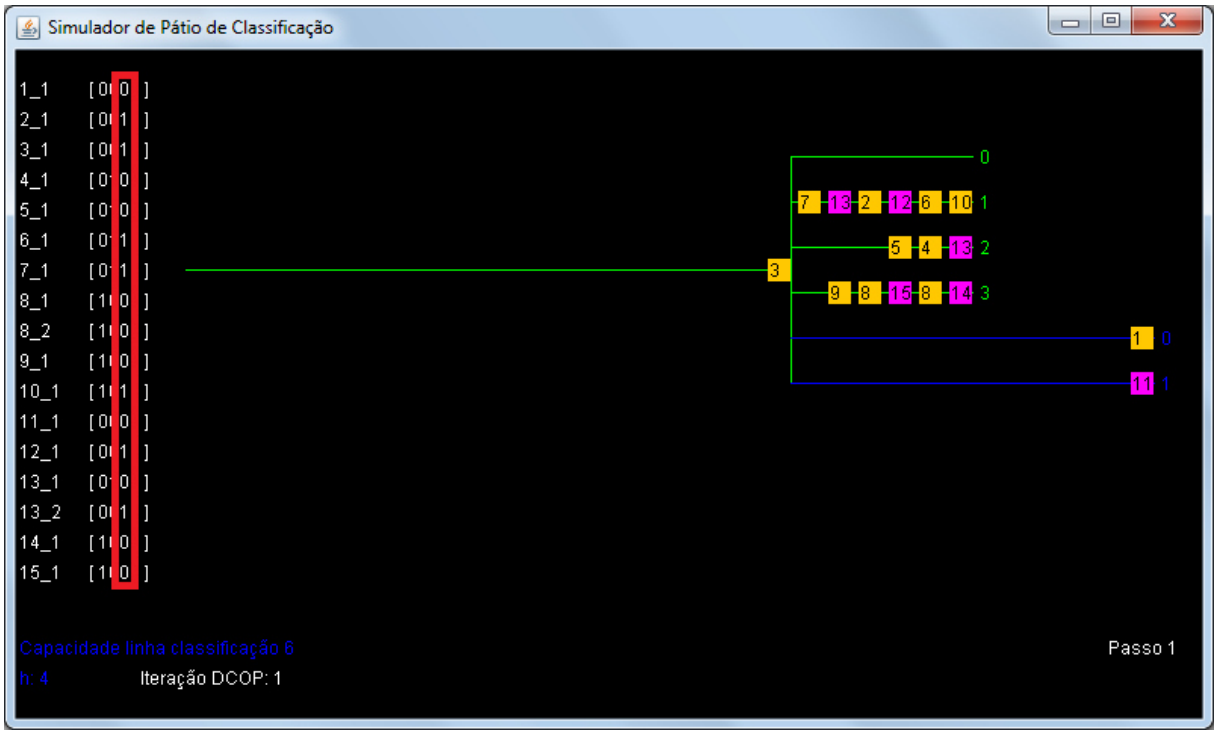

Figura 11 – Simulação do 1º passo de ordenação - capacidade da linha 1 excedida

Quando isso ocorre, o simulador encerra a simulação do passo de ordenação atual e altera o domínio de valores do vagão 3, para que o mesmo não possa mais ser direcionado à linha de ordenação 1. O novo domínio do vagão 3 é definido conforme a Figura 12:

```
<domain name="decimais3_1">1 4..5 8..9 12..13</domain>

Que representa as seguintes bitstrings:
0001
0100
0101
1000
1001
1100
1101
```

Figura 12 – Domínio reduzido para o vagão 3

A partir disso, o processo gera novamente o OMTC, seguindo os passos apresentados. Para execução completa do cenário exemplo apresentado na Figura 5, com uma

capacidade máxima de 6 vagões nas linhas de ordenação são necessárias 17 gerações do OMTC e um total de 5 linhas de ordenação, conforme plano final apresentado e encerrado pelo simulador na Figura 13:

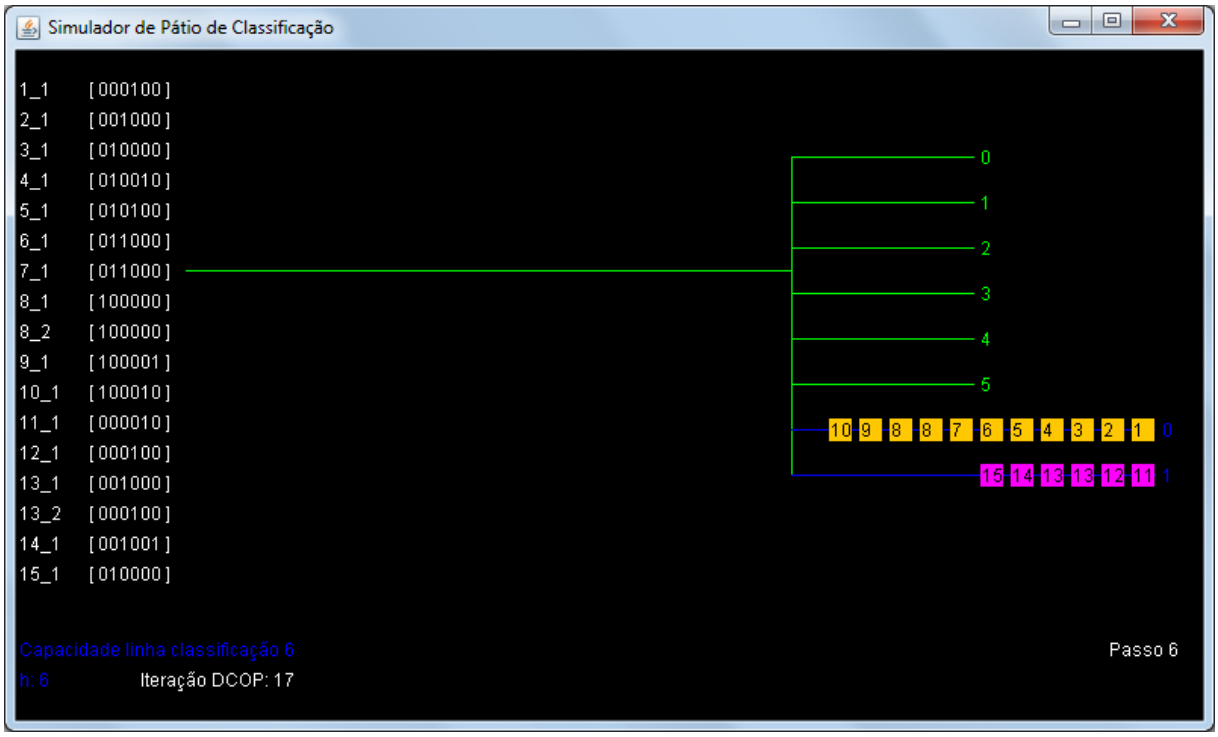

Figura 13 – Execução final do cenário da Figura 5 com capacidade limitada

Esse processo viabiliza a inclusão da restrição de capacidade de linha pela divisão do problema original em subproblemas, que são obtidos e formulados a partir de uma execução parcial do plano de classificação gerado. A partir do momento que a restrição de capacidade de linha é violada, o processo decide tratar o cenário como um subproblema, gerando um novo OMTC e reenviando-o ao algoritmo DCOP.

Com essa divisão do problema, a solução final deixa de ser uma solução ótima global, pois tem-se soluções ótimas para cada subproblema, sendo então a solução final sub-ótima.

4.4.3. DCOP Iterativo completo

O DCOP iterativo (I-DCOP) completo segue a mesma ideia base do processo iterativo parcial. Porém, nesse processo é incluída a restrição de quantidade máxima de linhas de ordenação. Só serão adicionadas linhas de ordenação enquanto não for excedida a

quantidade máxima especificada na configuração do pátio ferroviário. Para garantir isso, o processo efetua essa verificação antes de acrescentar uma nova linha de ordenação na configuração do pátio.

A proposta desse processo é simular a situação real encontrada nos pátios ferroviários, onde há quantidade limitada de linhas e elas possuem um tamanho fixo. Porém, como o processo iterativo não encontra a solução ótima global, pode ocorrer de não haver uma solução possível que respeite todas as restrições. Se isso ocorrer e o processo já tiver atingido a quantidade máxima de linhas de ordenação, a capacidade máxima das linhas será incrementada. Quando isso for necessário, o processo vai remover as restrições de capacidade previamente impostas ao modelo. Com isso os domínios que haviam sido reduzidos anteriormente serão reinicializados, ou seja, criados como se o processo buscasse um plano de classificação viável. Isso é necessário, pois, caso contrário, mesmo aumentando a capacidade das linhas de ordenação, essa capacidade não seria utilizada no OMTC.

Esse processo foi definido para buscar uma solução que atenda o cenário real do problema de classificação de trens, que possui recursos limitados. Os recursos estão relacionados à configuração física do pátio ferroviário. Um pátio possui uma quantidade limitada de linhas de ordenação disponíveis para o processo de classificação de trens, e essas linhas têm uma capacidade máxima de vagões que podem comportar. A Figura 14 apresenta o funcionamento desse processo.

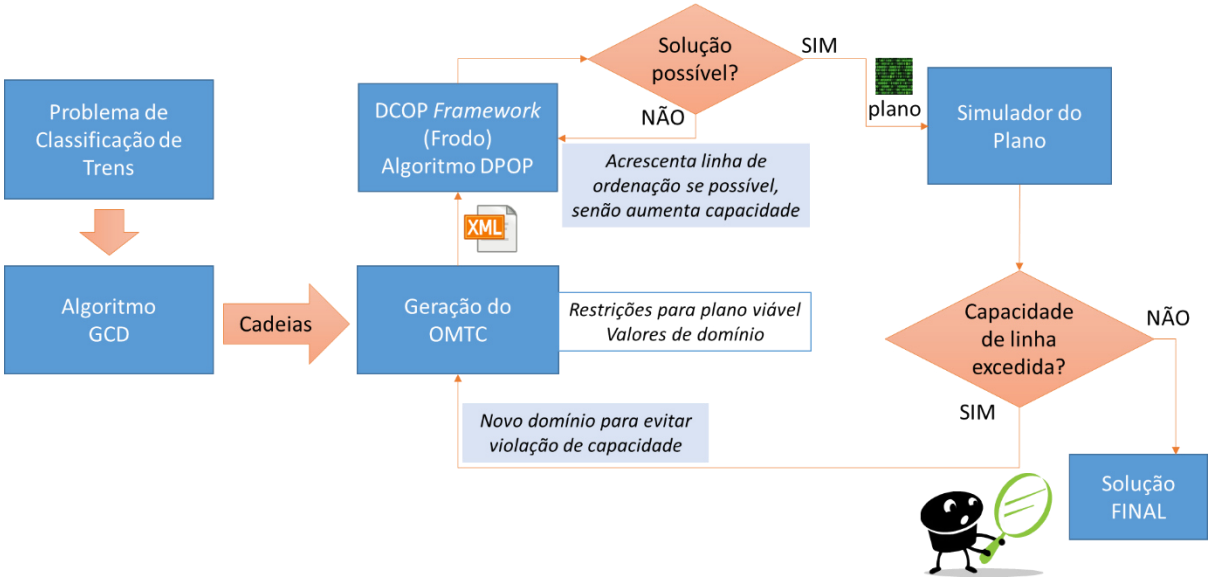

Figura 14 – DCOP Iterativo completo

4.5. Considerações do Capítulo

O OMTC trabalha com uma quantidade de valores de domínio grande, diretamente relacionada à quantidade de linhas de ordenação utilizadas no pátio (um pátio real pode ter, por exemplo, 10 linhas de classificação, o que representa uma grandeza de 2^{10} valores). Isso prejudica o desempenho computacional dos algoritmos DCOP, em sua busca de uma solução com grau de qualidade para o problema. Apesar da descentralização do problema, dividindo em vários subproblemas, um para cada trem de saída que deve ser formado, os vagões de um mesmo trem de saída acabam criando várias restrições entre si. A quantidade de restrições para cada trem de saída está relacionada a quantidade de cadeias geradas para o trem, que na prática indica o grau de pré-ordenação dos vagões. Porém, no OMTC, mesmo um trem com uma única cadeia, que tem seus vagões já na ordem correta em sua chegada ao pátio, terá um grande número de restrições. Para cada vagão da cadeia, será incluída uma restrição para garantir que ele receba uma *bitstring* maior ou igual ao vagão anterior, garantindo que os vagões não troquem de lugar no pátio. Isso gerou um modelo complexo, que prejudica o desempenho dos algoritmos DCOP.

A restrição de capacidade de linhas de ordenação, se incluída diretamente ao OMTC, gera um ponto de centralização entre os agentes envolvidos, o que compromete mais ainda o desempenho dos algoritmos DCOP. Como alternativa, foi desenvolvido o I-DCOP que, devido a divisão efetuada no cenário, acaba gerando soluções ótimas locais, mas não globais. Isso permite encontrar soluções sub-ótimas para os cenários, porém o tempo para obter-se a solução cresce à medida que mais linhas de ordenação são adicionadas no processo. Considerando que um pátio ferroviário real, conforme o utilizado nos dados recebidos, possui 10 linhas de ordenação, o tempo para solução dos problemas é aceitável.

