GILSON ATALIBA DE FARIA

DESENVOLVIMENTO DE HEURÍSTICAS PARA O PROBLEMA DE SEQUENCIAMENTO DE TAREFAS EM MÁQUINAS PARALELAS DE PROCESSAMENTO EM LOTE COM ENTREGAS

Dissertação apresentada à Universidade Federal de Viçosa, como parte das exigências do Programa de Pós-Graduação em Ciência da Computação, para obtenção do título de *Magister Scientiae*

VIÇOSA
MINAS GERAIS − BRASIL
2018

Ficha catalográfica preparada pela Biblioteca Central da Universidade Federal de Viçosa - Câmpus Viçosa

T

F224d
2018

Faria, Gilson Ataliba de, 1987-
Desenvolvimento de heurísticas para o problema de sequenciamento de tarefas em máquinas paralelas de processamento em lote com entregas / Gilson Ataliba de Faria. – Viçosa, MG, 2018.
ix, 76f. : il. ; 29 cm.

Orientador: José Elias Claudio Arroyo.
Dissertação (mestrado) - Universidade Federal de Viçosa.
Inclui bibliografia.

1. Programação heurística. 2. Logística. 3. Otimização combinatória. 4. Otimização matemática. I. Universidade Federal de Viçosa. Departamento de Informática. Programa de Pós-Graduação em Ciência da Computação. II. Título.

CDD 22. ed. 006.3

GILSON ATALIBA DE FARIA

DESENVOLVIMENTO DE HEURÍSTICAS PARA O PROBLEMA DE SEQUENCIAMENTO DE TAREFAS EM MÁQUINAS PARALELAS DE PROCESSAMENTO EM LOTE COM ENTREGAS

Dissertação apresentada à Universidade Federal de Viçosa, como parte das exigências do Programa de Pós-Graduação em Ciência da Computação, para obtenção do título de *Magister Scientiae.*

APROVADA: 06 de junho de 2018.

André Gustavo dos Santos
(Coorientador)

Thiago Henrique Nogueira
(Coorientador)

Heleno do Nascimento Santos

José Elias Cláudio Arroyo
(Orientador)

Dedico este trabalho a todos aqueles que de alguma forma contribuíram para minha formação pessoal e profissional.

"A cada dia que nasce, renasce o prazer de viver, de amar, de ser feliz. Renasce a esperança de dias melhores."

(Al Stevens)

Agradecimentos

Primeiramente a Deus, por ter me permitido chegar onde eu cheguei.

À minha mãe, Maria das Graças Monteiro de Faria, sempre atenciosa e preocupada comigo. Ao meu pai Pedro Pinheiro de Faria pelo apoio.

À minha esposa Érika Xavier Fidêncio, sempre compreensiva quanto às minhas ausências e em meus momentos de dificuldades, sempre me apoiando e me incentivando com todo seu amor.

Ao meu irmão Geraldo Lúcio de Faria e sua esposa Mariana Veronez Borri, amigos eternos com quem sempre posso contar e, claro, a reciproca é válida.

Ao meu sogro Odemir Afonso Fidêncio, à minha sogra Maria Efigênia Xavier Fidêncio e também às minhas cunhadas (irmãs) Aline Xavier Fidêncio e Karina Xavier Fidêncio por todo apoio e companheirismo.

Aos irmãos da República Jogatina.

Aos colegas da Universidade Federal de Viçosa, principalmente Jonatas Chagas que contribuiu muito para a execução deste trabalho.

Ao meu professor e orientador José Elias Cláudio Arroyo. E aos meus coorientadores André Gustavo dos Santos e Thiago Henrique Nogueira.

À Universidade Federal de Viçosa e ao Departamento de Informática.

A CAPES pelo financiamento deste projeto de vida.

Aos membros da banca por se apresentarem disponíveis para compor a minha banca de defesa de mestrado.

Sumário

Lista de Tabelas **vii**

Resumo **viii**

Abstract **ix**

1 Introdução **1**

1.1 Motivação 4

1.2 Contribuições 4

1.3 Organização do Documento 5

2 Revisão Bibliográfica **6**

2.1 Trabalhos Relacionados ao Problema de Sequenciamento de Máquinas Paralelas com Processamento em Lotes ($STMPL$) 8

2.2 Problema de Sequenciamento de Tarefas em Máquinas Paralelas de Processamento em Lotes ($STMPL$) com Entrega 11

3 Formulação do Problema e Métodos de Resolução Propostos **17**

3.1 Definição Formal do Problema 17

3.2 Formulação do Problema de Programação Linear Inteira Mista . . . 18

3.3 Estrutura de Solução 21

3.4 Um *Upper Bound* (UB) para as Soluções do Problema $STMPL$ com Entrega 22

3.5 Algoritmos Heurísticos Prospostos 23

3.5.1 Heurística NEH - Nawaz Enscore Ham 25

3.5.2 Heurística Heu3, baseada nas heurísticas propostas por Li et al. [2015] 27
3.5.3 Variable Neighborhood Descent (VND) 31
3.5.4 Simulated Annealing (SA) 37

4 Experimentos Computacionais **43**
4.1 Geração de Instâncias 43
4.2 Calibração dos Parâmetros do Algoritmo SA 47
4.3 Resultados Experimentais 52
4.3.1 Resultados Experimentais para as Instâncias da Família A . 52
4.3.2 Resultados Experimentais Tendo como Entrada as Instâncias da Família B 65

5 Considerações Finais **71**

Referências Bibliográficas **73**

Lista de Tabelas

4.1 Associação de cada Conjunto de Instâncias Base para a Família A de Instâncias com seu Respectivo ID. 45
4.2 Características das Instâncias da Família A 46
4.3 Características das Instâncias da Família B. 47
4.4 Combinações de Parâmetros Testadas na Calibração 47
4.5 Resultados do Modelo para as intancias da família A 53
4.6 Tabela de Resultados da Heurística SN, passo a passo, para as instâncias da familia A 59
4.7 Valores de FO obtidos pelas heurísticas SN, SA e o Modelo para as instâncias da família A 68
4.8 Valores de FO obtidos pelas heurísticas SN, SA e o Modelo para as instâncias da família A - continuação da tabela 4.7 69
4.9 Valores de FO obtidos pelas heurísticas SN e para as instâncias da família B 70

Resumo

FARIA, Gilson Ataliba de, M.Sc., Universidade Federal de Viçosa, julho de 2018. **Desenvolvimento de Heurísticas para o Problema de Sequenciamento de Tarefas em Máquinas Paralelas de Processamento em Lote com Entregas**. Orientador: José Elias Claudio Arroyo. Coorientadores: Thiago Henrique Nogueira e André Gustavo dos Santos.

Este trabalho consiste no estudo do problema de sequenciamento de tarefas com tempos de processamento distintos em máquinas paralelas idênticas que processam mais de uma tarefa simultaneamente, sendo as tarefas sequenciadas nos veículos capacitados que estão disponíveis para realizar as entregas em momentos específicos pós processamento. O objetivo do problema é determinar o sequenciamento das tarefas que maximize o ganho da empresa, sabido que uma tarefa só poderá ser sequenciada se seu prazo de entrega for respeitado. Neste trabalho, propõe se, primeiramente, uma formulação de Programação Linear Inteira Mista (PLIM) para o problema. Em seguida, propõe-se uma relaxação do método heurístico citado na literatura para determinar um limite superior (UB) para a solução. Na sequência aplica-se uma heurística construtiva, para gerar uma solução viável baseada na solução de UB. Por fim, são propostas diversas heurísticas de melhoria cujo princípio de funcionamento se baseia nas heurísticas *Variable Neighborhood Descent*(VND) e *Simulated Annealing* (SA), com o objetivo de determinar soluções de melhor qualidade. Os resultados obtidos mostraram a eficiência dos métodos heurísticos implementados em relação ao limite superior (UB).

Abstract

FARIA, Gilson Ataliba de, M.Sc., Universidade Federal de Viçosa, July, 2018. **Development of Heuristics for the Job Scheduling Problem on Parallel Batching Machines with Deliveries**. Adviser: José Elias Claudio Arroyo. Co-advisers: Thiago Henrique Nogueira and André Gustavo dos Santos.

This work deals with the job scheduling problem on identical parallel-batch machines and where the jobs have different processing times. When the jobs processing time finish, the jobs are sequenced in vehicles with limited capacity that are available to deliver at specific times. The goal of the problem is to determine the sequencing of tasks that maximizes the company's profit, knowing that a job can only be sequenced if its delivery deadline is respected. In this work it was first proposed a Mixed Integer Linear Programming for the problem. Then it was proposed a relaxation of the heuristic method mentioned in the literature to determine an upper bound (UB) for the solution. A constructive heuristic was then applied to generate a viable solution based on the UB solution. Finally,several heuristics of improvements were proposed, whose operating principle is based on the heuristic VND and on the heuristic Simulated Annealing (SA), aiming to determine better quality solutions. The obtained results showed the efficiency of the heuristic methods implemented in relation to the upper bound (UB).

Capítulo 1

Introdução

Com a globalização e o rápido desenvolvimento tecnológico, o mercado tem se tornado cada vez mais dinâmico e competitivo. Segundo Matias & Nascimento [2013], as constantes mudanças e evoluções do mundo globalizado fazem com que o ganho de eficiência seja um dos grandes desafios atuais, pois é cada vez menor a capacidade das empresas de influenciar o preço de seus produtos e/ou serviços com total liberdade. Dessa maneira, a permanência de uma determinada empresa no atual mercado consumidor está diretamente e fortemente ligada à eficiência de seus processos logísticos.

De acordo com Chang & Lee [2004], com a popularidade dos conceitos de *Just in Time* [Correa et al., 1996], sistema de administração onde a produção é realizada sob demanda de forma a minimizar os custos de estocagem, as empresas buscam tratar de maneira integrada diversos estados da cadeia de suprimentos na busca por níveis de inventários mais competitivos.

O planejamento e controle da produção surge neste contexto para auxiliar na melhoria da eficiência dos processos das empresas. Segundo Tubino [2006], o setor de planejamento e controle da produção de uma empresa tem como responsabilidade planejar e controlar o uso dos recursos produtivos de maneira eficiente.

A programação da produção é uma das funções exercidas pelo setor de planejamento e controle da produção. Segundo Correa et al. [2016], a função de programação da produção trata-se da organização de tarefas, por prioridades, a serem processadas em uma linha de produção. A priorização está relacionada

ao uso eficiente de recursos produtivos (máquinas, mão-de-obra, matéria-prima, tempo, instalações, insumos produtivos, etc.) limitados, a importância estratégica do cliente que possui demanda pelas tarefas e as atividades com prazos de entrega mais apertados.

A programação da produção, segundo Tubino [2006], pode ser dividida em três grupos: administração de estoques; sequenciamento e emissão; e liberação de ordens de compra, fabricação e montagem. Todos os grupos podem ser estudados com detalhe na obra do mesmo autor. Ainda de acordo com Tubino [2006], o sequenciamento trata-se da adequação das necessidades de produção aos recursos disponíveis (máquinas, homens e instalações).

Dentro do assunto sequenciamento de tarefas, observa-se o aumento da atenção dos pesquisadores frente ao Problema de Sequenciamento de Tarefas em Máquinas Paralelas de Processamento em Lote (STMPL). Segundo Leung & Chen [2013], a importância do estudo do problema de sequenciamento com entrega de tarefas vem ganhando importância à medida que cresce o número de empresas que adotam a produção por encomenda. Esse modelo de negócio é representado pela produção sobre medida, onde os produtos são entregues aos clientes, saindo diretamente da fábrica, de forma a minimizar o custo de estocagem.

De acordo com Li et al. [2015], os primeiros a estudarem o problema de $STMPL$ foram Ikura & Gimple [1986]. Desde então, observam-se diversos trabalhos na mesma linha, como: Damodaran & Chang [2008], Kashan et al. [2008], Jia & Leung [2015], Fidelis & Arroyo [2016], Fidelis & Arroyo [2017], entre outros.

Além do aumento dos estudos sobre o problema de STMPL, observa-se o interesse de autores quanto a sua integração com pós processamento, que seria a etapa de agendamento de entregas. Dentre os trabalhos que trataram o assunto estão Leung & Chen [2013], Agnetis et al. [2014], Cheng et al. [2015] e Li et al. [2015].

Li et al. [2015], utilizado como principal referência para a elaboração deste trabalho, definem o problema de STMPL com entrega, tal que, n tarefas devem ser processadas em m máquinas idênticas e paralelas de processamento em lotes. O processamento em lotes é caracterizado pelo processamento de um uma ou mais tarefas simultaneamente, de forma que o limite de capacidade K de cada máquina seja respeitada. Quanto ao pós processamento, tem-se disponíveis z momentos

de entrega para os quais cada tarefa poderá ser designada. Da mesma forma, cada momento de entrega apresenta uma série de veículos com capacidade C (de forma que as tarefas devam ser alocadas em lotes de tamanho C) disponíveis para a realização das entregas. Para o sequenciamento nas máquinas e momentos de entrega, deve-se respeitar o prazo limite de entrega para cada tarefa, de forma que o tempo de conclusão do lote designado para a tarefa e o seu momento de entrega deva ser menor ou igual a data limite. Caso a data limite seja respeitada, a tarefa sequenciada participa dos lucros da empresa. O problema trata-se da maximização destes lucros. Li et al. [2015] tratam o problema considerando a capacidade com um valor K independente da máquina e C independente do veículo e tempo de processamento de cada tarefa igual a p independente da máquina e da tarefa. Os mesmos autores citam como trabalho em aberto o estudo do mesmo problema quando as tarefas possuírem tempos de processamento diferentes. Eles propuseram uma heurística para solucionar o problema considerando as tarefas com tamanhos unitários e também propuseram duas heurísticas para resolver o problema assumindo que as tarefas apresentem tamanhos distintos. Os autores ainda propuseram a relaxação das próprias heurísticas propostas para obtenção de um limite superior (*Upper Bound* - UB) para a solução do problema.

O Problema de Sequenciamento de Tarefas em Máquinas Paralelas de Processamento em Lotes (STMPL) com Entrega é um problema NP-Difícil (ver [Li et al., 2015]) que apresenta grande importância prática. Segundo Fuchigami & Rangel [2014], o sequenciamento de tarefas está diretamente relacionado à otimização de diversas medidas de desempenho que visam o bom funcionamento do sistema e a satisfação dos clientes. Segundo Damodaran & Chang [2008], este problema pode ser encontrado em testes de circuitos elétricos e trabalhos com metal. O problema pode se apresentar também nas indústrias de semi-condutores, farmacêutica, aeronáutica e de logística (Mathirajan & Sivakumar [2006] citado por Li et al. [2015]).

1.1 Motivação

A motivação por estudar o problema de STMPL com Entrega é dada pela importância do problema, visto o grande e crescente número de aplicações do problema. Como é apontado por Leung & Chen [2013], o referido problema vem ganhando importância à medida que cresce o número de empresas que adotam a produção por encomenda. Esse modelo de negócio é representado pela produção sobre medida, onde os produtos são entregues aos clientes, saindo diretamente da fábrica. O fenômeno já vem ocorrendo há algum tempo, o que frisa a importância de estudar o tema.

Visto a relevância do problema propõe-se neste trabalho desenvolver heurísticas que permitam encontrar uma solução para o STMPL com Entrega, considerando tempos de processamento de tarefas diferentes entre si, tema sugerido por Li et al. [2015] como trabalho futuro.

1.2 Contribuições

Este trabalho trata o problema STMPL com Entregas considerando um caso mais genérico, considerando o tempo de processamento das tarefas diferentes entre si. Primeiramente é formulado um Modelo de Programação Linear Inteira Mista para o problema. Em seguida, utiliza-se um método heurístico relaxado proposto em Li et al. [2015] para determinar um limite superior (UB) para a solução, e juntamente uma heurística construtiva para a obtenção de uma solução inicial. Por fim, propõe-se heurísticas baseadas no *Variable Neighborhood Descent* (VND) e no *Simulated Annealing* (SA), com o objetivo de determinar soluções de melhor qualidade, tendo como entrada a solução gerada pela heurística construtiva. A partir dos resultados alcançados pode-se validar a heurística de obtenção do limite superior (UB) e comprovar a eficiência dos métodos heurísticos para melhoria de solução quando comparados ao limite superior (UB).

1.3 Organização do Documento

Além deste primeiro capítulo que faz uma breve introdução sobre o tema e apresenta a motivação do trabalho e as suas contribuições, esta dissertação está dividida em mais quatro capítulos. O Capítulo 2 apresenta uma revisão bibliográfica sobre o tema abordado. O Capítulo 3 apresenta a definição formal do problema, a proposição de um modelo de programação linear inteira mista e métodos heurísticos para solucionar o problema. No Capítulo 4, apresenta-se os experimentos realizados e os resultados alcançados. Finalmente, no Capítulo 5, são apresentadas as conclusões do trabalho e as sugestões para trabalhos futuros.

Capítulo 2

Revisão Bibliográfica

Com o objetivo de entender e caracterizar melhor o Problema de Sequenciamento de Tarefas em Máquinas Paralelas de Processamento em Lotes ($STMPL$) com Entrega, a seguir é realizada uma breve discussão sobre definições importantes e características de outros problemas semelhantes já tratados na literatura.

Pinedo [2005] define o sequenciamento de tarefas como a aplicação de modelos matemáticos e métodos heurísticos para identificar a melhor forma de alocar uma quantidade limitada de recursos para o processamento de uma série de tarefas demandadas. Sendo considerados como recursos, por exemplo, as máquinas em uma linha de produção, pistas em um aeroporto ou até unidades de processamento em um ambiente de desenvolvimento computacional. Quanto às tarefas pode-se considerar as operações ou produtos a serem processados por uma linha de produção, embarques e desembarques em um aeroporto, programas de computador que devem ser executados por um processador, dentre outras.

Segundo Brucker [2006], sendo um pouco mais formal, o problema de sequenciamento trata-se do estudo da melhor maneira de processar um conjunto de tarefas I_i $(i = 1, ..., n)$ em um conjunto de máquinas M_l $(l = 1, ..., m)$ de forma a satisfazer uma série de restrições técnicas. A cada tarefa J_i está associado um tempo de processamento p_{il} em cada máquina M_l e o momento em que a tarefa se torna disponível para o processamento, dado pela sua data de lançamento Q_i.

Brucker [2006] também define outros parâmetros que podem ser parte integrante de problemas de sequenciamento, são eles: data definida para finalização

da tarefa d_i e o peso R_i que mede o ganho relativo ao cumprimento do prazo.

Segundo Pinedo [2005], a melhor alocação de recursos limitados para o processamento de um determinado conjunto de tarefas deve ser feita de forma que se possibilite o alcance dos objetivos dos interessados de forma otimizada e eficiente. O autor supracitado apresenta como exemplo as seguintes possibilidades de objetivos para o problema de sequenciamento de tarefas: minimizar o tempo necessário para que todas as tarefas sejam totalmente processadas; minimizar o número de atividades que não foram completamente trabalhadas dentro do prazo estabelecido para entrega. Além dessas, Brucker [2006] citam como critério de otimalidade de uso mais comum na literatura a minimização do *makespan*. Dado que o tempo necessário para concluir a tarefa J_i seja representado por u_i, tem-se o *makespan* como $\max\{u_i \mid i = 1, ..., n\}$. Outro critério seria a minimização do tempo total de fluxo que é dado por: $\sum_{i=1}^{n} u_i$.

De acordo Müller et al. [2002] o sequenciamento de máquinas paralelas idênticas se dá quando os tempos de execução de cada tarefa permanecem o mesmo não importando para qual máquina a mesma tarefa seja atribuída.

Conforme Ikura & Gimple [1986] a produção em lote é aquela que permite o processamento de mais de uma tarefa ao mesmo tempo, em uma mesma máquina. Isso porque as tarefas são agrupadas antes de serem destinadas ao processamento em uma determinada máquina.

Segundo Leung & Chen [2013] o problema do agendamento de entregas trata-se do agrupamento de um conjunto de tarefas, pós processamento, para que seja entregue aos clientes. Para a realização das entregas tem-se um número fixo de veículos, cada um possuindo a capacidade de transportar um número limitado de tarefas. Desta forma, é necessário o agrupamento de tarefas em paletes de tamanho pré estabelecidos de forma que não se ultrapasse as limitações dos veículos para entrega.

Entendido os conceitos básicos, pode-se fazer uma breve descrição de trabalhos correlatos ao problema de STMPL e a integração do problema de $STMPL$ com o agendamento de entregas. Discussão necessária para um maior aprofundamento do assunto antes de tratar o problema estudado neste trabalho. As seções contidas neste capitulo apresentam alguns trabalhos publicados na literatura sobre os respectivos assuntos.

2.1 Trabalhos Relacionados ao Problema de Sequenciamento de Máquinas Paralelas com Processamento em Lotes ($STMPL$)

Ikura & Gimple [1986] definiram o problema de sequenciamento de tarefas em lotes considerando uma única máquina de processamento e n tarefas a serem sequenciadas. As tarefas devem ser processadas em lotes, sendo o tamanho de cada lote limitado, ou seja, a dimensão do lote não pode ultrapassar o limite de capacidade da máquina M. Essa capacidade é determinada por um valor inteiro K, dado pelo número máximo de tarefas que podem ser processadas ao mesmo tempo. O tempo de processamento é dado por um valor constante p, independente da tarefa. Cada tarefa J_i, com $i = 1, ..., n$, possui um tempo de início de seu processamento Q_i e uma data de entrega d_i. O objetivo do problema é definir um sequenciamento viável para a máquina M tal que minimize o tempo de conclusão do processamento de todos lotes. Para tal, considera-se o melhor agrupamento de tarefas para cada lote. Os autores apresentam como exemplos de aplicação desta especificidade de sequenciamento como: simples entrega realizadas por caminhões e a indústria de semi-condutores.

Ikura & Gimple [1986] tratam o problema de duas formas: assumindo que não haja um prazo de entrega para as tarefas e considerando que as tarefas são iguais, o que corresponde a atribuição de prazos de entrega a medida que representa seas tarefas se encontram prontas para iniciar o processamento de forma que $d_i - Q_i > p$. Para o primeiro caso, o autor desenvolve a Heurística FOE (*First-Only-Empty*), considerando o primeiro lote com sua capacidade parcialmente ocupada e o restante dos lotes totalmente ocupados. Para o segundo caso o autor propõe a Heurística GAB (*Greedy-Adjusted-Bunching*) para solucionar o problema. Esta Heurística trata-se da aplicação de um algoritmo guloso que tenta atribuir o máximo de tarefas a um lote, desde que o prazo de entrega seja cumprido. Desta forma os autores mostram que é possível desenvolver algoritmos eficientes para o problema de Sequenciamento de Tarefas em Lotes.

Damodaran & Chang [2008] definem o Problema de Sequenciamento de Tarefas em Máquinas Idênticas e Paralelas de Processamento em Lotes, como o se-

quenciamento de um conjunto de tarefas $i \in J$ em um conjunto de máquinas paralelas idênticas $m \in M$. As tarefas são processadas em lotes por qualquer máquina $m \in M$, considerando-se que as mesmas devam ser agrupadas de forma que a capacidade das máquinas (K) não seja ultrapassada. Os autores consideraram cada tarefa apresentando tempo de processamento p_i e tamanho s_i diferentes entre si. Considerando os tempos de processamentos diferentes, propôs-se a aceitação de que o tempo de processamento de cada lote (w_k) fosse representado pelo maior tempo p_i de processamento das tarefas que o compõe. Os autores consideram que, no pior caso, o número de lotes será igual ao número de tarefas.

O objetivo do problema proposto por Damodaran & Chang [2008] seria a minimização do *makespan* (U_{max}). Quanto à formação do lote os autores expõem a possibilidade de se considerar como critério de capacidade da máquina o número de tarefas. Damodaran & Chang [2008] aplicaram ao problema as regras de *bin-packing* combinadas com as regras de sequenciamento em programação com mais de uma máquina envolvida.

Kashan et al. [2008] propuseram para o mesmo problema um algoritmo genético hibrido. Jia & Leung [2015] desenvolveram o algoritmo *max–min ant system* e mostraram que o mesmo apresenta melhores resultados do que os algoritmos propostos por Kashan et al. [2008].

Fidelis & Arroyo [2016] definem o mesmo problema tratado por Damodaran & Chang [2008], considerando como um diferencial a classificação das tarefas em famílias, da seguinte maneira: Dado um conjunto de n tarefas a serem processadas em lotes em um conjunto de m máquinas dispostas em um arranjo de processamento paralelo, deseja-se minimizar o atraso ponderado total. Segundo os mesmos autores cada tarefa i apresenta tamanho unitário, e pertence a uma família de tarefas f (todas as tarefas pertencentes a uma mesma família f apresentam tempo de processamento idênticos). Toda tarefa i possui um tempo de chegada Q_i de forma que o processamento de cada tarefa i somente poderá ser iniciado após a chegada da tarefa. Cada tarefa i possui um peso ou prioridade r_i e uma data de entrega d_i. As m máquinas são de processamento em lote, de forma que as tarefas possam ser agrupadas e processadas juntas em uma mesma máquina em um mesmo período de tempo, no entanto, somente tarefas pertencentes a uma mesma família podem ser processadas juntas. Deve-se ater ao fato que um tarefa i só poderá

estar associada a um único lote k. O tamanho máximo de cada lote B_k é limitado de forma que cada máquina m pode processar no máximo K tarefas ao mesmo tempo. Um lote B_k com tarefas da família f estará disponível para ser processado no tempo $Q_k = \max\{Q_i | i \in B_k\}$. O tempo de processamento de um lote é igual ao tempo de processamento das tarefas pertencentes ao lote. Quando uma máquina começa a processar um lote B_k de tarefas, nenhuma interrupção é permitida, assim, não é possível adicionar ou retirar tarefas no lote até que o processamento do lote finalize. O atraso da tarefa i é definido como $A_i = \max(0, u_i - d_i)$, onde u_i é o tempo de conclusão da tarefa i. O atraso ponderado total é determinado por $TWT = \sum_i r_i \times A_i$.

Fidelis & Arroyo [2016] utilizaram a heurística Iterated Greedy para tratar o problema. Utilizou-se como solução inicial os resultados obtidos pelo algoritmo construtivo TWD (Time Window Decomposition) proposto por Biyk et al. [2014], melhorado por uma busca local. No processo de busca local consideraram 3 diferentes estruturas de vizinhanças: *JobInsert* onde uma tarefa é removida de um lote de uma certa família e movido para outro lote da mesma família que ainda não esteja cheio ou cria-se um novo lote no fim do sequenciamento da máquina; *JobSwap* onde duas tarefas da mesma família em diferentes lotes são trocadas; e *BatchSwap* onde dois lotes da mesma família em diferentes máquinas são trocados.

Fidelis & Arroyo [2017] propuseram o uso das heurísticas Adaptive Large Neighborhood Search ($ALNS$) e Simulated Annealing (SA) para o problema proposto por Fidelis & Arroyo [2016]. De acordo com Fidelis & Arroyo [2017] a heurística $ALNS$ consiste em uma busca local onde em cada iteração um algoritmo é escolhido para destruir parcialmente a solução corrente e outro é escolhido para reparar a solução destruida. Os algoritmos são analisados estatisticamente compondo seus históricos de desempenho. Os históricos de desempenho de cada algoritmo construtivo e destrutivo influenciaram durante a escolha de quais algoritmos serão utilizados em cada iteração do ALNS.

