DÉBORA ALVES RIBEIRO

SEQUENCIAMENTO DE MÁQUINAS PARALELAS NÃO RELACIONADAS COM TEMPO DE PREPARAÇÃO DEPENDENTES DA SEQUÊNCIA E DA MÁQUINA

Belo Horizonte
31 de agosto de 2015

DÉBORA ALVES RIBEIRO

SEQUENCIAMENTO DE MÁQUINAS PARALELAS NÃO RELACIONADAS COM TEMPO DE PREPARAÇÃO DEPENDENTES DA SEQUÊNCIA E DA MÁQUINA

Dissertação apresentada ao Curso de Pós-Graduação em Engenharia de Produção da Universidade Federal de Minas Gerais como requisito parcial para a obtenção do grau de Mestre em Engenharia de Produção.

Orientador: Ricardo Saraiva de Camargo

Belo Horizonte

31 de agosto de 2015

FOLHA DE APROVAÇÃO

Sequenciamento de máquinas paralelas não relacionadas com tempo de preparação dependentes da sequência e da máquina

DEBORA ALVES RIBEIRO

Dissertação submetida à Banca Examinadora designada pelo Colegiado do Programa de Pós-Graduação em ENGENHARIA DE PRODUÇÃO, como requisito para obtenção do grau de Mestre em ENGENHARIA DE PRODUÇÃO, área de concentração PESQUISA OPERACIONAL E ENGENHARIA DE MANUFATURA, linha de pesquisa Otimização de

Aprovada em 31 de agosto de 2015, pela banca constituída pelos membros:

Prof(a). Ricardo Saraiva de Camargo - Orientador
UFMG

Prof(a). Carlos Roberto Venâncio de Carvalho
UFMG

Prof(a). Elisangela Martins de Sá
CEFET-MG

Belo Horizonte, 31 de agosto de 2015.

Resumo

Pesquisas sobre problemas de sequenciamentos de máquinas paralelas são concentrados em sua maioria em heurísticas, devido à sua natureza teórica e desafiante. Apenas alguns poucos trabalhos possuem abordagens exatas, e a maioria deles restrigem-se ao ambiente que envolve sequenciamento de máquinas paralelas idênticas. Este trabalho aborda o problema de sequenciamento de máquinas paralelas não relacionadas com tempos de preparação dependentes da sequência e da máquina. A função objetivo é minimizar a soma ponderada dos tempos de conclusão das tarefas. Este problema é pouco estudado na literatura, havendo um número restrito de pesquisas envolvendo-o heurísticamente e não foi encontrado trabalhos que o aborde utilizando um método exato para sua resolução. Neste contexto, seis formulações de programação inteira mista (PIM) foram adaptadas e traduzidas para o problema. Esta pesquisa apresenta uma nova formulação matemática para o modelo e devido suas características foi aplicado e desenvolvido um algoritmo variante do método de decomposição de Benders. Um método de decomposição *logic-based* Benders da literatufa foi adaptado e comparado com o algoritmo mencionado anteriormente. Resultados computacionais mostram que a nova formulação tem um comportamento melhor que cinco entre as seis encontradas na literatura. Sobre os algoritmos comparados o primeiro tem um comportamento mais atraente e ambos salientam a necessidade de mais pesquisas envolvendo métodos exatos.

Palavras-chaves: Máquinas Paralelas Não Relacionadas; Tempos de Preparação; Método de Decomposição de Benders.

Abstract

Parallel machine scheduling researches are mostly concentrated on heuristics, because of their theoretical and challenging feature. Only few papers present exact approaches, and most of them are restricted to environment about identical parallel machine scheduling. This work studies a scheduling problem with unrelated parallel machines, sequence and machine dependent setup times. The objective function is minimize the total weighted completion times. This problem is not very explored in literature. There is a limited number of researches that study the problem from a heuristic point of view. Works that analyse the problem utilizing exact methods were not found. In this context, six mixed integer programing (MIP) scheduling formulations of the literature were adapted to the problem. This paper presents a new mathematical model. Due to the characteristics of the formulation developed, an algorithm variants based on the Benders decomposition method are presented to solve the problem. A logic-based Benders decomposition approach of the literature was adapted and compared with the algorithm previously mentioned. Computational results show that the new formulation have a better behavior than five of the six formulations found in the literature. About the two algorithms implemented, the first has a more attractive and behavior and both point out for the need of the more research involving exact methods.

Keywords: Unrelated Parallel Machines; Setup Time; Benders Decomposition Method.

Aos meus pais,
Nelson e Maria do Socorro.

"Há um tempo para cada coisa, diz o Eclesiastes. Nas letras há um tempo para ler e um tempo para ruminar o que se leu; há um tempo para se engravidar de uma intuição, idéia, imagem ou história, e um tempo para levá-las ao papel em sinais gráficos. Não vazes o teu talento em ansiedade. Dedica-te pacientemente ao trabalho e tem coragem de soltar feras, anjo, legiões de dêmonios e fantasmas que povoam cavernas e vulcões de tua topografia interior." (Frei Betto)

Agradecimentos

Eu gostaria de agradecer primeiramente a **Deus** e a **Maria** nossa mãe, meus grandes parceiros nessa caminhada.

Ao Prof. Ricardo Saraiva de Camargo, meu orientador, pela paciência, motivação e por compartilhar seus conhecimento levando-me ao crescimento profissional e pessoal.

Ao meus pais Nelson e Maria do Socorro, pelo amor, incentivo, orações e apoio. Obrigada por serem minha base e minha inspiração.

À minha irmã Regiane pelo apoio, paciência e por me acolher em sua casa. Aos meus irmãos Douglas, Nelson e Dyemes por me apoiarem sempre, pelas palavras de carinho e por me incentivarem em minhas buscas pessoais.

Aos meus amigos de Guanhães, Viçosa e Belo Horizonte pelo grande apoio e principalmente pela compreensão nos momentos em que me ausentei para me dedicar a este trabalho.

À FAPEMIG pelo incentivo financeiro concedido.

Sumário

Lista de Figuras

Lista de Tabelas

Lista de Algoritmos

Capítulo 1

Introdução

Os estudos de problemas de sequenciamento tiveram suas primeiras publicações na década de cinquenta (Johnson, 1954; Jackson, 1955; Smith, 1956; Jackson, 1956) e desde de então é percebido como um campo rico e próspero para investigações. Interesses de pesquisadores de diversas áreas com objetivos diferenciados compõem e têm proporcionado uma extensa literatura.

Segundo Pinedo (2008) o sequenciamento é um processo de tomada de decisão que é usado em uma base regular de indústrias de manufatura e indústrias de serviços, sendo importante também em configurações de transporte e de distribuição. O problema básico é trabalhar com a alocação de recursos limitados a atividades ao longo do tempo visando otimizar um ou vários objetivos.

Os recursos em uma organização podem assumir formas diferenciadas tais como máquinas em um centro de trabalho, pistas dos aeroportos, equipes em um canteiro de obras, unidades de processamento em um ambiente de computação, etc. As atividades podem ser as encomendas a distribuir para os clientes, operações em um processo de produção, decolagens e aterrissagens de aviões num aeroporto, execuções de programas de computador, entre outros. Os objetivos a serem otimizados também variam sendo os mais comuns os baseados nos tempos de conclusão e nas datas de entregas.

Os problemas de sequenciamento pertencem a uma classe especial de problemas de otimização combinatória. Sua abordagem para análise, segundo Blazewicz (2001), pode seguir linhas semelhantes, levando em consideração suas peculiaridades e a análise de sua complexidade.

Segundo Allahverdi et al. (2008) a maior parte dos artigos envolvendo sequenciamento negligênciam o tempo de preparação ou o emprega integrado em parte do tempo de processamento. Ele afirma que embora isso simplifique a análise e/ou reflita certas aplicações, afeta de forma negativa a qualidade da solução em muitos problemas de sequenciamento que requerem um tratamento explícito do tempo de preparação.

Allahverdi (2015) afirma que em pesquisas da literatura sobre problemas de sequenciamento mais de 90% da literatura sobre problemas de sequenciamento ignora tempos de preparação/custos.

Os problemas que tratam tempo de preparação de forma separada iniciaram em 1960 e desde então tem chamado atenção de pesquisadores nas mais diversas configurações, (Allahverdi et al., 1999, 2008; Allahverdi, 2015). Porém, quando se trata de máquinas paralelas, a maioria empenha-se em estudos de máquinas idênticas. Os trabalhos que abordam máquinas paralelas não relacionadas optam por otimizar o tempo máximo de conclusão ou o soma ponderada dos atrasos. São poucos os estudos que consideram a otimização da soma ponderada dos tempos de conclusão.

Este trabalho, estuda o problema na configuração de máquinas paralelas não relacionadas com tempos de preparação dependentes da sequência e da máquina, onde a função objetiva é minimizar a soma ponderada dos tempos de conclusão. Pretende-se transcrever o problema nas formulações matemáticas existentes na literatura, melhorando-as quando possível. Uma nova formulação será criada e solucionada com um algoritmo variante do método de decomposiçao de Benders.

1.1 Motivações

O problema de sequenciamento é motivado por questões que surgem no planejamento de produção, para equilibrar processos de computação, telecomunicações e, em geral em todas as situações em que recursos escassos devem ser alocados para atividades ao longo do tempo.

Trabalhar com máquinas em paralelo é uma configuração importante tanto do ponto de vista teórico quanto prático. Do ponto de vista teórico é uma generalização do caso de única máquina, é um caso especial do *flexible flow shop* e é NP-difícil. Do ponto de vista prático, é relevante pela ocorrência de recursos em paralelo serem comuns no mundo real. Além disso, as técnicas para máquinas em paralelo são frequentemente utilizadas em procedimentos de decomposição para sistemas multi-estágio.

Uma grande motivação para o problema abordado neste trabalho é o elevado número de aplicações industriais e de serviços no uso de tempos de preparação/custos (Kopanos et al., 2009) associado com poucas pesquisas envolvendo tempo de preparação dependentes da sequência num ambiente com máquinas paralelas não relacionadas, além da escassez de estudos com metódos exatos para sua resolução.

1.2 O problema de sequenciamento

Problema de sequenciamento são caracterizados por n_j tarefas e n_m máquinas onde os subcristos j e m se referem a tarefa a máquina, respectivamente.

Um problema de sequencimento de máquinas paralelas significa alocar tarefas a máquinas de forma a concluir o seu processamento respeitando condições impostas.

Há duas restrições gerais da teoria clássica de sequenciamento, sendo elas: cada tarefa deve ser processada por, no máximo, uma máquina de cada vez e cada máquina é capaz de processar, no máximo, uma tarefa de cada vez.

O problema de sequenciamento será descrito de acordo com a notação de Graham et al. (1979) que é dada pelo tripla $\alpha|\beta|\gamma$, onde α descreve a máquina envolvida, β fornece detalhes das características do processo e das restrições podendo conter de nenhuma a várias entradas e γ descreve o objetivo a ser minimizado.

1.2.1 Caracterização das máquinas

O problema poderá envolver uma única ou várias máquinas. As várias máquinas podem ser *paralelas* quando todas executam a mesma função ou *dedicadas* caso executem determinadas funções (Brucker, 2004; Blazewicz, 2001). Em máquinas paralelas há três tipos que são distinguindas de acordo com sua velocidade. Idênticas *(P_m)* se todas as máquinas tem velocidades iguais de processamento de tarefa; Uniformes *(Q_m)* se as máquinas difere na suas velocidade mas a velocidade de cada máquina é constante e não depende da tarefa; as máquinas são Não idênticas ou Não relacionadas *(R_m)* se a velocidade das máquinas depende das tarefas executadas. Em caso de *dedicadas* há três modelos de conjuntos de processamento de tarefas o *flow shop, open shop* e *job shop*.

1.2.2 Caracterização das tarefas

Uma tarefa consiste em uma série de operações elementares a serem realizada numa máquina. Assume que tarefas formam n subconjunto, cada subconjunto é chamado trabalho. Para o caso de sequenciamento de máquinas paralelas, os termos tarefas e trabalho não se diferem uma vez que cada trabalho realiza uma única operação elementar numa máquina. Para cada tarefa j tem-se os seguintes dados.

- *Tempo de processamento* (p_j^m) - Tempo de processo da tarefa j na máquina m. Tem-se $p_j^m = p_j$ quando o tempo não depende da máquina m ou o trabalho j só deve ser processado em uma única máquina.

- *Data de lançamento* (r_j) - Instante de tempo que a tarefa j fica disponível para ser processada. Se as datas de lançamento são as mesmas para todas as tarefas j então assume-se que $r_j = 0$ para todo j.

- *Data de entrega* (d_j) - Instante de tempo específico que a tarefa j deve ser concluída.

- *Data limite de entrega* ($\tilde{d}_j$) - Instante de tempo rígido que a tarefa j deve se concluída.

- *Peso ou prioridade* (w_j) - Peso ou prioridade associado a uma tarefa j.

- *Tempo de preparação* (s_{jk}) - Tempo de preparação dependentes da tarefas - s_{jk} representa a sequência dependente do tempo de preparação que é decorrido entre o processo do trabalho j e k; s_{0k} é o tempo de preparação para a tarefa k se ela é a primeira da sequência e s_{j0} se for a última (ambos valem zero). Se o tempo de preparação entre a tarefa j e k depende da máquina m então tem-se s^m_{jk}.

É comum o uso de valores inteiros para os parâmetros, porém isso não é restrito, uma vez que é permitido valores racionais arbitrários. Em relação às tarefas pode-se ter ainda outras informação como categorias de trabalhos, processamento em lotes, restrições de precedência, preempção entre outros.

Um sequenciamento é preemptivo quando a tarefa pode ser interrompida a qualquer momento e reiniciada mais tarde, não necessáriamente na mesma máquina. Se não é permitido a interrupção tem-se sequenciamento não preemptivo.

Podem ser definidas relações de dependência entre as tarefas, por exemplo, a relação $t_i < t_j$ onde uma tarefa t_j só pode começar depois que t_i terminar. Se esta relação for definida entre as tarefas elas são ditas dependentes caso contrário são independentes. Essas dependências são normalmente definidas em grafos onde os nós correspondem as tarefas e os arcos a restrição com prioridades.

1.2.3 Critérios de otimalidade

Os problemas de sequenciamento objetivam encontrar sequências viáveis que minimizem uma função custo tal que as seguintes condições sejam satisfeitas: cada tarefa deve ser processada por, no máximo, uma máquina de cada vez e cada máquina é capaz de processar, no máximo, uma tarefa de cada vez; uma determinada tarefa t_j é processada no intervalo de tempo $[r_j, \infty]$; todas as tarefas devem ser concluídas; se as tarefas t_i e t_j estão relacionadas de forma que $t_i < t_j$, o processo de t_j só começa depois que

t_i for concluído; em caso de sequenciamento não preemptivo as tarefas não são interrompidas, caso contrário as tarefas podem ser interrompidas; se houver restrições de recursos, elas devem ser satisfeitas. Sendo assim, tem-se para cada tarefa j:

- Tempo de conclusão, C_j;
- Desvio em relação à data de entrega, $L_j = C_j - d_j$;
- Atraso, $T_j = max(0, C_j - d_j)$;
- Adiantamento, $E_j = max(0, d_j - C_j)$;
- Tarefa em atraso, $U_j = 1$ se $C_j > d_j$ e $U_j = 0$ caso contrário;
- Tempo de permanência no sistema, $F_j = C_j - r_j$;
- Desvio absoluto, D_j;
- Desvio quadrado, S_j.

Para avaliar sequenciamento pode-se usar vários critérios de desempenho sendo eles:

- Tempo máximo de conclusão, $C_{max} = max\{C_j\}$;
- Maior desvio à data de entrega, $L_{max} = max\{L_j\}$;
- Soma dos tempos de conclusão, $\sum C_j$;
- Soma ponderada dos tempos de conclusão, $\sum w_j C_j$;
- Soma ponderada dos tempos de fluxo, $\sum w_j F_j$
- Soma dos atrasos, $\sum T_j$;
- Soma ponderada dos atrasos, $\sum w_j T_j$;
- Número de tarefas em atraso, $\sum U_j$;

1.2.4 Complexidade

Uma das principais direções de pesquisa em sequenciamento em máquinas é determinar a complexidade do problema. Análises de complexidade para problemas de sequenciamento são apresentados em Brucker et al. (2002). Como pode ser visto a partir destes resultados, a maioria dos problemas de sequenciamento são classificados como NP-difíceis.

1.3 Estrutura da dissertação

Problemas envolvendo máquinas paralelas não relacionadas com tempos de preparação dependentes da sequência e da máquina com o objetivo de minimizar o tempo de conclusão ponderado são escassos na literatura (Seção 2.3) demonstrando uma necessidade de estudos voltados para esta área. Diante disso será abordado neste trabalho o problema mencionado de forma que será apresentado e analisado uma nova formulação focando na sua resolução por métodos exatos. Sendo assim o trabalho é organizado da forma seguinte.

No Capítulo 2 são apresentados a definição formal do problema e uma revisão da literatura sobre as principais formulações existentes e metódos utilizados para resolvê-lo, concentrando nos metódos exatos.

No Capítulo 3 é exibida a nova formulação, juntamento com a abordagem do algoritmo variante do método de decomposição de Benders e do algoritmo do *Logic Based* Bender.

Os testes computacionais realizados são descritos, bem como seus resultados são apresentados no Capítulo 4. Além disso são comparadas as 6 formulações com a formulação proposta e os dois algoritmos utilizados na sua resolução.

Por último, no Capítulo 5 são feitas as considerações finais.

Capítulo 2

Exame da literatura

"O que dá o verdadeiro sentido ao encontro é a busca, e é preciso andar muito para se alcançar o que está perto"

José Saramago

Com o objetivo de examinar a literatura associada ao problema, este capítulo apresenta inicialmente a definição formal do problema abordado, na Seção 2.1. Em seguida, na Seção 2.2 são apresentadas as principais formulações matemáticas existentes na literatura e por fim, na Seção 2.3 os métodos de resoluções heurísticos e exatos são brevemente discutidos.

2.1 O problema

O assunto abordado pode ser definido como o problema de sequenciamento de máquinas paralelas não relacionadas com tempos de preparação dependentes da sequência e da máquina (PSMPRTS), em que pretende-se alocar e sequenciar n_j tarefas em n_m máquinas com o objetivo de minimizar a soma ponderada dos tempo de conclusões das tarefas. Representado segundo a notação de Graham et al. (1979) por $R_m|s_{ij}^m|\sum w_j C_j$.

Cada tarefa deve ser processada por exatamente uma máquina tendo um tempo de processamento e uma prioridade associada. O processamento das tarefas nas máquinas implica em um tempo de preparação dependente da sequência a processar. Todos os parâmetros são determinísticos e inteiros não negativos.

A notação usada descreve os conjuntos e parâmetros utilizadas nas formulações matemáticas que se seguem. As váriáveis e demais parâmetros que não são comuns a todos serão definidas conforme se fizer necessário.

Conjuntos

Sejam n_j e n_m o número de tarefas e máquinas respectivamente, tem-se:

$N = \{1, ..., n_j\}$ - Conjunto com n_j tarefas

$M = \{1, ..., n_m\}$ - Conjunto com n_m máquinas

$A = \{(i, j) \ : \ i \in N_0 \text{ e } j \in N, i \neq j\}$

Parâmetros

w_j - Prioridade ou peso da tarefa j.

p_j^m - Tempo de processamento da tarefa j na máquina m.

s_{ij}^m - Tempo de preparação dependente da sequência e da máquina onde processa a tarefa j depois de ter processado a i.

$\mathcal{M}$ - Um número suficientemente grande.

O problema PSMPRTS é classificado NP-difícil pois o problema de máquinas paralelas idênticas, caso particular de máquinas paralelas não relacionadas, $P||\sum w_j c_j$, é NP-difícil. Verificado em Lenstra et al. (1977), Weng et al. (2001) e Pinedo (2008).

Considere o Exemplo 1 cuja solução ótima é 1324. Sua solução é representada no gráfico de Gantt, Figura 2.1, onde J_i representa uma tarefa. O Exemplo é representado em figuras para a maioria das formulações encontradas na litertatura, Seção 2.2.