Para avaliar o OMTC e o I-DCOP parcial e completo foram utilizados cenários fictícios baseados em dados reais fornecidos pela *SBB*, referente a uma semana de tráfego no Pátio de Lausanne (Suíça), que serão descritos no próximo capítulo.

Capítulo 5

Resultados

Esse capítulo descreve os experimentos realizados aplicando-se o OMTC em cenários fictícios baseados em dados reais fornecidos pela *SBB* e seus resultados. Os experimentos foram baseados em dados referentes a uma semana de tráfego no pátio de Lausanne (Suíça) ocorridos no ano de 2005. Os cenários foram considerados fictícios pois, apesar da *SBB* ter fornecido planilhas eletrônicas contendo informações sobre os vagões que chegaram ao pátio de Lausanne e o planejamento dos trens que partiram do mesmo pátio, algumas informações sobre a operação, funcionamento e regras do pátio não estavam explícitas. Entre elas, por exemplo, a informação de para qual trem de saída cada vagão que chega ao pátio deve ser direcionado. O processo de elaboração desses cenários está descrito em maiores detalhes no Apêndice E.

Os experimentos foram realizados em uma máquina contendo 32Gb de memória RAM e com 2 processadores Intel Xeon X5660 com 2.80GHz. Foi criada uma seção para detalhar os experimentos iniciais, que avaliam OMTC e uma seção com os experimentos realizados com o I-DCOP parcial e o I-DCOP completo. Ao final do capítulo foi consolidada uma seção com lições aprendidas durante o processo de execução dos experimentos e a concepção do OMTC.

5.1. Avaliação do OMTC e escolha do algoritmo DCOP

Para validar o OMTC descrito, foram gerados alguns cenários de exemplo contendo poucos vagões, com o intuito de verificar de forma simplificada o plano de classificação gerado e também o comportamento dos algoritmos DCOP escolhidos: ADOPT e DPOP.

Tabela 3 – Configuração dos cenários utilizados para validação inicial do OMTC
n: total de vagões, *m:* trens de saída, n_{max}*:* quantidade de vagões do maior trem de saída, *h*: menor quantidade de passos de ordenação necessária, *k:* tamanho da decomposição de cadeia, k_{total}: total de cadeias geradas

instância	***n***	***m***	n_{max}	***h***	***k***	k_{total}
cenário 1	17	2	11	4	9	13
cenário 2	16	1	16	3	6	6
cenário 3	21	3	11	4	9	15

A Tabela 3 apresenta a configuração dos cenários utilizados como exemplos para validação do OMTC. Foram criados cenários contendo poucos vagões e uma quantidade de passos de ordenação que varia entre 3 e 4. O objetivo da execução desses cenários simples foi verificar se o plano de classificação gerado era viável e ótimo e também se a função de custo global definida para minimizar a quantidade de *roll-ins* estava atuando corretamente.

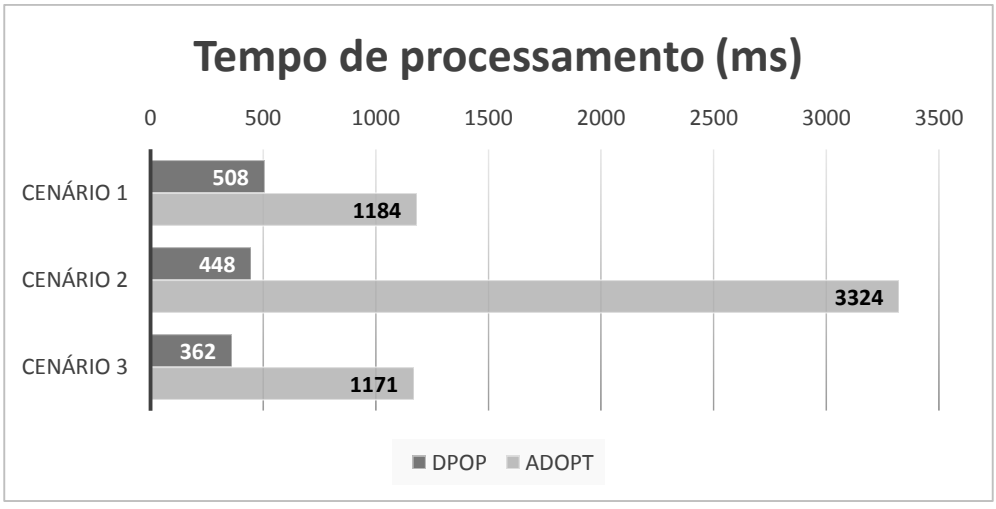

Figura 15 – Cenários de exemplo para validação do OMTC

A partir desses cenários, foi possível identificar que o OMTC gerou corretamente os planos de classificação, minimizando os *roll-ins*. Ficou também visível a diferença de desempenho entre os algoritmos DPOP e ADOPT para o modelo proposto. O DPOP

conseguiu resolver os cenários de exemplo em um tempo 71% menor, na média dos três cenários. No pior caso para o DPOP, que foi obtido no cenário 1, ele resolveu o problema em um tempo 57% menor.

Essa diferença de desempenho, identificada nos cenários pequenos de exemplo, foi confirmada pelo surgimento de um problema (memória da pilha excedida) na execução dos cenários reais com o algoritmo ADOPT, mesmo utilizando somente as restrições de viabilidade de plano.

O comportamento do algoritmo ADOPT em relação ao domínio do problema e o modelo proposto, coincide com o trabalho apresentado em (JUNGES e BAZZAN, 2008). Esses autores efetuaram uma avaliação de desempenho dos algoritmos ADOPT, OptApo e DPOP em um problema real complexo, envolvendo um cenário de sincronização de semáforos. Para o domínio do problema apresentado nessa pesquisa, o ADOPT é mais suscetível às variações na definição das restrições. O ADOPT necessita de um tempo maior de processamento para encontrar uma solução quando há um maior número de restrições com alto custo (ou seja, maior conflito).

No domínio do problema de classificação de trens, as variáveis que representam os vagões de um mesmo trem estão totalmente ligadas por restrições. Por isso, o comportamento do algoritmo ADOPT mostrou ter um desempenho inferior ao DPOP, assim como no problema de sincronização dos semáforos. Além disso, como o ADOPT possui um número maior de mensagens trocadas entre os agentes, a sua estratégia de busca em profundidade na pseudo-árvore de restrições pode ocupar uma quantidade muito grande de memória, antes mesmo que a solução seja encontrada.

Devido a isso, os experimentos envolvendo cenários baseados nos dados reais foram realizados somente com o algoritmo DPOP. Para uma análise mais consistente do OMTC foram realizados experimentos utilizando cenários fictícios baseados nos dados reais correspondentes a uma semana de tráfego no pátio de Lausanne na Suíça, fornecidos pela *SBB* e ocorridos no ano de 2005. Uma descrição mais detalhada de como esses cenários foram gerados a partir desses dados pode ser obtida no Apêndice E.

5.2. Experimentos do processo simplificado

Para realização dos experimentos foram utilizadas as cinco instâncias de problemas de classificação de trens, que representam dados dos cinco dias de tráfego do pátio de Lausanne, geradas a partir da base de dados reais enviada pela *SBB*.

O processo simplificado gerou um plano de classificação viável e ótimo corretamente, otimizando ainda a quantidade total de *roll-ins*, conforme função de custo global do OMTC. Para esse processo, foi considerado que a capacidade e a quantidade de linhas de ordenação eram ilimitados. A partir desses experimentos foi possível verificar que o OMTC atendeu ao seu propósito. Um resumo dos dados utilizados nesses experimentos é apresentado na Tabela 4.

Tabela 4 – Resumo dos experimentos do processo simplificado

instância	*n*	*m*	*h*	C_{max}	*tempo algoritmo DCOP (ms)*	*tempo total (ms)*
segunda	486	24	3	288	872	37444
terça	329	24	3	197	523	17757
quarta	310	25	3	184	530	23978
quinta	364	25	2	292	536	21866
final de semana	368	27	2	265	290	19437

A coluna *n* apresenta o total de vagões do cenário e a quantidade *h* indica a quantidade de passos de ordenação necessária para execução do plano, que considera todos os vagões sendo inicialmente direcionados à área de classificação (seguindo a característica operacional do pátio de Lausanne). A maior capacidade de linhas de ordenação necessária para execução do plano é apresentada na coluna C_{max}. Além dessas informações, foi computado o tempo de execução do algoritmo DCOP (obtido pelo *framework* FRODO) para o OMTC e o tempo total de execução, que é composto por: execução do algoritmo GCD, geração do OMTC baseada nas cadeias, solução OMTC pelo FRODO (DPOP) e a simulação do plano de classificação gerado.

A verificação do plano gerado foi realizada a partir de um simulador de execução de plano de classificação, desenvolvido durante a pesquisa. A Figura 16 mostra os 24 trens de saída referentes ao cenário de segunda-feira, formados corretamente. Lembrando que o critério adotado para formação dos trens de saída é que os vagões devem ser ordenados por tipo consecutivo em ordem crescente. Foi designada uma cor aleatória para cada trem de saída para facilitar a visualização. Sem a restrição de capacidade de linhas de ordenação o problema é resolvido enviando o OMTC uma única vez ao FRODO.

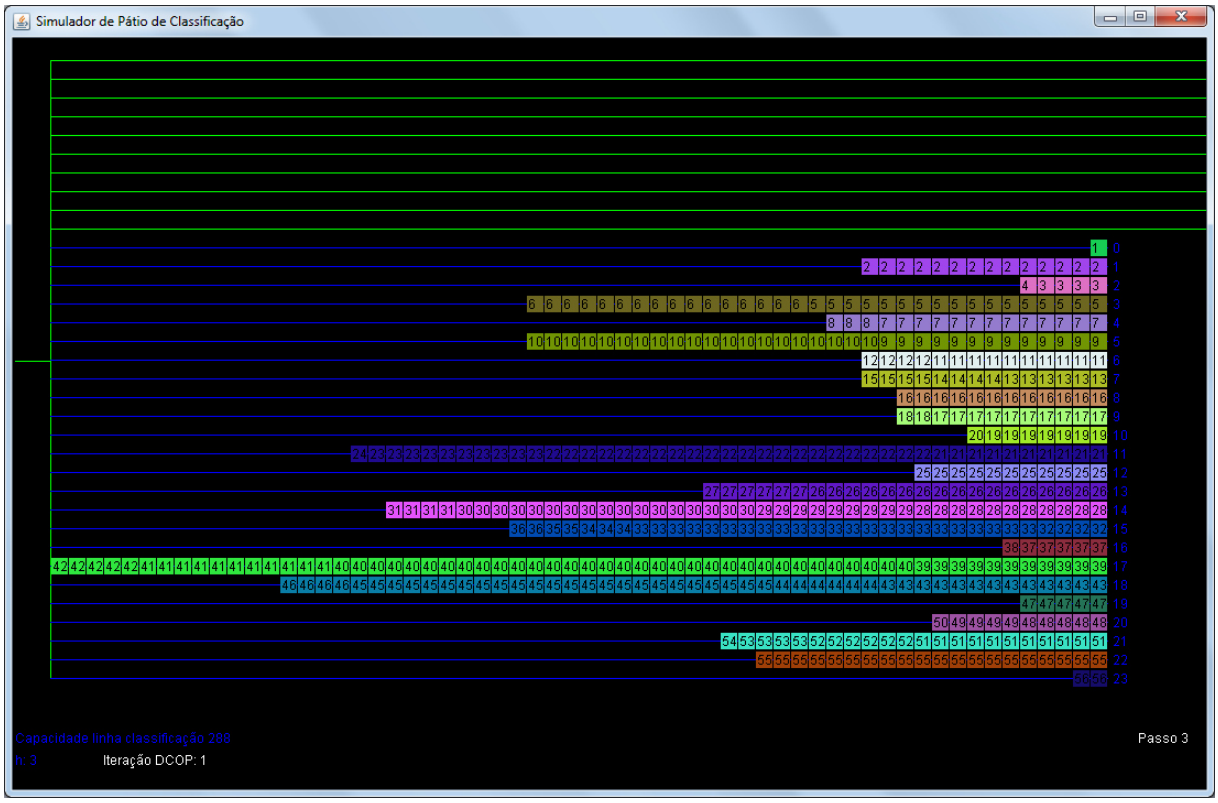

Figura 16 – Tela final da simulação do cenário simplificado (segunda-feira)

Após a execução dos cinco dias de tráfego no simulador, foi possível verificar que todos os planos de classificação gerados são viáveis e ótimos e que também minimizaram o custo total de *roll-ins*, conforme apresentado na Tabela 5. A otimização da quantidade total de *roll-ins* é melhor visualizada comparando o resultado do OMTC com a solução gerada pelo algoritmo GCD.

Tabela 5 – Comparativo entre o algoritmo GCD e o OMTC

instância	*n*	*k*	k_{total}	*GCD*		*OMTC*	
				h	*Custo*	*h*	*Custo*
segunda	486	4	46	3	539	3	490
terça	329	4	42	3	363	3	338
quarta	310	4	44	3	327	3	311
quinta	364	3	41	2	401	2	401
final de semana	368	3	47	2	395	2	395

Utilizando o OMTC foi possível obter um plano de classificação viável e ótimo de forma rápida (menos de 1 segundo no pior cenário) e ao mesmo tempo foi possível otimizar a quantidade de *roll-ins*, o que é um diferencial em relação a abordagem GCD. O custo de *roll-ins* foi menor nos três primeiros cenários executados.