A heurística SA proposta por Fidelis & Arroyo [2017] foi desenvolvida tendo como base a SA proposta por Damodaran & Velez-Gallego [2014]. Segundo Fidelis & Arroyo [2017] o principio do SA proposto por Damodaran & Velez-Gallego [2014] seria inicializar o problema com uma solução inicial com a construção de número reduzido de lotes, e pesquisar por soluções melhores sem permitir que o algoritmo

acrescente lotes desnecessários.

2.2 Problema de Sequenciamento de Tarefas em Máquinas Paralelas de Processamento em Lotes ($STMPL$) com Entrega

Leung & Chen [2013] tratam o problema de Sequenciamento de Tarefas com Entrega considerando uma única linha de produção que processa as tarefas, sem agrupá-las. Na sequência as tarefas são agrupadas e entregues aos clientes por um número fixo de veículos. Os autores definem o problema da seguinte maneira: Um fabricante recebe um conjunto de n ordens independentes, J=$J_1...J_n$. Cada ordem J_i possuindo um tempo de processamento p_i e um prazo de entrega d_i diferentes entre si. Finalizada a produção da ordem, a mesma deve ser entregue ao cliente. Para tal, tem-se um conjunto de veículos disponíveis para realizar a entrega. Os veículos apresentam tempos de saída pré determinados. Desta maneira, considera-se a existência de z possibilidades de instantes $T_1, ...T_z$ onde se tem v_t veículos disponíveis para a entrega em cada instante T_t. Os pedidos são embalados em paletes de tamanho padrão por conveniência da entrega, de forma que cada veículo poderá entregar C pedidos. Apenas os pedidos completados no tempo T_t poderão ser entregues pelos veículos v_t. Caso não seja possível a entrega de uma ordem J_i no tempo T_t a mesma deverá ser entregue em outro período T_t maior que o primeiro. Quando tal evento ocorrer deve-se levar em conta o atraso (A_i) dado por: $A_i = max(0, T_t - d_i)$;

Leung & Chen [2013] propõem algoritmos para resolver os problemas considerando os seguintes objetivos: minimizar o máximo atraso das ordens; minimizar o número de veículos utilizados sujeito a condição proposta no objetivo anterior; minimizar a soma ponderada do atraso máximo e o número de veículos.

Cheng et al. [2015] tratam o problema proposto por Leung & Chen [2013] considerando-se que a máquina é de processamento em Lotes. Os autores definiram que cada tarefa i possui dimensão s_i e tempo de processamento p_i. Para o processamento das tarefas tem-se disponível uma única máquina que tem a capacidade de processar mais de uma tarefa ao mesmo tempo, desde que a soma

das dimensões das tarefas não ultrapasse sua capacidade K (caracterizando uma produção em lotes).

Outra característica importante definida por Cheng et al. [2015] é que um lote B_k de tarefas não pode ser finalizado enquanto todas as tarefas designadas a ele não tiverem sido completamente processadas. Sendo assim, o tempo de processamento de um lote w_k é dado pela tarefa pertencente ao lote B_k que apresenta o maior tempo de processamento. Os autores definem como o tempo de termino do processamento do lote B_k por o_k. O primeiro lote B_0 apresenta o_0 e w_0 com valores iguais a zero. Considera-se a possibilidade da existência de lotes vazios que apresentaram neste caso valores de w_k igual a zero. Os autores definem uma variável LV para contabilizar o número de lotes vazios.

Quanto a entrega, Cheng et al. [2015] consideram que uma tarefa quando completada deve ser entregue ao cliente, para tal, tem-se disponíveis veículos com capacidade C. Sendo assim, um veículo poderá entregar as tarefas cujo o somatório de seus respectivos tamanhos seja menor do que C. Os instantes de entrega são denotados pelo conjunto $T = T_1, T_2, ..., T_z$, de forma que possam haver instantes de entrega vazios. O número de instantes vazios é denotado por IV.

A função objetivo definida por Cheng et al. [2015], consiste na minimização dos custos totais de produção e entrega. Os mesmos são dados por $PC = \sum_{k=1}^{n} w_k$ e por $DC = \delta I$ onde $\delta > 0$.

A seguir apresenta-se o Modelo matemático proposto por Cheng et al. [2015]. As variáveis deste modelo são definidas da seguinte forma:

- $I_k = 1$ se B_k é não vazio e 0 caso contrário;
- $y_t = 1$ se o intante de entrega T_t é não vazio e 0 caso contrário;
- $X_{ik} = 1$ se a tarefa i esta alocado no lote B_k e 0 caso contrário;
- $Z_{it} = 1$ se a tarefa i estiver alocada no instante T_t e 0 caso contrário;

O modelo é definido da seguinte maneira:

$$\min = TC = PC + DC \tag{2.1}$$

$$w_k = \max\{p_i | X_{ik} = 1\} \qquad \forall k = 1..n \tag{2.2}$$

$$o_k = w_{k-1} + o_k \qquad \forall k = 1..n \tag{2.3}$$

$$\sum_{k=1}^{n} X_{ik} = 1 \qquad \forall i = 1..n \tag{2.4}$$

$$\sum_{i=1}^{n} S_i X_{ik} \leq K \qquad \forall k = 1..n \tag{2.5}$$

$$X_{ik} \leq I_k \qquad \forall i = 1..n, k = 1..n \tag{2.6}$$

$$\sum_{t=1}^{n} T_{it} = 1 \qquad \forall i = 1..n \tag{2.7}$$

$$\sum_{t=1}^{n} T_{it} S_i \leq C \qquad \forall i = 1..n \tag{2.8}$$

$$T_{it} \leq y_t \qquad \forall i = 1..n, t = 1..n \tag{2.9}$$

$$LV = \sum_{t=1}^{n} y_t \tag{2.10}$$

$$IV = \sum_{k=1}^{n} I_k \tag{2.11}$$

$$PC = \sum_{k=1}^{n} o_k \tag{2.12}$$

$$DC = \delta LV \qquad \delta > 0 \tag{2.13}$$

$$C_0 = o_0 = I_0 = X_{j0} = 0 \tag{2.14}$$

$$I_k, y_l, X_{jk}, T_{jl} \in \{0, 1\} \qquad \forall i, k, t \tag{2.15}$$

A função objetivo 2.1 representa o custo total de produção representado por 2.12 e de distribuição por 2.13. A restrição 2.2 calcula o tempo de conclusão de cada lote. A restrição 2.3 calcula o tempo de conclusão do lote, considerando os lotes anteriores. O fato de uma tarefa poder pertencer apenas a um lote de processamento é representada pela restrição 2.4. Quanto as alocações das tarefas em lotes que deve respeitar a capacidade das máquinas é restringida em 2.5. A restrição 2.6 representa a necessidade de existir o lote k para que uma tarefa possa ser associada ao mesmo. A restrição 2.7 faz com que uma tarefa só possa ser associada a um intervalo de entrega. Quanto as alocações das tarefas nos momentos de entrega, deve respeitar a capacidade das máquinas, essa restrição está

representada em 2.8. A restrição 2.9 determina que uma tarefa para ser associada a um intervalo de entrega, o mesmo momento deve existir, mas o contrário não é válido. A restrição 2.10 calcula o número de momentos de entrega não vazios e a restrição 2.11 o número de lotes não vazios. A restrição 2.12 e 2.13 calculam respectivamente os custos de produção e entrega. A restrição 2.15 determina que as variáveis são binarias.

Cheng et al. [2015] definem o problema como NP-Hard, e frisam a necessidade de se obter um *Lower Bound* (LB), com o intuito de testar a performance das heurísticas a serem implementadas. Para tal os autores propõem a relaxação do problema da seguinte forma:

Os mesmos autores propõem primeiramente, considerando que cada tarefa i apresenta tamanho s_i, dividir cada tarefa i em s_i tarefas de tamanho unitário e com tempo de processamento igual ao tempo de processamento da tarefa i que lhe deu origem. A partir dessa nova lista de tarefas gerada, Cheng et al. [2015] definem o seguinte construtivo para a geração do LB: Ordenam-se as tarefas em ordem decrescente de tempo de processamento. Em seguida, organizam-se as tarefas em grupos de tamanho K que irão compor cada grupo um lote. Calcula-se então o valor de w'_k usando a fórmula 2.2. Quanto à distribuição utiliza-se a mesma ideia, obtendo-se o valor de LV através da equação $LV = \lceil \sum_{i=1}^{n} s_i K \rceil$. Por fim calcula-se o valor do LB dado por $\sum_{k=1}^{K} w'_k + \delta LV$.

Cheng et al. [2015] também propuseram para o problema a utilização da Heurística Colonia de Formigas, que se trata de uma algoritmo inteligente baseado no comportamento das formigas em busca de comida.

Li et al. [2015] definem uma vertente mais complexa dos problemas tratados por Cheng et al. [2015]. No trabalho de Li et al. [2015] é definido o problema de Sequenciamento de Tarefas em Máquinas Paralelas de Processamento em Lotes (STMPL) com entrega, considerando a construção de lotes tanto no processo de sequenciamento de tarefas nas máquinas, quanto nos veículos. Os autores representam o problema da seguinte maneira: Considera-se o problema de sequenciamento para n tarefas a serem realizadas em m máquinas idênticas e paralelas; as tarefas são trabalhadas em lotes (agrupada uma certa quantidade de tarefas, executadas em uma mesma máquina em um mesmo período de tempo) e apresentam o mesmo tempo de processamento p; cada tarefa i possui um tamanho s_i, uma data de

entrega d_i e um peso r_i; as máquinas apresentam capacidades iguais a K, tal que o somatório dos tamanhos das tarefas alocadas em um lote sequenciado em uma dada máquina não ultrapasse a sua capacidade K, levando em consideração que uma mesma tarefa não poderá ser executada por mais de um lote. Quanto ao processo de entrega, segundo Li et al. [2015], considera-se que são providos a empresa v_t veículos para realizar a entrega em cada instante $T_t(1 \leq t \leq z)$. Os veículos possuem capacidades iguais a C, sendo assim, poderá ser destinado a cada veículo um número máximo de tarefas, tal que o somatório dos tamanhos das tarefas não ultrapasse a capacidade C do veículo. lembrando-se que uma mesma tarefa não poderá ser entregue por mais de um veículo. O peso r_i relacionado a cada tarefa i só será convertido em ganho, se e somente se, as datas de entrega forem cumpridas. O objetivo do problema seria definir o sequenciamento das tarefas nas máquinas e nos veículos que maximize o ganho da empresa.

Li et al. [2015] consideram diversas possibilidades de estudo para o problema. No primeiro caso, eles consideram que as tarefas possuem tamanhos iguais a 1 unidade. Para esta vertente do problema, os autores observaram que o mesmo poderia ser resolvido em tempo polinomial. Para este caso, os autores desenvolveram um algoritmo que denominaram de Sch-Id-Size que apresentou uma complexidade dada por $O(n(z^2 + logn))$. Esse algoritmo trata-se de um cosntrutivo que possui como entrada uma lista de tarefas L ordenada de acordo com o critério peso dado por r.

Li et al. [2015] consideram como outra possibilidade de problema, quando as tarefas apresentam tamanhos diferentes entre si. Essa vertente de problema foi considerado pelos mesmos autores como NP-difícil. Nesta vertente do problema, Li et al. [2015] consideram dois casos diferentes para resolução, descritos a seguir.

No primeiro caso, assume-se que cada tarefa, independentemente do seu tamanho s_i, seja agrupada em paletes de tamanho padrão e então entregue por um veículo com capacidade de transportar C tarefas. Ou seja, considerando cada tarefa possuindo tamanho igual a uma unidade para alocação nos veículos. Enquanto que, no sequenciamento de tarefas nas máquinas, o tamanho s_i de cada tarefa i deve ser considerado na costrução de cada lote.

No segundo caso, considera-se que cada tarefa ocupará um montante da capacidade do veículo igual ao seu tamanho s_i. Nesse sentido, um veículo poderá

entregar menos de C tarefas e ter sua carga máxima alcançada, o mesmo ocorrendo no sequenciamento de tarefas nas máquinas, considerando a capacidade de cada máquina igual a K.

Baseado no algoritmo Sch-Id-Size os mesmos autores desenvolveram heurísticas para as vertentes do problema que consideram as tarefas com tamanhos diferentes de 1. Para o primeiro caso, os autores propõem uma heurística construtiva denominada Heu1. Nesta heurística assume-se que cada tarefa i, independentemente do seu tamanho s_i, seja agrupada em paletes de tamanho padrão, por conveniência da entrega. Então cada veículo entregará C tarefas. Neste caso a diferença é que não necessariamente se processará K tarefas em cada lote em cada determinada máquina, sabido que os tamanhos das tarefas são diferentes. Sendo assim, as tarefas serão ordenadas de forma decrescente, não mais somente pelos pesos r_i, mas pela razão entre o peso e o tamanho da tarefa ($\frac{r_i}{s_i}$). Esta heurística apresentou uma complexidade dada por $O(nz(z + logn))$.

Para o segundo caso, Li et al. [2015] propuseram outra heurística construtiva denominada Heu2. Neste caso considera-se que cada tarefa i ocupará um montante da capacidade do veículo igual ao seu tamanho s_i. Nesse sentido um veículo poderá entregar menos de C tarefas e ter sua carga máxima alcançada. Ou seja, as tarefas terão de ser agrupadas por duas vezes, uma por intervalo de tempo de processamento por máquina e outra por veículo por instante de entrega. Esta heurística apresenta como entrada uma lista L ordenada de maneira decrecente de ($\frac{r_i}{s_i}$) e apresentou uma complexidade dada por $O(nz(z + logn + logV))$.

Li et al. [2015] utilizam o cálculo do GAP para medir a qualidade dos resultados obtidos pelas Heurísticas Heu1 e Heu2, sendo $GAP = (\frac{UB}{Heu} - 1) * 100$, onde UB é o *Upper Bound*, ou seja, a melhor solução encontrada para o problema, e Heu o total obtido com a heurística desenvolvida.

Segundo Li et al. [2015], o *Upper Bound* para a Heu1 é obtido através do relaxamento da restrição de inteiro da própria heurística Heu1. Ou seja permitindo que uma tarefa inicie seu processo em uma primeira máquina e termine em uma segunda máquina. No caso da heurística Heu2, os autores propõem, além da relaxação já definida para Heu1, a possibilidade de uma mesma tarefa estar associada a mais de um veículo (sendo uma percentagem de seu tamanho alocada a um veículo v_t e o restante a outro veículo $v_{t'}$, tal que, $t \neq t'$).

Capítulo 3

Formulação do Problema e Métodos de Resolução Propostos

Este capítulo apresenta 4 seções. A seção 3.1 descreve formalmente o problema STMPL com entrega. A seção 3.2 define um modelo de programação linear inteira mista e a estrutura de solução para o problema estudado. A seção 3.4 apresenta a heurística para a obtenção valores de *Upper Bound* (UB) utilizados como critério de medição de qualidade das heurísticas implementadas. Por fim a seção 3.5 apresenta as heurísticas implementadas para o tratamento do problema STMPL com entrega.

3.1 Definição Formal do Problema

O problema a ser estudado neste trabalho é o Problema de Sequenciamento de Tarefas em Máquinas Paralelas de Processamento em Lotes ($STMPL$) com Entrega, considerando-se a construção de lotes tanto no processo de sequenciamento de tarefas nas máquinas, quanto nos veículos que realizarão as entregas em um dado momento. O problema pode ser definido como se segue.

Quanto ao processamento das tarefas, considera-se o problema de sequenciamento para n tarefas a serem realizadas em m máquinas idênticas e paralelas. Cada tarefa j possuindo um tamanho s_j, uma data de entrega d_j e um tempo de processamento p_j. As tarefas são sequenciadas em lotes respeitando-se a capacidade K de cada máquina l. Sendo assim, um número máximo de tarefas podem

ser agrupadas para formar um lote k, tal que o somatório dos tamanhos das mesmas não ultrapasse a capacidade K de cada máquina l em um dado momento. As tarefas agrupadas para composição de um lote k em uma máquina l são executadas em um mesmo período de tempo, dado pela tarefa j agrupada com maior tempo de processamento. Deve-se ater ao fato de que uma mesma tarefa não pode ser executada por mais de um lote em uma máquina.

Quanto à entrega das tarefas, as mesmas são recolhidas para entrega nos tempos $T_1 < T_2 < ... < T_z$, considerando-se que são providos a empresa v_t veículos para realizar as entregas em cada momento de entrega $T_t (1 \leq t \leq z)$. Os veículos possuem capacidade fixa igual a C. Sendo assim, poderá ser destinado a cada veículo um número máximo de tarefas, tal que o somatório dos tamanhos das mesmas não ultrapasse C. Uma tarefa i para ser associada a um momento de entrega T_t, seu momento de conclução u_i deve ser menor ou igual a T_t.Deve-se ater ao fato que uma mesma tarefa não poderá ser entregue por mais de um veículo.

Por fim, uma tarefa i será sequenciada, se e somente se, as datas de entrega d_i forem cumpridas. Quando uma tarefa j for sequenciada o seu ganho r_i será acrescido ao ganho total da empresa. O objetivo do problema é definir o sequenciamento das tarefas nas máquinas e nos veículos que maximize o ganho da empresa.

3.2 Formulação do Problema de Programação Linear Inteira Mista

Neste trabalho inicialmente é proposto um modelo de Programação Linear Inteira Mista (PLIM) para resolver o problema estudado. O modelo é uma adaptação do modelo proposto por Cheng et al. [2015] e apresentado na Seção 2.2 levando em consideração as características do problema STMPL com entrega que é baseada no trabalho de Li et al. [2015].

Dados do problema:

- s_i: tamanho da tarefa i
- K: capacidade da máquina das máquinas

- C: capacidade dos veículos de entrega
- T_t: momento de entrega t
- d_i: prazo de entrega da tarefa i
- p_i: tempo de processamento da tarefa i
- r_i: Ganho da tarefa i

Variáveis de decisão do modelo:

- x_{ik}^l: 1 se a tarefa i pertence ao lote k na máquina l, 0 caso contrário
- y_{it}^v: 1 se a tarefa i será associado ao veículo v pertencente ao intervalo t, 0 caso contrário
- w_k^l: tempo de processamento do lote k na máquina l
- o_k^l: momento em que o lote k termina de ser processado na maquina l
- u_i: momento em que a tarefa i termina de ser processada

Modelo matemático:

$$\max f = \sum_{i=1}^{n} \sum_{t=1}^{z} \sum_{v=1}^{V_t} r_i y_{it}^v \tag{3.1}$$

$$\sum_{k=1}^{n} \sum_{l=1}^{m} x_{ik}^l \leq 1 \qquad \forall i = 1..n \tag{3.2}$$

$$\sum_{t=1}^{z} \sum_{v=1}^{V_t} y_{it}^v \leq 1 \qquad \forall i = 1..n \tag{3.3}$$

$$\sum_{i=1}^{n} s_i x_{ik}^l \leq K \qquad \forall k = 1..n, l = 1..m \tag{3.4}$$

$$\sum_{i=1}^{n} s_i y_{it}^v \leq C \qquad \forall t = 1..z, v = 1..V_t \tag{3.5}$$

$$w_k^l \geq p_i x_{ik}^l \qquad \forall i = 1..n, k = 1..n, l = 1..m \tag{3.6}$$

$$o_1^l = w_1^l \qquad \forall l = 1..m \tag{3.7}$$

$$o_k^l = o_{k-1}^l + w_k^l \qquad \forall k = 2..n, l = 1..m \tag{3.8}$$

$$u_i \geq o_k^l - M(1 - x_{ik}^l) \qquad \forall i = 1..n, k = 1..n, l = 1..m \tag{3.9}$$

$$\sum_{t=1}^{z} \sum_{v=1}^{V_t} y_{it}^v T_t \leq d_i \qquad \forall i = 1..n \tag{3.10}$$

$$\sum_{k=1}^{n} \sum_{l=1}^{m} x_{ik}^l \geq \sum_{t=1}^{z} \sum_{v=1}^{V_t} y_{it}^v \qquad \forall i = 1..n \tag{3.11}$$

$$u_i \leq \sum_{t=1}^{z} \sum_{v=1}^{V_t} y_{it}^v T_t \qquad \forall i = 1..n \tag{3.12}$$

$$x_{ik}^l \in \{0, 1\} \qquad \forall i = 1..n, k = 1..n, l = 1..m \tag{3.13}$$

$$y_{it}^v \in \{0, 1\} \qquad \forall i = 1..n, t = 1..z, v = 1..V_t \tag{3.14}$$

$$u_i \geq 0 \qquad \forall i = 1..n \tag{3.15}$$

$$o_k^l \geq 0 \qquad \forall k = 1..n, l = 1..m \tag{3.16}$$

$$w_k^l \geq 0 \qquad \forall k = 1..n, l = 1..m \tag{3.17}$$

A função objetivo, expressa pela equação (3.1), trata-se da maximização do ganho obtido pelas tarefas que foram sequenciadas. A equação (3.2) assegura que cada tarefa é associada no máximo a um lote em uma única máquina. A restrição (3.3) garante que cada tarefa só pode estar associada no máximo a um momento de entrega. As restrições (3.4) e (3.5) impedem, respectivamente, que a capacidade das máquinas e a capacidade de entrega sejam extrapoladas. O tempo necessário para produzir um lote completo é igual ao tempo de maior processamento das tarefas que o compõe, restrição que é definida em (3.6). O momento de conclusão de cada lote é calculado pelas equações (3.7) e (3.8). A restrição (3.9) determina o tempo de conclusão de cada tarefa. A restrição (3.10) determina que cada tarefa deve ser entregue em um momento de entrega menor ou igual ao seu prazo de entrega. A restrição (3.11) define que, para uma tarefa ser alocada para uma posição em uma máquina ela deverá estar associada a um momento de entrega. A restrição (3.12) garante que o período em que cada tarefa é concluída deve ser menor ou igual ao momento de entrega ao qual estará associada. E, finalmente, as restrições (3.13) até a (3.17) definem o escopo e domínio das variáveis de decisão.

O valor de M é dado pelo somatório do tempo de processamento de todas as tarefas da instância.

A relaxação do modelo proposta trata-se da relaxação das variáveis inteiras do problema. Ou seja, uma tarefa poderá ser sequenciada parcialmente em uma máquina e um veículo. Considerando esse fato, poderá ocorrer de uma tarefa poder estar associada a mais de uma máquina e um veículo desde que suas dimensões sejam respeitadas.

3.3 Estrutura de Solução

A representação da solução para os algoritmos aplicados neste trabalho será dada pela seguinte estrutura:

- R: Somatório dos ganhos das tarefas sequenciadas;
- Lista de tarefas sequenciadas em cada lote por máquina (lote_máquina);
- Lista de tarefas sequenciadas em cada veículo por momento de entrega (veículo_momento_de_entrega);
- Lista de tarefas não sequenciadas.

A Figura 3.1 mostra como são as estruturas de armazenamento das listas de tarefas sequenciadas tanto nas máquinas como por intervalo de entrega. No caso do sequenciamento da tarefa i em uma máquina l deve-se ater ao seu tempo de processamento p_i para definir o tempo de processamento do lote k máquina l no qual será sequenciada, dado que o tempo de processamento de um lote é dado pela tarefa, sequenciada no lote, que apresente maior tempo de processamento. Quanto ao parâmetro s_i deve ser observado no sentido de determinar se uma tarefa cabe, ou não, em um lote_máquina e em um veículo_momento_de_entrega. O tempo de processamento do lote será essencial para o cálculo do tempo de conclusão de uma tarefa, que juntamente com o momento de entrega, permitirá definir se a tarefa poderá ser entregue no prazo d_i gerando acréscimo de ganho r_i em R.

Todas as heurísticas têm como entrada uma lista de tarefas L, previamente construída de acordo com critérios pré estabelecidos, essas listas serão utilizadas

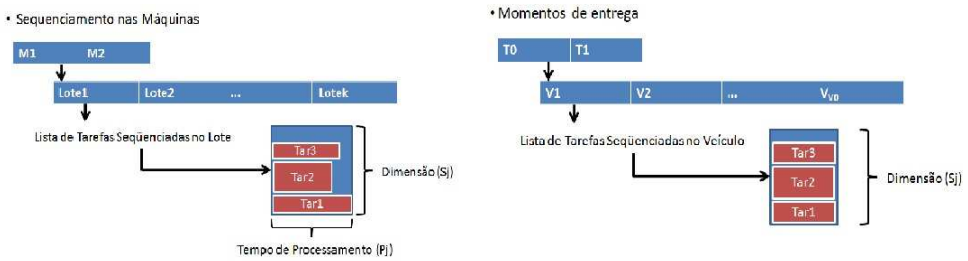

Figura 3.1: Estrutura de Solução

para a construção de uma solução, através de uma heurística construtiva. As heurísticas, exceto a utilizada para a geração do UB, apresentarão como entrada, além da lista L uma solução que será utilizada para definir o quanto a solução está sendo melhorada em cada iteração das heurísticas.

3.4 Um *Upper Bound* (UB) para as Soluções do Problema $STMPL$ com Entrega

A diferença entre o problema tratado por Li et al. [2015] e o proposto neste trabalho é o tempo de processamento das tarefas nas máquinas. No primeiro caso, o tempo de processamento é o mesmo, independente da tarefa e da máquina. No segundo, o tempo de processamento é diferente para cada tarefa, independente da máquina.

Então se propõe, como ideia inicial para a construção de um UB, uma relaxação do problema estudado neste trabalho, fazendo o tempo de processamento das tarefas iguais a 1 unidade de tempo. Com o tempo de processamento das tarefas sempre igual 1, pode-se tratar o problema estudado, utilizando-se a heurística heu2 relaxada proposta Li et al. [2015].

Sendo assim, propõe-se que o tempo de processamento de cada tarefa i de uma lista de candidatos L seja definido como uma unidade de tempo independente da tarefa e da máquina. Quando uma tarefa i possuir um tempo de processamento p_i maior do que 1 unidade de tempo (considerando o tempo de processamento pertencente ao conjunto dos números inteiros), a tarefa será replicada p_i vezes, e cada réplica apresentará tempo de processamento de 1 unidade de tempo. Mas quando este procedimento for realizado deve-se dividir o ganho r_i e o tamanho s_i

da tarefa, de forma que cada replica r_e das p_i tarefas originadas de i apresente o valor de $r_{r_e} = R_i \div p_i$ e de $s_{r_e} = s_i \div p_i$.

Feito isso, considera-se uma nova lista de candidatos L' com as réplicas das tarefas de L. O próximo passo é implementar a heurística heu2 proposta por Li et al. [2015] realizando as relaxações que os mesmos propõem para a resolução do problema, considerando tempos de processamento iguais a p.

Após obter a solução, observa-se que uma mesma tarefa pode ser sequenciada em mais de um lote e uma máquina, além de ser alocada em mais de um veículo e mais de um momento de entrega. Sendo assim, tem se uma restrição do modelo violada. Como neste trabalho pretende-se usar a solução do Upper Bound como solução inicial para as próximas heurísticas, é necessário viabilizar a solução.

Para solucionar o problema da quebra da restrição de não repetibilidade de tarefas durante as alocações propõe-se a construção de uma lista de candidatos L' formada pelas tarefas sequenciadas na solução do UB, eliminando-se as suas réplicas. A cada tarefa de L' será associada uma variável denominada $dataEntrega$ que receberá o valor correspondente ao maior momento de entrega em que ela ou uma de suas réplicas tenham sido sequenciadas. Então, ordenam-se as tarefas de forma decrescente de $dataEntrega$. Após esta construção, as tarefas da lista de candidatos L não sequenciadas pela heurística são incluídas na lista L'.

