Modelos e algoritmos para variações do problema
de balanceamento de linhas de produção e
designação de trabalhadores

Felipe Francisco Bezerra Araújo

SERVIÇO DE PÓS-GRADUAÇÃO DO ICMC-USP

Data de Depósito:

Assinatura: _____

Felipe Francisco Bezerra Araújo

**Modelos e algoritmos para variações do problema de
balanceamento de linhas de produção e designação de
trabalhadores**

Tese apresentada ao Instituto de Ciências Matemáticas e de Computação - ICMC-USP, como parte dos requisitos para obtenção do título de Doutor em Ciências - Ciências de Computação e Matemática Computacional. *VERSÃO REVISADA*

Área de Concentração: Ciências de Computação e Matemática Computacional

Orientador: Prof. Dr. Alysson Machado Costa

**USP – São Carlos
Novembro de 2016**

Ficha catalográfica elaborada pela Biblioteca Prof. Achille Bassi
e Seção Técnica de Informática, ICMC/USP,
com os dados fornecidos pelo(a) autor(a)

A658m Araújo, Felipe Francisco Bezerra
Modelos e algoritmos para variações do problema
de balanceamento de linhas de produção e designação
de trabalhadores / Felipe Francisco Bezerra Araújo;
orientador Alysson Machado Costa. -- São Carlos,
2016.
90 p.

Tese (Doutorado - Programa de Pós-Graduação em
Ciências de Computação e Matemática Computacional) --
Instituto de Ciências Matemáticas e de Computação,
Universidade de São Paulo, 2016.

1. Linhas de produção. 2. Estações paralelas. 3.
Múltiplas linhas. 4. Rotação d. I. Costa, Alysson
Machado, orient. II. Título.

Felipe Francisco Bezerra Araújo

**Models and algorithms for variations of the assembly line
worker assignment and balancing problem**

Doctoral dissertation submitted to the Instituto de Ciências Matemáticas e de Computação - ICMC-USP, in partial fulfillment of the requirements for the degree of the Doctorate Program in Computer Science and Computational Mathematics. *FINAL VERSION*

Concentration Area: Computer Science and Computational Mathematics

Advisor: Prof. Dr. Alysson Machado Costa

**USP – São Carlos
November 2016**

Resumo

O problema de balanceamento de linhas de produção e designação de trabalhadores é uma extensão do problema simples de balanceamento de linhas onde os tempos de execução de tarefas são dependentes dos trabalhadores. Este problema tem sua motivação prática oriunda de linhas de produção com trabalhadores com deficiência. Nesta tese de doutorado estudamos duas extensões para o problema: a primeira permite *layouts* de linhas com estações em paralelo, enquanto que a segunda possibilita o uso de múltiplas linhas. As extensões foram aplicadas tanto ao problema básico quanto para o problema de rotação de tarefas. Apresentamos formulações matemáticas e métodos exatos e heurísticos para todos os casos. Testes computacionais em instâncias da literatura e novas instâncias e uma análise detalhada dos resultados são apresentados.

Palavras-chave: *Linhas de produção, estações paralelas, múltiplas linhas, rotação de tarefas*

Abstract

The assembly line worker assignment and balancing problem is an extension of the simple assembly line balancing problem in which the task execution times depend on the assigned workers. This problem draws its practical motivation from assembly lines with workers with disabilities. In this doctoral thesis, we study two extensions for this problem: the first one allows layouts with parallel workstations, while the second one allows multiple parallel lines. These extensions were applied for the base problem as well as the job rotation problem. We present mathematical formulations and exact and heuristic methods for all cases. Computational tests in instances from the literature and new instances and a detailed analysis of the results are presented.

Keywords: *Assembly lines, parallel workstations, parallel lines, job rotation*

Sumário

Lista de Figuras

Lista de Tabelas

1	Introdução	p. 3
2	Estado da arte	p. 7
2.1	SALBP	p. 8
2.1.1	Revisão bibliográfica	p. 9
2.1.2	Formulação para o SALBP	p. 10
2.2	ALWABP	p. 11
2.2.1	Revisão bibliográfica	p. 13
2.2.2	Formulação para o ALWABP	p. 13
2.3	Extensões	p. 15
2.3.1	Estações em paralelo e múltiplas linhas	p. 15
2.3.1.1	Revisão bibliográfica	p. 17
2.3.2	Rotação de tarefas	p. 17
2.3.2.1	Revisão bibliográfica	p. 18
3	Extensões para o ALWABP	p. 21
3.1	ALBP com estações em paralelo	p. 21
3.2	ALWABP com estações em paralelo	p. 23
3.2.1	Formulação para o ALWABP com estações em paralelo	p. 25

3.2.2	Heurística baseada em busca em vizinhança	p. 29
3.2.2.1	Geração da solução inicial	p. 29
3.2.2.2	Busca em vizinhança	p. 33
3.2.2.3	Resultados	p. 34
3.3	ALWABP com múltiplas linhas	p. 39
3.3.1	Formulação para o ALWABP com múltiplas linhas	p. 40
3.3.1.1	Resultados	p. 43
3.3.2	Método exato	p. 44
3.3.2.1	Resultados	p. 45
3.3.3	Heurística para o ALWABP com múltiplas linhas	p. 46
3.3.3.1	Geração de conjuntos de tarefas	p. 48
	Exemplo:	p. 51
3.3.3.2	Geração dos conjuntos de trabalhadores	p. 52
	Exemplo:	p. 54
3.3.3.3	Avaliação dos conjuntos	p. 54
3.3.3.4	Seleção de conjuntos	p. 55
3.3.3.5	Resultados	p. 56
4	Rotação de tarefas	p. 59
4.1	Rotação de tarefas com estações em paralelo	p. 59
4.1.1	Modelo	p. 59
4.1.2	Heurística híbrida	p. 62
4.1.2.1	Geração do pool de soluções parciais	p. 63
4.1.2.2	Busca local	p. 64
4.1.2.3	Modelo	p. 65
4.2	Rotação de tarefas com múltiplas linhas	p. 66
4.2.1	Modelo	p. 66

4.2.2	Heurística híbrida	p. 68
4.2.2.1	Geração do pool de soluções parciais	p. 69
4.2.2.2	Modelo	p. 70
4.3	Rotação de tarefas com múltiplas linhas e estações em paralelo	p. 71
4.3.1	Modelo	p. 71
4.3.2	Heurística híbrida	p. 73
4.4	Medida de qualidade para a rotação de tarefas	p. 73
4.5	Resultados	p. 74
5	Conclusões e trabalhos futuros	p. 79
5.1	Trabalhos futuros	p. 80
	Apêndice A – Resultados detalhados dos testes	p. 81

1 *Introdução*

De acordo com a Organização Mundial de Saúde, cerca de 650 milhões de pessoas no mundo possuem algum tipo de deficiência. No Brasil, o Censo Demográfico 2010 aponta que 8,3% da população brasileira (15,8 milhões de pessoas) possuem algum tipo de deficiência. No entanto, pessoas com deficiência correspondem a apenas 2,05% do total de empregados, o que indica um alto índice de desemprego nesta parte da população (*Retratos da Deficiência no Brasil*; 2003).

Simonelli et al. (2006) e Simonelli (2009) analisam as políticas e práticas de inclusão de pessoas portadoras de deficiência no mercado de trabalho brasileiro. Uma das constatações interessantes destes estudos foi o fato da grande maioria de vagas existentes para trabalhadores portadores de deficiências se encontrarem em contextos produtivos. Esta conclusão é condizente com a iniciativa de diversos países com altos índices de desenvolvimento humano tais como Espanha, Inglaterra e Japão, que promovem a manutenção de Centros de Trabalho para Deficientes (CTD's) com linhas produtivas (*Human Development Reports* (2012)).

CTD's são organizações sem fins lucrativos que empregam trabalhadores com deficiência, muitas vezes em linhas de montagem terceirizadas de grandes empresas. Assim como qualquer outra empresa, os CTD's precisam competir com o mercado. Eventuais lucros que poderiam ser obtidos por estas organizações são convertidos em novos empregos.

Miralles et al. (2007) identificaram as seguintes características em CTD's:

- ☐ Os tempos de execução de cada tarefa costumam variar bastante entre os trabalhadores.
- ☐ Dependendo do tipo de deficiência, alguns trabalhadores não conseguem executar algumas tarefas.
- ☐ Algumas tarefas devem ser alocadas a alguns trabalhadores *a priori*, por motivos terapêuticos.

Em geral, não há trabalhadores universalmente lentos ou rápidos. Um trabalhador pode ser bastante eficiente para algumas tarefas e muito lento para outras.

Algumas deficiências requerem a designação destes trabalhadores a determinadas posições (estágios) da linha. Por exemplo, trabalhadores com deficiência auditiva em geral trabalham melhor na primeira ou última estação, uma vez que as estações intermediárias requerem um nível maior de coordenação e comunicação.

O objetivo principal dos CTD's é incentivar os trabalhadores a desenvolver e aprimorar suas habilidades, buscando integrá-los, assim que possível, ao mercado convencional de trabalho. Com isto, é comum que a escala de funcionários seja alterada constantemente, já que vários trabalhadores costumam deixar o CTD ao e passam a trabalhar em empresas convencionais.

Trabalhadores com deficiência costumam ter mais problemas de saúde e, portanto, costumam faltar com mais frequência ao trabalho.

Além disso, acompanhamento psicológico é essencial em um CTD.

Os três últimos itens requerem que a linha de produção seja replanejada constantemente, o que exige métodos rápidos para reconfigurar a linha para a nova escala de funcionários.

Neste trabalho estudamos duas extensões para o problema de balanceamento e designação de trabalhadores em linhas de produção (ALWABP, do inglês *assembly line worker assignment and balancing problem*), introduzido por Miralles et al. (2007). A primeira extensão permite que mais de um trabalhador execute o mesmo conjunto de tarefas, porém em produtos diferentes. Já a segunda permite dividir os trabalhadores em grupos, em que cada grupo é responsável por uma linha de produção independente.

As duas extensões propostas expandem o número de configurações possíveis para a linha de produção. Com isto, podemos encontrar soluções mais eficientes (em termos de produtividade) do que aquelas que conseguimos com o modelo serial. O ALWABP com estações em paralelo possibilita a execução de tarefas cujos tempos de execução sejam maiores que o tempo de ciclo da linha, bastando que dois ou mais trabalhadores executem esta tarefa em paralelo.

Já o ALWABP com linhas em paralelo leva esta característica ainda mais longe, permitindo linhas que, embora individualmente mais lentas que uma única linha serial com todos os trabalhadores, quando combinadas resultam em uma taxa de produção mais

rápida. Isto permite que os trabalhadores executem combinações de tarefas que eles não seriam capazes de executar no ALWABP serial ou com estações em paralelo.

Estas extensões serão também aplicadas ao problema de rotação de tarefas. A rotação de tarefas permite que cada trabalhador execute uma quantidade maior de tarefas, o que permite um treinamento mais completo do mesmo. O uso de múltiplas linhas e estações em paralelo permite que o trabalhador execute conjuntos de tarefas que o mesmo não conseguiria no problema serial, o que possibilita encontrar soluções para o problema de rotação de tarefas com pouca ou nenhuma perda de desempenho da linha de produção.

O restante deste trabalho está dividido da seguinte forma: no Capítulo 2 descrevemos o problema simples de balanceamento de linhas de produção (SALBP, do inglês *simple assembly line balancing problem*), o ALWABP e o problema de rotação de tarefas, bem como as extensões propostas neste trabalho: estações em paralelo e múltiplas linhas. Para cada problema, apresentamos uma revisão bibliográfica. No Capítulo 3 apresentamos modelos e métodos para as extensões do ALWABP supracitadas, enquanto que no Capítulo 4 fazemos o mesmo para o problema de rotação de tarefas. Por fim, o Capítulo 5 apresenta algumas conclusões, bem como propostas de trabalhos futuros.

2 *Estado da arte*

Uma linha de montagem é constituída de um conjunto finito e conhecido de tarefas e um conjunto de estações de trabalho nas quais estas tarefas devem ser executadas. Cada tarefa possui um tempo de processamento e um conjunto de relações de precedência determina a ordem relativa na qual essas tarefas podem ser executadas. Estas relações podem ser representadas por um grafo de precedência. Os nós do grafo representam as tarefas e cada arco representa uma relação de precedência a ser obedecida por um par de tarefas. Um arco partindo de um nó 1 para um nó 2 indica que a tarefa 2 só pode ser executada depois que a tarefa 1 for concluída. A Figura 1, retirada de Araújo (2011), mostra um exemplo de grafo de precedência.

Figura 1: Exemplo de Grafo de Precedência

O processo de fabricação de um item requer a execução de todas as tarefas. Em uma linha de montagem, esta execução é dividida em estágios contendo estações de trabalho. Em cada estágio um subconjunto das tarefas necessárias para a fabricação deste produto é realizado. Em linhas seriais, cada estágio corresponde a uma estação. O tempo de execução de um estágio corresponde ao tempo necessário para este estágio executar suas tarefas. Para que a produção da linha seja constante cada estágio dispõe do mesmo tempo para executar as tarefas nele alocadas. Portanto, a velocidade da linha é limitada pelo estágio mais lento. Chamamos de tempo de ciclo o tempo de execução do estágio mais carregado da linha.

O problema de balanceamento de linhas de produção (ALBP, do inglês *assembly line balancing problem*) consiste em alocar tarefas a estações de trabalho de modo a otimizar

certa característica do problema. Este problema possui quatro variantes de acordo com a característica que desejamos otimizar. No ALBP-1 desejamos minimizar o número de estações para um determinado tempo de ciclo. No ALBP-2 o número de estações seja fixo e desejamos minimizar o tempo de ciclo. Se desejamos minimizar o produto entre o número de estações e o tempo de ciclo, temos o ALBP-E. Por fim, se desejamos apenas verificar a existência de uma solução viável para um dado número de estações e um tempo de ciclo fixo, temos o ALBP-F.

Nas seções a seguir apresentamos dois problemas de balanceamento de linhas de produção: o clássico SALBP e o mais recente ALWABP, um dos nossos problemas em estudo. Também analisamos como a introdução de layouts com múltiplas linhas e estações em paralelo pode afetar ambos os problemas. Por fim, analisamos também o problema de rotação de tarefas num contexto com múltiplas linhas e estações em paralelo.

2.1 SALBP

O SALBP corresponde à versão mais simplificada do ALBP. No SALBP os tempos de execução das tarefas são fixos, conhecidos e independentes da estação de trabalho escolhida. Cada estágio possui uma única estação de trabalho, na qual são executadas as tarefas a ele alocadas, e cada tarefa só pode ser alocada a um estágio. Além disso, supomos que a linha de produção é ritmada, isto é, o produto fica disponível durante a mesma quantidade de tempo em cada estágio da linha.

A Fig. 2, retirada de Araújo (2011), mostra uma possível solução para o SALBP, utilizando o grafo de precedência da Fig. 1 e os tempos de execução da tabela 1. Esta linha de produção possui quatro estágios, T_1 , T_2 , T_3 e T_4 .

Tabela 1: Exemplos para os tempos de execução das tarefas da Fig. 1 para o caso SALBP

Tarefas	1	2	3	4	5	6	7	8	9	10
Tempos(s)	4	3	9	5	9	4	8	7	5	1

Nesta solução, temos 4 estações, cada uma com um trabalhador. Utilizando os dados da Tabela 1 temos que os tempos de execução em cada estágio correspondem, respectivamente, a 16s, 14s, 12s e 13s. O tempo de ciclo desta configuração é de 16s, ou seja, o tempo do estágio mais lento. Assim, a cada 16s, temos uma unidade completa produzida.

O SALBP é um problema bastante flexível, permitindo várias modificações para se adequar melhor a diferentes casos práticos. Becker e Scholl Becker e Scholl (2006) desta-

Figura 2: Exemplo de Solução para o SALBP para o caso da Fig. 1

cam as seguintes variações para o problema:

- Quanto ao número de modelos produzidos: modelo único, multi-modelos
- Quanto à forma de controle da linha: ritmada, arritmada síncrona e arritmada assíncrona
- Quanto ao nível de automação: linhas manuais e linhas automatizadas
- Quanto à frequência: instalação e reconfiguração
- Quanto ao formato da linha: serial, em U, múltiplos Us (n-U)
- Quanto ao uso de paralelismo: sem paralelismo, paralelismo de linhas, estações, tarefas e áreas de trabalho
- Quanto aos tempos das tarefas: fixos, estocásticos e dependentes da estação
- Outras configurações adicionais: uso de *buffers*, *feeders*, mudanças no posicionamento de equipamento, posicionamento e dimensionamento de material, entre outros

2.1.1 Revisão bibliográfica

O SALBP é um problema clássico da área de pesquisa operacional, tendo sido estudado por várias décadas. Diversas extensões para o problema foram propostas, tentando se adequar aos mais diferentes casos possíveis e proporcionar maior flexibilidade ao problema. Baybars (1986) apresenta a primeira revisão bibliográfica para o problema. A revisão discute diversos métodos exatos e variantes do SALBP. São excluídos, no entanto, métodos aproximados e algoritmos para o GALBP (*general assembly line balancing problem*). Becker e Scholl (2006) apresentam uma revisão mais recente para o GALBP. Este

trabalho identifica diversas extensões para o problema, como modelos orientados a custo e lucro, seleção de equipamentos, alternativas de processamento, uso de estações e tarefas em paralelo, linhas de produção em U, restrições relacionadas ao posicionamento dos equipamentos, às estações, tarefas e trabalhadores, tempos estocásticos e produção de diversos tipos de produtos diferentes. Boysen et al. (2007) apresentam uma classificação para problemas de balanceamento de linhas de produção. A classificação utiliza como base características do grafo de precedência, características das estações e da linha de produção, e o objetivo do modelo. Boysen et al. (2008) apresentam uma classificação para diversos artigos relacionados ao ALBP. A classificação apresentada leva em consideração o número de modelos a serem produzidos e a forma como esta produção é feita, o tipo de controle da linha (ritmada, arritmada síncrona e arritmada assíncrona), frequência de utilização (instalação e reconfiguração), nível de automação e aplicações específicas. Sivasankaran e Shahabudeen (2014) classificam problemas de balanceamento de linhas de produção em oito tipos, baseados no número de modelos, nos tempos das tarefas e no tipo de linha.

2.1.2 Formulação para o SALBP

A seguir apresentamos uma formulação para o SALBP (Scholl e Becker (2006)). Uma vez que, em CTD's, desejamos alocar todos os trabalhadores disponíveis maximizando a eficiência da linha, nos interessamos aqui pelo SALPB-2.

Utilizando a notação:

- $i; j$ Índices das tarefas
- s Índices dos estágios
- N Conjunto de tarefas
- S Conjunto de estágios
- p_i Tempo de execução da tarefa i
- D_j Conjunto de tarefas imediatamente predecessoras à tarefa j no grafo de precedência

E as variáveis:

- C Tempo de ciclo
- x_{si} Variável binária. Igual a 1 se a tarefa i é executada no estágio s , igual a 0 caso contrário

$$\text{Min } C \quad (2.1)$$

s.a.

$$\sum_{s \in S} x_{si} \leq \sum_{s \in S} x_{sj} \quad \forall j \in N; \forall i \in D_j \quad (2.2)$$

$$\sum_{s \in S} x_{si} = 1 \quad \forall i \in N \quad (2.3)$$

$$\sum_{i \in N} p_i x_{si} \leq C \quad \forall s \in S \quad (2.4)$$

A função objetivo (2.1) minimiza o tempo de ciclo. As restrições (2.2) estabelecem as relações de precedência entre as tarefas. As restrições (2.3) garantem que todas as tarefas serão executadas. Por fim, as restrições (2.4) determinam que o tempo de ciclo da linha de montagem corresponde ao tempo de execução da estação mais carregada.

Neste trabalho nos interessamos pela variação do SALBP em que os tempos das tarefas dependem do trabalhador ao qual elas são alocadas. Este problema, conhecido na literatura como problema de balanceamento e designação de trabalhadores (do inglês *assembly line worker assignment and balancing problem*) (ALWABP), é descrito na seção a seguir.

2.2 ALWABP

Em CTD's os tempos de execução das tarefas podem ser diferentes para cada trabalhador. Conforme dito anteriormente, não existem trabalhadores universalmente lentos ou rápidos. Um trabalhador pode ser muito eficiente para uma tarefa, porém muito lento para outra. Além disso, dependendo do tipo de deficiência que o trabalhador possui, ele pode não ser capaz de executar algumas tarefas ou o fazer de forma extremamente lenta, de tal forma que é impraticável alocar esta tarefa a este trabalhador.

O SALBP não modela adequadamente esta situação. Em vista disto, Miralles et al. (2007) propuseram o ALWABP, uma extensão do SALBP capaz de lidar diferentes tempos de execução para uma mesma tarefa, de acordo com o trabalhador escolhido. Desta forma, além de determinar quais tarefas devem ser executadas em cada estágio, devemos também determinar qual trabalhador deverá executar estas tarefas.

Da mesma forma que o ALBP, o ALWABP pode ser categorizado em ALWABP-1, ALWABP-2, ALWABP-E, de acordo com a função objetivo a otimizar, ou ALWABP-F, se o problema consiste em verificar a existência de uma solução viável dados um tempo de ciclo e um número de estações. Em CTD's, desejamos maximizar a eficiência da linha de produção empregando todos os trabalhadores disponíveis. Portanto, este trabalho se interessa, em particular, pelo ALWABP-2.

A Fig. 3 mostra uma possível solução para o ALWABP, utilizando o grafo de precedência da Fig. 1 e os tempos de execução da tabela 2, em que W_1 , W_2 , W_3 e W_4 representam quatro trabalhadores.

Tabela 2: Exemplos de tempos de execução das tarefas da Fig. 1 para o caso ALWABP

Tarefas		1	2	3	4	5	6	7	8	9	10
Tempos(s)	W_1	4	3	9	5	9	4	8	7	5	1
	W_2	-	1	2	3	4	4	-	4	2	1
	W_3	4	3	9	5	2	2	5	3	2	2
	W_4	4	1	5	1	2	3	4	2	-	1

Neste exemplo, ao contrário do anterior, os tempos das tarefas podem variar de acordo com o trabalhador que a executa. A tarefa 4, por exemplo, é executada em 5s pelos trabalhadores W_1 e W_3 , em 3s pelo trabalhador W_2 e em 1s pelo trabalhador W_4 . Além disto, o trabalhador W_2 não pode executar as tarefas 1 e 7, enquanto que o trabalhador W_4 não pode executar a tarefa 9.

Figura 3: Exemplo de Solução para o ALWABP para o caso da Fig. 1

Na Figura 3, retirada de Araújo (2011), temos uma configuração com quatro estágios,

em que cada estágio é definido pelas tarefas alocadas e pelo trabalhador que as executa. No primeiro estágio está alocado o trabalhador W_4 , cuja carga alocada é de 10s. No segundo, o trabalhador W_2 , que executa as tarefas 4 e 5 em 7s. O trabalhador W_1 está alocado no terceiro estágio, com tempo de execução de 12s. No último estágio temos o trabalhador W_3 , com tempo de execução de 7s. O tempo de ciclo da linha é, portanto, de 12s.

No SALBP, uma vez que os trabalhadores são suficientemente similares em termos de tempos de execução das tarefas, trocar trabalhadores de estação não altera a eficiência da linha. O mesmo não ocorre no ALWABP. Por exemplo, trocar os trabalhadores W_2 e W_4 de estação torna a solução inviável, pois o trabalhador W_2 não pode executar a tarefa 1.

2.2.1 Revisão bibliográfica

Miralles et al. (2007) realizaram o primeiro estudo sobre o ALWABP. Os autores mostram que a divisão do trabalho a ser realizado em tarefas menores é capaz de tornar certas deficiências dos trabalhadores transparentes. É apresentado um modelo para o ALWABP-2, cuja função objetivo minimiza o tempo de ciclo. Os autores apresentam também um estudo de caso em um CTD na Espanha, que obteve bons resultados.

Devido à complexidade do problema, vários autores propuseram métodos heurísticos para o mesmo. Chaves (2009); Chaves et al. (2007, 2008) estudam uma meta-heurística híbrida denominada busca por agrupamentos (CS, do inglês *clustering search*). Os autores propõem o uso de CS para diversos problemas, entre eles o ALWABP. Além disto, estes autores geraram um conjunto de instâncias para o problema, divididas em 4 famílias de acordo com o número de tarefas, a densidade do grafo de precedências e a variabilidade dos tempos de execução do problema. Outros autores propõem métodos como Busca Tabu (Moreira e Costa (2009a)), Iterated Beam Search (Blum e Miralles (2011); Blum (2011)), heurísticas construtivas (Moreira et al. (2012)) e algoritmos genéticos (Moreira e Costa (2012) e Mutlu et al. (2013)). Além disso, Miralles et al. (2008), Borba e Ritt (2012) e Borba e Ritt (2014) propõem métodos exatos do tipo *branch and bound* para o problema.

Em relação a extensões para o ALWABP, Araújo et al. (2010b); Araújo e Costa (2010) apresentam um modelo linear para o ALWABP com estações em paralelo. Araújo et al. (2010a) realizam testes com novas instâncias e Araújo et al. (2012) estudam o ALWABP colaborativo, em que os trabalhadores podem se alternar em uma mesma estação para executar as tarefas. Cortez e Costa (2011, 2013, 2014) apresentam um modelo matemático e uma heurística híbrida para o ALWABP com multi-modelos. Costa et al. (2009); Moreira

et al. (2015) estudam o problema de balanceamento e integração de trabalhadores em linhas de produção. Neste problema, os autores estudam o impacto de adicionar um trabalhador a uma linha existente. Na seção a seguir apresentamos uma formulação para o ALWABP-2.

2.2.2 Formulação para o ALWABP

A seguir é apresentada a formulação para o ALWABP-2 proposta por Miralles et al. (2007). A versão apresentada contém pequenas melhorias introduzidas por Moreira e Costa (2009a).

Utilizando a notação:

- i, j Índices das tarefas
- w Índices dos trabalhadores
- s Índices dos estágios
- N Conjunto de tarefas
- W Conjunto de trabalhadores
- S Conjunto de estágios
- p_{wi} Tempo de execução da tarefa i pelo trabalhador w
- D_j Conjunto de tarefas imediatamente predecessoras à tarefa j no grafo de precedência
- I_w Conjunto de tarefas que o trabalhador w não é capaz de executar

E as variáveis:

- C Tempo de ciclo
- x_{swi} Variável binária. Igual a 1 se a tarefa i é executada pelo trabalhador w no estágio s , igual a 0 caso contrário
- y_{sw} Variável binária. Igual a 1 se o trabalhador w está no estágio s , igual a 0 caso contrário

$$\text{Min } C \quad (2.5)$$

s.a.

$$\sum_{w \in W} \sum_{s \in S} x_{swi} = 1 \quad \forall i \in N \quad (2.6)$$

$$\sum_{s \in S} x_{sw} = 1 \quad \forall w \in W \quad (2.7)$$

$$\sum_{w \in W} x_{sw} = 1 \quad \forall s \in S \quad (2.8)$$

$$\sum_{w \in W} x_{swi} \leq \sum_{w \in W} x_{swj} \quad \forall j \in N; \forall i \in D_j \quad (2.9)$$

$$\sum_{w \in W} x_{swi} \leq p_{wi} \quad \forall s \in S \quad (2.10)$$

$$\sum_{i \in N} x_{swi} \leq n_j y_{sw} \quad \forall w \in W; \forall s \in S \quad (2.11)$$

$$x_{swi} = 0 \quad \forall w \in W; \forall s \in S; \forall i \in I_w \quad (2.12)$$

A função objetivo (2.5) minimiza o tempo de ciclo. As restrições (2.6) garantem que cada tarefa será executada por um trabalhador em um estágio. As restrições (2.7) e (2.8) estabelecem que uma relação biunívoca entre trabalhadores e estágios. As restrições (2.9) estabelecem as relações de precedência entre as tarefas. As restrições (2.10) definem o tempo de ciclo como o tempo do estágio mais lento. As restrições (2.11) indicam que uma tarefa só pode ser executada por um trabalhador em um estágio se este trabalhador estiver alocado a este estágio. Por fim, as restrições (2.12) tratam das tarefas que determinados trabalhadores não podem executar.

2.3 Extensões

Nos casos anteriores temos apenas uma linha de montagem e cada estágio possui apenas uma estação de trabalho. O uso de layouts alternativos permite novas configurações para a linha de produção, o que pode melhorar consideravelmente sua eficiência. Neste

trabalho, estudamos o ALWABP com estações em paralelo e o ALWABP com múltiplas linhas. Além disto, aplicamos as duas extensões ao problema de rotação de tarefas.