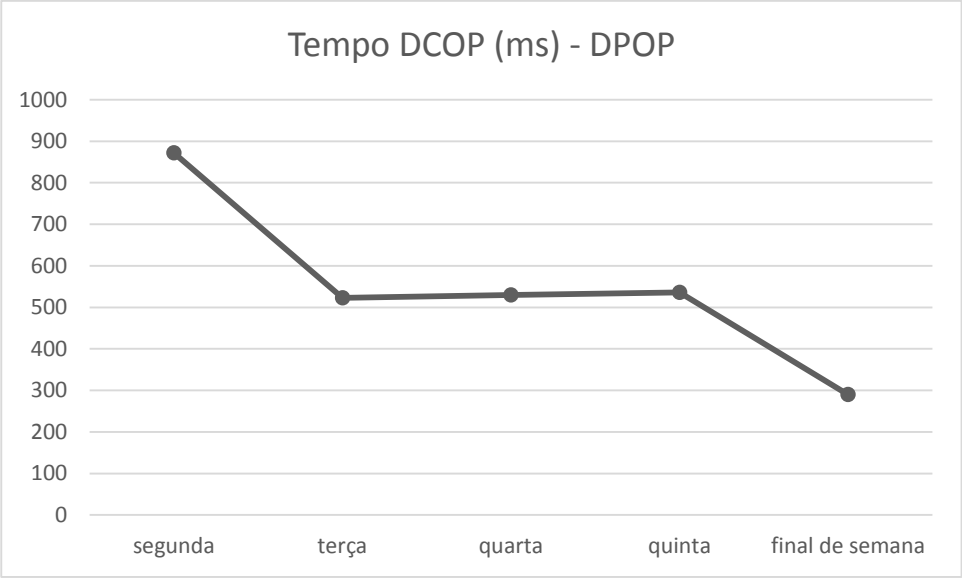

Figura 17 – Tempo de processamento do processo simplificado para cenários baseados em dados reais

O tempo de processamento da geração do plano de execução utilizando o OMTC e o algoritmo DPOP está representado no gráfico da Figura 17. O maior tempo de processamento ocorre no cenário de segunda-feira, que possui a maior quantidade de vagões e também maior quantidade de cadeias, o que torna o problema mais complexo. Entre o cenário de quarta e quinta, houve uma estabilidade aproximada no tempo de processamento, pois apesar de

quinta-feira ter uma quantidade aproximada 17,5% superior de vagões, possui um k menor (representa a maior quantidade de cadeias entre os trens de saída). O menor tempo para solução do problema ocorreu no cenário do final de semana, que apesar de ter uma quantidade de vagões e k quase igual a quinta-feira, apresenta maior número de trens de saída. Como os vagões cada trem de saída apresentam um problema independente dos vagões de outros trens, quanto maior a quantidade de trens de saída menos restrito é o problema.

5.3. Experimentos do I-DCOP parcial

Os experimentos do I-DCOP parcial foram realizados utilizando-se as mesmas cinco instâncias obtidas a partir dos dados reais fornecidos pela *SBB*, representando dados de uma semana de tráfego. Porém, para esse processo, as cinco instâncias foram combinadas com capacidades de linha de ordenação em um intervalo de [40, 50] vagões, similares a capacidade real do pátio de Lausanne ($C = 40$). Essa combinação gerou 55 cenários para validação do processo.

O objetivo principal foi avaliar o comportamento do OMTC e do I-DCOP, verificando como a quantidade de passos de ordenação h, a quantidade de iterações do processo e o tempo de processamento variam dentro desse intervalo de capacidades das linhas de ordenação.

O cenário de segunda-feira é o mais crítico. Ele possui a maior quantidade de vagões, ou seja, $n = 486$ e também a maior quantidade de cadeias $k = 4$. Para encontrar uma solução com a capacidade $C = 40$, foram necessárias 17 linhas de ordenação, o que fez com que o OMTC trabalhasse com um domínio de $1\ a\ 2^{17} - 1$. Com essa quantidade de valores de domínio, uma execução do algoritmo DCOP para um OMTC levou aproximadamente 25 minutos. Com isso, o tempo total para solução do cenário foi de aproximadamente 37 horas.

O gráfico da Figura 18 mostra a redução no tempo de processamento para o cenário de segunda-feira a medida que a capacidade das linhas de ordenação é incrementada. Quando isso ocorre, o OMTC necessita de uma quantidade menor de linhas de ordenação para chegar a uma solução. Com uma capacidade de 50 vagões, que representa um valor 25% superior a capacidade inicial, tem-se uma redução significativa no tempo de processamento, que passa de 37 horas para aproximadamente 26 minutos.

Com essa capacidade foram necessárias somente 13 linhas de ordenação para se encontrar a solução, o que representa um domínio de $1\ a\ 2^{13} - 1$, valor consideravelmente menor.

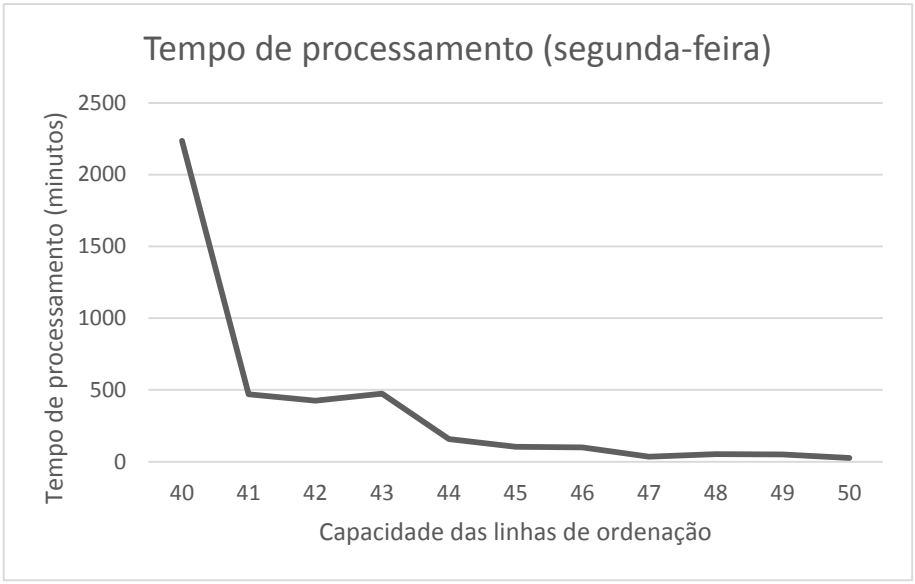

Figura 18 – Tempo de processamento I-DCOP parcial (segunda-feira)

Projetando o tempo de processamento do OMTC, em relação a quantidade de passos de ordenação h (Figura 19), é possível perceber que o aumento exponencial da quantidade de valores de domínio está diretamente relacionado ao aumento também do tempo de processamento do algoritmo DPOP. Com isso, é possível perceber que o OMTC, começa a enfrentar problemas de desempenho a partir da inclusão da 17ª linha de ordenação.

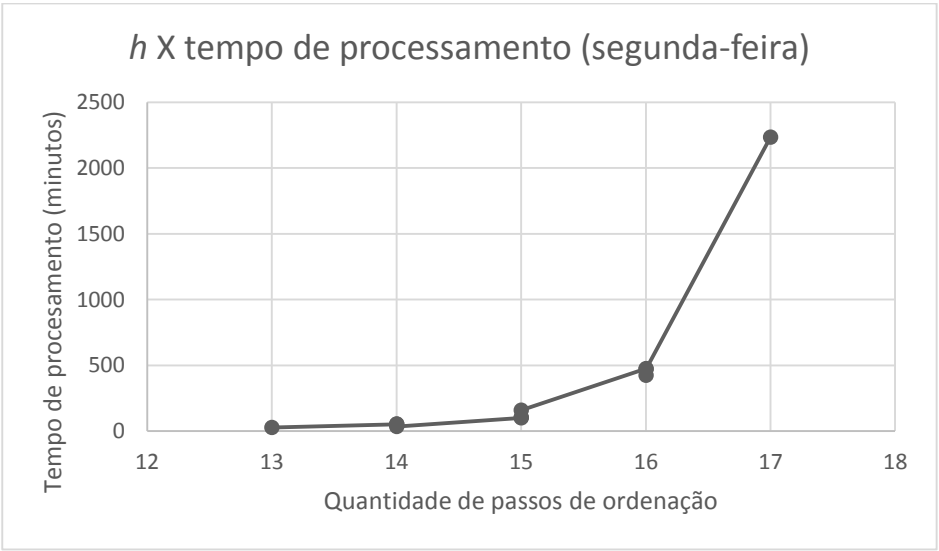

Figura 19 – Relação passos de ordenação e tempo de processamento (segunda-feira)

Já para os cenários dos outros dias de tráfego, nos quais a quantidade de vagões é menor, foi possível resolver o problema com no máximo 12 linhas de ordenação. O gráfico da Figura 20 mostra como o OMTC comportou-se nesses cenários, obtendo soluções para os cenários em um intervalo de aproximadamente 1 a 6 minutos.

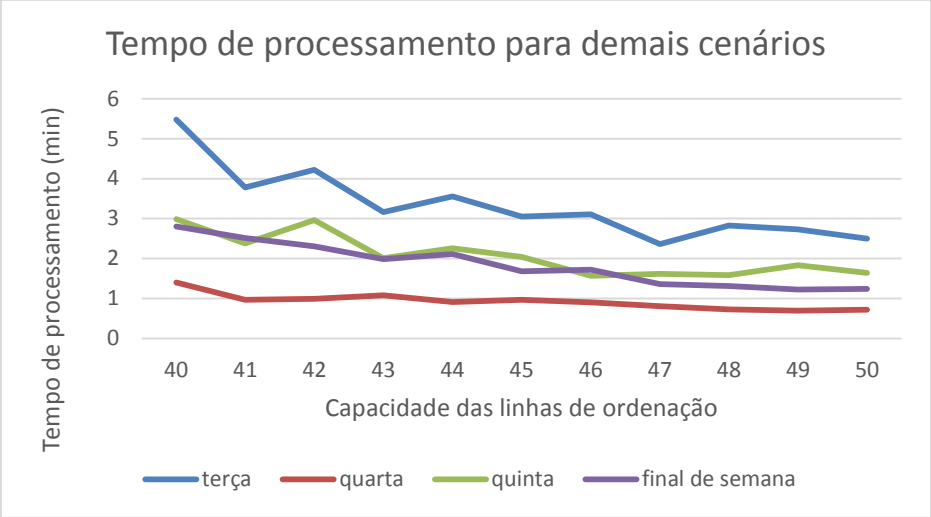

Figura 20 – Tempo de processamento I-DCOP parcial (demais cenários)

Outra métrica avaliada foi a quantidade de passos de ordenação necessário para obter uma solução com as diferentes capacidades. Em um pátio com uma rampa, a quantidade de passos de ordenação representa a quantidade de linhas de ordenação. O gráfico da Figura 21 mostra a relação entre a quantidade de linhas de ordenação e as diferentes capacidades utilizadas nos experimentos.

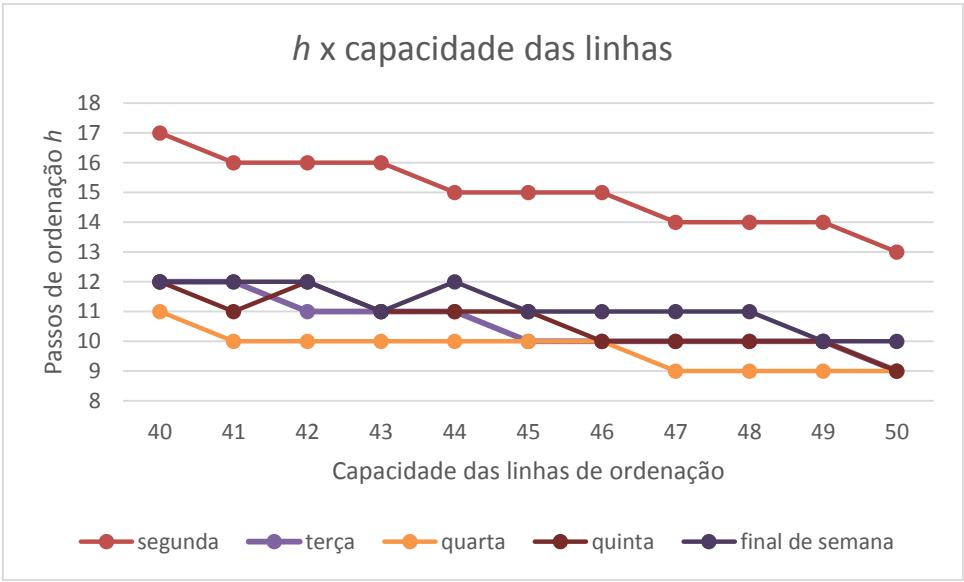

Figura 21 – Relação quantidade e capacidade de linhas de ordenação I-DCOP parcial

O cenário de quarta-feira poderia atender totalmente às restrições do pátio de Lausanne ($h = 10$), com o aumento de somente um vagão a capacidade das linhas de ordenação. Para o cenário de terça-feira, é necessária uma capacidade de 45 vagões nas linhas de ordenação para que o plano seja gerado com $h = 10$. Na quinta-feira, um plano com 10 passos de ordenação só pôde ser obtido com uma capacidade $C = 46$ e no cenário referente ao final de semana, foi necessária a capacidade $C = 49$. No cenário mais complexo, representado pelos dados de segunda-feira, o OMTC e o I-DCOP não obtiveram uma solução adequada. Mesmo com uma capacidade 25% superior nas linhas de ordenação ($C = 50$), seriam necessários 13 passos de ordenação para resolver o problema.