Exemplo 1. *Minimize* $\sum_{j=1}^{6} w_j C_j$ *considerando a instância de* $n_m = 2$*;* $n_j = 6$*; os tempos de processamento* $p_1^1 = 1$, $p_2^1 = 87$, $p_3^1 = 28$, $p_4^1 = 32$, $p_5^1 = 38$, $p_6^1 = 9$, $p_1^2 = 4$, $p_2^2 = 21$, $p_3^2 = 68$, $p_4^2 = 17$ $p_5^2 = 43$ *e* $p_6^2 = 48$*; pesos* $w_1 = 3$, $w_2 = 6$,$w_3 = 5$,$w_4 = 8$, $w_5 = 9$, $w_6 = 2$ *e os tempos de preparação*

s_{ij}^1

	1	2	3	4	5	6
1	0	1	8	1	3	9
2	4	0	7	3	7	8
3	7	3	0	2	3	2
4	3	8	3	0	5	2
5	8	3	7	9	0	3
6	8	8	1	2	2	0

s_{ij}^2

	1	2	3	4	5	6
1	0	5	1	6	1	7
2	6	0	7	7	6	2
3	7	6	0	9	6	9
4	3	7	3	0	1	7
5	5	8	5	6	0	9
6	7	4	1	7	9	0

2.2 Formulações matemáticas

As formulações encontradas na literatura para lidar com o problema de sequêciamento variam, principalmente, com base nas váriaveis de decisão sobre os quais elas dependem. As variáveis de decisão usadas neste documento são: variavéis tempo de conclusão

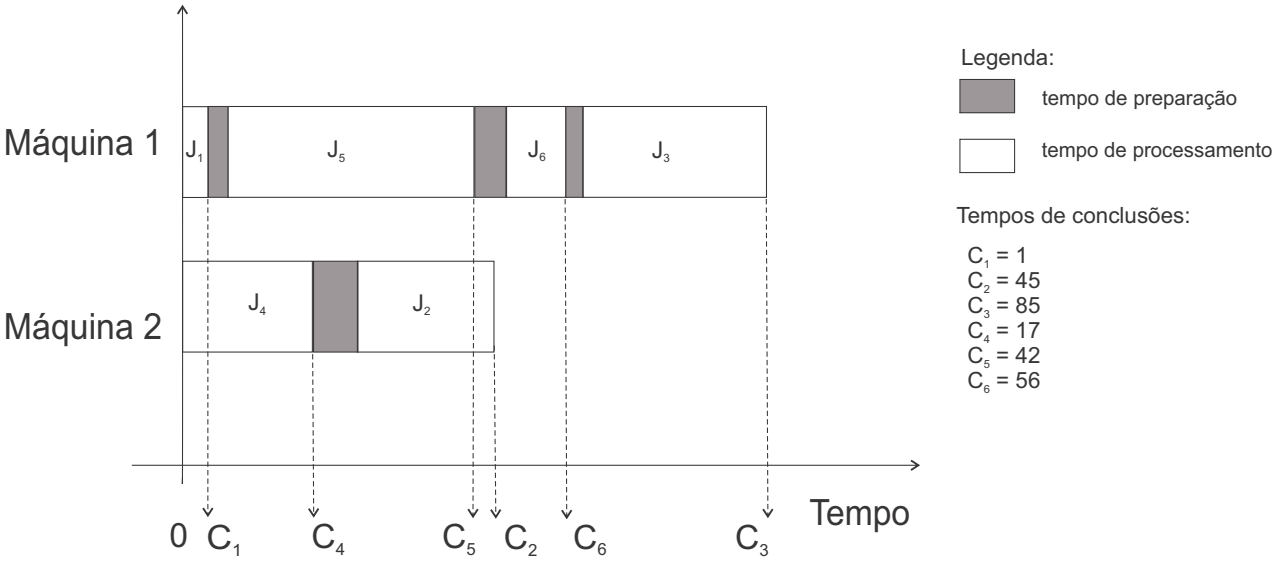

Figura 2.1: Solução do Exemplo 1 representado no gráfico de Gantt

(Manne, 1960), variavéis de ordenção linear (Potts, 1980), variáveis de atribuição e datas de posicionamento (Wagner, 1959), variáveis de rede (Blazewicz et al., 1991), variavéis indexada no tempo (Sousa and Wolsey, 1992) e variáveis arco tempo indexada (Pessoa et al., 2010).

Queyranne and Schulz (1994) e Schulz et al. (1996) exibe uma análise da abordagem poliédrica para problemas de sequêciamento de máquinas sem interrupção. É feito uma revisão das formulções baseadas em variáveis indexada no tempo, atribuição e data de posicionamento, variáveis de ordenação linear e variáveis de rede. Mostra as relações entre elas, entre os modelos e enfatiza o papel importante da estrutura poliédrica em muitas formulações. Li and Yang (2009) coleta e classifica modelos e suas relaxações, além de pesquisar heurística e técnicas de otimização.

Em Unlu and Mason (2010) é definido e comparado resultados computacionais para quatro formulações diferentes de programação inteira mista (PIM) para o problema de sequenciamento de máquina paralelas idênticas sem interrupção e sem considerar os tempos de preparação.

No estudo sobre formulações é relevante mencionar ainda Dyer and Wolsey (1990), Blazewicz et al. (1991), Allahverdi et al. (2008) e Pinedo (2008).

2.2.1 Formulação variável tempo de conclusão

Variáveis de tempo de conclusão muitas vezes referida como *variáveis de data natural* (Queyranne and Schulz, 1994), foram introduzida por Manne (1960) em um problema de sequenciamento de *job shop*. Essas variáveis foram usada por Balas (1985) em sua conhecida formulação disjuntiva de *job shop*.

Nesta formulação a sequência é determinada exclusivamente pelos tempos de conclusão das tarefas correspondentes.

Variáveis:

x_j^m $\begin{cases} 1, & \text{se a tarefa } j \text{ é processada na máquina } m, \\ 0, & \text{caso contrário.} \end{cases}$

y_{ij}^m $\begin{cases} 1, & \text{se as tarefas } i \text{ e } j \text{ são processadas na mesma máquina } m, \text{ onde } j \text{ procede } i, \\ 0, & \text{caso contrário.} \end{cases}$

C_j Tempo de conclusão da tarefa j.

$$min \sum_{j \in N} w_j C_j$$

$s.\ a:$

$$\sum_{m \in M} x_j^m = 1 \qquad \forall\ j \in N \tag{2.1}$$

$$y_{ij}^m + y_{ji}^m \geq x_j^m + x_i^m - 1 \qquad \forall\ m \in M;\ (i,j) \in A \tag{2.2}$$

$$C_j \geq C_i + \sum_{m \in M} (s_{ij}^m + p_j^m) y_{i,j}^m - (1 - \sum_{m \in M} y_{ij}^m)\mathcal{M} \qquad \forall\ (i,j) \in A \tag{2.3}$$

$$y_{0j}^m = x_j^m \qquad \forall\ m \in M;\ j \in N \tag{2.4}$$

$$C_j \geq 0 \qquad \forall\ m \in M;\ j \in N \cup \{0\} \tag{2.5}$$

$$x_j^m \in \{0,1\} \qquad \forall\ m \in M;\ j \in N \cup \{0\} \tag{2.6}$$

$$y_{ij}^m \in \{0,1\} \qquad \forall\ m \in M; \quad (i,j) \in A \cup \{(i,0) : i \in N\} \tag{2.7}$$

O conjunto de restrições (2.1) garante que toda tarefa j é alocada para alguma máquina m. Em (2.2) define a precedência entre as tarefas i e j. O cálculo do tempo de conclusão é fornecido pela restrição (2.3). Em (2.4) assegura que se alguma tarefa j é alocada para alguma máquina m então a disjunção y_{0j}^m está ativa. Finalizando em (2.5) assegura a não negatividade de C_j, já (2.6) e (2.7) é estabelecido a integralidade do domínio.

2.2.2 Formulação variável de ordenação linear

As variáveis de ordenação linear foram introduzidas por Potts (1980) e são referidas também como variáveis de sequenciamento por Pinedo (2008). Elas foram utilizadas

por Dyer and Wolsey (1990) em combinação com variáveis de tempo inicial, nos estudos do problema de uma única máquina, com datas de lançamento. Essas variavés foram estudadas também por Blazewicz et al. (1991), Nemhauser and Savelsbergh (1992) e Chudak and Hochbaum (1999).

Esta formulação se baseia no uso de três variavéis binárias. Uma que define se uma tarefa é processada numa determinada máquina, outra que determina relações de precedência entre todas as tarefas e por fim a terceira váriavel vincula tarefas em alguma máquina. Possibilitando assim definir em que máquina cada tarefa será processada e a ordem de processamento para cada máquina.

Variáveis:

x_j^m $\begin{cases} 1, \text{ se a tarefa } j \text{ é sequenciada na máquina } m, \\ 0, \text{ caso contrário.} \end{cases}$

y_{ij} $\begin{cases} 1, \text{ se a tarefa } i \text{ precede a tarefa } j, \\ 0, \text{ caso contrário.} \end{cases}$

λ_{ij} $\begin{cases} 1, \text{ se a tarefa } i \text{ e } j \text{ não estão na mesma máquina,} \\ 0, \text{ caso contrário.} \end{cases}$

C_j^m Tempo de conclusão da tarefa j na máquina m.

$$min \sum_{m \in M} \sum_{j \in N} w_j C_j^m$$

$s.\ a:$

$$\sum_{m \in M} x_j^m = 1 \qquad \forall\ j \in N \tag{2.8}$$

$$y_{ij} + y_{ji} + \lambda_{ij} = 1 \qquad \forall\ (i,j) \in A,\ i \neq 0 \tag{2.9}$$

$$y_{ij} + y_{jk} + y_{ki} \leq 2 \qquad \forall\ i,\ j,\ k \in N,\ i \neq j \neq k \tag{2.10}$$

$$x_i^m + x_j^m + \lambda_{ij} \leq 2 \qquad \forall\ m \in M; (i,j) \in A,\ i \neq 0 \tag{2.11}$$

$$C_j^m \geq p_j^m x_j^m \qquad \forall\ m \in M;\ j \in N \tag{2.12}$$

$$C_j^m \geq C_i^m - (1 - y_{ij})\mathcal{M} + (p_j^m + s_{ij}^m)(y_{ij} + x_i^m + x_j^m - 2) \qquad \forall\ m \in M;\ (i,j) \in A,\ i \neq 0 \tag{2.13}$$

$$C_j^m \geq 0 \qquad \forall\ m \in M;\ j \in N \tag{2.14}$$

$$x_j^m \in \{0,1\} \qquad \forall\ m \in M;\ j \in N \tag{2.15}$$

$$y_{ij},\ \lambda_{ij} \in \{0,1\} \qquad \forall\ (i,j) \in A,\ i \neq 0 \tag{2.16}$$

O conjunto de restrições (2.8) assegura que toda tarefa j é posicionada em alguma máquina, (2.9) garante a precedência entre as tarefas i e j, desde que as duas estejam na mesma máquina. O conjunto (2.10) representa as restrições de transitividades atestando uma ordem linear entre três tarefas. Em (2.11) calcula-se corretamente a variável λ_{ij}. As restrições (2.12) e (2.13) fornecem o cálculo do tempo de conclusão das tarefas. Finalizando tem-se a não negatividade de C_j^m em (2.14) e a integralidade das variáveis certificadas pelas restrições (2.15) e (2.16).

2.2.3 Formulação variável de atribuição e datas de posicionamento

Inicialmente Wagner (1959) apresentou um modelo de programação inteira para um problema *flow shop* de 3 máquinas com o objetivo de encontrar o instante de conclusão máximo. Lasserre and Queyranne (1992) apresentou o problema de sequenciamento para uma única máquina que considerava o problema como um sistema a ser controlado em instantes de tempos discretos usando combinações de controles discretos e contínuos.

Esta formulação especifica qual a próxima tarefa que será agendada e em que momento ela irá começar a ser processada. A formulação assume que uma máquina tem no máximo l posições para processar as tarefas, ou seja, pode receber no máximo l tarefas. Com o posicionamento das tarefas em cada máquina é possível definir a sequência,ver Figura 2.2.

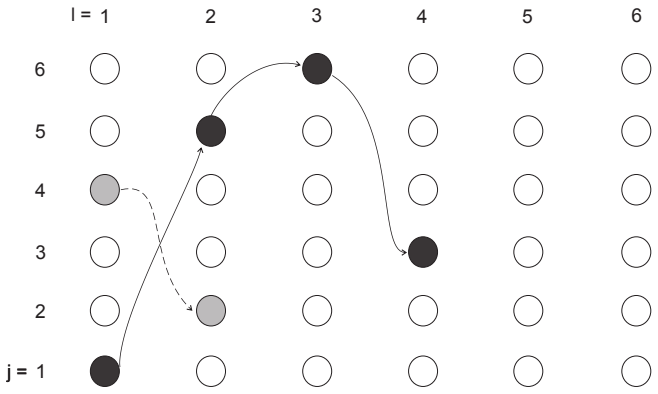

Figura 2.2: Solução do Exemplo 1 na formulação Atribuição e Datas de Posicionamento

A váriavel x_j^{ml} se refere à tarefa que é executada na posição l. Na definição do modelo é necessário fixar o número de posições que uma máquina tem disponíveis para processar. Seja $l \in P = \{1, ..., n_j\}$ onde n_j é o número de tarefas.

Variáveis:

x_j^{ml} $\begin{cases} 1, & \text{se a tarefa } j \text{ é atribuída a posição } l \text{ na máquina } m, \\ 0, & \text{caso contrário.} \end{cases}$

γ_m^l Denota o tempo de conclusão de uma determinada tarefa que esteja na máquina m e na posição l.

y_{ij}^{ml} Define que na máquina m as tarefas i e j são atribuídas às posições $l-1$ e l, respectivamente. Então, o tempo de preparação entre as tarefas i e j é usado entre essas posições.

C_j^m Tempo de conclusão da tarefa j na máquina m.

$$min \sum_{m\in M}\sum_{j\in N} w_j C_j^m$$

$s.\ a:$

$$\sum_{m\in M}\sum_{l\in P} x_j^{ml} = 1 \qquad \forall\ j \in N \tag{2.17}$$

$$\sum_{j\in N} x_j^{ml} \leq 1 \qquad \forall\ \ m \in M;\ l \in P \tag{2.18}$$

$$x_j^{ml} \leq \sum_{i\in N} x_i^{m(l-1)} \qquad \forall\ m \in M;\ j \in N;\ l \in P,\ l \neq 1 \tag{2.19}$$

$$y_{ij}^{m(l-1)} \geq x_i^{m(l-1)} + x_j^{ml} - 1 \qquad \forall\ m \in M,\ i,\ j \in N,\ i \neq j;\ l \in P,\ l \neq 1 \tag{2.20}$$

$$\gamma_m^l \geq \sum_{j\in N} p_j^m x_j^{ml} \qquad \forall\ m \in M;\ l \in P \tag{2.21}$$

$$\gamma_m^l \geq \gamma_m^{(l-1)} + \sum_{i\in N}\sum_{\substack{j\in N:\\ j\neq i}} y_{ij}^{m(l-1)} s_{ij}^m + \sum_{j\in N} p_j^m x_j^{ml} \qquad \forall\ m \in M;\ l \in P,\ l \neq 1 \tag{2.22}$$

$$C_j^m \geq \gamma_m^l - (1 - x_j^{ml})\mathcal{M} \qquad \forall\ m \in M;\ j \in N;\ l \in P \tag{2.23}$$

$$C_j^m \geq 0 \qquad \forall\ m \in M;\ j \in N; \tag{2.24}$$

$$\gamma_m^l \geq 0 \qquad \forall\ m \in M;\ l \in P \tag{2.25}$$

$$x_j^{ml} \in \{0,1\} \qquad \forall\ m \in M;\ j\ \ \in N;\ l \in P \tag{2.26}$$

$$y_{ij}^{ml} \in \{0,1\} \qquad \forall\ m \in M;\ i,\ j \in N;\ l \in P \tag{2.27}$$

O conjunto de restrições (2.17) garante que toda tarefa é atribuída a uma máquina e a uma posição. Por sua vez o conjunto (2.18) assegura que cada posição em cada máquina possui no máximo uma tarefa. Em (2.19) afirma que se uma posição l é ocupada então a posição anterior $l-1$ também será. O conjunto de restrições (2.20) atribui valor para a variável $y_{ij}^{m(l-1)}$. As restrições (2.21) e (2.22) garantem que a

variável γ_m^l recebe o valor do tempo de conclusão da tarefa j na posição l. A partir desse valor a restrição (2.23) calcula o tempo de conclusão para cada tarefa em relação a máquina que ela se encontra. A não negatividade de C_j^m e γ_m^l são asseguradas pelas restrições (2.24) e (2.25) respectivamente a integralidade de x_j^{ml} e y_{ij}^{ml} são asseguradas por (2.26) e (2.27).

2.2.4 Formulação variável em rede

A formulação variáveis em rede ou variável do caixeiro viajante (Queyranne and Schulz, 1994), foi usada incialmente para modelar problemas de programação em uma única máquina com sequência dependente de tempo de processamento por parecer com o problema do caixeiro viajante dependente do tempo. Blazewicz et al. (1991) expôs uma formulação baseado em Miller et al. (1960) para o problema de programação de uma única máquina que foi estendida considerando instante de lançamento.

Nesta formulação as tarefas são percebidas como os nós da rede e um caminho nela representa o planejamento para uma máquina. Os nós visitados nesse caminho traduzem a ordem pela quais as tarefas são processadas, ou seja, a sequência para uma máquina. O objetivo neste modelo é descobrir os melhores caminhos para o conjunto das máquinas, Figura 2.3.

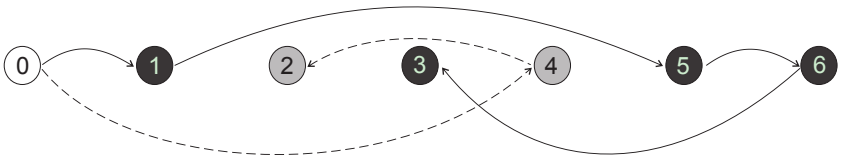

Figura 2.3: Solução do Exemplo 1 na formulação em Rede

Variáveis:

$$x_{ij}^m \begin{cases} 1, & \text{se na máquina } m \text{ a tarefa } i \text{ precede a tarefa } j, \\ 0, & \text{caso contrário.} \end{cases}$$

C_j Tempo de conclusão da tarefa j.

Observe que x_{0j}^m é usado para indicar que em uma máquina m a tarefa j procede a tarefa fictícia 0.

$$min \sum_{j \in N} w_j C_j$$

$$s.\ a:$$

$$\sum_{m \in M} \sum_{\substack{i \in N \cup \{0\}: \\ i \neq j}} x_{ij}^m = 1 \qquad \forall\ j \in N \tag{2.28}$$

$$\sum_{j \in N} x_{0j}^m \leq 1 \qquad \forall\ m \in M \tag{2.29}$$

$$\sum_{j \in N: i \neq j} x_{ij}^m \leq 1 \qquad \forall\ m \in M;\ i \in N \tag{2.30}$$

$$\sum_{i \in N \cup \{0\}: i \neq j} x_{ij}^m = \sum_{i \in N \cup \{n_j+1\}: j \neq i} x_{ji}^m \qquad \forall\ m \in M;\ j \in N \tag{2.31}$$

$$C_j \geq C_i - (1 - \sum_{m \in M} x_{ij}^m)\mathcal{M} + \sum_{m \in M} (s_{ij}^m + p_j^m) x_{ij}^m \qquad \forall\ (i,j) \in A \tag{2.32}$$

$$C_j \geq 0 \qquad \forall\ j \in N \tag{2.33}$$

$$x_{ij}^m \in \{0,1\} \qquad \forall\ m \in M;\ (i,j) \in A \cup \{n_j + 1\} \tag{2.34}$$

O conjunto de restrições (2.28) garante que toda tarefa é alocada a uma máquina possuindo somente um antecessor. A variável $x_{0j}^m = 1$ indica que a tarefa j é a primeira a ser executada na máquina m daí a restrição (2.29) limita o número máximo de tarefas imediatamente sucessora à tarefa fictícia zero em cada máquina. Em (2.30) garante que em cada máquina uma determinada tarefa i poderá ter no máximo um sucessor. A sequência das tarefas em cada máquina é assegurada por (2.31). O conjunto (2.32) calcula o tempo de conclusão de cada tarefa j. Finalizando (2.33) e (2.34) asseguram respectivamente a não negatividade de C_j e a integralidade das variáveis binárias.

2.2.5 Formulação variável indexada no tempo

A formulação clássica variável indexada no tempo para sequenciamento foi inicialmente apresentada por Sousa and Wolsey (1992) no estudo do caso de uma única máquina. Sua criação foi motivada pelo fato das outras formulações existentes naquele momento serem comparativamente fracas (Dyer and Wolsey, 1990).