2.3.1 Estações em paralelo e múltiplas linhas

Nos casos com múltiplas linhas ou estações em paralelo cada unidade duplicada (linha ou estação) deve executar as mesmas tarefas que as demais unidades, porém com trabalhadores diferentes, visto que não podemos alocar um trabalhador a mais de uma estação. Definimos a taxa de execução de uma estação, estágio ou linha como o número de itens processados em uma unidade de tempo. Para uma estação, isto equivale ao inverso da soma dos tempos de execução das tarefas alocadas, enquanto que para uma linha ele equivale ao inverso do tempo de ciclo. A taxa de execução de um conjunto de unidades é, portanto, dada pela soma das taxas de execução de cada unidade.

Uma das vantagens da introdução de paralelismo é que elas permitem tarefas com tempo de execução maiores que o tempo de ciclo. No modelo serial, se uma tarefa tem tempo de execução muito grande, o estágio à qual esta for alocado pode ter um tempo de execução muito maior que os demais. Utilizando estações em paralelo, podemos adicionar mais estações àquele estágio. O mesmo acontece no caso com múltiplas linhas, visto que o impacto de uma tarefa com tempo de execução muito grande é reduzido pelo uso de várias linhas de montagem. Isto é especialmente interessante no caso do ALWABP, uma vez que permite que trabalhadores executem tarefas para as quais não são muito eficientes, possibilitando uma maior variedade de configurações da linha de montagem.

Paralelismo permite também um maior aproveitamento de cada estágio. No modelo serial, a adição de muitas tarefas a um mesmo estágio pode prejudicar o tempo de ciclo. No modelo paralelo, no entanto, a adição de mais estações de trabalho a este estágio reduz o tempo total de execução. Assim, temos um maior número de tarefas executando em um estágio sem redução da taxa de produção.

Em contrapartida, a execução das mesmas tarefas em várias unidades aumenta o custo total dos equipamentos a serem adquiridos, uma vez que o equipamento necessário para a execução de uma dada tarefa deverá ser replicado em cada estação de trabalho em que a tarefa se encontra alocada. Além disso, o espaço disponível no local de trabalho pode restringir o número máximo de estações e linhas duplicadas.

Sejam $t_1; t_2; \dots; t_i; \dots; t_n$ os tempos de execução de n unidades para um mesmo conjunto de tarefas, o tempo de execução T destas unidades em paralelo é dado por

$$\frac{1}{T} = \frac{1}{t_1} + \frac{1}{t_2} + \dots + \frac{1}{t_i} + \dots + \frac{1}{t_n} > \frac{1}{t_i}. \text{ Logo,}$$

$$\frac{1}{T} > \frac{1}{t_i} \quad ! \quad T < t_i.$$

Ou seja, o tempo de execução de duas ou mais unidades duplicadas é sempre menor que o tempo de execução de cada um destas unidades, executando as mesmas tarefas. Esta característica proporciona uma grande vantagem ao ALWABP caso desejemos adicionar um novo trabalhador à linha de produção que seja pouco eficiente em relação aos demais.

Uma analogia que pode ser feita é um conjunto de torneiras enchendo um conjunto de baldes. Se, por exemplo, um das torneiras enche cinco baldes por minuto enquanto que a outra enche três baldes por minuto, juntas elas conseguem encher oito baldes por minuto. Neste caso cada balde corresponde a um item a ser processado, enquanto que as torneiras correspondem aos trabalhadores (no caso de estações em paralelo) ou linhas (múltiplas linhas).

2.3.1.1 Revisão bibliográfica

Diversos trabalhos tratam do ALBP com estações em paralelo, tanto de forma exata quanto utilizando heurísticas. Boysen e Flidner (2006) apresentam um algoritmo gráfico de duas fases que pode ser adaptado para trabalhar com estações em paralelo. Ege et al. (2009) propõem dois métodos *branch and bound*, um exato e um heurístico. Outros autores estudam este problema em conjunto com outras extensões. Zhou (1997) propõem uma heurística para o ALBP com linhas mistas e estações em paralelo. Para o mesmo problema, Süer (1998) propõe um método de resolução em três fases, McMullen e Frazier (1997) propõem uma heurística para o problema com tempos estocásticos e Simaria e Vilarinho (2004) definem um modelo matemático e um Algoritmo Genético iterativo. Bukchin e Rubinovitz (2003) apresentam duas formulações e um algoritmo *branch and bound* para o ALBP com estações paralelas e seleção de equipamentos. Plebani e Chen (2006) apresentam um modelo e um algoritmo *branch and bound* para o ALBP em linhas em U com máquinas paralelas. Tiacci e Saetta (2007) apresentam um simulador orientado a processo para resolução do ALBP capaz de tratar modelos mistos, tempos estocásticos, estações paralelas e diferentes sequências de agendamento (*scheduling sequences*).

O uso de estações em paralelo para o ALWABP começou a ser estudado recentemente. Araújo et al. (2010a,b) apresentaram um modelo linear para o ALWABP com estações paralelas. Os autores realizaram testes em instâncias disponíveis na literatura e em instâncias modificadas, geradas pelos próprios autores. Estas instâncias buscam

simular situações em que alguns trabalhadores possuem deficiências semelhantes. Araújo e Costa (2010) estenderam o trabalho anterior, gerando novas instâncias, que simulam a existência de um trabalhador que é muito lento em relação aos demais.

No Capítulo 3 apresentamos modelos e métodos para as duas extensões.

2.3.2 Rotação de tarefas

Dados um conjunto de trabalhadores, um conjunto de tarefas a serem realizadas e um conjunto de períodos, o problema de rotação de tarefas consiste em criar diversas alocações destas tarefas aos trabalhadores, sendo que cada alocação será usada em um período. No contexto de linhas de produção desejamos obter uma configuração da linha para cada período, em que cada trabalhador pode executar tarefas diferentes em cada período.

Com isto, espera-se que o trabalhador ganhe experiência mais rapidamente e se sinta mais motivado que aquele que não faz rotação de tarefas, além de reduzir o risco de doenças do trabalho. Para o caso encontrado em CTD's, a rotação de tarefas pode fazer com que cada trabalhador execute tarefas que não faria em situações com uma única configuração da linha. Neste caso, desejamos maximizar o número de tarefas distintas que cada trabalhador executa dada uma demanda a ser atendida.

Este é um problema complexo, mesmo para linhas de produção convencionais. Para o ALWABP, este problema se torna ainda mais difícil de resolver, uma vez que a simples troca de trabalhadores entre períodos não garante a qualidade (e nem sequer a viabilidade) da solução, seja pela alocação de tarefas ineficazes a um trabalhador ou por não ser capaz de atender a demanda devido a um trabalhador executar tarefas para as quais é pouco eficiente.

No Capítulo 4 apresentamos modelos e métodos para o problema de rotação de tarefas com estações em paralelo, com múltiplas linhas e com ambas as extensões.

2.3.2.1 Revisão bibliográfica

A importância da rotação de tarefas em linhas de produção é bastante evidente na literatura. Michalos et al. (2013) examina o efeito de fatores como fadiga, competência do operador e realização de tarefas bastante repetitivas no desempenho do trabalhador. Carnahan et al. (2000) foram os primeiros a modelar e resolver o problema de rotação de tarefas para evitar lesões nas costas, utilizando programação inteira e algoritmos

genéticos. Tharmmaphornphilas e Norman (2007) utilizam um método heurístico para programar a rotação de tarefas que visa minimizar o risco de lesões nas costas resultantes da manipulação de objetos pesados. Seçkiner e Kurt (2007) utilizam um método baseado em *simulated annealing* para resolver o problema com o objetivo de minimizar a carga de trabalho. Sato e Coury (2009) estudam os efeitos da rotação de tarefas sobre o desconforto, distúrbios e faltas por doenças dos trabalhadores, enquanto que Azizi et al. (2010) realizam um estudo similar em relação ao tédio dos trabalhadores. Diego-Mas et al. (2009) propõem um algoritmo genético para o problema que visa reduzir o risco de problemas esquelético-musculares, levando em consideração deficiências temporárias, permanentes e preferências dos trabalhadores. Otto e Scholl (2011) definem formalmente o problema ErgoSALBP que leva em consideração esforço físico e tempo de execução das tarefas.

O uso de rotação de tarefas aplicada ao ALWABP foi proposto por Costa e Miralles (2009). Os autores propuseram um método de decomposição que, embora tenha obtido bons resultados para instâncias pequenas, se mostrou inviável para instâncias maiores. Moreira e Costa (2012) propuseram heurísticas híbridas para a solução do problema, em que o método inicialmente gera um conjunto de soluções parciais e em seguida utiliza um método exato para obter uma solução a partir destas. Araújo et al. (2014) apresentam modelos e métodos para o problema de rotação de tarefas com estações em paralelo.

3 Extensões para o ALWABP

Neste capítulo descrevemos em detalhes o ALWABP com múltiplas linhas e para o ALWABP com estações em paralelo. Na Seção 3.1 explicamos o impacto do uso de estações em paralelo e múltiplas linhas no tempo de execução de linhas de produção convencionais. Em seguida, na seção 3.2 detalhamos o uso de estações em paralelo em linhas com trabalhadores com deficiência, enquanto que na Seção 3.3 detalhamos o uso de múltiplas linhas para o mesmo caso.

3.1 ALBP com estações em paralelo

Conforme dito anteriormente, o tempo de execução de um estágio para o SALBP corresponde ao tempo de execução da estação que pertence a este estágio, isto é, a soma dos tempos de execução das tarefas nele executadas. Caso desejemos utilizar estações em paralelo em casos com trabalhadores semelhantes, o tempo de execução de cada estágio corresponde ao tempo total das tarefas dividido pelo número de estações neste estágio. Por exemplo, se uma máquina executa um conjunto de tarefas em 60 segundos, ou seja, o conjunto de tarefas é completado uma vez a cada minuto, três máquinas similares executarão o mesmo conjunto de tarefas, em média, três vezes por minuto, o que equivale a um tempo de ciclo de 20 segundos.

A Figura 4 mostra uma solução para o ALBP com estações paralelas, utilizando o grafo de precedência da Figura 1 os tempos de execução da Tabela 1.

Nesta solução, temos dois estágios, cada um com dois trabalhadores. Para facilitar a visualização, as relações de precedência entre os estágios foram representadas efetuando-se cópias das tarefas 4, 5, 6 e 8. Cada um dos dois trabalhadores do primeiro estágio executa as tarefas em 30s. Desta forma, trabalhando juntos, eles executam estas mesmas tarefas, em média, a cada 15s. Da mesma forma, os dois trabalhadores do segundo estágio executariam suas tarefas em 25s, mas trabalhando juntos equivalem a um único

Figura 4: Exemplo de Solução para o ALBP utilizando estações paralelas para o caso da Figura 1

trabalhador executando estas tarefas em 12,5s.

Uma vez que o tempos de execução das estações em paralelo são os mesmos, podemos efetivamente produzir um produto a cada 15 segundos se introduzirmos um atraso em um dos trabalhadores do primeiro estágio. Se o primeiro trabalhador iniciar sua produção no tempo 0 e o segundo trabalhador esperar 15 segundos para iniciar a sua produção, teremos um produto finalizado a cada 15 segundos. De fato, isto pode ser estendido para qualquer estágio em que o tempo total das tarefas é T e que possui n trabalhadores. Para isto, basta que atrasemos o início da produção de trabalhadores em paralelo. Em cada estágio, um trabalhador deve iniciar a produção juntamente com o resto da linha, enquanto que cada trabalhador adicional deve esperar $\frac{T}{n}$ para iniciar a produção.

Este mesmo cálculo pode ser usado também em casos com linhas de produção duplicadas. O tempo de ciclo equivalente do sistema corresponde ao tempo de ciclo de uma única linha de produção dividido pelo número de linhas. Assim como foi feito no caso com estações em paralelo, é possível introduzir um atraso em uma das linhas de modo a obter uma taxa de produção constante.

É importante ressaltar que o uso de estações em paralelo não requer o reposicionamento das estações de trabalho. Basta que os trabalhadores de cada estágio processem os itens alternadamente.

3.2 ALWABP com estações em paralelo

No ALWABP, uma vez que os tempos das tarefas podem variar dependendo do trabalhador que as executa, o cálculo do tempo de ciclo torna-se mais complexo. Se um trabalhador executa um conjunto de tarefas em 40 segundos (1,5 vezes por minuto) e outro executa as mesmas tarefas em 60 segundos (1 vez por minuto), trabalhando em paralelo estes dois trabalhadores executariam estas tarefas, em média, 2,5 vezes por minuto, o que corresponde a um tempo de ciclo de 24 segundos.

A Figura 5 mostra uma solução para o ALWABP com estações paralelas, utilizando o grafo de precedência da figura 1 os tempos de execução da tabela 2.

Figura 5: Exemplo de Solução para o ALWABP utilizando estações paralelas para o caso da Fig. 1

Nesta solução, o trabalhador W_4 está no primeiro estágio, os trabalhadores W_2 e W_3 estão juntos no segundo estágio e o trabalhador W_1 está no último estágio. O trabalhador W_2 executa as tarefas 4, 5, 6, 8 e 9 em 17s, enquanto que o trabalhador W_3 executa estas mesmas tarefas em 14s. Trabalhando em paralelo, o tempo de execução destes dois trabalhadores é de $\frac{1}{\frac{1}{17} + \frac{1}{14}} = 7,6s$. Com isto, conseguimos um tempo de ciclo de 10s para esta configuração da linha de produção.

Diferentemente do ALBP com estações em paralelo, caso os tempos de execução das estações sejam diferentes, não é possível produzir os itens a uma taxa constante usando paralelismo. Na solução anterior, por exemplo, ao término do primeiro ciclo o trabalhador W_4 termina o processamento de um item. Digamos que este item seja entregue ao trabalhador W_2 . Ao término do segundo ciclo de 10 segundos o trabalhador W_4 termina de processar outro item, porém o trabalhador W_2 ainda não terminou de processar seu item, uma vez que ele precisa de 17 segundos. Este item só será totalmente processado

durante o terceiro ciclo. Em vista disso, para linhas ritmadas, esta adaptação exige o uso de *buffers* nos estágios com mais de uma estação. Cada item cujo processamento seja concluído nesses estágios será colocado em um *buffer*. Quando um item for solicitado no estágio seguinte este item deverá ser retirado do *buffer*.

Para garantir que a produção seja constante podemos iniciar o *buffer* com uma quantidade suficiente de itens. Suponha um estágio com n trabalhadores em paralelo, com tempos de execução $T_1, T_2, \dots, T_n$, respectivamente. Seja T o tempo de execução deste estágio, dado por:

$$\frac{1}{T} = \sum_{i=1}^n \frac{1}{T_i}$$

Seja j o trabalhador mais lento deste estágio, isto é, $T_j = \max_{i \in [1:n]} T_i$. Observe que em T_j unidades de tempo o estágio processa $T_j/T > 1$ itens. Podemos adicionar ao *buffer* um número de itens suficientes para compensar a diferença nos tempos de execução entre o trabalhador j e este estágio, isto é, $dT_j = T \left(\frac{T_j}{T} - 1 \right)$ itens. Com isto, ao final de cada ciclo de T unidades de tempo, o estágio posterior da linha de produção terá sempre itens disponíveis no *buffer* até que os trabalhadores iniciem a produção. Portanto, podemos assumir que produção da linha é constante se iniciarmos com um *buffer* contendo $dT_j = T \left(\frac{T_j}{T} - 1 \right)$ itens.

Esta fórmula estabelece um limite superior para o número de itens necessários no *buffer*. Na prática o número de itens no *buffer* pode ser menor. Por exemplo, sejam **A** e **B** dois trabalhadores, cujos tempos de execução para um certo conjunto de tarefas são respectivamente 10s e 20s. O tempo de execução T de um estágio com esses trabalhadores em paralelo corresponde a 6,7s. Assuma que este estágio é o gargalo da linha. A Figura 6 ilustra o funcionamento deste estágio.

Figura 6: Exemplo de execução para o ALWABP com estações em paralelo

Pela fórmula apresentada anteriormente precisaríamos de $d \frac{20}{1 - \frac{1}{10 + \frac{1}{20}}} = d 20 \left(\frac{1}{10} + \frac{1}{20} \right) = 3$ itens no *buffer*. Em $t = 6,7$ s, o primeiro item é solicitado pelo estágio seguinte. Porém, os trabalhadores **A** e **B** ainda não terminaram de processar nenhum item, logo

este item é retirado do *buffer*. Em $t = 10s$ o trabalhador **A** termina de processar seu primeiro item e o coloca no *buffer*. Em $t = 13;3s$ o estágio seguinte remove um dos itens do *buffer*. Por fim, em $t = 20s$ ambos os trabalhadores terminam um item cada. Um destes itens é recolhido para o estágio seguinte. Como podemos ver, o *buffer* sempre se manteve com pelo menos um item. Desta forma, bastaria apenas um item no *buffer* para manter a produção constante, ao invés de dois. Na Seção 3.2.1 apresentamos um modelo linear para o ALWABP com estações em paralelo enquanto que na Seção 3.2.2 apresentamos uma heurística baseada em busca em vizinhança.

3.2.1 Formulação para o ALWABP com estações em paralelo

A seguir, apresentamos uma formulação para o ALWABP com estações em paralelo descrita por Araújo et al. (2010a). Os autores utilizaram variáveis z_{sk} para indicar o nível de paralelismo de cada estágio, permitindo que o valor de k varie de 0 até K_{Max} . Esta modificação será útil para a linearização do modelo.

Com a adição das variáveis:

- t_{is} Variável binária. Igual a 1 se a tarefa i é executada no estágio s , igual a 0 caso contrário
- z_{sk} Variável binária. Igual a 1 se o estágio s possui k estações em paralelo, igual a 0 caso contrário

Pode-se escrever um modelo para o ALWABP com estações paralelas como:

$$\text{Min } C \quad (3.1)$$

s.a.

$$\sum_{s \in S} st_{is} - \sum_{s \in S} st_{js} \leq 8i; j \in N; j \geq D_j \quad (3.2)$$

$$\sum_{w \in W} x_{swi} \leq k(t_{is} + z_{sk}) \quad 8s \in S; 8k \in [0; K_{Max}]; 8i \in N \quad (3.3)$$

$$x_{swi} \leq t_{is} \quad \forall i \in S; \forall w \in W; \forall i \in N \quad (3.4)$$

$$\sum_{k=0}^{K_{\max}} z_{sk} = 1 \quad \forall s \in S \quad (3.5)$$

$$\sum_{w \in W} y_{sw} = \sum_{k=0}^{K_{\max}} k z_{sk} \quad \forall s \in S \quad (3.6)$$

$$\sum_{s \in S} y_{sw} = 1 \quad \forall w \in W \quad (3.7)$$

$$\sum_{s \in S} t_{is} = 1 \quad \forall i \in N \quad (3.8)$$

$$x_{swi} \leq y_{sw} \quad \forall i \in S; \forall w \in W; \forall i \in N \quad (3.9)$$

$$x_{swi} = 0 \quad \forall w \in W; \forall s \in S; \forall i \in I_w \quad (3.10)$$

$$\sum_{w \in W} \sum_{j: y_{sw} = 1} \frac{1}{p_{wi}} x_{swi} \leq \frac{1}{C} \quad \forall s \in S; \forall z_{s0} = 0 \quad (3.11)$$

A função objetivo (3.1) minimiza o tempo de ciclo. As restrições (3.2) definem as relações de precedência entre as tarefas. As restrições (3.3) indicam que em um estágio com k máquinas paralelas cada tarefa atribuída àquele estágio deve ser executada por pelo menos k trabalhadores. As restrições (3.4) indicam que um trabalhador só pode executar uma tarefa em um estágio caso aquela tarefa tenha sido atribuída àquele estágio. As restrições (3.5) indicam que cada estágio possui exatamente um nível de paralelismo.

As restrições (3.6) e (3.7) estabelecem que cada estágio possui k trabalhadores e cada trabalhador executa tarefas em apenas um estágio. As restrições (3.8) obrigam cada tarefa a ser executada em exatamente um estágio. As restrições (3.9) indicam que um trabalhador só pode executar tarefas em um estágio ao qual tenha sido alocado. As restrições (3.10) tratam das tarefas que alguns trabalhadores não podem executar. Por fim, as restrições (3.11) definem a taxa de produção como a menor entre as frequências de execução de cada estágio não vazio, em que um estágio vazio possui zero máquinas ($z_{s0} = 1$). A frequência de execução de um estágio corresponde à soma das frequências de execução de cada máquina que, por sua vez, equivalem ao inverso da soma dos tempos de execução de suas tarefas. As restrições (3.11) tornam o modelo (3.1)-(3.11) não-linear.

Com o intuito de linearizar (3.1)-(3.11), definimos F como a frequência de produção da linha de montagem. Desejamos então maximizar a frequência de produção, o que é equivalente a minimizar o tempo de ciclo. Seja f_{sw} a frequência de execução do trabalhador w no estágio s . Caso o trabalhador w seja designado ao estágio s , isto é, $y_{sw} = 1$, temos que $f_{sw} = \frac{1}{\sum_{i \in N} p_{wi} x_{swi}}$, o que é equivalente a $\sum_{i \in N} p_{wi} f_{sw} x_{swi} = 1$. Caso contrário, $f_{sw} = 0$. Para tornar estas restrições lineares, definimos a variável v_{swi} igual a f_{sw} se a tarefa i é executada pelo trabalhador w no estágio s ou 0 caso contrário, ou seja, $v_{swi} = f_{sw} x_{swi}$. Seja M um limitante superior para a taxa de produção e M_w um limitante superior para a frequência de execução do trabalhador w . Estes termos podem ser calculados da seguinte forma:

$$M = \frac{|W|}{\sum_{i \in N} p_{wi}^{\max}}$$

$$M_w = \frac{1}{\min_{i \in N} p_{wi}}$$

O valor de M corresponde ao inverso de um limitante inferior para o tempo de ciclo. Este tempo de ciclo equivale a alocar $|W|$ trabalhadores em um único estágio, em que os tempos destes trabalhadores para cada tarefa i correspondem ao menor tempo de execução desta tarefa entre os trabalhadores do problema original ($p_{wi}^{\max}$). O valor de M_w para um dado trabalhador w equivale a alocar a este trabalhador apenas a tarefa que ele executa mais rapidamente.

O modelo pode então ser reescrito da seguinte forma:

$$\text{Max } F \tag{3.12}$$

$$(3.2) \quad (3.10)$$

$$\begin{aligned} & \sum_{w \in W} f_{sw} \leq F_s \quad \forall s \in S \\ & \sum_{s \in S} f_{sw} \leq M_w \quad \forall w \in W \end{aligned} \quad (3.13)$$

$$\begin{aligned} & p_{wi} v_{swi} = y_{sw} \quad \forall w \in W, s \in S, i \in N \\ & v_{swi} \leq 1 \quad \forall w \in W, s \in S, i \in N \end{aligned} \quad (3.14)$$

$$f_{sw} \leq M_w y_{sw} \quad \forall w \in W, s \in S \quad (3.15)$$

$$v_{swi} \leq f_{sw} \quad \forall w \in W, s \in S, i \in N \quad (3.16)$$

$$v_{swi} \leq N_j x_{swi} \quad \forall w \in W, s \in S, i \in N \quad (3.17)$$

A função objetivo (3.12) maximiza a frequência de produção. As restrições (3.13) definem a taxa de produção como a frequência de execução do estágio não vazio mais lento. As restrições (3.14) definem as variáveis v_{swi} como uma ponderação dos tempos de execução de cada tarefa. As restrições (3.15) forçam $f_{sw} = 0$ caso o trabalhador w não tenha sido alocado ao estágio s . As restrições (3.16) e (3.17) definem os limitantes para as variáveis v_{swi} . As restrições (3.14) e (3.16) forçam $v_{swi} = f_{sw}$ se $x_{swi} = 1$, enquanto que as restrições (3.17) forçam $v_{swi} = 0$ se $x_{swi} = 0$.

O modelo (3.12) - (3.17) se mostrou de difícil resolução, sendo capaz de resolver apenas instâncias de pequeno porte. Em vista disto, optamos pelo uso de métodos heurísticos para a resolução do ALWABP com estações em paralelo, que serão descritos na seção a seguir.

3.2.2 Heurística baseada em busca em vizinhança

Para resolver o ALWABP com estações em paralelo, desenvolvemos uma heurística baseada em busca em vizinhança. Neste método, a partir de uma solução inicial, procuramos na vizinhança por soluções com melhor valor da função objetivo, ou seja, com taxa de produção maior. O método é então executado novamente a partir da nova solução. Este processo se repete até que um critério de parada seja atendido. Na seção 3.2.2.1 descrevemos a heurística construtiva utilizada para gerar a solução inicial do método, enquanto que na seção 3.2.2.2 apresentamos o método propriamente dito.

3.2.2.1 Geração da solução inicial

Para gerar a solução inicial utilizamos as heurísticas construtivas propostas por Araújo et al. (2012). Estas heurísticas são inicializadas com um limitante inferior para o tempo de ciclo e tentam gerar uma solução para o ALWABP com estações em paralelo utilizando diversos critérios para escolha de trabalhadores e tarefas. Caso encontre uma solução viável, esta é retornada pelo método. Caso contrário, o método incrementa o tempo de ciclo e tenta gerar uma nova solução. Este processo continua até que o método encontre uma solução viável ou atinja um limitante superior para o tempo de execução. O Algoritmo 1 explica a geração de uma solução para um dado valor do tempo de ciclo $\bar{C}_{\text{ini}}$ um número máximo de estações em paralelo K_{Max} .

Este algoritmo inicia com uma estação vazia e verifica quais tarefas cada combinação possível de trabalhadores pode executar (Passo 5). Para isto, o método utiliza uma série de critérios de prioridades para tarefas e trabalhadores. Seja c_i o custo de adicionar a tarefa i ao conjunto de tarefas atual, a ser executado pelo conjunto de trabalhadores atual e t_i o menor tempo de execução da tarefa i entre todos os trabalhadores. Este custo corresponde ao aumento do tempo de execução do estágio causado pela adição desta tarefa. Os autores utilizaram os seguintes critérios de prioridades para as tarefas:

1. MinD: $c_i \cdot \frac{t_i}{t_{\text{ini}}}$;
2. MinR: $\frac{c_i}{t_i}$;
3. MaxFTime: valores decrescentes da razão entre o número de seguidores e o custo da tarefa, dados por $\frac{j+1-2N_j}{c_i}$;
4. MaxIFTime: valores decrescentes da razão entre o número de seguidores imediatos e o custo da tarefa, dados por $\frac{j+1-2N_j}{c_i}$;

Algoritmo 1 Procedimento de designação orientado a estações para o ALWABP-1 com estações em paralelo

Entrada: Tempo de ciclo máximo $C_{\max}$, Número máximo de estações em paralelo $K_{\max}$

- 1: Inicie em uma estação vazia $n = 0$, $t(S_n) = 0$.
- 2: Adicione os trabalhadores não alocados a U_w e W , adicione as tarefas não alocadas a U e N .
- 3: **para** cada número de trabalhadores k variando de 1 até $K_{\max}$ **faça**
- 4: **para** cada subconjunto $W^0 \subseteq U_w$ contendo k trabalhadores **faça**
- 5: Calcule T_W^0 , o conjunto de tarefas que poderiam ser executadas pelos trabalhadores em W^0 de acordo com a regra de prioridade utilizada.
- 6: **fim**
- 7: **fim**
- 8: Aloque ao estágio k o conjunto de trabalhadores W^0 que possuir a maior prioridade e suas tarefas T_W^0 . $U_w = U_w \setminus W^0$, $U = U \cup T_W^0$, $n = n + 1$.
- 9: **se** $U_w = \emptyset$; **então**
- 10: Vá para o Passo 2.
- 11: **senão**
- 12: **se** $U = \emptyset$; **então**
- 13: Retorne a solução.
- 14: **senão**
- 15: Retorne "solução não encontrada para o tempo de ciclo $C_{\max}$ ".
- 16: **fim**
- 17: **fim**

5. $\text{MaxPWS}_P^{\min}$: valores decrescentes dos *positional weights* estáticos mínimos, dados por $C_i + \min_{j \in I_2N} w_{2W}(t_{wj})$;
6. MaxPWS_P^+ : valores decrescentes dos *positional weights* estáticos máximos, dados por $C_i + \max_{j \in I_2N} w_{2W}(t_{wj})$;
7. $\text{MaxPWD}_P^{\min}$: valores decrescentes dos *positional weights* dinâmicos mínimos, dados por $C_i + \min_{j \in I_2N} w_{2W^0}(t_{wj})$, em que W^0 corresponde aos trabalhadores não alocados
8. MaxPWD_P^+ : valores decrescentes dos *positional weights* dinâmicos máximos, dados por $C_i + \max_{j \in I_2N} w_{2W^0}(t_{wj})$;

O custo de uma tarefa corresponde ao incremento no tempo de execução do estágio caso esta tarefa seja alocada aos trabalhadores atuais. Para um trabalhador isto corresponde ao tempo de execução da tarefa, enquanto que para mais de um trabalhador é necessário calcular o novo tempo de execução do estágio e compará-lo ao valor original. O menor tempo de uma tarefa se refere aos tempos de execução entre todos os trabalhadores não alocados. Por fim, o *positional weight* de uma tarefa é dado pela soma do

tempo de execução desta tarefa para o trabalhador atual e o somatório dos tempos de execução das tarefas sucessoras. O *positional weight* dinâmico difere do estático por levar em consideração os trabalhadores alocados, o que aumenta o custo computacional da heurística.