Durante a execução desses experimentos, foi necessário remover a otimização do custo total de *roll-ins* para geração do plano de classificação. Isso ocorreu, pois, à medida que o domínio de valores das variáveis vai exponencialmente crescendo, o problema vai se tornando mais restrito. Isso ocorre devido aos domínios definidos iterativamente para garantir a capacidade máxima das linhas de ordenação. Com quantidades de valores de domínio acima de 2^5, o algoritmo DPOP não conseguiu mais obter uma solução para o problema em um tempo inferior a 30 minutos. Esse valor foi estipulado como tempo máximo para que o algoritmo DPOP retornasse uma solução e é configurável.

5.4. Experimentos do I-DCOP completo

Os experimentos do processo iterativo completo foram realizados com o objetivo de gerar um plano de classificação para um cenário o mais próximo possível da realidade, baseado nos dados do pátio de Lausanne. A proposta desses experimentos foi verificar se algum plano gerado pelo processo poderia funcionar na prática no cenário avaliado. Para isso, foram gerados cinco cenários combinando os dados dos cinco dias de tráfego com as seguintes restrições: capacidade máxima de linhas de ordenação $C = 40$ e 10 linhas de ordenação, representadas por $h = 10$. Essas restrições representam a configuração do pátio de Lausanne. Conforme descrito anteriormente, caso não seja possível gerar um plano viável que atenda a todas essas restrições, a capacidade será incrementada até que seja possível gerar uma solução para o problema.

A Tabela 6 abaixo apresenta um resumo dos resultados desses experimentos. Porém, nenhum dos experimentos conseguiu chegar a uma solução que atendesse todas as restrições

do pátio real, utilizado como estudo de caso. Como no I-DCOP completo, a quantidade de passos de ordenação é limitada (nesses experimentos limitada a 10), o tempo total de processamento não apresenta uma variação tão grande entre os cenários como no I-DCOP parcial. Para o cenário mais complexo chegou-se a uma solução em aproximadamente 89 minutos.

Tabela 6 – Resumo dos dados dos cenários do I-DCOP completo

instância	*n*	*h*	*C*	*iterações*	*tempo total aprox.. (min)*
segunda	486	10	67	589	89
terça	329	10	45	172	13
quarta	310	10	41	59	4
quinta	364	10	46	178	19
final de semana	368	10	49	212	18

Comparativamente à abordagem proposta por Maue (2011), que permite gerar uma solução ótima utilizando uma abordagem de programação inteira, o I-DCOP completo necessita de capacidades maiores nas linhas de ordenação para execução do cenário. Isso se deve ao fato do I-DCOP não obter uma solução ótima para o problema.

Entre os planos de classificação gerados no trabalho de Maue (2011), um foi analisado em maiores detalhes, pois foi o plano de classificação que conseguiu reduzir um passo de ordenação em uma das rampas, liberando uma linha de ordenação durante todo o processo. Esse plano de classificação, referente ao cenário de quarta-feira, utilizou portanto somente 9 linhas de ordenação (com $h = 5$), atendendo a capacidade máxima $C = 40$ vagões por linha. Porém, com essa abordagem foi necessário um tempo de 5,75 horas para geração desse plano de classificação. O I-DCOP completo, para cenário mais complexo (segunda-feira - maior quantidade de vagões e cadeias), levou aproximadamente 89 minutos em sua execução. Essa comparação de tempo ilustra uma das vantagens em uma solução sub-ótima, porém não pode ser avaliada de forma pragmática. Maue (2011) utiliza os cenários

considerando adicionalmente o horário de saída dos trens e também a existência de duas rampas na área de classificação.

Devido ao processo ter sido implementado de forma iterativa, incluindo as restrições para que a capacidade da linha de ordenação não seja extrapolada a medida que o plano é executado e com uma visão parcial do problema, a solução final gerada não é ótima. Mas apresenta uma abordagem distribuída capaz de subdividir um problema complexo em subproblemas menores, buscando um grau de qualidade com menor tempo.

É importante destacar que no OMTC e no simulador desenvolvido, foi utilizado um *layout* de pátio simplificado, contendo uma única rampa. Por isso, a quantidade de passos de ordenação h representa também a quantidade de linhas de ordenação utilizadas. Diferentemente do apresentado por Maue (2011) em sua abordagem de programação inteira.

5.5. Lições aprendidas

Durante a concepção do OMTC, diferentes alternativas foram avaliadas para definir as variáveis que representavam os vagões do cenário problema, no intuito de reduzir o tempo de execução do modelo e também reduzir o domínio de valores.

A preocupação inicial era com a quantidade de valores de domínio para as variáveis dos vagões, que cresce exponencialmente à medida que a quantidade de passos de ordenação é incrementada, pois representa o intervalo entre 1 e $2^{qtde\ linhas} - 1$. Para tentar evitar isso, uma das opções verificadas foi a decomposição da variável do vagão contendo a *bitstring* completa em diversas variáveis, uma para cada *bit* do vagão. Com isso, as variáveis dos *bits* poderiam sempre assinalar valores 0 ou 1, o que representa uma redução considerável no domínio. Apesar disso, ainda havia a necessidade da criação da variável que representava a *bitstring* completa do vagão, para garantir as restrições de viabilidade do plano gerado. A principal diferença é que, a definição de valores dessa variável, seria efetuada pela composição das variáveis que representavam cada *bit*. Essa composição foi feita pela aplicação de uma função matemática que trabalhava com os valores binários de cada *bit*, transformando-os em um valor decimal com a representação da *bitstring* completa. Porém, essa alternativa de modelo demorava um tempo muito maior para encontrar a solução do problema, mesmo em cenários de exemplo. Isso ocorreu, pois essa decomposição aumentou a quantidade restrições do modelo, tornando-o ainda mais restrito.

Para tentar reduzir a quantidade de restrições do problema, foi também avaliada a decomposição da *bitstring* completa em variáveis representando 2 ou 3 *bits*. Mesmo assim, apesar da redução nos valores de domínio, o modelo ainda permanecia muito restrito, necessitando um tempo maior para sua solução.

O principal ganho de desempenho na solução do modelo ocorreu quando foi efetuada uma alteração na quantidade de agentes criados para o problema. Inicialmente havia sido incluído um agente por variável, o que é mais comumente encontrado na literatura sobre DCOP, o que na prática implicava na criação de um agente por vagão do cenário. Com isso, a medida que iam sendo incluídos mais passos de ordenação, o problema não podia mais ser resolvido, gerando falta de memória na máquina (mesmo em uma máquina com 32Gb de RAM). A falta de memória ocorria geralmente após a inclusão do sexto passo de ordenação, o que representava um domínio de 1 e $2^6 - 1$. A solução para permitir que o problema fosse resolvido foi diminuir a quantidade de agentes. Foi designado um agente por "computador" disponível por resolver a solução no cenário real. Assim, foi criado um agente por trem de saída, pois, na prática, cada trem vai possuir seu próprio computador de bordo. Dessa forma, seria possível que cada trem se responsabilizasse pelo processamento do plano de classificação de seus vagões. Esse agente ficou responsável por controlar as variáveis representando as *bitstrings* dos vagões que compõe o seu trem de saída. Essa mudança no modelo gerou uma economia de memória e tempo de processamento muito grande. O OMTC gerado para o cenário de exemplo da Figura 5, que inicialmente não obtinha solução a partir da inclusão da 6ª linha de classificação, passou a ser executado em aproximadamente 4 minutos com 10 linhas de classificação. O cenário real de segunda-feira, que enfrentava problemas de memória após a inserção da 6ª linha de ordenação, pôde ser resolvido com até 17 linhas de ordenação, como apresentado nos resultados do I-DCOP parcial. Essa alteração foi a principal responsável por permitir que o OMTC proposto pudesse ser executado utilizando os cenários baseados em dados reais.

5.6. Considerações do Capítulo

Os experimentos realizados com cenários baseados em dados reais indicaram que o OMTC atende ao seu propósito inicial. Ele permite a geração de um plano de classificação viável e ótimo para as situações problema. Além disso, o OMTC em seu processo simplificado, fornece um plano que minimiza a quantidade total de *roll-ins* dos vagões.

Porém, a extensão desse modelo para inclusão das restrições de quantidade e capacidade de linhas de ordenação pelo I-DCOP, gera uma busca exaustiva de soluções, o que faz com que sejam necessárias muitas iterações para obtenção da solução final para o problema. Além disso, nos cenários baseados em dados reais, nos quais a quantidade de linhas de ordenação necessária para solução é próxima ou superior a 10, o OMTC só pôde ser executado sem o cálculo do custo referente a quantidade total de *roll-ins* de vagões (minimização de *roll-ins*). Portanto, nesses experimentos o algoritmo DCOP foi utilizado como um DCSP, com a característica de fornecer uma solução satisfatória ou não (custo zero ou infinito). A retirada da função de custo para minimizar os *roll-ins* foi necessária para que o algoritmo conseguisse encontrar uma solução sem gerar problemas de memória. Com isso retirou-se do modelo a possibilidade de encontrar uma solução com um grau de qualidade, característica que diferencia um DCOP de um DCSP.

Foi possível, por meio dos experimentos realizados utilizando o I-DCOP parcial, mostrar como o OMTC e o algoritmo DPOP se comportam com grandes domínios de valores. Até uma quantidade de valores de domínio na ordem de 2^{16} o algoritmo resolveu o problema em aproximadamente 6 minutos. Somente a partir da inclusão da 17ª linha de ordenação, aumentando a quantidade de valores de domínio para a ordem de 2^{17}, o algoritmo DPOP passou a levar um tempo de aproximadamente 26 minutos para resolver o problema.

Os cenários fictícios gerados para os experimentos, apesar de envolverem os mesmos vagões por dia de tráfego, não refletem exatamente os mesmos cenários utilizados no trabalho de Maue (2011), conforme descrito no Apêndice E. Todavia, os cenários produzidos geraram uma proximidade maior da complexidade real de um plano de classificação para um pátio ferroviário e refletem uma complexidade similar ao proposto pelo autor, considerando quantidade de vagões e quantidade de cadeias.

Considerações Finais

A pesquisa descrita nesse trabalho consistiu em analisar um problema do mundo real, original do campo da engenharia ferroviária e aplicar uma técnica de Otimização Distribuída de Restrições para gerar uma solução viável ao problema. Inicialmente nenhuma restrição de recursos como capacidade ou quantidade das linhas de classificação do pátio ferroviário foi incluída. O problema consistiu, portanto, na geração de planos de classificação viáveis e ótimos para um cenário de classificação de trens. Relembrando, viável e ótimo estão relacionados, nesse caso, a menor quantidade de passos de ordenação necessária para gerar corretamente os trens de saída. O OMTC atingiu esse objetivo, pois foi capaz de gerar planos de classificação viáveis e ótimos para cenários fictícios baseados em dados reais.

O problema da classificação de trens pode ser caracterizado como complexo devido ao número de variáveis e restrições envolvidas, como por exemplo: vagões, quantidade de linhas de classificação, plano de viagem dos trens, etc., e devido a isso é um problema interessante para uma modelagem distribuída baseada em restrições. Inicialmente, como se trata de busca por uma solução ótima, o DCOP não seria totalmente necessário, podendo ser tratado somente como um Problema de Satisfação Distribuída de Restrições (DCSP). A opção pelo DCOP foi feita por permitir estender o problema atual, considerando demais restrições como: quantidade de *roll-ins*, quantidade ou capacidade limitadas de linhas de classificação.

O OMTC foi concebido a partir dos conceitos e da codificação proposta por Maue (2011). O mesmo foi estendido inicialmente para minimizar a quantidade de *roll-ins*, pela inclusão da função de custo global OMTC, permitindo gerar soluções com pouco tempo de processamento e com essa restrição adicional. Entretanto, a inclusão da restrição de capacidade de linha de classificação acabou descaracterizando a descentralização do modelo, gerando uma interdependência entre os agentes envolvidos no processo. Apesar disso, um processo iterativo chamado de I-DCOP foi definido, permitindo obter soluções sub-ótimas contendo essa restrição adicional.

O I-DCOP gera vários OMTCs, obtendo para cada modelo uma solução aplicando um algoritmo DCOP e, posteriormente, simulando a solução obtida. A simulação verifica se a restrição de capacidade foi violada, e nesse caso, efetua uma alteração no domínio do próximo OMTC gerado. O I-DCOP continua esse processo até que a solução possa ser simulada sem violar nenhuma restrição. A principal contribuição do I-DCOP é apresentar uma subdivisão do problema total em subproblemas por meio de iterações e simulações, que incluem restrições parciais ao problema. Cada subproblema é processado por um algoritmo DCOP e retornado ao processo iterativo, que tenta executar o resultado, e caso não consiga inclui-se mais restrições. Esse processo abre mão de uma solução ótima, porém gera uma solução sub-ótima viável para um problema complexo.