3.5 Algoritmos Heurísticos Prospostos

No presente trabalho, apresentam-se 5 métodos heurísticos para a obtenção de uma solução para o problema de sequenciamento de tarefas em máquinas paralelas de processamento em lotes com entrega: Com NEH (CN), Sem NEH (SN), *First Improvement* Com NEH ($FICN$), First Improvement Sem NEH ($FISN$) (que apresentam o VND como heurística de melhoria) e o *Simulated Annealing* (SA). Em todos os casos, as vizinhaças foram baseadas em alterações em uma lista de tarefas e postriormente a aplicação da heurística Heu3 para construir as soluções.

A Heurística Nawaz Enscore Ham (NEH), considerando um problema de sequenciamento, é um procedimento de inserção onde cada tarefa, pertencente a uma lista ordenada de tarefas a serem sequenciadas, é testada em todas as

posições disponíveis e sequenciada naquela que apresenta o melhor resultado para o problema.

A heurística VND é um procedimento, no qual, dada uma solução inicial, faz-se uma busca por soluções melhores em um espaço de soluções. Para, tal, faz-se uso de um conjunto de estruturas de vizinhança que são eleitas de forma sistemática a cada iteração, dado um critério de parada.

A Heurística SA é um procedimento em que, dada uma temperatura inicial elevada, faz-se um resfriamento da mesma até um valor próximo de 0, lembrando que o valor da temperatura influenciará o critério de aceitação das soluções. A cada valor de temperatura, faz-se uma busca aleatória dentro de uma estrutura de vizinhança, respeitando-se critérios de parada pré estabelecido.

A heurística Heu3 é um construtivo guloso desenvolvido, neste trabalho, baseado nas heurísticas propostas por Li et al. [2015]. Nesta heurística, dada uma lista de tarefas ordenadas de acordo com um critério pré estabelecido, faz-se o sequenciamento das tarefas de maneira sequencial nas máquinas de processamento em lotes e nos momentos de entrega, respeitando-se as limitações de processo.

A heurística Com NEH (CN) recebe esse nome devido ao fato de utilizar-se da heurística NEH para a obtenção da solução inicial. Na heurística CN, dada uma lista de candidatos L, faz-se:

- Cálculo do UB, gerando uma solução S do tipo definido em 3.3 e uma lista de tarefas L' definida em 3.4;
- Tendo como dado de entrada a lista L', executa-se a Heurística NEH (Heurística Construtiva HC);
- Constrói-se uma lista L" que concatena as tarefas sequenciadas e não sequenciadas na solução obtida no passo anterior (Preparação para a heurística de Melhoria - HM);
- Tendo como dado de entrada L' e a solução fornecida pelo passo anterior, executa-se o VND tendo como estratégia o *best improvement* (Heurística de Melhoria - HM).

A heurística Sem NEH (SN) recebe esse nome devido ao fato de não utilizar-se da heurística NEH para a obtenção da solução inicial, mas sim a heurística construtiva Heu3. A heurística SN, dada uma lista de candidatos L, faz-se:

- Cálculo do UB, gerando uma solução S do tipo definido em 3.3 e uma lista de tarefas L' definida em 3.4;
- Tendo L' como entrada executa-se a heurística Heu3 (Heurística de Construtiva - HC);
- Tendo como dado de entrada L' e a solução fornecida pelo passo anterior, executa-se o VND tendo como estratégia o *best improvement*.(Heurística de Melhoria - HM)

A heurística $FICN$ é semelhante à CN, assim como o $FISN$ é semelhante à SN. A diferença está apenas na estratégia busca nas vizinhaças utilizada: no caso do CN e do SN utilizou-se a estratégia *best improvement*, já no caso do $FICN$ e do $FISN$ utilizou-se a estratégia *first improvement*, por isso os seus nomes *first improvement* com NEH ($FICN$) e *first improvement* Sem NEH ($FISN$). Nas próximas seções serão explicadas detalhadamente todas os heurísticas que compuseram os quatro métodos CN, SN, $FISN$ e $FICN$.

Na Heurística SA implementada, dada uma lista de tarefas candidatas L, faz-se:

- Cálculo do UB, gerando uma solução S do tipo definido em 3.3 e uma lista de tarefas L' definida em 3.4;
- Tendo L' como entrada executa-se o Heu3 (Heurística de Construtiva - HC);
- Tendo como dado de entrada L' e a solução fornecida pelo passo anterior, executa-se a heurística SA.

3.5.1 Heurística NEH - Nawaz Enscore Ham

No presente trabalho, para o problema de sequenciamento de tarefas em máquinas paralelas de processamento em lote com entregas propôs-se a utilização da heurística NEH para geração de uma solução inicial.

A heurística NEH foi proposta por Nawaz et al. [1982], para a resolução de um problema de sequenciamento de n tarefas em m máquinas de processamento em série. Os autores se propuseram a encontrar uma sequência que gerasse um valor ótimo ou pelo menos próximo do ótimo para o problema. Para tal, apresentou técnicas exatas e aproximadas.

A técnica aproximada proposta por Nawaz et al. [1982] trata-se da Heurística NEH, palavra originada das primeiras letras dos sobre nomes dos altores: Nawaz, Enscore e Ham.

Os autores definiram o problema da seguinte forma: Dadas n tarefas, tal que, cada tarefa i apresente tempo de processamento t_{il} na máquina l. Dado a existência de m máquinas dispostas em série para o processamento das tarefas; o objetivo é encontrar o sequenciamento que minimiza o maior tempo de processamento de tarefas, ou seja, o *makespan*. Então a ideia central da técnica está baseada em aceitar que as tarefas com maior tempo de processamento em todas as máquinas tenham maior prioridade de sequenciamento. Na sequência testa-se a melhor máquina e melhor posição para as tarefas seguindo a ordem de prioridade.

Esta Heurística proposta é definida pela seguinte lógica, dado o problema estudado pelos autores:

1. Para cada tarefa i calcula-se: $T_i = \sum_{l=1}^{m} t_{il}$;

2. Ordena-se as tarefas de forma decrescente de T_i gerando-se uma lista L;

3. Seleciona-se as duas primeiras tarefas de L e encontra-se a melhor sequência para elas através do calculo do *makespan* para as possíveis ordens de sequenciamento. O sequenciamento aceito é aquele que gera o menor *makespan*. Nós próximos passos do algoritmo as posições relativas das tarefas já sequenciadas devem ser mantidas;

4. A partir da tarefa 3 faz-se o seguinte procedimento: Para cada i-ésima tarefa de L identifica-se a melhor sequência dentre todas as possibilidades. Para tal, considera-se as tarefas já sequenciadas e o fato de não poder alterar a sua posição relativa. Esse procedimento é feito até que todas as tarefas tenham sido sequenciadas.

Em 2007 de Paula et al. [2007] propôs a utilização da Heurística NEH para o problema de sequenciamento em máquinas paralelas com sequência dependente. Os autores definiram o problema da seguinte forma: Considerando-se m máquinas e n tarefas com tempo de processamento p_{il} (tempo de processamento da tarefa i na máquina l) e um peso r_i associado ao atraso associado a tarefa i, o objetivo seria minimizar o *makespan* somado aos pesos por atraso. Segundo de Paula et al. [2007] a heurística NEH apresenta muitas vantagens em relação a outras abordagens polinomiais para a obtenção de uma solução inicial para o problema de sequenciamento em máquinas paralelas com sequência dependente.

No presente trabalho, tendo como entrada uma lista L de candidatos, aplica-se a ideia da heurística NEH durante o sequenciamento na máquina, levando-se em consideração que o sequenciamento na máquina só será aceito como solução se as tarefas sequenciadas puderem ser alocadas a um veículo. A heurística NEH implementada neste estudo se comporta da seguinte forma: Cada tarefa pertencente a lista L, em ordem, será testada em todos lotes em todas as máquinas (considerando o número de lotes uma variável dada pelo número de tarefas sequenciadas mais 1); a tarefa é sequenciada no lote e máquina que apresentar o melhor resultado e este sequenciamento será dado como solução inicial para a próxima iteração; as tarefas já sequenciadas não poderão ter suas posições alteradas para a alocação de outras tarefas. Os testes que determinam o melhor resultado estão associados à melhoria do valor de R (ganho por entrega no prazo), caso haja empate de R compara-se o tempo de conclusão da tarefa, escolhendo aquele que apresenta o menor tempo de conclusão. Antes de alocar uma tarefa a um lote em uma máquina deve-se testar se existe algum veículo disponível para realizar a entrega da tarefa no intervalo de tempo entre o seu momento de conclusão e o seu prazo de entrega. Como solução a heurística retorna a lista L e uma solução S definida em 3.3.

3.5.2 Heurística Heu3, baseada nas heurísticas propostas por Li et al. [2015]

Li et al. [2015] propuseram heurísticas construtivas gulosas para solucionar problemas de $STMPL$ com entrega. O grande diferencial do presente trabalho em relação ao apresentado em Li et al. [2015] é o tempo de processamento. Li et al. [2015]

consideraram em seu trabalho os tempos de processamento das tarefas sendo iguais a um valor p, independente da tarefa e da máquina. Já neste trabalho, pretende-se tratar o problema considerando-se o tempo de processamento p_i diferente para cada tarefa i, independente da máquina. Sendo assim, partiu-se do princípio de identificar possíveis pontos de adaptação nas heurísticas propostas por Li et al. [2015], para o estabelecimento de um construtivo a ser aplicado na variação do problema estudado.

Como Li et al. [2015] denominaram as suas heurísticas de Heu1 e Heu2, aqui optou-se por chamar a heurística construtiva de Heu3. A Heurística então pode ser definida da seguinte maneira:

É passado como dado de entrada uma lista de tarefas candidatas L onde cada tarefa i apresenta as seguintes características:

- s_i = tamanho da tarefa i;
- p_i = tempo de processamento da tarefa i;
- d_i = prazo máximo de entrega da tarefa i;
- r_i = ganho se a tarefa i for entregue antes do seu prazo máximo de entrega d_i;

O momento de conclusão da tarefa i, sequenciada, está relacionada ao tempo de conclusão do lote k, no qual esteja sequenciada, na máquina l (o_k^l). o_k^l é calculado da seguinte maneira: primeiramente identifica-se a tarefa i, sequenciada neste lote, com maior tempo de processamento (denominado tempo de processamento do lote w_k^l). Por fim este valor é somado ao somatório do tempo de processamento dos lotes que o antecedem na máquina l, obtendo se o momento de conclusão do lote dado por o_k^l.

Sendo assim, dados z momentos de entrega, em cada momento T_t existem V_t veículos disponíveis para entrega com capacidade de entrega K cada. A capacidade de um veículo é diminuída de acordo com as dimensões das tarefas alocadas no mesmo. O momento T_t de entrega que cada tarefa i estiver associada deve ser menor ou igual ao seu prazo máximo de entrega d_i. Então primeiramente deve-se identificar o momento T_t menor ou igual d_i, (mais próximo possível do valor

de d_i) para cada tarefa i. Este ultimo valor será representado por TD_i. A nova informação TD_i é associada a cada tarefa i pertencente a Lista L.

O próximo passo é criar uma lista de tarefas auxiliar para cada momento de entrega T_t, denominada LT_t, para cada $t = 0$ até $t = z$. Então retiram-se as tarefas da lista L de forma que cada tarefa i retirada seja alocada para a lista LT_t cujo momento de entrega seja igual a TD_i. O procedimento será realizado até que L esteja vazia.

Por fim, a cada LT_t, com t variando de $t = 0$ até $t = z$, corresponderá uma lista de tarefas com data de entrega equivalente a T_t. Então, enquanto LT_t não for vazia fazer:

1. Se a Lista LT_t não for fazia faz-se: verifica-se se já existe um lote em alguma máquina l cujo o valor de w_k^l (valor máximo dos tempos de processamentos das tarefas que compõem o lote k na máquina l) seja maior ou igual ao tempo de processamento p_i da tarefa i a ser incluída. Caso a verificação retorne uma sentença verdadeira ir para o passo 3 senão passo 2. Este teste somente poderá ser realizado em máquinas cujo tempo de conclusão seja menor do que o prazo d_i de entrega da tarefa i, e que ainda possua capacidade ociosa com valor menor que o tamanho s_i da tarefa i.

2. Busca-se a máquina l com menor tempo de conclusão. Se o tempo de conclusão do último lote k da máquina l, até o momento, acrescido do valor p_i da tarefa i a ser inserida for menor que TD_i cria-se um novo lote $k = k + 1$ na máquina l. Inicializa-se o tempo de processamento do novo lote k (w_k^l) com o tempo de processamento p_i, que será considerado como o maior valor de processamento que o lote k aceitará (de forma que as próximas tarefas que poderão ser alocadas ao lote k devam possuir tempo de processamento inferior a w_k^l). Se não for possível criar um lote respeitando-se o momento de conclusão do ultimo lote das máquinas disponíveis ir para o passo 5.

3. Verifica-se a existência de um veículo v em um momento t disponível, ou seja, com capacidade ociosa maior que o tamanho s_i da tarefa i a ser inserida. Para que o teste seja realizado o veículo v deve pertencer a um momento de entrega T_t, tal que, $T_t \leq TD_i$ e o valor de $T_t \geq u_i$ (momento de conclusão

da tarefa i que é igual o_{kl} momento de conclusão do lote k na maquina l em que i fora sequenciada, determinados no passo 1 ou 2). Se a condição não for verificada passar para o passo 5.

4. Realizar as respectivas alocações (No Lote k na Máquina l e nos veículo v no momento de Entrega T_t, e retirar a tarefa da lista LT_t a que pertence, faz-se $R = R + r_i$ e atualizam-se as taxas de ocupação dos veículos nos momentos de entrega e lotes nas máquinas e voltar ao passo 1.

5. A tarefa é inserida novamente na lista L e é eliminada da lista LT_t.

Então quando todas as listas LT_t estiverem vazias, retorna-se uma solução S definida em 3.3 que contém em sua estrutura a lista L final com as tarefas não sequenciadas.

A figura 3.2 apresenta um esquema como exemplo da aplicação do construtivo Heu3.

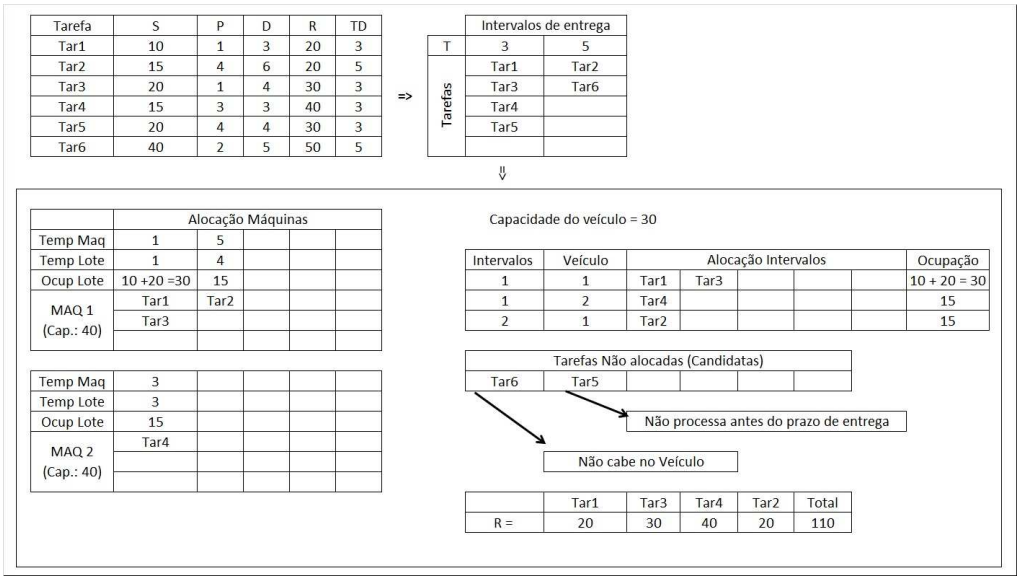

Figura 3.2: Esquema de uma Solução Obtida Através da Heurística Heu3

3.5.3 Variable Neighborhood Descent (VND)

A heurística Variable Neighborhood Search (VNS) foi proposta por Mlaswnovic & Hansen [1997]. A ideia principal do algorítmo proposto por Mlaswnovic & Hansen [1997] é definir um conjunto de estruturas de vizinhanças dado por $N_1, N_2, ..., N_k$, com k entre 1 e k_{max}, onde $N_k(x)$ representa a solução do *k-ésimo* vizinho de x. Dada uma solução inicial x, gera-se uma nova solução x' de maneira aleatória, para então aplicar a *k-ésima* (começando com $k = 1$) estrutura de vizinhança obtendo-se a solução x". Se a solução x" obtida for melhor que a solução inicial x então $x = x$" e faz-se $k = 1$, gera-se uma nova solução x' de maneira aleatória e realiza-se uma nova busca, aplicando-se a *k-ésima* estrutura de vizinhança para a obtenção de uma nova solução x". Caso a solução encontrada seja pior que a inicial então faz-se $k = k + 1$, gera-se uma nova solução x' de maneira aleatória e realiza-se uma nova busca, aplicando-se a *k-ésima* estrutura de vizinhança para a obtenção de uma nova solução x". O procedimento é realizado enquanto $k <= k_{max}$. Além da iteração controlada pelo número de estruturas de vizinhança, segundo os autores pode-se fazer uma iteração externa a última. A iteração externa pode ser controlada pelas seguintes condições de parada:

- Número máximo de iterações;
- máximo tempo de CPU;
- ou número máximo de iterações entre duas melhoras.

Segundo Gendeau & Potvin [2010] Variable Neighborhood Descent (VNS) é um método onde o desempenho das estruturas de vizinhança é obtido por meio de processo aleatório. Enquanto no VND o desempenho das estruturas de vizinhança é obtido por meio de processo determinístico.

No presente trabalho utilizou-se a heurística VND como uma busca local abrangendo três vizinhanças tratadas nas seções 3.5.3.1, 3.5.3.2 e 3.5.3.3. A heurística VND e a sistemática abordada na escolha das estruturas de vizinhança utilizadas estão apresentadas no algoritmo 1. Considerou-se como dado de entrada para a heurística uma lista de tarefas candidatas L e uma solução S definida em 3.3. A lista de candidatos dado como entrada do algoritmo trata-se da lista

originada do algorítimo de UB definida em 3.4 e a solução S resultante da heurística construtiva que a antecede. As vizinhaças são visitadas em cada iteração seguindo a seguinte ordem: Na primeira iteração executa-se a vizinhaça 3 dada como entrada S e L; na iteração seguinte se houver melhoria de solução volta-se a executar a vizinhaça 3, senão executa-se a vizinhaça seguinte (2 depois 1); quando na vizinhaça 1 não houver melhoria de solução o algoritmo é finalizado.

Algorithm 1 VND

```
função VND(L, S)
    j = 3
    enquanto true faça
        se j = 3 então
            SCorrente = BL3(L, S)
        fim se
        se j = 2 então
            SCorrente = BL2(L, S)
        fim se
        se j = 1 então
            SCorrente = BL1(L, S)
        fim se
        se SCorrente > S então
            S = SCorrente
            j = 3
        senão
            se j = 1 então
                break
            senão
                j − −
            fim se
        fim se
    fim enquanto
    retorna S
fim função
```

3.5.3.1 Busca Local 1 ($BL1$)

Como entrada para a $BL1$ tem-se uma solução S definida em 3.3 (na primeira iteração dada como entrada para o VND e nas iterações seguintes obtidas pela

execução dela mesma ou de outra estrutura de vizinhança, dependendo da iteração e da condição) e uma lista de candidatos L com as tarefas utilizadas como base para a construção da solução S a ser fornecida como entrada para a BL1.

O critério para estabelecer a vizinhança utilizada como base para a implementação da Busca local 1, trata-se da troca de ordem de duas tarefas da lista de candidatos L ordenada obtendo-se uma nova lista L'. Na sequência utiliza-se a nova lista L' como entrada para a construção de uma nova solução $S1$, pela heurística Heu3.

As estruturas de vizinhança podem ser tratadas realizando-se o caminhamento em toda vizinhança, denominada como *best improvement*. Neste caso, dada uma lista de candidatos como entrada, faz-se todas as trocas possíveis. Para cada troca realizada executa-se o construtivo Heu3 (definido na seção 3.5.2) e identifica-se o sequenciamento que possuir melhor valor de R. Outra estratégia para realizar o caminhamento é percorrer a vizinhança até encontrar o primeiro melhor vizinho, denominadas *first improvement*. Ou seja, as trocas na Lista L são realizadas enquanto não houver melhora na solução, quando ocorrer a primeira melhora deve-se parar as iterações do algorítmo.

O algoritmo 2 apresenta o pseudo-código da busca local considerando como estratégia o *best improvement*. O algoritmo 3 apresenta o pseudo-código da busca local considerando como estratégia o *First Improvement*.

Em ambos os algoritmos (2 e 3) retorna-se uma solução final S e a lista que deu origem a essa solução.

3.5.3.2 Busca Local 2 ($BL2$)

Dada uma solução inicial S definida em 3.3 (obtidas pela execução dela mesma ou de outra estrutura de vizinhança, dependendo da iteração e da condição) e uma lista de candidatos L com as tarefas utilizadas para a construção da solução S, como entrada para a $BL2$, constrói-se duas listas $L1$ e $L2$. A lista $L1$ receberá as tarefas sequenciadas, em $S0$. A lista $L2$ receberá as tarefas não sequenciadas em $S0$. Então troca-se uma tarefa de $L1$ com outra de $L2$, gerando duas novas listas $L1'$ e $L2'$. Na sequência constrói-se a lista de candidatos L', concatenando $L1'$ e $L2'$. Para então executar o construtivo definido na seção 3.5.2 para que se tenha

Algorithm 2 Busca Local 1 - Best Improvement

```
função Busca Local 1(S, L)
    RespostaCorrente = RespostaMelhor = S
    CCorrente = CMelhor = CMelhor = L
    enquanto true faça
        teste = 0
        para i ← 0 até Tamanho de L-1 faça
            para j ← i+1 até Tamanho de L faça
                i recebe uma posição de L, sorteada aleatóriamente
                j recebe uma posição de L, diferente de i, sorteada aleatóriamente
                CCorrente recebe a lista L e na sequência faz-se as trocas das
tarefas das posições i e j
                RespostaCorrente = heu3(CCorrente)
                se RespostaCorrente > RespostaMelhor então
                    RespostaMelhor = RespostaCorrente
                    CMelhor = CCorrente
                    teste = 1
                fim se
            fim para
        fim para
        se teste == 0 então break;
        fim se
    fim enquanto
    S = RespostaMelhor
    L = CMelhor
    retorna S e L
fim função
```

a nova solução $S1$.

O algoritmo 4 apresenta o pseudo código da $BL2$, usando como estratégia o *Best Improvement*. O algoritmo 5 apresenta o pseudo código da $BL2$, usando como estratégia o *First Improvement*.

Em ambos os algoritmos (4 e 5) retorna-se uma solução final S e a lista que deu origem a essa solução.

Algorithm 3 Busca Local 1 - First Improvement

```
função Busca Local 1(S, L)
    RespostaCorrente = RespostaMelhor = S
    CCorrente = CMelhor = L
    teste = 0
    para i ← 0 até Tamanho de L-1 && teste == 0 faça
        para j ← i+1 até Tamanho de L && teste == 0 faça
            i recebe uma posição de L, sorteada aleatóriamente
            j recebe uma posição de L diferente de i, sorteada aleatóriamente
            CCorrente recebe a lista L e na sequência faz-se as trocas das tarefas
das posições i e j
            RespostaCorrente = heu3(CCorrente)
            se RespostaCorrentel > RespostaMelhor então
                RespostaMelhor = RespostaCorrente
                CMelhor = CCorrente
                teste = 1
            fim se
        fim para
    fim para
    se teste = 0 então break;
    fim se
    S = RespostaMelhor
    L = CMelhor
    retorna S e L
fim função
```

3.5.3.3 Busca Local 3 ($BL3$)

Dadas a solução inicial S definida em 3.3 (obtidas pela execução dela mesma ou de outra estrutura de vizinhança, dependendo da iteração e da condição) , como entradas de $BL3$, constrói-se duas listas $L1$ e $L2$. A lista $L1$ recebe as tarefas sequenciadas em S e a lista $L2$ recebe as tarefas não sequenciadas em S. Então faz-se uma troca entre duas tarefas de $L1$ gerando uma nova lista $L1'$. Na sequência constrói-se a lista de candidatos L', concatenando $L1'$ e $L2$. Para então construir uma solução $S1$ através da heurística Heu3 definida na seção 3.5.2.

Realizar a simples troca de duas tarefas já alocadas, não fará com que o valor da função objetivo aumente. O valor da função objetivo só poderá aumentar se a troca for suficiente para reduzir o tempo de conclusão de uma máquina, a ponto

Algorithm 4 Busca Local 2 - Best Improvement

```
função Busca Local 2(S, L)
    RespostaCorrente = RespostaMelhor = S0
    CCorrente = CMelhor = L
    enquanto true faça
        L1 = RespostaMelhor_Tarefas_sequenciadas
        L2 = RespostaMelhor_Tarefas_nao_sequenciadas
        teste = 0
        para i ← 0 até Tamanho de L1 faça
            para j ← 0 até Tamanho de L2 faça
                i recebe uma posição de L1, sorteada aleatóriamente
                j recebe uma posição de L2 sorteada aleatóriamente
                troca-se a tarefa da posição i de L1 com a tarefa do posição j de
L2
                CCorrente recebe a lista L1 concatenada com a lista L2
                RespostaCorrente = heu3(CCorrente)
                se RespostaCorrente > RespostaMelhor então
                    RespostaMelhor = RespostaCorrente
                    CMelhor = CCorrente
                    teste = 1
                fim se
            fim para
        fim para
    fim enquanto
    S = RespostaMelhor
    L = CMelhor
    retorna S e L
fim função
```

de permitir sequenciar mais uma tarefa ainda não sequenciada. Por esse motivo é importante concatenar $L1'$ com $L2$.

O algoritmo 6 apresenta o pseudo código da busca local 3, utilizando-se a estratégia *Best Improvement*. O algoritmo 7 apresenta o pseudo código da busca local 3, com a estratégia *first improvement*.

Em ambos os algoritmos (6 e 7) retorna-se uma solução final S e a lista que deu origem a essa solução.

Algorithm 5 Busca Local 2 - First Improvement

```
função Busca Local 2(S, L)
    RespostaCorrente = RespostaMelhor = S0
    CCorrente = CMelhor = L
    L1 = RespostaMelhor_Tarefas_sequenciadas
    L2 = RespostaMelhor_Tarefas_nao_sequenciadas
    teste = 0
    para i ← 0 até Tamanho de L1 && teste==0 faça
        para j ← 0 até Tamanho de L2 && teste==0 faça
            i recebe uma posição de L1, sorteada aleatóriamente
            j recebe uma posição de L2 sorteada aleatóriamente
            troca-se a tarefa da posição i de L1 com a tarefa do posição j de L2
            CCorrente recebe a lista L1 concatenada com a lista L2
            RespostaCorrente = heu3(CCorrente)
            se RespostaCorrente > RespostaMelhor então
                RespostaMelhor = RespostaCorrente
                CMelhor = CCorrente
                teste = 1
            fim se
        fim para
    fim para
    S = RespostaMelhor
    L = CMelhor
    retorna S e L
fim função
```

3.5.4 Simulated Annealing (SA)

A heurística *Simulated Anealing* (SA) foi proposta por Kirkpatrick et al. [1983] e já foi utilizada para resolver diversos problemas de sequenciamento de tarefas em Máquinas Paralelas. Como exemplos de trabalhos de sequenciamento de tarefas onde foi aplicado o SA tem-se: Laarhoven et al. [1992], Lo & Bavarian [1992], Kim et al. [2002], Fidelis & Arroyo [2017].