Para selecionar os trabalhadores (Passo 8 do Algoritmo 1) os autores utilizaram os seguintes critérios de prioridades que buscam limitar o uso de paralelismo, em que n corresponde ao número de tarefas alocadas e W^0 é o conjunto de trabalhadores a ser considerado. Em todos os casos baseados em MaxTasks, os critérios priorizam conjuntos de trabalhadores que obtenham os maiores valores, enquanto que o MinRLB seleciona o conjunto com menor valor:

1. MaxTasks: n ;
2. MaxTasks1.2: se $jW^0 = 1$ então n , senão $\frac{n}{1;2}$;
3. MaxTasks1.4: se $jW^0 = 1$ então n , senão $\frac{n}{1;4}$;
4. MaxTasks1.6: se $jW^0 = 1$ então n , senão $\frac{n}{1;6}$;
5. MaxTasks1.8: se $jW^0 = 1$ então n , senão $\frac{n}{1;8}$;
6. MaxTasks2: se $jW^0 = 1$ então n , senão $\frac{n}{2}$;
7. MaxTasks3: se $jW^0 = 1$ então n , senão $\frac{n}{3}$;
8. MaxTasksK: $\frac{n}{jW^0}$;
9. MinRLB: $\frac{P}{i2N \ominus \min_{w2W} \omega^{twi}}$, em que N^0 e W^0 correspondem aos conjuntos das tarefas e dos trabalhadores não alocados, respectivamente;

Os critérios MaxTasks selecionam o trabalhador capaz de alocar o maior número de tarefas. Uma vez que o uso de estações em paralelo implica num número maior de tarefas por estágio os autores aplicam uma penalidade para casos com dois ou mais trabalhadores. Já o critério MinRLB calcula um limitante inferior para o tempo de ciclo das estações restantes. Observe que para casos com mais de um trabalhador em um mesmo estágio o denominador desta fórmula é maior. Por outro lado, quanto maior o número de trabalhadores maior o número de tarefas que podem ser executadas para o mesmo valor do tempo de ciclo. Em contrapartida, uma vez que todos os trabalhadores devem ser capazes de executar as tarefas do estágio, a presença de tarefas ineficazes podem limitar as tarefas

disponíveis para todos os trabalhadores. Para gerar uma solução para o ALWABP com estações em paralelo os autores realizam chamadas ao Algoritmo 1 para diversos valores do tempo de ciclo, conforme o Algoritmo 2.

Algoritmo 2 Procedimento de designação orientado a estações para o SALBP-2

Entrada: Tempo de ciclo mínimo LC , tempo de ciclo máximo UC

```

1: Inicie com  $LC$ 
2: Execute o Algoritmo 1 utilizando o valor de  $LC_{atual}$ 
3: se Uma solução válida foi encontrada então
4:   Retorne a solução
5: senão
6:   se  $LC = UC$  então
7:      $LC = LC + \text{passo}$ 
8:     Vá para o passo 2
9:   senão
10:    Retorne "Solução não encontrada"
11:  fim
12: fim

```

Este algoritmo é iniciado com um limitante inferior do tempo de ciclo e faz chamadas ao Algoritmo 1, incrementando o tempo de ciclo caso necessário (Passo 7), até encontrar uma solução viável ou atingir o limitante superior fornecido como entrada. Um limitante inferior para o tempo de ciclo pode ser dado pela seguinte fórmula:

$$LC = \max_i \max_{j \in N} t_{ij}^P, d_{i \in N}^P t_{ij}^P \sum_{j \in N} w_j e_j$$

O termo $\max_{i \in N} t_{ij}^P$ indica que o tempo de ciclo é maior que a tarefa mais demorada (quando executada pelo trabalhador mais rápido para ela), enquanto que o termo $d_{i \in N}^P t_{ij}^P \sum_{j \in N} w_j e_j$ corresponde a um limitante relaxado para o tempo de ciclo, em que cada tarefa é alocada ao trabalhador que a executa mais rapidamente.

Os autores propuseram o tamanho do passo de uma unidade. Embora este valor seja adequado para instâncias presentes na literatura, dependendo da magnitude dos tempos de ciclo o método pode realizar diversas chamadas ao Algoritmo 1 com valores de tempo de ciclo que retornam a mesma solução. Idealmente o tamanho do passo deve ser suficiente grande para que o algoritmo selecione uma tarefa ou trabalhador que não o fez na execução anterior, porém suficientemente pequeno para não descartar possíveis soluções viáveis.

Em vista disto, propomos alterar o tamanho do passo dinamicamente. No Passo 5 do Algoritmo 1 adicionamos tantas tarefas quantas forem possíveis para cada trabalhador, respeitando o tempo de ciclo desejado. O tamanho do passo corresponde ao menor incremento no tempo de ciclo necessário para que algum trabalhador possa alocar uma nova tarefa. Isto inclui as alocações de trabalhadores e tarefas que não fazem parte da solução.

Na seção a seguir apresentamos um método baseado em busca em vizinhança que utiliza esta heurística construtiva como solução inicial.

3.2.2.2 Busca em vizinhança

Nesta seção descrevemos um método baseado em busca em vizinhança para resolver o ALWABP com estações em paralelo. Neste método, iniciamos com uma vizinhança e efetuamos buscas locais até encontrar um ponto de mínimo local. A partir daí, o método expande a vizinhança, buscando englobar novas soluções para tentar escapar do mínimo local. Para este método, definimos seis vizinhanças que fixam tarefas e trabalhadores de alguns estágios enquanto liberam outros, permitindo que estes se movam para outros estágios livremente:

1. Liberar as tarefas do estágio mais carregado;
2. Liberar as tarefas do estágio mais carregado e dos estágios vizinhos deste;
3. Liberar as tarefas e os trabalhadores do estágio mais carregado;
4. Liberar as tarefas de todos os estágios mais carregados;
5. Liberar as tarefas de todos os estágios mais carregados e das estações vizinhas destas;
6. Liberar as tarefas e os trabalhadores de todos os estágios mais carregados;

Visando aumentar a eficiência da linha, todas as vizinhanças focam no estágio gargalo, isto é, aquele com maior tempo de execução. A primeira vizinhança permite apenas a realocação de tarefas deste estágio para os demais. A segunda vizinhança libera também tarefas de estágios vizinhos, o que permite a troca de tarefas entre estágios. Já a terceira vizinhança permite a realocação das tarefas e dos trabalhadores do estágio mais carregado. Esta vizinhança pode causar grandes alterações na solução, por permitir adicionar ou remover trabalhadores em paralelo. As três últimas vizinhanças são análogas às primeiras, porém lidam também com casos em que há mais de um estágio gargalo.

Para encontrar a melhor solução na vizinhança escolhida resolvemos o modelo descrito na Seção 3.2.1 fixando as variáveis $\mathbf{x}$, $\mathbf{y}$ e $\mathbf{t}$, com exceção daquelas correspondentes aos trabalhadores e tarefas liberados pela vizinhança. Apesar do modelo ser pouco eficiente para resolver o problema básico, ele é capaz de encontrar uma solução sub-ótima rapidamente caso algumas variáveis estejam fixas. De fato, verificou-se que as variáveis associadas a

quais trabalhadores estão alocados em cada estágio possuem um grande impacto no tempo de resolução do modelo. O Algoritmo 3 descreve o funcionamento deste método.

Algoritmo 3 Método baseado em busca em vizinhança para o ALWABP com estações em paralelo

Entrada: Solução inicial s

```

1: Inicie com uma vizinhança  $v$ 
2: Faça  $v_{\text{ultimo}} = v$ ,  $s_{\text{melhor}} = s$ 
3: repetir
4:   repetir
5:     Determine de forma exata  $s_{\text{vizinho}}$ , a melhor solução vizinha de  $s_{\text{melhor}}$  na vizinhança  $v$ 
6:     se Tempo de ciclo de  $s_{\text{vizinho}} <$  Tempo de ciclo de  $s_{\text{melhor}}$  então
7:       Faça  $s_{\text{melhor}} = s_{\text{vizinho}}$ ,  $v_{\text{ultimo}} = v$ 
8:     fim
9:   até Tempo de ciclo de  $s_{\text{vizinho}} \geq$  Tempo de ciclo de  $s$ 
10:  Seleccione uma nova vizinhança  $v$ 
11: até  $v_{\text{ultimo}} = v$ 
12: Retorne  $s_{\text{melhor}}$ 

```

Este algoritmo itera sobre três das vizinhanças citadas, buscando a solução com o menor tempo de ciclo (Passo 6). Existem duas versões do método: a primeira utiliza as vizinhanças 1, 2 e 3, enquanto que a segunda utiliza as vizinhanças 4, 5 e 6. Em ambos os casos o método inicia com a vizinhança mais simples (1 ou 4) e executa diversas buscas locais até encontrar um ponto de mínimo local. O método então seleciona uma nova vizinhança, seguindo a ordem 1 ! 2 ! 3 ! 1 ou 4 ! 5 ! 6 ! 4. Observe que no caso das vizinhanças maiores é possível que o método não encontre a solução ótima do subproblema descrito no passo 5 dentro do limite de tempo estipulado para este subproblema. Neste caso, o método pode continuar a partir de uma solução subótima, o que justifica o retorno à vizinhança menor. Este processo se repete até que o método percorra todas as vizinhanças sem encontrar uma nova solução. O método então retorna a solução atual. Na seção a seguir apresentamos os resultados obtidos para este método.

3.2.2.3 Resultados

Para testar a heurística híbrida descrita na Seção 3.2.2 utilizamos 360 instâncias definidas por Chaves et al. (2007), divididas em 4 famílias de 80 instâncias cada. Além disto, utilizamos 160 instâncias definidas por Araújo et al. (2010a). Para gerar a solução inicial utilizamos a heurística construtiva descrita na Seção 3.2.2.1, com todas as combinações de critérios de seleção trabalhadores e tarefas.

As tabelas a seguir apresentam de forma resumida os resultados obtidos. Os valores apresentados referem-se às médias obtidas para todas as instâncias testadas para o grupo de instâncias de Chaves et al. (2007). O tempo de execução da heurística híbrida foi limitado em 1800 segundos para cada critério. Para cada critério, apresentamos a distância média entre as soluções e as melhores soluções conhecidas para o ALWABP serial (coluna C), o percentual de soluções utilizando paralelismo (coluna P) e o tempo médio de execução da heurística em segundos (coluna T). Os resultados estão agrupados por critério utilizado. As tabelas 3 e 4 contém os resultados obtidos pela heurística híbrida utilizando $K_{\text{Max}} = 2$ e $K_{\text{Max}} = 3$, respectivamente, após a aplicação dos critérios 4, 5 e 6, conforme o Algoritmo 3. Resultados detalhados destes critérios podem ser vistos no Apêndice A.

A heurística foi capaz de encontrar uma solução melhor ou igual às melhores soluções seriais conhecidas para a maioria das instâncias de pequeno e médio porte. Para instâncias de grande porte, em três instâncias a heurística foi capaz de encontrar soluções melhores que as melhores soluções conhecidas para o problema serial. O tempo de execução, em média, foi relativamente baixo. Porém, assim como aconteceu com cada critério isoladamente, em alguns casos um mesmo critério pode ser aplicado várias vezes sobre uma solução, o que resultou em um tempo computacional elevado, em alguns casos chegando ao limite de meia hora de execução.

Um dos problemas das instâncias de Chaves et al. (2007) é que elas possuem pouca variabilidade em relação ao grafo de precedência. Todas as 80 instâncias de uma mesma família possuem o mesmo grafo de precedência, o que pode levar a casos em que os resultados encontrados nestas instâncias não condigam com casos mais gerais. Em vista disto, criamos novas instâncias com diversos tipos de grafos de precedência, a fim de melhor verificar a aplicabilidade dos métodos apresentados. As instâncias geradas utilizam como base um conjunto de instâncias disponíveis na literatura para o SALBP, definidas por Otto et al. (2011). Os autores geraram instâncias que diferem em diversas características, como o tipo de grafo (gargalo, cadeia e misto), *order strength* e a variabilidade entre os tempos das tarefas.

A partir de cada uma destas instâncias, geramos novas instâncias para o problema ALWABP da seguinte forma: o número de trabalhadores corresponde ao limitante superior para o número de estações do problema original. Os tempos das tarefas para cada trabalhador foram gerados aleatoriamente, variando de t até $2t$ ou de t até $5t$, em que t corresponde ao tempo da tarefa no problema original. Aplicamos também um percentual

Tabela 3: Resultados para o método baseado em busca em vizinhança com $K_{max} = 2$

	MaxTasks			MaxTasks:2			MaxTasks:4			MaxTasks:6			MaxTasks:8			MaxTasks2			MaxTasks3			MaxTasksK			MinRLB		
	C	P	T	C	P	T	C	P	T	C	P	T	C	P	T	C	P	T	C	P	T	C	P	T	C	P	T
MinD	30.53%	83.33%	0.21	23.35%	72.29%	0.21	17.86%	58.54%	0.25	15.41%	44.79%	0.25	14.36%	33.54%	0.26	13.76%	16.46%	0.26	13.65%	8.13%	0.27	13.93%	10.21%	0.27	9.48%	38.75%	0.26
MinR	30.00%	85.00%	0.17	22.15%	76.07%	0.19	16.59%	61.43%	0.17	14.29%	45.54%	0.24	13.21%	32.14%	0.25	12.88%	17.50%	0.78	12.72%	10.36%	0.83	12.88%	12.32%	0.80	9.29%	35.36%	0.24
MaxFTime	36.17%	84.29%	0.17	27.52%	74.64%	0.49	22.47%	61.79%	0.19	20.82%	46.43%	0.32	19.67%	35.00%	0.37	19.24%	20.00%	0.37	19.47%	11.79%	0.32	19.50%	13.57%	0.32	13.00%	38.21%	0.69
MaxFTime	36.22%	84.46%	0.36	27.90%	75.71%	0.37	21.88%	63.75%	0.39	19.55%	47.68%	0.25	18.63%	34.82%	0.23	18.71%	18.39%	0.26	18.43%	9.64%	0.33	18.48%	11.79%	0.33	11.86%	37.86%	2.67
MaxPWS+	32.66%	84.64%	0.15	26.82%	77.50%	0.23	22.38%	68.57%	0.24	19.13%	52.86%	0.34	17.51%	37.14%	0.58	16.33%	20.18%	0.42	15.69%	6.43%	0.48	16.60%	14.11%	0.45	11.61%	32.14%	0.81
MaxPWS+	32.31%	81.96%	0.17	28.14%	75.89%	0.22	22.80%	67.86%	0.35	20.30%	52.86%	0.22	17.97%	37.50%	0.35	16.66%	23.57%	0.34	15.96%	5.54%	0.34	16.69%	16.43%	0.34	13.01%	34.82%	0.94
MaxPWD+	28.55%	85.36%	0.18	22.20%	77.68%	0.21	17.61%	64.46%	0.48	14.77%	51.43%	1.11	13.59%	37.68%	1.10	12.91%	20.18%	3.11	11.85%	8.57%	3.13	12.35%	13.21%	3.10	8.70%	33.57%	3.52
MaxPWD+	27.27%	83.04%	0.35	21.58%	76.79%	0.48	17.08%	64.64%	0.55	14.03%	53.21%	0.60	12.63%	40.00%	0.84	11.88%	21.43%	2.70	11.58%	6.61%	2.81	11.70%	14.64%	2.70	8.20%	32.68%	1.78

Tabela 4: Resultados para o método baseado em busca em vizinhança com $K_{max} = 3$

	MaxTasks			MaxTaskst.2			MaxTaskst.4			MaxTaskst.6			MaxTaskst.8			MaxTaskst.2			MaxTaskst.3			MaxTaskst.K			MinRLB		
	C	P	T	C	P	T	C	P	T	C	P	T	C	P	T	C	P	T	C	P	T	C	P	T	C	P	T
MinD	35.66%	84.82%	0.24	28.07%	76.07%	2.08	22.78%	62.50%	0.28	19.20%	53.04%	1.66	17.08%	41.79%	1.65	14.40%	25.36%	2.31	13.38%	9.29%	0.94	13.28%	10.54%	0.94	9.66%	40.36%	0.29
MinR	36.15%	84.29%	0.21	28.99%	77.50%	0.21	22.96%	64.11%	0.25	18.62%	49.82%	0.33	16.75%	40.71%	0.57	14.74%	24.82%	1.44	13.17%	11.61%	0.77	13.03%	12.14%	0.75	9.23%	36.43%	0.25
MaxFTime	41.29%	83.57%	0.20	34.04%	76.07%	0.19	28.82%	64.29%	0.33	24.37%	51.61%	0.35	21.98%	42.14%	0.34	20.67%	27.32%	0.47	19.58%	12.68%	0.27	19.31%	14.64%	0.27	13.09%	39.29%	1.68
MaxFTime	40.91%	83.39%	0.20	33.55%	77.68%	0.25	27.68%	64.82%	0.38	23.77%	52.68%	0.87	21.76%	41.79%	2.37	20.37%	24.64%	0.49	19.37%	10.71%	0.39	19.01%	11.61%	0.38	11.60%	39.11%	4.23
MaxPWS-	38.04%	83.21%	0.23	33.31%	78.39%	0.63	28.85%	70.54%	0.68	24.20%	59.82%	0.92	21.20%	46.43%	0.88	19.02%	29.29%	0.55	15.98%	6.61%	0.70	16.23%	13.57%	0.70	11.58%	32.14%	3.60
MaxPWS+	37.81%	82.32%	0.25	33.61%	77.50%	0.65	29.80%	71.25%	0.70	25.67%	62.32%	0.34	22.96%	49.29%	2.35	19.40%	32.68%	2.49	15.78%	8.39%	0.55	16.31%	17.50%	0.67	13.03%	33.93%	3.93
MaxPWD-	35.84%	85.18%	0.23	30.32%	78.75%	0.31	24.98%	68.39%	0.32	20.32%	58.04%	0.91	17.40%	46.25%	0.43	15.69%	27.14%	1.99	12.75%	10.36%	2.21	12.75%	12.86%	3.78	8.58%	35.18%	5.62
MaxPWD+	34.80%	85.00%	0.26	29.21%	79.64%	0.44	24.33%	69.82%	0.34	19.87%	58.21%	0.98	16.13%	47.68%	0.65	13.85%	27.32%	0.72	11.61%	9.46%	5.16	11.61%	15.71%	3.19	8.09%	33.04%	3.75

de tarefas incompatíveis de 10% e 20%. Desta forma, cada instância original gerou 4 novas instâncias para o problema ALWABP. Chamamos estas instâncias de família Araújo.

As seções a seguir detalham os resultados obtidos para estas instâncias usando a heurística baseada em busca em vizinhança para o ALWABP com estações em paralelo e para o método exato e a heurística para o ALWABP com múltiplas linhas. Não foram feitos testes com os demais métodos porque os mesmos se mostraram ineficientes mesmo para as instâncias originais.

A tabela 5 apresenta de forma resumida os resultados obtidos para a heurística descrita na seção 3 com as novas instâncias. As colunas C indica o tempo de ciclo médio das soluções obtidas em comparação com o tempo de ciclo médio das melhores soluções conhecidas. Estes valores foram gentilmente fornecidos por Marcus Ritt, professor da Universidade Federal do Rio Grande do Sul. A coluna P indica o percentual de soluções utilizando paralelismo. Por fim, a coluna T apresenta o tempo médio de execução da heurística, em segundos.

O critério MinRLB encontrou as melhores soluções, em média, especialmente quando combinado com os critérios MinR, MinD e MaxPWD-. A heurística conseguiu encontrar soluções cujos tempos de ciclo são menores que as respectivas soluções ótimas seriais em 16,44% das instâncias, com uma melhoria média de 4,76%. A maior melhoria encontrada foi de 14,80%.

A tabela 6 apresenta os melhores resultados obtidos pela heurística de acordo com o tipo de instância. A nova coluna Melhor C indica a maior melhoria entre as soluções encontradas para cada tipo de instância.

Por fim, a tabela 7 apresenta os resultados agrupados por tipo de instância, considerando apenas as melhorias. A coluna P+ indica o percentual de instâncias que apresentaram melhoria em relação à instância original, enquanto que a coluna C+ indica a melhoria média.

Instâncias com grafos do tipo misto apresentaram o maior número de soluções utilizando paralelismo. Para instâncias com taxa de infactibilidade alta, os resultados encontrados para os três tipos de grafos foram similares para os casos com fator de variação de tempo alto, enquanto que, entre as instâncias com fator de variação de tempo baixo, aquelas com grafos do tipo gargalo obtiveram melhores resultados. De modo geral, a heurística teve dificuldades para encontrar boas soluções para instâncias com grafos do tipo cadeia. Porém, é importante observar que uma solução com este tipo de grafo apresentou

Tabela 5: Resultados para o método baseado em busca em vizinhança com $K_{Max} = 2$ para a família Araújo

	MaxTasks			MaxTasks1.2			MaxTasks1.4			MaxTasks1.6			MaxTasks1.8			MaxTasks2			MaxTasks3			MaxTasksK			MinRLB		
	C	P	T	C	P	T	C	P	T	C	P	T	C	P	T	C	P	T	C	P	T	C	P	T	C	P	T
MinD	14.99%	75.75%	124.97	13.36%	71.19%	117.36	10.43%	60.96%	112.69	7.28%	50.83%	99.45	6.75%	45.16%	94.98	6.75%	31.59%	86.70	7.86%	12.24%	81.56	7.96%	11.90%	82.88	2.43%	38.04%	89.72
MinR	15.09%	75.75%	117.53	13.59%	71.75%	115.92	10.63%	61.74%	114.65	7.21%	51.61%	97.27	6.72%	45.38%	94.84	6.38%	31.81%	87.45	7.44%	15.02%	84.17	7.56%	14.13%	84.93	2.23%	37.15%	90.27
MaxFTime	20.22%	75.75%	118.99	18.98%	73.08%	116.65	16.56%	66.52%	112.42	13.24%	58.51%	98.73	12.76%	51.17%	94.56	12.42%	37.82%	92.31	13.71%	14.13%	85.63	13.82%	15.57%	85.78	6.30%	41.27%	95.85
MaxFTime	20.21%	76.08%	111.79	18.83%	73.08%	111.41	16.43%	65.63%	105.62	13.23%	57.84%	114.63	12.70%	49.61%	111.23	12.33%	36.26%	110.62	13.48%	14.35%	105.15	13.71%	15.24%	105.17	6.22%	40.82%	100.65
MaxPVS+	21.44%	75.75%	124.91	20.07%	73.97%	125.29	18.44%	69.86%	126.53	15.12%	63.63%	108.99	14.18%	54.84%	101.33	13.38%	40.16%	100.13	14.60%	17.02%	98.87	14.74%	19.24%	98.25	7.41%	42.38%	108.35
MaxPVS-	21.86%	75.86%	122.08	20.60%	73.86%	123.37	18.74%	70.30%	125.08	15.66%	64.29%	104.98	14.78%	58.62%	102.39	14.12%	45.61%	97.41	15.39%	17.15%	90.70	15.59%	20.91%	92.29	7.41%	42.38%	109.61
MaxPVD+	17.43%	74.08%	116.91	15.51%	71.86%	118.65	12.94%	65.41%	113.03	9.47%	57.84%	99.41	8.43%	47.72%	100.18	7.86%	35.01%	95.67	8.81%	12.35%	93.63	8.96%	13.24%	92.29	3.23%	34.59%	99.08
MaxPVD-	16.88%	74.30%	125.17	14.72%	71.08%	122.18	12.01%	65.29%	139.93	8.38%	58.51%	124.28	7.61%	48.16%	99.01	7.08%	36.48%	97.06	7.94%	13.68%	91.39	8.11%	14.13%	91.00	2.86%	34.48%	95.70

Fator de tempo	2							
Infactibilidade	10%				20%			
	C	T	P	Melhor C	C	T	P	Melhor C
Gargalo	-0,77%	123,50	57,11%	-11,57%	-0,92%	89,19	42,81%	-12,69%
Cadeia	-0,49%	135,66	55,20%	-14,80%	0,07%	92,61	43,65%	-11,35%
Misto	-0,89%	176,38	59,14%	-11,49%	-0,51%	92,49	45,89%	-11,26%

Fator de tempo	5							
Infactibilidade	10%				20%			
	C	T	P	Melhor C	C	T	P	Melhor C
Gargalo	0,74%	97,71	47,67%	-9,28%	0,78%	77,49	37,06%	-7,40%
Cadeia	1,87%	99,64	48,22%	-4,85%	1,38%	79,92	39,04%	-9,81%
Misto	1,20%	106,88	51,44%	-4,69%	0,97%	78,74	40,39%	-11,63%

Tabela 6: Resultados para a heurística baseada em busca em vizinhança com $K_{\text{Max}} = 2$ para a família Araújo, classificado por tipo de instância

Fator de tempo	2			
Infactibilidade	10%		20%	
	P+	C+	P+	C+
Gargalo	30,67%	-1,30%	24,00%	-1,49%
Cadeia	24,00%	-1,20%	20,00%	-0,88%
Misto	26,32%	-1,48%	23,68%	-1,18%

Fator de tempo	5			
Infactibilidade	10%		20%	
	P+	C+	P+	C+
Gargalo	8,00%	-0,42%	8,00%	-0,34%
Cadeia	4,00%	-0,10%	8,00%	-0,36%
Misto	10,53%	-0,21%	9,21%	-0,37%

Tabela 7: Resultados para a heurística para o ALWABP com estações em paralelo com $K_{\text{Max}} = 2$ para a família Araújo - melhorias

a maior melhoria encontrada pela heurística (14,80%).

3.3 ALWABP com múltiplas linhas

Conforme dito anteriormente, o uso de múltiplas linhas permite expandir o espaço de busca, possibilitando configurações que não são possíveis para o caso serial. A figura 7 apresenta um exemplo de solução para o ALWABP utilizando duas linhas. Os trabalhadores W1 e W3 foram alocados à primeira linha, enquanto que os trabalhadores W2 e W4 foram alocados à segunda linha. O tempo de ciclo dessas linhas corresponde a 21 s e 18 s, respectivamente. Portanto, o tempo de ciclo desse sistema corresponde a $\frac{1}{\frac{1}{21} + \frac{1}{18}}$ 10; 24s.

Observe que, ao alocarmos os trabalhadores a linhas independentes, aumentamos o número de tarefas que cada trabalhador deve executar em média. Embora este fenômeno

Figura 7: Exemplo de execução para o ALWABP com linhas em paralelo

também ocorra no caso com estações em paralelo, ele está limitado aos trabalhadores em estágios com mais de uma estação, enquanto que para o caso com múltiplas linhas ele se aplica a todo o sistema. Com isto, a presença de tarefas que alguns trabalhadores não podem executar ou o fazem de forma ineficiente se torna uma complicação ainda maior do que nos casos serial e com estações em paralelo. Por outro lado, esta pode ser uma característica desejável, em especial nos casos em que desejamos uma configuração sub-ótima da linha. De fato, esta característica será de grande utilidade para o problema de rotação de tarefas, conforme será visto no Capítulo 4. Na Seção 3.3.1 apresentamos um modelo linear para o ALWABP com múltiplas linhas. Nas Seções 3.3.2 e 3.3.3 apresentamos respectivamente um método exato e uma heurística para o mesmo problema.

3.3.1 Formulação para o ALWABP com múltiplas linhas

A seguir, apresentamos o modelo desenvolvido para o ALWABP com múltiplas linhas. Da mesma forma que o modelo para o ALWABP com estações em paralelo (visto na seção 3.2.1), este modelo requer a linearização das restrições que calculam o tempo de ciclo. Desta forma, este modelo se baseia fortemente no modelo para o ALWABP com estações em paralelo.