Essas restrições não foram incorporadas ao modelo como restrições entre variáveis, mas como novos domínios. Isso faz com que a árvore de restrições não seja alterada a cada nova iteração do modelo, pois somente são criados domínios específicos para determinadas variáveis, reduzindo os valores de escolha possíveis. Portanto, com essa abordagem iterativa o domínio de valores é alterado, o que apresenta uma opção para problemas complexos envolvendo muitas restrições e domínios com grande quantidade de valores.

Outra contribuição da pesquisa foi mostrar que o algoritmo DPOP consegue trabalhar com domínios com grande quantidade de valores, como apresentado nos experimentos do I-DCOP parcial, no qual o algoritmo chegou a trabalhar com uma quantidade de valores igual a 2^{17}.

O OMTC e sua utilização dentro dos diferentes processos propostos conseguiu gerar planos de classificação viáveis e também permitiu a inclusão das restrições de quantidade e capacidade de linhas de ordenação. A comparação entre a abordagem utilizada e a abordagem previamente implementada por (MAUE, 2011) não pôde ser realizada diretamente devido a algumas diferenças identificadas no resumo dos dados enviados, provavelmente resultado de algum tipo de interpretação diferente específico da operacionalização do pátio de Lausanne. Apesar disso, foi possível analisar o comportamento do modelo e seus processos com cenários fictícios, baseados nos dados reais.

Como trabalhos futuros, outros algoritmos DCOP podem ser avaliados para o OMTC proposto, como por exemplo os algoritmos ASODPOP e E[DPOP], já implementados no *framework* FRODO (LÉAUTÉ, OTTENS e SZYMANEK, 2009). O OMTC também pode ser alterado para incluir a restrição de capacidade das linhas de ordenação do pátio utilizando

abordagens que incluem o conceito de recursos em um DCOP. A capacidade das linhas de ordenação, nesse caso, representaria o recurso que os vagões podem consumir e que precisa ser minimizado globalmente. Uma possibilidade seria utilizar um *Multiply-Constrained* DCOP (MC-DCOP), que otimiza um objetivo global e garante que recursos limitados não sejam excedidos. Apesar de um DCOP ser muito útil para coordenar agentes distribuídos buscando otimizar a solução global, no problema de classificação de trens, a capacidade de linhas de ordenação representa um recurso consumido por cada vagão (de forma local), mas que deve ser otimizado na solução global (MAHESWARAN, TAMBE, *et al.*, 2004). Algoritmos para solucionar esse tipo de problemas exigem uma gestão mais complexa de escalabilidade e eficiência, e nos cenários avaliados não foi possível encontrar uma solução diretamente no OMTC para essa restrição utilizando os algoritmos DPOP e ADOPT. Outro *framework* que inclui o conceito de recursos a um DCOP é o *Resource Constrained* DCOP (RCDCOP), que separa a função de custo global das restrições de recursos, e também poderia ser avaliado para o OMTC definido (MATSUI, SILAGHI, *et al.*, 2008).

Também como trabalho futuro, o I-DCOP apresentado pode ser utilizado em outros domínios de problemas que podem se beneficiar pela inclusão de restrições adicionais com a definição dos domínios de forma iterativa, preferencialmente problemas que possuam muitas restrições envolvendo muitas variáveis.

Referências

ANTF. Glossário Agência Nacional de Transportes Terrestres (ANTF). Disponível em: <http://www.antf.org.br/pdfs/glossario.pdf>. Acesso em: 18/01/2013.

BOWRING, E.; TAMBE, M.; YOKOO, M. Multiply-constrained distributed constraint optimization. AAMAS. [S.l.]: ACM. 2006. p. 1413-1420.

CHANDRA, S. Railway Engineering. [S.l.]: Oxford University Press, Incorporated, 2007. ISBN ISBN: 9780195687798. Disponivel em: <http://books.google.com.br/books?id=I1hdPQAACAAJ>.

CNT 2001. Pesquisa CNT de Ferrovias, realizada pela Confederação Nacional de Transporte. Disponível em: <http://www.cnt.org.br/Paginas/Pesquisas_Detalhes.aspx?p=7>. Acesso em: 20/12/2012.

DAGANZO, C. F. Static Blocking at Railyards: Sorting Implications and Track Requirements. Transportation Science, v. 20, n. 3, p. 189-199, 1986. ISSN DOI: 10.1287/trsc.20.3.189. Disponivel em: <http://transci.journal.informs.org/content/20/3/189.abstract>.

DAGANZO, C. F.; DOWLING, R. G.; HALL, R. W. Railroad classification yard throughput: The case of multistage triangular sorting. Transportation Research Part A: General, v. 17, n. 2, p. 95-106, 1983. ISSN ISSN: 0191-2607 DOI: 10.1016/0191-2607(83)90063-8. Disponivel em: <http://www.sciencedirect.com/science/article/pii/0191260783900638>.

DAHLHAUS, E. et al. Algorithms for Combinatorial Problems Related to Train Marshalling. IN PROCEEDINGS OF AWOCA 2000, IN HUNTER VALLEY. [S.l.]: [s.n.]. 2000. p. 7-16.

DAHLHAUS, E. et al. The train marshalling problem. Discrete Applied Mathematics, v. 103, p. 41-54, 2000. ISSN ISSN: 0166-218X DOI: 10.1016/S0166-218X(99)00219-X. Disponivel em: <http://www.sciencedirect.com/science/article/pii/S0166218X9900219X>.

FALAVINHA, L. A. Curso Professor Joronymo Monteiro Filho de Atualização em Engenharia Ferroviária para Graduados. [S.l.]: UFRJ/ENGEFER, 1982.

GATTO, M. et al. Shunting for Dummies: An Introductory Algorithmic Survey. In: AHUJA, R. K.; MÖHRING, R.; ZAROLIAGIS, C. D. Robust and Online Large-Scale Optimization. [S.l.]: Springer Berlin Heidelberg, v. 5868, 2009. p. 310-337. ISBN ISBN: 978-3-642-05464-8 DOI: 10.1007/978-3-642-05465-5_13. Disponivel em: <http://dx.doi.org/10.1007/978-3-642-05465-5_13>.

HANSMANN, R. S.; ZIMMERMANN, U. T. Optimal Sorting of Rolling Stock at Hump Yards. In: KREBS, H.-J.; JÄGER, W. Mathematics – Key Technology for the Future. [S.l.]: Springer Berlin Heidelberg, 2008. p. 189-203. ISBN ISBN: 978-3-540-77202-6 DOI: 10.1007/978-3-540-77203-3_14. Disponivel em: <http://dx.doi.org/10.1007/978-3-540-77203-3_14>.

JACOB, R. et al. Multistage methods for freight train classification. Networks, v. 57, n. 1, p. 87-105, 2011. ISSN ISSN: 1097-0037 DOI: 10.1002/net.20385. Disponivel em: <http://dx.doi.org/10.1002/net.20385>.

JUNGES, R.; BAZZAN, A. L. C. Evaluating the Performance of DCOP Algorithms in a Real World, Dynamic Problem. Proceedings of the 7th International Joint Conference on Autonomous Agents and Multiagent Systems - Volume 2. Richland, SC: International Foundation for Autonomous Agents and Multiagent Systems. 2008. p. 599-606.

LÉAUTÉ, T.; OTTENS, B.; SZYMANEK, R. FRODO 2.0: An Open-Source Framework for Distributed Constraint Optimization. Proceedings of the IJCAI'09 Distributed Constraint Reasoning Workshop (DCR'09). Pasadena, California, USA: [s.n.]. July~13 2009. p. 160-164. \url

LESSER, V.; ORTIZ, C.; TAMBE, M. (Eds.). Distributed Sensor Networks: A Multiagent Perspective (Edited book). [S.l.]: Kluwer Academic Publishers, v. 9, 2003. Disponivel em: <http://mas.cs.umass.edu/paper/255>.

LYNCH, N. A. Distributed Algorithms. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1996. ISBN ISBN: 1558603484.

MAHESWARAN, R. T. et al. Taking DCOP to the Real World: Efficient Complete Solutions for Distributed Multi-Event Scheduling. Proceedings of the Third International Joint Conference on Autonomous Agents and Multiagent Systems - Volume 1. Washington, DC, USA: IEEE Computer Society. 2004. p. 310-317.

MAILLER, R.; LESSER, V. Solving Distributed Constraint Optimization Problems Using Cooperative Mediation. Proceedings of Third International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS 2004). [S.l.]: IEEE Computer Society. 2004. p. 438-445.

MÁRTON, P.; MAUE, J.; NUNKESSER, M. An Improved Train Classification Procedure for the Hump Yard Lausanne Triage. ATMOS 2009 - 9th Workshop on Algorithmic Approaches for Transportation Modeling, Optimization, and Systems. Dagstuhl, Germany: Schloss Dagstuhl--Leibniz-Zentrum fuer Informatik, Germany. 2009.

MATSUI, T. et al. Resource constrained distributed constraint optimization using resource constraint free pseudo-tree. AAMAS '08: Proceedings of the 7th international joint conference on Autonomous agents and multiagent systems. Richland, SC: International Foundation for Autonomous Agents and Multiagent Systems. 2008. p. 1405-1408.

MAUE, J. H. On the Problem of Sorting Railway Freight Cars: An Algorithmic Perspective. [S.l.]. 2011.

MODI, P. J. et al. An asynchronous complete method for distributed constraint optimization. Proceedings of the second international joint conference on Autonomous agents and multiagent systems. New York, NY, USA: ACM. 2003. p. 161-168.

MODI, P. J. et al. Adopt: Asynchronous Distributed Constraint Optimization with Quality Guarantees. Artif. Intell., Essex, UK, v. 161, n. 1-2, p. 149-180, #jan# 2005. ISSN ISSN: 0004-3702 DOI: 10.1016/j.artint.2004.09.003. Disponivel em: <http://dx.doi.org/10.1016/j.artint.2004.09.003>.

PETCU, A.; FALTINGS, B. A Scalable Method for Multiagent Constraint Optimization. Proceedings of the 19th International Joint Conference on Artificial Intelligence. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc. 2005. p. 266-271.

PETERSEN, E. R. Railyard Modeling: Part I. Prediction of Put-Through Time. Transportation Science, v. 11, n. 1, p. 37-49, 1977. ISSN DOI: 10.1287/trsc.11.1.37. Disponivel em: <http://transci.journal.informs.org/content/11/1/37.abstract>.

REIS, S. A. D. Demanda por Transporte Ferroviário: O caso do transporte de açúcar na malha ferroviária da região centro-sul. Pontifícia Universidade Católica do Rio de Janeiro - PUC-RIO. [S.l.]. 2007.

STEFANO, G. D. et al. Models for Rearranging Train Cars. ARRIVAL Project. [S.l.]. 2007.

TSANG, E. P. K. Foundations of constraint satisfaction. [S.l.]: Academic Press, 1993. I-XVIII, 1-421 p. ISBN ISBN: 978-0-12-701610-8.

WRIGHT, P. H.; ASHFORD, N.; STAMMER, R. Transportation engineering: planning and design. [S.l.]: J. Wiley, 1998. ISBN ISBN: 9780471173960 LCCN: 97015256. Disponivel em: <http://books.google.com.br/books?id=gu1RAAAAMAAJ>.

Glossário de Termos Ferroviários

Área de Classificação	Área do pátio ferroviário composta por diversas linhas onde os vagões são direcionados para o processo de classificação.
Área de Despacho	Área do pátio ferroviário composta de linhas nas quais os trens prontos para partida aguardam sua saída do pátio.
Área de Recepção	Área do pátio ferroviário onde os trens de entrada são recebidos.
Classificação de Trens	Processo que define as operações dos vagões que chegaram ao pátio permitindo que os mesmos sejam organizados de acordo com algum critério pré-definido para formar os trens de saída desejados.
Lead Track	Linha principal onde os vagões vindos da área de recepção são direcionados antes de uma operação de *roll-in*.
Linha de ordenação	Linha dentro da área de classificação de um pátio ferroviário utilizada para o processo de ordenação dos vagões (*roll-in* e *pull-out*).
Linha de saída	Linha reservada para formação dos trens saída.
Passo de ordenação	Uma distribuição dos vagões vindos de uma linha de ordenação entre as linhas de ordenação disponíveis, sabendo previamente para qual linha cada vagão deve ser destinado. É composto de um *pull-out* e um *roll-in*.
Pátio de Classificação	Pátio ferroviário que permite que a operação de classificação de trens seja realizada.
Pátio Ferroviário	Área composta por linhas férreas e que pode ter diferentes funções, como por exemplo: embarque/desembarque de passageiros, carga/descarga, classificação de trens, reparo e manutenção, etc.

Plano de classificação	Descreve as operações (*roll-ins* e *pull-outs*) que os vagões que chegaram ao pátio devem efetuar para formar corretamente os trens de saída.
Plano de classificação viável	Plano de classificação que para uma determinada sequência de vagões de entrada forma os trens de saída de acordo com os requisitos pré-especificados.
Pull-out	Operação que "puxa" todos os vagões que estão em uma linha de ordenação e os conduz novamente a *lead track*, para uma nova operação de *roll-in*.
Rampa	Normalmente posicionada ao final da *lead track*, permite por meio da força da gravidade que os vagões sejam direcionados às linhas de ordenação sem a necessidade de serem "puxados" por uma locomotiva.
Roll-in	Operação que separa e direciona individualmente uma sequência de vagões que estão na *lead trak* para uma linhas de ordenação ou saída, utilizando *switches*. Nessa operação é necessário saber para qual linha o vagão será direcionado.
Trem de entrada	Trem que chegou ao pátio e deve passar pelo processo de classificação.
Trem de saída	Trem formado a partir da execução do processo de classificação e que partirá do pátio.