Segundo Dowsland [1993], o SA baseia-se em conceitos da termodinâmica, mais precisamente a técnica chamada de recozimento. O recozimento trata-se do aquecimento do metal até a sua temperatura de ponto de fusão, e na sequência seu resfriamento lento até a temperatura ambiente, permitindo a obtenção de uma estrutura cristalina, livre de defeitos. Segundo o mesmo autor temperaturas altas

Algorithm 6 Busca Local 3 - *best improvement*

```
função Busca Local 3(S, L)
    RespostaCorrente = RespostaMelhor = S
    CCorrente = CMelhor = L
    enquanto true faça
        L1 = CMelhor_Tarefas_sequenciadas
        L2 = CMelhor_Tarefas_nao_sequenciadas
        teste = 0
        para i ← 0 até Tamanho de L1-1 faça
            para j ← i+1 até Tamanho de L1 faça
                i recebe uma posição de L1, sorteada aleatóriamente
                j recebe uma posição de L1, diferente de i sorteada aleatóriamente
                troca-se as tarefas da posição i e j da lista L1
                CCorrente recebe a lista L1 concatenada com a lista L2
                RespostaCorrente = heu3(CCorrente)
                se RespostaCorrente > RespostMelhor.R então
                    RespostaMelhor = RespostaCorrente
                    CMelhor = CCorrente
                    teste = 1
                fim se
            fim para
        fim para
        se teste == 0 então break;
        fim se
    fim enquanto
    S = RespostaMelhor
    L = CMelhor
    retorna S e L
fim função
```

estão associadas a alta probabilidade de aceitação de movimentos de piora, e à medida que a temperatura diminui a probabilidade de se aceitar movimentos de piora é reduzida.

O Algoritmo 8 apresenta a lógica de implementação do SA utilizada no presente trabalho. Além do critério de parada temperatura tem-se um critério de parada mais interno (CP) calculado pela equação 3.18 sendo: m o número de máquinas, n o número de tarefas, z o número de intervalos de entrega, e β recebendo um valor entre 0 e 1 a ser calibrado. O SA tem como entrada os parâmetros *alpha*,

Algorithm 7 Busca Local 3 - *first improvement*

```
função Busca Local 3(S, L)
    RespostaCorrente = RespostaMelhor = S
    CCorrente = CMelhor = L
    L1 = CMelhor_Tarefas_sequenciadas
    L2 = CMelhor_Tarefas_nao_sequenciadas
    teste = 0
    para i ← 0 até Tamanho de L1-1 && teste==0 faça
        para j ← i+1 até Tamanho de L1 && teste==0 faça
            i recebe uma posição de L1, sorteada aleatóriamente
            j recebe uma posição de L2 sorteada aleatóriamente
            troca-se a tarefa da posição i de L1 com a tarefa do posição j de L2
            CCorrente recebe a lista L1 concatenada com a lista L2
            RespostaCorrente = heu3(CCorrente)
            se RespostaCorrente > RespostaMelhor então
                RespostaMelhor = RespostaCorrente
                CMelhor = CCorrente
                teste = 1
            fim se
        fim para
    fim para
    S.Saida = RespostaMelhor
    L = CMelhor
    retorna S e L
fim função
```

T_{final}, CP, T e uma solução inicial S e L. A temperatura final adotada para o SA foi $T_{final} = 0,1$. Quanto a taxa de resfriamento (α) da heurística SA foi obtida a partir de calibração. O valor da temperatura inicial T foi identificada, para cada instância, de acordo com o pseudocódigo apresentado no algoritmo 9 para cada instância, o que faz com que o valor a ser assumido por T possa ser diferente para cada instância.

$$CP = m \times n \times z \times \beta \tag{3.18}$$

Dado como entrada uma lista de tarefas candidatas L e uma solução S definida em 3.3 (a lista de candidatos dado como entrada do algoritmo trata-se da lista originada do algorítimo de UB definida em 3.4 e a solução S resul-

tante da heurística construtiva que a antecede) no algoritmo 8 primeiramente inicializa-se $RespostaCorrente$ e $RespostaMelhor$ com o conteúdo da solução inicial (S). Na sequência enquanto a temperatura T for maior do que a temperatura T_{Final} faz-se: gera-se uma nova solução ($RespostaCorrente$), se o valor da função objetivo da nova solução ($RespostaCorrente$) for maior do que a solução inicial (S), atualiza-se a solução inicial (S) que passa a assumir o conteúdo de $RespostaCorrente$ e se a nova solução for maior do que a melhor solução encontrada até o momento ($RespostaMelhor$) então $RespostaMelhor$ passa a assumir também os valores de $RespostaCorrente$; Se o valor da função objetivo da nova solução ($RespostaCorrente$) for menor do que a solução inicial (S), avalia-se a probabilidade de se aceitar movimentos de piora e caso seja verdadeira faz-se $S = RespostaCorrente$; atualiza-se o valor de T.

A geração de uma solução ocorre da seguinte maneira: dada a lista de candidatos L sorteia-se duas posições i e j, tal que, $i \neq j$. Após realização do sorteio, as tarefas pertencentes as posições i e j são trocadas entre elas e por fim aplica-se a heurística Heu3 na lista L com as tarefas trocadas e obtém-se uma solução.

O cálculo da temperatura inicial conforme o algoritmo 9, trata-se da elevação da temperatura a partir de uma temperatura inicial $T = 0,01$ até que o número de soluções aceitas for igual a 80% do valor estabelecido como critério de parada (CP).

Algorithm 8 SA

função SA(S, L, α, T_{final}, CP, T)
  $RespostaCorrente = RespostaMelhor = S$
  enquanto $T > T_{final}$ **faça**
    para i $\leftarrow$ 0 **até** CP **faça**
      LCorrente = o resultado da troca de duas posições da lista L, obtidas por sorteios aleatórios
      RespostaCorrente recebe uma nova solução construída a partir da lista LCorrente, através da execução da heurística Heu3
      $\delta = RespostaCorrente - S$
      se $\delta > 0$ **então**
        $S = RespostaCorrente$
        $L = LCorrente$
        se $S > RespostaMelhor$ **então**
          $RespostaMelhor = S$
        fim se
      senão
        x = um valor aleatório entre 0 e 1
        se $x < e^{(\frac{\delta}{T})}$ **então**
          $S = RespostaCorrente$
          $L = LCorrente$
        fim se
      fim se
    fim para
    $T = T \times (1 - \alpha)$
  fim enquanto
  retorna $RespostaMelhor$
fim função

Algorithm 9 Cálculo da temperatura inicial

```
função CÁLCULO DA TEMPERATURA INICIAL(S,L,α, CP, T)
    RespostaCorrente = RespostaMelhor = S
    Taxadeaumentodetemperatura = 0.10
    T = 5
    enquanto aceitos < 0.80 × CP faça
        aceitos = 0
        para i ← 0 até CP faça
            LCorrente = o resultado da troca de duas posições da lista L, obtidas
por sorteios aleatórios
            RespostaCorrente = recebe uma nova solução construída a partir da
lista LCorrente, através da execução da heurística Heu3
            delta = RespostaCorrente − S
            se δ > 0 então
                aceitos + +
            senão
                x = um valor aleatório entre 0 e 1
                se x < e^(δ/T) então
                    aceitos + +
                fim se
            fim se
        fim para
        T = T × (1 + B)
    fim enquanto
    retorna T
fim função
```

Capítulo 4

Experimentos Computacionais

Neste capítulo, são apresentados o processo de geração das instâncias, a metodologia utilizada para calibrar os parâmetros do algoritmo SA descrito na Seção 3.5.4 e os resultados obtidos pelas heurísticas e pelo modelo de programação linear inteira mista ($PLIM$) propostos neste trabalho.

A formulação de programação linear inteira mista ($PLIM$) foi implementada via ILOG Concert Technology C++ e executada utilizando o software ILOG CPLEX, versão acadêmica 12.8, com todas as configurações padrões, exceto o tempo de execução que foi limitado à 1000 segundos. Já os algoritmos heurísticos propostos foram codificados na linguagem C++, compilados com GNU g++ versão 4.8.4 e executados de modo sequencial. Todos os experimentos foram realizados em um computador Opteron(TM) 6272 CPU @ 1400MHz, com 96GB de memória RAM e sistema operacional Ubuntu 14.04.5 LTS, 64 bits.

4.1 Geração de Instâncias

Uma vez que o Problema de Sequenciamento de Tarefas em Máquinas Paralelas de Processamento em Lotes ($STMPL$) com Entrega, considerando o tempo de processamento das tarefas diferentes não foi abordado na literatura até o início deste trabalho (agosto de 2015), foi necessário criar conjuntos de instâncias (problemas teste) para validar a qualidade das abordagens propostas. Portanto, para realização dos experimentos criaram-se duas famílias de instâncias referenciadas pelas

letras A e B. Tais famílias foram criadas baseando-se na metodologia utilizada na geração de instâncias proposta por Li et al. [2015] para o problema de sequenciamento de tarefas em máquinas paralelas com processamento em lotes e com entrega. A família A refere-se às instâncias de pequeno e médio porte, enquanto a família B às instâncias de grande porte.

A família A contém 360 instâncias obtidas a partir de 5 replicações de um conjunto base de 72 possibilidades de intervalos de valores dos quais os parâmetros são aleatoriamente escolhidos. Os intervalos de valores dos 72 conjuntos bases podem ser entendidos a partir da figura 4.1. Os conjuntos bases podem ser referenciados pela combinação dos índices a, b e c. Por exemplo, se $a = b = c = 0$, tem-se: $n = 20$, $z = 2$, $m = 3$, $T_z = 30$, $MaxVeic = 3$, $MinTem = 5$ e $MaxTem = 28$.

Na Figura 4.1, tratando-se do sequenciamento nos momentos de entrega tem-se: número de momentos de entrega (z), momento de entrega de maior valor (T_z), número máximo de veículos disponíveis por momento de entrega t ($MaxVeic$), número mínimo de veículos disponíveis para cada momento t ($MinVec$), de forma que $MinVeic \leq V_t \leq MaxVeic$. Além disso, tem-se $MinTem$ e $MaxTem$ como sendo o valor mínimo e máximo, respectivamente, que cada momento de entrega T_t pode assumir, tal que, $1 \leq t \leq z$. Quanto aos demais parâmetros tem-se: número de máquinas (m), número de tarefas (n) que assumem valores fixos a depender do conjunto base.

índice a	n	z
0	20	2
1	30	2
2	40	3
3	60	4
4	80	5
5	100	6

índice b	m
0	3
1	4
2	5

índice c	Tz	MaxVeíc	MinTem	MaxTem
0	30	3	5	28
1	45	3	6	39
2	30	7	5	28
3	45	7	6	39

Tamanho da		Ganho da Tarefa	
Min	Max	Min	Max
1	40	1	100

MinVeic
1
Capacidade da Máquina
40

Prazo de entrega	
min	max
T1	1,4xTz

Tempo de Processamento	
Min	Max
1	6

Figura 4.1: Base para a Geração das Instâncias.

Para cada tarefa i, os parâmetros ganho (r_i), tamanho da tarefa (s_i), tempo de processamento (p_i) e prazo de entrega (d_i), foram sorteados dentro dos intervalos

$[Min, Max]$ indicados na Figura 4.1. A capacidade da máquina é fixa de acordo com o valor em tabela e a capacidade do veículo é dada pela Equação (4.1).

$$\text{Capacidade_do_Veículo} = \frac{0,8 \times (\sum_{i=0}^{n} s_i)}{\sum_{t=0}^{z-1} v_t} \tag{4.1}$$

Todos os valores dos parâmetros foram escolhidos aleatoriamente, seguindo uma distribuição uniforme tendo como limites os respectivos intervalos estipulados. Para o problema, consideraram-se os parâmetros tempo de processamento, prazo de entrega e intervalos de entrega, assumindo valores inteiros e os demais valores reais.

Para facilitar a representação dos resultados para cada conjunto base de instâncias da família A representada por *abc*, foi criado um ID de forma que a primeira instancia 000 ($a = 0$, $b = 0$ e $c = 0$) receba o ID = 1, a segunda 001 ($a = 0$, $b = 0$ e $c = 1$) receba ID = 2 até o ultimo conjunto base de instâncias, variando primeiro o índice a, depois o b e, por fim, o índice c. A Tabela 4.1 apresenta a associação de cada índice com cada conjunto base para a família A de instâncias.

Tabela 4.1: Associação de cada Conjunto de Instâncias Base para a Família A de Instâncias com seu Respectivo ID.

ID	abc	ID	abc	ID	abc	ID	abc	ID	abc	ID	abc	ID	abc	ID	abc
1	000	10	021	19	112	28	203	37	300	46	321	55	412	64	503
2	001	11	022	20	113	29	210	38	301	47	322	56	413	65	510
3	002	12	023	21	120	30	211	39	302	48	323	57	420	66	511
4	003	13	100	22	121	31	212	40	303	49	400	58	421	67	512
5	010	14	101	23	122	32	213	41	310	50	401	59	422	68	513
6	011	15	102	24	123	33	220	42	311	51	402	60	423	69	520
7	012	16	103	25	200	34	221	43	312	52	403	61	500	70	521
8	013	17	110	26	201	35	222	44	313	53	410	62	501	71	522
9	020	18	111	27	202	36	223	45	320	54	411	63	502	72	523

A Tabela 4.2, representam as 72 possíveis combinações para o conjunto base. As colunas representam: ID do conjunto base, número de tarefas (n), número de máquinas (m), ganho se a tarefa for entregue no prazo (r), número de momentos de entrega (z), valores de tempos de entrega para cada um dos z momentos de entrega (T_z), número de veículos por momento de entrega (v), capacidade de cada máquina (K), capacidade dos veículos (C), prazo de entrega (d), tamanho de cada tarefa (s), e tempo de processamento de cada tarefa independente da máquina

(p). Os valores de r, T, v, d, s e p são valores sorteados dentro de um intervalo uniforme fechado A e B ($U[A, B]$).

Tabela 4.2: Características das Instâncias da Família A

ID	n	m	r	z	T	Tz	v	K	C	d	s	p
1	20	3	U[1,100]	2	U[5,28]	30	U[1,3]	40	$\frac{\sum_{i=0}^{n-1} S_i \times 0,8}{\sum_{t=0}^{T-1} v_t}$	U[T1, 1,4Tz]	U[1,40]	U[1,6]
2	20	3	U[1,100]	2	U[6,39]	45	U[1,3]	40	$\frac{\sum_{i=0}^{n-1} S_i \times 0,8}{\sum_{t=0}^{T-1} v_t}$	U[T1, 1,4Tz]	U[1,40]	U[1,6]
3	20	3	U[1,100]	2	U[5,28]	30	U[1,7]	40	$\frac{\sum_{i=0}^{n-1} S_i \times 0,8}{\sum_{t=0}^{T-1} v_t}$	U[T1, 1,4Tz]	U[1,40]	U[1,6]
4	20	3	U[1,100]	2	U[6,39]	45	U[1,7]	40	$\frac{\sum_{i=0}^{n-1} S_i \times 0,8}{\sum_{t=0}^{T-1} v_t}$	U[T1, 1,4Tz]	U[1,40]	U[1,6]
5	20	4	U[1,100]	2	U[5,28]	30	U[1,3]	40	$\frac{\sum_{i=0}^{n-1} S_i \times 0,8}{\sum_{t=0}^{T-1} v_t}$	U[T1, 1,4Tz]	U[1,40]	U[1,6]
6	20	4	U[1,100]	2	U[6,39]	45	U[1,3]	40	$\frac{\sum_{i=0}^{n-1} S_i \times 0,8}{\sum_{t=0}^{T-1} v_t}$	U[T1, 1,4Tz]	U[1,40]	U[1,6]
7	20	4	U[1,100]	2	U[5,28]	30	U[1,7]	40	$\frac{\sum_{i=0}^{n-1} S_i \times 0,8}{\sum_{t=0}^{T-1} v_t}$	U[T1, 1,4Tz]	U[1,40]	U[1,6]
8	20	4	U[1,100]	2	U[6,39]	45	U[1,7]	40	$\frac{\sum_{i=0}^{n-1} S_i \times 0,8}{\sum_{t=0}^{T-1} v_t}$	U[T1, 1,4Tz]	U[1,40]	U[1,6]
9	20	5	U[1,100]	2	U[5,28]	30	U[1,3]	40	$\frac{\sum_{i=0}^{n-1} S_i \times 0,8}{\sum_{t=0}^{T-1} v_t}$	U[T1, 1,4Tz]	U[1,40]	U[1,6]
10	20	5	U[1,100]	2	U[6,39]	45	U[1,3]	40	$\frac{\sum_{i=0}^{n-1} S_i \times 0,8}{\sum_{t=0}^{T-1} v_t}$	U[T1, 1,4Tz]	U[1,40]	U[1,6]
11	20	5	U[1,100]	2	U[5,28]	30	U[1,7]	40	$\frac{\sum_{i=0}^{n-1} S_i \times 0,8}{\sum_{t=0}^{T-1} v_t}$	U[T1, 1,4Tz]	U[1,40]	U[1,6]
12	20	5	U[1,100]	2	U[6,39]	45	U[1,7]	40	$\frac{\sum_{i=0}^{n-1} S_i \times 0,8}{\sum_{t=0}^{T-1} v_t}$	U[T1, 1,4Tz]	U[1,40]	U[1,6]
13	30	3	U[1,100]	2	U[5,28]	30	U[1,3]	40	$\frac{\sum_{i=0}^{n-1} S_i \times 0,8}{\sum_{t=0}^{T-1} v_t}$	U[T1, 1,4Tz]	U[1,40]	U[1,6]
14	30	3	U[1,100]	2	U[6,39]	45	U[1,3]	40	$\frac{\sum_{i=0}^{n-1} S_i \times 0,8}{\sum_{t=0}^{T-1} v_t}$	U[T1, 1,4Tz]	U[1,40]	U[1,6]
15	30	3	U[1,100]	2	U[5,28]	30	U[1,7]	40	$\frac{\sum_{i=0}^{n-1} S_i \times 0,8}{\sum_{t=0}^{T-1} v_t}$	U[T1, 1,4Tz]	U[1,40]	U[1,6]
16	30	3	U[1,100]	2	U[6,39]	45	U[1,7]	40	$\frac{\sum_{i=0}^{n-1} S_i \times 0,8}{\sum_{t=0}^{T-1} v_t}$	U[T1, 1,4Tz]	U[1,40]	U[1,6]
17	30	4	U[1,100]	2	U[5,28]	30	U[1,3]	40	$\frac{\sum_{i=0}^{n-1} S_i \times 0,8}{\sum_{t=0}^{T-1} v_t}$	U[T1, 1,4Tz]	U[1,40]	U[1,6]
18	30	4	U[1,100]	2	U[6,39]	45	U[1,3]	40	$\frac{\sum_{i=0}^{n-1} S_i \times 0,8}{\sum_{t=0}^{T-1} v_t}$	U[T1, 1,4Tz]	U[1,40]	U[1,6]
19	30	4	U[1,100]	2	U[5,28]	30	U[1,7]	40	$\frac{\sum_{i=0}^{n-1} S_i \times 0,8}{\sum_{t=0}^{T-1} v_t}$	U[T1, 1,4Tz]	U[1,40]	U[1,6]
20	30	4	U[1,100]	2	U[6,39]	45	U[1,7]	40	$\frac{\sum_{i=0}^{n-1} S_i \times 0,8}{\sum_{t=0}^{T-1} v_t}$	U[T1, 1,4Tz]	U[1,40]	U[1,6]
21	30	5	U[1,100]	2	U[5,28]	30	U[1,3]	40	$\frac{\sum_{i=0}^{n-1} S_i \times 0,8}{\sum_{t=0}^{T-1} v_t}$	U[T1, 1,4Tz]	U[1,40]	U[1,6]
22	30	5	U[1,100]	2	U[6,39]	45	U[1,3]	40	$\frac{\sum_{i=0}^{n-1} S_i \times 0,8}{\sum_{t=0}^{T-1} v_t}$	U[T1, 1,4Tz]	U[1,40]	U[1,6]
23	30	5	U[1,100]	2	U[5,28]	30	U[1,7]	40	$\frac{\sum_{i=0}^{n-1} S_i \times 0,8}{\sum_{t=0}^{T-1} v_t}$	U[T1, 1,4Tz]	U[1,40]	U[1,6]
24	30	5	U[1,100]	2	U[6,39]	45	U[1,7]	40	$\frac{\sum_{i=0}^{n-1} S_i \times 0,8}{\sum_{t=0}^{T-1} v_t}$	U[T1, 1,4Tz]	U[1,40]	U[1,6]
25	40	3	U[1,100]	3	U[5,28]	30	U[1,3]	40	$\frac{\sum_{i=0}^{n-1} S_i \times 0,8}{\sum_{t=0}^{T-1} v_t}$	U[T1, 1,4Tz]	U[1,40]	U[1,6]
26	40	3	U[1,100]	3	U[6,39]	45	U[1,3]	40	$\frac{\sum_{i=0}^{n-1} S_i \times 0,8}{\sum_{t=0}^{T-1} v_t}$	U[T1, 1,4Tz]	U[1,40]	U[1,6]
27	40	3	U[1,100]	3	U[5,28]	30	U[1,7]	40	$\frac{\sum_{i=0}^{n-1} S_i \times 0,8}{\sum_{t=0}^{T-1} v_t}$	U[T1, 1,4Tz]	U[1,40]	U[1,6]
28	40	3	U[1,100]	3	U[6,39]	45	U[1,7]	40	$\frac{\sum_{i=0}^{n-1} S_i \times 0,8}{\sum_{t=0}^{T-1} v_t}$	U[T1, 1,4Tz]	U[1,40]	U[1,6]
29	40	4	U[1,100]	3	U[5,28]	30	U[1,3]	40	$\frac{\sum_{i=0}^{n-1} S_i \times 0,8}{\sum_{t=0}^{T-1} v_t}$	U[T1, 1,4Tz]	U[1,40]	U[1,6]
30	40	4	U[1,100]	3	U[6,39]	45	U[1,3]	40	$\frac{\sum_{i=0}^{n-1} S_i \times 0,8}{\sum_{t=0}^{T-1} v_t}$	U[T1, 1,4Tz]	U[1,40]	U[1,6]
31	40	4	U[1,100]	3	U[5,28]	30	U[1,7]	40	$\frac{\sum_{i=0}^{n-1} S_i \times 0,8}{\sum_{t=0}^{T-1} v_t}$	U[T1, 1,4Tz]	U[1,40]	U[1,6]
32	40	4	U[1,100]	3	U[6,39]	45	U[1,7]	40	$\frac{\sum_{i=0}^{n-1} S_i \times 0,8}{\sum_{t=0}^{T-1} v_t}$	U[T1, 1,4Tz]	U[1,40]	U[1,6]
33	40	5	U[1,100]	3	U[5,28]	30	U[1,3]	40	$\frac{\sum_{i=0}^{n-1} S_i \times 0,8}{\sum_{t=0}^{T-1} v_t}$	U[T1, 1,4Tz]	U[1,40]	U[1,6]
34	40	5	U[1,100]	3	U[6,39]	45	U[1,3]	40	$\frac{\sum_{i=0}^{n-1} S_i \times 0,8}{\sum_{t=0}^{T-1} v_t}$	U[T1, 1,4Tz]	U[1,40]	U[1,6]
35	40	5	U[1,100]	3	U[5,28]	30	U[1,7]	40	$\frac{\sum_{i=0}^{n-1} S_i \times 0,8}{\sum_{t=0}^{T-1} v_t}$	U[T1, 1,4Tz]	U[1,40]	U[1,6]
36	40	5	U[1,100]	3	U[6,39]	45	U[1,7]	40	$\frac{\sum_{i=0}^{n-1} S_i \times 0,8}{\sum_{t=0}^{T-1} v_t}$	U[T1, 1,4Tz]	U[1,40]	U[1,6]
37	60	3	U[1,100]	4	U[5,28]	30	U[1,3]	40	$\frac{\sum_{i=0}^{n-1} S_i \times 0,8}{\sum_{t=0}^{T-1} v_t}$	U[T1, 1,4Tz]	U[1,40]	U[1,6]
38	60	3	U[1,100]	4	U[6,39]	45	U[1,3]	40	$\frac{\sum_{i=0}^{n-1} S_i \times 0,8}{\sum_{t=0}^{T-1} v_t}$	U[T1, 1,4Tz]	U[1,40]	U[1,6]
39	60	3	U[1,100]	4	U[5,28]	30	U[1,7]	40	$\frac{\sum_{i=0}^{n-1} S_i \times 0,8}{\sum_{t=0}^{T-1} v_t}$	U[T1, 1,4Tz]	U[1,40]	U[1,6]
40	60	3	U[1,100]	4	U[6,39]	45	U[1,7]	40	$\frac{\sum_{i=0}^{n-1} S_i \times 0,8}{\sum_{t=0}^{T-1} v_t}$	U[T1, 1,4Tz]	U[1,40]	U[1,6]
41	60	4	U[1,100]	4	U[5,28]	30	U[1,3]	40	$\frac{\sum_{i=0}^{n-1} S_i \times 0,8}{\sum_{t=0}^{T-1} v_t}$	U[T1, 1,4Tz]	U[1,40]	U[1,6]
42	60	4	U[1,100]	4	U[6,39]	45	U[1,3]	40	$\frac{\sum_{i=0}^{n-1} S_i \times 0,8}{\sum_{t=0}^{T-1} v_t}$	U[T1, 1,4Tz]	U[1,40]	U[1,6]
43	60	4	U[1,100]	4	U[5,28]	30	U[1,7]	40	$\frac{\sum_{i=0}^{n-1} S_i \times 0,8}{\sum_{t=0}^{T-1} v_t}$	U[T1, 1,4Tz]	U[1,40]	U[1,6]
44	60	4	U[1,100]	4	U[6,39]	45	U[1,7]	40	$\frac{\sum_{i=0}^{n-1} S_i \times 0,8}{\sum_{t=0}^{T-1} v_t}$	U[T1, 1,4Tz]	U[1,40]	U[1,6]
45	60	5	U[1,100]	4	U[5,28]	30	U[1,3]	40	$\frac{\sum_{i=0}^{n-1} S_i \times 0,8}{\sum_{t=0}^{T-1} v_t}$	U[T1, 1,4Tz]	U[1,40]	U[1,6]
46	60	5	U[1,100]	4	U[6,39]	45	U[1,3]	40	$\frac{\sum_{i=0}^{n-1} S_i \times 0,8}{\sum_{t=0}^{T-1} v_t}$	U[T1, 1,4Tz]	U[1,40]	U[1,6]
47	60	5	U[1,100]	4	U[5,28]	30	U[1,7]	40	$\frac{\sum_{i=0}^{n-1} S_i \times 0,8}{\sum_{t=0}^{T-1} v_t}$	U[T1, 1,4Tz]	U[1,40]	U[1,6]
48	60	5	U[1,100]	4	U[6,39]	45	U[1,7]	40	$\frac{\sum_{i=0}^{n-1} S_i \times 0,8}{\sum_{t=0}^{T-1} v_t}$	U[T1, 1,4Tz]	U[1,40]	U[1,6]
49	80	3	U[1,100]	5	U[5,28]	30	U[1,3]	40	$\frac{\sum_{i=0}^{n-1} S_i \times 0,8}{\sum_{t=0}^{T-1} v_t}$	U[T1, 1,4Tz]	U[1,40]	U[1,6]
50	80	3	U[1,100]	5	U[6,39]	45	U[1,3]	40	$\frac{\sum_{i=0}^{n-1} S_i \times 0,8}{\sum_{t=0}^{T-1} v_t}$	U[T1, 1,4Tz]	U[1,40]	U[1,6]
51	80	3	U[1,100]	5	U[5,28]	30	U[1,7]	40	$\frac{\sum_{i=0}^{n-1} S_i \times 0,8}{\sum_{t=0}^{T-1} v_t}$	U[T1, 1,4Tz]	U[1,40]	U[1,6]
52	80	3	U[1,100]	5	U[6,39]	45	U[1,7]	40	$\frac{\sum_{i=0}^{n-1} S_i \times 0,8}{\sum_{t=0}^{T-1} v_t}$	U[T1, 1,4Tz]	U[1,40]	U[1,6]
53	80	4	U[1,100]	5	U[5,28]	30	U[1,3]	40	$\frac{\sum_{i=0}^{n-1} S_i \times 0,8}{\sum_{t=0}^{T-1} v_t}$	U[T1, 1,4Tz]	U[1,40]	U[1,6]
54	80	4	U[1,100]	5	U[6,39]	45	U[1,3]	40	$\frac{\sum_{i=0}^{n-1} S_i \times 0,8}{\sum_{t=0}^{T-1} v_t}$	U[T1, 1,4Tz]	U[1,40]	U[1,6]
55	80	4	U[1,100]	5	U[5,28]	30	U[1,7]	40	$\frac{\sum_{i=0}^{n-1} S_i \times 0,8}{\sum_{t=0}^{T-1} v_t}$	U[T1, 1,4Tz]	U[1,40]	U[1,6]
56	80	4	U[1,100]	5	U[6,39]	45	U[1,7]	40	$\frac{\sum_{i=0}^{n-1} S_i \times 0,8}{\sum_{t=0}^{T-1} v_t}$	U[T1, 1,4Tz]	U[1,40]	U[1,6]
57	80	5	U[1,100]	5	U[5,28]	30	U[1,3]	40	$\frac{\sum_{i=0}^{n-1} S_i \times 0,8}{\sum_{t=0}^{T-1} v_t}$	U[T1, 1,4Tz]	U[1,40]	U[1,6]
58	80	5	U[1,100]	5	U[6,39]	45	U[1,3]	40	$\frac{\sum_{i=0}^{n-1} S_i \times 0,8}{\sum_{t=0}^{T-1} v_t}$	U[T1, 1,4Tz]	U[1,40]	U[1,6]
59	80	5	U[1,100]	5	U[5,28]	30	U[1,7]	40	$\frac{\sum_{i=0}^{n-1} S_i \times 0,8}{\sum_{t=0}^{T-1} v_t}$	U[T1, 1,4Tz]	U[1,40]	U[1,6]
60	80	5	U[1,100]	5	U[6,39]	45	U[1,7]	40	$\frac{\sum_{i=0}^{n-1} S_i \times 0,8}{\sum_{t=0}^{T-1} v_t}$	U[T1, 1,4Tz]	U[1,40]	U[1,6]
61	100	6	U[1,100]	6	U[5,28]	30	U[1,3]	40	$\frac{\sum_{i=0}^{n-1} S_i \times 0,8}{\sum_{t=0}^{T-1} v_t}$	U[T1, 1,4Tz]	U[1,40]	U[1,6]
62	100	3	U[1,100]	6	U[6,39]	45	U[1,3]	40	$\frac{\sum_{i=0}^{n-1} S_i \times 0,8}{\sum_{t=0}^{T-1} v_t}$	U[T1, 1,4Tz]	U[1,40]	U[1,6]
63	100	3	U[1,100]	6	U[5,28]	30	U[1,7]	40	$\frac{\sum_{i=0}^{n-1} S_i \times 0,8}{\sum_{t=0}^{T-1} v_t}$	U[T1, 1,4Tz]	U[1,40]	U[1,6]
64	100	3	U[1,100]	6	U[6,39]	45	U[1,7]	40	$\frac{\sum_{i=0}^{n-1} S_i \times 0,8}{\sum_{t=0}^{T-1} v_t}$	U[T1, 1,4Tz]	U[1,40]	U[1,6]
65	100	4	U[1,100]	6	U[5,28]	30	U[1,3]	40	$\frac{\sum_{i=0}^{n-1} S_i \times 0,8}{\sum_{t=0}^{T-1} v_t}$	U[T1, 1,4Tz]	U[1,40]	U[1,6]
66	100	4	U[1,100]	6	U[6,39]	45	U[1,3]	40	$\frac{\sum_{i=0}^{n-1} S_i \times 0,8}{\sum_{t=0}^{T-1} v_t}$	U[T1, 1,4Tz]	U[1,40]	U[1,6]
67	100	4	U[1,100]	6	U[5,28]	30	U[1,7]	40	$\frac{\sum_{i=0}^{n-1} S_i \times 0,8}{\sum_{t=0}^{T-1} v_t}$	U[T1, 1,4Tz]	U[1,40]	U[1,6]
68	100	4	U[1,100]	6	U[6,39]	45	U[1,7]	40	$\frac{\sum_{i=0}^{n-1} S_i \times 0,8}{\sum_{t=0}^{T-1} v_t}$	U[T1, 1,4Tz]	U[1,40]	U[1,6]
69	100	5	U[1,100]	6	U[5,28]	30	U[1,3]	40	$\frac{\sum_{i=0}^{n-1} S_i \times 0,8}{\sum_{t=0}^{T-1} v_t}$	U[T1, 1,4Tz]	U[1,40]	U[1,6]
70	100	5	U[1,100]	6	U[6,39]	45	U[1,3]	40	$\frac{\sum_{i=0}^{n-1} S_i \times 0,8}{\sum_{t=0}^{T-1} v_t}$	U[T1, 1,4Tz]	U[1,40]	U[1,6]
71	100	5	U[1,100]	6	U[5,28]	30	U[1,7]	40	$\frac{\sum_{i=0}^{n-1} S_i \times 0,8}{\sum_{t=0}^{T-1} v_t}$	U[T1, 1,4Tz]	U[1,40]	U[1,6]
72	100	5	U[1,100]	6	U[6,39]	45	U[1,7]	40	$\frac{\sum_{i=0}^{n-1} S_i \times 0,8}{\sum_{t=0}^{T-1} v_t}$	U[T1, 1,4Tz]	U[1,40]	U[1,6]