A formulação considera que o tempo é dividido em períodos $1, ..., n_t$ onde n_t é o limite superior dos tempo de conclusão máximo das tarefas. De uma forma geral, o que se pretende é decidir qual máquina e qual período de tempo que cada tarefa começará a ser processada, Figura 2.5.

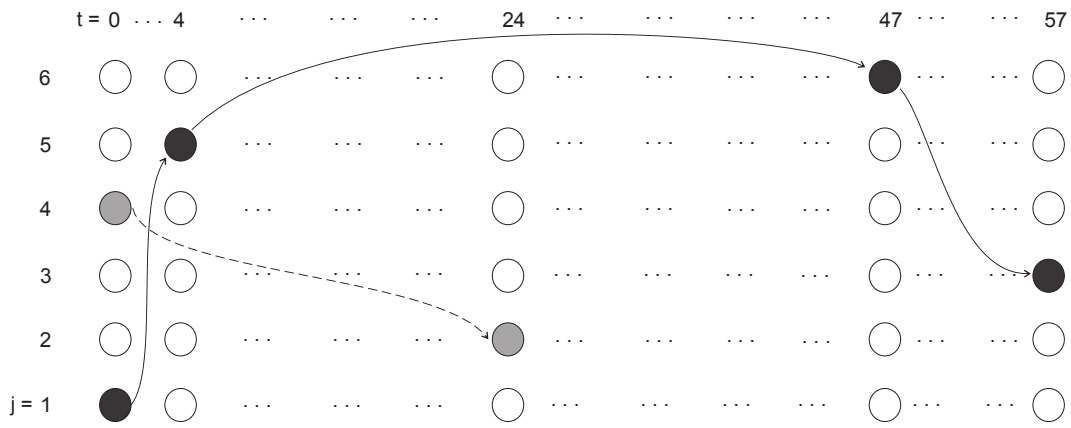

Figura 2.4: Solução do Exemplo 1 na formulação Tempo Indexada

Variáveis:

x_j^{mt} $\begin{cases} 1, & \text{se na máquina } m \text{ a tarefa } j \text{ começa a ser executada em um tempo } t, \\ 0, & \text{caso contrário.} \end{cases}$

C_j Tempo de conclusão da tarefa j.

$$\sum_{j \in N} w_j C_j$$

$s.\ a:$

$$\sum_{m \in M} \sum_{t=0}^{n_t - p_j^m + 1} x_j^{mt} = 1 \qquad \forall\ j \in N \tag{2.35}$$

$$x_j^{mt} + \sum_{r=max(0, t-p_i^m-s_{ij}^m+1)}^{t} x_i^{mr} \leq 1 \qquad \forall\ m \in M;\ i,\ j \in N,\ j \neq i;$$

$$t \in \{0, ..., n_t - p_j^m + 1\} \tag{2.36}$$

$$C_j \geq \sum_{m \in M} \sum_{t=0}^{n_t - p_j^m + 1} (t + p_j^m) x_j^{mt} \qquad \forall\ j \in N \tag{2.37}$$

$$C_j \geq 0 \qquad \forall\ j \in N \tag{2.38}$$

$$x_j^{mt} \in \{0, 1\} \qquad \forall\ m \in M;\ j \in N;\ t \in \{0, ..., n_t - p_j^m\} \tag{2.39}$$

O conjunto de restrições (2.35) assegura que toda tarefa estará em uma máquina e começará o processo em um período t. Em (2.36) garante que se a tarefa j é processada no período de tempo t, nenhuma outra tarefa poderá ser processada entre o período $t - p_i^m - s_{ij}^m + 1$ e t. A restrição (2.37) calcula corretamente o tempo de conclusão das

tarefas. Finalizando tem-se assegurada a não negatividade da variável C_j por (2.38) e a integralidade da variável x_j^{mt} por (2.39).

2.2.6 Formulação variável arco indexado no tempo

Esta formulação foi apresentada por Pessoa et al. (2010) onde exibe um algoritmo exato para o problema de sequenciamento de máquinas paralelas idênticas tendo em vista minimizar a soma ponderada dos atrasos.

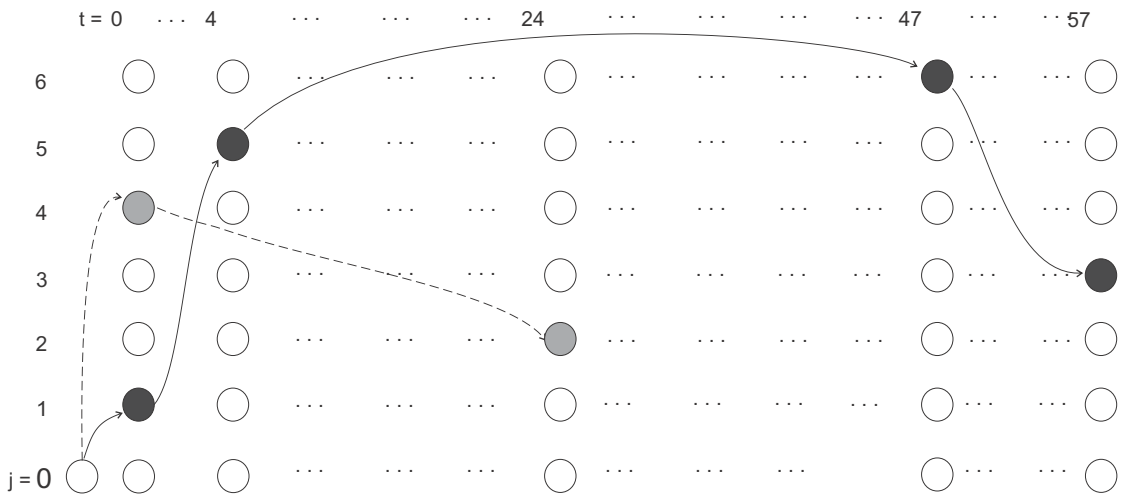

Figura 2.5: Solução do Exemplo 1 na formulação Arco Indexado no Tempo

Considere os conjuntos:
$A' = \{(i,j) : i \in N \cup \{0\} \text{ e } j \in N \cup \{n_j + 1\}, i \neq j\}$.
$T = \{0, ..., n_t\}$, um plano horizonte discreto para o período e n_t assume o mesmo valor que em 2.2.5.

Variáveis:

x_{ij}^{mt} $\begin{cases} 1, & \text{se a tarefa } j \text{ inicia em } t \text{ numa máquina } m \text{ após } i \text{ ter sido terminada,} \\ 0, & \text{caso contrário.} \end{cases}$

C_j Tempo de conclusão da tarefa j.

Usa-se a tarefa fictícia 0 para indicar o começo e $n_j + 1$ o fim da sequência na formulação. Todos os valores de parâmetros em relação a essas tarefas são nulos e a máquina estará inativa em $t = 0$ e $t = n_t$.

$$min \sum_{j \in N} w_j C_j$$

$s.\ a:$

$$\sum_{m \in M} \sum_{(i,j) \in A'} \sum_{t\ =\ p_i^m + s_{ij}^m}^{n_t - p_j^m} x_{ij}^{mt} = 1 \qquad \forall\ j \in N \tag{2.40}$$

$$\sum_{(0,j) \in A'} \sum_{t\ =\ p_0^m + s_{0j}^m}^{n_t - p_j^m} x_{0j}^{mt} \leq 1 \qquad \forall\ m \in M \tag{2.41}$$

$$\sum_{\substack{(i,j) \in A': \\ t \geq p_i^m + s_{ij}^m}} x_{ij}^{mt} - \sum_{\substack{(j,i) \in A': \\ t + p_j^m + s_{ji}^m \leq \\ n_t - p_i^m}} x_{ji}^{m(t + p_j^m + s_{ji}^m)} = 0 \qquad \forall\ m \in M;\ j \in N;$$

$$t \in T,\ t \leq n_t - p_j^m \tag{2.42}$$

$$C_j \geq \sum_{m \in M} \sum_{(i,j) \in A'} \sum_{t\ =\ p_i^m + s_{ij}^m}^{n_t - p_j^m} (t + p_j^m) x_{ij}^{mt} \qquad \forall\ j \in N \tag{2.43}$$

$$C_j \geq 0 \qquad \forall\ j \in N \tag{2.44}$$

$$\forall\ m \in M;\ (i,j) \in A';$$

$$x_{ij}^{mt} \in \{0,1\} \qquad t \in \{(p_i^m + s_{ij}^m), ..., (n_t - p_j^m)\} \tag{2.45}$$

O conjunto de restrições (2.40) impõe que toda tarefa deverá ser visitada por exatamente um caminho. Já (2.41) assegura que toda máquina terá no máximo uma tarefa iniciando. Em (2.42) assegura que se uma tarefa j é processado no período t a próxima tarefa na sequência i, $(j \neq i)$ deverá ser processado no período $t + p_j^m + s_{ji}^m$. O cálculo do tempo de conclusão de cada tarefa é realizado pela restrições (2.43). Em (2.44) e (2.45) é garantido a não negatividade de C_j a integralidade de x_{ij}^{mt} respectivamente.

2.3 Métodos de resolução

Nesta Seção será apresentado uma revisão da literatura sobre o problema de máquinas paralelas direcionada aos métodos de solução exata.

Como mencionado anteriormente os problemas de sequenciamento são complexos e difíceis de resolver. Devido a este fato, muitos metódos heurísticos foram propostos para sua solução. Alguns procedimentos de solução exata foram também desenvolvidos mas se apresentaram incapazes de resolver problemas de grandes dimensões. No entanto, o método de enumeração implicíta, permitiu o desenvovilmento de algoritmos

de solução exata para alguns problemas.

Os métodos heurísticos não garantem a otimalidade da solução encontrada, no entanto, tais métodos tem sido bastante utilizados nos últimos anos, pois na prática eles têm sido capazes de encontrar boas soluções em um tempo viável.

Os métodos exatos por sua vez, garatem a otimalidade da solução sendo os métodos de enumeração os mais utilizados. Eles consistem em organizar uma busca no conjunto de solução viáveis do problema de maneira que este conjunto será todo analisado e a solução ótima encontrada.

Com o intuito de uma revisão bibliográfica mais voltada para os métodos exatos faz-se uma busca na literatura para casos envolvendo tanto uma única máquina quanto máquinas paralelas. Para o caso de uma única máquina procurou-se os artigos que abordaram tempos de preparação dependentes da sequência com o critério de minimizar a soma ponderada dos tempos de conclusão das tarefas, já para o caso de máquinas paralelas pesquisou-se trabalhos que diferenciam no tipo de máquina utilizada ou critério de otimalidade com a abordargem do tempo de preparação dependente da sequência.

Nessa direção trabalhos da literatura envolvendo uma única máquina com as característica mencionadas no paragráfo anterior foi estudado por Chou et al. (2009) e Nogueira et al. (2014). O primeiro apresentou dois métodos exatos, um modelo de *Constraint Programming* e um algoritmo *Branch and Bound* e duas heurísticas e o segundo propôs e analisou diferentes formulações de PIM.

Para estudos utilizandos metódos heurísticos, o problema de máquinas paralelas com tempo de preparação dependentes da sequência no ambiente de máquinas idênticas, $P||s_{ij}^m||\sum w_jC_j$ foi abordado por Cochran et al. (2003), apresenta um algoritmo genético híbrido, abrangendo outros objetivos tais como o atraso ponderado e o tempo máximo de conclusão. Pesquisas envolvendo máquinas paralelas não relacionadas, $R||s_{ij}^m||\sum w_jC_j$, tem-se o trabalho de Weng et al. (2001) que apresenta sete heurísticas para minimizar a soma ponderada dos tempos de conclusões das tarefas encontrando soluções para até 120 tarefas e 12 máquinas.

Na abordagem heurística envolvendo máquinas paralelas não relacionadas com tempo de preparação dependente da sequência e variando o critério de otimalidade tem-se, para o problema $R||s_{ij}^m||C_{max}$, os estudos de Vallada and Ruiz (2011), Arnaout et al. (2010), Niu et al. (2011), Lin and Hsieh (2014) e Avalos-Rosales et al. (2015); e para $R||s_{ij}^m||\sum w_jT_j$ os trabalhos de de Paula et al. (2010) e Lee et al. (2013).

Voltando os olhares para a abordagem exata de problemas parecidos, van den Akker et al. (1999) considera o problema de máquinas paralelas idênticas minimizando a soma ponderada dos tempo de conclusão. O problema foi formulado como um problema de cobertura conjunto. Os resultados computacionais mostram que o limite inferior é particularmente forte, e que o resultado do problema é frequentemente in-

teiro. Seu algoritmo resolve problemas com 100 tarefas e 10 máquinas dentro de um tempo computacional razoável.

Chen and Powell (1999) considera o problema de máquinas idênticas, uniforme ou não relacionadas com objetivos de minimizar a soma ponderada dos tempos de conclusão ou minimizar o atraso ponderado. Usa o metódo decomposição de Dantzig-Wolfe. Os resultados computacionais indicam que esta abordagem é promissora e capaz de resolver problemas com 100 tarefas e 20 máquinas para o primeiro objetivo, e 100 tarefas e 10 máquinas para o segundo objetivo.

Em máquinas paralelas não relacionadas Zhu and Heady (2000) desenvolveram uma formulação de programação inteira mista para o problema com tempo de preparação dependentes da sequência minimizando a soma de adiantamento/atraso o qual fornece uma solução ótima em tempo razoável para nove postos de trabalho e três máquinas.

Chen and Lee (2002) considera o problema de sequencimento de um conjunto independentes de trabalhos em máquinas paralelas idênticas com o objetivo de minimizar a avanço/atraso total dos trabalhos penalizados. Todos os trabalhos têm um determinada janela comum. Resultados computacional mostram que a abordagem resolvem em tempo computacional rasoáveis para 40 trabalhos e 6 máquinas.

Chen and Powell (2003) estudou o problema de sequenciamento com famílias de tarefas em máquinas paralelas idênticas, com sequência dependentes ou independentes de tempos de preparação onde minimiza a soma ponderada dos tempos de conclusão das tarefas. Desenvolve um algorítimo *branch-and-bound* (B&B) onde os resultados computacionais mostraram que ele resolve em um tempo computacional rasoável, problemas com 40 tarefas, 4 máquinas e 6 famílias.

Para o problema de máquinas não relacionadas Pereira Lopes and de Carvalho (2007) desenvolve um algoritmo *branch-and-price* para minimizar o atraso ponderado total em problemas de sequenciamento com máquinas paralelas não relacionadas com tempos de preparação dependentes da sequência, datas de disponibilidade para as máquinas e datas de vencimento para os postos de trabalho. Ele propõe uma técnica de aceleração *primal box* para o algorítmo de geração de coluna. A abordagem resolve problemas com 50 máquinas e 150 tarefas em um tempo computacional razoável. É relevante mencionar que a formulação apresenta uma restrição não linear.

Rocha et al. (2008) trabalha com duas formulações uma baseada em variavéis de tempo de conclusões e outra de atribuição e data de posicionamento minimizando o atraso ponderado. Desenvolve um algoritmo (B&B) e utiliza método polinomial para obter rapidamente limites inferiores no problema de máquinas paralelas não relacionadas com sequência dependentes de tempo de preparação e datas de entregas. Uma solução da methaheurística GRASP é usada como um limite superior. A performace do algorítmo é satisfatória para instâncias de até 25 tarefas e 6 máquinas.

Balakrishnan et al. (1999) aborda o problema de máquinas paralelas uniforme onde minimizar a soma dos custos de adiantamento e de atraso. Para problemas de grande escala aplicando o metódo de decomposição de Benders. Embora foi capaz de resolver os problemas com até 12 tarefas e 3 máquinas , o esforço computacional foi bastante substancial consumindo muito tempo.

Pessoa et al. (2010) desenvolve a formulação variável arco indexada e abordou o problema $P||\sum w_j T_j$. Propõe um algoritmo *branch cut and price* reforçado por técnicas práticas, incluindo um procedimento de programação dinâmica. O método resultante permite a solução de instâncias de até 100 postos de trabalho, tendo 2 ou 4 máquinas.

Avalos-Rosales et al. (2013) propõe uma reformulação da formulação em rede. O problema de máquinas paralelas não relacionadas com tempos de preparação dependentes da sequência e da máquina minimizando o tempo máximo conclusão é resolvido usando um *branch and bound* de um *solver* comercial. Segundo o autor o modelo possibilita resolver instâncias quatro vezes maior que o resolvido anteriormente na literatura e três vezes mais rápido.

Na proxíma Subseção 2.3.1 é feito uma introdução ao método de decomposição de Benders com o objetivo de dar suporte à capítulos posteriores.

2.3.1 Método de decomposição de Benders

O método de decomposição de Benders foi proposto por Benders (1962) para trabalhar com problemas de programação linear inteira mista e não linear. O método lida com situações onde estão presentes variáveis complicantes. A ideia principal por trás do método é buscar, de forma iterativa, um valor para as variáveis complicadoras que possa ser fixado para então resolver o problema de forma decomposta. Benders define um algoritmo de relaxação que decompõe o problema original em dois problemas mais simples denominados problema mestre (PM) e o subproblema de decomposição de Benders (SP).

O PM é uma versão relaxada do problema original com o conjunto de variáveis inteiras e suas respectivas restrições. Como ele é um problema aproximado, sua solução ótima é sempre um limite inferior (LI) para o valor da solução real. Enquanto que o SP é o próprio problema original com os valores das variáveis inteiras fixadas, temporariamente, pelo PM. A conjução de soluções do PM e do primal do subproblema providência um limite superior (LS) do problema original.

O algoritmo resolve iterativamente e separadamente o PM e o SP. Em cada iteração uma nova restrição, corte de Benders, é acrescentada ao PM. O corte de Benders emana da resolução do SP. O algoritmo se encerra quando os LI e LS se igualarem.

Considere o seguinte problema:

$$min\ c^T x + f^T y \tag{2.46}$$
$$s.a.\ Ax + By \geq b \tag{2.47}$$
$$Dy \geq d \tag{2.48}$$
$$x \geq 0 \tag{2.49}$$
$$y \geq 0 \text{ e inteiro} \tag{2.50}$$

Onde A é uma matriz mxn, B e D são matrizes do tipo mxp e qxp, respectivamente. x e c são vetores de tamanho n, enquanto y e f são vetores de tamanho p e finalizando os vetores b e d, são vetores de tamanho m e q na devida ordem.

Este problema pode ser reescrito da seguinte forma:

$$min\ f^T y + v(y) \tag{2.51}$$
$$s.a.\ Dy \geq d \tag{2.52}$$
$$y \in Z_+ \tag{2.53}$$

Onde,

$$v(y) = min\ c^T x \tag{2.54}$$
$$s.a.\ Ax \geq b - By \tag{2.55}$$
$$x \geq 0 \tag{2.56}$$

Considerando $Y = \{y \in Z^+ : Dy \geq d\}$ tem-se que para cada $y \in Y$ o valor de $v(y)$ é a solução ótima de um problema de programação linear cujo dual é escrito como:

$$max\ (b - By)^T u \tag{2.57}$$
$$s.a.\ A^T u \leq c \tag{2.58}$$
$$u \geq 0 \tag{2.59}$$

O problema (2.57 - 2.59) é conhecido como o subproblema de decomposição de Benders (SP). Usando a teoria da dualidade, o valor de $v(y)$ pode ser encontrado usando a formulação primal ou dual, no entanto, é mais conveniente utilizar o dual pois a região

de viabilidade independe da variável y. Definindo o conjunto $\Omega = \{u \geq 0 : A^T u \leq c\}$ como sendo o conjunto de soluções viavéis do SP. Considere que $\Omega \neq \emptyset$, de outro modo, o problema primal será inviável ou ilimitado. Assim pelo teorema de Minkowski, o conjunto Ω é composto de pontos extremos u^i $(\forall i = 1, ..., N)$, representando as soluções limitadas, e raios extremos u^r $(\forall r = 1, ..., Q)$, representando as soluções duais ilimitadas.