Apêndice A – Termo de Confidencialidade dos Dados da SBB

Pontifícia Universidade Católica do Paraná
Escola Politécnica
Programa de Pós Graduação em Informática

Confidentiality agreement between researcher and Swiss Railways (SBB)

Name: Prof. Edson Emilio Scalabrin (advisor from Denise Maria Vecino Sato master science student)

Thesis Title: Train Classification Problem: comparing different approaches

I, the undersigned, acknowledge, understand and agree to adhere to the following conditions of access.

- I will use the received data only for the purpose of the thesis.
- I will not disclose data or information to anyone other than those to whom I am authorised to do so.
- I will maintain the privacy and confidentiality of all accessible data and understand that unauthorised disclosure of personal/confidential data is an invasion of privacy.
- I will access data only for the purposes for which I am authorised explicitly. On no occasion will I use data, including personal or confidential information, for my personal interest or advantage, or for any other business purposes.
- I understand that where I have been given access to confidential information I am under a duty of confidence and would be liable under common law for any inappropriate breach of confidence in terms of disclosure to third parties and also for invasion of privacy if I were to access more information than that for which I have been given approval or for which consent is in place.
- I will provide all results of the thesis to SBB free for change, inducing any part achieved using the received data.
- I will inform SBB in advance about all publications referring the thesis result. I will also send at least two copies of each publication to SBB.

Prof. Edson Emilio Scalabrin
Pontifical Catholic University of Paraná, Brazil

Date: 2013/03/07

Apêndice B - Panorama do Transporte Ferroviário no Brasil

O Instituto de Pesquisa Econômica Aplicada (Ipea) define uma ferrovia como um "caminho de ferro", formado por trilhos paralelos de aço, assentados sobre dormentes de madeira, concreto e outros materiais. Após 15 anos da implantação do modelo de concessões no setor ferroviário, o transporte ferroviário no Brasil tem aumentado sua participação de forma considerável. O resultado da evolução do setor pode ser verificado pelo crescimento da produção ferroviária, como mostrado na Figura 22.

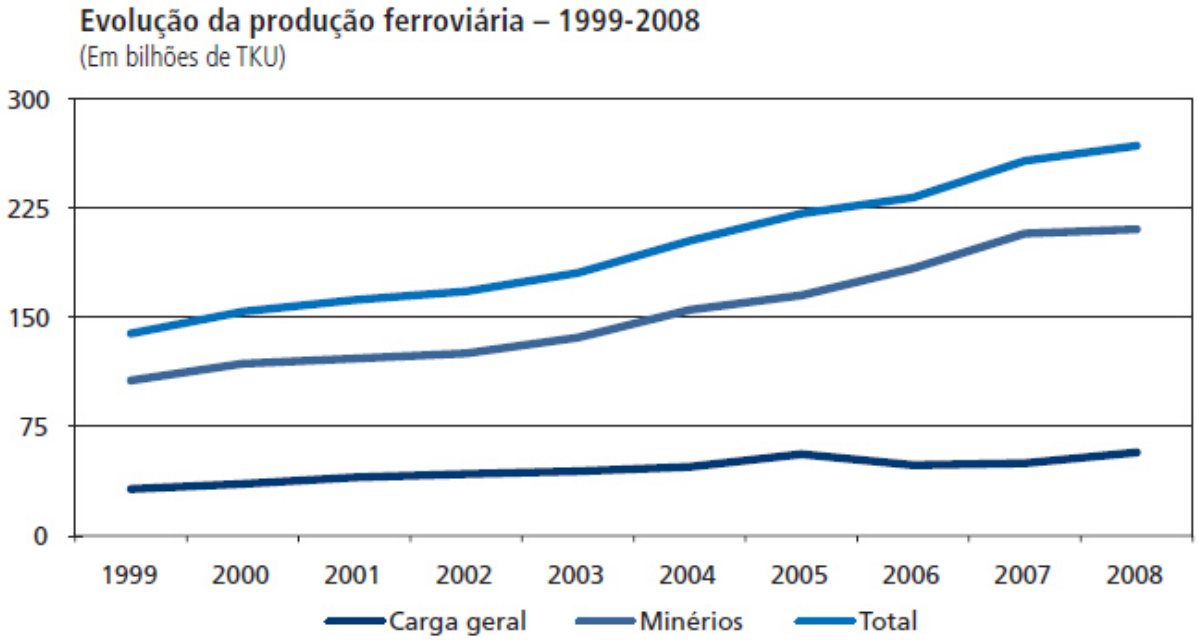

Figura 22 – Evolução da produção ferroviária (1999-2008)

Fonte: Eixos do Desenvolvimento Brasileiro – Transporte Ferroviário de Cargas – IPEA (2010)

O Instituto de Pesquisa Econômica Aplicada (Ipea), fundação pública federal vinculada à Secretaria de Assuntos Estratégicos da Presidência da República do Brasil, efetuou um amplo estudo sobre os desafios e oportunidades do desenvolvimento nacional brasileiro. Nesse estudo foram analisados diferentes setores da infraestrutura econômica do país que exercem impacto direto sobre as empresas e indústrias e, entre eles, o setor de transporte ferroviário de cargas.

A infraestrutura de transportes é um fator importante no desenvolvimento social e econômico de um país, além de influenciar na integração regional. Em um país como o Brasil,

de dimensão continental (8.514.876,599 km^2 de área[1]), essa influência torna-se ainda maior, visto que a infraestrutura de transportes afeta diretamente o custo da produção nacional.

Apesar de o transporte ferroviário ser utilizado massivamente em países com grande extensão territorial como Rússia, Canadá e Estados Unidos (como pode ser observado na Figura 23), no Brasil o principal modal de transporte utilizando ainda é o rodoviário. De acordo com o Ipea (2010), as dimensões dos países listados na Figura 23 são: Rússia – 17,08 milhões de km^2; Canadá – 9,98 milhões de km^2; Estados Unidos – 9,63 milhões de km^2; Brasil – 8,51 milhões de km^2; Austrália – 7,74 milhões de km^2; e México – 1,96 milhão de km^2.

Pode-se perceber uma desproporção no uso de ferrovias no Brasil, que ainda aproveita pouco as vantagens competitivas do transporte ferroviário. Além da extensa dimensão territorial, se o Brasil ampliasse o uso de sua infraestrutura ferroviária poderia também usufruir mais do transporte aquaviário, visto que boa parte das ferrovias brasileiras é destinada aos portos.

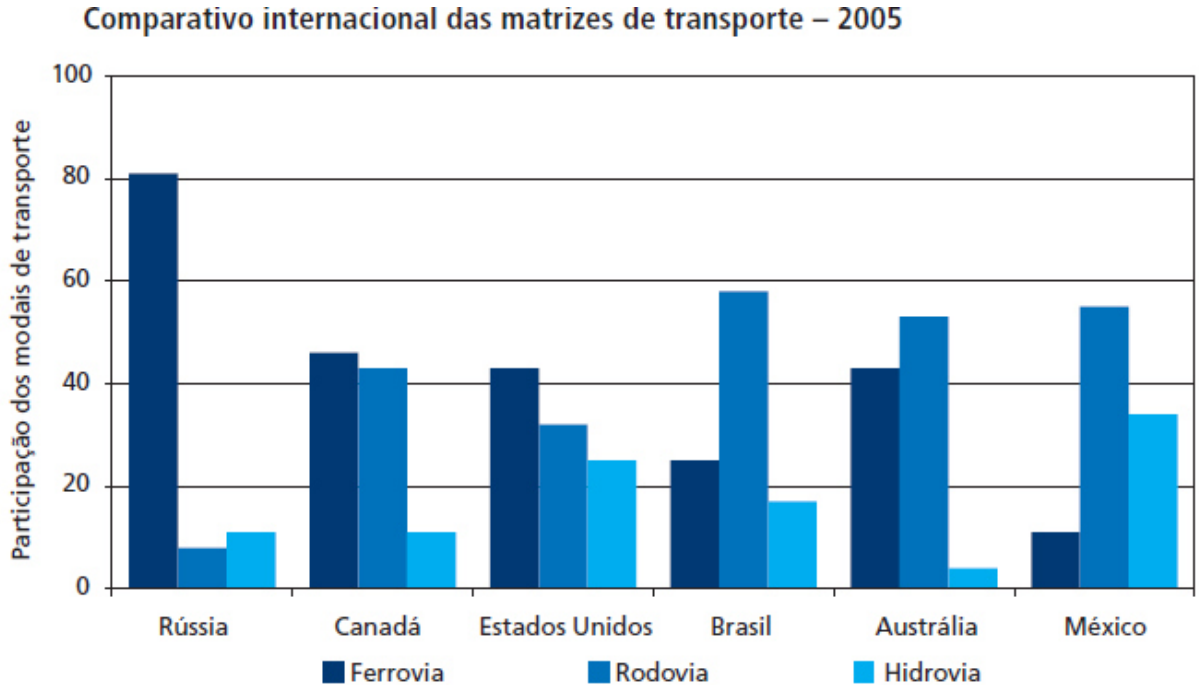

Figura 23 – Comparativo internacional das matrizes de transporte (2005)

Fonte: Eixos do Desenvolvimento Brasileiro – Transporte Ferroviário de Cargas – IPEA (2010)

[1] Fonte IBGE: (Publicado no Diário Oficial da União Nº 198 - Seção 1, de 11/10/2002, p. 48 à 65)

De acordo com a Agência Nacional de Transportes Ferroviários (ANTF), o transporte ferroviário caracteriza-se por sua capacidade de transporte de grandes volumes, com uma elevada eficiência energética, principalmente nos deslocamentos envolvendo médias e grandes distâncias. Além disso, também apresenta menor índice de acidentes, furtos e roubos do que o modal rodoviário. Portanto, pode-se perceber que o Brasil ainda pode usufruir muito mais desse modal de transporte.

Um estudo de mercado efetuado pela Revista Ferroviária destacou que em 2012 a malha ferroviária brasileira ganhará mais 3,1 mil km. Além disso, até 2015 (segundo a mesma fonte), estima-se que serão entregues mais 616,3 km de malha ferroviária. Além dos investimentos na ampliação da malha ferroviária é necessário usufruir ao máximo a infraestrutura atual.

Segundo (REIS, 2007), o transporte ferroviário possui um custo fixo elevado, devido ao arrendamento da malha, terminais e materiais rodantes e um custo variável (mão de obra, combustível e energia) relativamente baixo. Devido a isso, a escala exerce fator fundamental nesse tipo de transporte para diluir os custos fixos e aumentar a margem de lucro da ferrovia. Portanto, uma das formas de melhorar as margens de lucro no transporte ferroviário é aumentar sua capacidade operacional. Os pátios ferroviários em geral estão diretamente relacionados a capacidade operacional da ferrovia, pois definem uma capacidade de trens que podem receber e despachar, sendo assim um dos limitadores da capacidade da malha ferroviária. Além disso, especificamente os pátios de classificação em sua própria finalidade buscam reduzir os custos operacionais, pois quando a demanda para um determinado destino não é suficiente, composições de trens são agregadas para compartilhar trechos de viagem, e posteriormente classificadas e reorganizadas nos pátios de classificação.