A família B de instâncias contém 20 instâncias obtidas a partir de 5 replicações de um conjunto base de 4 possibilidades de intervalos de valores dos quais os parâmetros são sorteados. As características de cada conjunto base estão representadas na Tabela 4.3, em que as colunas têm o mesmo significado que aqueles apresentados nas colunas da Tabela 4.2.

Tabela 4.3: Características das Instâncias da Família B.

ID	n	m	R	z	T	Tz	v	k	c	D	S	p
1	300	3	U[1,100]	4	U[5,39]	45	U[1,3]	40	480	U[T1, 1,4Tz]	U[1,40]	U[1,10]
2	450	3	U[1,100]	6	U[5,39]	60	U[1,3]	40	480	U[T1, 1,4Tz]	U[1,40]	U[1,10]
3	500	5	U[1,100]	4	U[5,39]	45	U[1,3]	40	480	U[T1, 1,4Tz]	U[1,40]	U[1,10]
4	750	5	U[1,100]	6	U[5,39]	60	U[1,3]	40	480	U[T1, 1,4Tz]	U[1,40]	U[1,10]

4.2 Calibração dos Parâmetros do Algoritmo SA

As possibilidades de combinações de parâmetros tratam-se da aplicação de cada valor de β e cada valor α para cada instância. Sendo β um valor entre 0 e 1 aplicado à Equação 3.18 para o cálculo do critério de parada e α o parâmetro que estabelece a taxa de resfriamento da heurística SA. A Tabela 4.4 apresenta os possíveis tratamentos com os valores de α e β utilizados durante a calibração.

Tabela 4.4: Combinações de Parâmetros Testadas na Calibração

Tratamento	α	β	Tratamento	α	β
T1	0,01	0,2	T11	0,05	0,2
T2	0,01	0,4	T12	0,05	0,4
T3	0,01	0,6	T13	0,05	0,6
T4	0,01	0,8	T14	0,05	0,8
T5	0,01	1,0	T15	0,05	1,0
T6	0,02	0,2	T16	0,10	0,2
T7	0,02	0,4	T17	0,10	0,4
T8	0,02	0,6	T18	0,10	0,6
T9	0,02	0,8	T19	0,10	0,8
T10	0,02	1,0	T20	0,10	1,0

Foram geradas outras instâncias testes, de acordo com a lógica dos conjuntos bases definidos na Seção 4.1 . No caso da família A, foram geradas 72 instâncias uma para cada conjunto base. Quanto a família B, foram geradas 2 instâncias de cada conjunto base, perfazendo 8 instâncias. Para cada tratamento e cada instância foram realizadas 20 execuções durante a calibração de parâmetros.

Dadas as soluções encontradas, calculou-se o RPD para cada instância e cada tratamento. Na sequência, classificou-se as instâncias por tratamento e calculou-se a média dos valores de RPD das instâncias para cada tratamento, considerando-se

as 20 replicações do tratamento para cada instâncias. Na sequencia, aplicou-se ANOVA e o teste de *Scott-Knott*.

O cálculo do RPD é dado pela equação 4.2, sendo FO a solução da replicação e instância corrente para cada tratamento, e $FO_melhorSolucao$ o melhor valor de FO obtido dentre todas as replicas para cada instância e cada tratamento.

$$RPD = \frac{FO - FO_melhorSolucao}{FO_melhorSolucao} \times 100\% \tag{4.2}$$

A análise de variância (ANOVA) foi utilizada para determinar se as diferenças observadas são estatisticamente significantes com um nível de confiança de 99%.

Através da análise de *p-value* durante a ANOVA, observa-se que existem pelo menos dois tratamentos com desempenho significativamente diferentes, visto que o *p-value* em todos os casos assumiu um valor menor do que o nível de significância estabelecido de 0,01. No caso da análise das instâncias da família A, $p - value = 2.5443e{-}86$, quanto as instâncias da família B, $p{-}value = 5.4965e{-}112$ e a análise de todo o conjunto de instâncias (família A mais família B) $p - value = 7.4894e - 125$. Então visto que existe uma diferença de desempenhos dos tratamentos, fez-se o teste de *Scott-Knott* (Scott & Knott [1974]) para determinar quais tratamentos apresentam diferenças significativa entre si.

As Figuras 4.2 e 4.5 apresentam os resultados do teste de Scott e Knott para as instâncias das Famílias A e B em formas de tabelas. Essas tabelas apresentam quatro colunas. Em cada uma das tabelas a primeira linha apresenta o ID da linha, a segunda coluna apresenta o identificador Grupos que índica se os tratamentos obtiveram resultados com diferenças significativa entre si (tratamentos pertencentes a um mesmo grupo não apresentaram diferenças significativas e pertencentes a grupos distintos apresentaram diferenças significativas), a terceira coluna apresenta o tratamento e a última coluna apresenta a média dos RPD's por tratamento.

Os *boxplot* apresentados nas Figuras 4.3, 4.4 e 4.6 apresentam a dispersão dos valores de RPD para cada tratamento, considerando todas as instâncias e as replicações, por meio de quartis representados pelas linhas horizontais que fazem parte do limite da caixa que o representa (primeiro e terceiro quartil e a linha central que representa o segundo quartil ou mediana. Além dessas linhas o *boxplot* apresenta uma reta vertical chamada fio de bigode que estende–se verticalmente

a partir dos limites superior e inferior da caixa que representam as amplitudes para 25% dos valores de dados acima e abaixo da caixa, excluindo *outliers* (valores atípicos). Ainda apresenta pontos fora da estrutura já explicada denominados *outliers*, que representam os valores atípicos ou discrepantes em relação a massa de dados analisados.

Por meio das figuras supracitadas, percebe-se que em todos os casos que o tratamento T6 (combinação de parâmetros com $\alpha = 0,1$ e $\beta = 0,2$) apresentou os melhores resultados para o SA, considerando-se o problema estudado. Desta maneira optou-se por fazer uso da combinação dos parametros do tratamento T6 durante a execução da heurística SA.

Família A

Teste de Scott-Knott

	Grupos	Tratamentos	Medias
1	a	T5	9.074787
2	a	T4	8.934683
3	a	T10	8.582845
4	a	T3	8.557010
5	b	T9	8.345948
6	b	T2	8.313026
7	c	T8	7.783142
8	d	T7	7.392705
9	d	T1	7.346384
10	d	T15	7.329755
11	d	T14	7.113230
12	e	T13	6.600727
13	f	T6	6.217206
14	f	T12	6.013209
15	f	T20	5.977245
16	f	T19	5.808967
17	g	T18	5.042818
18	h	T11	4.322693
19	h	T17	4.310608
20	i	T16	2.494887

Família B

Teste de Scott-Knott

	Grupos	Tratamentos	Medias
1	a	T5	0.820453
2	a	T4	0.815950
3	b	T3	0.789644
4	b	T10	0.774664
5	b	T2	0.768447
6	c	T9	0.752048
7	c	T8	0.738078
8	c	T1	0.724613
9	c	T7	0.718781
10	c	T15	0.715627
11	c	T14	0.700310
12	d	T13	0.649180
13	d	T6	0.629774
14	d	T20	0.626929
15	d	T19	0.624915
16	d	T12	0.622182
17	e	T18	0.588344
18	f	T17	0.554128
19	f	T11	0.529873
20	g	T16	0.397627

Figura 4.2: Testes de Scott Knott - Calibração de parâmetros do SA para as Instâncias da Família A e B Separadamente

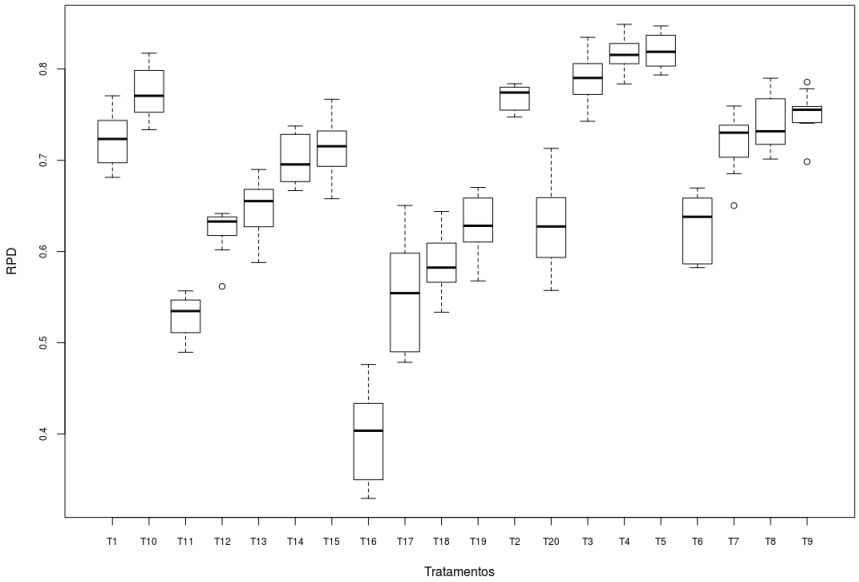

Figura 4.3: *Box-Plot* - Calibração de parâmetros para as instâncias da família A

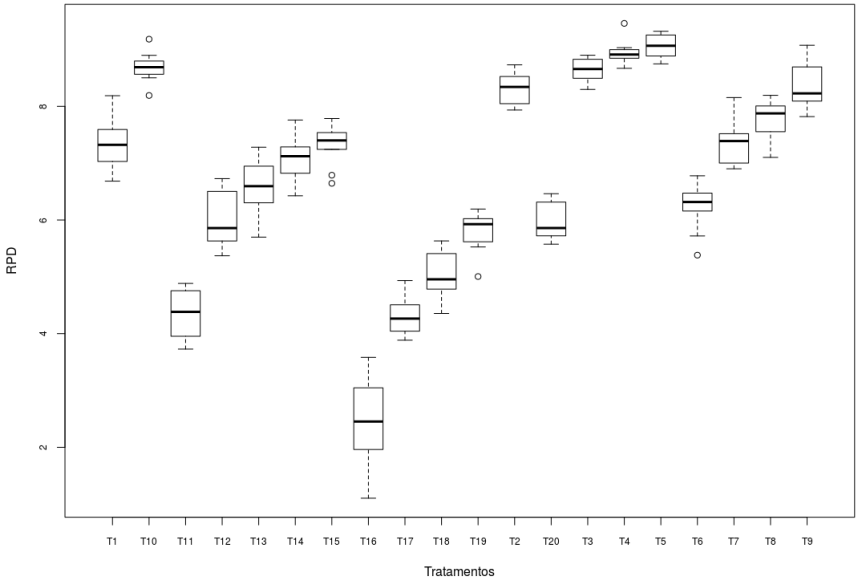

Figura 4.4: *Box-Plot* - Calibração de Parâmetros para as Instâncias da Família B

Teste de Scott-Knott

	Grupos	Tratamentos	Medias
1	a	T5	1.645884
2	a	T4	1.627825
3	b	T3	1.576380
4	b	T10	1.565483
5	c	T2	1.522904
6	c	T9	1.511436
7	d	T8	1.442584
8	e	T1	1.386789
9	e	T7	1.386172
10	e	T15	1.377039
11	e	T14	1.341603
12	f	T13	1.244336
13	g	T6	1.188517
14	g	T20	1.161962
15	g	T12	1.161285
16	g	T19	1.143320
17	h	T18	1.033791
18	i	T17	0.929774
19	i	T11	0.909154
20	j	T16	0.607354

Figura 4.5: Testes de Scott Knott - Calibração de parâmetros do SA para as Instâncias da Família A e B em uma Única Análise

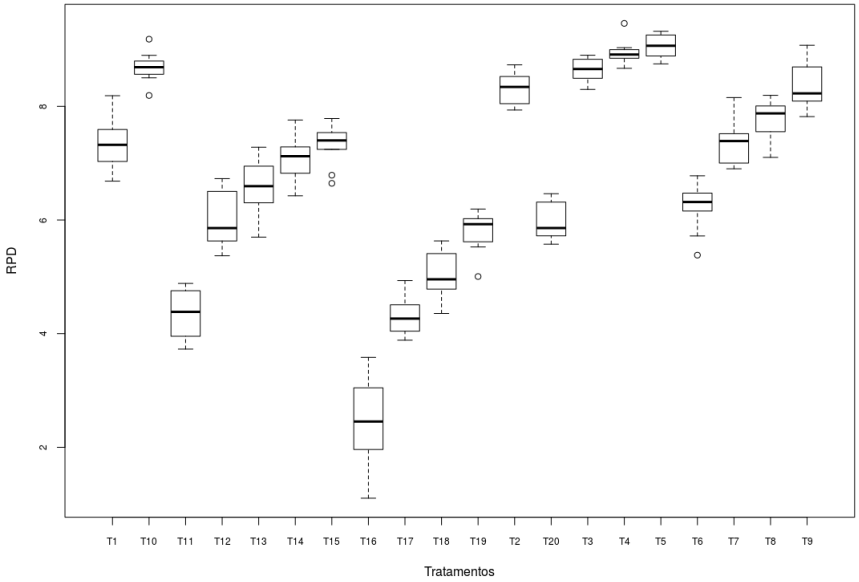

Figura 4.6: *Box-Plot* - Calibração de Parâmetros de SA considerando todas as Instâncias Teste

4.3 Resultados Experimentais

Nesta seção são apresentados os resultados obtidos através do modelo de programação linear inteira mista (PLIM) e pelas heurísticas propostas neste trabalho.

4.3.1 Resultados Experimentais para as Instâncias da Família A

A Tabela 4.5 apresenta as soluções encontradas pelo modelo e o GAP fornecido pelo CPLEX para as instancias da familia A. Neste caso, na primeira coluna acrescenta-se um índice d em abc, que irá representar as réplicas de cada conjunto de instância dentre os 72 conjuntos trabalhados. Ao analisar as soluções apresentadas na Tabela 4.5, observou-se que dentre as 360 instâncias testadas conseguiu-se alcançar o valor de GAP nulo em 53% dos casos. Dentre os 47% cujo o GAP se apresentou com valores maiores do que 0, para 4 instâncias nenhuma solução viável foi encontrada pelo modelo matemático considerando o tempo limite de execução para o modelo de 1000 segundos. Duas das instâncias sem solução pertencem ao conjunto base 72, uma ao conjunto base 71 e uma ao conjunto base 69. Essas observações mostram a dificuldade do modelo matemático proposto em resolver as instâncias maiores, visto que já apresenta dificuldade em resolver as intancias de médio porte da família A, considerando o problema tratado. Visto essa dificuldade, a alternativa utilizada para resolver tais instâncias foi utilizar algoritmos heurísticos, que buscam encontrar soluções de qualidade, mas sem garantia de otimalidade.