Como deseja-se somente as soluções primais viáveis e tem-se que Ω é um poliedro possuindo um número finito de geradores pode-se eliminar os valores de y que geram as soluções duais ilimitadas utilizando as seguintes restrições obtidas pelo lema de Farkas.

$$(b - By)^T u^r \leq 0 \qquad r = 1, ..., Q \tag{2.60}$$

Assim o problema (2.51 - 2.53), reformulado é:

$$min\ f^T + max\ \{(b - By)^T u^i : i = 1, ..., N\} \tag{2.61}$$
$$s.a.\ (b - By)^T u^r \leq 0 \qquad r = 1, ..., Q \tag{2.62}$$
$$y \in Y \tag{2.63}$$

Observe que a solução do problema interno de maximização é sempre um ponto extremo de Ω, e então ele pode ser substituído por uma variável contínua η na formulação anterior. Daí obtem-se o problema mestre de decomposição de Benders.

$$min\ f^T y + \eta \tag{2.64}$$
$$s.a.\ \eta \geq (b - By)^T u^i \qquad \forall i = 1, ..., N \tag{2.65}$$
$$(b - By)^T u^r \leq 0 \qquad \forall r = 1, ..., Q \tag{2.66}$$
$$y \in Y \tag{2.67}$$

O PM é equivalente ao problema original com menos variáveis e mais restrições, uma vez que possui uma restrição para cada ponto e raio extremo do poliedro Ω. Dependendo da dimensão do problema pode-se ter um número muito elevado de restrições dificultando sua resolução. Entretanto, nem todas as restrições estão ativas na solução ótima do problema original. Assim pode-se resolver o PM relaxado, i.e. resolver PM apenas com o conjunto Y de restrições e ir adicionando as restrições (2.65) e (2.66) iterativamente. Estas restrições são chamadas de cortes de Benders e são geradas pela solução do SP.

A cada iteração obtêm-se temporárias configurações de y e em cada configuração um SP é resolvido encontrando ou um raio extremo ou um ponto extremo. Se a solução ótima encontrada do SP corresponder a um ponto extremo a restrição (2.65) é adicionada ao PM, ele é conhecido como corte de otimalidade (corte tipo I). Caso ao resolver SP obtenha um raio extremo a restrição (2.66) é adicionada ao PM, ela é dita ser o corte de viabilidade (corte tipo II). O algoritmo resolve então o PM e SP iterativamente até os limites LS e LI convergirem. Obtendo assim a solução ótima do problema original.

O método de decomposição de Benders é uma técnica bem conhecida e bem sucedida na resolução de problemas de programação inteira mista. Sua eficiência computacional para problema de grande porte foi comprovada no trabalho pioneiro de Geoffrion and Graves (1974) em um problema de desenho de sistema de distribuição de *multicommodities*. O método também já foi utilizado com sucesso em problemas de localização de facilidades p-mediana (Magnanti and Wong, 1981); problemas de localização de concentradores (de Camargo et al., 2008, 2009b,a; Contreras et al., 2011); problemas de desenho de rede tais como desenho com custo fixo (Costa, 2005), desenho com duas camadas (Fortz and Poss, 2009), desenho de redes do tipo eixo-raio (Gelareh, 2008; de Sá et al., 2013; Martins de Sá et al., 2015); problemas de atribuição de locomotivas e carros (Cordeau et al., 2000); problemas de rotemamento de aeronaves e alocação de tripulação (Mercier, 2008; Papadakos, 2008, 2009); problemas de programação de produção (Aardal and Larsson, 1990; Tran and Beck, 2012; Şen and Bülbül, 2015; Bülbül and Şen, 2015), entre outros.

A maioria dos autores citados no parágrafo anterior observaram que a versão clássica do método de decomposição de Benders apresenta um desanimador desempenho computacional. Sendo assim, a ideia é introduzir técnicas e/ou cortes adicionais para acelerar a convergência do método. Na Subseção seguinte será apresentado algumas delas, as quais, serão utilizadas no trabalho abordado.

2.3.2 Técnicas de aceleração do método

Nesta Subseção são apresentadas algumas técnicas para acelerar a covergência do algoritmo de decomposição de Benders sendo elas o acréscimo de cortes desagregados e cortes pareto ótimos.

2.3.2.1 Múltiplos cortes

A desagregação do corte de Benders Primal foi usado por Dogan and Goetschalckx (1999) em um sistema de produção-distribuição multi-período e por Üster et al. (2007) no projeto de rede da cadeia de abastecimento de circuito fechado. Os cortes desa-

gregadas são obtidos a partir de problemas com uma estrutura específica, isto é, o subproblema de Benders deve permitir ser separado em subproblemas independentes. Assim em cada iteração, os cortes múltiplos desagregados formados pelas melhores soluções duais correspondentes aos subproblemas independentes serão anexado ao PM simultaneamente. Estes cortes, que incluem exatamente as mesmas informações que o corte de Benders primal irá restringir o espaço de solução do PM de uma forma mais precisa.

A desagregação é um metódo de multipla geração de corte onde a ideia é acrescentar ao PM múltiplos cortes de Benders simultaneamente. Essa técnica é uma boa estratégia para a convergência de forma rápida diante do algoritmo de decomposição de Benders original além de manter o equilíbrio entre o número de iterações e o tempo de CPU gasto em cada iteração para gerar estes cortes adicionais. Segundo Tang et al. (2013) esse equilíbrio deve ser baseado na ideia que para produzir um corte, tempo adicional é gasto para completar uma iteração, mas o tempo total gasto para resolver o PM de forma iterativa é menor devido a um número menor de iterações.

2.3.2.2 Cortes pareto ótimo

Quando o subproblema de Benders é degenerados, ou seja existe mais de uma solução ótima, Magnanti and Wong (1981) propõe adicionar ao PM, a cada iteração, cortes pareto ótimo. Magnanti and Wong define um corte pareto ótimo como o corte que domina todos os outros cortes, ou seja se $\Pi = \{u \geq 0 : Au \leq c\}$ é o conjunto de valores viáveis para as variáveis duais u então o corte de Benders associado a $u_1 \in \Pi$ domina o corte associado a $u_2 \in \Pi$, se $(b - By)^T u_1 \geq (b - By)^T u_2$.

Para gerar esses cortes, Magnanti and Wong propõe utilizar pontos auxiliares, os chamados *core point*. Um ponto y^0 é um *core point* se ele pertence ao interior relativo da casca convexa, isto é, se $y^0 \in \mathbf{ir}(Y^c)$, onde $\mathbf{ir}(Y^c)$ é o interior relativo da casca convexa Y^C de Y.

Considere y^0 um *core point* válido, u^h e $f(u^h)$, a solução ótima e o valor da função objetiva do subproblema de Benders, associado à solução y^h do PM na iteração h. O subproblema Magnanti and Wong (SMW) será:

$$max\ (b - By^0)^T u \tag{2.68}$$
$$s.a.\ (b - By^h)^T u = f(u^h) \tag{2.69}$$
$$A^T u \leq c \tag{2.70}$$
$$u \geq 0 \tag{2.71}$$

Esse subproblema gera à cada iteração do método o corte pareto ótimo (2.72):

$$\eta \geq (b - By)^T u^h \tag{2.72}$$

A cada iteração do método de decomposição de Benders Magnanti and Wong (1981) em seu algoritmo inicial, resolve um SP relacionado a solução corrente do problema mestre e o SMW relacionado a solução corrente do PM, a solução do SP e a um determinado *core point*. Essa depêndencia SMW ao SP causa desvantagens a técnica, primeiro se o SP for complicado para se resolver, soluções sub-ótimas não poderão ser utilizadas. Segundo, a restrição da dependência (2.69) pode não colaborar com o desempenho do algoritmo, tornando o método frágil á instabilidade numérica. Papadakos (2008); Mercier et al. (2005) alerta para outro ponto desvantajoso da técnica, a necessidade de obter *core point* a cada iteração, uma vez que, em cada iteração o interior relativo é modificado.

Papadakos (2008) propõe simplificar o SMW eliminando a restrição (2.72). Esse novo subproblema é nomeado subproblema independente de Magnanti and Wong (SIMW).

$$max\ (b - By^0)^T u \tag{2.73}$$
$$A^T u \leq c \tag{2.74}$$
$$u \geq 0 \tag{2.75}$$

Para atualização do *core point* ele utiliza a combinação convexa entre um *core point* válido e o ponto inteiro viável referente a solução do PM. Sendo assim considere a solução ótima do PM na iteração h, $y^h \in Y$, tem-se então o *core point* y^{0h} atualizado da seguinte forma:

$$y^{0(h+1)} = \lambda y^{0h} + (1 - \lambda) y^h \tag{2.76}$$

Onde o λ acima representa um valor $0 \leq \lambda \leq 1$.

A teoria de Papadakos (2008) tem a desvantagem de considerar que o PM fornece somente soluções viáveis, caso o PM apresentar uma solução inviável a *core point* atualizado não possui nenhuma garantia de validade.

2.3.2.3 Cortes *lifting*

Lifting foi originalmente estudado por Gomory (1969) e foi estudado também por Padberg (1973) e Wolsey (1976). Essa técnica toma como uma nova desigualdade válida que é gerada sob um subconjunto apropriado de variáveis e anexa a ela novas variáveis que não estavam originalmente presentes. Coeficientes diferentes de zero são introduzidos de modo que a nova desigualdade permanece válida.

Tang et al. (2013) propõe um corte que melhora a qualidade dos corte pareto-otimo. Segundo ele um corte pareto-otimo é *lifted* apartir do corresponde corte de otimalidade de Benders mas o efeito do *lifting* é limitado pois depende da solução do problema dual. Para melhorar ainda mais o *lifting* no corte pareto-ótimo apresenta a geração de cortes pareto de alta densidade. Um corte de alta densidade é um corte que abrange um elevado número de variáveis de decisão do problema principal restringindo seu espaço de solução.

Capítulo 3

Formulação matemática

"Desde de que não seja impossível o improvável é provável que aconteça"

Autor desconhecido

A abordagem clássica de Benders (Benders, 1962; Geoffrion and Graves, 1974) é tida como inadequado para o problema de sequenciamento, porque requer que o subproblema seja um problema linear contínuo ou problema de programação não-linear. O problema de sequenciamento é altamente combinatória e não tem nenhum modelo de programação linear ou não-linear prático. Daí a idéia de decomposição de Benders ser estendido para a decomposição *logic-based* Benders que acomoda um subproblema arbitrária, como um problema de programação discreta.

3.1 Método de decomposição *logic-based* Benders

O método de decomposição *logic-based* Benders foi introduzido por Hooker (1994) e Hooker and Yan (1995), a teoria geral foi abordada por Hooker and Yan (1995) e aplicação em planejamento e sequenciamento foi implementado em Jain and Grossmann (2001).

O método é uma generalização do método de decomposição de Benders que permite explorar a relação entre um problema de programação linear inteira mista (PLIM) e programação por restrições (PR). O PLIM ocorre na abordagem da atribuição das tarefas à máquinas no problema mestre (PM) e PR fornece uma técnica de sequenciamento eficiente que pode ser aplicada no subproblema. Portanto, o metódo de decomposição *logic-based* Benders une PLIM e PR numa abordagem integrada que obtém soluções ótimas.

Tran and Beck (2012) aplicou o método em um problema de sequenciamento de máquinas paralelas não relacionadas com tempos de preparação dependentes da sequência

e da máquina minimizando o tempo máximo conclusão das tarefas. Percebe-se o subproblema como m independentes problemas do caxeiro viajante assimétrico. Então, utiliza a abordagem de um modelo híbrido que faz uso de uma mistura do PM e um *solver* especializado para o subproblema do caixeiro viajante(PCV).

O problema mestre é usado para atribuir tarefas para máquinas enquanto os subproblemas obtém sequenciamentos ideais em cada máquina, dadas as atribuições do problema mestre. Segundo Tran and Beck (2012) o algoritmo *logic-based* Benders é capaz de encontrar as melhores soluções até seis ordens de grandeza mais rápido, bem como resolver problemas seis vezes de tamanho possível anteriormente com um modelo de programação inteira mista e duas vezes o tamanho que o algoritmo de *branch-and-bound* utilizado por Rocha et al. (2008) para um problema semelhante.

A seguir será desenvolvido a ideia do método de decomposição do *logic-based Benders* utilizada por Tran and Beck (2012) no problema PSMPRTS, abordado nesta dissertação. Para tal abordagem será utilizado a definições de conjuntos e variáves abaixo. A formulação variável em rede, Subseção 2.2.4 será utilizada assim como em Tran and Beck (2012).

Considere $x_j^m, j \in N, m \in M$ fixadas tal que satisfaça a restrição (3.9). O subproblema é desagregado em m subproblemas independentes de uma única máquina. O problema de uma única máquina pode ser percebido como um PCV assimétrico. Assim, seja $x_{jm}^h = x_j^m$ as variáveis fixadas e considere as seguintes definições:

Conjuntos:

$A = \{(i,j) : i \in N \cup \{0\} \ e \ j \in N\}$

$A^1 = \{(i,j) : i \in N \cup \{0\} \ e \ j \in N \cup \{n_j + 1\}\}$

$A^2 = \{(i,j) : i,j \in N \cup \{0\}\}$

Variáveis:

y_{ijm}^{sub} $\begin{cases} 1, & \text{se na máquina } m \text{ a tarefa } i \text{ precede a tarefa } j, \\ 0, & \text{caso contrário.} \end{cases}$

C_j^{sub} Tempo de conclusão da tarefa j no subproblema.

O subproblema *logic-based* Benders para cada máquina ($SPLG_m$) é dado por:

$$min \sum_{j \in N} w_j C_j^{sub} \tag{3.1}$$

$s.\ a:$

$$y_{ijm}^{sub} + y_{jim}^{sub} \geq x_{im}^{h} + x_{jm}^{h} - 1 \qquad \forall\ (i,j) \in A^2 \tag{3.2}$$

$$C_j^{sub} \geq C_i^{sub} + s_{ij}^{m} + p_j^{m} - (1 - y_{ijm}^{sub})\mathcal{M} \qquad \forall\ (i,j) \in A \tag{3.3}$$

$$y_{0jm}^{sub} = x_{jm}^{h} \qquad \forall\ j \in N \tag{3.4}$$

$$C_j^{sub} \geq 0 \qquad \forall\ j \in N \tag{3.5}$$

$$y_{ijm}^{sub} \in \{0,1\} \qquad \forall\ (i,j) \in A^2 \tag{3.6}$$

O conjunto de restrições (3.2) garante que se duas tarefas estão atribuídas a mesma máquina então existe uma disjunção. Em (3.3) é realizado o cálculo do tempo de conclusão das tarefas e o sequenciamento na máquina m. O conjunto de restrições (3.4) dar valor para a variável y_{0jm}^{sub}. Finalizando (3.5) e (3.6) asseguram respectivamente a não negatividade de C_j^{sub} e a integralidade das variáveis binárias.

Se a soma ponderada dos tempos de conclusão das tarefas encontrada no subproblema for menor ou igual ao do problema mestre do *logic-based* Benders (PMLG) então não há corte para ser acrescentado. Caso contrário, cortes são criados e adicionados ao PMLG. O corte é dado da seguinte forma:

$$\sum_{\substack{j \in N_1^{mh}: \\ j \neq 0}} w_j C_j \geq C_{mh}^{sub*}(1 - \sum_{j \in N_1^{mh}} (1 - x_j^m)) \qquad \forall m \in M, h \in H \tag{3.7}$$

Onde N_1^{mh} é o conjunto das tarefas atribuídas à máquina m na iteração h e C_{mh}^{sub*} é o valor ótimo do subproblema na máquina m na iteração h. Este corte é um corte simples de Benders, corte *nogood*, ele é muito fraco e afirma que a solução do subproblema não pode ser melhorada a não ser que as variáveis fixadas originadas do mestre sejam diferentes. Para o problema mestre considere as variáveis:

y_{ij}^m $\begin{cases} 1, & \text{se na máquina } m \text{ a tarefa } i \text{ precede a tarefa } j, \\ 0, & \text{caso contrário.} \end{cases}$

x_j^m se a tarefa j é atribuída a máquina m.

C_j Tempo de conclusão da tarefa j.

O problema mestre do *logic-based* Benders (PMLG) é dado por:

$$min \sum_{j \in N} w_j C_j \tag{3.8}$$

$s.\ a:$

$$\sum_{m \in M} x_j^m = 1 \qquad \forall\ j \in N \tag{3.9}$$

$$x_j^m = \sum_{(i,j) \in A} y_{ij}^m \qquad \forall\ m \in M,\ j \in N \tag{3.10}$$

$$x_i^m = \sum_{(i,j) \in A^1} y_{ij}^m \qquad \forall\ m \in M,\ i \in N \tag{3.11}$$

$$C_j \geq p_j^m x_j^m + \sum_{(i,j) \in A} s_{ij}^m y_{ij}^m \qquad \forall\ m \in M,\ j \in N \tag{3.12}$$

$$\sum_{\substack{j \in N_1^{mh}: \\ j \neq 0}} w_j C_j \geq C_{mh}^{sub*} \big(1 - \sum_{j \in N_1^{mh}} (1 - x_j^m)\big) \qquad \forall m \in M, h \in H \tag{3.13}$$

$$C_j \geq 0 \qquad \forall\ j \in N \tag{3.14}$$

$$x_j^m \in \{0, 1\} \qquad \forall\ m \in M;\ j \in N \tag{3.15}$$

$$0 \leq y_{ij}^m \leq 1 \qquad \forall\ m \in M;\ (i,j) \in A^1 \tag{3.16}$$

O conjunto de restrições (3.9) garante que toda tarefa é alocada a uma máquina. Em (3.10) garante que se uma tarefa j é atribuída a uma máquina então ela será sucessora de uma única tarefa e (3.11) afirma que se a tarefa i é atribuída a máquina m então ela será antecessora de exatamente uma tarefa j. No conjunto de restrições (3.12) tem-se um limitante inferior para o tempo de conclusão de cada tarefa em dada máquina. As restrições (3.10 - 3.12) são consideradas como o subproblema relaxado. Os cortes adicionados são representados em (3.13). Finalizando (3.14), (3.15) e (3.16) asseguram respectivamente a não negatividade de C_j, a integralidade das variáveis binárias e por último que a variável y_{ij}^m é um valor entre 0 e 1.

O problema mestre e o subproblema irão iteragir entre si até uma solução ótima ser encontrada. A otimalidade é garantida se o valor da função objetivo do PMLG é igual ao menor valor corrente encontrado pela soma das n_m soluções dos SPLGs.

3.1.1 Algoritmo do método de decomposição *logic-based* Benders

Seja S_{PMLG}^*, a solução ótima do problema mestre, $S_{SPLG_m}^*$ a solução do subproblema para a máquina m. Então, o algoritmo do *logic based* Benders é escrito como:

Algoritmo 1: Algoritmo de decomposição *logic-based* Benders

1 Faça $LS = +\infty$, $LI = 0$, $GAP = 1$
2 Se $GAP < \epsilon$, então pare. Fim da execução, a solução ótima foi obtida.
3 Resolva o PMLG (3.8)-(3.16) atualize x_j^m e faça $LI = S^*_{PMLG}$.
4 Atualize x_j^m no SPLG (3.1)-(3.6).
5 Para cada máquina, resolva o SPLG (3.1)-(3.6).
Adicione ao PMLG o corte (3.7).
6 Faça: LS =min $\{LS\ ,\ S^*_{SPLG_m}\ \}$ e volte ao passo 2.

Na linha 1 é inicializado os limites superior, inferior e o gap. Na linha 2, o algoritmo é executado até que o gap se aproxime de zero, baseado na tolerância (10^{-4}). A execução do laço principal do algoritmo consiste em (1º) resolver o PMLG na Linha 3, obter o valor da variáveis x_j^m e atualizar o limite inferior; (3º) fixar as variáveis x_j^m no SPLG, linha 4. Na linha 5, (4º) para cada máquinas; resolve SPLG adicionando o corte (3.7). A linha 6 atualiza o limite superior.

3.2 Formulação variável de posicionamento e fluxo

As formulações existentes até o momento foram surgindo e sendo estudadas de acordo que se fazia necessárias novas conquistas na busca de conhecimento na área. Após estudar as formulações apresentadas na Seção 2.2 sentiu-se a necessidade de criar uma nova formulação. Dessa aspiração surgiu a formulação baseada em duas variáveis, uma responsável pela atribuição da tarefa à máquina e a posição e a variável de fluxo encarregada de fazer o sequenciamento. Para a nova formulação considere N, M, P os conjuntos de tarefas, máquinas e posições respectivamente já definidos anteriormente e w_i a prioridade da tarefa i. Admita também a notação dos seguintes dados:

$N_x = \{0, ..., 2n_j^2\}$ - Conjunto de nós.

E_x - Conjunto de arcos.

$G(N_x, E_x)$ - Hipergrafo

$\phi_i^l = u$ - Função que retorna algum $u \in N_x$, onde $i \in N,\ l \in P$.

$\psi_u^1 = j$ - Função que retorna a tarefa j que se posiciona em u, onde $u \in N_x$.

$\psi_u^2 = l$ - Função que retorna a posição l quando u é utilizado, onde $u \in N_x,\ l \in P$.