Apêndice C – XML contendo o OMTC

```
<?xml version="1.0" encoding="UTF-8"?>
<instance>
  <presentation name="trem_classification_problem" maxConstraintArity="2"
maximize="false" format="XCSP 2.1_FRODO" />
  <domains>
    <domain name="decimais" nbValues="15">1..15</domain>
  </domains>
  <agents>
    <agent name="agent_T1" />
    <agent name="agent_T2" />
  </agents>
  <variables>
    <variable name="V1_1_decimal" agent="agent_T1" domain="decimais" />
    <variable name="V2_1_decimal" agent="agent_T1" domain="decimais" />
    <variable name="V3_1_decimal" agent="agent_T1" domain="decimais" />
    <variable name="V4_1_decimal" agent="agent_T1" domain="decimais" />
    <variable name="V5_1_decimal" agent="agent_T1" domain="decimais" />
    <variable name="V6_1_decimal" agent="agent_T1" domain="decimais" />
    <variable name="V7_1_decimal" agent="agent_T1" domain="decimais" />
    <variable name="V8_1_decimal" agent="agent_T1" domain="decimais" />
    <variable name="V8_2_decimal" agent="agent_T1" domain="decimais" />
    <variable name="V9_1_decimal" agent="agent_T1" domain="decimais" />
    <variable name="V10_1_decimal" agent="agent_T1" domain="decimais" />
    <variable name="V11_1_decimal" agent="agent_T2" domain="decimais" />
    <variable name="V12_1_decimal" agent="agent_T2" domain="decimais" />
    <variable name="V13_2_decimal" agent="agent_T2" domain="decimais" />
    <variable name="V13_1_decimal" agent="agent_T2" domain="decimais" />
    <variable name="V14_1_decimal" agent="agent_T2" domain="decimais" />
    <variable name="V15_1_decimal" agent="agent_T2" domain="decimais" />
  </variables>
  <predicates nbPredicates="2">
    <predicate name="P0">
      <parameters>int X0 int X1</parameters>
      <expression>
        <functional>gt(X0, X1)</functional>
      </expression>
    </predicate>
    <predicate name="P1">
      <parameters>int X0 int X1</parameters>
      <expression>
        <functional>ge(X0, X1)</functional>
      </expression>
    </predicate>
  </predicates>
  <functions>
    <function name="costFunction" return="int">
      <parameters>int X0</parameters>
      <expression>
        <functional>add(add(add(mod(X0, 2), mod(div(X0, 2), 2)), mod(div(X0,
pow(2, 2)), 2)), mod(div(X0, pow(2, 3)), 2))</functional>
      </expression>
    </function>
  </functions>
  <constraints>
    <constraint name="cost_1_1_decimal" arity="1" reference="costFunction"
scope="V1_1_decimal">
```

```
      <parameters>V1_1_decimal</parameters>
    </constraint>
    <constraint name="cost_2_1_decimal" arity="1" reference="costFunction"
scope="V2_1_decimal">
      <parameters>V2_1_decimal</parameters>
    </constraint>
    <constraint name="cost_3_1_decimal" arity="1" reference="costFunction"
scope="V3_1_decimal">
      <parameters>V3_1_decimal</parameters>
    </constraint>
    <constraint name="cost_4_1_decimal" arity="1" reference="costFunction"
scope="V4_1_decimal">
      <parameters>V4_1_decimal</parameters>
    </constraint>
    <constraint name="cost_5_1_decimal" arity="1" reference="costFunction"
scope="V5_1_decimal">
      <parameters>V5_1_decimal</parameters>
    </constraint>
    <constraint name="cost_6_1_decimal" arity="1" reference="costFunction"
scope="V6_1_decimal">
      <parameters>V6_1_decimal</parameters>
    </constraint>
    <constraint name="cost_7_1_decimal" arity="1" reference="costFunction"
scope="V7_1_decimal">
      <parameters>V7_1_decimal</parameters>
    </constraint>
    <constraint name="cost_8_1_decimal" arity="1" reference="costFunction"
scope="V8_1_decimal">
      <parameters>V8_1_decimal</parameters>
    </constraint>
    <constraint name="cost_8_2_decimal" arity="1" reference="costFunction"
scope="V8_2_decimal">
      <parameters>V8_2_decimal</parameters>
    </constraint>
    <constraint name="cost_9_1_decimal" arity="1" reference="costFunction"
scope="V9_1_decimal">
      <parameters>V9_1_decimal</parameters>
    </constraint>
    <constraint name="cost_10_1_decimal" arity="1" reference="costFunction"
scope="V10_1_decimal">
      <parameters>V10_1_decimal</parameters>
    </constraint>
    <constraint name="2_1gt1_1" arity="2" scope="V2_1_decimal V1_1_decimal"
reference="P0">
      <parameters>V2_1_decimal V1_1_decimal</parameters>
    </constraint>
    <constraint name="3_1gt2_1" arity="2" scope="V3_1_decimal V2_1_decimal"
reference="P0">
      <parameters>V3_1_decimal V2_1_decimal</parameters>
    </constraint>
    <constraint name="4_1gt3_1" arity="2" scope="V4_1_decimal V3_1_decimal"
reference="P0">
      <parameters>V4_1_decimal V3_1_decimal</parameters>
    </constraint>
    <constraint name="5_1gt4_1" arity="2" scope="V5_1_decimal V4_1_decimal"
reference="P0">
      <parameters>V5_1_decimal V4_1_decimal</parameters>
    </constraint>
    <constraint name="6_1gt5_1" arity="2" scope="V6_1_decimal V5_1_decimal"
```

```
reference="P0">
      <parameters>V6_1_decimal V5_1_decimal</parameters>
    </constraint>
    <constraint name="7_1gte6_1" arity="2" scope="V7_1_decimal V6_1_decimal"
reference="P1">
      <parameters>V7_1_decimal V6_1_decimal</parameters>
    </constraint>
    <constraint name="8_1gt7_1" arity="2" scope="V8_1_decimal V7_1_decimal"
reference="P0">
      <parameters>V8_1_decimal V7_1_decimal</parameters>
    </constraint>
    <constraint name="8_2gte8_1" arity="2" scope="V8_2_decimal V8_1_decimal"
reference="P1">
      <parameters>V8_2_decimal V8_1_decimal</parameters>
    </constraint>
    <constraint name="9_1gt8_1" arity="2" scope="V9_1_decimal V8_1_decimal"
reference="P0">
      <parameters>V9_1_decimal V8_1_decimal</parameters>
    </constraint>
    <constraint name="9_1gt8_2" arity="2" scope="V9_1_decimal V8_2_decimal"
reference="P0">
      <parameters>V9_1_decimal V8_2_decimal</parameters>
    </constraint>
    <constraint name="10_1gt9_1" arity="2" scope="V10_1_decimal V9_1_decimal"
reference="P0">
      <parameters>V10_1_decimal V9_1_decimal</parameters>
    </constraint>
    <constraint name="cost_11_1_decimal" arity="1" reference="costFunction"
scope="V11_1_decimal">
      <parameters>V11_1_decimal</parameters>
    </constraint>
    <constraint name="cost_12_1_decimal" arity="1" reference="costFunction"
scope="V12_1_decimal">
      <parameters>V12_1_decimal</parameters>
    </constraint>
    <constraint name="cost_13_2_decimal" arity="1" reference="costFunction"
scope="V13_2_decimal">
      <parameters>V13_2_decimal</parameters>
    </constraint>
    <constraint name="cost_13_1_decimal" arity="1" reference="costFunction"
scope="V13_1_decimal">
      <parameters>V13_1_decimal</parameters>
    </constraint>
    <constraint name="cost_14_1_decimal" arity="1" reference="costFunction"
scope="V14_1_decimal">
      <parameters>V14_1_decimal</parameters>
    </constraint>
    <constraint name="cost_15_1_decimal" arity="1" reference="costFunction"
scope="V15_1_decimal">
      <parameters>V15_1_decimal</parameters>
    </constraint>
    <constraint name="12_1gt11_1" arity="2" scope="V12_1_decimal V11_1_decimal"
reference="P0">
      <parameters>V12_1_decimal V11_1_decimal</parameters>
    </constraint>
    <constraint name="13_2gte12_1" arity="2" scope="V13_2_decimal V12_1_decimal"
reference="P1">
      <parameters>V13_2_decimal V12_1_decimal</parameters>
    </constraint>
```

```
    <constraint name="13_1gt13_2" arity="2" scope="V13_1_decimal V13_2_decimal"
reference="P0">
      <parameters>V13_1_decimal V13_2_decimal</parameters>
    </constraint>
    <constraint name="14_1gt13_1" arity="2" scope="V14_1_decimal V13_1_decimal"
reference="P0">
      <parameters>V14_1_decimal V13_1_decimal</parameters>
    </constraint>
    <constraint name="15_1gte14_1" arity="2" scope="V15_1_decimal V14_1_decimal"
reference="P1">
      <parameters>V15_1_decimal V14_1_decimal</parameters>
    </constraint>
  </constraints>
</instance>
```

Apêndice D – Simulação do plano de classificação

A seguir são apresentadas as telas com o estágio final de cada passo de ordenação simulado para o plano de classificação gerado pelo processo simplificado. Nesse processo o plano de classificação gerado é viável e ótimo, porém sem nenhuma restrição adicional. O OMTC é enviado uma única vez ao *framework* FRODO que gera o plano de classificação mostrado à esquerda nas telas.

Essas telas são resultado da simulação do cenário apresentado na Figura 5.