Tabela 4.5: Resultados do Modelo para as intancias da família A

Inst (abcd)	FO	Tempo	GAP	Inst (abcd)	FO	Tempo	GAP	Inst (abcd)	FO	Tempo	GAP	Inst (abcd)	FO	Tempo	GAP
0000	879.000	3.893	0.000	1120	1412.000	3.718	0.000	3000	2303.000	1000.000	0.192	4120	3148.000	1000.000	0.183
0001	1017.000	0.637	0.000	1121	1556.000	0.909	0.000	3001	2752.000	1000.000	0.142	4121	3427.000	1000.000	0.151
0002	1140.000	3.789	0.000	1122	1575.000	1.219	0.000	3002	2867.000	1000.000	0.115	4122	3817.000	1000.000	0.157
0003	771.000	0.684	0.000	1123	1688.000	2.127	0.000	3003	2717.000	1000.000	0.062	4123	3770.000	1000.000	0.060
0004	992.000	0.278	0.000	1124	1474.000	0.980	0.000	3004	2584.000	1000.000	0.072	4124	3425.000	1000.000	0.125
0010	1025.000	3.994	0.000	1130	1564.000	0.977	0.000	3010	2950.000	1000.000	0.011	4130	3922.000	1000.000	0.007
0011	1032.000	0.372	0.000	1131	1174.000	1.236	0.000	3011	3005.000	1000.000	0.003	4131	4054.000	185.512	0.000
0012	1057.000	0.166	0.000	1132	1205.000	1000.000	0.061	3012	3154.000	94.656	0.000	4132	3962.000	1000.000	0.010
0013	1292.000	0.193	0.000	1133	1521.000	1.132	0.000	3013	3121.000	1000.000	0.004	4133	3729.000	1000.000	0.037
0014	973.000	0.534	0.000	1134	1619.000	1.092	0.000	3014	2799.000	1000.000	0.019	4134	3950.000	1000.000	0.019
0020	956.000	0.335	0.000	1200	1407.000	1.538	0.000	3020	2269.000	1000.000	0.268	4200	3513.000	1000.000	0.169
0021	1171.000	0.536	0.000	1201	1499.000	3.352	0.000	3021	3263.000	1000.000	0.057	4201	3193.000	1000.000	0.170
0022	975.000	1.125	0.000	1202	1514.000	1.672	0.000	3022	2863.000	1000.000	0.019	4202	3691.000	1000.000	0.038
0023	918.000	0.578	0.000	1203	1923.000	1.368	0.000	3023	3130.000	1000.000	0.086	4203	3496.000	1000.000	0.189
0024	991.000	0.598	0.000	1204	1421.000	1.072	0.000	3024	2645.000	1000.000	0.050	4204	3383.000	1000.000	0.204
0030	1106.000	0.838	0.000	1210	1263.000	1.720	0.000	3030	2837.000	1000.000	0.115	4210	3351.000	1000.000	0.209
0031	1120.000	1.048	0.000	1211	1284.000	1.329	0.000	3031	2943.000	125.989	0.000	4211	3894.000	1000.000	0.003
0032	1001.000	1.167	0.000	1212	1818.000	1.550	0.000	3032	3153.000	686.596	0.000	4212	4016.000	1000.000	0.008
0033	1044.000	6.830	0.000	1213	1480.000	2.185	0.000	3033	2731.000	48.489	0.000	4213	3855.000	1000.000	0.000
0034	762.000	0.915	0.000	1214	1467.000	2.353	0.000	3034	2762.000	78.677	0.000	4214	4014.000	1000.000	0.039
0100	603.000	1.008	0.000	1220	1679.000	1.444	0.000	3100	3191.000	1000.000	0.005	4220	3857.000	1000.000	0.104
0101	1204.000	0.444	0.000	1221	1612.000	1.352	0.000	3101	2774.000	19.212	0.000	4221	2863.000	1000.000	0.448
0102	801.000	0.793	0.000	1222	1723.000	1.307	0.000	3102	2851.000	32.964	0.000	4222	2892.000	1000.000	0.326
0103	712.000	0.716	0.000	1223	1361.000	1.176	0.000	3103	3384.000	1000.000	0.014	4223	3048.000	1000.000	0.324
0104	937.000	0.454	0.000	1224	1348.000	1.361	0.000	3104	3198.000	295.836	0.000	4224	2159.000	1000.000	1.027
0110	992.000	0.545	0.000	1230	1600.000	1.961	0.000	3110	2927.000	98.935	0.000	4230	2370.000	1000.000	0.735
0111	890.000	0.764	0.000	1231	1438.000	2.752	0.000	3111	2694.000	10.902	0.000	4231	4176.000	710.807	0.000
0112	891.000	0.505	0.000	1232	1689.000	1.457	0.000	3112	3142.000	1000.000	0.004	4232	2738.000	1000.000	0.304
0113	1190.000	0.679	0.000	1233	1573.000	1.178	0.000	3113	3322.000	31.425	0.000	4233	3916.000	1000.000	0.082
0114	868.000	0.540	0.000	1234	1361.000	4.697	0.000	3114	3266.000	30.179	0.000	4234	2766.000	1000.000	0.543
0120	885.000	1.002	0.000	2000	2244.000	17.318	0.000	3120	2664.000	1000.000	0.071	5000	2532.000	1000.000	0.982
0121	822.000	1000.000	0.052	2001	2129.000	46.473	0.000	3121	2892.000	92.127	0.000	5001	3420.000	1000.000	0.450
0122	902.000	0.293	0.000	2002	2109.000	221.200	0.000	3122	2975.000	1000.000	0.020	5002	1989.000	1000.000	1.502
0123	934.000	0.390	0.000	2003	2003.000	50.019	0.000	3123	2513.000	1000.000	0.063	5003	3006.000	1000.000	0.638
0124	1192.000	0.240	0.000	2004	2330.000	10.080	0.000	3124	2964.000	86.097	0.000	5004	3140.000	1000.000	0.544
0130	822.000	0.319	0.000	2010	1920.000	3.303	0.000	3130	2936.000	13.871	0.000	5010	4019.000	1000.000	0.257
0131	857.000	0.345	0.000	2011	2058.000	25.971	0.000	3131	3294.000	73.118	0.000	5011	3717.000	1000.000	0.392
0132	1184.000	0.401	0.000	2012	2077.000	4.927	0.000	3132	2817.000	11.553	0.000	5012	4085.000	1000.000	0.181
0133	988.000	0.298	0.000	2013	2028.000	12.719	0.000	3133	2780.000	8.623	0.000	5013	3912.000	1000.000	0.243
0134	987.000	1000.000	0.018	2014	2066.000	10.579	0.000	3134	3275.000	1000.000	0.002	5014	3435.000	1000.000	0.525
0200	964.000	0.439	0.000	2020	2025.000	1000.000	0.010	3200	3058.000	186.082	0.000	5020	3446.000	1000.000	0.466
0201	975.000	1.205	0.000	2021	1849.000	64.777	0.000	3201	2894.000	1000.000	0.015	5021	2848.000	1000.000	0.906
0202	1186.000	0.511	0.000	2022	1607.000	1000.000	0.159	3202	2791.000	1000.000	0.010	5022	2625.000	1000.000	1.165
0203	729.000	0.439	0.000	2023	1914.000	1000.000	0.018	3203	3164.000	31.880	0.000	5023	2339.000	1000.000	0.842
0204	995.000	0.381	0.000	2024	1990.000	13.282	0.000	3204	3232.000	1000.000	0.010	5024	3052.000	1000.000	0.650
0210	736.000	0.270	0.000	2030	1977.000	5.391	0.000	3210	2936.000	1000.000	0.043	5030	4500.000	1000.000	0.158
0211	1033.000	0.421	0.000	2031	2161.000	2.513	0.000	3211	2733.000	432.558	0.000	5031	3674.000	1000.000	0.390
0212	950.000	1.721	0.000	2032	1929.000	4.716	0.000	3212	3190.000	76.948	0.000	5032	2731.000	1000.000	0.841
0213	1176.000	0.304	0.000	2033	2229.000	2.292	0.000	3213	3113.000	16.805	0.000	5033	4409.000	1000.000	0.240
0214	809.000	0.624	0.000	2034	1698.000	1000.000	0.084	3214	2888.000	363.144	0.000	5034	3124.000	1000.000	0.709
0220	1066.000	0.284	0.000	2100	1892.000	12.877	0.000	3220	2922.000	131.058	0.000	5100	3002.000	1000.000	0.647
0221	1072.000	0.425	0.000	2101	2299.000	20.671	0.000	3221	2519.000	29.605	0.000	5101	4260.000	1000.000	0.127
0222	852.000	0.261	0.000	2102	1795.000	6.943	0.000	3222	2961.000	1000.000	0.005	5102	3089.000	1000.000	0.539
0223	974.000	0.427	0.000	2103	1874.000	32.792	0.000	3223	2802.000	84.433	0.000	5103	3053.000	1000.000	0.585
0224	926.000	0.331	0.000	2104	2164.000	2.932	0.000	3224	2912.000	1000.000	0.022	5104	3024.000	1000.000	0.805
0230	1098.000	0.512	0.000	2110	2236.000	2.658	0.000	3230	2810.000	99.628	0.000	5110	4858.000	1000.000	0.129
0231	1109.000	3.529	0.000	2111	2008.000	62.137	0.000	3231	2906.000	48.483	0.000	5111	5411.000	1000.000	0.001
0232	1057.000	1000.000	0.028	2112	1884.000	2.342	0.000	3232	3066.000	12.228	0.000	5112	3318.000	1000.000	0.298
0233	789.000	0.275	0.000	2113	2237.000	2.179	0.000	3233	2905.000	35.302	0.000	5113	3323.000	1000.000	0.328
0234	1250.000	0.473	0.000	2114	2205.000	3.879	0.000	3234	3310.000	13.114	0.000	5114	4701.000	1000.000	0.114
1000	1398.000	1000.000	0.149	2120	2239.000	9.136	0.000	4000	2883.000	1000.000	0.262	5120	3516.000	1000.000	0.424
1001	1555.000	0.867	0.000	2121	1992.000	1000.000	0.014	4001	2723.000	1000.000	0.527	5121	2342.000	1000.000	1.277
1002	1652.000	3.216	0.000	2122	2173.000	2.585	0.000	4002	3325.000	1000.000	0.168	5122	2795.000	1000.000	0.833
1003	1430.000	1.534	0.000	2123	2202.000	10.555	0.000	4003	3226.000	1000.000	0.283	5123	2510.000	1000.000	0.880
1004	1596.000	1000.000	0.019	2124	2054.000	5.803	0.000	4004	3423.000	1000.000	0.320	5124	4098.000	1000.000	0.245
1010	1446.000	1.765	0.000	2130	1576.000	1000.000	0.020	4010	3433.000	1000.000	0.110	5130	3782.000	1000.000	0.265
1011	1413.000	0.715	0.000	2131	1743.000	1000.000	0.006	4011	3938.000	1000.000	0.021	5131	4554.000	1000.000	0.001
1012	1641.000	6.637	0.000	2132	1884.000	6.352	0.000	4012	3896.000	1000.000	0.074	5132	4561.000	1000.000	0.076
1013	1437.000	1.048	0.000	2133	1826.000	4.147	0.000	4013	4015.000	1000.000	0.061	5133	3811.000	1000.000	0.302
1014	1174.000	0.903	0.000	2134	2121.000	4.098	0.000	4014	3563.000	1000.000	0.089	5134	5062.000	1000.000	0.041
1020	1737.000	1000.000	0.077	2200	1540.000	11.548	0.000	4020	3616.000	1000.000	0.192	5200	-	1000.000	-
1021	1681.000	1.013	0.000	2201	1980.000	4.728	0.000	4021	2856.000	1000.000	0.323	5201	2465.000	1000.000	1.211
1022	1358.000	1000.000	0.234	2202	1656.000	1000.000	0.022	4022	2823.000	1000.000	0.352	5202	3668.000	1000.000	0.317
1023	1531.000	1000.000	0.011	2203	2179.000	1000.000	0.015	4023	3417.000	1000.000	0.179	5203	2850.000	1000.000	0.876
1024	1699.000	6.724	0.000	2204	2142.000	4.250	0.000	4024	2684.000	1000.000	0.397	5204	2793.000	1000.000	0.858
1030	1568.000	2.459	0.000	2210	2163.000	3.049	0.000	4030	3305.000	1000.000	0.133	5210	3517.000	1000.000	0.413
1031	1447.000	1000.000	0.024	2211	2047.000	7.739	0.000	4031	3541.000	1000.000	0.058	5211	4632.000	1000.000	0.150
1032	1217.000	1000.000	0.051	2212	2150.000	3.341	0.000	4032	3461.000	1000.000	0.223	5212	4689.000	1000.000	0.096
1033	1719.000	0.820	0.000	2213	2029.000	17.682	0.000	4033	3957.000	1000.000	0.079	5213	4206.000	1000.000	0.233
1034	1545.000	0.930	0.000	2214	2165.000	4.946	0.000	4034	3933.000	1000.000	0.106	5214	3862.000	1000.000	0.261
1100	1296.000	1000.000	0.025	2220	2385.000	6.578	0.000	4100	2611.000	1000.000	0.277	5220	-	1000.000	-
1101	1311.000	3.873	0.000	2221	2096.000	33.835	0.000	4101	2763.000	1000.000	0.480	5221	3866.000	1000.000	0.191
1102	1308.000	2.964	0.000	2222	2216.000	2.712	0.000	4102	2996.000	1000.000	0.274	5222	3002.000	1000.000	0.461
1103	1532.000	3.221	0.000	2223	1767.000	5.976	0.028	4103	3026.000	1000.000	0.168	5223	2898.000	1000.000	0.677
1104	1760.000	1.148	0.000	2224	1828.000	4.341	0.000	4104	3151.000	1000.000	0.314	5224	2682.000	1000.000	0.864
1110	1517.000	1.538	0.000	2230	1901.000	5.553	0.000	4110	3980.000	1000.000	0.033	5230	-	1000.000	-
1111	1486.000	1.056	0.000	2231	1921.000	2.697	0.000	4111	4261.000	749.632	0.000	5231	5280.000	1000.000	0.029
1112	1205.000	1000.000	0.093	2232	2059.000	3.153	0.000	4112	3466.000	1000.000	0.035	5232	-	1000.000	-
1113	1431.000	1.566	0.000	2233	1777.000	2.986	0.000	4113	4533.000	282.155	0.000	5233	2895.000	1000.000	0.859
1114	1408.000	1.726	0.000	2234	1983.000	9.585	0.000	4114	3670.000	1000.000	0.074	5234	3254.000	1000.000	0.407

4.3.1.1 Validação do *Upper Bound*

Para validar a heurística de construção do upper-bound, como medida de qualidade de soluções heurísticas, fez-se a relação entre os valores das soluções encontradas para cada instância pelo modelo relaxado e pela heurística de construção do UB (definido na seção 3.4). Essas relações são chamadas de GAP, conforme definidos a seguir.

Para o cálculo do $GAP_{ModeloRelaxado}$, considerou-se o modelo e sua relaxação, comparando-se o valor obtido para a função objetivo (FO) em cada caso. A Equação 4.3 mostra como o $GAP_{ModeloRelaxado}$ é calculado.

$$GAP_{ModeloRelaxado} = \frac{FO_Modelo_Relaxado - FO_Modelo}{FO_Modelo_Relaxado} \tag{4.3}$$

Para o cálculo do GAP_{UB} comparou-se o valor da função objetivo encontrada pela heurística relaxada utilizada para a obtenção do UB com a função objetivo (FO) obtida pelo modelo. A Equação (4.4) foi utilizada para o cálculo do GAP_{UB}

$$GAP_{UB} = \frac{FO_UB - FO_Modelo}{FO_UB} \tag{4.4}$$

A Figura 4.7 mostra as curvas de GAP para o Upper Bound (GAP_{UB}) e modelo relaxado ($GAP_{ModeloRelaxado}$). O fato de as curvas estarem praticamente sobrepostas comprova a eficácia da heurística relaxada na obtenção de *Upper Bound* para o problema. Quanto ao tempo, pode-se perceber pela Figura 4.8 que o tempo para se alcançar um valor de FO na heurística relaxada é consideravelmente menor em relação ao tempo necessário para que o modelo relaxado encontre um valor de FO, comprovando a eficiência da heurística em relação a relaxação do modelo.

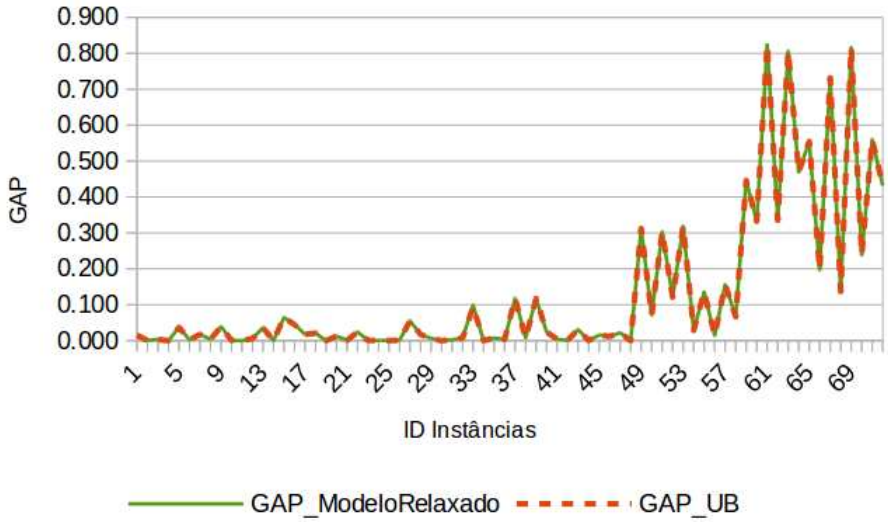

Figura 4.7: Comparação Entre o GAP médio apresentado pelo Modelo Relaxado e o apresentado pela heurística de UB

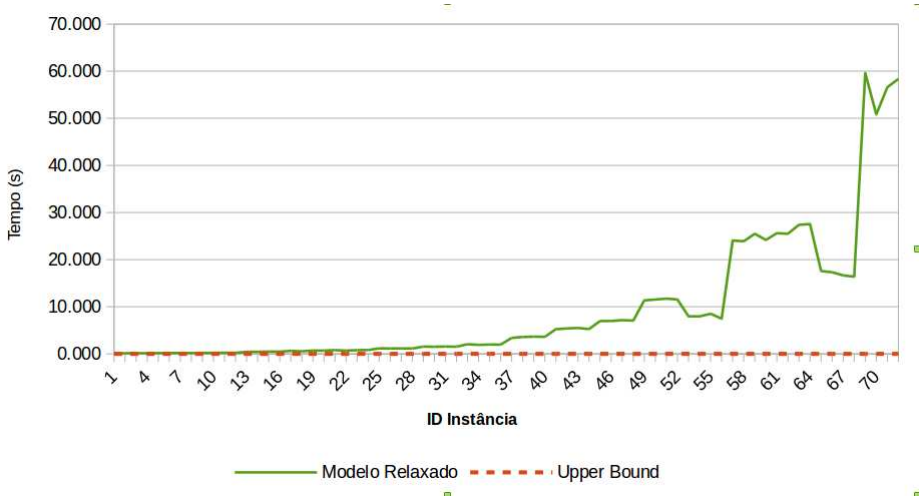

Figura 4.8: Comparação entre Tempo Médio de Execução da Relaxação do Problema com o Tempo Médio de Execução do Modelo Relaxado.

4.3.1.2 Resultados das Heurísticas implemtadas

Para a avaliação das soluções obtidas pelas heurísticas implementadas (SN, $FISN$, $FICN$, CN) fez-se primeiramente a identificação daquela que seria a melhor solução dentre todas (M), para cada instância. Na sequência, fez-se a relação

da solução obtida por cada heurística (LB) com M ($GAP1$) e com o UB ($GAP2$). Sendo $GAP1$ e $GAP2$ calculados pelas Equações 4.5 e 4.6 respectivamente.

$$GAP1 = \frac{FO_M - FO_LB}{FO_M} \tag{4.5}$$

$$GAP2 = \frac{FO_UB - FO_LB}{FO_UB} \tag{4.6}$$

Ao analisar o GAP1 (Figura 4.9) e o GAP2 (Figura 4.10) para as heurísticas implementadas observa-se que as curvas se apresentam praticamente sobrepostas, o que nos mostra o quão próxima está a eficácia das heurísticas implementadas. Quando analisa-se o gráfico da Figura 4.11 observa-se que as heurísticas SN e $FISN$ apresentam menor tempo de execução, principalmente em se tratando de instâncias de médio porte.

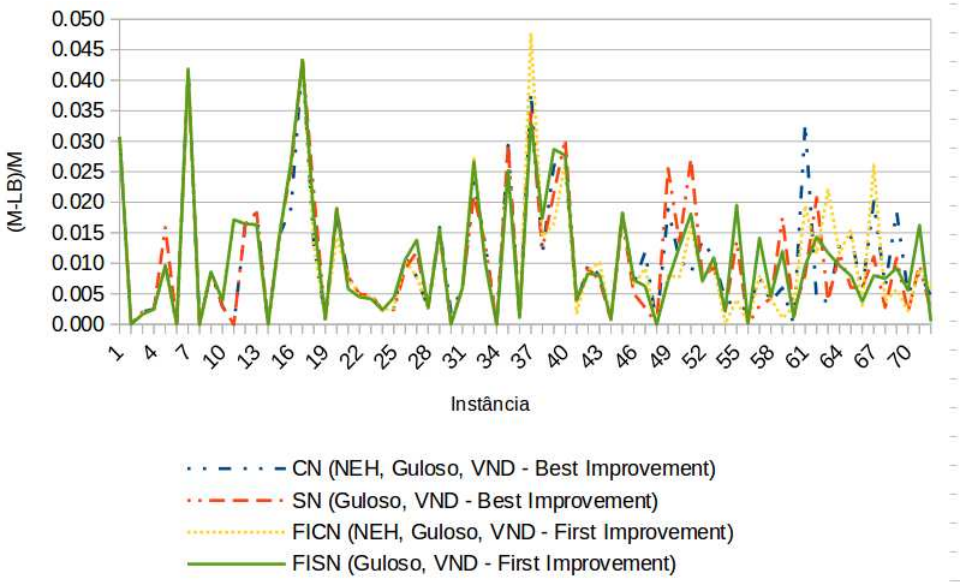

Figura 4.9: Comparação entre os Valores de GAP1 Médios Encontrados para as Heurísticas Implementadas

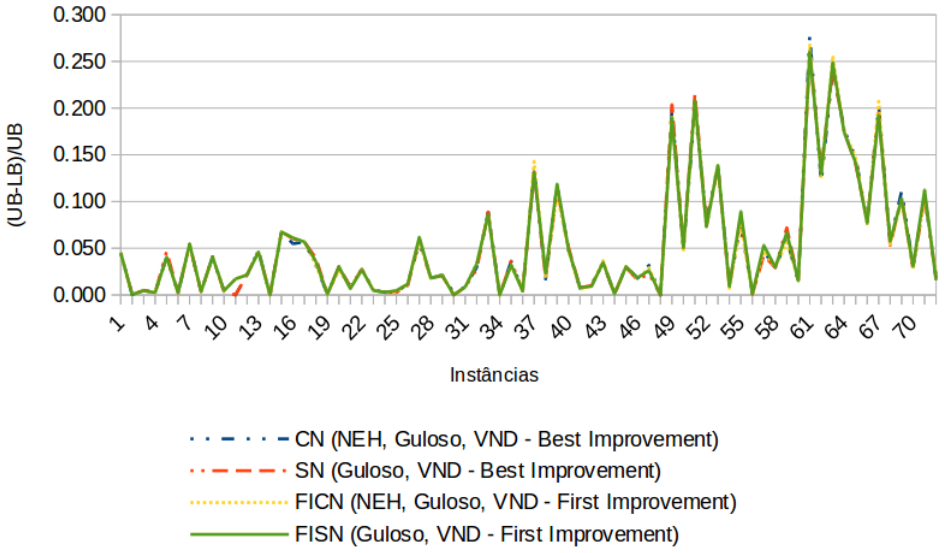

Figura 4.10: Comparação entre os Valores de GAP2 Médios Encontrados para as Heurísticas Implementadas

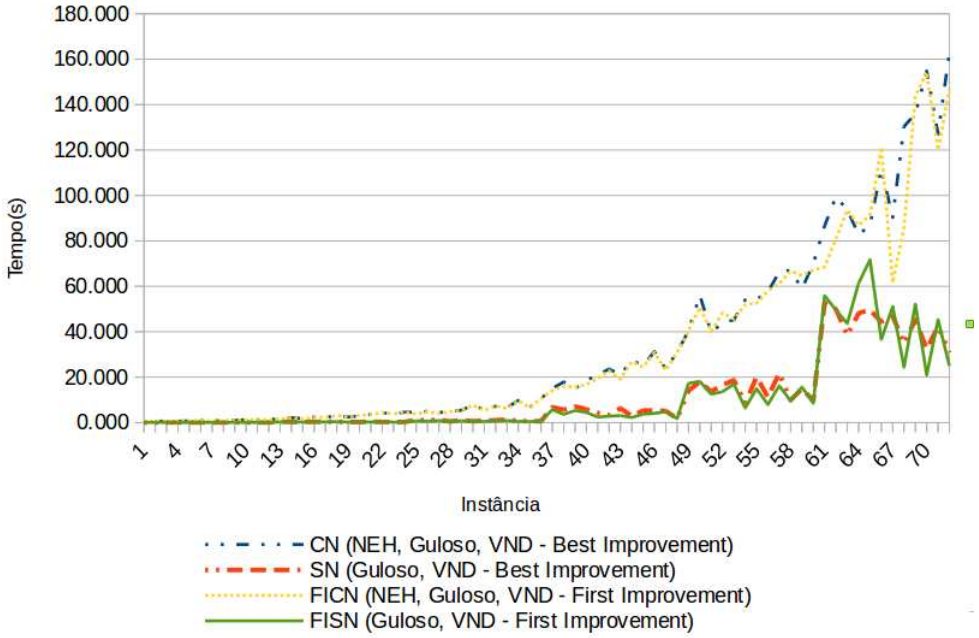

Figura 4.11: Comparação entre os Tempos Médios de Execução das Heurísticas Implementadas

Como ambas as heurísticas SN e FISN apresentaram resultados próximos e apresentam a mesma lógica de execução, exceto pela estratégia de busca pelas vizinhanças que é diferente para cada heurística, poderia-se escolher qualquer uma delas para a realização de um aprofundamento da análise da evolução dos resultados em cada passo do algoritmo. Neste trabalho optou-se por fazer um aprofundamento da análise dos resultados obtidos utilizando-se a heurística SN. Para simplificação da explicação optou-se por considerar a mesma dividida em duas partes: Heurística Construtiva (HC) e Heurística de Melhoria (HM). A HC é estruturada da seguinte forma: inicia-se com a construção de um limite superior (Upper Bound - UB) explicada na Seção 3.4; com a informação deste UB, uma solução viável é construída (Lower Bound - LB) utilizando-se a heurística definida na Seção 3.5.2. A heurística de melhoria no caso trata-se da aplicação do VND (Seção 3.5.3).

Para a avaliação das soluções obtidas pelas heurísticas implementadas, fez-se a relação da solução obtida por cada passo da heurística SN com o valor encontrado de UB. De forma que, o GAP_{HC} faz a relação entre valor da função objetivo (FO) obtido para o UB e o obtido conforme a HC (Equação (4.7)). O GAP_{HM} é obtido pela diferença entre o valor de FO do UB com o valor obtido de FO obtido pela HM (Equação (4.8)).

$$GAP_{HC} = \frac{FO_UB - FO_HC}{FO_UB} \tag{4.7}$$

$$GAP_{HM} = \frac{FO_UB - FO_HM}{FO_UB} \tag{4.8}$$

A Tabela 4.6 apresenta os resultados alcançados pelas estratégias de solução mencionadas anteriormente. Cada linha da tabela informa o resultado médio obtido para as 5 instâncias de cada conjunto base. As duas primeiras colunas (ID e *abc*) da tabela identificam os conjuntos de instâncias e as demais colunas informam o valor médio do GAP e do tempo médio obtidos através da execução da formulação de programação linear inteira mista (PLIM)[1], da heurística proposta para determinar um valor de limite superior (UB) para a solução, da heurística construtiva (HC) e da heurística de melhoria (HM).

[1]As colunas GAP-r e t-r(s) informam os resultados obtidos pela execução da PLIM relaxada.

Tabela 4.6: Tabela de Resultados da Heurística SN, passo a passo, para as instâncias da familia A

Instâncias		PLIM				UB		HC		HM	
ID	*abc*	GAP	t(s)	GAP-r	t-r(s)	GAP	t(s)	GAP	t(s)	GAP	t(s) HC+HM
1	000	0.000	1.856	0.016	0.094	0.015	0.001	0.144	0.001	0.045	0.111
2	001	0.000	1.052	0.000	0.125	0.000	0.001	0.024	0.001	0.000	0.065
3	002	0.000	0.635	0.004	0.133	0.003	0.001	0.038	0.001	0.005	0.087
4	003	0.000	2.160	0.000	0.137	0.000	0.001	0.009	0.001	0.002	0.088
5	010	0.000	0.683	0.037	0.166	0.037	0.001	0.105	0.001	0.047	0.081
6	011	0.000	0.606	0.002	0.161	0.002	0.001	0.008	0.001	0.002	0.071
7	012	0.010	200.492	0.018	0.187	0.018	0.001	0.069	0.001	0.054	0.083
8	013	0.004	200.443	0.004	0.173	0.004	0.001	0.010	0.001	0.004	0.072
9	020	0.000	0.595	0.039	0.186	0.038	0.001	0.104	0.001	0.041	0.102
10	021	0.000	0.668	0.001	0.211	0.001	0.001	0.025	0.001	0.004	0.084
11	022	0.000	0.346	0.000	0.227	0.000	0.001	0.046	0.001	0.000	0.079
12	023	0.006	201.068	0.006	0.218	0.006	0.001	0.039	0.001	0.022	0.071
13	100	0.034	402.074	0.036	0.434	0.036	0.001	0.104	0.001	0.048	0.438
14	101	0.000	2.214	0.000	0.430	0.000	0.001	0.024	0.002	0.000	0.300
15	102	0.064	602.042	0.064	0.473	0.064	0.001	0.083	0.002	0.068	0.324
16	103	0.015	401.095	0.046	0.474	0.044	0.001	0.101	0.002	0.060	0.344
17	110	0.005	202.460	0.017	0.632	0.017	0.001	0.082	0.002	0.057	0.375
18	111	0.019	201.418	0.021	0.545	0.021	0.001	0.066	0.001	0.039	0.348
19	112	0.000	1.790	0.000	0.696	0.000	0.001	0.008	0.002	0.001	0.249
20	113	0.012	201.447	0.013	0.683	0.013	0.001	0.044	0.002	0.030	0.274
21	120	0.000	1.801	0.001	0.781	0.001	0.001	0.029	0.001	0.009	0.367
22	121	0.000	1.827	0.024	0.680	0.024	0.001	0.113	0.001	0.028	0.309
23	122	0.000	1.328	0.001	0.789	0.001	0.001	0.019	0.002	0.005	0.235
24	123	0.000	2.409	0.001	0.812	0.001	0.001	0.027	0.002	0.003	0.294
25	200	0.000	69.018	0.000	1.162	0.000	0.001	0.087	0.002	0.003	1.238
26	201	0.000	11.500	0.001	1.137	0.001	0.001	0.066	0.002	0.010	0.939
27	202	0.037	617.134	0.055	1.154	0.055	0.001	0.102	0.002	0.060	1.249
28	203	0.017	204.015	0.017	1.131	0.017	0.001	0.023	0.002	0.018	0.504
29	210	0.000	15.243	0.006	1.540	0.006	0.001	0.084	0.002	0.021	1.069
30	211	0.000	14.639	0.000	1.512	0.000	0.001	0.009	0.002	0.000	0.595
31	212	0.003	206.070	0.003	1.556	0.003	0.001	0.022	0.002	0.009	0.782
32	213	0.005	405.143	0.007	1.520	0.007	0.001	0.080	0.002	0.028	1.053
33	220	0.007	404.472	0.099	2.037	0.094	0.001	0.162	0.002	0.089	1.252
34	221	0.000	7.351	0.000	1.908	0.000	0.001	0.001	0.002	0.000	0.696
35	222	0.006	10.688	0.007	1.978	0.007	0.001	0.046	0.002	0.036	0.589
36	223	0.000	4.795	0.003	1.961	0.003	0.001	0.013	0.002	0.004	0.620
37	300	0.117	1000.000	0.118	3.385	0.117	0.002	0.287	0.003	0.132	6.802
38	301	0.007	821.560	0.008	3.577	0.007	0.002	0.091	0.003	0.018	5.666
39	302	0.096	1000.000	0.117	3.660	0.117	0.002	0.206	0.003	0.113	7.101
40	303	0.023	389.793	0.024	3.607	0.024	0.002	0.102	0.003	0.051	5.851
41	310	0.004	472.561	0.004	5.251	0.004	0.002	0.041	0.003	0.008	4.231
42	311	0.001	234.323	0.001	5.398	0.001	0.002	0.033	0.003	0.010	3.339
43	312	0.031	635.837	0.031	5.492	0.031	0.002	0.051	0.003	0.034	6.310
44	313	0.000	221.466	0.000	5.256	0.000	0.002	0.007	0.003	0.001	2.844
45	320	0.007	643.708	0.016	6.977	0.016	0.002	0.080	0.003	0.029	5.348
46	321	0.009	377.930	0.012	6.976	0.012	0.002	0.048	0.003	0.016	5.336
47	322	0.005	449.173	0.022	7.148	0.022	0.002	0.060	0.003	0.023	5.140
48	323	0.000	41.751	0.000	7.062	0.000	0.002	0.000	0.003	0.000	1.762
49	400	0.312	1000.000	0.312	11.353	0.312	0.002	0.309	0.004	0.204	14.003
50	401	0.071	1000.000	0.072	11.531	0.072	0.002	0.135	0.004	0.051	17.967
51	402	0.289	1000.000	0.303	11.730	0.302	0.002	0.298	0.004	0.215	13.659
52	403	0.120	1000.000	0.121	11.534	0.121	0.002	0.146	0.005	0.074	16.583
53	410	0.302	1000.000	0.318	7.969	0.318	0.002	0.216	0.004	0.138	18.658
54	411	0.029	806.528	0.029	7.944	0.029	0.002	0.028	0.005	0.009	8.566
55	412	0.135	1000.000	0.135	8.519	0.135	0.002	0.197	0.004	0.084	20.104
56	413	0.015	837.368	0.015	7.461	0.015	0.002	0.012	0.005	0.001	11.228
57	420	0.154	1000.000	0.156	24.056	0.156	0.002	0.121	0.004	0.042	21.773
58	421	0.052	1000.000	0.067	23.891	0.066	0.002	0.062	0.004	0.029	9.912
59	422	0.446	1000.000	0.446	25.496	0.446	0.002	0.124	0.005	0.072	14.937
60	423	0.333	946.087	0.333	24.168	0.333	0.002	0.041	0.005	0.016	9.712
61	500	0.823	1000.000	0.824	25.620	0.824	0.003	0.462	0.005	0.259	52.769
62	501	0.320	1000.000	0.335	25.490	0.335	0.003	0.295	0.006	0.135	49.933
63	502	0.806	1000.000	0.806	27.393	0.806	0.003	0.334	0.006	0.242	38.414
64	503	0.468	1000.000	0.469	27.546	0.469	0.003	0.283	0.006	0.176	48.209
65	510	0.541	1000.000	0.554	17.576	0.554	0.003	0.258	0.006	0.140	49.668
66	511	0.174	1000.000	0.197	17.324	0.197	0.003	0.170	0.006	0.079	44.332
67	512	0.732	1000.000	0.732	16.652	0.732	0.003	0.265	0.006	0.195	46.943
68	513	0.137	1000.000	0.137	16.385	0.137	0.003	0.112	0.006	0.053	34.852
69	520	0.816	1000.000	0.816	59.598	0.816	0.003	0.205	0.006	0.105	45.291
70	521	0.230	1000.000	0.239	50.825	0.239	0.003	0.086	0.006	0.027	32.385
71	522	0.548	1000.000	0.560	56.603	0.560	0.003	0.180	0.006	0.106	42.311
72	523	0.432	1000.000	0.432	58.365	0.431	0.003	0.057	0.006	0.017	30.837

Os resultados descritos na Tabela 4.6 foram utilizados na construção dos gráficos apresentados nas Figuras 4.12 e 4.13, com o objetivo de destacar alguns aspectos observados neste trabalho. A Figura 4.12 apresenta as curvas de GAP médio para o modelo (GAP_{Modelo}), para a HC (GAP_{HC}) e para HM (GAP_{HM}). Pode-se observar que até a instância 37, apesar das oscilações, as instâncias foram resolvidas pelo modelo obtendo soluções com GAP_{Modelo} próximo ou igual a 0 e em um tempo inferior a 600 segundos. A partir da instância 37, o modelo apresenta mais dificuldade de conseguir um valor de FO com GAP próximo de zero, necessitando de um tempo de execução próximo ao limite máximo de 1000 segundos. Então pode-se concluir que as instâncias tornam-se mais difíceis de resolver à medida que aumenta-se o número de máquinas, tarefas e veículos, além do tamanho do intervalo de onde são sorteados os momentos de entrega. Todas as análises foram realizadas baseadas nos valores médios de GAP e tempo das replicas geradas para cada uma das 72 combinações, o que explica o fato de algumas instâncias apresentarem um valor de FO com um GAP diferente de 0 em um tempo menor que o limite de execução do modelo. Dentro de um mesmo conjunto de instâncias, devido ao fato de alguns parâmetros serem sorteados aleatoriamente, encontram-se instâncias com complexidade diferentes que deslocam o valor da média para mais ou para menos.