Para construção do conjunto E_x considere os conjuntos $N_{xi} = \{1, 3, ..., 2n_j^2 - 1\}$ e $N_{xp} = \{2, 4, ..., 2n_j^2\}$ daí

$$E_x = \{(0,u) : \forall i \in N, u = 2n_j(i-1)+1\} \bigcup \{(u,u+1) : u \in N_{xi} \text{ e } v = u+1\} \bigcup$$
$$\{(u,v) : u \in N_{xp}, v \in N_{xi} \text{ onde } \psi^1_{u-1} \neq \psi^1_v \text{ e } \psi^2_{u-1} + 1 = \psi^2_v\}$$

A Figura 3.1 mostra o grafo para três tarefas e a tarefa fictícia 0 e a Figura 3.2 o hipergrafo $G(N_x, E_x)$ para essas mesmas três tarefas. Considere l as posições, os valores duplicados representam as tarefas e os demais números são os nós pertencentes a N_x. Nele o caminho só poderá ir de um nó para seu consecutivo se ambos estiverem na mesma posição, essa condição permite contabilizar o tempo de processamento de cada tarefa alocada a alguma máquina e posição. Caso a ligação entre os nós aconteça de uma posição para outra tem-se a contabilização do tempo de preparação entre as tarefas dessas posições.

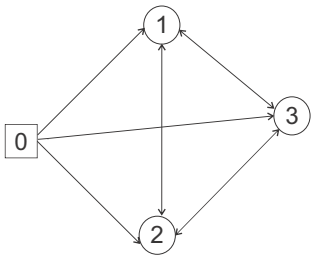

Figura 3.1: Representação do grafo para 3 tarefas

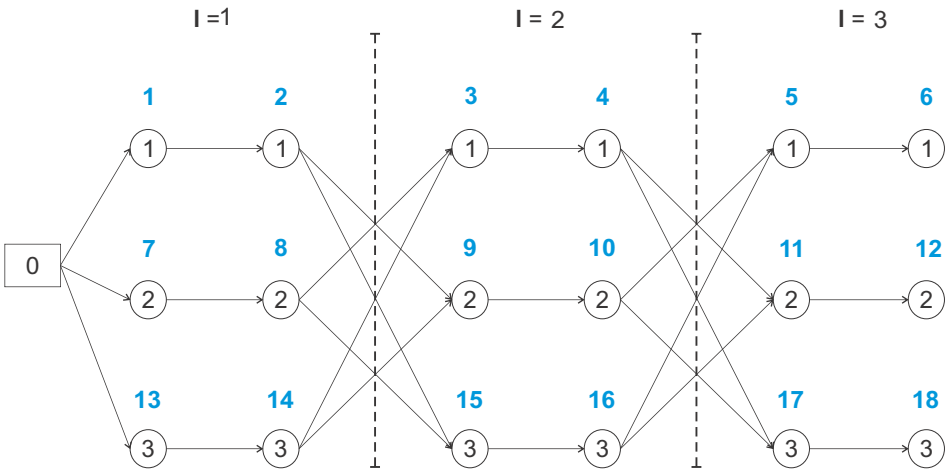

Figura 3.2: Construção do hipergrafo $G(N_x, E_x)$ para 3 tarefas

Sendo assim, o cálculo do tempo de processamento e o tempo de preparação das tarefas são obtidos através da função custo onde o arco $(u,v) \in E_x$ na máquina m possui um valor associado.

$$c_{uv}^{m} \begin{cases} 0 & \text{se } u = 0 \\ p_{\psi_u^1}^{m} & \text{se } v = u + 1\ , \quad u \in N_{xi} \text{ e } v \in N_{xp} \\ s_{\psi_{(u-1)}^1 \psi_v^1}^{m} & \text{se } \psi_{(u-1)}^1 \neq \psi_v^1 \ \text{ e } \ \psi_v^2 = \psi_{(u-1)}^2 + 1 \end{cases}$$

Observe que na definição acima os índices do tempo de processamento e do tempo de preparação estão associados as funções que retornam as tarefas que estão no nó especificado por elas.

As variáveis utilizadas são apresentadas a seguir onde a variável y_i^{ml} já é bem conhecida e não traz novidades uma vez que são utilizadas na formulação de atribuição e datas de posicionamento, Subseção 2.2.3. A outra variável f_{uv}^{ilm} trabalha com *multi-commodity* onde tem-se a duplicação de nós. Essa duplicação possibilitará contabilizar o tempo de processamento de uma tarefa atribuída a uma máquina e a uma posição. Quando o fluxo existente ocorre entre tarefas de posições diferentes tem-se contabilizado o tempo de preparação entre essas duas tarefas. A Figura 3.3 representa o Exemplo 1 nesta formulação.

f_{uv}^{ilm} Fluxo do arco (u, v) na máquina m com destino a *commodity* fictícia (i, l)

$$y_i^{ml} \begin{cases} 1 & \text{se a tarefa } i \text{ é alocada para a posição } l \text{ na máquina } m, \\ 0 & \text{caso contrário.} \end{cases}$$

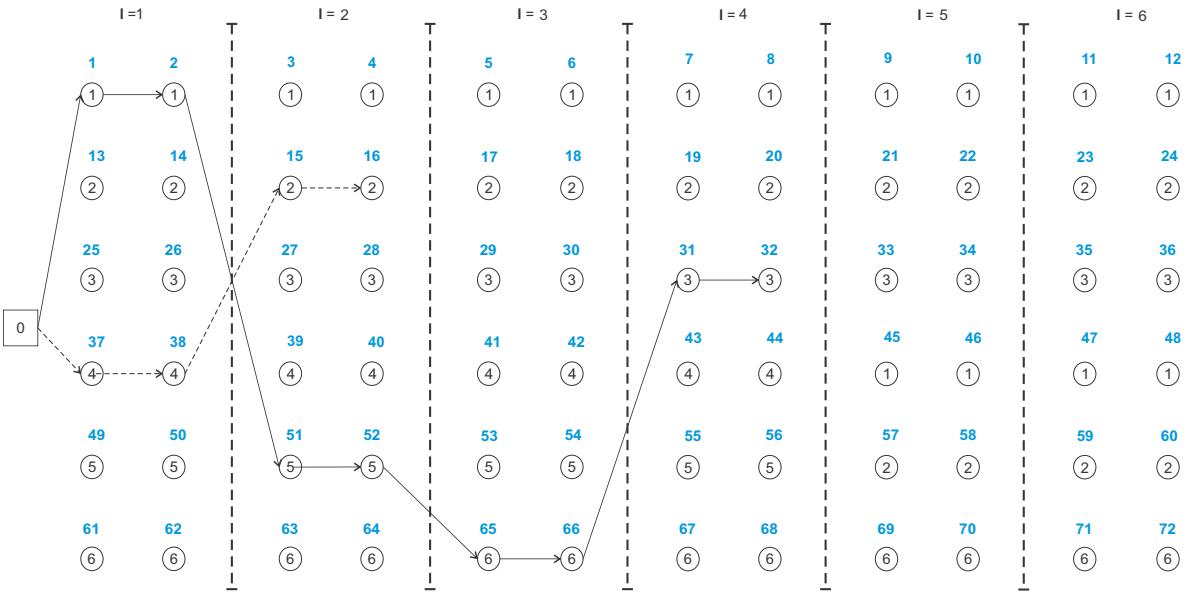

Figura 3.3: Solução do Exemplo 1 representado na nova formulação

Com o intuito de simplicar a apresentação dos modelos matemáticos considere no restante deste documento os índices $m \in M$; $i, j \in N$; $l,\ r \in P$; $u,\ v \in N_x$ e o

arco $(u, v) \in E_x$. A nova formulação para o problema de sequenciamento de máquinas paralelas não relacionadas com tempo de preparação dependentes da sequência e da máquina é dada por:

$$min \sum_{m \in M} \sum_{i \in N} \sum_{l \in P} \sum_{(u,v) \in E_x} w_i \; c_{uv}^{m} \; f_{uv}^{ilm} \tag{3.17}$$

$s.\ a:$

$$\sum_{m} \sum_{l} y_i^{ml} = 1 \qquad \forall\ i \tag{3.18}$$

$$\sum_{i} y_i^{ml} \leq 1 \qquad \forall\ m,\ l \tag{3.19}$$

$$\sum_{i} y_i^{ml} \leq \sum_{i} y_i^{m(l-1)} \qquad \forall\ m,\ l:\ l > 1 \tag{3.20}$$

$$\sum_{(0,v)} f_{0v}^{ilm} = y_i^{ml} \qquad \forall\ m,\ l,\ i \tag{3.21}$$

$$\sum_{(u,v)} f_{uv}^{ilm} = \sum_{(v,u)} f_{vu}^{ilm} \qquad \forall\ m,\ l,\ i, v : 0 \neq v \neq (\phi_i^l + 1) \tag{3.22}$$

$$\sum_{(u,\phi_i^l+1)} f_{u(\phi_i^l+1)}^{ilm} = y_i^{ml} \qquad \forall\ m,\ l,\ i \tag{3.23}$$

$$f_{\phi_j^r(\phi_j^r+1)}^{ilm} \leq y_j^{mr} \qquad \forall\ m,\ l,\ i,\ r,\ j \tag{3.24}$$

$$f_{uv}^{ilm} \geq 0 \qquad \forall\ m,\ l,\ i,\ (u, v) \tag{3.25}$$

$$y_i^{ml} \in \{0, 1\} \qquad \forall\ m,\ l,\ i \tag{3.26}$$

A função objetiva apresentada em (3.17) calcula o tempo de conclusão ponderada das tarefas. O conjunto de restrições (3.18) assegura que toda tarefa i é atribuída a uma máquina m e a uma posição l. Em (3.19) afirma que toda máquina e toda posição pode ter ou não uma tarefa atribuída. Se em uma máquina, a posição $l > 1$ possuir uma tarefa atribuída então outra tarefa deve ocupar a posição anterior, (3.20). O conjunto de restrições (3.21) afirma que se uma tarefa i é atribuída a posição l e a máquinq m então terá um fluxo saindo do nó fictício zero para outro nó. Para todo nó v tal que $0 \neq v \neq (\phi_i^l + 1)$ e a tarefa i é atribuída a máquina m e a posição l, o balanço de fluxo é garantido em (3.22). O conjunto de restrições (3.23) afirma que toda tarefa atribuída a uma posição e a uma máquina terá um fluxo de um nó para outro, eles podem está mesma posição ou em posições consecutivas. O conjunto de restrições (3.24) garante que só terá uma única variável de fluxo associada com uma variável de atribuição. Finalizando tem-se a não negatividade de f_{uv}^{ilm} sustentada por (3.25) e integralidade de y_i^{ml} por (3.26).

A formulação apresentada possui a característica que permite a decomposição, o que sugere como estratégia de resolução a utilização do método de decomposição de Benders. Essa decomponibilidade é devido a possibilidade de fixação da variável y_i^{ml} permitindo a decomposição em m subproblemas menores que consiste em encontrar o menor caminho para as tarefas atribuídas a determinada máquina e a determina posição. Sendo assim a Seção 3.3 apresenta o desenvolvimento do método para o problema abordado, introduzido na Seção 2.3.1. O desenvolvimento de um algoritmo de decomposição de Benders com cortes reforçados para o problema FPF será a contribuição metodológica deste trabalho.

3.3 Método de decomposição de Benders

O problema original é decomposto em dois problema menores, o problema mestre (PM) e o subproblema de Benders (SP), os quais serão resolvidos iterativamente e separadamente até os limites convergirem para a solução ótima. O limite superior é concebido pelo SP em conjução com o mestre, e o limite inferior origina-se do PM.

Esta Seção apresenta as formulações para o PM e SP, além das formulações adicionais necessárias para acelerar a convergência do método.

3.3.1 Subproblema de Benders

Considere as variáveis y_i^{ml}, $i \in N, m \in M$ e $l \in P$ fixadas tal que satisfaça as restrições (3.18) - (3.20). Então o problema é decomposto em independentes problemas de caminho mínimo. Seja $y_{ilm}^h = y_i^{ml}$, as variáveis fixadas. Daí encontra-se o seguinte subproblema primal de Benders.

$$min \sum_{m} \sum_{i} \sum_{l} \sum_{(u,v)} w_i \; c_{uv}^m \; f_{uv}^{ilm} \tag{3.27}$$

$s.\ a:$

$$\sum_{(0,v)} f_{0v}^{ilm} = y_{ilm}^h \qquad \forall\ m,\ l,\ i \tag{3.28}$$

$$\sum_{(u,v)} f_{uv}^{ilm} = \sum_{(v,u)} f_{vu}^{ilm} \qquad \forall\ m,\ l,\ i,\ v:\ 0 \neq v \neq (\phi_i^l + 1) \tag{3.29}$$

$$\sum_{(u,\phi_i^l+1)} f_{u(\phi_i^l+1)}^{ilm} = y_{ilm}^h \qquad \forall\ m,\ l,\ i \tag{3.30}$$

$$f_{\phi_j^r(\phi_j^r+1)}^{ilm} \leq y_{jrm}^h \qquad \forall\ m,\ l,\ i,\ r,\ j \tag{3.31}$$

$$f_{uv}^{ilm} \geq 0 \qquad \forall\ m,\ l,\ i,\ (u,v) \tag{3.32}$$

Esse subproblema pode ser desagregado em independentes subproblemas de caminho mínimo, um para cada máquina m.

$$min \sum_{m} \sum_{i} \sum_{l} \sum_{(u,v)} w_i \; c_{uv}^{m} \; f_{uv}^{ilm} \tag{3.33}$$

$s.\ a:$

$$\sum_{(0,v)} f_{0v}^{ilm} = y_{ilm}^{h} \qquad \forall\ l,\ i \tag{3.34}$$

$$\sum_{(u,v)} f_{uv}^{ilm} = \sum_{(v,u)} f_{vu}^{ilm} \qquad \forall\ l,\ i,\ v:\ 0 \neq v \neq (\phi_i^l + 1) \tag{3.35}$$

$$\sum_{(u,\phi_i^l+1)} f_{u(\phi_i^l+1)}^{ilm} = y_{ilm}^{h} \qquad \forall\ l,\ i \tag{3.36}$$

$$-f_{\phi_j^r(\phi_j^r+1)}^{ilm} \geq -y_{jrm}^{h} \qquad \forall\ l,\ i,\ r,\ j \tag{3.37}$$

$$f_{uv}^{ilm} \geq 0 \qquad \forall\ l,\ i,\ (u,v) \tag{3.38}$$

Para cada máquina m, sejam as variáveis duais π_{mli}, α_{mliv}, β_{mli} livres e $\gamma_{mlirj} \geq 0$ associadas as restrições (3.34), (3.35), (3.36) e (3.37) respectivamente. O subproblema de Benders para cada máquina (SP_m) é formulado como:

$$max \sum_{i} \sum_{l} y_{ilm}^{h} \pi_{mli} + \sum_{i} \sum_{l} y_{ilm}^{h} \beta_{mli} - \sum_{i} \sum_{l} \sum_{j} \sum_{r} y_{jrm}^{h} \gamma_{mlirj} \tag{3.39}$$

$s.\ a:$

$$\pi_{mli} + \alpha_{mliv} \leq 0 \qquad \forall\ l,\ i,\ (0,v) \tag{3.40}$$

$$-\alpha_{ilu} + \beta_{mli} - \sum_{j} \sum_{\substack{r: \\ \phi_j^r = u \\ \phi_j^r + 1 = v}} \gamma_{mlirj} \leq w_i c_{uv} \qquad \forall\ l,\ i,\ (u,v):\ v = \phi_i^l + 1 \tag{3.41}$$

$$\alpha_{mliv} - \alpha_{ilu} - \sum_{j} \sum_{\substack{r: \\ \phi_j^r = u \\ \phi_j^r + 1 = v}} \gamma_{mlirj} \leq w_i c_{uv} \qquad \forall\ l,\ i,\ (u,v):\ v = \phi_i^l + 1 \tag{3.42}$$

A solução ótima desse problema será sempre limitada o que oferece uma nova restrição para o PM, o conhecido corte de otimalidade ou corte tipo I. No corte abaixo considere π^h, α^h, β^h e γ^h pontos extremos correspondente a soluções do SP_m na iteração h e η_m uma variável contínua subestimadora a solução do SP_m:

$$\eta_m \geq \sum_i \sum_l y_i^{ml} \pi_{mli}^h + \sum_i \sum_l y_i^{ml} \beta_{mli}^h - \sum_i \sum_l \sum_j \sum_r y_j^{mr} \gamma_{mlirj}^h \tag{3.43}$$

Observando o corte acima nota-se que ele pode ser ser descomposto também para cada tarefa i. Sendo assim, tem-se na iteração h para toda máquina m e toda tarefa i o corte:

$$\eta_{mi} \geq \sum_l y_i^{ml} \pi_{mli}^h + \sum_l y_i^{ml} \beta_{mli}^h - \sum_l \sum_j \sum_r y_j^{mr} \gamma_{mlirj}^h \tag{3.44}$$

Seja H é o conjunto de iterações. Então, o PM com múltiplos cortes é dado por:

$$min \sum_m \sum_i \eta_{m,i} \tag{3.45}$$

$s.\ a:$

$$\sum_{m \in M} \sum_{l \in P} y_i^{ml} = 1 \qquad \forall\ i \tag{3.46}$$

$$\sum_{i \in N} y_i^{ml} \leq 1 \qquad \forall\ m,\ l \tag{3.47}$$

$$\sum_{i \in N} y_i^{ml} \leq \sum_{i \in N} y_i^{m(l-1)} \qquad \forall\ m,\ l : l > 1 \tag{3.48}$$

$$\eta_{mi} \geq \sum_l y_i^{ml} \pi_{mli}^h + \sum_l y_i^{ml} \beta_{mli}^h - \sum_l \sum_r \sum_j y_j^{mr} \gamma_{mlirj}^h \qquad \forall\ m,\ i,\ h \in H \tag{3.49}$$

$$y_i^{ml} \in \{0, 1\} \qquad \forall\ m,\ l,\ i \tag{3.50}$$

Muitos trabalhos da literatura relatam que uma implementação direta do metódo de decomposição de Benders produz um desempenho terrível a partir do ponto de vista do tempo computacional (Magnanti and Wong, 1981; Van Roy, 1986; Üster and Agrahari, 2011) entre outros. Isso é notado em problemas que são muito degenerado como é o caso do problema abordado. Sendo assim, para melhorar a corvergência do método de decomposição de Benders será gerado a cada iteração cortes paretos ótimos e cortes *lifting*, apoiado na Subseção 2.3.2.

3.3.2 Técnicas de aceleração de convergência

Para aplicar a ideia de Magnanti and Wong (1981) e Papadakos (2008) exposta na Subseção 2.3.2 considere y^0 um *core point* válido e y^h uma solução viável, que será atualizado a cada iteração h, através da combinação convexa:

$$y^{0(h+1)} = 0.5y^{0h} + 0.5y^{h} \tag{3.51}$$

O subproblema independente de Magnanti-Wong (SIMW) desagregado por máquinas é apresentado abaixo.

$$max \sum_{i}\sum_{l} y_{ilm}^{0h}\pi_{mli} + \sum_{i}\sum_{l} y_{ilm}^{0h}\beta_{mli} - \sum_{i}\sum_{l}\sum_{j}\sum_{r} y_{jrm}^{0h}\gamma_{mlirj} \tag{3.52}$$

$s.\ a:$

(3.40) – (3.42) (3.53)

Üster and Agrahari (2011) em seu trabalho nota que em determinados problemas é possível diminuir o espaço de soluções do subproblema contribuindo assim para a diminuição do tempo computacional de sua resolução. Para tal, Üster and Agrahari reescreve a função objetiva quando as variáveis fixadas são zeros, modificando o valor do coeficiente da variável dual associada à parâmetro, isso sem alterar o resultado da função objetivo do subproblema original. Para obter os cortes fortalecidos resolve-se dois novos subproblemas dentro de um domínio modificado. Esses novos subproblemas dão valores para as variáveis duais que providênciará o novo corte.