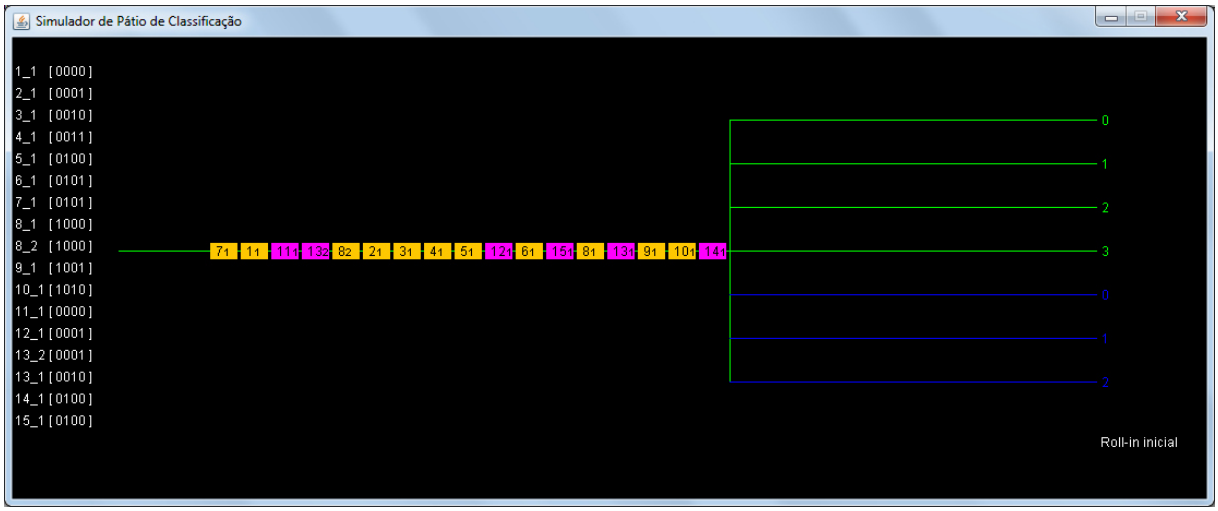

Figura 24 (a) – Simulação do plano de classificação – cenário inicial

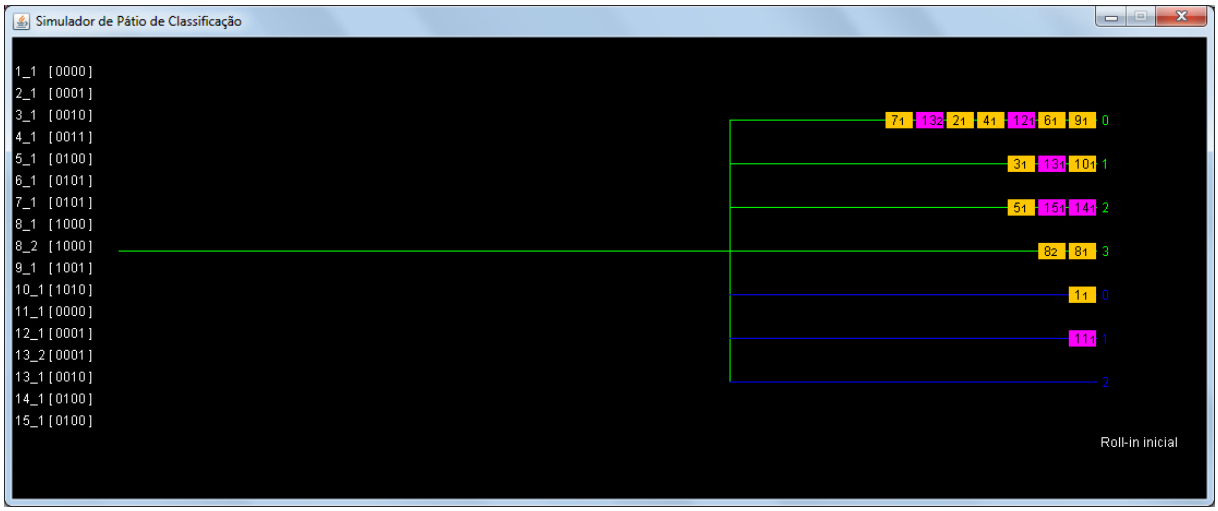

Figura 24 (b) – Simulação do plano de classificação – *roll-in* inicial

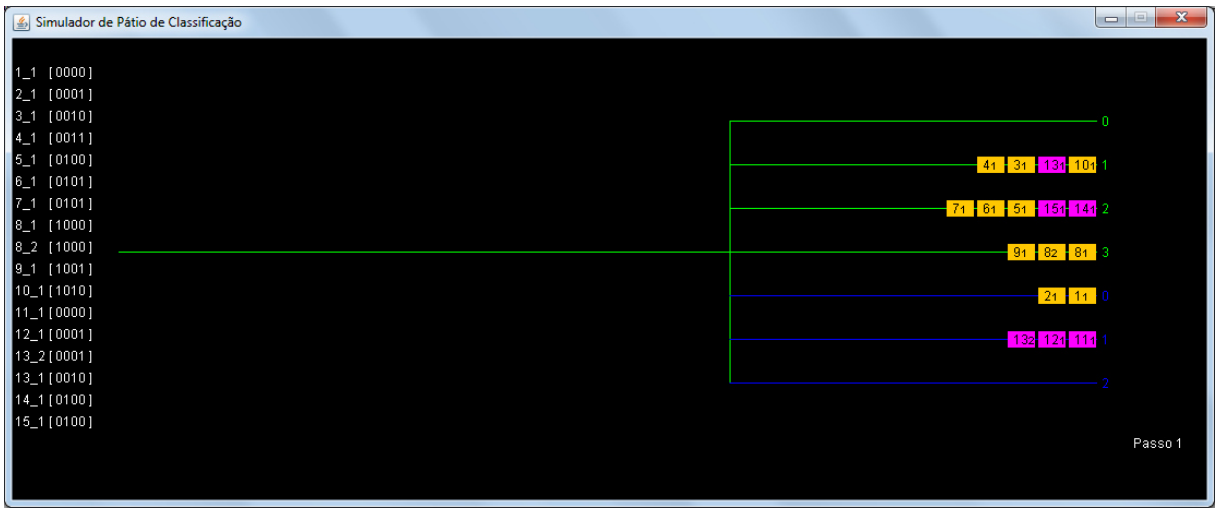

Figura 24 (c) – Simulação do plano de classificação – 1º passo de ordenação

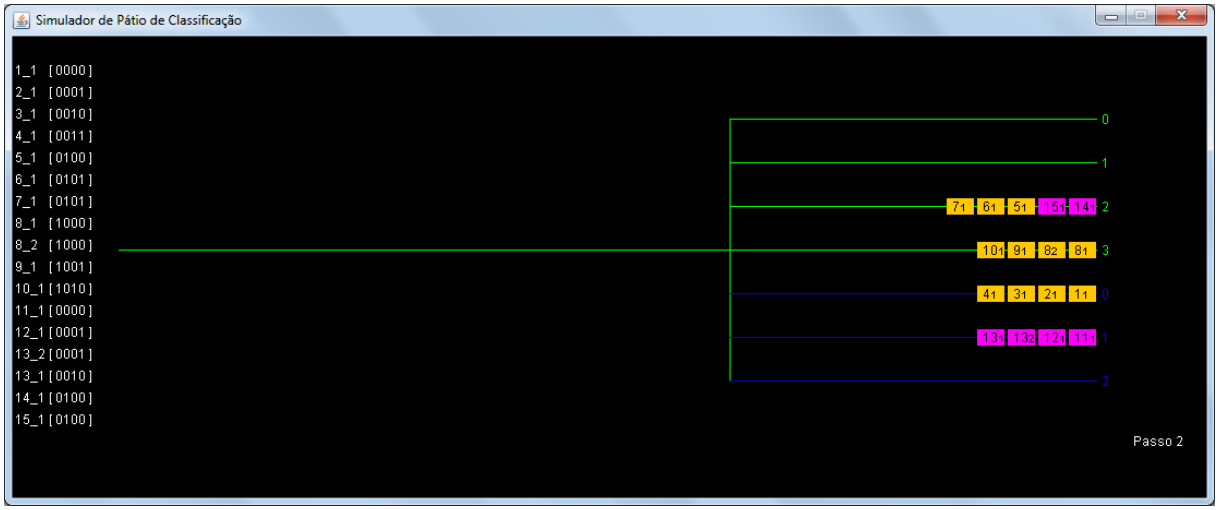

Figura 24 (d) – Simulação do plano de classificação – 2º passo de ordenação

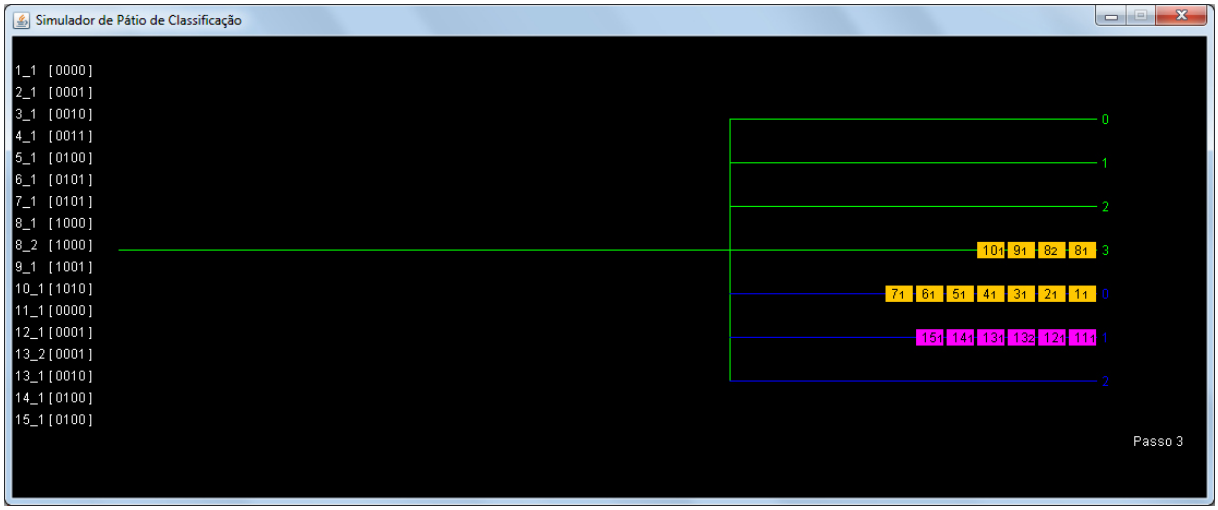

Figura 24 (e) – Simulação do plano de classificação – 3º passo de ordenação

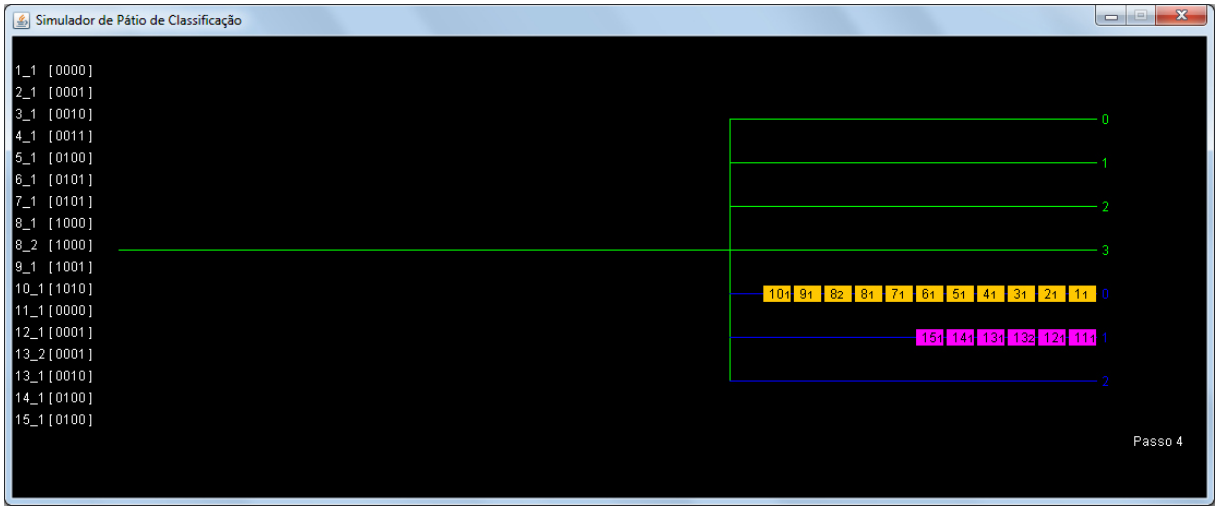

Figura 24 (f) – Simulação do plano de classificação – 4º passo de ordenação

Apêndice E - Estudo de Caso baseado nos dados do Pátio de Lausanne

Para avaliar o OMTC proposto para o problema de classificação de trens em um pátio ferroviário foram realizados experimentos utilizando cenários fictícios baseados nos dados reais de uma semana de tráfego no pátio de Lausanne na Suíça, fornecidos pela *SBB* e ocorridos no ano de 2005. O pátio de Lausanne é um pátio com rampa, composto de um pátio de recepção, uma área de classificação com 38 linhas, duas *lead tracks* paralelas e sem área de expedição. O pátio possui 10 linhas reservadas para o processo de classificação com capacidade $C = 40$ vagões. Todos os vagões que passam pelo *multistage sorting* são inicialmente direcionados para uma dessas 10 linhas de ordenação. As duas *lead tracks* paralelas permitem que em um mesmo passo de ordenação seja efetuado *pull-out* de duas linhas de ordenação, reduzindo assim a quantidade *h* de passos de ordenação pela metade.

Os dados cedidos continham informação sobre os vagões que chegaram ao pátio e o planejamento de trens que deveriam deixar o pátio, em planilhas eletrônicas, porém tais dados precisaram ser interpretados para a criação das instâncias de problema para os experimentos. Após essa interpretação, foi possível obter os vagões que passaram pelo *multistage sorting*, por dia de tráfego, conforme apresentado na Tabela 7, coluna *n* (número de vagões recebidos no dia - da 01:00 da manhã até 01:00 do dia seguinte), obtida diretamente das planilhas.

Tabela 7 – Resumo dos dados reais utilizados para experimentos
n: total de vagões, *l*: trens de entrada, *m:* trens de saída, *k:* tamanho da decomposição de cadeia, k_{total}: total de cadeias geradas, τ: quantidade de tipos

instância	n	l	m	n_{max}	k	k_{total}	τ
segunda	486	49	24	**60**	4	**46**	56
terça	329	44	24	**47**	4	42	55
quarta	310	47	**25**	31	4	44	54
quinta	364	44	**25**	46	3	41	55
final de semana	368	**44**	27	41	3	47	52

Outra informação da planilha, mas que teve uma diferença nos dados do final de semana é a quantidade de trens de entrada l. As demais informações referentes a cada dia de

tráfego não puderam ser obtidas de forma direta, e para tal foi necessário desenvolver um processo automatizado para organização e definição dessas informações, que pode ser visualizado de forma gráfica na Figura 25.

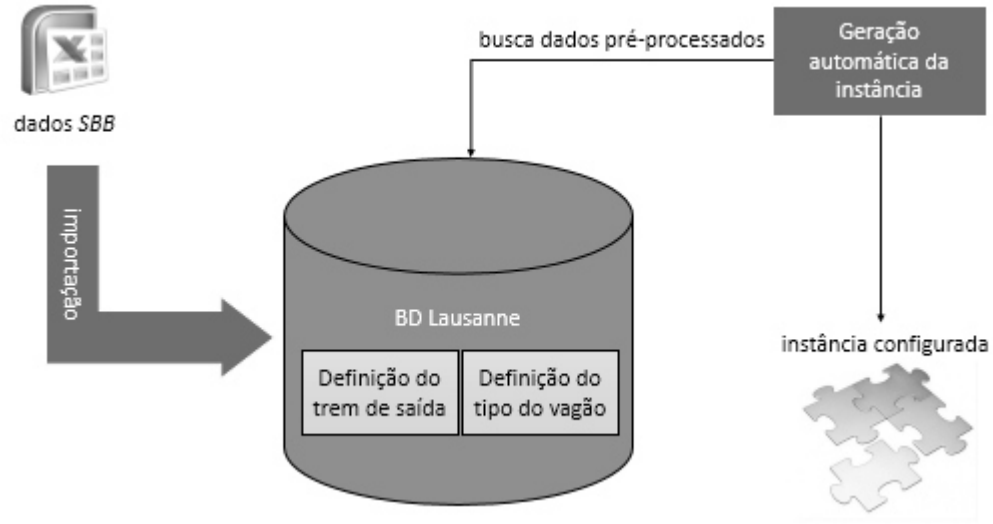

Figura 25 – Processo para geração automática da instância do problema

Apesar do prestado pelo Sr. Peter Márton, responsável por gerar uma simulação de tráfego para o pátio de Lausanne em 2005, foi necessário assumir algumas regras para designar o trem de saída e o tipo de cada vagão, dados imprescindíveis aos experimentos, e que não puderam ser obtidos diretamente a partir dos dados recebidos.

Para o processo de definição do trem de saída foi identificado no planejamento de trens de saída aqueles que possuíam em um de seus destinos o destino do vagão. No planejamento de trens que deixaram o pátio, cada trem de saída pode conter até 8 destinos. Esses trens foram considerados candidatos a trem de saída do vagão em questão. Entre essa primeira lista de trens candidatos, foram selecionados somente aqueles com partida planejada para depois da chegada do vagão ao pátio, gerando uma nova lista de trens de saída candidatos, entre esses foi escolhido o trem que no plano vai deixar antes o pátio. A quantidade de trens de saída por dia de tráfego está representada na coluna m da Tabela 7, bem como a quantidade de vagões do maior trem de saída (n_{max}).

Essas regras foram construídas a partir da análise dos dados recebidos e das informações obtidas em documentos e conversas com o Sr. Peter Márton. Apesar de todas as considerações, a quantidade de trens de saída (m) e o número de vagões do maior trem de

saída (n_{max}) não são iguais aos valores apresentados no trabalho de Maue (2011). Os valores diferentes estão destacados em negrito na Tabela 7.

Para gerar os dados da coluna k (indica o tamanho da decomposição de cadeia do cenário) e da coluna k_{total} (indica o total de cadeias do cenário), foi utilizado o algoritmo GCD, que faz parte do processo de execução OMTC. Como os trens de saída assinalados para os vagões do cenário não seguem exatamente os trens utilizados nos cenários de Maue (2011), a quantidade de cadeias também sofreu algumas alterações.

O processo de geração automática da instância do problema, configura o problema de acordo com o necessário para gerar a decomposição de cadeia pelo algoritmo GCD. Para configurar corretamente essas instâncias foram utilizados os seguintes dados: dia e horário de chegada do vagão ao pátio, trem de saída designado (de acordo com critérios descritos anteriormente), dia e horário de saída do trem designado e destino do vagão.