Usando a notação:

$i; j$	Índices das tarefas
w	Índices dos trabalhadores
s	Índices dos estágios
k	Índice das linhas em paralelo
N	Conjunto de tarefas
W	Conjunto de trabalhadores
S	Conjunto de estágios
p_{wi}	Tempo de execução da tarefa i pelo trabalhador w
D_j	Conjunto de tarefas imediatamente predecessoras à tarefa j no grafo de precedência
I_w	Conjunto de tarefas que o trabalhador w não é capaz de executar
K_{Max}	Número máximo de linhas em paralelo

E as variáveis:

x_{swik}	Variável binária. Igual a 1 se a tarefa i é executada pelo trabalhador w no estágio s da linha k , igual a 0 caso contrário
y_{swk}	Variável binária. Igual a 1 se o trabalhador w está no estágio s da linha k , igual a 0 caso contrário
z_k	Variável binária. Igual a 1 se a linha k é utilizada, igual a 0 caso contrário

Podemos escrever um modelo para o ALWABP com linhas em paralelo conforme segue:

$$\text{Max} \sum_{k=1}^{K_{Max}} F_k \quad (3.18)$$

$$\sum_{w \in W} \sum_{s \in S} x_{swik} = z_k \quad \forall i \in N; \forall k \in [1; K_{Max}] \quad (3.19)$$

$$\sum_{s \in S} \sum_{k=1}^{K_{Max}} y_{swp} = 1 \quad \forall w \in W \quad (3.20)$$

$$\sum_{w \in W} y_{swk} \leq z_k \quad \forall s \in S; \forall k \in [1; K_{Max}] \quad (3.21)$$

$$\sum_{w \in W} \sum_{s \in S} x_{swip} = \sum_{w \in W} \sum_{s \in S} x_{swjk} \quad \forall i; j \in N; i \neq j; k \in [1; K_{\max}] \quad (3.22)$$

$$\sum_{i \in N} x_{swik} \leq M y_{swk} \quad \forall w \in W; s \in S; k \in [1; K_{\max}] \quad (3.23)$$

$$x_{swik} = 0 \quad \forall w \in W; s \in S; i \in I_w; k \in [1; K_{\max}] \quad (3.24)$$

$$\sum_{w \in W} f_{swk} \leq F_p \leq M (1 - \sum_{w \in W} y_{swk}) \quad \forall s \in S; k \in [1; K_{\max}] \quad (3.25)$$

$$\sum_{i \in N} p_{wi} v_{swik} = y_{swk} \quad \forall w \in W; s \in S; k \in [1; K_{\max}] \quad (3.26)$$

$$f_{swk} \leq M y_{swk} \quad \forall w \in W; s \in S; k \in [1; K_{\max}] \quad (3.27)$$

$$v_{swik} \leq f_{swk} \leq M (1 - x_{swik}) \quad \forall s \in S; w \in W; i \in N; k \in [1; K_{\max}] \quad (3.28)$$

$$v_{swik} \leq M x_{swik} \quad \forall s \in S; w \in W; i \in N; k \in [1; K_{\max}] \quad (3.29)$$

A função objetivo (3.18) maximiza a frequência do sistema, que corresponde à soma das frequências de cada linha de produção. As restrições (3.19) garante que em cada linha de produção ativa (isto é, uma linha com pelo menos um trabalhador) todas as tarefas serão executadas por exatamente um trabalhador em um estágio. As restrições 3.20 garantem que cada trabalhador será alocado a uma estação de trabalho. Observe que um trabalhador pode ser alocado a apenas uma linha de produção. As restrições (3.21) permitem apenas um trabalhador em cada estação de cada linha ativa e zero em linhas

inativas. As restrições (3.22) estabelecem as relações de precedência entre as estações de cada linha. As restrições (3.23) estabelecem que um trabalhador só pode executar tarefas em um estágio de uma linha caso este trabalhador esteja alocado a esta linha. As restrições (3.24) evitam as tarefas que os trabalhadores não podem executar. As restrições (3.25)-(3.29) definem a frequência de execução do sistema como a soma da frequência de execução das linhas, de forma análoga ao modelo descrito na seção 3.2.1. As restrições (3.25) definem a taxa de produção de cada linha ativa como a frequência de execução do estágio não vazio mais lento. As restrições (3.26) definem as variáveis v_{swik} como uma ponderação dos tempos de execução de cada tarefa. As restrições (3.29) forçam $f_{swk} = 0$ caso o trabalhador w não tenha sido alocado ao estágio s da linha k . As restrições (3.27) e (3.28) definem os limitantes para as variáveis v_{swik} . As restrições (3.26) e (3.27) forçam $v_{swik} = f_{swk}$ se $x_{swik} = 1$, enquanto que as restrições (3.28) forçam $v_{swik} = 0$ se $x_{swik} = 0$.

3.3.1.1 Resultados

Este modelo foi implementado em C++ e utilizado para resolver 40 instâncias da família Roszieg e 40 da família Heskia. As tabelas 8 e 9 contém os resultados obtidos para as famílias Roszieg e Heskia, respectivamente.

Tabela 8: Resultados do modelo para a família Roszieg

Grupo	Tempo de ciclo	Tempo de Execução	Melhoria
1	20,10	0,07	0,00%
2	31,50	0,06	0,00%
3	28,10	0,06	0,00%
4	28,00	0,05	0,00%

Tabela 9: Resultados do modelo para a família Heskia

Grupo	Tempo de ciclo	Tempo de Execução	Melhoria
1	102,30	1,09	0,00%
2	122,60	1,09	0,00%
3	172,50	0,62	0,00%
4	171,20	0,59	0,00%

O modelo não foi capaz de encontrar soluções melhores que as melhores soluções seriais conhecidas nas instâncias testadas. Este resultado era esperado, visto que estas instâncias possuem apenas quatro trabalhadores, o que torna difícil o uso de múltiplas linhas. Para instâncias maiores, o modelo chegou a rodar por várias horas sem encontrar uma solução. Isto levou à necessidade de desenvolver novas estratégias de resolução. Uma vez que este problema pode ser decomposto em um conjunto de subproblemas ALWABP em que cada

subproblema corresponde a uma das linhas, desenvolvemos um método alternativo para a resolução do mesmo, que será visto na seção a seguir.

3.3.2 Método exato

Nesta seção descrevemos um método que resolve o ALWABP com múltiplas linhas. Uma solução para este problema pode ser vista como um conjunto de soluções para o ALWABP, em que cada uma destas soluções contém uma partição do conjunto dos trabalhadores. Em vista disto, podemos resolver o ALWABP com múltiplas linhas resolvendo os subproblemas definidos por cada partição e combinando os resultados. Por exemplo, seja $W = \{w_1; w_2; w_3\}$ um conjunto de trabalhadores. Para este problema temos sete subproblemas: $\{w_1\}$, $\{w_2\}$, $\{w_3\}$, $\{w_1; w_2\}$, $\{w_1; w_3\}$, $\{w_2; w_3\}$ e $\{w_1; w_2; w_3\}$. Os resultados são então combinados para formar soluções para o problema original, em que cada solução de uma partição corresponde a uma das linhas de produção. Para este exemplo temos cinco possíveis soluções, definidas pelos problemas $\{w_1\}$, $\{w_2\}$, $\{w_3\}$, $\{w_1; w_2\}$ e $\{w_1; w_2; w_3\}$.

Utilizamos dois critérios de corte para reduzir o número de subproblemas a serem resolvidos. O primeiro critério evita a solução de subproblemas que não resultarão em soluções viáveis. Após a resolução de um subproblema, caso verifiquemos que este não possui solução viável, podemos descartar o subproblema complementar deste, ou seja, aquele contendo todos os trabalhadores que não pertencem a este subproblema. No exemplo acima se o subproblema $\{w_2\}$ não possui solução viável então não é necessário resolver o subproblema $\{w_1; w_3\}$.

O segundo critério limita superiormente o tempo de ciclo. Inicialmente o método resolve diversos problemas ALWABP seriais com um número de estágios menor que o número de trabalhadores. Isto é feito resolvendo o modelo descrito na seção 2.2.2, relaxando-se a restrição 2.7 e variando o número de estágios $|S|$, de tal forma que $1 \leq |S| < |W|$. Esta restrição indica que todos os trabalhadores devem ser alocados. Portanto ao relaxarmos esta restrição temos um limitante inferior para o tempo de ciclo para cada possível tamanho dos subproblemas. Assim, ao resolvermos o subproblema para um dado número de trabalhadores, podemos determinar se é possível obter uma solução melhor que a melhor solução encontrada pelo método até o momento. Por exemplo, em um problema com 4 trabalhadores, suponhamos que os tempos de ciclo das soluções ótimas para 1, 2 e 3 estações correspondem a 25 s, 20 s e 18 s, respectivamente. Além disso suponha que a melhor solução encontrada até o momento possua tempo de ciclo de 10 s. Ao resolvermos

um subproblema contendo um trabalhador, encontramos uma solução com tempo de ciclo de 40s. Observe que se combinarmos a solução ótima deste subproblema com a melhor solução possível para 3 trabalhadores, teremos um tempo de ciclo de $\frac{1}{\frac{1}{40} + \frac{1}{18}}$ [12] 12; 41s. Isto quer dizer que não é possível obter uma solução com tempo de ciclo menor que o melhor conhecido a partir deste subproblema e, portanto, podemos ignorar os demais subproblemas com 3 trabalhadores. Por outro lado, em uma configuração contendo um trabalhador em duas linhas e dois trabalhadores numa terceira linha possui como tempo de ciclo mínimo corresponde a $\frac{1}{\frac{1}{40} + \frac{1}{25} + \frac{1}{20}}$ [8] 8; 7s, então é necessário resolver os demais subproblemas para esta configuração.

3.3.2.1 Resultados

A seguir, apresentamos os resultados obtidos pelo método descrito na seção 3.3.2, utilizando 80 instâncias da família Roszieg (tabela 10) e 80 da família Heskia (tabela 11). Os resultados estão divididos por grupo de instâncias. Apresentamos o tempo de ciclo médio das soluções, o tempo de execução médio do método em segundos e a melhoria média do tempo de ciclo em relação às soluções ótimas do ALWABP serial.

Tabela 10: Resultados do método exato para a família Roszieg

Grupo	Tempo de ciclo	Tempo de Execução	Melhoria
1	20,1	1,13	0,00%
2	31,5	0,84	0,00%
3	28,1	1,82	0,00%
4	28	1,41	0,00%
5	9,7	104,32	0,00%
6	10,91	76,57	0,86%
7	16	127,32	0,00%
8	15,07	111,05	0,17%

Tabela 11: Resultados do método exato para a família Heskia

Grupo	Tempo de ciclo	Tempo de Execução	Melhoria
1	102,3	1,41	0,00%
2	122,6	0,96	0,00%
3	172,5	1,76	0,00%
4	171,2	1,67	0,00%
5	34,71	194,45	0,49%
6	41,64	216,08	1,91%
7	69,41	219,39	5,33%
8	65,77	240,57	1,65%

Para as instâncias testadas, o método é consideravelmente mais rápido que a resolução

utilizando um pacote comercial aplicado ao modelo para o problema ALWABP com linhas em paralelo, nunca ultrapassando 10 minutos de execução mesmo em instâncias dos grupos 5 a 8, para as quais o solver utilizado (CPLEX) não foi capaz de resolver após uma hora de execução. Isto permitiu resolver de forma exata instâncias de médio porte das famílias Roszieg e Heskia. Com o aumento do número de trabalhadores, o uso de linhas em paralelo conseguiu encontrar soluções melhores para estas as instâncias.

Para a família Roszieg, três instâncias apresentaram melhoria em relação à melhor solução conhecida para o problema serial, com uma melhoria máxima de 8.09%. Já para a família Heskia, em 12 instâncias a solução ótima encontrada era melhor que a solução ótima do problema serial. Quatro destas instâncias apresentaram melhoria superior a 10% e a maior melhoria encontrada corresponde a 22,44%.

A tabela 12 apresenta os resultados obtidos para as instâncias da família Araújo utilizando tempos das tarefas variando de t até $2t$ e 10% de incompatibilidade. Os resultados estão agrupados por número de trabalhadores das instâncias. Cada linha indica os resultados para um tipo de grafo: gargalo, cadeia ou misto. Para cada tipo de grafo e número de trabalhadores indicamos o número de instâncias (coluna N), o número de instâncias que apresentaram melhoria quando comparadas com as soluções ótimas das respectivas instâncias para o problema serial (coluna N+), o tempo médio de resolução de cada instância, em segundos (coluna T) e a melhoria média entre as soluções que apresentaram alguma melhoria.

Estes resultados indicam que problemas com grafos mistos são mais propícios a terem boas soluções utilizando linhas em paralelo. Estes resultados também ilustram o aumento do tempo de execução do método quando aumentamos o número de trabalhadores. Instâncias com 7 e 8 trabalhadores requereram várias horas de execução para encontrar uma solução ótima.

O método não foi capaz de resolver instâncias de grande porte mesmo após várias horas de execução, o que motivou o desenvolvimento de métodos mais eficientes.

3.3.3 Heurística para o ALWABP com múltiplas linhas

Nesta seção apresentamos uma heurística de quatro fases para o ALWABP com múltiplas linhas. Esta heurística utiliza uma ideia similar ao método apresentado na Seção 3.3.2, isto é, gerar partições do conjunto de trabalhadores, utilizar um método para resolver os subproblemas resultantes e combiná-los para encontrar uma solução para o pro-

Tabela 12: Resultados do método exato para o ALWABP com múltiplas linhas para a família Araújo

		Número de trabalhadores																												
		4						5						6						7						8				
		3												5																
		N	N +	T	Melhoria	N	N +	T	Melhoria	N	N +	T	Melhoria	N	N +	T	Melhoria	N	N +	T	Melhoria	N	N +	T	Melhoria	N	N +	T	Melhoria	
Gargalo	39	0	2,29	0,00%	11	0	20,65	0,00%	28	2	139,93	0,45%	10	2	1276,53	1,02%	2	10714,47	0,28%	0	0	0,00	-%							
Cadeia	47	0	2,15	0,00%	12	0	20,94	0,00%	24	2	155,59	0,75%	15	4	1518,18	1,01%	2	9879,65	0,00%	0	0	0,00	-%							
Misto	70	5	1,55	2,63%	15	0	20,57	0,00%	37	6	126,68	2,18%	25	14	1042,73	2,59%	2	14774,19	3,72%	1	1	36395,80	6,27%							

blema. Porém, em vez de enumerar todas as possíveis partições de trabalhadores, esta heurística utiliza um procedimento para identificar partições que potencialmente possuam soluções viáveis. Além disso, utilizamos uma heurística para resolver os subproblemas, visto que métodos exatos podem resultar em tempos de execução impraticáveis para os subproblemas para instâncias de médio e grande porte. Isto se tornou evidente com os resultados vistos na tabela 12, em que os tempos de execução para instâncias com oito trabalhadores chegaram a ultrapassar dez horas de execução.

Esta heurística está dividida em quatro fases, conforme a figura 8. Na primeira fase são gerados conjuntos de tarefas que cada trabalhador é capaz de executar. Na segunda fase, estes conjuntos são combinados para formar partições de trabalhadores que potencialmente possuam uma solução viável. Na terceira fase, as partições de trabalhadores são avaliadas utilizando uma heurística para o ALWABP serial (Moreira et al.; 2012). Por fim, na quarta fase, a heurística seleciona K_{Max} partições que irão compor a solução final do método. As seções a seguir detalham cada uma destas fases.

Figura 8: Visão geral da heurística para o ALWABP com múltiplas linhas

3.3.3.1 Geração de conjuntos de tarefas

Na primeira fase, a heurística gera conjuntos de tarefas que podem ser executadas simultaneamente por um trabalhador em uma estação. Estes conjuntos contêm apenas tarefas que o trabalhador é capaz de executar e devem obedecer as relações de precedencia, podendo supor que algumas tarefas foram executadas por outro trabalhador. Usamos a notação $i < j$ para indicar que i precede j no grafo de precedencia. Supondo que $i < k$ e $k < j$, se i e j pertencem a um conjunto válido de tarefas, k também deve pertencer a

este conjunto. Seja x_i uma variável binária igual a 1 se i pertence ao conjunto atual, as seguintes condições representam as restrições supracitadas:

$$x_i = 0; \quad \forall i \in I_w; \quad (3.30)$$

$$x_i + x_j \leq 1 + x_k; \quad \forall i, j, k \in N \mid w_j < k; k < j; \quad (3.31)$$

As restrições (3.30) garantem que o conjunto contém apenas tarefas factíveis para o trabalhador w . As restrições (3.31) garantem que, se duas tarefas i e j pertencem ao conjunto, todas as tarefas k entre elas também devem pertencer ao conjunto.

A fim de evitar redundância, desejamos que cada conjunto contenha o maior número de tarefas possível. Assim, para gerar um conjunto de tarefas, resolvemos o seguinte problema:

$$\text{Max} \sum_i x_i \quad (3.32)$$

sujeito a (3.30) e (3.31).

Para gerar todos os conjuntos de tarefas, podemos resolver o problema diversas vezes, em cada execução forçando a adição de uma nova tarefa. Para isto, adicionamos a restrição $x_i = 1$, para alguma tarefa i que não pertença aos conjuntos gerados até então. Isto é feito até que a heurística tenha gerado conjuntos abrangendo todas as tarefas. Esta estratégia, no entanto, possui dois problemas. Em primeiro lugar, ela pode gerar cada conjunto múltiplas vezes, uma vez que um conjunto pode ser máximo para mais de uma tarefa. Além disso, podem ocorrer casos com simetria, isto é, casos em que uma mesma tarefa pertence a mais de um conjunto máximo. Por exemplo, suponha um conjunto de quatro tarefas x_1, x_2, x_3, x_4 , em que $x_1 < x_2 < x_3$ e x_4 não depende das demais. Suponha também que o trabalhador atual não consegue executar a tarefa x_2 . Existem dois conjuntos máximos que contém a tarefa x_4 : $\{x_1, x_4\}$ e $\{x_3, x_4\}$. Neste caso, o método deve gerar ambos os conjuntos de tarefas.

Observe que, ao relaxarmos as restrições referentes ao tempo de ciclo, as tarefas que podem ser alocadas a um estágio estão limitadas apenas pelas tarefas infactíveis para o trabalhador alocado a este estágio e pelas relações de precedencias. Considere por exemplo o grafo de precedencia da Figura 9. A tarefa 0 é uma tarefa artificial que será usada para reduzir o número de falsos positivos (isto é, configurações que não possuam

soluções viáveis) em etapas posteriores do método. Suponha que a tarefa 1 é infactível para o trabalhador atual. Para gerar os conjuntos máximos devemos considerar dois casos: se esta tarefa está alocada a um estágio anterior ou posterior na linha de produção. No primeiro caso a tarefa 0 também deve ser alocada a um estágio anterior de modo a respeitar as relações de precedencia. Logo um conjunto máximo corresponde a todas as tarefas exceto 0 e 1. Por outro lado, caso a tarefa 1 seja alocada a um estágio posterior então as tarefas 2 e 5 também devem ser alocadas a outro estágio. Com isto, o segundo conjunto máximo para este trabalhador contém todas as tarefas exceto 1, 2 e 5.

Figura 9: Exemplo de grafo de precedencia

Utilizando está lógica, desenvolvemos o Algoritmo 4 para gerar os conjuntos de tarefa para um dado trabalhador w . Este algoritmo percorre todos os subconjuntos do conjunto de tarefas infactíveis para o trabalhador w e calcula os conjuntos máximos, supondo que as tarefas infactíveis do subconjunto atual foram alocadas a um estágio anterior.

Algoritmo 4 Geração dos conjuntos de tarefas

Entrada: $w \in W$

- 1: $T_w = \{ \}$;
 - 2: **para todo** $I \subseteq I_w$ **faça**
 - 3: Determine s o conjunto de todas as tarefas i que não sejam predecessoras das tarefas em I e que não sejam sucessoras das tarefas em $I_w \setminus I$
 - 4: **se** $s \neq \{ \}$ **então**
 - 5: $T_w = T_w \cup s$
 - 6: **fim**
 - 7: **fim**
 - 8: Retorne T_w
-

Este algoritmo considera todos os subconjuntos do conjunto de tarefas infactíveis para o trabalhador w . Para cada subconjunto I , o algoritmo supõe que as tarefas de I foram alocadas a alguma estação e, no passo 3, verifica quais tarefas podem ser alocadas. É importante ressaltar que no passo 2 devemos também gerar o conjunto máximo para $I = \{ \}$, o que simula o caso em que alocamos o trabalhador ao primeiro estágio da linha.

Com o aumento do número de tarefas inactiváveis, o número de subconjuntos de I_w pode se tornar muito grande. Idealmente, ao garantirmos que cada trabalhador executa apenas tarefas de um conjunto e que os conjuntos cobrem todas as tarefas, gostaríamos também de garantir a existencia de uma solução viável para o problema. Isto nem sempre ocorre, uma vez que os conjuntos de tarefas inactiváveis podem ser diferentes para cada trabalhador. Considere novamente o grafo da figura 9 e dois trabalhadores w_1 e w_2 . O primeiro não consegue executar as tarefas 3 e 7, enquanto que o segundo não consegue executar a tarefa 1. Considere os conjuntos máximos $f\ 1; 2; 4; 5; 8g$ para w_1 e para $f\ 2; 3; 4; 5; 6; 7; 8g$ para w_2 . Embora estes conjuntos contenham todas as tarefas eles representam uma configuração inactivável da linha, uma vez que o primeiro supõe que as tarefas 3 e 7 já foram alocadas, enquanto que o segundo o faz para a tarefa 1. Isto gera um conflito entre as precedencias das tarefas.

Para amenizar este problema utilizamos duas estratégias: a primeira remove do grafo todas as tarefas inactiváveis sem sucessores, enquanto que a segunda faz com que o algoritmo considere apenas subconjuntos de tarefas inactiváveis que sejam suficientes para adicionar uma nova tarefa. O exemplo a seguir ilustra estas estratégias.

Exemplo: Considere o grafo de precedencia da Figura 9. Neste grafo, adicionamos uma tarefa artificial 0 que precede todas as tarefas sem predecessoras do grafo. Em outras palavras, um conjunto de tarefas pode ser alocado à primeira estação da linha se, e somente se, ele contiver a tarefa artificial. Esta tarefa pertence apenas ao conjunto máximo gerado a partir de $I = ;$. Considere o trabalhador w , com $I_w = f\ 5; 7g$.

Os subconjuntos de I_w são $f;$; $f\ 5g$; $f\ 7g$; $f\ 5; 7gg$. Para $I = ;$, $s = f\ 0; 1; 2; 3; 4; 6g$. Observe que para $I = f\ 5g$, $s = f\ 6g$ é um subconjunto do conjunto anterior. Isto ocorre porque a tarefa 5 não possui sucessoras e, portanto, não impede a execução de nenhuma tarefa. Assim, para gerar os subconjuntos de um trabalhador, podemos remover do grafo as tarefas inactiváveis para este trabalhador que não possuam sucessoras. Com isto, eliminamos também $I = f\ 5; 7g$. Para $I = f\ 7g$, $s = f\ 1; 2; 3; 4; 8g$. A Figura 10 mostra os dois conjuntos de tarefa para o trabalhador w , referenciados como $S_{w,1}$ e $S_{w,2}$, respectivamente.

A próxima fase do método consiste em combinar estes conjuntos de tarefas de modo a encontrar partições de trabalhadores que potencialmente levem a soluções viáveis.

Figura 10: Subconjuntos de tarefa para o trabalhador w

3.3.3.2 Geração dos conjuntos de trabalhadores

O objetivo desta fase é gerar conjuntos de trabalhadores de tal forma que cada trabalhador contribui com um conjunto de tarefas e cada tarefa pertence a pelo menos um conjunto. Com isto, desejamos garantir a existencia de uma solução viável para o ALWABP contendo estes trabalhadores. Para cada trabalhador w , sejam T_w os conjuntos de tarefas gerados pelo Algoritmo 4 e a_{wti} um parâmetro igual a 1 se a tarefa i pertence ao conjunto t do trabalhador w . Usando as variáveis y_{wt} igual a 1 se o trabalhador w executa as tarefas do conjunto t , um conjunto viável de trabalhadores deve obedecer as seguintes restrições

$$\sum_{t=1}^T y_{wt} \leq 1; \quad \forall w \in W \quad (3.33)$$

$$\sum_{w \in W} \sum_{t=1}^T a_{wti} y_{wt} \geq 1; \quad \forall i \in N: \quad (3.34)$$

As restrições (3.33) indicam que um trabalhador não pode executar tarefas de mais de um conjunto de tarefas, caso este trabalhador seja selecionado. As restrições (3.34) exigem que cada tarefa pertença a pelo menos um conjunto selecionado.

Observe que, caso um conjunto W_i de trabalhadores seja suficiente para formar uma linha de produção, então o conjunto $W_i \cup \{w\}$ também forma uma linha de produção. Desta forma, estamos interessados em conjuntos com o menor número possível de trabalhadores, isto é, desejamos resolver o problema:

$$\text{Min} \sum_{w \in W} \sum_{t=1}^T x_{wt} y_{wt} \quad (3.35)$$

sujeito a (3.33) e (3.34).

Podemos obter um conjunto viável de trabalhadores a partir da solução acima, bastando escolhermos os trabalhadores para os quais $\sum_{t=1}^T y_{wt} = 1$. Para gerar um novo conjunto de trabalhadores, basta eliminar a restrição $\sum_{t=1}^T y_{w_1t} + \sum_{t=1}^T y_{w_2t} + \dots + \sum_{t=1}^T y_{w_nt} \leq n$, em que $w_1, w_2, \dots, w_n$ são os trabalhadores da solução atual. Como alternativa, propomos o Algoritmo 5 para gerar os conjuntos de trabalhadores.

Algoritmo 5 Geração de conjuntos de trabalhadores

Entrada: T_w

```

1:  $W_r = ;$ ,  $L_c = ;$ ,  $L_w = ;$ 
2:  $i = 0$ 
3: enquanto  $|W_r| < |L_c|$  faca
4:    $L_c = L_c \cup \{w\}$ 
5:    $L_w = L_w \cup \{w\}$ 
6:   se  $(|L_c| = |L_w|)$  então
7:      $W_r = W_r \cup \{w\}$ 
8:     Remova o último elemento de  $L_w$ 
9:     Remova o último elemento de  $L_c$ 
10:  fim
11:  Selecione  $i \in \{1, \dots, |L_c|\}$ 
12:  fim
13: se  $\text{set\_list} \neq \emptyset$  então
14:   Remova o último elemento de  $L_w$ 
15:   Remova o último elemento de  $L_c$ 
16:   Vá para 3
17: senão
18:   Retorne  $W_r$ 
19: fim

```

Este algoritmo inicia com uma lista vazia de conjuntos de tarefas. A cada iteração, se existe uma tarefa que não pertence a nenhum conjunto da lista, o algoritmo seleciona um conjunto de um trabalhador não alocado que contenha a tarefa (passo 3). O trabalhador e o conjunto são adicionados às respectivas listas (passos 4 e 5). Se todas as tarefas pertencerem a pelo menos um conjunto, adicionamos o subproblema à lista de subproblemas a serem resolvidos na próxima fase (passo 7). Observe que, se o subproblema contendo os trabalhadores $\{w_1, w_2, \dots, w_n\}$ é válido, então o subproblema contendo os trabalhadores $\{w_1, w_2, \dots, w_n, w_i\}$ também é válido. Em vista disto, este algoritmo gera apenas subproblemas de tamanho mínimo. Se não houverem mais tarefas a serem alocadas ou se não houverem mais trabalhadores capazes de executar a tarefa selecionada, o método seleciona

um novo conjunto de tarefas do último trabalhador da lista. Se não restarem mais conjuntos deste trabalhador para serem testados, o mesmo é removido da lista. Este processo é repetido até que a lista de conjuntos esteja vazia (passo 18).