Adaptando a ideia para o problema tratado constroi-se a partir das soluções do PM na iteração h os conjuntos $N_1^{mh} = \{(l,i) : y_i^{ml} = 1\}$ e $N_0^{mh} = \{(l,i) : y_i^{ml} = 0\}$ onde $m \in M, i \in N$ e $l \in P$. Será construído um único subproblema, chamado de subproblema reduzido para cada máquina m (SPR_m) pois não há corte quando a variável fixada for zero, $y_{ilm}^{h} = 0$. Então para prover cortes fortalecidos será resolvido o SPR_m abaixo e não mais SP_m.

$$max \sum_{(l,i)\in N_1^{mh}} y_{ilm}^h \pi_{mli} + \sum_{(l,i)\in N_1^{mh}} y_{ilm}^h \beta_{mli} - \sum_{(l,i)\in N_1^{mh}} \sum_j \sum_r y_{jrm}^h \gamma_{mlirj} \tag{3.54}$$

$s.\ a:$

$$\pi_{mli} + \alpha_{mliv} \leq 0 \qquad \forall\ (l,i) \in N_1^{mh},\ (0,v) \in E_x \tag{3.55}$$

$$-\alpha_{ilu} + \beta_{mli} - \sum_j \sum_{\substack{r: \\ \phi_j^r = u \\ \phi_j^r + 1 = v}} \gamma_{mlirj} \leq w_i c_{uv} \qquad \forall\ (l,i) \in N_1^{mh},\ (u,v) \in E_x : v = \phi_i^l + 1 \tag{3.56}$$

$$\alpha_{mliv} - \alpha_{ilu} - \sum_j \sum_{\substack{r: \\ \phi_j^r = u \\ \phi_j^r + 1 = v}} \gamma_{mlirj} \leq w_i c_{uv} \qquad \forall\ (l,i) \in N_1^{mh},\ (u,v) \in E_x : v = \phi_i^l + 1 \tag{3.57}$$

Para $(l,i) \in N_0^{mh}$ tem-se as variáveis duais igualadas à zero.

Para melhorar a convergência Tang et al. (2013) propõe geração de cortes pareto de alta densidade melhorando a convergência por *lifting* nos cortes pareto-ótimo. Neste trabalho será utizado a ideia do *lifting* no SPR_m por possuir o espaço de soluções reduzido. O subproblema *lifting* utilizado foi desenvolvido e é estudado por um grupo de pesquisadores do programa de mestrado de engenharia de produção da Universidade Federal de Minas Gerais. Sendo assim considere o parâmetro $\theta = 0.1$, a variável de erro $e \geq 0$ e SPR_m^* o valor da função objetivo do SPR_m na máquina m na iteração h. Daí o subproblema de *lifting* em cada máquina ($SPRL_m$) é da forma:

$$min \sum_{(l,i)\in N_1^{mh}} \sum_j \sum_r \gamma_{mlirj} + \theta e^2 \tag{3.58}$$

$s.\ a:$

(3.55) – (3.57)

$$\sum_{(l,i)\in N_1^{mh}} y_{ilm}^h \pi_{mli} + \sum_{(l,i)\in N_1^{mh}} y_{ilm}^h \beta_{mli} - \sum_{(l,i)\in N_1^{mh}} \sum_j \sum_r y_{jrm}^h \gamma_{mlirj} + e \geq SPR_m^* \tag{3.59}$$

Sejam $\hat{\pi}_{mli}^h, \hat{\beta}_{mli}^h$ e $\hat{\gamma}_{mlirj}^h$ os valores duais obtidos do $SPRL_m$. O corte que será adicionado ao mestre para cada máquina m e cada tarefa i é:

$$\eta_{mi} \geq \sum_l y_i^{ml} \hat{\pi}_{mli}^h + \sum_l y_i^{ml} \hat{\beta}_{mli}^h - \sum_l \sum_j \sum_r y_j^{mr} \hat{\gamma}_{mlirj}^h \tag{3.60}$$

O problema de sequenciamento é muito combinatório e uma forma de amenizar esse comportamento é oferecer ao PM uma solução inicial. Posto isto, considere o algoritmo 2 onde sua solução é adicionada ao PM através da restrição (3.61). Essa restrição funciona como um limite inferior para o problema. Sendo assim, considere o parâmetro ϑ_{il}^{m} o tempo de conclusão para a tarefa i alocada a máquina m e à posição l.

$$\eta_{mi} \geq \sum_{l} \vartheta_{il}^{m} y_{il}^{m} \qquad \forall m \in M, i \in N \tag{3.61}$$

No algoritmo 2 considere $\$_j$ o conjuntos de pares (i, j) dos tempo de preparação já contabilizados e o conjunto de L_j, de tarefas processadas. O algoritmo da solução

inicial é dado por:

Algoritmo 2: Pseudo-código da solução inicial.

para $m \in M$ **faça**
  para $j \in N$ **faça**
    $\$_j \leftarrow \{\}$
    $L_j \leftarrow \{\}$
    $\vartheta_j^m \leftarrow p_j^m$
    $L_j \leftarrow \{j\}$
    $j_l \leftarrow j$
    para $l \in P$ **faça**
      $l_l \leftarrow l$
      se $l_l = 1$ **então**
        $\vartheta_{jl_l}^m \leftarrow \vartheta_j^m$
      fim
      senão
        enquanto $l_l > 1$ **faça**
          $(u, v) \leftarrow argmin\{s_{ij_l}^m : (i, j_l) \notin \$_j \ e \ (k, j_l) \notin \$_j\}$
          $\$_j \leftarrow \$_j \cup \{(u, v)\}$
          $\vartheta_j^m \leftarrow \vartheta_j^m + s_{uv}^m$
          $k \leftarrow argmin\{p_i^m : i \neq j_l \ e \ i \notin L_j\}$
          $L_j \leftarrow L_j \cup \{k\}$
          $\vartheta_j^m \leftarrow \vartheta_j^m + p_k^m$
          $l_l \leftarrow l_l - 1$
        fim
      fim
      $\vartheta_j^{ml} \leftarrow \vartheta_j^m$
    fim
  fim
fim

3.3.3 Problema mestre

Após as discussões das Subseções anteriores o PM completo é:

$$min \sum_{m} \sum_{i} \eta_{m,i} \tag{3.62}$$

$s.\ a:$

$$\sum_{m \in M} \sum_{l \in P} y_i^{ml} = 1 \qquad \forall\ i \tag{3.63}$$

$$\sum_{i \in N} y_i^{ml} \leq 1 \qquad \forall\ m,\ l \tag{3.64}$$

$$\sum_{i \in N} y_i^{ml} \leq \sum_{i \in N} y_i^{m(l-1)} \qquad \forall\ m,\ l : l > 1 \tag{3.65}$$

$$\eta_{mi} \geq \sum_{l} y_i^{ml} \pi_{mli}^h + \sum_{l} y_i^{ml} \beta_{mli}^h - \sum_{l} \sum_{r} \sum_{j} y_j^{mr} \gamma_{mlirj}^h \qquad \forall\ m,\ i,\ h \in H \tag{3.66}$$

$$\eta_{mi} \geq \sum_{l} y_i^{ml} \hat{\pi}_{mli}^h + \sum_{l} y_i^{ml} \hat{\beta}_{mli}^h - \sum_{l} \sum_{j} \sum_{r} y_j^{mr} \hat{\gamma}_{mlirj}^h \qquad \forall\ m,\ i,\ h \in H \tag{3.67}$$

$$\eta_{mi} \geq \sum_{l} \vartheta_{il}^m y_{il}^m \qquad \forall\ m,\ i \tag{3.68}$$

$$y_i^{ml} \in \{0, 1\} \qquad \forall\ m,\ l,\ i \tag{3.69}$$

Onde H é o conjunto de iterações.

3.3.4 Algoritmo do método de decomposição de Benders fortalecido

Seja S_{PM}^*, a solução ótima do problema mestre, $S_{SPR_m}^*$ a solução do subproblema reduzido para a máquina m e GAP dado por $\frac{LS-LI}{LS}100$. Então, o algoritmo de decomposição de Benders usando cortes fortalecidos é escrito como:

Algoritmo 3: Algoritmo de decomposição de Benders fortalecido

1 Encontre solução inicial, (2):
  Adicione ao PM a restrição (3.61).
2 Faça $LS = +\infty$, $LI = -\infty$, $GAP = 1$
3 Se $GAP < \epsilon$, então pare. Fim da execução, a solução ótima foi obtida.
4 Resolva o SIMW (3.52)-(3.53) para cada máquina m:
  Adicione ao PM um corte tipo I usando (3.44).
5 Resolva o PM (3.62)-(3.69) e faça $LI = S^*_{PM}$.
6 Atualize y_i^{ml} no SPR_m (3.54)-(3.57).
7 Para cada máquina, resolva o SPR_m (3.54)-(3.57).
  Adicione ao PM o corte de otimalidade usando (3.44).
  Resolva o $SPRL_m$ (3.58)-(3.59).
  Adicione ao PM o corte de *lifting* usando (3.60).
8 Faça: LS =min $\{LS\ ,\ S^*_{SPR_m}\}$.
9 Atualize os *core points* usando (3.51) e volte ao passo 3.

Na linha 1 do algoritmo, é encontrada uma solução inicial através do algoritmo 2 adicionando a restrição limitante. Na linha 2 é inicializado os limites superior e inferior e o gap. Na linha 3, o algoritmo é executado até que o gap se aproxime de zero, baseado na tolerância (10^{-4}). A execução do laço principal do algoritmo consiste em (1º) resolver o SIMW para cada máquina, como será sempre viável irá gerar múltiplos cortes de otimalidade na Linha 4; (2º) na Linha 5, resolve o PM obtêm-se o valor da variáveis y_i^{ml} e atualiza o limite inferior; (3º) fixa as variáveis y_i^{ml} no SPR_m, linha 6. Na linha 7, tem-se o (4º) momento onde para cada máquinas; resolve SPR adicionando corte de otimalidade (3.44), resolve $SPRL_m$ adicionando o corte de *lifting* (3.60). A linha 8 atualiza o limite superior e na linha 9 atualiza o *core points*.

Os resultados de testes computacionais comparando a implementação dos Algoritmos variantes do Benders, *logic based Benders* e o Bender fortalecido com o aplicativo comercial CPLEX são apresentados no próximo Capítulo.

Capítulo 4

Resultados computacionais

O objetivo desde trabalho é estudar o problema com o intuito de utilizar um método exato em sua resolução. Para alcançar este objetivo foi proposto uma nova formulação e aplicado o método de decomposição de Benders. Realizou uma série de testes para avaliar a performace das formulações já existententes e da nova formulação além de avaliar o algoritmo proposto. Os testes possibilitaram também, avalia-lo e compara-lo com um algoritmo de decomposição *logic-based* Benders. Para tal abordou $n_m \in \{2, 4\}$ e $n_j \in \{6, 8, 10, 12, 15, 20\}$. Os tempo de processamento, tempo de preparação e a prioridade foram escolhidos sobre uma distribuição uniforme tais que $p_j^m \sim DU[1,\ \sigma_1]$ onde $\sigma_1 \in \{10, 100\}$, $s_{ij}^m \sim DU[1,\ \sigma_2]$ com $\sigma_2 \in \{25, 50\}$, $w_j \sim DU[1,\ n_j]$. Todas as instâncias foram geradas aleatóriamente e modificadas para respeitar a desigualdade triângular. Cada modelo de PIM foi modelado em AMPL e resolvido sobre um tempo limite de 3600 segundos pelo CPLEX 12.6 em um computador com sistema operacional *Linux* 64 *bits* com o processador *Intel(R) Xeon(R) CPU E5630* com $2.53GHz$ e $24GB$ de memória *RAM* utilizando 1 *threads*.

4.1 Comparação das formulações

Utilizou uma série de medidas para avaliar cada formulação PIM. As medidas incluem o tempo computacional gasto dentro de um tempo limite de 3600 segundos, o *gap* de linearização, o *gap* de otimalidade, a quantidade de cortes aplicados pelo CPLEX e o número de nós *Branch and Bound* (B & B) explorados.

Para melhor compreensão das tabelas considere as notações abaixo:

VTC - Formulação variável tempo de conclusão

VOL - Formulação variável de ordenação linear

VAP - Formulação variável de atribuição e data de posicionamento

VR - Formulação variável em rede

VIT - Formulação variável indexada no tempo

VAIT - Formulação variável arco indexado no tempo

VPF - Formulação variável de posicionamento e fluxo

Tam Form - Tamanho da formulação

gap final - *gap* de otimalidade

O problema estudado neste trabalho foi pouco explorado na literatura, muitas vezes há estudo voltado para máquinas paralelas idênticas, outras vezes utiliza-se máquinas não relacionada, tempo de preparação dependentes mas com funções objetivas diferentes tais como o tempo máximo de conclusão das tarefas ou a soma dos atrasos ponderado, ver Seção 2.3. Das formulações apresentadas VOL foi adaptado de Unlu and Mason (2010), VAP inspirada em Rocha et al. (2008) e Unlu and Mason (2010), VR foi baseada em Pereira Lopes and de Carvalho (2007) e Tran and Beck (2012), VTC foi melhorada e inspirada do artigo de Rocha et al. (2008), VIT baseada em de Paula et al. (2010) e a formulação matemática VAIT do caso original Pessoa et al. (2010).

Com exceção das formulações matemáticas VR, VAIT e VPF todas as outras precisam respeitar a desigualde triângular $s^m_{jk} \leq s^m_{ji} + p^m_i + s^m_{ik}$. Caso contrário, essas formulações poderão não encontrar a solução ótima. Por exemplo considere três tarefas i, j e k em uma determinada máquina onde $p^m_j = 4$, $p^m_i = 7$, $p^m_k = 3$, $s^m_{ji} = 4$, $s^m_{ik} = 2$ e $s^m_{jk} = 20$. Ao avaliar a sequência ótima $j \rightarrow i \rightarrow k$ a formulação VTC (Subseção 2.2.1) por exemplo, abrirá as disjunções $j \rightarrow i$, $i \rightarrow k$ e $j \rightarrow k$, Figura 4.1. Daí no cálculo do tempo de conclusão da tarefa k, efetuado pela restrição (2.3), na sequência $j \rightarrow i \rightarrow k$, será $C_k \geq p^m_j + s^m_{ji} + p^m_i + s^m_{ik} + p^m_k = 20$ e ao mesmo tempo pela disjunção $j \rightarrow k$ será $C_k \geq p^m_j + s^m_{jk} + p_k = 27$ o que emplicará no tempo de conclusão da tarefa k ser igual 27.

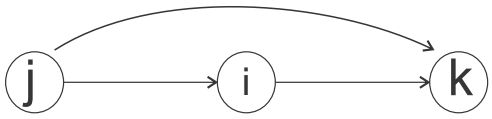

Figura 4.1: Desigualdade triângular

Para respeitar a desigualdade triângular as intâncias, após serem geradas aleatóriamente, foram modificadas pelo algoritmo 5 do artigo de Rocha et al. (2008).

As restrições de programação matemática pode ser classificada como ou conjuntivo ou disjuntivo. As restrições conjuntivos devem ser satisfeitas (obedecendo a condição

"e"), na disjuntiva apenas uma restrição dentro de um conjunto de restrições disjuntas deve ser satisfeitas (obedecendo a condição "ou"). Um exemplo de formulação matemática disjuntiva é a formulação VTC (Subseção 2.2.1), onde a disjunção acontece na restrição (2.3). Embora essas restrições são úteis em alguns contextos, sua desvantagem é o impacto sobre a relaxação do problema linear. Para resultar formulações de PIM válidas é necessário o uso de valores suficientemente grandes, $\mathcal{M}$. Neste trabalho utilizou-se $\mathcal{M} = \sum_{m \in M} \sum_{i \in N} \sum_{j \in N: i \neq j} (p_j^m + s_{ij}^m)$

A relaxação de problemas linear desempenha um papel muito importante na resolução de problemas de programação inteira mista. Estas são de interesse em algoritmos *branch and bound* que particiona todo o espaço de solução (*branching*) e desenvolve limites inferiores para partes do espaço de solução (*bounding*), Unlu and Mason (2010). Além de ser um dos melhores métodos para encontrar limites inferiores.

Nas Tabelas 4.2, 4.3, 4.4 e 4.5 observa-se que o *gap* da relaxação linear é sempre 100% nas formulações VAP e VR enquanto VTC e VOL possuem um valor elevado para o *gap* que diminui quando se aumenta o intervalo de variação do tempo de processamento. Em VIT apresenta um *gap* levemente menor que as anteriores e com comportamento semelhante com o aumento da variação do tempo de processamento. A formulação VPF expõe o segundo melhor *gap*, ele é pequeno e quando apresenta um valor alto, esse valor, é bem menor quando se compara com outros cinco modelos matemáticos. Seu *gap* de linearização somente é maior que o valor do VAIT que na maioria das vezes fecha em 0%.

Todos os resultados computacionais obtidos pelos testes são apresentados nas Tabelas 4.2 a 4.5. Nota-se que todos os modelos matemáticos apresentaram soluções ótimas para instâncias de até $n_j = 8$ e $n_m = 4$. Para instâncias maiores algumas já não consegue encontrar a solução ótima dentro do tempo limite especificado.

A formulação VAP não consegue resolver o problema dentro do tempo estipulado desde quando $n_j = 10$ e $n_m = 2$. Ela apresenta um *gap* linear de 100% em todos os casos. Apresenta também, um elevado número de nós $B\&B$ sendo na maioria das vezes o valor mais elevado entre as sete formulações comparadas.

VR mantêm o *gap* linear em 100% em todas as intâncias mas possui um tempo de resolução melhor que VAP. Em um determinado momento não encontra a solução ótima dentro do tempo limite mas ainda assim possui um *gap* final menor que a VAP além de apresentar menos nós $B\&B$ e menos cortes aplicado pelo CPLEX.

Os modelos VTC e VOL possuem na maioria das vezes o mesmo *gap* de linearização porém VTC resolve o problema em menor tempo e predominantemente tem menor número de nós $B\&B$ e cortes aplicados pelo CPLEX. Inclusive o *gap* final é menor que o de VOL quando a formulação atinge o tempo limite.

VIT é a terceira formulação com melhor *gap* de linearização, são poucas as vezes

que explora os nós $B\&B$ e quando o faz aplica poucos comparados a maioria das outras formulações. Essa formulação é sensível ao tamanho do tempo de processamento assim como a formulação VAIT. Essa formulação para instâncias onde $p_j^m \sim U[1, 100]$ não encontra solução inteira dentro tempo limite apartir de 15 tarefas.

A formulação VAIT é a que mais surpreende com seus resultados. Os *gap* são geralmente nulos ou quase nulos. O tempo computacional é menor que maior parte das formulações sendo maior que o de VPF, nas instâncias onde o número de tarefas ultrapassava 15 e que encontra solução dentro do tempo limite seu tempo de resolução é o menor que todas. Resolve o problema no nó raiz em quase todas as instâncias analisadas e poucos cortes são aplicados pelo CPLEX.

A nova formulação VPF demonstra resultados interessantes diante das demais já existentes, seu *gap* de linearização é o segundo menor, o tempo gasto para encontrar a solução ótima é muitas vezes o menor de todas as outras, porém por crescer muito, Tabela 4.1, em instâncias de vinte tarefas acaba atingindo o tempo limite na relaxação do problema. Ela explora poucos nós $B\&B$ e aplica poucos cortes pelo CPLEX.

O tamanho de uma formulação de PIM (requisito de memória) é de interesse uma vez que desempenha um papel significativo na definição do espaço necessário para armazenar e o tempo necessário para resolver a relaxação do problema linear e o problema de programação inteira mista. São utilizadas diferentes variáveis de decisões nas sete formulações abordadas. Com o interesse em avaliar e analisar o efeitos dessas variáveis no tamanho da formulação seja n_j, n_m e h o número de tarefas, o de número máquinas e o limite superior para o tempo de conclusão das tarefas, respectivamente.

A Tabela 4.1 nos mostra que as formulações VTC, VAP e VR crescem na ordem de $O(n_m n_j^2)$, VOL na ordem de $O(n_m n_j^3)$. Em VIT e VAIT possuem $O(n_m n_j^2 h)$, elas crescem muito dependendo do h por isso usou-se nesse trabalho o menor valor possível encontrado, para amenizar o impacto na formulação e fazer a comparação de forma justa. Usou-se h sendo $\sum_{i\in N}\sum_{j\in N:i\neq j}\left(\frac{max_{\{m\in M\}}(p_j^m+s_{ij}^m)}{n_m}\right) + p_{max}$ onde $p_{max} = max_{\left\{i\in N\cup\{0\}, j\in N, m\in M:i\neq j\right\}}(s_{ij}^m + p_j^m)$. A nova formulação VPF é a que mais cresce sendo da ordem de $O(n_m n_j^5)$.

Tabela 4.1: Tamanho do modelo para cada formulação.