Ao comparar-se os valores de GAP obtidos pelo modelo, pela HC e pela HM, observa-se que nos casos das instâncias mais fáceis os melhores resultados oscilam entre o modelo e a HM, sendo a HC a que apresenta os piores resultados em todos os casos. Quando analisadas as instâncias com o ID a partir de 45, observa-se que HM ganha na maioria dos casos do modelo. A partir da instância 57, a HM ganha em todos os casos do modelo.

Quanto à HC, apresenta valores de GAP maiores que o modelo até a conjunto base de instâncias 57. Mesmo assim o uso de HC utilizada conjuntamente com a HM apresentou-se bastante eficaz, visto que a heurística construtiva Heu3 é utilizada para gerar a solução inicial viável necessária para a heurística de melhoria e internamente na própria heurística de melhoria.

Analisando-se a Figura 4.13 observa-se que o tempo de execução de HC mais HM apresenta menores valores em relação a execução do modelo. Desta forma, apesar de HC perder, em relação ao GAP, para o modelo, pode-se considerá-la

eficiente como solução inicial para a heurística de melhoria, visto que o conjunto consegue alcançar resultados próximo ou melhor que o modelo com um tempo de execução bem mais baixo. Além disso o tempo de execução de HC representa 0,1% do tempo total do conjunto HC mais HM.

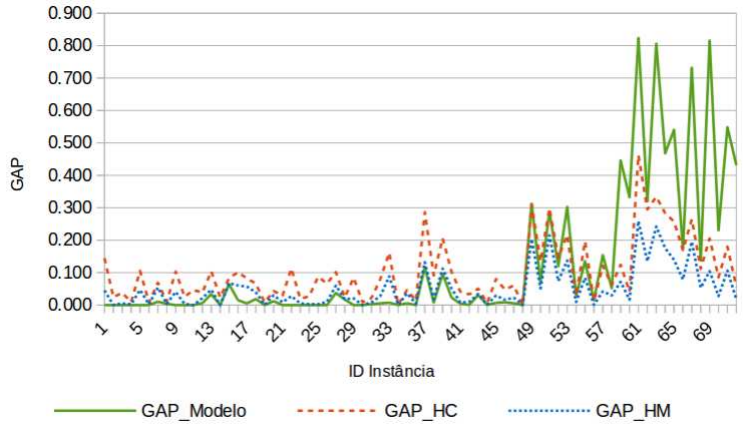

Figura 4.12: Comparação entre os GAPs Médios Calculados para HC, e HM e o Obtido pelo CPLEX para o Modelo

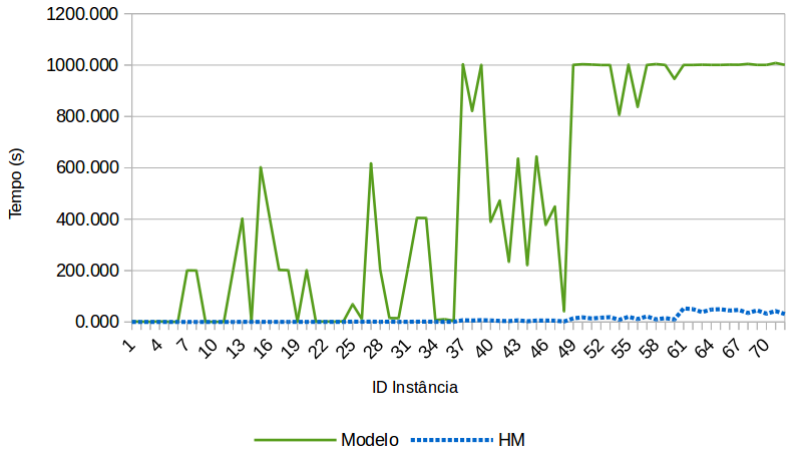

Figura 4.13: Comparação entre Tempo de Execução do Modelo com o Tempo de Execução de HM.

Após a identificação da melhor eficiência das heurísticas SN e FISN em relação às heurísticas CN, FICN, optou-se por implementar mais uma heurística para melhoria dos resultados. Neste trabalho optou-se por implementar o *Simulate Annealing* (SA).

Para avaliar a eficácia das heurísticas, visto que verificou-se que a análise visual não foi suficiente para garantir qual das heurísticas (FISN ou SN) seria a melhor heurística, decidiu-se utilizar o cálculo do RPD como critério de avaliação. Mas, no cálculo de RPD considerou-se também a heurística SA, CN e FICN, para garantir com mais segurança qual seria a heurística mais eficiente. Então calculou-se o valor de RPD para cada instância e heurística e fez-se a média desses valores de todas as instâncias para cada heurística. Na sequência, aplicou-se o teste de *Scott-Knott*.

O cálculo do RPD é dado pela Equação 4.2, sendo FO a solução da instância corrente para cada heurística, e $FO_melhorSolucao$ o melhor valor de FO obtido dentre todas as heurísticas para a instancia corrente em cada heurística. O teste de *Scott-Knott* (Scott & Knott [1974]) foi utilizado para determinar quais tratamentos apresentam diferenças significativa entre si.

A Figura 4.14 apresenta os resultado do teste de *Scott-Knott* para as instâncias das Família A em forma de tabela. A tabela apresenta quatro colunas, onde a primeira linha apresenta o ID da linha, a segunda coluna apresenta o identificador Grupos que indica se os tratamentos obtiveram resultados com diferenças significativa entre si (tratamentos pertencentes a um mesmo grupo não apresentaram diferenças significativas e pertencentes a grupos distintos apresentaram diferenças significativas), a terceira coluna apresenta o tratamento e a última coluna apresenta a média dos RPD's por heurística. O *boxplot* apresentado na Figura 4.15 apresenta a dispersão dos valores de RPD para cada heurística, considerando todas as instâncias. Observa-se que os valores de RPD estão mais concentrados abaixo do segundo quartil, isso representa que os valores estão menos dispersos para 50 porcento das instâncias e próximos de zero. O restante dos dados se encontram mais dispersos acima da média, havendo alguns poucos que se destacam por possuírem valores mais elevados de RPD, que os diferenciam dos resultados obtidos pela maioria das instâncias. As instancias mais dispersas se encontram representadas pelos círculos acima dos bigodes do boxplot, e são consideradas *outliers*.

Observa-se que a presença de valores dispersos se apresenta menos frequente na heurística SA, onde os resultados de RPD se encontram mais concentrados em torno de 0.

```
Teste de Scott-Knott
---------------------------------
  Grupos Tratamentos     Medias
1      a        FISN 0.81358333
2      a          CN 0.79224917
3      a          SN 0.79095889
4      a        FICN 0.78062139
5      b          SA 0.02336056
---------------------------------
```

Figura 4.14: Testes de Scott Knott - Heurísticas aplicadas ás instâncias da Família A

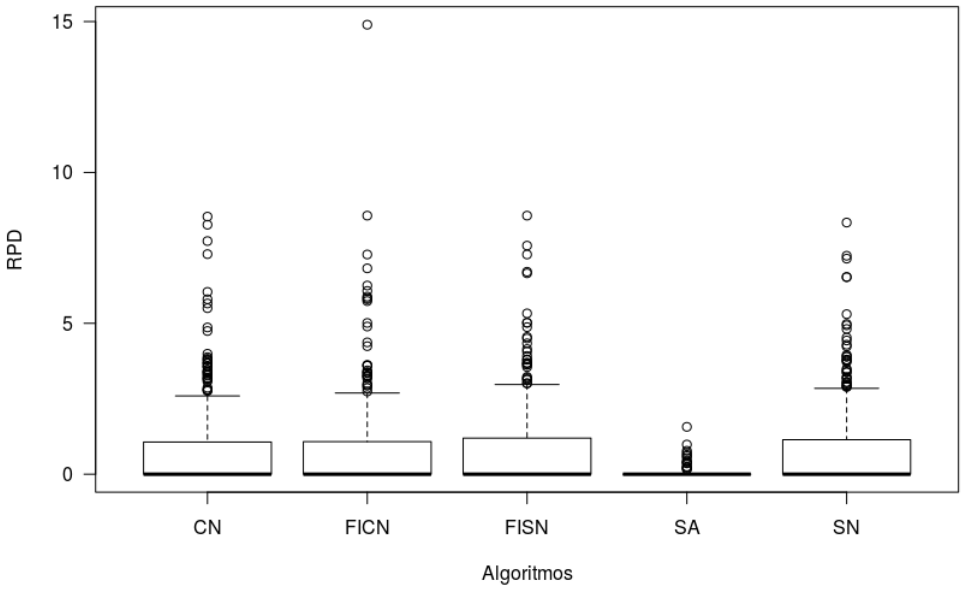

Figura 4.15: *Box-Plot* - Resultados das heurísticas implementadas considerando a Família A de Instâncias

Sendo assim, por meio das Figuras 4.14 e 4.15, percebe-se que a heurística SA apresentou os melhores resultados. As demais heurísticas (SN, CN, FISN e FICN) não apresentaram diferenças significativas de eficácia, sendo o maior diferencial entre elas o tempo de execução. Desta forma considerando-se o tempo de

execução como critério, observa-se através da Figura 4.11 que dentre as heurísticas SN, CN, FISN e FICN as heurísticas SN e FISN se apresentaram mais eficientes. Analisando-se a Figura 4.14 como critério de desempate entre as Heurísticas SN e FISN, observa-se que a Heurística SN apresenta os melhores resultados. O que nos permite identificar as Heurísticas SA e SN como aquelas que apresentaram os melhores resultados, em ordem decrescente de eficiência.

O gráfico 4.16 apresenta os valores médios de tempo de execução das heurísticas SN e SA, que apresentaram os melhores resultados.

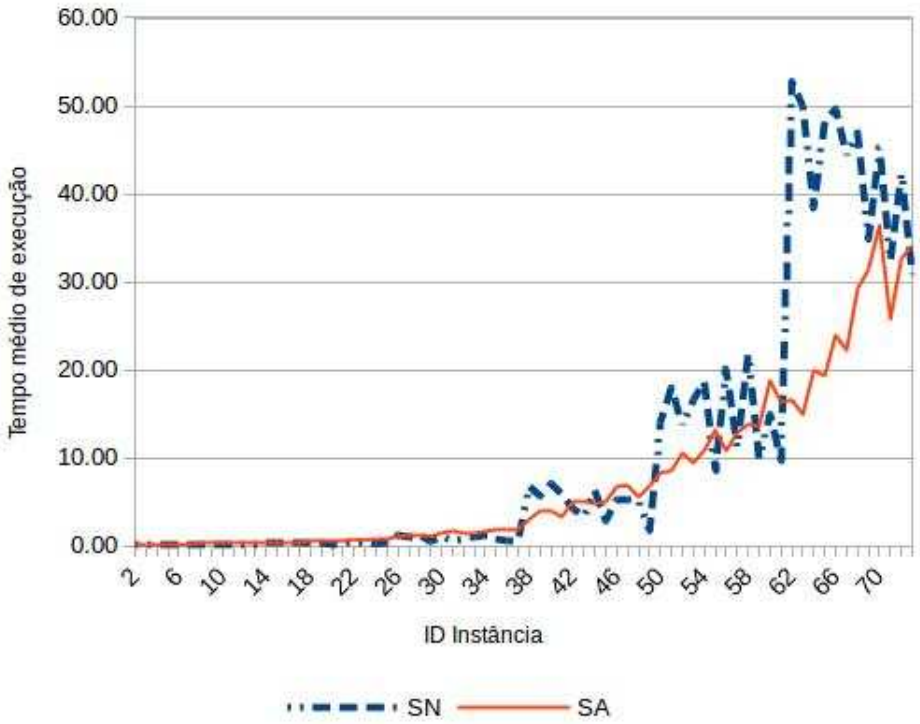

Figura 4.16: Comparação entre Tempo de Execução do SN com o Tempo de Execução do SA - Família A de Instâncias

Ao analisar o Gráfico 4.16 observa-se que, na maioria dos conjuntos de instâncias, o tempo de execução de SA é inferior ao tempo de execução de SN. Então pode-se concluir que a heurística SA é mais eficiente do que a heurística SN para a maioria das instâncias testadas.

4.3.2 Resultados Experimentais Tendo como Entrada as Instâncias da Família B

Dado o tamanho pequeno e médio das instâncias da família A, optou-se por gerar instâncias maiores (Família B de instâncias) para evidenciar a diferença de qualidade das soluções encontradas pelas heurísticas identificadas como as mais eficientes (SA e SN) na análise descrita na Seção 4.3.1. Para tal, calculou-se o RPD para cada instância e heurística e analisou-se a média dos valores de RPD de todas instâncias para cada heurística. Na sequência, aplicou-se o teste de *Scott-Knott*, da mesma maneira que na seção 4.3.1

```
Teste de Scott-Knott
-----------------------------
  Grupos Tratamentos  Medias
1      a          SN 2.86574
2      b          SA 0.27793
-----------------------------
```

Figura 4.17: Testes de Scott Knott - Heurísticas aplicadas as instâncias da Família B

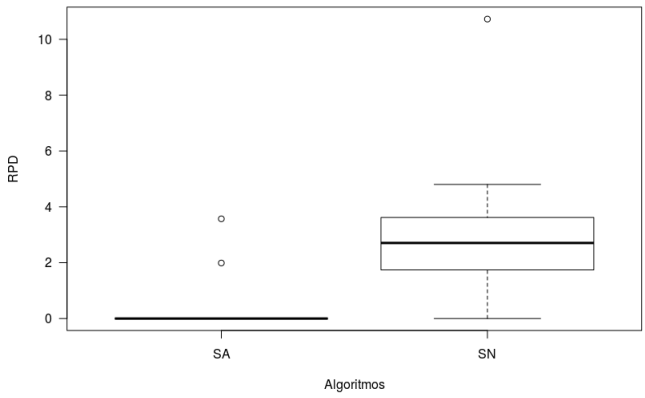

Figura 4.18: *Box-Plot* - Resultados da aplicação das heurísticas implementadas considerando a família B de instâncias

Ao analisar os resultados do teste de *Scott-Knott*, apresentados na Figura 4.17 observa-se na coluna grupos que cada heurística foi classificada em um grupo diferente, esse fato mostra haver uma diferença significativa entre elas. Evedenci-

ada pelos valores de média na terceira coluna onde a heurística SA apresenta um valor bem inferior ao obtido pela heurística SN.

O *boxplot* apresentado na Figura 4.18 apresenta a dispersão dos valores de RPD para cada heurística, considerando as instâncias da família B. Observa-se que no caso da heurística SA, os valores de RPD estão mais concentrados próximos de 0, de forma que torna-se difícil identificar os quartis, havendo apenas dois *outliers*. No caso da Heurística SN, os dados se apresentam mais dispersos, sendo possível observar certa homogeneidade dos dados em torno da mediana que apresenta um valor de RPD entre 2 e 4. No caso da SN, apesar de haver apenas um *outlier* e da homogeneidade dos dados, os mesmos apresentam a grande maioria dos valores de RPD distantes de 0 em relação a heurística SA.

Então, sendo assim, por meio das Figuras 4.17 e 4.18, percebe-se que a heurística SA apresentou os melhores resultados, confirmando-se as primeiras análises.

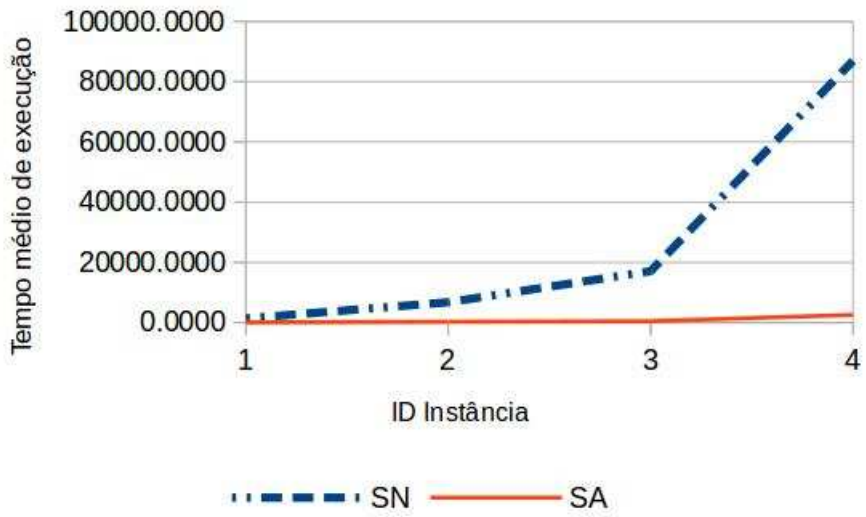

Figura 4.19: Comparação entre Tempo de Execução do SN com o Tempo de Execução do SA. - Família B de Instâncias

Ao analisar o Gráfico 4.19 observa-se que a curva de tempo de execução da heurística SA encontra-se bem abaixo da curva de tempo de execução da heurística SN, Dessa maneira pode-se concluir que SA é mais eficiente do que SN.

As Tabelas 4.7, 4.8 e 4.9 apresentam os resultados das heurísticas mais eficientes dentre as aplicadas para este trabalho, no caso o SA e o SN. As tabelas 4.7

e 4.8 apresentam os resultados obtidos para as instâncias da família A; neste caso, pode-se comparar com o modelo, pois na maioria dos casos o modelo chegou em uma solução viável em tempo hábil. No caso das instancias da família B não foi possível obter solução viável por meio do modelo, por esse motivo apresenta-se na Tabela 4.9 apenas os valores de FO obtidos através das Heurísticas SN e SA.

Tabela 4.7: Valores de FO obtidos pelas heurísticas SN, SA e o Modelo para as instâncias da família A

Inst (abcd)	Modelo		SN		SA	
	FO	Tempo	FO	Tempo	FO	Tempo
0000	879.000	3.893	862	0.14576	862	0.17325
0001	1017.000	0.637	941	0.10046	958	0.16087
0002	1140.000	3.789	1072	0.15536	1072	0.16515
0003	771.000	0.684	771	0.0982	771	0.16995
0004	992.000	0.278	992	0.05617	992	0.18802
0010	1025.000	3.994	1025	0.06387	1025	0.20318
0011	1032.000	0.372	1032	0.05796	1032	0.19227
0012	1057.000	0.166	1057	0.05576	1057	0.18846
0013	1292.000	0.193	1292	0.0875	1292	0.16383
0014	973.000	0.534	973	0.05842	973	0.19468
0020	956.000	0.335	956	0.06365	956	0.21567
0021	1171.000	0.536	1171	0.06111	1171	0.20375
0022	975.000	1.125	965	0.17901	975	0.16169
0023	918.000	0.578	918	0.06641	918	0.21682
0024	991.000	0.598	991	0.06284	991	0.21524
0030	1106.000	0.838	1106	0.05833	1106	0.19649
0031	1120.000	1.048	1120	0.06346	1120	0.2124
0032	1001.000	1.167	1001	0.1158	1001	0.1915
0033	1044.000	6.830	1031	0.13939	1031	0.19066
0034	762.000	0.915	762	0.06513	762	0.21313
0100	603.000	1.008	592	0.08672	592	0.23654
0101	1204.000	0.444	1204	0.06078	1204	0.27199
0102	801.000	0.793	752	0.14076	777	0.24121
0103	712.000	0.716	712	0.06146	712	0.28492
0104	937.000	0.454	937	0.05667	937	0.26681
0110	992.000	0.545	992	0.05861	992	0.26896
0111	890.000	0.764	890	0.05534	890	0.25487
0112	891.000	0.505	891	0.05422	891	0.25723
0113	1190.000	0.679	1190	0.05978	1190	0.28755
0114	868.000	0.540	868	0.12548	868	0.23652
0120	885.000	1.002	885	0.05897	885	0.26932
0121	822.000	1000.000	653	0.1744	653	0.23169
0122	902.000	0.293	902	0.05879	902	0.26136
0123	934.000	0.390	934	0.06172	934	0.28098
0124	1192.000	0.240	1192	0.06147	1192	0.28419
0130	822.000	0.319	822	0.06092	822	0.28216
0131	857.000	0.345	857	0.05462	857	0.25881
0132	1184.000	0.401	1184	0.05861	1184	0.2701
0133	988.000	0.298	988	0.06442	988	0.29131
0134	987.000	1000.000	987	0.11961	987	0.29631
0200	964.000	0.439	964	0.05833	964	0.32829
0201	975.000	1.205	975	0.11718	975	0.28275
0202	1186.000	0.511	1153	0.12343	1153	0.29808
0203	729.000	0.439	718	0.14708	729	0.27538
0204	995.000	0.381	995	0.06246	995	0.37687
0210	736.000	0.270	736	0.06181	736	0.37534
0211	1033.000	0.421	1033	0.05897	1033	0.35814
0212	950.000	1.721	942	0.15027	942	0.33901
0213	1176.000	0.304	1176	0.05777	1176	0.35216
0214	809.000	0.624	804	0.09292	809	0.30402
0220	1066.000	0.284	1066	0.0637	1066	0.35366
0221	1072.000	0.425	1072	0.06338	1072	0.34568
0222	852.000	0.261	852	0.13883	852	0.29014
0223	974.000	0.427	974	0.06451	974	0.35509
0224	926.000	0.331	926	0.0658	926	0.38787
0230	1098.000	0.512	1098	0.06224	1098	0.3813
0231	1109.000	3.529	1109	0.06808	1109	0.41036
0232	1057.000	1000.000	970	0.11157	970	0.35378
0233	789.000	0.275	789	0.05655	789	0.32742
0234	1250.000	0.473	1250	0.05685	1250	0.32226
1000	1398.000	1000.000	1297	0.80163	1311	0.30733
1001	1555.000	0.867	1555	0.17772	1555	0.40777
1002	1652.000	3.216	1652	0.27623	1652	0.37986
1003	1430.000	1.534	1430	0.38956	1430	0.35231
1004	1596.000	1000.000	1565	0.54254	1565	0.38907
1010	1446.000	1.765	1446	0.18316	1446	0.40443
1011	1413.000	0.715	1413	0.54852	1413	0.36154
1012	1641.000	6.637	1641	0.41068	1641	0.36424
1013	1437.000	1.048	1437	0.17405	1437	0.39708
1014	1174.000	0.903	1174	0.18313	1174	0.44257
1020	1737.000	1000.000	1690	0.41277	1690	0.38282
1021	1681.000	1.013	1681	0.19986	1681	0.45771
1022	1358.000	1000.000	1317	0.52877	1317	0.35398
1023	1531.000	1000.000	1506	0.18353	1516	0.39456
1024	1699.000	6.724	1699	0.29583	1699	0.40061
1030	1568.000	2.459	1568	0.19483	1568	0.42167
1031	1447.000	1000.000	1382	0.46176	1436	0.42339
1032	1217.000	1000.000	1117	0.68162	1195	0.3267
1033	1719.000	0.820	1719	0.19415	1719	0.43511
1034	1545.000	0.930	1545	0.18911	1545	0.42042
1100	1296.000	1000.000	1031	0.33469	1043	0.43875
1101	1311.000	3.873	1311	0.28172	1311	0.52169
1102	1308.000	2.964	1307	0.50685	1307	0.49265
1103	1532.000	3.221	1515	0.56677	1515	0.49765
1104	1760.000	1.148	1760	0.18286	1760	0.57985
1110	1517.000	1.538	1517	0.38444	1517	0.50227
1111	1486.000	1.056	1486	0.18409	1486	0.57146
1112	1205.000	1000.000	1100	0.29538	1151	0.58473
1113	1431.000	1.566	1402	0.39045	1399	0.47574
1114	1408.000	1.726	1408	0.4863	1408	0.51196
1120	1412.000	3.718	1406	0.31484	1412	0.57418
1121	1556.000	0.909	1556	0.16917	1556	0.5053
1122	1575.000	1.219	1575	0.20373	1575	0.61702
1123	1688.000	2.127	1688	0.36618	1688	0.55829
1124	1474.000	0.980	1474	0.18963	1474	0.59644
1130	1564.000	0.977	1564	0.18235	1564	0.55796