Exemplo: Para o grafo da figura 9, a primeira tarefa não alocada é $i = 0$. Começamos com $w = w_1$ e $s = S_{w_1;1} = \{0; 1; 3; 6; 7; 8\}g$. Uma vez que $i \notin s$, $L_w = \{w_1\}g$ e $L_c = \{S_{w_1;1}\}g$. Então fazemos $w = w_2$ e selecionamos o primeiro subconjunto de w_2 , isto é, $s = S_{w_2;1} = \{1; 2; 3; 4; 6\}g$. A próxima tarefa não executada é $i = 2$. Logo, $L_w = \{w_1; w_2\}g$ e $L_c = \{S_{w_1;1}; S_{w_2;1}\}g$. Como não há mais trabalhadores para executar a tarefa 5, nós removemos w_2 de L_w e $S_{w_2;1}$ de L_c e buscamos um novo conjunto que contenha a tarefa 2. Neste caso, $s = S_{w_2;2} = \{1; 2; 3; 4; 8\}g$, então $L_w = \{w_1; w_2\}g$ e $L_c = \{S_{w_1;1}; S_{w_2;2}\}g$. Mais uma vez não há trabalhadores capazes de executar a tarefa 5, então nós removemos w_2 e $S_{w_2;2}$ das respectivas listas. Como não há mais conjuntos para $w = w_2$, removemos também w_1 e $S_{w_1;1}$ das suas listas e selecionamos um novo conjunto que contenha a tarefa 0. Neste caso, o único subconjunto restante é $s = S_{w_2;1} = \{0; 1; 2; 3; 4; 6\}g$. Logo, $L_w = \{w_2\}g$ e $L_c = \{S_{w_2;1}\}g$. Então, selecionamos a próxima tarefa não executada $i = 5$. O único subconjunto de um trabalhador não alocado capaz de executar esta tarefa é $s = S_{w_1;2} = \{5; 6; 7; 8\}g$. Portanto, $L_w = \{w_2; w_1\}g$ e $L_c = \{S_{w_2;1}; S_{w_1;2}\}g$. Uma vez que todas as tarefas foram executadas, este conjunto é adicionado à lista de subproblemas válidos. As demais iterações do algoritmo irão tornar vazias as listas L_w e L_c . A próxima fase do algoritmo irá avaliar estes subproblemas para encontrar soluções parciais.

3.3.3.3 Avaliação dos conjuntos

As duas primeiras fases do método geraram diversos subproblemas, em que cada subproblema corresponde a um problema ALWABP serial com uma partição do conjunto de trabalhadores. A solução de cada um destes subproblemas é uma solução parcial do ALWABP com múltiplas linhas. Nesta fase, as soluções parciais são resolvidas a fim de obter o tempo de ciclo de cada possível configuração da linha de produção. Para resolver estes subproblemas, podemos usar o modelo proposto por Miralles et al. (2007), conforme visto na seção 2.2.2. Esta abordagem pode se mostrar inviável devido ao grande número de subproblemas a serem resolvidos.

Alternativamente, podemos usar a heurística construtiva proposta por Moreira et al. (2012). Esta heurística calcula um limitante inferior para o tempo de ciclo e tenta construir uma solução para o ALWABP serial com este tempo de ciclo. Se não

for possível encontrar uma solução, o tempo de ciclo é incrementado em uma unidade e o procedimento é repetido até que uma solução seja encontrada ou um limite superior para o tempo de ciclo seja atingido. Uma adaptação desta heurística pode ser vista na Seção 3.2.2.1. Utilizamos esta adaptação para resolver os subproblemas, uma vez que esta heurística é capaz de resolver o ALWABP serial bastando para isto utilizarmos $K_{Max} = 1$.

3.3.3.4 Seleção de conjuntos

Na etapa anterior, a heurística gerou diversas soluções parciais associadas a conjuntos de trabalhadores. A última etapa da heurística irá combinar estas soluções parciais para formar soluções do ALWABP com múltiplas linhas. Seja $W_R = \{W_1; W_2; \dots; W_{|W_R|}\}$ o conjunto de todos os conjuntos viáveis de trabalhadores e C_r o tempo de ciclo associado à linha contendo os trabalhadores W_r . Seja q_{wr} um parâmetro igual a 1 se o trabalhador w pertence ao conjunto W_r e e_r uma variável igual a 1 se o subconjunto W_r for selecionado. O problema de encontrar uma solução para o ALWABP com múltiplas linhas corresponde a escolher conjuntos de W_R tais que cada trabalhador deve pertencer a exatamente um conjunto. Uma vez que desejamos encontrar a solução com o menor tempo de ciclo, este problema pode ser escrito como:

$$\text{Max} \sum_{r \in W_R} \frac{1}{C_r} e_r \quad (3.36)$$

sujeito a:

$$\sum_{r=1}^{|W_R|} q_{wr} e_r = 1; \quad \forall w \in W \quad (3.37)$$

Para o caso em que $K_{Max} = 2$, para cada conjunto de trabalhadores $W_i \in W_R$, basta verificar se existe $W_j \in W_R$ tal que $W_i \cap W_j = \emptyset$. Sejam C_i e C_j os tempos de ciclo dos subproblemas formados pelos trabalhadores W_i e W_j , respectivamente. O tempo de ciclo da solução com duas linhas formadas por estes conjuntos de trabalhadores é dado por $C = \frac{1}{\frac{1}{C_1} + \frac{1}{C_2}} = \frac{C_1 C_2}{C_1 + C_2}$.

Para $K_{Max} \in \mathbb{N}$ precisamos encontrar combinações de conjuntos de trabalhadores em que cada trabalhador pertence a exatamente um conjunto. O Algoritmo 6 enumera as possíveis combinações de conjuntos de trabalhadores. Usamos a notação $[W_i]$ para indicar

a união de todos os conjuntos de W_i .

Algoritmo 6 Seleção de conjuntos

Entrada: W_r

```

1: solucoes = ;
2: para todo  $W_i \in W_r$  faça
3:   se [  $W_i = W$  então
4:     se  $W_1 \setminus W_2 = ; \ 8W_1; W_2 \subseteq W_i$  então
5:       solucoes = solucoes [  $f W_i$ g
6:     fim
7:   fim
8: fim
9: Retorne solucoes

```

Conforme dito anteriormente, se um problema com k trabalhadores possui uma solução válida, o problema contendo estes mesmos trabalhadores acrescido de um novo trabalhador também possui solução válida. Isto implica que, se um conjunto de trabalhadores não possui solução, qualquer subconjunto deste conjunto de trabalhadores também não possui solução. Isto permite reduzir o número de subconjuntos a serem testados no passo 2. A última etapa da heurística escolhe a solução com menor tempo de ciclo entre todas as soluções encontradas.

3.3.3.5 Resultados

As tabelas 13 e 14 apresentam os resultados obtidos pela heurística para o ALWABP com múltiplas linhas com as novas instâncias usando $K_{Max} = 2$ e $K_{Max} = 3$, respectivamente. Os resultados estão agrupados por tipo de grafo, pelo fator de tempo e pela taxa de infactibilidade tarefa-trabalhador de cada instância. A coluna C indica a diferença, em média, entre as melhores soluções conhecidas para as instâncias daquele grupo e as soluções encontradas pela heurística. Neste cálculo, foram consideradas apenas instâncias em que a solução encontrada pela heurística utiliza mais de uma linha em paralelo. A coluna T indica o tempo médio de execução da heurística, em segundos. A coluna P indica o número de instâncias com soluções viáveis encontradas. Por fim, a coluna Melhor C indica a maior melhoria com relação à melhor solução conhecida, entre todas as instâncias do grupo.

As tabelas 15 e 16 apresentam os resultados para as mesmas instâncias, considerando apenas aquelas em que as soluções encontradas pela heurística são melhores que as melhores soluções conhecidas destas instâncias para o ALWABP serial, usando $K_{Max} = 2$ e $K_{Max} = 3$, respectivamente. A coluna C+ indica a melhoria média destas instâncias,

enquanto que a coluna P+ indica o percentual de instâncias em que houve melhoria.

Fator de tempo	2							
Infactibilidade	10%				20%			
	C	T	P	Melhor C	C	T	P	Melhor C
Gargalo	-0,75%	31,67	54,05%	-15,37%	-2,92%	34,18	36,49%	-14,94%
Cadeia	1,62%	23,32	66,22%	-17,56%	-0,78%	15,89	39,19%	-15,73%
Misto	-0,19%	34,25	60,81%	-15,10%	-2,36%	23,73	41,89%	-16,15%

Fator de tempo	5							
Infactibilidade	10%				20%			
	C	T	P	Melhor C	C	T	P	Melhor C
Gargalo	7,58%	41,61	54,05%	-20,95%	5,04%	42,07	36,49%	-7,31%
Cadeia	9,85%	29,95	66,22%	-15,85%	6,98%	19,33	39,19%	-11,33%
Misto	9,76%	46,69	60,81%	-10,67%	6,88%	29,43	41,89%	-9,96%

Tabela 13: Resultados para a heurística para o ALWABP com múltiplas linhas com $K_{Max} = 2$ (novas instâncias)

Fator de tempo	2							
Infactibilidade	10%				20%			
	C	T	P	Melhor C	C	T	P	Melhor C
Gargalo	-3,22%	71,25	33,78%	14,28%	-1,22%	6,57	29,73%	13,96%
Cadeia	-0,39%	71,03	37,84%	19,39%	0,23%	16,83	29,73%	15,37%
Misto	-0,75%	90,61	41,89%	13,97%	-1,20%	15,88	33,78%	14,38%

Fator de tempo	5							
Infactibilidade	10%				20%			
	C	T	P	Melhor C	C	T	P	Melhor C
Gargalo	8,98%	57,09	33,78%	14,67%	17,87%	10,87	29,73%	-3,69%
Cadeia	15,68%	71,72	37,84%	12,00%	17,24%	28,32	29,73%	1,61%
Misto	16,10%	107,40	41,89%	6,27%	16,21%	22,57	33,78%	-0,06%

Tabela 14: Resultados para a heurística para o ALWABP com múltiplas linhas com $K_{Max} = 3$ (novas instâncias)

Fator de tempo	2			
Infactibilidade	10%		20%	
	P+	C+	P+	C+
Gargalo	29,73%	-6,36%	22,97%	-6,43%
Cadeia	21,62%	-7,11%	20,27%	-5,88%
Misto	24,32%	-8,23%	22,97%	-8,19%

Fator de tempo	5			
Infactibilidade	10%		20%	
	P+	C+	P+	C+
Gargalo	13,51%	-5,79%	8,11%	-3,72%
Cadeia	6,76%	-5,84%	8,11%	-6,54%
Misto	10,81%	-2,96%	10,81%	-3,42%

Tabela 15: Resultados para a heurística para o ALWABP com múltiplas linhas com $K_{Max} = 2$ (novas instâncias) - melhorias

Fator de tempo	2			
Infactibilidade	10%		20%	
	P+	C+	P+	C+
Gargalo	24,32%	-5,72%	16,22%	-5,48%
Cadeia	18,92%	-7,01%	9,46%	-8,97%
Misto	21,62%	-8,13%	20,27%	-7,12%

Fator de tempo	5			
Infactibilidade	10%		20%	
	P+	C+	P+	C+
Gargalo	4,05%	-8,50%	0,00%	0,00%
Cadeia	2,70%	-6,90%	1,35%	-1,61%
Misto	1,35%	-6,27%	0,00%	0,00%

Tabela 16: Resultados para a heurística para o ALWABP com múltiplas linhas com $K_{\text{Max}} = 3$ (novas instâncias) - melhorias

Estes resultados mostram que, quanto maior a taxa de infactibilidade tarefa-trabalhador para uma instância, mais difícil é encontrar uma solução utilizando linhas em paralelo. No entanto, se compararmos as colunas P e P+, vemos que em muitas instâncias com baixa taxa de infactibilidade, as soluções encontradas possuem tempo de ciclo maior que o tempo de ciclo da melhor solução conhecida.

Instâncias com grafos do tipo cadeia apresentaram o maior número de soluções utilizando paralelismo. Porém, estas instâncias também apresentaram o menor número de melhorias. Em contrapartida, instâncias com grafos do tipo gargalo apresentaram o maior número de melhorias.

Embora a heurística não tenha encontrado tantas soluções utilizando linhas em paralelo quanto o método exato, ela o fez em um tempo computacional muito menor. Além disto, assim como o método exato, a heurística conseguiu identificar instâncias sem solução utilizando linhas em paralelo em menos de um segundo.

4 Rotação de tarefas

Neste capítulo, descrevemos os métodos desenvolvidos para extensões do problema de rotação de tarefas. Nos problemas anteriores desejamos obter uma configuração fixa do sistema, maximizando a produtividade total. Já na rotação de tarefas a produção é dividida em períodos. Desejamos obter uma configuração para cada período de modo a atender uma demanda. Além disso, queremos também que as configurações sejam as mais distintas possíveis, mais especificamente, que cada trabalhador execute o máximo de tarefas distintas possíveis. Na Seção 4.1 apresentamos um modelo linear e uma heurística híbrida para o problema de rotação de tarefas com estações em paralelo, enquanto que na Seção 4.2 adaptamos este modelo e este método para o caso com múltiplas linhas. Finalmente, na Seção 4.3 apresentamos modelos e métodos para uma generalização do problema de rotação de tarefas que pode utilizar ambas as extensões simultaneamente.

4.1 Rotação de tarefas com estações em paralelo

Conforme dito anteriormente, na rotação de tarefas desejamos que os trabalhadores executem conjuntos de tarefas distintos em cada período, esperando com isto que o trabalhador ganhe experiência mais rapidamente. Ao utilizarmos estações em paralelo permitimos que dois ou mais trabalhadores executem o mesmo conjunto de tarefas no mesmo período. Nas seções a seguir apresentamos um modelo linear (Seção 4.1.1) e uma heurística híbrida (Seção 4.1.2) para o problema.

4.1.1 Modelo

Nesta seção apresentamos um modelo para o problema de rotação de tarefas com estações em paralelo. O modelo é baseado naquele apresentado por Araújo et al. (2012) para o ALWABP com estações em paralelo, bem como no modelo descrito por Costa e Miralles (2009) para rotação de tarefas. É importante ressaltar que no modelo de Costa

e Miralles (2009) é imposto um limite para o tempo de ciclo médio, o que foi substituído no nosso modelo por uma taxa de produção mínima. Usando a notação:

w	Índice dos trabalhadores
$i; j$	Índices das tarefas
s	Índice das estações
r	Índice dos períodos
k	Índice do número de estações em paralelo
W	Conjunto de trabalhadores
N	Conjunto de tarefas
S	Conjunto de estações
R	Conjunto de períodos
D_j	Conjunto das tarefas predecessoras da tarefa j
I_w	Conjunto das tarefas infactíveis para o trabalhador w
$K_{\max}$	Número máximo de estações em paralelo
p_{wi}	Tempo de execução da tarefa i pelo trabalhador w
F	Número mínimo de itens a serem processados

E as variáveis:

z_{wi}	Variável binária. Igual a um se o trabalhador w executa a tarefa i em pelo menos um período
t_{isr}	Variável binária. Igual a um se a tarefa i é alocada à estação s no período r
x_{swr}	Variável binária. Igual a um se a tarefa i é alocada ao trabalhador w na estação s no período r
y_{swr}	Variável binária. Igual a um se o trabalhador w é alocado à estação w no período r
a_{skr}	Variável binária. Igual a um se o estágio s do período r possui k estações em paralelo
F_r	Taxa de produção da linha de produção no período r
f_{swr}	Taxa de produção do trabalhador w no estágio s do período r
v_{swr}	Variável auxiliar, usada na linearização das taxas de produção. Corresponde a $f_{swr} \frac{p_{wi}}{p_{wi}}$

Podemos escrever o seguinte modelo para o problema de rotação de tarefas com estações em paralelo:

$$\text{Max} \sum_{w \in W} \sum_{i \in N} z_{wi} \quad (4.1)$$

$$s.t.: \quad \sum_{i \in N} \sum_{j \in N} x_{ij} \leq D_j; 8r \leq R \quad (4.2)$$

$$\sum_{w \in W} \sum_{i \in N} x_{swir} \leq k(t_{isr} + a_{skr}) \quad 8s \leq S; 8k \leq [0; K_{\text{Max}}]; 8i \leq N; 8r \leq R \quad (4.3)$$

$$\sum_{w \in W} \sum_{i \in N} x_{swir} \leq t_{isr} \quad 8s \leq S; 8w \leq W; 8i \leq N; 8r \leq R \quad (4.4)$$

$$\sum_{k=0}^{K_{\text{Max}}} a_{skr} = 1 \quad 8s \leq S; 8r \leq R \quad (4.5)$$

$$\sum_{w \in W} \sum_{k=0}^{K_{\text{Max}}} y_{swr} = k a_{skr} \quad 8s \leq S; 8r \leq R \quad (4.6)$$

$$\sum_{w \in W} \sum_{k=0}^{K_{\text{Max}}} y_{swr} = 1 \quad 8w \leq W; 8r \leq R \quad (4.7)$$

$$\sum_{s \in S} t_{isr} = 1 \quad 8i \leq N; 8r \leq R \quad (4.8)$$

$$\sum_{s \in S} \sum_{w \in W} x_{swir} \leq y_{swr} \quad 8s \leq S; 8w \leq W; 8i \leq N; 8r \leq R \quad (4.9)$$

$$\sum_{w \in W} x_{swir} = 0 \quad 8w \leq W; 8s \leq S; 8i \leq I_w; 8r \leq R \quad (4.10)$$

$$\sum_{w \in W} \sum_{i \in N} f_{swr} \leq F_r \quad 8s \leq S; 8r \leq R \quad (4.11)$$

$$\sum_{w \in W} \sum_{i \in N} p_{wi} v_{swir} = y_{swr} \quad 8w \leq W; 8s \leq S; 8r \leq R \quad (4.12)$$

$$\sum_{i \in N} \sum_{w \in W} f_{swr} \leq M y_{swr} \quad 8w \leq W; 8s \leq S; 8r \leq R \quad (4.13)$$

$$\sum_{w \in W} \sum_{i \in N} f_{swr} \leq M (1 - x_{swir}) \quad 8s \leq S; 8w \leq W; 8i \leq N; 8r \leq R \quad (4.14)$$

$$\sum_{w \in W} \sum_{i \in N} v_{swir} \leq M x_{swir} \quad 8s \leq S; 8w \leq W; 8i \leq N; 8r \leq R \quad (4.15)$$

$$\sum_{w \in W} \sum_{i \in N} z_{wi} \leq x_{swir} \quad 8s \leq S; 8w \leq W; 8i \leq N \quad (4.16)$$

$$\sum_{r \in R} \sum_{i \in N} F_r \leq F \quad (4.17)$$

A função objetivo (4.1) maximiza o número de alocações trabalhador-tarefa distintas. As restrições (4.2) tratam das relações de precedência entre as tarefas. As restrições (4.3) - (4.15) garantem que cada período contenha uma configuração válida para a linha. As restrições (4.3) exigem que em um estágio com k estações em paralelo cada tarefa seja executada por k trabalhadores. As restrições (4.4) dizem que um trabalhador só pode executar tarefas em um estágio ao qual estas tarefas foram alocadas. As restrições (4.5) definem o nível de paralelismo de cada estágio. As restrições (4.6) e (4.7) dizem que cada

trabalhador deve ser alocado a um estágio e que cada estágio deve possuir k trabalhadores. As restrições (4.8) dizem que cada tarefa deve ser alocada a um estágio em cada período. As restrições (4.9) dizem que cada trabalhador só pode executar tarefas no estágio ao qual foi alocado. As restrições (4.10) evitam alocações de tarefas a trabalhadores que não as podem executar. As restrições (4.16) contam o número de tarefas distintas executadas por cada trabalhador. finalmente, a restrição (4.17) garante que uma quantidade mínima de itens seja processada em todos os períodos.

As restrições (4.11) - (4.15) correspondem às restrições utilizadas por Araújo et al. (2012) para linearizar o cálculo da taxa de produção. As restrições (4.11) definem a taxa de produção de uma linha inativa, isto é, uma linha sem trabalhadores ou tarefas como zero. As restrições (4.12) calculam v_{swir} como uma ponderação dos tempos de execução das tarefas. As restrições (4.13) definem que $v_{swir} = f_{swr}$ se $x_{swir} = 1$, enquanto que as restrições (4.14) garantem que $v_{swir} = 0$ caso contrário.

Este modelo se mostrou de difícil resolução, em especial devido às variáveis com quatro índices x_{swir} e v_{swir} . Testes iniciais com instâncias dos quatro primeiros grupos da família Roszieg, que são instâncias pequenas com quatro trabalhadores e 25 tarefas, não encontraram soluções viáveis após uma hora de execução. Em vista disto, este modelo foi abandonado em favor de uma heurística mais eficiente, descrita na seção a seguir.

4.1.2 Heurística híbrida

Nesta seção descrevemos a heurística híbrida para o problema de rotação de tarefas com estações em paralelo. Esta heurística é uma adaptação daquela proposta por Moreira e Costa (2012) para o problema de rotação de tarefas no caso ALWABP serial. Para resolver este problema, os autores utilizam uma heurística construtiva para gerar um *pool* de configurações da linha, ou seja, soluções para o ALWABP serial. Após uma etapa de busca local, os autores utilizam um modelo linear para encontrar a melhor combinação de configurações, que corresponde à solução retornada pelo método.

A heurística adaptada está dividida em três etapas: geração de um *pool* de configurações (isto é, soluções para o ALWABP com estações em paralelo), busca local e resolução de um modelo linear para encontrar a melhor combinação de configurações. Estas etapas são detalhadas nas seções a seguir.

4.1.2.1 Geração do pool de soluções parciais

A primeira etapa da heurística consiste na geração do *pool* de configurações. Estas configurações devem ser suficientemente próximas da solução ótima do ALWABP com estações em paralelo de modo a permitir a existência de soluções para a rotação de tarefas com limitantes apertados para o número mínimo de itens produzidos. Ao mesmo tempo, elas devem ser suficientemente distintas entre si para encontrar soluções de qualidade para este problema.

Para gerar o *pool* de configurações utilizamos as soluções encontradas pelas heurísticas construtivas propostas por Araújo et al. (2012). Estas heurísticas são descritas em detalhes na Seção 3.2.2.1. Além disto utilizamos as configurações geradas por mil iterações de uma heurística baseada em GRASP (do inglês *greedy randomized adaptative search procedure*), descrita a seguir.

A metaheurística GRASP (Feo e Resende; 1995) propõe utilizar uma heurística construtiva para gerar várias soluções para um problema. Em métodos gulosos, a cada passo selecionamos o melhor a melhor ação de acordo com um certo critério, com base nas informações atuais. Já o GRASP adiciona diversidade às soluções geradas ao mesmo tempo em que tenta manter a qualidade das soluções ao selecionar aleatoriamente uma das melhores ações disponíveis.

O método GRASP desenvolvido para o ALWABP com estações em paralelo se baseia nas heurísticas construtivas propostas por Araújo et al. (2012). No método original temos critérios de seleção de tarefas e trabalhadores. Em cada passo as tarefas são classificadas de acordo com um critério e selecionamos as melhores tarefas para cada conjunto de trabalhadores. Além disso, há um critério que determina quais trabalhadores com seus respectivos conjuntos de tarefas serão adicionados à solução. A heurística GRASP desenvolvida para este problema altera os critérios de seleção de tarefas, de modo que em cada iteração o método cria uma tabela com as melhores tarefas de acordo com o critério selecionado. Embora seja possível também alterar os critérios de seleção de trabalhadores, isto não foi feito pois estes critérios incluem mecanismos para garantir a viabilidade da solução. Por exemplo, o critério MinRLB (Moreira e Costa; 2009b) busca evitar situações em que os trabalhadores restantes não são suficientes para realizar as demais tarefas. Ao permitirmos a escolha de conjuntos de trabalhadores sub-ótimos, a heurística pode requerer iterações adicionais (com tempos de ciclo maiores) para encontrar uma solução viável.

O GRASP requer dois parâmetros adicionais: α e β . O primeiro indica a diferença máxima entre o valor de uma tarefa candidata (isto é, uma tarefa que pode ser selecionada pela heurística) e a melhor tarefa disponível de acordo com o critério de seleção de trabalhadores, enquanto que o segundo representa o número máximo de tarefas candidatas. Por exemplo, suponha que em um dado ponto de execução da heurística temos seis tarefas disponíveis, cujos valores de acordo com o critério de seleção de tarefas são 20, 25, 27, 28, 29 e 35. Suponha também que este critério seleciona a tarefa de menor valor do critério e que os parâmetros α e β são 50% e 3, respectivamente. O parâmetro α permite tarefas com valor máximo de 30 (o que exclui apenas a última tarefa), porém devido ao parâmetro β temos apenas três tarefas candidatas. Logo o método deve selecionar aleatoriamente uma das três primeiras tarefas.

Para induzir as heurísticas a obter soluções diversas, utilizamos uma tabela contendo o número de vezes que cada tarefa foi alocada a cada trabalhador. Os critérios de seleção de tarefas foram então alterados de modo a penalizar alocações tarefa-trabalhador já utilizadas em outras soluções. Assim, se c é o valor da tarefa i de acordo com o critério adotado e esta tarefa foi alocada d_{wi} vezes para o trabalhador w em outras soluções geradas pelo método, o valor modificado da tarefa corresponde a $\bar{c} = c \cdot (1 + d_{wi})$ para critérios que selecionam a tarefa de menor valor e $\bar{c} = c / (1 + d_{wi})$ caso contrário.

Outra modificação realizada no GRASP corresponde ao número de soluções parciais geradas por iteração. Em vez de retornar apenas a primeira solução parcial encontrada, o método continua por mais algumas iterações até encontrar uma nova solução, de modo a acrescentar soluções parciais subótimas ao *pool*. Isto é importante para aumentar a variedade do *pool*, pois nem sempre a solução ótima da rotação de tarefas inclui apenas soluções parciais ótimas. Além disso, elas podem permitir à busca local encontrar novas soluções parciais de boa qualidade.

4.1.2.2 Busca local

A próxima etapa da heurística consiste em realizar duas buscas locais nas soluções parciais presentes no *pool*: a primeira visa encontrar soluções de melhor qualidade (menor tempo de ciclo), enquanto a segunda visa explorar novas alocações tarefa-trabalhador. Em ambos os casos o único movimento permitido é deslocar uma tarefa para um estágio vizinho, respeitando as relações de precedência e as tarefas que cada trabalhador não pode executar. No primeiro caso, a heurística seleciona a tarefa candidata que resulte na maior redução do tempo de ciclo e encerra a busca local caso nenhuma tarefa possa ser movida

sem reduzir o tempo de ciclo. A última solução encontrada é então adicionada ao *pool*.

Já no segundo caso a heurística mantém uma tabela contendo as novas alocações tarefa-trabalhador. Uma tarefa é candidata a ser movida caso este movimento resulte em uma alocação inédita nesta busca local. No caso de múltiplas tarefas candidatas, a heurística seleciona aquela que resulta no menor tempo de ciclo (mesmo que este seja maior que o tempo de ciclo da solução original). Nesta busca local, adicionamos ao *pool* a última solução encontrada, bem como a solução de menor tempo de ciclo.

4.1.2.3 Modelo

A última etapa da heurística consiste em resolver um modelo linear para selecionar o conjunto de soluções parciais que resulte na melhor rotação de tarefas. Usando a notação:

R *Pool* de soluções

F_r Taxa de produção da solução r

a_{rwi} Variável binária. Igual a 1 se a tarefa i é alocada ao trabalhador w na solução r

F Taxa de produção mínima

E as variáveis:

x_r Número de vezes que a solução r pertence à solução

z_{wi} Variável binária. Igual a 1 se a tarefa i é alocada ao trabalhador w em ao menos um subperíodo

A última fase da heurística consiste na resolução do seguinte modelo:

$$\text{Max} \sum_{w \in W} \sum_{i \in N} z_{wi} \quad (4.18)$$

$$\text{s.a:} \sum_{r \in R} F_r x_r \geq F \quad (4.19)$$

$$\sum_{r \in R} z_{wi} a_{rwi} x_r \geq 1 \quad \forall w \in W; i \in N \quad (4.20)$$

$$\sum_{r \in R} x_r = |P| \quad (4.21)$$

A função objetivo (4.18) maximiza o número de alocações tarefa-trabalhador distintas.

A restrição (4.19) garante a produtividade mínima. As restrições (4.20) calculam o número de alocações tarefa-trabalhador. Por fim, a restrição (4.21) estabelece que devemos selecionar uma solução parcial por período.

De modo a tornar o método mais rápido eliminamos do *pool* as soluções com tempo de ciclo muito alto. Para isto, consideramos a restrição (4.19). Seja i uma solução parcial qualquer e $r_{[0,1]}$ solução parcial de maior taxa de produção presente no *pool*. Para que i pertença a uma solução viável sua taxa de produção deve ser suficiente para atender a demanda em conjunto com outras soluções parciais. Assim, considerando o caso em temos i em um período e $r_{[0,1]}$ nos demais, temos que $F_i + F_{r_{[0,1]}}(jP_j) \leq F$. Logo:

$$F_i + F_{r_{[0,1]}}(jP_j) \leq F \quad (4.22)$$

A inequação (4.22) permite eliminar do *pool* soluções de baixa qualidade sem interferir na solução ótima da rotação de tarefas.