Form	**Nº de restrições**	**Nº var. binárias**	**Nº var. contínuas**	**Tam for**
VTC	$n_j(n_j+1)(n_m+1)$	$n_m(n_j+1)^2$	n_j+1	$O(n_m n_j^2)$
VOL	$n_j(n_j+2n_m n_j-n_m)+n_j(n_j-1)(n_j-2)$	$n_j(2n_j+n_m-2)$	$n_m n_j$	$O(n_m n_j^2)$
VAP	$n_m n_j^3+3n_m n_j+n_j-n_m$	$n_m n_j^3$	$2n_m n_j$	$O(n_m n_j^3)$
VR	$n_j^2+2n_m n_j+n_j+n_m$	$n_m n_j(n_j+1)$	n_j+1	$O(n_m n_j^2)$
VIT	$n_j(2+n_m(n_j-1)(h+1))$	$n_m n_j(h+1)$	n_j	$O(n_m n_j^2 h)$
VAIT	$n_m+2n_j+n_m n_j(h+1)$	$n_m(h+1)(n_j^2+n_j+1)$	n_j	$O(n_m n_j^2 h)$
VPF	$3n_m n_j^4+n_m n_j^2+2n_m n_j+n_j-n_m$	$n_m n_j^2$	$n_m n_j^3(n_j+1+(n_j-1)^2)$	$O(n_m n_j^5)$

Tabela 4.2: Comparações entre formulações para $p_j^m \sim U[1,10]$ e $s_{ij}^m \sim U[1,25]$

$p_j^m \sim U[1,10]$ e $s_{ij}^m \sim U[1,25]$								
Instância n_j **e** n_m		**Formulações**						
		VTC	**VOL**	**VAP**	**VR**	**VIT**	**VAIT**	**VPF**
6 e 2	Tempo [s]	0	0	4.00	0	6.00	1.00	0.00
	B&B	119	864	2550	407	0	0	0
	Cortes CPLEX	53	58	92	52	0	0	0
	Gap Linear [%]	63.88	63.88	100.00	100.00	55.75	0	0
	Gap Final	0	0	0	0	0	0	0
6 e 4	Tempo [s]	0	0	4.00	0	8.00	1.00	0.00
	B&B	0	318	4530	171	0	0	0
	Cortes CPLEX	29	30	172	46	7	0	0
	Gap Linear [%]	35.16	35.16	100.00	100.00	16.48	0	0
	Gap Final	0	0	0	0	0	0	0
8 e 2	Tempo [s]	0	7.00	322.00	2.00	26.00	3.00	1.00
	B&B	835	12578	72046	4001	0	0	0
	Cortes CPLEX	121	164	238	24	48	0	6
	Gap Linear [%]	67.76	67.76	100.00	100.00	63.50	0	1.01
	Gap Final	0	0	0	0	0	0	0
8 e 4	Tempo [s]	0	11.00	804.00	1.00	28.00	14.00	2.00
	B&B	1405	32420	60954	681	0	0	0
	Cortes CPLEX	120	114	262	24	15	2	1
	Gap Linear [%]	49.21	49.21	100.00	100.00	45.47	1.96	2.60
	Gap Final	0	0	0	0	0	0	0
10 e 2	Tempo [s]	38.00	67.00	**	338.00	59.00	20.00	22.00
	B&B	44890	121625	313306	703820	0	0	67
	Cortes CPLEX	372	304	328	72	54	2	7
	Gap Linear [%]	77.57	77.57	100.00	100.00	73.52	1.71	18.96
	Gap Final	0	0	69.21	0	0	0	0
10 e 4	Tempo [s]	93.00	416.00	**	4.00	68.00	15.00	14.00
	B&B	83273	667078	92065	4345	0	0	19
	Cortes CPLEX	454	392	375	22	30	0	2
	Gap Linear [%]	65.25	65.25	100.00	100.00	59.12	0	2.79
	Gap Final	0	0	0	0	0	0	0
12 e 2	Tempo [s]	245.00	2810.00	**	**	148.00	23.00	45.00
	B&B	238951	2503934	250475	5567882	0	0	11
	Cortes CPLEX	336	588	382	144	77	0	1
	Gap Linear [%]	77.61	77.61	100.00	100.00	74.14	0	4.58
	Gap Final	0	0	82.11	8.97	0	0	0
12 e 4	Tempo [s]	833.00	3210.00	**	32.00	155.00	21.00	44.00
	B&B	470101	3208370	85100	29147	0	0	0
	Cortes CPLEX	766	847	624	21	83	0	0
	Gap Linear [%]	64.66	64.66	100.00	100.00	59.32	0	1.45
	Gap Final	0	0	90.37	0	0	0	0
15 e 2	Tempo [s]	**	**	**	**	484.00	148.00	**
	B&B	1148219	1435922	69847	3305264	0	0	104
	Cortes CPLEX	896	712	1831	159	83	0	0
	Gap Linear [%]	82.79	82.87	100.00	100.00	79.65	0.73	9.83
	Gap Final	32.72	39.23	98.39	49.83	0	0	2.79
15 e 4	Tempo [s]	**	**	**	**	438.00	125.00	563.00
	B&B	1027234	1409763	70747	2696884	16	0	25
	Cortes CPLEX	1070	893	1831	48	96	3	1
	Gap Linear [%]	68.3597	69.2058	100.00	100.00	63.94	3.71	5.35
	Gap Final	24.294	53.0548	98.3899	2.3666	0	0	0
20 e 2	Tempo [s]	**	**	**	**	2226.00	653.00	**
	B&B	246765	370599	21919	1845764	102	0	–
	Cortes CPLEX	1330	922	619	206	428	0	–
	Gap Linear [%]	86.01	86.99	100	100	83.36	0.19	–
	Gap Final	69.65	77.76	99.19	61.93	0	0	–
20 e 4	Tempo [s]	**	**	**	**	1826.00	651.00	**
	B&B	309258	678454	19578	1120328	0	0	–
	Cortes CPLEX	1308	758	2076	63	118	0	–
	Gap Linear [%]	76.56	81.38	100	100	71.83	0.19	–
	Gap Final	59.42	73.93	98.63	40.29	0	0	–

** superou tempo limite.

Tabela 4.3: Comparações entre formulações para $p_j^m \sim U[1,10]$ e $s_{ij}^m \sim U[1,50]$

$p_j^m \sim U[1,10]$ e $s_{ij}^m \sim U[1,50]$								
Instância n_j **e** n_m		**Formulações**						
		VTC	**VOL**	**VAP**	**VR**	**VIT**	**VAIT**	**VPF**
6 e 2	Tempo [s]	0	0	2.00	0	15.00	1.00	0.00
	B&B	98	1527	2979	312	0	0	0
	Cortes CPLEX	58	54	192	53	11	0	8
	Gap Linear [%]	73.39	73.39	100.00	100.00	67.39	0	2.78
	Gap Final	0	0	0	0	0	0	0
6 e 4	Tempo [s]	0	0	8.00	0	20.00	1.00	1.00
	B&B	0	447	2753	107	0	0	0
	Cortes CPLEX	25	25	195	27	11	0	0
	Gap Linear [%]	39.17	39.17	100.00	100.00	21.64	0	0
	Gap Final	0	0	0	0	0	0	0
8 e 2	Tempo [s]	1.00	6.00	320.00	1.00	60.00	24.00	2.00
	B&B	911	13263	76792	3560	0	0	5
	Cortes CPLEX	117	165	249	24	34	4	5
	Gap Linear [%]	75.28	75.28	100.00	100.00	72.01	1.33	4.51
	Gap Final	0	0	0	0	0	0	0
8 e 4	Tempo [s]	0	14.00	630.00	1.00	70.00	26.00	3.00
	B&B	1696	40246	61093	1015	0	0	0
	Cortes CPLEX	125	78	313	14	15	2	0
	Gap Linear [%]	55.67	55.67	100.00	100.00	52.40	2.92	3.95
	Gap Final	0	0	0	0	0	0	0
10 e 2	Tempo [s]	25.00	104.00	**	419.00	129.00	35.00	35.00
	B&B	37627	132193	267697	907704	11	0	163
	Cortes CPLEX	295	380	397	60	112	9	1
	Gap Linear [%]	83.53	83.53	100.00	100.00	80.55	4.52	27.54
	Gap Final	0	0	75.18	0	0	0	0
10 e 4	Tempo [s]	11.00	505.00	**	1.00	137.00	12.00	12.00
	B&B	8120	874160	111406	950	0	0	0
	Cortes CPLEX	319	434	542	15	36	0	4
	Gap Linear [%]	68.64	68.64	100.00	100.00	63.10	0	1.06
	Gap Final	0	0	82.53	0	0	0	0
12 e 2	Tempo [s]	734.00	769.00	**	2793.00	339.00	79.00	58.00
	B&B	757043	744994	176220	4679095	0	0	19
	Cortes CPLEX	516	528	438	119	69	2	0
	Gap Linear [%]	82.55	82.55	100.00	100.00	79.84	1.51	7.93
	Gap Final	0	0	83.98	0	0	0	0
12 e 4	Tempo [s]	2336.00	**	**	24.00	363.00	55.00	54.00
	B&B	1221586	3163856	46094	20007	0	0	0
	Cortes CPLEX	723	875	427	31	69	3	35
	Gap Linear [%]	70.68	70.68	100.00	100.00	66.25	1.80	3.51
	Gap Final	0	22.93	94.95	0	0	0	0
15 e 2	Tempo [s]	**	**	**	**	885.00	199.00	**
	B&B	1555457	1639645	51400	3531057	0	0	62
	Cortes CPLEX	885	682	440	163	110	0	0
	Gap Linear [%]	86.19	86.63	100.00	100.00	83.67	0.33	15.52
	Gap Final	20.19	35.29	98.19	51.34	0	0	0.10
15 e 4	Tempo [s]	**	**	**	1309.00	864.00	217.00	502.00
	B&B	885146	1226749	17137	984419	0	1	13
	Cortes CPLEX	1079	551	924	42	0	1	1
	Gap Linear [%]	72.64	72.64	100.00	100.00	68.92	3.04	5.59
	Gap Final	17.03	61.50	98.37	0	0	0	0
20 e 2	Tempo [s]	**	**	**	**	**	1663.00	**
	B&B	159667	381861	22273	1922404	224	0	–
	Cortes CPLEX	1405	832	706	151	472	0	–
	Gap Linear [%]	89.39	89.33	100	100	86.14	2.76	–
	Gap Final	72.26	82.22	99.35	65.23	2.30	0	–
20 e 4	Tempo [s]	**	**	**	**	**	1114.00	**
	B&B	278622	834299	18296	1070493	337	40	–
	Cortes CPLEX	1184	725	1674	61	247	10	–
	Gap Linear [%]	80.33	84.75	100	100	77.06	4.64	–
	Gap Final	63.15	78.67	99.36	45.09	0	0	–

** superou tempo limite.

Tabela 4.4: Comparações entre formulações para $p_j^m \sim U[1,100]$ e $s_{ij}^m \sim U[1,25]$

$p_j^m \sim U[1,100]$ e $s_{ij}^m \sim U[1,25]$								
Instância n_j **e** n_m		**Formulações**						
		VTC	**VOL**	**VADP**	**VR**	**VIT**	**VAIT**	**VPF**
6 e 2	Tempo [s]	0	0	4.00	0	93.00	3.00	1.00
	B&B	162	574	3263	919	0	0	0
	Cortes CPLEX	56	47	83	62	6	0	0
	Gap Linear [%]	43.04	43.04	100.00	100.00	41.70	0	1.71
	Gap Final	0	0	0	0	0	0	0
6 e 4	Tempo [s]	0	0	7.00	0	100.00	3.00	1.00
	B&B	0	22	2582	5	0	0	0
	Cortes CPLEX	6	10	199	42	12	0	0
	Gap Linear [%]	21.51	21.51	100.00	100.00	18.99	0	0
	Gap Final	0	0	0	0	0	0	0
8 e 2	Tempo [s]	0	2.00	134.00	3.00	501.00	25.00	2.00
	B&B	1602	4796	44680	7274	0	0	37
	Cortes CPLEX	109	77	234	22	47	0	1
	Gap Linear [%]	56.39	56.39	100.00	100.00	55.14	0	12.98
	Gap Final	0	0	0	0	0	0	0
8 e 4	Tempo [s]	0	0	150.00	1.00	353.00	12.00	2.00
	B&B	46	756	22829	528	0	0	0
	Cortes CPLEX	64	43	281	9	24	0	0
	Gap Linear [%]	32.61	32.61	100.00	100.00	30.62	0	1.73
	Gap Final	0	0	0	0	0	0	0
10 e 2	Tempo [s]	7.00	11.00	**	126.00	919.00	92.00	36.00
	B&B	9480	19187	393246	281053	0	0	209
	Cortes CPLEX	265	216	397	53	69	0	2
	Gap Linear [%]	57.87	57.87	100.00	100.00	56.68	0	19.31
	Gap Final	0	0	49.0780	0	0	0	0
10 e 4	Tempo [s]	3.00	4.00	**	1.00	885.00	37.00	10.00
	B&B	4216	4817	162081	669	0	0	0
	Cortes CPLEX	152	113	454	9	0	0	3
	Gap Linear [%]	42.57	42.57	100.00	100.00	40.16	0	2.91
	Gap Final	0	0	59.37	0	0	0	0
12 e 2	Tempo [s]	57.00	144.00	**	**	3468.00	360.00	298.00
	B&B	56750	112717	344582	5584365	0	0	348
	Cortes CPLEX	437	378	424	131	0	0	3
	Gap Linear [%]	60.78	60.78	100.00	100.00	59.60	0	23.57
	Gap Final	0	0	71.03	9.23	0	0	0
12 e 4	Tempo [s]	370.00	177.00	**	28.00	3574.00	121.00	41.00
	B&B	441049	210236	90774	29258	0	0	1
	Cortes CPLEX	436	294	599	23	59	0	0
	Gap Linear [%]	47.83	47.83	100.00	100.00	45.77	0	4.26
	Gap Final	0	0	84.9031	0	0	0	0
15 e 2	Tempo [s]	**	**	**	**	**	1750.00	**
	B&B	1570766	1682842	65459	3895959	–	0	13
	Cortes CPLEX	803	690	422	103	–	0	2
	Gap Linear [%]	69.63	69.63	100.00	100.00	–	0	30.83
	Gap Final	25.79	19.68	92.56	38.25	–	0	26.61
15 e 4	Tempo [s]	**	**	**	**	**	1032.00	**
	B&B	2032196	1565012	24774	2493017	–	0	425
	Cortes CPLEX	856	928	323	31	–	0	3
	Gap Linear [%]	51.61	51.61	100.00	100.00	–	0.03	13.54
	Gap Final	14.27	19.39	95.37	13.64	–	0	4.01
20 e 2	Tempo [s]	**	**	**	**	**	**	**
	B&B	776997	432211	1978719	21881	–	–	–
	Cortes CPLEX		871	94	629	–	–	–
	Gap Linear [%]	75.47	76.04	100	100	–	–	–
	Gap Final	53.73	61.47	54.06	97.07	–	–	–
20 e 4	Tempo [s]	**	**	**	**	**	**	**
	B&B	940044	936762	1155495	20277	–	0	–
	Cortes CPLEX	781	645	69	1260	–	0	–
	Gap Linear [%]	59.25	63.59	100	100	–	0	–
	Gap Final	37.67	44.03	30.74	100	–	0	–

** superou tempo limite.

Tabela 4.5: Comparações entre formulações para $p_j^m \sim U[1,100]$ e $s_{ij}^m \sim U[1,50]$

$p_j^m \sim U[1,100]$ e $s_{ij}^m \sim U[1,50]$								
Instância n_j **e** n_m		**Formulações**						
		VTC	**VOL**	**VADP**	**VR**	**VIT**	**VAIT**	**VPF**
6 e 2	Tempo [s]	0	0	1.00	0	137.00	4.00	0.00
	B&B	18	376	1281	81	0	0	0
	Cortes CPLEX	33	40	96	51	0	0	0
	Gap Linear [%]	43.71	43.71	100.00	100.00	41.73	0	0
	Gap Final	0	0	0	0	0	0	0
6 e 4	Tempo [s]	0	0	2.00	0	143.00	5.00	0
	B&B	0	13	1443	0	0	0	0
	Cortes CPLEX	4	13	141	9	24	0	0
	Gap Linear [%]	28.34	28.34	100.00	100.00	23.88	0	0
	Gap Final	0	0	0	0	0	0	00
8 e 2	Tempo [s]	1.00	3.00	196.00	2.00	986.00	33.00	2.00
	B&B	2282	6904	64343	5007	0	0	17
	Cortes CPLEX	141	102	240	21	0	0	3
	Gap Linear [%]	54.91	54.91	100.00	100.00	53.81	0	9.22
	Gap Final	0	0	0	0	0	0	0
8 e 4	Tempo [s]	0	1.00	243.00	0	477.00	16.00	2.00
	B&B	138	2535	39580	495	0	0	0
	Cortes CPLEX	68	42	329	7	0	0	0
	Gap Linear [%]	31.01	31.01	100.00	100.00	29.11	0	1.91
	Gap Final	0	0	0	0	0	0	0
10 e 2	Tempo [s]	10.00	48.00	**	150.00	1701.00	110.00	21.00
	B&B	14106	82019	318572	352308	0	0	84
	Cortes CPLEX	255	239	467	45	106	0	1
	Gap Linear [%]	61.61	61.61	100.00	100.00	60.52	0.01	17.34
	Gap Final	0	0	56.14	0	0	0	0
10 e 4	Tempo [s]	3.00	12.00	**	2.00	1227.00	46.00	11.00
	B&B	4525	20904	124044	550	0	0	0
	Cortes CPLEX	178	172	397	22	38	0	0
	Gap Linear [%]	48.91	48.91	100.00	100.00	46.77	0	0
	Gap Final	0	0	65.15	0	0	0	0
12 e 2	Tempo [s]	163.00	229.00	**	**	**	387.00	73.00
	B&B	154266	174854	157023	5140736	0	0	75
	Cortes CPLEX	448	405	363	142	41	0	2
	Gap Linear [%]	64.39	64.39	100.00	100.00	98.02	0	16.13
	Gap Final	0	0	75.21	04.44	96.60	0	0
12 e 4	Tempo [s]	779.00	842.00	**	71.00	3285.00	196.00	48.00
	B&B	845574	838968	90246	92150	0	0	3
	Cortes CPLEX	516	486	487	18	0	0	15
	Gap Linear [%]	54.41	54.41	100.00	100.00	52.60	0	3.17
	Gap Final	0	0	87.71	0	0	0	0
15 e 2	Tempo [s]	**	**	**	**	**	1783.00	**
	B&B	1758778	1498621	91178	3841248	–	0	19
	Cortes CPLEX	806	586	391	116	–	0	0
	Gap Linear [%]	73.03	73.41	100.00	100.00	–	0	25.19
	Gap Final	29.62	30.53	94.39	39.33	–	0	22.07
15 e 4	Tempo [s]	**	**	**	**	**	874.00	1182.00
	B&B	1956750	1365353	20563	2538357	–	0	103
	Cortes CPLEX	836	1056	3027	51	–	0	2
	Gap Linear [%]	54.9956	54.9956	100.00	100.00	–	0	10.91
	Gap Final	11.6785	18.2596	99.9187	13.4691	–	0	0
20 e 2	Tempo [s]	**	**	**	**	**	**	**
	B&B	361588	470337	24084	2022829	–	–	–
	Cortes CPLEX	1338	936	771	108	–	–	–
	Gap Linear [%]	77.97	78.46	100	100	–	–	–
	Gap Final	62.00	64.32	98.56	54.13	–	–	–
20 e 4	Tempo [s]	**	**	**	**	**	**	**
	B&B	816586	762928	10537	1137292	–	–	–
	Cortes CPLEX	1068	795	1552	71	–	–	–
	Gap Linear [%]	64.06	65.17	100	100	–	–	–
	Gap Final	36.67	44.24	100	32.66	–	–	–

** superou tempo limite.

4.2 Comparação entre os algoritmos

Nesta Seção é realizada a comparação entre a formulação original VPF utilizando o CPLEX 12.6 e os algoritmos variantes de Benders; *logic-based* Benders (algoritmo 1) e o Benders fortalecido (algoritmo 3). Para tal será utilizado o mesmo conjunto de instâncias da Seção 4.1 dentro de um tempo limite de 3600 s.

Os resultados encontram-se nas tabelas 4.6 - 4.9. Através delas observa-se que o CPLEX resolve o problema mais rápido para instâncias com menos de 15 tarefas porém como a formulação cresce muito, após 20 tarefas o CPLEX atinge o tempo limite na relaxação linear do problema.