Inst (abcd)	Modelo		SN	SA	SA	
	FO	Tempo	FO	Tempo	FO	Tempo
3000	2303.000	1000.000	2217	6.93129	2390	2.95205
3001	2752.000	1000.000	2582	4.64274	2687	2.85585
3002	2867.000	1000.000	2799	7.82652	2781	3.22501
3003	2717.000	1000.000	2690	9.59384	2748	3.12592
3004	2584.000	1000.000	2507	5.01522	2527	2.89211
3010	2950.000	1000.000	2828	6.97119	2905	2.81206
3011	3005.000	1000.000	2964	5.21536	3005	3.25672
3012	3154.000	94.656	3154	3.14443	3154	4.74436
3013	3121.000	1000.000	3099	4.87033	3118	5.20053
3014	2799.000	1000.000	2820	8.12638	2800	3.98715
3020	2269.000	1000.000	2233	7.78524	2225	3.36977
3021	3263.000	1000.000	3154	5.74051	3205	4.84213
3022	2863.000	1000.000	2791	6.29051	2860	4.86903
3023	3130.000	1000.000	3040	8.54459	3129	3.05596
3024	2645.000	1000.000	2649	7.14375	2663	3.44247
3030	2837.000	1000.000	2725	4.92163	2744	3.32193
3031	2943.000	125.989	2835	7.54381	2860	2.96334
3032	3153.000	686.596	3143	3.54765	3148	3.90667
3033	2731.000	48.489	2683	3.91675	2719	3.18315
3034	2762.000	78.677	2611	9.32341	2633	3.13408
3100	3191.000	1000.000	3143	7.23683	3179	7.28873
3101	2774.000	19.212	2774	2.97962	2774	4.61069
3102	2851.000	32.964	2851	1.56567	2851	4.53273
3103	3384.000	1000.000	3370	3.6302	3390	4.57539
3104	3198.000	295.836	3198	5.74175	3198	4.21857
3110	2927.000	98.935	2850	6.53439	2857	4.17366
3111	2694.000	10.902	2694	1.65726	2694	4.82866
3112	3142.000	1000.000	3080	3.76182	3119	6.95538
3113	3322.000	31.425	3322	3.1844	3322	4.71326
3114	3266.000	30.179	3266	1.55754	3266	4.59939
3120	2664.000	1000.000	2605	4.96087	2606	4.44888
3121	2892.000	92.127	2883	6.5287	2886	4.75912
3122	2975.000	1000.000	2946	6.60056	2971	5.86223
3123	2513.000	1000.000	2537	8.01626	2540	4.22164
3124	2964.000	86.097	2964	5.44329	2964	4.66792
3130	2936.000	13.871	2936	1.66929	2936	4.86499
3131	3294.000	73.118	3294	4.16046	3294	5.15267
3132	2817.000	11.553	2817	1.80452	2817	5.42293
3133	2780.000	8.623	2780	1.71499	2780	4.96839
3134	3275.000	1000.000	3262	4.87178	3262	4.75693
3200	3058.000	186.082	3058	3.34403	3058	10.5949
3201	2894.000	1000.000	2928	4.07271	2936	6.76606
3202	2791.000	1000.000	2608	5.75473	2608	4.94233
3203	3164.000	31.880	3098	7.12299	3098	5.50969
3204	3232.000	1000.000	3232	6.44752	3232	5.92196
3210	2936.000	1000.000	2882	1.5134	2901	9.8372
3211	2733.000	432.558	2728	15.58396	2732	5.31778
3212	3190.000	76.948	3181	4.97001	3181	5.2057
3213	3113.000	16.805	3113	1.58557	3113	5.80901
3214	2888.000	363.144	2882	3.02855	2888	8.16235
3220	2922.000	131.058	2922	3.24673	2922	6.23581
3221	2519.000	29.605	2519	5.66224	2519	5.29391
3222	2961.000	1000.000	2948	6.28323	2948	5.11019
3223	2802.000	84.433	2776	4.37319	2796	5.10294
3224	2912.000	1000.000	2933	6.13685	2933	6.05915
3230	2810.000	99.628	2810	1.91054	2810	7.438
3231	2906.000	48.483	2906	1.51256	2906	5.83916
3232	3066.000	12.228	3066	1.86247	3066	6.95172
3233	2905.000	35.302	2905	1.7668	2905	6.90805
3234	3310.000	13.114	3310	1.7568	3310	6.88137
4000	2883.000	1000.000	3073	13.82534	3102	8.75595
4001	2723.000	1000.000	2803	14.12825	3058	5.93224
4002	3325.000	1000.000	3208	13.09287	3332	7.86656
4003	3226.000	1000.000	3488	10.15888	3553	7.35148
4004	3423.000	1000.000	3591	18.80912	3752	11.68066
4010	3433.000	1000.000	3604	16.64134	3682	8.17793
4011	3938.000	1000.000	3862	19.95368	3895	7.56855
4012	3896.000	1000.000	3954	24.67777	3892	7.50294
4013	4015.000	1000.000	4133	13.4	4196	8.47847
4014	3563.000	1000.000	3591	15.16371	3699	10.86921
4020	3616.000	1000.000	3794	14.41486	3798	10.62831
4021	2856.000	1000.000	2941	7.01234	3071	13.65492
4022	2823.000	1000.000	2966	18.9295	3086	9.98936
4023	3417.000	1000.000	3224	14.48605	3472	11.04079
4024	2684.000	1000.000	2725	13.4529	2815	7.47428
4030	3305.000	1000.000	3353	9.80915	3527	6.60036
4031	3541.000	1000.000	3558	22.75265	3606	6.56636
4032	3461.000	1000.000	3842	15.29177	3945	13.64621
4033	3957.000	1000.000	3988	16.75213	4063	11.82572
4034	3933.000	1000.000	4135	18.30707	4149	8.57103
4100	2611.000	1000.000	2972	17.53986	2961	10.85161
4101	2763.000	1000.000	3661	21.2053	3776	11.46465
4102	2996.000	1000.000	3426	32. 35751	3473	10.1718
4103	3026.000	1000.000	3016	10.96954	3056	11.17983
4104	3151.000	1000.000	3421	11.21837	3493	10.62456
4110	3980.000	1000.000	4024	7.18969	4048	11.73293
4111	4261.000	749.632	4261	7.67918	4261	12.25607
4112	3466.000	1000.000	3530	3.69702	3577	17.49978
4113	4533.000	282.155	4533	7.82014	4533	12.24033
4114	3670.000	1000.000	3910	16.44212	3910	12.03442
4120	3148.000	1000.000	3342	20.33818	3388	13.47762
4121	3427.000	1000.000	3572	30.22068	3656	9.37579
4122	3817.000	1000.000	3815	16.81487	3949	9.77827
4123	3770.000	1000.000	3825	15.10771	3926	12.21556
4124	3425.000	1000.000	3687	18.03812	3707	9.22561
4130	3922.000	1000.000	3951	4.06369	3951	15.04232

Tabela 4.8: Valores de FO obtidos pelas heurísticas SN, SA e o Modelo para as instâncias da família A - continuação da tabela 4.7

Inst (abcd)	Modelo		SN		SA	
	FO	Tempo	FO	Tempo	FO	Tempo
1131	1174.000	1.236	1126	0.65226	1126	0.49114
1132	1205.000	1000.000	1140	0.17977	1185	0.76413
1133	1521.000	1.132	1521	0.17197	1521	0.52585
1134	1619.000	1.092	1619	0.18168	1619	0.59674
1200	1407.000	1.538	1352	0.50763	1366	0.6476
1201	1499.000	3.352	1499	0.60254	1499	0.70139
1202	1514.000	1.672	1514	0.35903	1514	0.67959
1203	1923.000	1.368	1923	0.19747	1923	0.87365
1204	1421.000	1.072	1421	0.16872	1421	0.70906
1210	1263.000	1.720	1260	0.30008	1263	0.65353
1211	1284.000	1.329	1284	0.18627	1284	0.7285
1212	1818.000	1.550	1818	0.18209	1818	0.68598
1213	1480.000	2.185	1447	0.40111	1480	0.62827
1214	1467.000	2.353	1467	0.47308	1467	0.68624
1220	1679.000	1.444	1639	0.39822	1644	0.62636
1221	1612.000	1.352	1612	0.20793	1612	0.78058
1222	1723.000	1.307	1723	0.20411	1723	0.75073
1223	1361.000	1.176	1361	0.18124	1361	0.69522
1224	1348.000	1.361	1348	0.18138	1348	0.69578
1230	1600.000	1.961	1600	0.18739	1600	0.71346
1231	1438.000	2.752	1438	0.18744	1438	0.68158
1232	1689.000	1.457	1689	0.17902	1689	0.67954
1233	1573.000	1.178	1573	0.1817	1573	0.76384
1234	1361.000	4.697	1345	0.73583	1345	0.63273
2000	2244.000	17.318	2244	1.0338	2244	1.08811
2001	2129.000	46.473	2115	1.67504	2115	0.93986
2002	2109.000	221.200	2104	1.22175	2104	1.03534
2003	2003.000	50.019	1997	1.81835	1997	1.00612
2004	2330.000	10.080	2330	0.43958	2330	1.25373
2010	1920.000	3.303	1920	0.41756	1920	1.08554
2011	2058.000	25.971	2058	1.23309	2047	1.17462
2012	2077.000	4.927	2077	0.45012	2077	1.18195
2013	2028.000	12.719	1936	2.13533	1936	1.05064
2014	2066.000	10.579	2066	0.45721	2066	1.16616
2020	2025.000	1000.000	2015	1.13786	2015	0.99775
2021	1849.000	64.777	1833	2.33291	1815	0.97424
2022	1607.000	1000.000	1554	0.88162	1583	1.0387
2023	1914.000	1000.000	1906	0.95802	1906	1.50511
2024	1990.000	13.282	1973	0.93406	1973	1.39112
2030	1977.000	5.391	1977	0.47443	1977	1.15746
2031	2161.000	2.513	2161	0.45839	2161	1.11845
2032	1929.000	4.716	1929	0.42664	1929	1.03279
2033	2229.000	2.292	2229	0.46554	2229	1.14285
2034	1698.000	1000.000	1675	0.69341	1675	1.02087
2100	1892.000	12.877	1892	0.45232	1892	1.53144
2101	2299.000	20.671	2289	0.7876	2299	1.60694
2102	1795.000	6.943	1759	1.62553	1772	1.38308
2103	1874.000	32.792	1778	2.00808	1778	1.27419
2104	2164.000	2.932	2164	0.4727	2164	1.54688
2110	2236.000	2.658	2236	0.47326	2236	1.65203
2111	2008.000	62.137	2008	1.0947	2008	1.99875
2112	1884.000	2.342	1884	0.48075	1884	1.59322
2113	2237.000	2.179	2237	0.46044	2237	1.5433
2114	2205.000	3.879	2205	0.46508	2205	1.7984
2120	2239.000	9.136	2239	0.48543	2239	1.55462
2121	1992.000	1000.000	1933	1.98016	1933	1.34465
2122	2173.000	2.585	2173	0.49322	2173	1.60825
2123	2202.000	10.555	2202	0.46518	2202	1.50286
2124	2054.000	5.803	2054	0.48633	2054	1.60259
2130	1576.000	1000.000	1511	0.86204	1519	1.48962
2131	1743.000	1000.000	1741	0.86036	1741	1.53198
2132	1884.000	6.352	1797	1.37807	1822	1.24827
2133	1826.000	4.147	1795	1.70435	1798	1.43075
2134	2121.000	4.098	2121	0.45886	2121	1.67209
2200	1540.000	11.548	1536	1.69425	1540	1.43317
2201	1980.000	4.728	1980	0.46174	1980	1.91318
2202	1656.000	1000.000	1601	2.26101	1601	1.49777
2203	2179.000	1000.000	2114	1.39434	2151	1.65757
2204	2142.000	4.250	2142	0.44829	2142	1.84724
2210	2163.000	3.049	2163	0.47898	2163	1.94244
2211	2047.000	7.739	2047	0.45098	2047	1.88163
2212	2150.000	3.341	2150	0.43745	2150	1.85268
2213	2029.000	17.682	2029	1.67312	2029	1.8999
2214	2165.000	4.946	2165	0.43914	2165	1.81321
2220	2385.000	6.578	2385	0.49072	2385	1.989
2221	2096.000	33.835	1926	0.89236	1926	1.86421
2222	2216.000	2.712	2216	0.44072	2216	1.78805
2223	1767.000	5.976	1651	0.66082	1688	1.6178
2224	1828.000	4.341	1828	0.45833	1828	1.85624
2230	1901.000	5.553	1901	0.49087	1901	2.05383
2231	1921.000	2.697	1921	0.47559	1921	1.94782
2232	2059.000	3.153	2059	0.44373	2059	1.84166
2233	1777.000	2.986	1777	0.46776	1777	1.94323
2234	1983.000	9.585	1972	1.22209	1983	1.63241

Inst (abcd)	Modelo		SN	SA	SA	
	FO	Tempo	FO	Tempo	FO	Tempo
4131	4054.000	185.512	4054	4.51563	4054	12.68087
4132	3962.000	1000.000	4002	12.4122	4002	10.75634
4133	3729.000	1000.000	3851	25.43813	3851	13.03138
4134	3950.000	1000.000	4024	9.70989	4024	12.82818
4200	3513.000	1000.000	3920	19.16635	4001	12.55386
4201	3193.000	1000.000	3620	18.3224	3651	13.36422
4202	3691.000	1000.000	3811	21.58058	3816	12.13415
4203	3496.000	1000.000	3773	22.15061	3856	17.62642
4204	3383.000	1000.000	3947	27.64302	3961	13.42195
4210	3351.000	1000.000	3808	15.28568	3842	14.33996
4211	3894.000	1000.000	3905	3.98545	3905	14.4933
4212	4016.000	1000.000	4049	3.94727	4049	13.67012
4213	3855.000	1000.000	3783	22.13806	3794	11.07629
4214	4014.000	1000.000	4170	4.20393	4170	14.46093
4220	3857.000	1000.000	4126	13.08631	4199	18.45233
4221	2863.000	1000.000	3750	23.3137	3960	10.90546
4222	2892.000	1000.000	3628	4.53288	3771	28.17971
4223	3048.000	1000.000	3873	21.5278	3932	22.88321
4224	2159.000	1000.000	3770	12.22452	3893	13.44502
4230	2370.000	1000.000	4112	4.56057	4112	16.61122
4231	4176.000	710.807	4176	4.10628	4176	14.21294
4232	2738.000	1000.000	3419	4.82411	3419	16.17716
4233	3916.000	1000.000	4070	30.70679	4147	20.00952
4234	2766.000	1000.000	4267	4.36205	4267	14.68962
5000	2532.000	1000.000	3915	65.67242	4028	21.52047
5001	3420.000	1000.000	3812	43.75845	3913	23.18922
5002	1989.000	1000.000	3559	46.4305	3663	13.7405
5003	3006.000	1000.000	3653	40.82771	3714	11.72035
5004	3140.000	1000.000	3398	67.15554	3576	12.73615
5010	4019.000	1000.000	4032	44.33265	4314	12.96831
5011	3717.000	1000.000	4391	63.44593	4440	23.34184
5012	4085.000	1000.000	4481	34.27146	4643	12.21532
5013	3912.000	1000.000	4536	51.96075	4602	12.69421
5014	3435.000	1000.000	4582	55.65628	4598	13.51187
5020	3446.000	1000.000	4142	31.8518	4266	22.11277
5021	2848.000	1000.000	4018	31.91319	4022	21.33455
5022	2625.000	1000.000	3721	26.53152	3874	19.55207
5023	2339.000	1000.000	3294	66.40599	3330	22.29329
5024	3052.000	1000.000	4075	35.36665	4281	14.4195
5030	4500.000	1000.000	4434	63.10653	4569	18.07077
5031	3674.000	1000.000	3995	55.32583	4034	14.20554
5032	2731.000	1000.000	3867	45.98582	3982	23.33854
5033	4409.000	1000.000	4809	31.68689	4958	26.60773
5034	3124.000	1000.000	4508	44.9406	4528	14.4184
5100	3002.000	1000.000	4240	77.82443	4216	16.83769
5101	4260.000	1000.000	4445	51.02766	4556	30.23007
5102	3089.000	1000.000	4065	33.25018	4095	25.96686
5103	3053.000	1000.000	4357	59.20453	4385	17.77515
5104	3024.000	1000.000	4367	27.03177	4574	28.94716
5110	4858.000	1000.000	5196	57.53666	5265	19.92008
5111	5411.000	1000.000	5418	20.5996	5418	20.54125
5112	3318.000	1000.000	4010	42.29052	4078	18.06395
5113	3323.000	1000.000	3811	54.03042	3888	22.50508
5114	4701.000	1000.000	4917	47.20528	4985	30.18171
5120	3516.000	1000.000	4083	29.20417	4190	26.62043
5121	2342.000	1000.000	4399	24.77676	4556	31.93447
5122	2795.000	1000.000	4125	62.80811	4140	31.64819
5123	2510.000	1000.000	3606	52.2057	3644	29.58781
5124	4098.000	1000.000	4153	65.71958	4178	26.41991
5130	3782.000	1000.000	3896	35.3485	3919	23.57508
5131	4554.000	1000.000	4559	28.51754	4560	37.50799
5132	4561.000	1000.000	4812	52.64743	4875	33.10831
5133	3811.000	1000.000	4754	48.47396	4758	19.13335
5134	5062.000	1000.000	5172	9.27432	5241	43.71556
5200	-	1000.000	4103	61.23741	4160	34.39638
5201	2465.000	1000.000	4674	68.96257	4821	27.48342
5202	3668.000	1000.000	4268	41.66864	4321	39.51784
5203	2850.000	1000.000	4886	47.19565	4967	37.95889
5204	2793.000	1000.000	4700	7.39026	4837	42.72967
5210	3517.000	1000.000	4588	55.53225	4647	24.03651
5211	4632.000	1000.000	5324	23.91902	5324	25.54253
5212	4689.000	1000.000	5094	15.88183	5119	26.3461
5213	4206.000	1000.000	5184	17.6479	5184	28.82886
5214	3862.000	1000.000	4780	48.94584	4806	24.29377
5220	-	1000.000	4934	38.8921	4989	47.34795
5221	3866.000	1000.000	4216	61.33542	4298	35.6349
5222	3002.000	1000.000	3983	48.45984	4032	21.59718
5223	2898.000	1000.000	4387	37.92344	4540	24.99015
5224	2682.000	1000.000	4136	24.94515	4253	33.60175
5230	-	1000.000	4944	19.73179	4944	34.65054
5231	5280.000	1000.000	5433	7.99435	5433	25.82202
5232	-	1000.000	4743	41.00328	4797	29.66525
5233	2895.000	1000.000	5355	35.26498	5382	42.08406
5234	3254.000	1000.000	4329	50.19016	4386	38.51289

Tabela 4.9: Valores de FO obtidos pelas heurísticas SN e para as instâncias da família B

Inst (ad)	SN		SA	
00	6790	1281.84931	7022	75.02341
01	6542	830.10203	6872	74.44156
02	6243	1863.22001	6374	75.97363
03	5743	1661.59901	5897	75.27382
04	7018	1314.1958	7861	75.87652
10	9471	7452.37277	9646	339.62448
11	9854	7501.84808	10138	296.77857
12	9575	5899.25987	9957	288.17556
13	9631	6054.55737	10082	299.01856
14	9258	6792.24728	9415	304.93744
20	11247	19065.00606	11527	462.96623
21	11544	11275.42702	12011	454.87742
22	10712	24681.31334	10499	466.9802
23	10097	10047.34031	10165	531.16308
24	12044	20276.77234	11614	488.60489
30	16833	77871.07106	17342	1940.77836
31	15340	104695.35506	15814	3423.97096
32	16061	70422.92919	16413	1949.84283
33	16310	97733.94359	16884	3614.62923
34	14316	83664.08131	14426	1935.182

Capítulo 5

Considerações Finais

Este trabalho foi estudado o problema de sequenciamento de tarefas em máquinas paralelas idênticas de processamento em lotes com entrega, considerando os tempos de processamento distintos das tarefas. Para resolver o problema foram proposta uma formulação de programação linear inteira mista, uma relaxação do método heurístico citado na literatura para determinar um limite superior (UB) para a solução e duas heurísticas construtivas denominadas NEH e Heu3, para gerar uma solução viável baseada na solução de UB. Com o objetivo de melhorar a solução gerada pelas heurísticas construtivas, aplicou-se a heurística VND tendo como estratégias de busca nas estruturas de vizinhança o *best Improvement* e *first Improvement*, além da heurística SA.

Com este trabalho, foi possível validar o uso da heurística da literatura para definir o valor de UB para o problema estudado, comparando-se os tempos de execução e os $GAPs$ da heurística de geração do UB com os do modelo relaxado, e definir a heurística Heu3 como a heurística construtiva de maior eficiência. Foi possível, ainda, notar uma maior eficiência da heurística SA proposta para o conjunto de instâncias utilizado, visto que se conseguiu obter soluções boas em um tempo de execução reduzido, se comparado à formulação de programação linear inteira mista e as demais heurísticas propostas.

Como fruto deste trabalho, parte dos resultados alcançados foram apresentados e publicados no XLIX Simpósio Brasileiro de Pesquisa Operacional (SBPO) de 2017:

Título: Sequenciamento de Tarefas em Máquinas Paralelas de Processamento em Lotes com Entregas; **Autores**: Gilson Ataliba de Faria, José Elias Claudio Arroyo, André Gustavo dos Santos, Thiago Henrique Nogueira e Jonatas Batista Costa das Chagas.

Referências Bibliográficas

Agnetis, A.; Fu, L. & Aloulou, M. A. (2014). Coordination of production and interstage batch delivery with outsourced distribution. *European Journal of Operational Research*, 238(1):130–142.

Biyk, A.; Monch, L. & Almeder, C. (2014). Scheduling jobs with ready times and precedence constraints on parallel batch machines using metaheuristics. *Computers & Industrial Engineering, 78:175–185.*

Brucker, P. (2006). *Scheduling Algorithms.* Springer-Verlag Berlin Heidelberg, Alemanha, 5 edição.

Chang, Y.-C. & Lee, C.-Y. (2004). Machine scheduling with job delivery coordination. *European Journal of Operational Research*, 158(2):470--487.

Cheng, B.-Y.; Leung, J. Y.-T. & Li, K. (2015). Integrated scheduling of production and distribution to minimize total cost using an improved ant colony optimization method. *Computers & Industrial Engineering*, 83:217–225.

Correa, H. L.; Gianesi, I. G. N. & Caon, M. (1996). *Just In Time, Mrp Ii E Opt: Um enfoque esárinaségico. Páginas 56-104.* Editora Atlas S.A., São Paulo, 2 edição.

Correa, H. L.; Gianesi, I. G. N. & Caon, M. (2016). *Planejamento, Programação e Controle da Produção.* Editora Atlas S.A., São Paulo, 5 edição.

Damodaran, P. & Chang, P.-Y. (2008). Heuristics to minimize makespan of parallel batchprocessing machines. *International Journal of Advanced Manufacturing Technologies*, 37:1005–1013.

Damodaran, P. & Velez-Gallego, M. C. (2014). A simulated annealing algorithm to minimize makespan of parallel batch processing machines with unequal job ready times. *Expert Systems with Applications, 39(1):1451–1458.*

de Paula, M. R.; Ravetti, M. G.; Mateus, G. R. & Pardalos, P. M. (2007). Solving parallel machines scheduling problems with sequence-dependent setup times using variable neighbourhood search. *IMA Journal of Management Mathematics.*

Dowsland, K. A. (1993). *Simulated Annealing in C. R. Reeves editor: Modern Heuristic Techniques for Combinatorial Problems, Advanced Topics in Computer Science Series. Páginas 20-69.* Halsted Press, London.

Fidelis, M. B. & Arroyo, J. E. C. (2016). Heurística iterated greedy para o problema de sequenciamento de lotes de tarefas em máquinas paralelas. *Anais do XLVIII SBPO, Simpósio Brasileiro de Pesquisa Operacional, Vitória, ES, 27 a 30 de setembro de 2016, p. 1828-1839.*

Fidelis, M. B. & Arroyo, J. E. C. (2017). Heurísticas para o problema de sequenciamento de lotes de tarefas em máquinas paralelas. *XLIX Simpósio Brasileiro de Pesquisa Operacional, Blumenau-SC, 27 a 30 de Agosto de 2017.*

Fuchigami, H. Y. & Rangel, S. (2014). Uma análise de estudos de casos em sequenciamento da produção. *XLVI simpósio de brasileiro de pesquisa operacional, Salvador Bahia*, pp. 159 – 170.

Gendeau, M. & Potvin, J.-Y. (2010). *Handbook of Metaheuristics.* Springer, New York Dordrecht Heidelberg London, segunda edição.

Ikura, Y. & Gimple, M. (1986). Scheduling algorithms for a single batching processing machine. *Operations Research Letters*, 5:61–65.

Jia, Z.-H. & Leung, J. Y.-T. (2015). A meta-heuristic to minimize makespan for parallel batch machines with arbitrary job sizes. *European Journal of Operational Research*, 240(3):649–665.

Kashan, A. H.; Karimi, B. & Jenabi, M. (2008). A hybrid genetic heuristic for scheduling parallel batch processing machines with arbitrary job sizes. *Computers and Operations Research.*

Kim, D.-W.; Kim, K.-H.; a Jang, W. & Chen, F. F. (2002). Unrelated parallel machine scheduling with setup times using simulated annealing. *Robotics and Computer-Integrated Manufacturing Volume 18, Issues 3–4.*

Kirkpatrick, S.; Gellat, D. & Vecchi, M. (1983). Optimization by simulated annealing. *Science. 220:671–680.*

Laarhoven, P. J. M. V.; Aarts, E. H. L. & Lenstra, J. K. (1992). Job shop scheduling by simulate annealing. *Operations Research Vol.40, No. 1, Pages 113-125.*

Leung, J. Y.-T. & Chen, Z.-L. (2013). Integrated production and distribution with fixed delivery departure dates. *Operations Research Letters*, 41(3):290–293.

Li, K.; hong Jia, Z. & Leung, J. Y.-T. (2015). Integrated production and delivery on parallel batching machine. *European Journal of Operational Research*, 247(3):756–763.

Lo, Z.-P. & Bavarian, B. (1992). Optimization of job scheduling on parallel machines by simulated annealing algorithms. *Expert Systems with Applications Volume 4, Issue 3, 1992, Pages 323-328.*

Mathirajan, M. & Sivakumar, A. I. (2006). A literature review, classification and simple meta-analysis on scheduling of batch processors in semiconductor. *International Journal of Advanced Manufacturing Technology*, 29(9):990–1001.

Matias, M. A. & Nascimento, S. A. (2013). Just-in-time e seus efeitos sobre o desempenho financeiro das empresas: uma abordagem bibliográfica. *Revista de C. Humanas, Viçosa, v. 13, n. 1, p. 9-22, jan./jun. 2013.*

Mlaswnovic, N. & Hansen, P. (1997). Variable neighborhood search. *Computers & Operations Research*, 24(11):1097–1100.

Müller, F. M.; Dias, O. B. & de Araújo, O. C. B. (2002). Algoritmo para o problema de seqüenciamento em máquinas paralelas não-relacionadas. *Prod. vol.12 no.2 São Paulo 2002.*

Nawaz, M.; jr, E. E. & Ham, I. (1982). A heuristic algorithm for the m-manchine, n-job flow-shop sequencing problem. *Great Britain.*

Pinedo, M. L. (2005). *Planning and Scheduling in Manufacturing and Services.* Springer, Estados Unidos.

Scott, A. J. & Knott, M. (1974). A cluster analysis method for grouping means in the analyses of variance. *Biometrics, 30(3):507–512.*

Tubino, D. F. (2006). *Manual de Planejamento e Controle da Produção.* Editora Atlas S.A., São Paulo, 2 edição.