4.2 Rotação de tarefas com múltiplas linhas

Conforme será visto na Seção 4.5 o uso de estações em paralelo se mostrou bastante benéfico para o problema de rotação de tarefas. Isto nos incentivou a estudar também este problema num contexto com múltiplas linhas. Neste caso, temos uma ou mais linhas operando independentemente em cada período, o que pode acelerar ainda mais o treinamento dos trabalhadores. A seguir, apresentamos um modelo matemático e as heurísticas desenvolvidas para este problema.

4.2.1 Modelo

A seguir apresentamos um modelo linear para o problema de rotação de tarefas com linhas em paralelo. Com os novos parâmetros:

- l Índice das linhas
- $L_{\max}$ Número máximo de linhas por período

Com as novas variáveis:

- x_{swilr} Variável binária. Igual a 1 se o trabalhador w executa a tarefa i na estação s da linha l do período r
- y_{swlr} Variável binária. Igual a 1 se o trabalhador w está alocado na estação s da linha l do período r
- f_{swlr} Taxa de produção do trabalhador w na estação s da linha l do período r
- v_{swilr} Variável auxiliar. Utilizada na linearização do modelo
- F_{lr} Taxa de produção da linha l do período r
- z_{wi} Igual a 1 se a tarefa i é alocada ao trabalhador w em ao menos um subperíodo

$$\text{Max} \sum_{w \in W} \sum_{i \in N} x_{wi} z_{wi} \quad (4.23)$$

$$s.a: \sum_{w \in W} \sum_{s \in S} x_{swilr} = 1 \quad 8i \in N; 8l \in [1; L_{max}]; 8r \in R \quad (4.24)$$

$$\sum_{s \in S} y_{swlr} = 1 \quad 8w \in W; 8l \in [1; L_{max}]; 8r \in R \quad (4.25)$$

$$\sum_{w \in W} y_{swlr} = 1 \quad 8s \in S; 8l \in [1; L_{max}]; 8r \in R \quad (4.26)$$

$$\sum_{w \in W} \sum_{s \in S} x_{swilr} = \sum_{w \in W} \sum_{s \in S} x_{swjlr} \quad 8i, j \in N; 8j \in D_j; 8l \in [1; L_{max}]; 8r \in R \quad (4.27)$$

$$\sum_{i \in N} x_{swilr} = M y_{swlr} \quad 8w \in W; 8s \in S; 8l \in [1; L_{max}]; 8r \in R \quad (4.28)$$

$$x_{swilr} = 0 \quad 8w \in W; 8s \in S; 8i \in I_w; 8l \in [1; L_{max}]; 8r \in R \quad (4.29)$$

$$\sum_{w \in W} f_{swlr} = M (1 - \sum_{w \in W} y_{swlr}) \quad 8s \in S; 8l \in [1; L_{max}]; 8r \in R \quad (4.30)$$

$$\sum_{i \in N} e_{wi} v_{swilr} = y_{swlr} \quad 8w \in W; 8s \in S; 8l \in [1; L_{max}]; 8r \in R \quad (4.31)$$

$$\sum_{i \in N} f_{swilr} = M y_{swlr} \quad 8w \in W; 8s \in S; 8l \in [1; L_{max}]; 8r \in R \quad (4.32)$$

$$V_{swilr} \begin{bmatrix} E & 0 \\ 0 & 0 \end{bmatrix} f_{swlr} \begin{bmatrix} E & 0 \\ 0 & 0 \end{bmatrix} M (1 \begin{bmatrix} E & 0 \\ 0 & 0 \end{bmatrix} x_{swilr}) \quad 8s \ 2 \ S; 8w \ 2 \ W; 8i \ 2 \ N; \quad (4.33)$$

$$8l \ 2 \ [1; L_{Max}]; 8r \ 2 \ R$$

$$V_{swilr} \begin{bmatrix} E & 0 \\ 0 & 0 \end{bmatrix} M x_{swilr} \quad 8s \ 2 \ S; 8w \ 2 \ W; 8i \ 2 \ N; \quad (4.34)$$

$$8l \ 2 \ [1; L_{Max}]; 8r \ 2 \ R$$

$$Z_{wi} \begin{bmatrix} E & 0 \\ 0 & 0 \end{bmatrix} \begin{matrix} X \\ r2R \end{matrix} x_{swilr} \quad 8s \ 2 \ S; 8w \ 2 \ W; 8i \ 2 \ N; \quad (4.35)$$

$$8l \ 2 \ [1; L_{Max}]$$

$$\begin{matrix} X & X \\ & F_{lr} \begin{bmatrix} E & 0 \\ 0 & 0 \end{bmatrix} F \end{matrix} \quad (4.36)$$

$$l2[1; L_{Max}] \ r2R$$

A função objetivo (4.23) maximiza o número de alocações distintas. As restrições (4.24)-(4.34) são análogas àquelas descritas por Araújo et al. (2012), garantindo que em cada período temos uma configuração válida. As restrições (4.35) calculam o número de alocações distintas. Por fim, a restrição (4.36) garante a produtividade mínima.

Assim como ocorreu para o caso com estações em paralelo, após uma hora de execução este modelo não foi capaz de encontrar soluções viáveis mesmo para instâncias pequenas. Em vista disso optamos por desenvolver uma heurística para o problema, descrita a seguir.

4.2.2 Heurística híbrida

Para resolver este problema, adaptamos também o método desenvolvido por Moreira e Costa (2012). A estratégia utilizada nesta heurística é análoga àquela descrita na Seção 4.1.2, isto é, gerar um *pool* de configurações e utilizar um modelo linear para encontrar a melhor combinação de configurações. Para gerar o *pool* de configurações poderíamos adaptar a heurística descrita na Seção 3.3.3. No entanto, este método não é rápido o suficiente para gerar o grande número de configurações para o ALWABP com múltiplas linhas necessário para o funcionamento desta heurística. Por outro lado, a heurística construtiva mostrou que é possível gerar rapidamente um grande número de soluções parciais do ALWABP com múltiplas linhas. Em vista disto, decidimos adaptar a heurística para gerar as soluções parciais do *pool*. Portanto, o modelo linear deverá encontrar soluções para o problema de rotação de tarefas com múltiplas linhas a partir de soluções para o ALWABP serial com partições do conjunto de trabalhadores.

4.2.2.1 Geração do pool de soluções parciais

A primeira etapa da heurística consiste na geração de soluções parciais para o ALWABP com múltiplas linhas. Para isto utilizamos uma heurística baseada em GRASP similar à desenvolvida para o caso com estações em paralelo. Para gerar as soluções do GRASP adaptamos a heurística descrita na Seção 3.2.2.1, conforme o Algoritmo 7.

Algoritmo 7 Procedimento de designação orientado a estações para o ALWABP-1

Entrada: Tempo de ciclo máximo $c_{\max}$

Entrada: Número máximo de estágios k

- 1: Inicie em uma estação vazia $n = 0$, $t(S_n) = 0$.
 - 2: Adicione os trabalhadores não alocados a $U_w = W$, adicione as tarefas não alocadas a $U = N$.
 - 3: **para** cada trabalhador $w \in W$ **faça**
 - 4: Calcule T_w , o conjunto de tarefas que seriam executadas pelo trabalhador w de acordo com a regra de prioridade utilizada.
 - 5: **fim**
 - 6: Aloque ao estágio k o trabalhador w que possuir a maior prioridade e suas tarefas T_w .
 $U_w = U_w \setminus T_w$, $U = U \cup T_w$, $n = n + 1$.
 - 7: **se** Há estágios vazios **então**
 - 8: Vá para o Passo 2.
 - 9: **senão**
 - 10: **se** $U = \emptyset$; **então**
 - 11: Retorne a solução.
 - 12: **senão**
 - 13: Retorne "solução não encontrada para o tempo de ciclo $c_{\max}$ "
 - 14: **fim**
 - 15: **fim**
-

Este algoritmo é similar ao Algoritmo 1. A principal diferença é que uma solução é válida para este algoritmo se todas as tarefas forem alocadas, mesmo que alguns trabalhadores não o sejam. É importante ressaltar que no passo 4 o método seleciona aleatoriamente entre as melhores tarefas, conforme descrito na Seção 4.1.2.1.

A heurística é então executada um determinado número de vezes para cada possível número de estágios (variando de um até o número máximo de trabalhadores). Utilizamos três estratégias para decidir quantas vezes executamos a heurística. A primeira distribui igualmente o número de execuções para cada número de estágios, utilizando como base o número máximo de execuções desejado. Já as outras duas dedicam mais execuções a valores próximos do máximo, já que são nestes casos que esperamos encontrar mais soluções distintas. A segunda estratégia define o número de execuções de acordo com uma progressão aritmética. Seja n o número total de execuções da heurística. A razão desta progressão aritmética é dada por:

$$2 \left\lceil \frac{EW}{1 + jWj} \right\rceil \quad (4.37)$$

Por fim, a terceira estratégia utiliza o mesmo número de execuções da segunda estratégia, porém caso encontre um determinado número de soluções repetidas consecutivas o método aborta as execuções para o número de estágios atual. Com isto, evitamos que o método dedique muito tempo explorando casos em que potencialmente há um número muito pequeno de soluções.

4.2.2.2 Modelo

De posse do *pool* de soluções parciais, utilizamos um modelo linear para encontrar a melhor solução a partir do *pool*. Usando a notação:

R *Pool* de configurações

F_r Taxa de produção da configuração r

a_{rwi} Igual a 1 se a tarefa i é alocada ao trabalhador w na configuração r

b_{rw} Igual a 1 se o trabalhador w pertence à configuração r

F Número mínimo de itens a serem processados

E as variáveis:

x_{rp} Variável binária. Igual a 1 se a configuração r é usada no período p .

z_{wi} Variável binária. Igual a 1 se a tarefa i é alocada ao trabalhador w em ao menos um subperíodo

A última fase da heurística consiste na resolução do seguinte modelo:

$$\text{Max} \sum_{w \in W} \sum_{i \in N} z_{wi} \quad (4.38)$$

$$\text{s.a:} \sum_{r \in R} F_r x_{rp} \geq F \quad (4.39)$$

$$\sum_{w \in W} \sum_{i \in N} a_{rwi} x_{rp} \geq 8w \in W; 8i \in N \quad (4.40)$$

$$\sum_{r \in R} b_{rw} x_{rp} = 1 \quad 8w \in W; 8p \in P \quad (4.41)$$

A função objetivo (4.38) maximiza o número de alocações distintas. A restrição (4.39) garante a produtividade mínima. As restrições (4.40) calculam o número de alocações dis-

tintas. Por fim, as restrições (4.41) garantem que cada trabalhador é alocado exatamente uma vez em cada período.

4.3 Rotação de tarefas com múltiplas linhas e estações em paralelo

É possível também aplicar ambas as extensões ao mesmo tempo ao problema de rotação de tarefas. Na Seção 4.3.1 apresentamos um modelo para o problema de rotação de tarefas com múltiplas linhas e estações em paralelo, enquanto que na seção 4.3.2 apresentamos uma heurística híbrida para o mesmo problema.

4.3.1 Modelo

A seguir descrevemos um modelo para o problema de rotação de tarefas capaz de suportar múltiplas linhas e estações em paralelo. Com as novas variáveis:

- t_{islr} Variável binária. Igual a um se a tarefa i é alocada à estação s na linha l durante o período r
- x_{swilr} Variável binária. Igual a um se a tarefa i é alocada ao trabalhador w na estação s da linha l durante o período r
- a_{sklr} Variável binária. Igual a um se o estágio s da linha l e período r tem k estações em paralelo
- F_{lr} Taxa de produção da linha l durante o período r
- f_{swlr} Taxa de produção do trabalhador w no estágio s da linha l durante o período r
- v_{swilr} Variável auxiliar, usada na linearização do modelo.

Podemos escrever o seguinte modelo para o problema:

$$\text{Max} \sum_{w \in W} \sum_{i \in N} x_{wi} z_{wi} \quad (4.42)$$

$$s:a: \quad \begin{matrix} X \\ s2S \end{matrix} st_{islr} \begin{matrix} E0 \\ E0 \\ 04 \\ 03 \end{matrix} \begin{matrix} X \\ s2S \end{matrix} st_{j\ sr} \quad 8i; j \ 2 \ Nj i \ 2 \ D_j; 8l \ 2 \ [1; L_{M\ ax}]; \quad (4.43)$$

$$\begin{matrix} X \\ w2W \end{matrix} x_{swilr} \begin{matrix} E0 \\ E0 \\ 04 \\ 04 \end{matrix} k(t_{islr} + a_{sklr} \begin{matrix} E0 \\ E0 \\ 00 \\ 00 \end{matrix} 1) \quad 8r \ 2 \ R \\ 8s \ 2 \ S; 8k \ 2 \ [0; K_{M\ ax}]; 8i \ 2 \ N; \quad (4.44)$$

$$x_{swilr} \begin{matrix} E0 \\ E0 \\ 05 \\ 05 \end{matrix} t_{islr} \quad 8l \ 2 \ [1; L_{M\ ax}]; 8r \ 2 \ R \\ 8s \ 2 \ S; 8w \ 2 \ W; 8i \ 2 \ N; \quad (4.45)$$

$$\begin{matrix} k \\ k=0 \end{matrix} x_{l\ ax} a_{sklr} = 1 \quad 8s \ 2 \ S; 8l \ 2 \ [1; L_{M\ ax}]; 8r \ 2 \ R \quad (4.46)$$

$$\begin{matrix} X \\ w2W \end{matrix} y_{swlr} = \begin{matrix} k \\ k=0 \end{matrix} x_{l\ ax} k a_{sklr} \quad 8s \ 2 \ S; 8l \ 2 \ [1; L_{M\ ax}]; 8r \ 2 \ R \quad (4.47)$$

$$\begin{matrix} l \\ l=1 \end{matrix} x_{l\ ax} X y_{swlr} = 1 \quad 8w \ 2 \ W; 8r \ 2 \ R \quad (4.48)$$

$$\begin{matrix} s2S \\ s2S \end{matrix} t_{islr} = 1 \quad 8i \ 2 \ N; 8l \ 2 \ [1; L_{M\ ax}]; 8r \ 2 \ R \quad (4.49)$$

$$x_{swilr} \begin{matrix} E0 \\ E0 \\ 05 \\ 05 \end{matrix} y_{swlr} \quad 8s \ 2 \ S; 8w \ 2 \ W; 8i \ 2 \ N; \\ 8l \ 2 \ [1; L_{M\ ax}]; 8r \ 2 \ R \quad (4.50)$$

$$x_{swilr} = 0 \quad 8w \ 2 \ W; 8s \ 2 \ S; 8i \ 2 \ l_w; \\ 8l \ 2 \ [1; L_{M\ ax}]; 8r \ 2 \ R \quad (4.51)$$

$$\begin{matrix} X \\ w2W \end{matrix} f_{swlr} \begin{matrix} E0 \\ E0 \\ 04 \\ 04 \end{matrix} f_{lr} \begin{matrix} E0 \\ E0 \\ 00 \\ 00 \end{matrix} M a_{s0lr} \quad 8s \ 2 \ S; 8l \ 2 \ [1; L_{M\ ax}]; 8r \ 2 \ R \quad (4.52)$$

$$\begin{matrix} w2W \\ i2N \end{matrix} e_{wi} v_{swilr} = y_{swlr} \quad 8w \ 2 \ W; 8s \ 2 \ S; 8l \ 2 \ [1; L_{M\ ax}]; \\ 8r \ 2 \ R \quad (4.53)$$

$$f_{swlr} \begin{matrix} E0 \\ E0 \\ 05 \\ 05 \end{matrix} M y_{swlr} \quad 8w \ 2 \ W; 8s \ 2 \ S; 8l \ 2 \ [1; L_{M\ ax}]; \\ 8r \ 2 \ R \quad (4.54)$$

$$v_{swilr} \begin{matrix} E0 \\ E0 \\ 04 \\ 04 \end{matrix} f_{swlr} \begin{matrix} E0 \\ E0 \\ 00 \\ 00 \end{matrix} M (1 \begin{matrix} E0 \\ E0 \\ 00 \\ 00 \end{matrix} x_{swilr}) \quad 8s \ 2 \ S; 8w \ 2 \ W; 8i \ 2 \ N; \\ 8l \ 2 \ [1; L_{M\ ax}]; 8r \ 2 \ R \quad (4.55)$$

$$v_{swilr} \begin{matrix} E0 \\ E0 \\ 03 \\ 03 \end{matrix} M x_{swilr} \quad 8s \ 2 \ S; 8w \ 2 \ W; 8i \ 2 \ N; \\ 8l \ 2 \ [1; L_{M\ ax}]; 8r \ 2 \ R \quad (4.56)$$

$$z_{wi} \begin{matrix} E0 \\ E0 \\ 05 \\ 05 \end{matrix} X x_{swilr} \quad 8s \ 2 \ S; 8w \ 2 \ W; 8i \ 2 \ N; \\ r2R \quad (4.57)$$

$$\begin{matrix} l \\ l=1 \end{matrix} x_{l\ ax} X \quad 8l \ 2 \ [1; L_{M\ ax}] \quad (4.58)$$

$$f_{lr} \begin{matrix} E0 \\ E0 \\ 04 \\ 04 \end{matrix} F \quad (4.59)$$

Estas restrições são análogas às apresentadas na seção 4.1.1, com a adição de um índice para as múltiplas linhas. Este modelo pode ser reduzido àqueles apresentados nas Seções 4.1.1 e 4.2.1, bastando para isto limitar o número de linhas ou de estações em paralelo a um, respectivamente. Consequentemente este modelo também não conseguiu encontrar soluções viáveis mesmo para problemas de pequeno porte, o que nos levou a adaptar também as heurísticas híbridas desenvolvidas para as duas extensões do problema de rotação de tarefas.

4.3.2 Heurística híbrida

Para resolver o problema de rotação de tarefas com estações em paralelo e múltiplas linhas, adaptamos a heurística descrita na Seção 4.2.2. Ao gerar o *pool* de soluções parciais, adaptamos o método GRASP descrito na Seção 4.2.2.1 para suportar estações em paralelo. Para isto, alteramos o passo 3 do Algoritmo 7 para verificar também combinações de trabalhadores, conforme o passo 8 do Algoritmo 1. Em seguida, o modelo linear se encarrega de decidir quais tipos de soluções parciais utilizar, podendo combinar soluções seriais, com estações em paralelo e com múltiplas linhas em uma mesma solução.

4.4 Medida de qualidade para a rotação de tarefas

De modo a permitir a comparação entre diferentes extensões para o problema de rotação de tarefas desenvolvemos uma medida de qualidade para o mesmo. Esta medida leva em consideração limitantes inferior e superior para o número de alocações distintas para o problema serial. Valores relaxados para estes limitantes são fáceis de serem calculados a partir do número de tarefas, trabalhadores e períodos.

Seja z o número de alocações distintas de uma dada solução. Sejam $z_{\min}$ e $z_{\max}$ os limitantes inferior e superior para o número de alocações para o problema serial. Desejamos que, caso $z = z_{\min}$, $Q = 0\%$, enquanto que se $z = z_{\max}$, $Q = 100\%$. Para isto, definimos a medida de qualidade Q através da seguinte expressão:

$$Q = \frac{z - z_{\min}}{z_{\max} - z_{\min}} \quad (4.60)$$

No problema serial cada configuração da linha possui exatamente $z = |N|$ alocações distintas. A pior solução possível consiste em duplicar a mesma configuração em cada

período. Desta forma, $z_{\min} = jNj$. Por outro lado, se relaxarmos a restrição referente à produtividade mínima e desconsiderarmos as tarefas infactíveis, na melhor solução para $jPj \leq jWj$ cada tarefa é executada por um trabalhador diferente em cada período. O número máximo de alocações, desconsiderando as tarefas infactíveis, corresponde a $jWjjNj$. Desta forma, temos que $z_{\max} = \min(jPj; jWj)jNj$. Portanto, a medida de qualidade (4.60) corresponde a:

$$Q = \frac{z_{\min} jNj}{\min(jPj; jWj)jNj} = \frac{z_{\min} jNj}{(\min(jPj; jWj) \cdot 1)jNj} \quad (4.61)$$

É importante notar que o uso de estações em paralelo e/ou múltiplas linhas podem levar a soluções com $Q > 100\%$. Isto ocorre porque nestes casos algumas tarefas são executadas por mais de um trabalhador em cada período. Com isto, o número total de alocações distintas para estas soluções é maior do que em qualquer solução factível para o problema serial.

4.5 Resultados

A seguir relatamos, de forma resumida, os resultados dos testes realizados para os métodos descritos anteriormente. Todos os testes foram realizados em um computador com processador Intel Core 2 Duo T5450, 1,66 GHz e 3 GB de RAM. Para a resolução dos modelos, utilizamos o pacote comercial IBM ILOG CPLEX 12.5. As soluções são comparadas de acordo com a medida de qualidade apresentada na seção 4.4. Uma solução é considerada melhor que outra caso sua qualidade seja maior de acordo com esta medida.

Os modelos descritos nas seções 4.1.1, 4.2.1 e 4.3.1 se mostraram difíceis de resolver. Após uma hora de execução, eles não foram capazes de encontrar soluções viáveis para os problemas, mesmo para instâncias de pequeno porte. Isto incentivou o desenvolvimento das heurísticas descritas nas seções 4.1.2, 4.2.2 e 4.3.2. A seguir apresentamos os resultados obtidos para estas heurísticas.

Para testar as heurísticas utilizamos 320 instâncias desenvolvidas por Chaves et al. (2007). Nós fixamos os parâmetros α e β de cada método após testes preliminares com 40 instâncias da família Heskia, com $\alpha \in [25\%; 50\%; 100\%; 150\%; 200\%]$ e $\beta \in [3::10]$. A tabela 17 mostra os melhores parâmetros para cada heurística: rotação de tarefas com estações em paralelo (H1), com múltiplas linhas (H2) e com ambas as extensões (H3).

Para H2 e H3 utilizamos a terceira estratégia de distribuição do número de execuções do GRASP por número de trabalhadores, visto que esta encontrou melhores resultados em testes preliminares.

Tabela 17: Parâmetros para as heurísticas

Heurística	$\frac{EO}{GR}$	$\frac{EO}{OT}$
H1	150	6
H2	200	6
H3	100	5

De posse destes parâmetros, as heurísticas foram utilizadas para resolver as 320 instâncias desenvolvidas por Chaves et al. (2007), com 1000 iterações para o GRASP. O número de itens a serem processados ($\underline{F}$) corresponde à taxa de produção da melhor solução conhecida para o ALWABP multiplicada pelo número de subperíodos e dividida por $R = [1:05; 1:1; 1:25; 1:5]$. Isto nos dá taxas de produção equivalentes aos tempos de ciclo médios utilizados por Moreira e Costa (2012).

As tabelas 18, 19, 20 e 21 apresentam os resultados para as famílias Roszieg, Heskia, Tonge e Weemag, respectivamente. Nós comparamos os resultados das três abordagens (colunas H1, H2 e H3) com aqueles encontrados por Moreira e Costa (2012) (coluna HAJR) e a melhor solução conhecida do problema de rotação de tarefas serial (coluna F*). As colunas Q mostram a qualidade média das soluções de acordo com a medida descrita na seção 4.4. As colunas Improv. indicam a percentagem de soluções cuja qualidade é maior ou igual à da solução encontrada pelo HAJR. As colunas Tempo apresentam o tempo de execução médio, em segundos. Os resultados estão agrupados pelo número de subperíodos (coluna |P|) e R.

Tabela 18: Resultados para a família Roszieg

	P	R	F*	HAJR	H1			H2			H3		
			Q	Q	Q	Improv.	Tempo	Q	Improv	Tempo	Q	Improv	Tempo
2		5%	70,15%	69,95%	63,20%	37,50%	2,2	46,10%	16,25%	0,5	54,05%	26,25%	1,3
		10%	85,80%	85,70%	94,25%	65,00%	2,2	70,60%	20,00%	0,5	82,40%	43,75%	1,4
		25%	96,40%	96,20%	135,65%	91,25%	2,2	92,25%	31,25%	0,5	106,15%	58,75%	1,3
		50%	98,50%	98,45%	157,55%	96,25%	2,2	101,05%	36,25%	0,5	117,20%	67,50%	1,4
4		5%	48,30%	48,12%	38,78%	23,75%	2,2	27,43%	7,50%	1,0	31,63%	10,00%	1,7
		10%	60,65%	60,53%	55,08%	37,50%	2,3	38,88%	7,50%	0,9	42,93%	8,75%	1,7
		25%	77,78%	77,70%	75,85%	42,50%	2,3	50,02%	5,00%	1,2	54,32%	5,00%	1,8
		50%	86,47%	86,22%	85,50%	38,75%	2,4	63,98%	6,25%	1,3	69,77%	8,75%	1,8
7		5%	31,86%	31,68%	26,85%	20,00%	2,2	19,13%	7,50%	1,9	21,06%	11,25%	2,3
		10%	39,76%	39,67%	36,25%	31,25%	2,3	23,66%	5,00%	2,3	25,92%	2,50%	2,4
		25%	49,29%	49,17%	46,16%	26,25%	2,3	33,26%	1,25%	3,0	36,14%	2,50%	2,5
		50%	54,06%	53,83%	49,33%	18,75%	2,3	39,34%	0,00%	3,3	42,23%	0,00%	2,7

As heurísticas conseguiram resolver todas as instâncias em um tempo computacional relativamente curto, exceto para instâncias de grande porte com valores altos de |P| e

Tabela 19: Resultados para a família Heskia

		F*	HAJR	H1			H2			H3		
P	R	Q	Q	Q	Improv.	Tempo	Q	Improv	Tempo	Q	Improv	Tempo
2	5%	75,49%	74,60%	50,54%	16,25%	19,3	61,88%	16,25%	3,7	65,54%	25,00%	12,9
	10%	85,98%	85,45%	82,37%	46,25%	19,4	99,38%	42,50%	3,7	98,62%	51,25%	12,9
	25%	96,74%	96,16%	127,50%	86,25%	19,4	146,65%	76,25%	3,8	147,41%	90,00%	13,0
	50%	99,69%	99,42%	151,21%	92,50%	19,4	168,57%	78,75%	4,0	176,29%	93,75%	13,1
4	5%	54,45%	53,35%	35,49%	6,25%	19,4	41,80%	23,75%	5,0	42,60%	16,25%	13,7
	10%	62,87%	62,17%	51,40%	15,00%	19,4	58,69%	52,50%	5,0	58,72%	45,00%	13,8
	25%	75,48%	74,40%	74,96%	52,50%	19,7	74,32%	53,75%	5,7	77,68%	65,00%	14,0
	50%	84,36%	82,98%	85,45%	53,75%	20,0	82,92%	37,50%	8,1	86,80%	60,00%	14,7
7	5%	36,27%	35,28%	25,65%	5,00%	19,4	28,76%	23,75%	6,8	29,43%	12,50%	14,9
	10%	42,32%	41,57%	35,39%	10,00%	19,5	36,38%	38,75%	8,1	37,89%	35,00%	15,3
	25%	50,25%	49,38%	47,54%	36,25%	19,8	44,29%	28,75%	11,8	46,44%	40,00%	16,4
	50%	55,67%	54,25%	52,31%	28,75%	20,2	48,29%	16,25%	14,6	50,97%	32,50%	18,4

Tabela 20: Resultados para a família Tonge

		F*	HAJR	H1			H2			H3		
P	R	Q	Q	Q	Improv.	Tempo	Q	Improv	Tempo	Q	Improv	Tempo
2	5%	73,45%	73,45%	8,77%	5,00%	496,2	13,88%	7,50%	105,8	12,55%	8,75%	821,7
	10%	88,95%	88,95%	35,66%	10,00%	502,8	44,21%	12,50%	106,1	43,20%	12,50%	822,5
	25%	98,61%	98,61%	109,38%	80,00%	506,2	120,27%	45,00%	108,3	118,23%	61,25%	824,4
	50%	99,93%	99,93%	145,13%	100,00%	507,8	151,02%	60,00%	117,9	146,36%	90,00%	829,5
4	5%	53,10%	53,10%	12,99%	6,25%	496,4	15,50%	7,50%	152,1	14,71%	7,50%	862,2
	10%	68,07%	68,07%	31,79%	6,25%	505,1	36,54%	13,75%	144,9	36,62%	10,00%	850,3
	25%	85,80%	85,80%	76,10%	18,75%	508,4	76,92%	38,75%	167,6	75,94%	35,00%	856,1
	50%	94,01%	94,01%	103,38%	90,00%	532,4	95,79%	52,50%	364,8	95,79%	53,75%	993,9
7	5%	38,41%	38,41%	12,44%	5,00%	498,1	14,41%	10,00%	242,0	14,24%	8,75%	916,1
	10%	50,75%	50,75%	26,51%	6,25%	505,3	28,06%	8,75%	276,4	27,24%	6,25%	945,0
	25%	68,15%	68,15%	56,76%	6,25%	528,2	52,42%	8,75%	454,0	52,90%	6,25%	1014,3
	50%	78,55%	78,55%	77,83%	45,00%	1368,8	65,64%	7,50%	803,7	69,15%	13,75%	1241,0

Tabela 21: Resultados para a família Weemag

		F*	HAJR	H1			H2			H3		
P	R	Q	Q	Q	Improv.	Tempo	Q	Improv	Tempo	Q	Improv	Tempo
2	5%	82,53%	95,57%	1,54%	0,00%	437,4	2,04%	0,00%	115,0	2,04%	0,00%	1002,3
	10%	92,37%	106,11%	6,98%	1,25%	450,1	15,55%	0,00%	107,8	17,05%	5,00%	1002,3
	25%	99,67%	113,93%	84,46%	53,75%	466,9	79,39%	6,25%	113,1	88,16%	30,00%	1009,5
	50%	100,00%	114,29%	142,39%	100,00%	467,2	140,41%	45,00%	128,5	118,57%	71,25%	1031,6
4	5%	55,59%	61,94%	1,38%	0,00%	437,8	2,49%	0,00%	197,3	3,02%	0,00%	1059,2
	10%	66,44%	73,57%	8,29%	0,00%	450,2	11,72%	0,00%	178,9	12,76%	2,50%	1056,7
	25%	82,44%	90,71%	57,20%	2,50%	470,3	60,18%	23,75%	210,4	57,38%	11,25%	1070,6
	50%	93,33%	102,38%	100,28%	81,25%	487,5	101,21%	56,25%	551,0	87,68%	41,25%	1228,9
7	5%	38,78%	42,74%	1,77%	0,00%	437,4	2,45%	0,00%	342,0	2,75%	1,25%	1172,9
	10%	48,18%	52,81%	8,54%	0,00%	451,1	10,33%	5,00%	351,7	11,07%	5,00%	1204,0
	25%	61,95%	67,57%	42,89%	6,25%	476,1	41,97%	25,00%	668,7	38,88%	11,25%	1373,7
	50%	73,52%	79,96%	75,00%	65,00%	1404,4	67,08%	41,25%	1150,2	65,05%	18,75%	1731,8

R. H2 foi o mais rápido entre os três métodos e encontrou as melhores soluções para as famílias Heskia e Tonge, enquanto que H1 se saiu melhor para instâncias pequenas (família Roszieg) e H3 para instâncias grandes (família Weemag). Os três métodos são capazes de, em alguns casos, encontrar soluções melhores que aquelas encontradas pelo HAJR. Quanto maior o número de períodos e o valor de R, maior a melhoria destes métodos em relação ao HAJR, exceto para as famílias Roszieg e Heskia. Isto ocorre porque o número de trabalhadores nestas famílias é limitado a 6 e 7 respectivamente. Uma vez que $\|P_j - W_j\|$, isto nos impede de encontrar soluções com $Q > 100\%$.

5 *Conclusões e trabalhos futuros*

Neste trabalho estudamos duas extensões para o ALWABP. A primeira permite o uso de estações em paralelo, enquanto que a segunda possibilita múltiplas linhas em paralelo. Estas extensões foram estudadas isoladamente e também como parte do problema de rotação de tarefas. Para cada problema desenvolvemos modelos e métodos, que foram testados usando instâncias disponíveis na literatura. Além disso, propusemos uma nova família de instâncias que permitiram uma análise mais completa dos resultados por conter uma maior variabilidade dos grafos de precedência (agrupados em três tipos) e do número de trabalhadores.

Desenvolvemos modelos e métodos para cada problema citado. Devido à complexidade dos problemas nos focamos em métodos heurísticos e meta-heurísticos. Para o ALWABP com estações em paralelo desenvolvemos um método baseado em busca em vizinhança. Embora este método tenha encontrado dificuldades para resolver instâncias de grande porte conhecidas da literatura, ele foi capaz de resolver rapidamente instâncias de pequeno e médio porte, bem como as novas instâncias, encontrando bons resultados.

Para o ALWABP com múltiplas linhas desenvolvemos um modelo linear, um método exato e uma heurística baseada em GRASP. O modelo foi capaz de resolver apenas instâncias de pequeno porte, que nem sempre possuem soluções viáveis com mais de uma linha. Como alternativa, desenvolvemos um método exato que divide o ALWABP com múltiplas linhas em vários subproblemas ALWABP do tipo serial. Com isto, conseguimos resolver instâncias de médio porte de forma exata em um tempo computacional relativamente baixo, encontrando boas soluções em alguns casos. A dificuldade para resolver instâncias de grande porte nos levou a desenvolver um método heurístico baseado em GRASP. Utilizamos este método para resolver as novas instâncias, chegando à conclusão de que instâncias com grafos do tipo cadeia possuem grande chance de possuir boas soluções utilizando múltiplas linhas, enquanto que em instâncias com grafos do tipo gargalo a chance é menor, porém a melhoria em relação à solução ótima do problema serial é mais significativa.

Para o problema de rotação de tarefas estudamos o caso com estações em paralelo, com múltiplas linhas e com ambas as extensões. Para os três casos, desenvolvemos modelos matemáticos e métodos heurísticos. Os modelos se mostraram de difícil resolução, não conseguindo encontrar sequer soluções viáveis após uma hora de execução. Isto já era esperado, dada a complexidade tanto do problema de rotação de tarefas quanto das extensões. Em vista disto, desenvolvemos heurísticas híbridas, que consistem na geração de um *pool* de soluções parciais e na execução de um modelo linear para encontrar a melhor combinação de soluções do *pool*. Estes métodos tiveram dificuldade em obter resultados competitivos em relação às heurísticas presentes na literatura para casos com poucos períodos e grande demanda por itens. Para instâncias de pequeno porte, ao reduzirmos o número mínimo de itens a serem produzidos conseguimos obter soluções de alta qualidade, em muitos casos melhores que aquelas encontradas sem as extensões. No entanto, os métodos não conseguiram encontrar soluções viáveis para diversas instâncias de grande porte. Mesmo assim, em alguns casos os métodos encontraram soluções de alta qualidade. Isto demonstra uma das maiores dificuldades deste problema: a necessidade de balancear a presença de soluções de boa qualidade com a variabilidade do *pool* de soluções.

5.1 Trabalhos futuros

Uma vez que ambas as extensões são de difícil resolução, métodos heurísticos e meta-heurísticos são os mais promissores para obtenção de soluções de boa qualidade em tempos apropriados. Para o ALWABP com estações em paralelo, uma vez que soluções ótimas tendem a conter trabalhadores similares em um mesmo estágio, futuros métodos podem agrupar estes trabalhadores como forma de pré-processamento. Já para o ALWABP com múltiplas linhas a ideia de decompor o problema em múltiplos ALWABPs com subconjuntos dos trabalhadores parece promissora. Um passo inicial seria estudar estratégias eficientes para identificar bons subconjuntos de trabalhadores, além de determinar os tamanhos ideais destes subconjuntos.

No caso da rotação de tarefas com estas extensões o primeiro passo corresponde a desenvolver métodos que explorem melhor o espaço de busca. Queremos obter soluções parciais que sejam quase ótimas. Esta é uma tarefa difícil, visto que no ALWABP pequenas alterações na solução podem alterar drasticamente a qualidade da mesma.

Referências Bibliográficas

- Araújo, F. F. B. (2011). *Balanceamento de linhas de produção com trabalhadores deficientes e máquinas paralelas*, Master's thesis, Universidade de São Paulo.
- Araújo, F. F. B. e Costa, A. M. (2010). A linear model for the assembly line and worker assignment balancing problem with parallel workstations, *ALIO-INFORMS Joint International meeting*.
- Araújo, F. F. B., Costa, A. M. e Miralles, C. (2010a). Balanceando linhas de produção com trabalhadores deficientes e estações paralelas, *V International conference on production research - Americas*.
- Araújo, F. F. B., Costa, A. M. e Miralles, C. (2010b). Extensão do problema alwabp: melhorando a flexibilidade através de estações paralelas, *XLII Simpósio Brasileiro de Pesquisa Operacional*.
- Araújo, F. F. B., Costa, A. M. e Miralles, C. (2014). Rotação de tarefas em linhas de produção com trabalhadores deficientes e estações em paralelo, *Anais do XLVI Simpósio Brasileiro de Pesquisa Operacional* **46**.
- Araújo, F. F., Costa, A. M. e Miralles, C. (2012). Two extensions for the alwabp: Parallel stations and collaborative approach, *International Journal of Production Economics* **140**: 483–495.
- Azizi, N., Zolfaghari, S. e Liang, M. (2010). Modeling job rotation in manufacturing systems: The study of employee's boredom and skill variations, *International Journal of Production Economics* **123**: 69–85.
- Baybars, I. (1986). A survey of exact algorithms for the simple assembly line balancing problem, *Management Science* **32**: 909–932.
- Becker, C. e Scholl, A. (2006). A survey on methods in generalized assembly line balancing, *European Journal of Operational Research* **168**: 694–715.
- Blum, C. (2011). Iterative beam search for simple assembly line balancing with a fixed number of work stations, *SORT* **35**: 145–164.

- Blum, C. e Miralles, C. (2011). On solving the assembly line worker assignment and balancing problem via beam search, *Computer & Operations Research* **38**: 328–339.
- Borba, L. D. M. e Ritt, M. (2012). A task-oriented branch-and-bound method for the assembly line worker assignment and balancing problem, *Anais do XLIV Simpósio Brasileiro de Pesquisa Operacional*, Rio de Janeiro, pp. 1–10.
- Borba, L. M. e Ritt, M. (2014). A parallel branch-and-bound for the assembly line worker assignment and balancing problem, *Anais do XLVI Simpósio Brasileiro de Pesquisa Operacional*.
- Boysen, N. e Fliedner, M. (2006). A versatile algorithm for assembly line balancing, *European Journal of Operational Research* **184**: 39–56.
- Boysen, N., Fliedner, M. e Scholl, A. (2007). A classification of assembly line balancing problem, *European Journal of Operational Research* **183**: 674–693.
- Boysen, N., Fliedner, M. e Scholl, A. (2008). Assembly line balancing: Which model to use and when?, *International Journal of Production Economics* **111**: 509–528.
- Bukchin, J. e Rubinovitz, J. (2003). A weighted approach for assembly line design with station paralleling and equipment selection, *IIE Transactions* **35**: 73–85.
- Carnahan, B., Redfern, M. e Norman, B. (2000). Designing safe job rotation schedules using optimization and heuristic search, *Ergonomics* **43**: 543–560.
- Chaves, A. A. (2009). *Uma metaheurística híbrida de busca por agrupamentos aplicada a problemas de otimização combinatória*, Tese de doutorado., Instituto Nacional de Pesquisas Espaciais.
- Chaves, A. A., Lorena, L. A. N. e Miralles, C. (2008). Hybrid metaheuristic for the assembly line worker assignment and balancing problem, *Lecture Notes on Computer Science* **5818**: 1–14.
- Chaves, A. A., Miralles, C. e Lorena, L. A. N. (2007). Clustering search approach for the assembly line worker assignment and balancing problem, *International Conference on Computers and Industrial Engineering* **37**: 10.
- Cortez, P. M. C. e Costa, A. M. (2011). A mathematical model and a hybrid heuristic for sequencing mixed-model assembly lines with disabled workers, *Anais do XLIII Simpósio Brasileiro de Pesquisa Operacional* **43**: 2046–2055.

- Cortez, P. M. C. e Costa, A. M. (2013). Sequenciamento de linhas de produção multi-modelo com trabalhadores heterogêneos, *XLV Simpósio Brasileiro de Pesquisa Operacional (Aceito)* **45**: 1–12.
- Cortez, P. M. e Costa, A. M. (2014). Sequencing mixed-model assembly lines operating with a heterogeneous workforce, *International Journal of Production Research* **53**: 3419–3432.
- Costa, A. M. e Miralles, C. (2009). Job rotation in assembly lines employing disabled workers, *International Journal of Production Economics* **120**: 625–632.
- Costa, A. M., Santos, L. M. R., Moreira, M. C. O. e Miralles, C. (2009). Análise da eficiência de linhas de produção com trabalhadores deficientes, *Anais do XLI SBPO*.
- Diego-Mas, J., Asensio-Cuesta, S., Sanchez-Romero, M. e Artacho-Ramirez, M. (2009). A multi-criteria genetic algorithm for the generation of job rotation schedules, *International Journal of Industrial Ergonomics* **39**: 23–33.
- Ege, Y., Azizoglu, M. e Ozdemirel, N. E. (2009). Assembly line balancing with station paralleling, *Computers & Industrial Engineering* **57**: 1218–1225.
- Feo, T. e Resende, M. (1995). Greedy randomized adaptive search procedures, *Journal of Global Optimization* **6**: 109–134.
- Human Development Reports* (2012). URL: <http://hdr.undp.org>, acesso em novembro de 2012.
URL: <http://hdr.undp.org>
- McMullen, P. R. e Frazier, G. V. (1997). A heuristic for solving mixed-model line balancing problems with stochastic task durations and parallel stations, *International Journal of Production Economics* **51**: 177–190.
- Michalos, G., Makris, S. e Chryssolouris, G. (2013). The effect of job rotation during assembly on the quality of final product, *IRP Journal of Manufacturing Science and Technology* **6, Issue 3**: 187–197.
- Miralles, C., Garcia-Sabater, J. P., Andres, C. e Cardos, M. (2008). Branch and bound procedures for solving the assembly line worker assignment and balancing problem: Application to sheltered work centres for disabled, *Discrete Applied Mathematics* **156**: 352–367.

- Miralles, C., Garcia-Sabater, J. P., Andrés, C. e Cardos, M. (2007). Advantages of assembly lines in sheltered work centres for disabled, *International Journal of Production Economics* **110**: 187–197.
- Moreira, M. C. O. e Costa, A. M. (2009a). A minimalist yet efficient tabu search algorithm for balancing assembly lines with disabled workers, *Simpósio Brasileiro de Pesquisa Operacional*.
- Moreira, M. C. O. e Costa, A. M. (2009b). Simple heuristics for the assembly line and worker assignment balancing problem, Working Paper .
- Moreira, M. C. O. e Costa, A. M. (2012). Hybrid heuristics for planning job rotation in assembly lines with disabled workers, *International Journal of Production Economics* (Accepted) .
- Moreira, M. C. O., Miralles, C. e Costa, A. M. (2015). Model and heuristics for the assembly line worker integration and balancing problem, *Computers & Operations Research* **54**: 64–73.
- Moreira, M. C. O., Ritt, M., Costa, A. M. e Chaves, A. a. (2012). Simple heuristics for the assembly line worker assignment and balancing problem, *Journal of Heuristics* **18**(3): 505–524.
URL: <http://www.springerlink.com/index/10.1007/s10732-012-9195-5>
- Mutlu, O., Polat, O. e Supciller, A. A. (2013). An iterative genetic algorithm for the assembly line worker assignment and balancing problem of type-II, *Computers & Operations Research* **40**: 418–426.
URL: <http://dx.doi.org/10.1016/j.cor.2012.07.010>
- Otto, A., Otto, C. e Scholl, A. (2011). Salbp-gen - a systematic data generator for (simple) assembly line balancing, *Jena Research Papers in Business and Economics* **05/2011**.
- Otto, A. e Scholl, A. (2011). Incorporating ergonomic risks into assembly line balancing, *European Journal of Operational Research* **212**: 277–285.
- Plebani, L. J. e Chen, S. (2006). Balancing u-shaped assembly lines with parallel stations, *Technical report*, Lehigh University.
- Retratos da Deficiência no Brasil* (2003). Disponível em: http://www.fgv.br/cps/deficiencia_br/PDF/PPD_Sumario_Executivo.pdf, acesso em 26 de janeiro de 2010.

- Sato, T. d. O. e Coury, H. J. (2009). Evaluation of musculoskeletal health outcomes in the context of job rotation and multifunctional jobs, *Applied Ergonomics* **40**: 707–712.
- Scholl, A. e Becker, C. (2006). State-of-the-art exact and heuristic solution procedures for simple assembly line balancing, *European Journal of Operational Research* **168**: 666–693.
- Seçkiner, S. U. e Kurt, M. (2007). A simulated annealing approach to the solution of job rotation scheduling problems, *Applied Mathematics and Computation* **188**: 31–45.
- Süer, G. A. (1998). Designing parallel assembly lines, *Computers & industrial engineering* **35**: 467–470.
- Simaria, A. S. e Vilarinho, P. M. (2004). A genetic algorithm based approach to the mixed-model assembly line balancing problem of type ii, *Computers & Industrial Engineering* **47**: 391–407.
- Simonelli, A. P. (2009). *Contribuições da análise da atividade e do modelo social para a inclusão no trabalho de pessoas com deficiência*, PhD thesis, Universidade Federal de São Carlos.
- Simonelli, A. P., da Silva Rodrigues, D. e Soares, L. (2006). Caracterização do perfil de trabalhadores afastados e de pessoas com deficiência no mercado de trabalho do município de são carlos., *Anais das XIV Jornadas de Jovens pesquisadores da Associação de Universidades do Grupo Montevidéu*.
- Sivasankaran, P. e Shahabudeen, P. (2014). Literature review of assembly line balancing problems, *The International Journal of Advanced Manufacturing Technology* **73**: 1665–1694.
- Tharmmaphornphilas, W. e Norman, B. A. (2007). A methodology to create robust job rotation schedules, *Annals of operations research*.
- Tiacci, L. e Saetta, S. (2007). Process-oriented simulation for mixed-model assembly lines, *Summer Computer Simulation Conference*.
- Zhou, R. G. A. . M. (1997). A parallel station heuristic for the mixed-model production line balancing problem, *International Journal of Production Research* **35**: 3095–3106.

APÊNDICE A – Resultados detalhados dos testes

Neste capítulo, apresentamos de forma detalhada os resultados da heurística descrita na seção 3.2.2. Estes resultados complementam aqueles apresentados na seção 3.2.2.3. Utilizamos 360 instâncias definidas por Chaves et al. (2007), divididas em 4 famílias de 80 instâncias cada. Além disto, utilizamos 160 instâncias definidas por Araújo et al. (2010a). As tabelas a seguir apresentam de forma resumida os resultados obtidos. Os resultados representam a média entre todas as instâncias testadas. O tempo de execução da heurística híbrida foi limitado em 1800 segundos para cada critério. Para cada critério, apresentamos a distância média entre as soluções e as melhores soluções conhecidas para o ALWABP serial (coluna C), o percentual de soluções utilizando paralelismo (coluna P) e o tempo médio de execução da heurística em segundos (coluna T). Os resultados estão agrupados por critério utilizado. As tabelas 22, 23 e 24 mostram os resultados para a mat heurísticheurística híbrida utilizando os critérios 1, 2 e 3, respectivamente utilizando $K_{Max} = 2$, enquanto que as tabelas 25, 26 e 27 apresentam os resultados para os mesmos critérios utilizando $K_{Max} = 3$.

Os três critérios tiveram dificuldades em melhorar as soluções obtidas pela heurística construtiva. As melhores soluções foram encontradas utilizando a solução encontrada pela heurística construtiva com os critérios MinRLB e MaxPWD-. Quando comparadas com as melhores soluções conhecidas para o ALWABP serial, as soluções encontradas estavam, em média, a 9,97%, 8,56% e 9,78% para os critério 1, 2 e 3, respectivamente. O critério 2 se sobressaiu em relação aos demais por causar um impacto maior na solução, permitindo que tarefas da estação mais carregada e das estações vizinhas a esta pudessem ser realocadas. O tempo de execução dos critérios foi bastante pequeno, raramente ultrapassando 5 s mesmo em instâncias de grande porte.

As tabelas 28, 29 e 30 apresentam os resultados para os critérios 4, 5 e 6, respectivamente, utilizando $K_{Max} = 2$, enquanto que as tabelas 31, 32 e 33 apresentam os resultados

Tabela 22: Resultados para o método baseado em busca em vizinhança com o critério 1 com $K_{max} = 2$																											
	MaxTasks			MaxTasks1.2			MaxTasks1.4			MaxTasks1.6			MaxTasks1.8			MaxTasks2			MaxTasks3			MaxTaskK			MinRLB		
	C	P	T	C	P	T	C	P	T	C	P	T	C	P	T	C	P	T	C	P	T	C	P	T	C	P	T
MinD	37.14%	86.79%	0.12	25.67%	75.36%	0.07	19.64%	60.54%	0.05	17.44%	40.36%	0.05	16.84%	26.96%	0.04	16.23%	8.75%	0.04	16.36%	2.86%	0.04	16.41%	3.04%	0.04	12.33%	37.32%	0.04
MinR	38.93%	86.96%	0.10	27.78%	78.21%	0.07	18.41%	62.14%	0.05	17.11%	41.96%	0.04	15.92%	29.64%	0.04	15.72%	10.89%	0.04	15.85%	3.04%	0.04	15.86%	3.39%	0.04	12.00%	34.64%	0.04
MaxFTime	45.80%	85.00%	0.13	35.62%	76.43%	0.07	29.52%	64.46%	0.05	26.00%	43.21%	0.05	25.37%	27.86%	0.04	25.00%	10.89%	0.04	24.94%	3.21%	0.04	24.91%	4.11%	0.04	17.71%	37.68%	0.04
MaxFTime	44.84%	85.00%	0.12	36.17%	77.14%	0.07	28.79%	64.64%	0.05	25.47%	45.71%	0.04	24.65%	30.36%	0.04	24.53%	10.54%	0.04	24.77%	3.39%	0.04	24.71%	4.11%	0.04	16.48%	38.39%	0.04
MaxPWS+	48.13%	85.71%	0.18	40.74%	79.11%	0.11	35.14%	71.25%	0.06	31.67%	54.64%	0.05	29.79%	38.93%	0.04	29.22%	24.29%	0.04	29.67%	8.04%	0.04	29.25%	18.21%	0.04	26.47%	36.25%	0.04
MaxPWS-	49.23%	84.11%	0.16	43.58%	78.75%	0.14	39.75%	69.29%	0.09	37.02%	55.89%	0.05	34.91%	41.96%	0.04	33.77%	28.39%	0.04	34.24%	9.64%	0.04	33.91%	20.71%	0.04	31.60%	38.21%	0.04
MaxPWD+	39.79%	86.79%	0.10	30.22%	79.46%	0.07	23.88%	66.07%	0.05	19.99%	49.82%	0.05	18.68%	32.86%	0.04	17.82%	12.94%	0.04	17.50%	2.50%	0.04	17.66%	6.25%	0.04	12.47%	30.89%	0.04
MaxPWD-	36.56%	85.71%	0.10	26.94%	80.71%	0.06	20.54%	66.96%	0.05	16.47%	52.50%	0.05	15.46%	34.46%	0.04	15.35%	13.93%	0.04	15.11%	2.14%	0.04	15.19%	8.21%	0.04	9.97%	27.50%	0.04

Tabela 22: Resultados para o método baseado em busca em vizinhança com o critério 1 com $K_{max} = 2$

Tabela 23: Resultados para o método baseado em busca em vizinhança com o critério 2 com $K_{max} = 2$																											
	MaxTasks			MaxTask1.2			MaxTask1.4			MaxTask1.6			MaxTask1.8			MaxTask2			MaxTask3			MaxTaskK					
	C	P	T	C	P	T	C	P	T	C	P	T	C	P	T	C	P	T	C	P	T	C	P	T			
MinD	37.14%	86.79%	0.12	25.67%	75.36%	0.07	19.64%	60.54%	0.05	17.44%	40.36%	0.05	16.84%	26.96%	0.04	16.23%	8.75%	0.04	16.36%	2.86%	0.04	16.41%	3.04%	0.04	12.33%	37.32%	0.04
MinR	38.93%	86.96%	0.10	27.78%	78.21%	0.07	18.41%	62.14%	0.05	17.11%	41.96%	0.04	15.92%	29.64%	0.04	15.72%	10.89%	0.04	15.85%	3.04%	0.04	15.86%	3.39%	0.04	12.00%	34.64%	0.04
MaxFTime	45.80%	85.00%	0.13	35.62%	76.43%	0.07	29.52%	64.46%	0.05	26.00%	43.21%	0.05	25.37%	27.86%	0.04	25.00%	10.89%	0.04	24.97%	3.21%	0.04	24.91%	4.11%	0.04	17.71%	37.68%	0.04
MaxFTime	44.84%	85.00%	0.12	36.17%	77.14%	0.07	28.79%	64.64%	0.05	25.47%	45.71%	0.04	24.65%	30.36%	0.04	24.53%	10.84%	0.04	24.79%	3.39%	0.04	24.71%	4.11%	0.04	16.48%	38.39%	0.04
MaxPWS+	48.13%	85.71%	0.18	40.74%	79.11%	0.11	35.14%	71.25%	0.06	31.67%	54.64%	0.05	29.79%	38.93%	0.04	29.22%	24.29%	0.04	29.67%	8.04%	0.04	29.25%	18.21%	0.04	26.47%	36.25%	0.04
MaxPWS-	49.23%	84.11%	0.16	43.58%	78.75%	0.14	39.75%	69.29%	0.09	37.02%	55.89%	0.05	34.91%	41.96%	0.04	33.77%	28.39%	0.04	34.24%	9.64%	0.04	33.91%	20.71%	0.04	31.60%	38.21%	0.04
MaxPWD+	39.79%	86.79%	0.10	30.22%	79.46%	0.07	23.88%	66.07%	0.05	19.99%	49.82%	0.04	18.68%	32.86%	0.04	17.82%	12.14%	0.04	17.50%	2.50%	0.04	17.66%	6.25%	0.04	12.47%	30.89%	0.04
MaxPWD-	36.56%	85.71%	0.10	26.94%	80.71%	0.06	20.54%	66.96%	0.05	16.47%	52.50%	0.05	15.46%	34.46%	0.04	15.35%	13.93%	0.04	15.11%	2.14%	0.04	15.19%	8.21%	0.04	9.97%	27.50%	0.04

Tabela 23: Resultados para o método baseado em busca em vizinhança com o critério 2 com $K_{max} = 2$

Tabela 24: Resultados para o método baseado em busca em vizinhança com o critério 3 com $K_{max} = 2$																							
MaxTasks			MaxTasks1.2			MaxTasks1.4			MaxTasks1.6			MaxTasks1.8			MaxTasks2			MaxTasks3			MaxTaskK		
	C	P	T	C	P	T	C	P	T	C	P	T	C	P	T	C	P	T	C	P	T		
MinD	32.35%	74.29%	0.33	23.88%	66.61%	0.15	18.35%	52.32%	0.09	16.53%	37.32%	0.08	16.03%	27.86%	0.07	15.61%	14.11%	0.06	15.82%	10.54%	11.96%	37.86%	
MinR	32.97%	75.71%	0.29	24.60%	71.25%	0.15	17.41%	53.57%	0.09	16.23%	39.46%	0.07	14.92%	33.21%	0.07	14.98%	18.01%	0.06	15.10%	13.01%	12.68%		
MaxFTime	42.14%	73.04%	0.36	33.40%	65.36%	0.21	27.89%	57.50%	0.10	24.73%	44.82%	0.08	24.21%	35.71%	0.07	23.48%	25.00%	0.07	23.95%	19.64%	16.95%		
MaxFTime	41.18%	75.54%	0.36	33.82%	67.32%	0.22	27.91%	57.50%	0.09	24.41%	45.54%	0.08	23.60%	33.39%	0.07	23.66%	19.46%	0.07	23.91%	14.82%	14.25%		
MaxPWS+	41.57%	74.64%	0.40	38.06%	67.50%	0.24	33.33%	62.50%	0.13	30.95%	52.08%	0.08	29.37%	38.93%	0.06	28.96%	26.79%	0.06	29.50%	12.32%	26.28%		
MaxPWS-	43.31%	72.32%	0.37	40.84%	69.29%	0.31	37.86%	59.82%	0.19	36.39%	51.79%	0.09	34.31%	41.07%	0.07	33.24%	28.99%	0.05	34.08%	11.96%	31.49%		
MaxPWD+	33.56%	74.29%	0.29	28.17%	70.89%	0.16	22.71%	60.54%	0.09	19.19%	44.82%	0.08	17.77%	32.32%	0.07	17.32%	13.93%	0.05	17.82%	6.79%	12.17%		
MaxPWD-	30.50%	72.86%	0.27	24.55%	69.82%	0.16	19.47%	58.75%	0.11	15.83%	46.07%	0.08	14.84%	33.39%	0.07	15.14%	14.11%	0.06	14.93%	4.82%	9.78%		

Tabela 24: Resultados para o método baseado em busca em vizinhança com o critério 3 com $K_{max} = 2$

Tabela 25: Resultados para o método baseado em busca em vizinhança com o critério 1 com $K_{Max} = 3$

	MaxTasks			MaxTasks1.2			MaxTasks1.4			MaxTasks1.6			MaxTasks1.8			MaxTasks2			MaxTasks3			MaxTasksK			MinRLB		
	C	P	T	C	P	T	C	P	T	C	P	T	C	P	T	C	P	T	C	P	T	C	P	T	C	P	T
MinD	53.32%	87.81%	0.20	42.90%	78.75%	0.15	31.69%	68.28%	0.10	26.31%	55.94%	0.08	22.95%	42.34%	0.07	19.91%	21.88%	0.07	18.66%	5.94%	0.06	18.60%	5.78%	0.06	15.18%	39.38%	0.06
MinR	55.43%	87.97%	0.18	44.89%	81.56%	0.16	33.80%	69.69%	0.11	25.57%	54.06%	0.08	21.99%	41.72%	0.07	19.61%	22.34%	0.07	17.96%	6.25%	0.06	17.82%	5.78%	0.06	14.86%	35.94%	0.06
MaxIFTime	59.92%	86.25%	0.19	51.66%	79.69%	0.17	43.29%	70.47%	0.13	36.17%	56.56%	0.09	32.29%	44.84%	0.07	28.46%	23.59%	0.07	26.42%	6.09%	0.06	26.27%	6.72%	0.06	20.12%	39.69%	0.06
MaxFTime	59.77%	86.25%	0.19	50.90%	80.94%	0.17	43.25%	70.63%	0.11	36.45%	57.66%	0.09	31.85%	46.41%	0.07	28.12%	22.66%	0.06	26.43%	7.03%	0.06	26.10%	7.19%	0.06	18.76%	39.53%	0.06
MaxPWS+	60.06%	86.88%	0.20	53.16%	82.97%	0.20	47.93%	75.78%	0.16	42.66%	65.94%	0.11	38.85%	52.97%	0.08	34.17%	37.03%	0.06	29.93%	13.75%	0.05	30.02%	19.53%	0.05	27.65%	38.44%	0.05
MaxPWS-	60.56%	86.09%	0.19	53.83%	82.19%	0.20	49.69%	76.09%	0.19	45.62%	67.34%	0.16	42.55%	55.16%	0.11	38.84%	39.38%	0.08	34.45%	16.72%	0.06	34.85%	22.81%	0.06	33.11%	40.31%	0.05
MaxPWD+	56.54%	88.28%	0.18	47.91%	82.50%	0.16	40.41%	73.59%	0.12	33.97%	63.75%	0.08	28.16%	51.09%	0.07	24.02%	27.50%	0.06	19.45%	5.47%	0.05	19.46%	7.81%	0.05	14.79%	32.34%	0.05
MaxPWD-	55.08%	88.28%	0.17	41.99%	83.44%	0.18	37.73%	74.84%	0.12	30.18%	63.13%	0.09	24.68%	50.94%	0.08	20.00%	25.78%	0.06	16.86%	5.47%	0.05	16.80%	10.16%	0.06	12.45%	29.38%	0.05

Tabela 26: Resultados para o método baseado em busca em vizinhança com o critério 2 com $K_{Max} = 3$

	MaxTasks			MaxTasks1.2			MaxTasks1.4			MaxTasks1.6			MaxTasks1.8			MaxTasks2			MaxTasks3			MaxTasksK			MinRLB		
	C	P	T	C	P	T	C	P	T	C	P	T	C	P	T	C	P	T	C	P	T	C	P	T	C	P	T
MinD	47.42%	87.81%	7.99	37.91%	78.75%	0.85	27.90%	68.28%	0.25	23.06%	55.94%	0.18	20.16%	42.34%	0.15	17.60%	21.88%	0.12	16.50%	5.94%	0.11	16.42%	5.78%	0.11	13.34%	39.38%	0.13
MinR	49.09%	87.97%	0.48	40.09%	81.56%	3.97	30.89%	69.69%	0.46	23.16%	54.06%	0.16	19.65%	41.72%	0.14	17.79%	22.34%	0.12	16.30%	6.25%	0.10	16.18%	5.78%	0.10	13.43%	35.94%	0.11
MaxIFTime	52.70%	86.25%	7.64	45.19%	79.69%	0.48	38.59%	70.47%	0.33	32.90%	56.56%	0.18	28.91%	44.84%	0.15	25.51%	23.59%	0.12	23.68%	6.09%	0.10	23.55%	6.72%	0.11	18.21%	39.69%	0.12
MaxFTime	52.06%	86.25%	4.25	44.48%	80.94%	7.64	38.44%	70.63%	7.45	32.75%	57.66%	3.75	28.42%	46.41%	0.14	25.00%	22.66%	0.12	23.30%	7.03%	0.11	22.99%	7.19%	0.11	16.96%	39.53%	0.12
MaxPWS+	52.38%	86.88%	0.60	46.54%	82.97%	0.52	42.01%	75.78%	0.46	37.40%	65.94%	0.29	34.28%	52.97%	0.19	31.01%	37.03%	0.13	27.29%	13.75%	0.09	27.24%	19.53%	0.10	25.69%	38.44%	0.09
MaxPWS-	52.28%	86.09%	0.57	47.15%	82.19%	0.51	43.67%	76.09%	0.46	39.77%	67.34%	0.39	37.51%	55.16%	0.27	34.89%	39.38%	0.17	31.62%	16.72%	0.10	32.03%	22.81%	0.11	31.10%	40.31%	0.10
MaxPWD+	49.81%	88.28%	0.57	43.12%	82.50%	0.39	36.32%	73.59%	0.28	30.42%	63.75%	0.18	25.55%	51.09%	0.13	21.93%	27.50%	0.11	17.45%	5.47%	0.09	17.41%	7.81%	0.09	13.62%	32.34%	0.09
MaxPWD-	48.73%	88.28%	4.17	40.06%	83.44%	0.55	33.73%	74.84%	0.30	27.32%	63.13%	0.19	22.44%	50.94%	0.15	18.03%	25.78%	0.12	14.98%	5.47%	0.09	14.86%	10.16%	0.11	11.18%	29.38%	0.09

Tabela 27: Resultados para o método baseado em busca em vizinhança com o critério 3 com $K_{Max} = 3$

	MaxTasks			MaxTasks1.2			MaxTasks1.4			MaxTasks1.6			MaxTasks1.8			MaxTasks2			MaxTasks3			MaxTasksK			MinRLB		
	C	P	T	C	P	T	C	P	T	C	P	T	C	P	T	C	P	T	C	P	T	C	P	T	C	P	T
MinD	45.83%	80.31%	1.59	39.06%	72.81%	1.40	29.23%	63.44%	0.95	24.33%	53.75%	0.85	21.54%	43.13%	0.41	19.07%	26.25%	0.25	18.16%	12.81%	0.08	18.05%	11.41%	0.08	14.69%	39.69%	0.13
MinR	46.68%	79.06%	1.61	40.04%	74.22%	1.41	31.46%	66.72%	0.94	24.32%	51.72%	0.75	20.91%	44.69%	0.35	18.76%	27.81%	0.22	17.27%	14.84%	0.09	17.12%	13.91%	0.08	14.55%	35.47%	0.11
MaxIFTime	52.60%	76.88%	1.66	46.81%	70.47%	1.41	40.60%	66.25%	0.78	34.29%	58.44%	0.48	31.03%	50.16%	0.32	26.99%	34.38%	0.23	25.25%	20.16%	0.09	25.02%	20.31%	0.10	19.32%	42.19%	0.14
MaxFTime	52.10%	77.66%	1.65	46.00%	72.66%	1.36	40.78%	65.00%	0.51	34.58%	56.72%	0.53	30.59%	49.06%	0.37	26.93%	30.63%	0.22	25.63%	17.19%	0.11	25.31%	16.72%	0.08	18.12%	42.03%	0.11
MaxPWS+	49.84%	76.72%	1.62	46.26%	73.59%	1.44	43.92%	68.75%	1.10	40.04%	60.94%	0.77	37.31%	51.25%	0.55	33.41%	39.84%	0.37	29.89%	16.88%	0.07	29.85%	21.88%	0.07	27.48%	40.63%	0.08
MaxPWS-	50.15%	75.78%	1.45	47.19%	73.44%	1.33	45.50%	69.38%	1.07	42.74%	61.88%	0.87	40.52%	52.97%	0.74	37.71%	39.06%	0.44	34.34%	18.59%	0.09	34.70%	23.44%	0.08	32.94%	40.94%	0.08
MaxPWD+	47.32%	77.81%	1.46	42.03%	72.34%	1.28	37.35%	67.03%	0.86	31.82%	59.06%	0.65	27.02%	49.22%	0.41	23.23%	28.59%	0.25	19.26%	8.91%	0.10	19.29%	9.69%	0.06	14.54%	35.00%	0.07
MaxPWD-	45.75%	78.59%	1.43	40.67%	73.91%	1.27	35.15%	68.13%	0.93	28.84%	59.38%	0.60	23.91%	50.16%	0.36	19.55%	26.88%	0.21	16.70%	7.66%	0.08	16.60%	10.78%	0.07	12.25%	32.03%	0.07

para os mesmos critérios utilizando $K_{Max} = 3$. O tempo de execução da família Wee-mag cresceu muito rapidamente, em geral ultrapassando uma hora de execução. Os testes para esta família ainda se encontram em execução e, em virtude disto, os resultados das tabelas a seguir correspondem às 7 famílias restantes.

Os critérios 4, 5 e 6 conseguiram obter resultados melhores que os critérios 1, 2 e 3, em média. O critério 4 obteve os melhores resultados entre todos os critérios, especialmente quando utilizando as soluções obtidas pela heurística construtiva com os critérios MinRLB e MaxPWD-. As melhores soluções encontradas utilizando estes critérios estavam respectivamente, a 11,63%, 8,07% e 9,08% para os critérios 4, 5 e 6.

Conforme dito anteriormente, o tempo de execução do método para estes critérios é maior que o tempo de execução para os critérios 1 a 3. A maioria das instâncias foi resolvida em menos de 10 s, porém em alguns casos o tempo de execução chegou a 20 min. Nestas instâncias, a solução encontrada pode ser melhorada várias vezes, resultando num tempo de execução maior.

Para a família Tonge, o critério 5 encontrou cinco soluções cujos tempos de execução são menores que os tempos de execução das melhores soluções conhecidas para as respectivas instâncias, com melhoria média de 2,68%.

Tabela 28: Resultados para o método baseado em busca em vizinhança com o critério 4 com $K_{Max} = 2$

	MaxTasks			MaxTasks1.2			MaxTasks1.4			MaxTasks1.6			MaxTasks1.8			MaxTasks2			MaxTasks3			MaxTasksK			MinRLB		
	C	P	T	C	P	T	C	P	T	C	P	T	C	P	T	C	P	T	C	P	T	C	P	T	C	P	T
MinD	42.37%	86.79%	0.15	29.68%	77.50%	0.14	22.28%	64.64%	0.13	19.66%	46.96%	0.14	18.40%	33.75%	0.13	17.59%	14.11%	0.13	17.59%	5.36%	0.12	17.84%	6.07%	0.12	13.83%	37.68%	0.14
MinR	42.45%	88.59%	0.13	30.49%	80.94%	0.12	20.23%	66.72%	0.12	17.74%	46.72%	0.14	16.30%	32.50%	0.12	16.15%	14.38%	0.12	16.24%	4.69%	0.12	16.34%	5.78%	0.12	13.37%	34.69%	0.11
MaxFTime	48.42%	87.19%	0.13	37.42%	80.00%	0.13	29.59%	67.66%	0.14	25.94%	47.34%	0.89	24.78%	32.97%	0.89	24.18%	13.44%	0.93	24.14%	5.31%	0.94	24.23%	6.09%	0.93	18.08%	39.22%	0.12
MaxFTime	48.02%	86.88%	0.14	38.03%	80.00%	0.13	29.30%	69.22%	0.14	25.30%	50.63%	0.14	23.85%	34.38%	0.15	23.66%	14.84%	0.19	23.55%	5.94%	0.19	23.58%	7.19%	0.19	17.13%	37.97%	0.14
MaxPWS+	51.90%	87.50%	0.13	43.06%	81.72%	0.14	36.20%	74.53%	0.14	31.12%	59.06%	0.14	28.29%	44.53%	0.17	27.13%	26.09%	0.18	26.65%	8.44%	0.22	27.20%	18.75%	0.19	25.02%	37.81%	0.18
MaxPWS-	52.91%	86.41%	0.12	46.16%	81.41%	0.13	40.38%	72.97%	0.13	36.32%	59.22%	0.12	32.97%	45.16%	0.17	31.55%	31.25%	0.18	29.97%	9.84%	0.25	30.99%	22.34%	0.22	29.87%	40.16%	0.17
MaxPWD+	42.88%	88.44%	0.11	32.00%	81.72%	0.15	24.32%	69.38%	0.19	19.83%	53.75%	1.88	18.58%	36.09%	0.18	17.16%	15.47%	0.17	16.48%	4.06%	0.17	16.77%	7.81%	0.17	13.34%	31.88%	0.14
MaxPWD-	40.31%	87.19%	0.11	29.95%	81.88%	0.13	22.18%	69.84%	0.13	17.51%	56.25%	0.23	15.85%	39.53%	0.15	15.00%	17.50%	0.26	14.76%	3.59%	0.34	14.79%	10.31%	0.25	11.63%	29.53%	0.12

Tabela 29: Resultados para o método baseado em busca em vizinhança com o critério 5 com $K_{Max} = 2$

	MaxTasks			MaxTasks1.2			MaxTasks1.4			MaxTasks1.6			MaxTasks1.8			MaxTasks2			MaxTasks3			MaxTasksK			MinRLB		
	C	P	T	C	P	T	C	P	T	C	P	T	C	P	T	C	P	T	C	P	T	C	P	T	C	P	T
MinD	29.47%	84.58%	0.43	22.08%	73.75%	0.30	15.79%	59.79%	0.36	13.96%	44.79%	0.34	12.99%	32.71%	0.41	12.33%	13.54%	0.40	12.15%	4.17%	0.42	12.51%	6.25%	0.41	8.31%	36.88%	0.25
MinR	29.63%	86.96%	0.31	21.50%	78.21%	0.37	15.26%	64.11%	0.51	13.03%	44.29%	0.54	12.02%	30.36%	0.56	11.96%	13.39%	0.51	11.85%	4.11%	0.54	12.04%	6.25%	0.52	8.14%	33.75%	0.32
MaxFTime	32.52%	85.36%	0.24	24.95%	77.14%	0.59	20.03%	63.39%	0.32	18.91%	45.00%	0.52	18.01%	31.43%	0.33	17.92%	13.75%	0.32	17.85%	5.18%	0.28	17.87%	6.79%	0.28	11.79%	38.21%	0.51
MaxFTime	32.80%	85.00%	0.25	25.88%	77.14%	1.00	20.02%	65.18%	0.48	18.26%	47.32%	0.71	17.20%	33.21%	0.62	17.23%	15.18%	0.61	16.73%	5.71%	0.67	16.87%	8.04%	0.71	10.94%	36.07%	0.25
MaxPWS+	31.72%	85.71%	0.24	23.91%	79.11%	0.29	19.69%	70.89%	0.25	16.43%	53.21%	0.29	14.79%	36.79%	0.47	13.93%	17.68%	0.47	13.12%	3.57%	0.63	13.94%	11.25%	0.52	9.90%	30.18%	0.83
MaxPWS-	32.02%	84.46%	0.24	25.32%	78.75%	0.25	20.50%	69.11%	0.42	17.45%	53.39%	0.37	15.58%	37.50%	0.30	13.89%	22.32%	0.55	13.10%	4.11%	0.68	13.70%	15.18%	0.55	11.24%	32.14%	0.78
MaxPWD+	28.68%	86.79%	0.25	20.57%	79.11%	0.28	16.07%	65.18%	0.24	13.47%	51.25%	0.30	12.64%	35.36%	0.35	11.48%	16.61%	0.31	10.82%	3.93%	0.45	11.22%	8.75%	0.44	8.22%	30.71%	0.38
MaxPWD-	28.04%	85.36%	0.24	20.08%	79.29%	0.31	15.91%	66.25%	0.92	12.91%	54.46%	0.30	11.40%	39.29%	0.30	10.78%	18.57%	2.66	10.44%	3.39%	2.97	10.54%	11.61%	2.80	8.07%	30.00%	0.53

Tabela 30: Resultados para o método baseado em busca em vizinhança com o critério 6 com $K_{Max} = 2$

	MaxTasks			MaxTasks1.2			MaxTasks1.4			MaxTasks1.6			MaxTasks1.8			MaxTasks2			MaxTasks3			MaxTasksK			MinRLB		
	C	P	T	C	P	T	C	P	T	C	P	T	C	P	T	C	P	T	C	P	T	C	P	T	C	P	T
MinD	27.72%	73.33%	0.75	22.65%	67.08%	0.44	17.37%	54.79%	0.65	15.62%	44.79%	0.58	14.87%	37.29%	0.58	14.09%	25.42%	0.49	14.23%	20.21%	0.48	14.54%	21.46%	0.47	10.12%	40.00%	1.09
MinR	27.31%	75.18%	0.60	21.45%	71.61%	0.57	15.80%	57.14%	0.74	13.99%	46.61%	0.62	12.62%	38.39%	0.69	12.47%	26.79%	0.77	12.62%	21.96%	0.77	12.64%	23.39%	0.78	9.57%	39.46%	1.14
MaxFTime	32.97%	73.04%	0.61	26.40%	67.68%	0.45	22.41%	57.50%	0.68	21.12%	49.11%	0.73	20.41%	43.04%	0.75	19.43%	35.36%	0.76	19.42%	30.36%	1.21	19.39%	31.61%	0.76	13.40%	42.86%	1.04
MaxFTime	32.86%	75.54%	0.84	26.31%	67.86%	0.77	21.99%	59.64%	0.78	20.22%	50.00%	0.65	19.30%	41.07%	1.19	19.22%	32.86%	1.35	19.22%	25.89%	3.73	19.34%	26.96%	3.39	12.76%	41.25%	3.14
MaxPWS+	29.46%	76.25%	0.75	25.06%	70.18%	0.91	21.00%	61.79%	1.29	18.85%	56.07%	1.62	17.36%	45.54%	2.07	16.78%	35.00%	2.05	16.06%	27.50%	2.34	16.89%	31.43%	2.13	12.45%	45.89%	4.42
MaxPWS-	29.68%	74.46%	0.65	26.27%	71.25%	0.39	21.44%	63.57%	1.02	19.65%	55.00%	1.25	17.84%	43.75%	2.11	16.65%	35.71%	2.38	16.02%	26.96%	1.94	16.88%	30.00%	1.93	13.78%	45.36%	5.07
MaxPWD+	25.81%	73.93%	0.50	20.75%	71.07%	0.71	17.04%	59.64%	2.81	14.54%	49.64%	1.09	13.51%	41.79%	2.56	12.86%	28.75%	5.62	12.26%	22.14%	7.74	12.58%	23.39%	8.54	9.61%	38.39%	7.07
MaxPWD-	25.08%	72.50%	1.02	20.13%	69.11%	1.11	16.73%	60.89%	1.49	13.71%	49.64%	1.48	12.59%	42.68%	3.39	12.04%	30.36%	3.85	11.86%	19.64%	4.23	11.82%	24.82%	4.33	9.40%	38.04%	2.87

		MaxTasks			MaxTasks1.2			MaxTasks1.4			MaxTasks1.6			MaxTasks1.8			MaxTasks2			MaxTasks3			MaxTasksK			MinRLB		
		C	P	T	C	P	T	C	P	T	C	P	T	C	P	T	C	P	T	C	P	T	C	P	T	C	P	T
MinD		52.24%	87.81%	0.16	41.35%	78.75%	0.15	30.19%	68.28%	0.15	24.70%	55.94%	0.13	21.24%	42.34%	0.13	18.13%	21.88%	0.13	16.91%	5.94%	0.11	16.88%	5.78%	0.11	13.41%	39.38%	0.13
MinR		54.33%	87.97%	0.16	43.62%	81.56%	0.15	32.31%	69.69%	0.14	23.98%	54.06%	0.14	20.47%	41.72%	0.12	18.00%	22.34%	0.12	16.37%	6.25%	0.11	16.33%	5.78%	0.11	13.27%	35.94%	0.11
MaxFTime		58.88%	86.25%	0.14	50.21%	79.69%	0.15	41.92%	70.47%	0.16	34.26%	56.56%	0.15	29.95%	44.84%	0.15	26.21%	23.59%	0.89	24.05%	6.09%	0.94	23.97%	6.56%	0.93	18.18%	39.69%	0.42
MaxFTime		58.57%	86.25%	0.15	49.26%	80.94%	0.16	41.73%	70.63%	0.15	34.74%	57.66%	0.16	30.03%	46.41%	0.16	25.88%	22.50%	0.24	24.40%	7.03%	0.19	24.09%	7.03%	0.18	17.05%	39.53%	0.13
MaxPWS+		59.22%	86.88%	0.16	52.27%	82.97%	0.16	46.22%	75.78%	0.15	40.38%	65.94%	0.15	36.08%	52.97%	0.13	31.04%	37.03%	0.15	26.16%	13.75%	0.24	26.24%	19.53%	0.20	24.52%	38.59%	0.18
MaxPWS-		59.49%	86.09%	0.16	52.96%	82.19%	0.16	48.07%	76.09%	0.15	43.53%	67.34%	0.15	40.05%	55.16%	0.14	35.39%	39.22%	0.15	30.30%	16.72%	0.19	30.55%	22.97%	0.21	29.93%	40.47%	0.16
MaxPWD+		55.37%	88.28%	0.15	46.92%	82.50%	0.14	38.74%	73.59%	0.15	31.19%	63.75%	0.16	25.53%	51.09%	0.17	21.32%	27.50%	0.15	17.03%	5.31%	0.17	16.97%	7.81%	0.17	13.27%	32.50%	0.13
MaxPWD-		54.13%	88.28%	0.13	44.01%	83.44%	0.14	36.49%	74.84%	0.14	28.24%	63.13%	0.15	22.50%	50.94%	0.14	17.76%	25.78%	0.15	14.50%	5.31%	0.35	14.57%	10.16%	0.25	11.29%	29.38%	0.13

Tabela 31: Resultados para o método baseado em busca em vizinhança com o critério 4 com $K_{max} = 3$

		MaxTasks			MaxTasks1.2			MaxTasks1.4			MaxTasks1.6			MaxTasks1.8			MaxTasks2			MaxTasks3			MaxTasksK			MinRLB		
		C	P	T	C	P	T	C	P	T	C	P	T	C	P	T	C	P	T	C	P	T	C	P	T	C	P	T
MinD		35.54%	86.07%	0.63	26.67%	75.71%	2.15	21.08%	63.75%	0.33	17.92%	51.61%	0.32	15.98%	38.75%	0.34	13.68%	21.43%	0.32	12.48%	4.46%	0.35	12.48%	5.89%	0.34	8.91%	38.75%	0.19
MinR		37.16%	86.25%	0.23	28.31%	78.03%	0.23	21.87%	65.54%	0.32	17.38%	49.29%	1.03	15.52%	38.57%	0.56	13.93%	20.54%	0.61	12.64%	5.71%	0.50	12.53%	6.25%	0.49	8.60%	34.64%	0.28
MaxFTime		40.52%	84.29%	0.21	32.23%	76.79%	0.25	26.86%	66.25%	2.32	22.30%	50.54%	0.67	20.63%	39.29%	0.31	19.45%	22.52%	0.30	18.17%	5.89%	0.25	18.03%	7.50%	0.25	12.31%	38.57%	0.51
MaxFTime		40.15%	84.29%	0.22	32.00%	78.21%	0.26	26.18%	66.43%	2.28	22.85%	52.14%	1.25	20.84%	40.54%	0.62	19.37%	21.79%	0.58	18.62%	6.96%	0.62	18.18%	8.04%	0.67	11.46%	37.50%	0.24
MaxPWS+		39.28%	85.00%	0.20	32.41%	80.54%	0.20	26.64%	72.32%	0.22	21.98%	61.07%	0.27	18.88%	46.25%	0.28	16.63%	28.04%	0.36	14.22%	4.29%	0.70	14.40%	11.43%	0.47	10.62%	30.36%	0.79
MaxPWS-		39.00%	84.11%	0.19	32.67%	79.64%	0.20	27.91%	72.08%	0.22	23.68%	62.68%	0.26	20.78%	48.75%	0.29	17.43%	30.71%	0.59	13.79%	5.89%	0.68	14.20%	15.18%	0.51	11.85%	31.79%	0.75
MaxPWD+		37.38%	86.61%	1.29	30.00%	80.00%	1.30	23.06%	69.82%	0.27	19.23%	58.75%	0.26	16.45%	46.25%	0.29	14.78%	25.89%	0.27	11.97%	5.54%	0.40	12.00%	8.75%	0.39	8.76%	31.43%	0.35
MaxPWD-		37.06%	86.61%	0.20	27.52%	81.07%	0.22	23.13%	71.25%	0.22	18.65%	58.75%	0.27	15.34%	46.61%	0.31	13.09%	25.18%	0.31	11.00%	5.54%	2.79	10.91%	11.43%	2.72	8.43%	29.64%	0.51

Tabela 32: Resultados para o método baseado em busca em vizinhança com o critério 5 com $K_{max} = 3$

		MaxTasks			MaxTasks1.2			MaxTasks1.4			MaxTasks1.6			MaxTasks1.8			MaxTasks2			MaxTasks3			MaxTasksK			MinRLB		
		C	P	T	C	P	T	C	P	T	C	P	T	C	P	T	C	P	T	C	P	T	C	P	T	C	P	T
MinD		31.36%	76.96%	2.34	25.92%	70.36%	1.82	21.30%	59.46%	2.36	18.40%	53.57%	3.31	16.57%	45.54%	2.82	14.65%	33.21%	1.98	13.89%	21.96%	2.59	13.82%	22.14%	2.73	9.97%	44.11%	1.91
MinR		31.71%	76.07%	2.09	26.59%	71.25%	2.06	21.04%	63.57%	3.80	18.00%	50.71%	1.38	16.28%	47.32%	1.11	14.51%	35.71%	1.66	12.82%	25.00%	5.12	12.77%	25.18%	5.12	9.50%	40.80%	4.99
MaxFTime		37.42%	74.29%	2.01	31.12%	68.57%	1.80	27.40%	62.86%	2.67	23.66%	56.79%	5.59	22.52%	50.00%	6.33	20.97%	40.36%	2.22	19.84%	29.64%	2.85	19.56%	30.89%	2.16	13.55%	43.75%	4.57
MaxFTime		36.74%	74.82%	2.05	30.36%	70.36%	2.18	26.97%	61.79%	1.23	23.76%	53.75%	2.87	22.41%	47.86%	5.14	20.97%	38.21%	1.69	20.29%	28.39%	2.57	19.91%	28.75%	5.15	12.69%	43.39%	6.99
MaxPWS+		32.78%	73.93%	2.27	29.06%	71.96%	2.27	26.97%	64.82%	2.38	22.63%	57.68%	2.27	20.36%	52.52%	3.00	18.54%	42.86%	2.23	16.29%	23.57%	6.17	16.48%	29.29%	10.58	12.61%	44.64%	13.16
MaxPWS-		32.38%	73.21%	2.06	29.33%	70.71%	2.10	26.42%	65.89%	2.86	23.59%	60.89%	2.01	21.10%	54.82%	4.92	18.83%	44.29%	8.68	15.93%	28.21%	9.98	16.45%	32.50%	8.45	13.61%	43.75%	13.77
MaxPWD+		31.15%	73.93%	2.58	26.42%	68.99%	2.41	22.82%	62.14%	1.85	18.83%	58.21%	2.52	17.15%	49.46%	2.02	15.15%	36.25%	4.24	13.13%	23.57%	7.59	12.94%	23.39%	10.73	9.71%	39.82%	17.74
MaxPWD-		30.94%	74.64%	1.87	25.63%	69.11%	2.04	22.22%	64.61%	5.58	19.06%	58.39%	2.20	15.86%	50.71%	2.17	14.12%	35.54%	8.72	11.80%	21.79%	10.72	11.77%	25.36%	8.70	9.08%	37.86%	7.47

Tabela 33: Resultados para o método baseado em busca em vizinhança com o critério 6 com $K_{max} = 3$