Analisando os dois algoritmos variantes de Benders nota-se que o tempo computacional é menor para a maioria das instâncias onde $p_j^m \sim U[1, 10]$ no algoritmo do Benders fortalecido, enquanto para $p_j^m \sim U[1, 100]$ o tempo é menor na maioria das instâncias do algoritmo do *logic-based* Benders. É importante mencionar que o Benders fortalecio aplica um número muito pequeno de iterações e nas instâncias que alcança o tempo limite ele apresenta um *gap* inferior ao apresentado pelo *logic-based* Benders e pelo CPLEX.

Tabela 4.6: Comparação entre CPLEX, *Benders* fortalecido e *logic-basic Benders* com $p_j^m \sim U[1,10]$ e $s_{ij}^m \sim U[1,25]$.

Instâncias	CPLEX				Benders fortalecido					logic-basic Benders				
n_j e n_m	Tempo [s]	LI	LS	Gap [%]	Tempo [s]	#Iter.	LI	LS	Gap [%]	Tempo [s]	#Iter.	LI	LS	Gap [%]
6 e 2	0.00	252.00	252.00	0	1.00	4	252.00	252.00	0	1.00	27	252.00	252.00	0
6 e 4	0.00	91.00	91.00	0	1.00	2	91.00	91.00	0	**0.00**	13	91.00	91.00	0
8 e 2	1.00	540.50	546.00	1.01	**7.00**	4	546.0	546.0	0	9.00	145	546.00	546.00	0
8 e 4	2.00	247.40	254.00	2.60	23.00	7	254.00	254.00	0	**17.00**	83	254.00	254.00	0
10 e 2	22.00	802.29	990.00	18.96	346.00	28	990.00	990.0	0	**250.00**	672	990.00	990.00	0
10 e 4	14.00	414.13	426.00	2.79	**75.00**	7	426.00	426.00	0	724.00	426	426.00	426.00	0
12 e 2	45.00	1368.32	1434.00	4.58	**281.00**	14	1434.00	1434.00	0	*3597.00	1952	1201.00	1434.00	16.25
12 e 4	44.00	705.60	716.00	1.45	**334.00**	9	716.00	716.00	0	*3590.00	517	506.00	749.00	32.44
15 e 2	**	2331.88	2586.00	9.83	*3539.00	24	2527.00	2685.00	5.88	*3600.00	1999	610.00	2603.00	76.56
15 e 4	563.00	1133.00	1197.00	5.35	**2982.00**	15	1197.00	1197.00	0	*3599.00	912	535.00	1588.00	66.31
20 e 2	**	4388.57	-	-	*3197.00	6	5222.00	7833.00	33.33	*3594.00	302	966.00	6586.00	85.33
20 e 4	**	2457.10	-	-	*3337.00	3	2311.00	3955.00	41.57	*3586.00	250	860.00	3250.00	73.54

** Tempo limite excedido.

* Tempo da última iteração antes de exceder o tempo limite

Tabela 4.7: Comparação entre CPLEX, *Benders* fortalecido e *logic-basic Benders* com $p_j^m \sim U[1,10]$ e $s_{ij}^m \sim U[1,50]$.

Instâncias	CPLEX				Benders fortalecido					logic-basic Benders				
n_j e n_m	Tempo [s]	LI	LS	Gap [%]	Tempo [s]	#Iter.	LI	LS	Gap [%]	Tempo [s]	#Iter.	LI	LS	Gap [%]
6 e 2	0.00	332.50	342.00	2.78	1.00	4	342.00	342.00	0	1.00	28	342.00	342.00	0
6 e 4	1.00	97.00	97.00	0	**0.00**	2	97.00	97.00	0	1.00	16	97.00	97.00	0
8 e 2	2.00	679.86	712.00	4.51	**13.00**	8	712.00	712.00	0	14.00	156	712.00	712.00	0
8 e 4	3.00	279.50	291.00	3.95	22.00	7	291.00	291.00	0	**16.00**	83	291.00	291.00	0
10 e 2	35.00	976.71	1348.00	27.54	2363.00	48	1348.00	1348.00	0	**315.00**	714	1348.00	1348.00	0
10 e 4	12.00	467.00	472.00	1.06	**76.00**	7	472.00	472.00	0	469.00	303	472.00	472.00	0
12 e 2	58.00	1694.11	1840.00	7.93	**394.0**	18	1840.00	1840.00	0	*3598.00	1979	1540.00	1840.00	16.30
12 e 4	54.00	832.70	863.00	3.51	**332.00**	9	863.00	863.00	0	*3600.00	650	535.00	1103.00	51.45
15 e 2	**	2773.50	3283.00	15.52	*3153.00	18	3055.0	3640.0	16.07	*3597.00	2000	620.00	3291.00	81.16
15 e 4	502.00	1311.36	1389.00	5.59	**2556.00**	13	1389.00	1389.00	0	*3593.00	862	535.00	2090.00	74.40
20 e 2	**	5195.36	-	-	*3299.00	6	5940.00	12234.00	51.45	*3596.00	104	953.00	8168.00	88.33
20 e 4	**	2847.85	-	-	*3569.00	3	2619.00	6138.00	57.33	*3569.00	193	848.00	4690.00	81.92

** Tempo limite excedido.

* Tempo da última iteração antes de exceder o tempo limite

Tabela 4.8: Comparação entre CPLEX, *Benders* fortalecido e *logic-basic Benders* com $p_j^m \sim U[1, 100]$ e $s_{ij}^m \sim U[1, 25]$.

Instâncias	CPLEX				Benders fortalecido					logic-basic Benders				
n_j e n_m	Tempo [s]	LI	LS	Gap [%]	Tempo [s]	#Iter.	LI	LS	Gap [%]	Tempo [s]	#Iter.	LI	LS	Gap [%]
6 e 2	0.00	1454.75	1480.00	1.71	1.00	3	1480.00	1480.00	0	1.00	29	1480.00	1480.00	0
6 e 4	1.00	753.00	753.00	0	**0.00**	1	753.00	753.00	0	1.00	6	753.00	753.00	0
8 e 2	2.00	2504.50	2878.00	12.98	26.00	14	2878.00	2878.00	0	**5.00**	95	2878.00	2878.00	0
8 e 4	2.00	1331.50	1355.00	1.73	14.00	5	1355.00	1355.00	0	**1.00**	11	1355.00	1355.00	0
10 e 2	36.00	3725.45	4617.00	19.31	381.00	32	4617.00	4617.00	0	**76.00**	374	4617.00	4617.00	0
10 e 4	10.00	1915.5	1973.00	2.91	44.00	5	1973.00	1973.00	0	**11.00**	51	1973.00	1973.00	0
12 e 2	298.00	5227.15	6839.00	23.57	3507.00	39	6839.00	6839.00	0	**1652.00**	1311	6839.00	6839.00	0
12 e 4	41.00	3450.33	3604.00	4.26	**236.00**	8	3604.00	3604.00	0	792.00	232	3604.00	3604.00	0
15 e 2	**	8827.09	12761.00	30.83	*3447.00	22	10736.00	13359.00	19.63	*3596.00	1911	5218.00	12278.00	57.50
15 e 4	**	5473.89	6331.00	13.54	*3582.00	21	6183.00	6934.00	10.83	*3590.00	433	4538.00	6601.00	31.24
20 e 2	**	18864.43	-	-	*3315.00	6	22298.00	39899.00	0	*3522.00	35	8141.00	33656.00	75.81
20 e 4	**	10257.68	-	-	*2766.00	3	10109.00	13892.00	27.23	*3563.00	441	6841.00	14575.00	53.06

** Tempo limite excedido.

* Tempo da última iteração antes de exceder o tempo limite

Tabela 4.9: Comparação entre CPLEX, *Benders* fortalecido e *logic-basic Benders* com $p_j^m \sim U[1, 100]$ e $s_{ij}^m \sim U[1, 50]$.

Instâncias	CPLEX				Benders fortalecido					logic-basic Benders				
n_j e n_m	Tempo [s]	LI	LS	Gap [%]	Tempo [s]	#Iter.	LI	LS	Gap [%]	Tempo [s]	#Iter.	LI	LS	Gap [%]
6 e 2	0.00	1265.00	1265.00	0	1.00	3	1265.00	1265.00	0	**0.00**	16	1265.00	1265.00	0
6 e 4	0.00	494.00	494.00	0	0.00	1	494.00	494.00	0	0.00	5	494.00	494.00	0
8 e 2	2.00	3140.89	3460.00	9.22	23.00	13	3460.00	3460.00	0	**7.00**	130	3460.00	3460.00	0
8 e 4	2.00	1439.00	1467.00	1.91	12.00	4	1467.00	1467.00	0	**1.00**	16	1467.00	1467.00	0
10 e 2	21.00	4188.45	5067.00	17.34	335.00	31	5067.00	5067.00	0	**91.00**	418	5067.00	5067.00	0
10 e 4	11.00	2218.00	2218.00	0	37.00	4	2218.00	2218.00	0	**27.00**	78	2218.00	2218.00	0
12 e 2	73.00	6317.73	7533.00	16.13	**882.00**	29	7533.00	7533.00	0	2687.00	1643	7533.00	7533.00	0
12 e 4	48.00	3993.25	4124.00	3.17	**248.00**	8	4124.00	4124.00	0	*3584.00	389	4114	4234.00	2.83
15 e 2	**	11020.49	14731.00	25.19	*3341.00	22	12909.00	15583.00	17.16	*3596.00	1955	5234.00	14064.00	62.78
15 e 4	1182.00	6116.63	6866.00	10.91	**3208.00**	19	6866.00	6866.00	0	*3588.00	440	4643.00	7677.00	39.52
20 e 2	**	22728.70	-	-	*3540.00	7	27391.00	41988.00	34.76	*3561.00	69	8237.00	36857.00	77.65
20 e 4	**	11395.73	-	-	*2824.00	3	11106.00	18991.00	41.52	*3588.00	464	6744.00	16442.00	58.98

** Tempo limite excedido.

* Tempo da última iteração antes de exceder o tempo limite

Capítulo 5

Conclusão

"Problemas não são obstáculos, mas oportunidades ímpares de superação e evolução"

Nelson Rodrigues

O problema de sequenciamento de máquinas paralelas com tempo de preparação dependentes da sequência foi pesquisado e analisado neste trabalho. Buscou-se as formulações existentes na literatura que abordam problemas de sequenciamento, as seis encontradas foram adaptadas e algumas melhoradas para o problema abordado. Sentiu-se a necessidade de criar uma nova formulação que possibilitou uma abordagem exata. Devidos a suas características optou-se pela aplicação do metódo de decomposição de Benders, fortalecido com técnicas da literatura. Foi adaptado e comparado um algoritmo de decomposição *logic-basic* Benders por ser o único trabalho encontrado que utilizou o metódo de Benders para um problema parecido.

Os resultados mostram que a nova formulação VPF possui um *gap* linear mais apertado que cinco entre as seis formulações analisadas, seu tempo computacional só não é melhor que o da formulação variável arco indexado no tempo, Subseção 2.2.6. E quando atingido o tempo limite essa formulação demonstra um valor inferior para o *gap* final diante das formulações analisadas exceto para VAIT.

Os algoritmos variantes do método de decomposição de Benders denominados de algoritmos de Benders fortalecido e algoritmo de decomposição *logic-based* Benders mostram que o Benders fortalecido é mais atraente que o outro por resolver o problema em um número muito inferior de iterações e na maioria das vezes em menor tempo computacional, porém ambos não conseguem superar o tempo computacional do CPLEX. O Benders fortalecido ainda tem a vantagem de conseguir um *gap* mais apertado que o CPLEX e o *logic-based* Benders para instâncias maiores.

Acredita-se que pesquisas voltadas para técnicas de fortalecimento do método de decomposição de Benders em problemas desse tipo, devem ser incentivadas pois elas tem um grande impacto no tempo computacional. As técnicas utilizadas nesse trabalho

diminuíram de forma significativa o tempo computacional quando comparados com o algoritmo de decomposição de Benders original.

Para trabalhos futuros pretende-se pesquisar e aplicar uma heurística que obtenha melhores soluções iniciais, em virtude do impacto positivo no início do algoritmo abordado. Pretende-se melhorar a formulação, criar a formulação *leontief* e a formulação *Convex Hull* para o problema. Espera-se que essas formulações possibilite o trabalho com metódos exatos em grandes instâncias.

Referências Bibliográficas

Aardal, K. and Larsson, T. (1990). A benders decomposition based heuristic for the hierarchical production planning problem. *European Journal of Operational Research*, 45(1):4–14.

Allahverdi, A. (2015). The third comprehensive survey on scheduling problems with setup times/costs. *European Journal of Operational Research*.

Allahverdi, A., Gupta, J. N. D., and Aldowaisan, T. (1999). A Review of Scheduling Reserach Involving Setup Considerations. *Omega - The International Journal of Management Science*, 27(2):219–239.

Allahverdi, A., Ng, C., Cheng, T. E., and Kovalyov, M. Y. (2008). A survey of scheduling problems with setup times or costs. *European Journal of Operational Research*, 187(3):985–1032.

Arnaout, J.-P., Rabadi, G., and Musa, R. (2010). A two-stage ant colony optimization algorithm to minimize the makespan on unrelated parallel machines with sequence-dependent setup times. *Journal of Intelligent Manufacturing*, 21(6):693–701.

Avalos-Rosales, O., Alvarez, A. M., and Angel-Bello, F. (2013). A reformulation for the problem of scheduling unrelated parallel machines with sequence and machine dependent setup times. In *Twenty-Third International Conference on Automated Planning and Scheduling*.

Avalos-Rosales, O., Angel-Bello, F., and Alvarez, A. (2015). Efficient metaheuristic algorithm and re-formulations for the unrelated parallel machine scheduling problem with sequence and machine-dependent setup times. *The International Journal of Advanced Manufacturing Technology*, 76(9-12):1705–1718.

Balakrishnan, N., Kanet, J. J., and Sridharan, V. (1999). Early/tardy scheduling with sequence dependent setups on uniform parallel machines. *Computers and Operations Research*, 26(2):127–141.

Balas, E. (1985). *On the facial structure of scheduling polyhedra*. Springer.

Benders, J. F. (1962). Partitioning procedures for solving mixed-variables programming problems. *Numerische mathematik*, 4(1):238–252.

Blazewicz, J. (2001). *Scheduling computer and manufacturing processes*. Springer.

Blazewicz, J., Dror, M., and Weglarz, J. (1991). Mathematical programming formulations for machine scheduling: a survey. *European Journal of Operational Research*, 51(3):283–300.

Brucker, P. (2004). *Scheduling Algorithms*. Springer-Verlag.

Brucker, P., Dhaenens-Flipo, C., Knust, S., Kravchenko, S. A., and Werner, F. (2002). Complexity results for parallel machine problems with a single server. *Journal of Scheduling*, 5(6):429–457.

Bülbül, K. and Şen, H. (2015). An exact extended formulation for the unrelated parallel machine total weighted completion time problem. *Journal of Scheduling*.

Chen, Z.-L. and Lee, C.-Y. (2002). Parallel machine scheduling with a common due window. *European Journal of Operational Research*, 136(3):512–527.

Chen, Z.-L. and Powell, W. B. (1999). Solving parallel machine scheduling problems by column generation. *INFORMS Journal on Computing*, 11(1):78–94.

Chen, Z.-L. and Powell, W. B. (2003). Exact algorithms for scheduling multiple families of jobs on parallel machines. *Naval Research Logistics (NRL)*, 50(7):823–840.

Chou, F.-D., Wang, H.-M., and Chang, T.-Y. (2009). Algorithms for the single machine total weighted completion time scheduling problem with release times and sequence-dependent setups. *The International Journal of Advanced Manufacturing Technology*, 43(7-8):810–821.

Chudak, F. A. and Hochbaum, D. S. (1999). A half-integral linear programming relaxation for scheduling precedence-constrained jobs on a single machine. *Operations Research Letters*, 25(5):199–204.

Cochran, J. K., Horng, S.-M., and Fowler, J. W. (2003). A multi-population genetic algorithm to solve multi-objective scheduling problems for parallel machines. *Computers & Operations Research*, 30(7):1087–1102.

Contreras, I., Cordeau, J.-F., and Laporte, G. (2011). Benders decomposition for large-scale uncapacitated hub location. *Operations research*, 59(6):1477–1490.

Cordeau, J.-F., Soumis, F., and Desrosiers, J. (2000). A benders decomposition approach for the locomotive and car assignment problem. *Transportation science*, 34(2):133–149.

Costa, A. M. (2005). A survey on benders decomposition applied to fixed-charge network design problems. *Computers & Operations Research*, 32(6):1429–1450.

de Camargo, R. S., de Miranda Jr, G., and Luna, H. P. L. (2009a). Benders decomposition for hub location problems with economies of scale. *Transportation Science*, 43(1):86–97.

de Camargo, R. S., Miranda Jr, G., Ferreira, R. P. M., and Luna, H. (2009b). Multiple allocation hub-and-spoke network design under hub congestion. *Computers & Operations Research*, 36(12):3097–3106.

de Camargo, R. S., Miranda Jr, G., and Luna, H. (2008). Benders decomposition for the uncapacitated multiple allocation hub location problem. *Computers & Operations Research*, 35(4):1047–1064.

de Paula, M. R., Mateus, G. R., and Ravetti, M. G. (2010). A non-delayed relax-and-cut algorithm for scheduling problems with parallel machines, due dates and sequence-dependent setup times. *Computers & Operations Research*, 37(5):938–949.

de Sá, E. M., de Camargo, R. S., and de Miranda, G. (2013). An improved benders decomposition algorithm for the tree of hubs location problem. *European Journal of Operational Research*, 226(2):185–202.

Dogan, K. and Goetschalckx, M. (1999). A primal decomposition method for the integrated design of multi-period production–distribution systems. *Iie Transactions*, 31(11):1027–1036.

Dyer, M. E. and Wolsey, L. A. (1990). Formulating the single machine sequencing problem with release dates as a mixed integer program. *Discrete Applied Mathematics*, 26(2):255–270.

Fortz, B. and Poss, M. (2009). An improved benders decomposition applied to a multi-layer network design problem. *Operations research letters*, 37(5):359–364.

Gelareh, S. (2008). *Hub location models in public transport planning*. VDM Publishing.

Geoffrion, A. M. and Graves, G. W. (1974). Multicommodity distribution system design by benders decomposition. *Management science*, 20(5):822–844.

Gomory, R. E. (1969). Some polyhedra related to combinatorial problems. *Linear algebra and its applications*, 2(4):451–558.

Graham, R. L., Lawler, E. L., Lenstra, J. K., and Kan, A. (1979). Optimization and approximation in deterministic sequencing and scheduling: a survey. *Annals of discrete mathematics*, 5:287–326.

Hooker, J. N. (1994). Logic-based methods for optimization. In *Principles and Practice of Constraint Programming*, pages 336–349. Springer.

Hooker, J. N. and Yan, H. (1995). Logic circuit verification by benders decomposition. *Principles and practice of constraint programming: the newport papers*, pages 267–288.

Jackson, J. R. (1955). Scheduling a production line to minimize maximum tardiness. Technical report, DTIC Document.

Jackson, J. R. (1956). An extension of johnson's results on job idt scheduling. *Naval Research Logistics Quarterly*, 3(3):201–203.

Jain, V. and Grossmann, I. E. (2001). Algorithms for hybrid milp/cp models for a class of optimization problems. *INFORMS Journal on computing*, 13(4):258–276.

Johnson, S. M. (1954). Optimal two-and three-stage production schedules with setup times included. *Naval research logistics quarterly*, 1(1):61–68.

Kopanos, G. M., Laínez, J. M., and Puigjaner, L. (2009). An efficient mixed-integer linear programming scheduling framework for addressing sequence-dependent setup issues in batch plants. *Industrial & Engineering Chemistry Research*, 48(13):6346–6357.

Lasserre, J. B. and Queyranne, M. (1992). Generic Scheduling Polyhedra and a New Mixed-Integer Formulation for Single-Machine Scheduling. In *IPCO*, pages 136–149.

Lee, J.-H., Yu, J.-M., and Lee, D.-H. (2013). A tabu search algorithm for unrelated parallel machine scheduling with sequence-and machine-dependent setups: minimizing total tardiness. *The International Journal of Advanced Manufacturing Technology*, 69(9-12):2081–2089.

Lenstra, J. K., Rinnooy Kan, A., and Brucker, P. (1977). Complexity of machine scheduling problems. *Annals of discrete mathematics*, 1:343–362.

Li, K. and Yang, S.-l. (2009). Non-identical parallel-machine scheduling research with minimizing total weighted completion times: Models, relaxations and algorithms. *Applied mathematical modelling*, 33(4):2145–2158.

Lin, Y.-K. and Hsieh, F.-Y. (2014). Unrelated parallel machine scheduling with setup times and ready times. *International Journal of Production Research*, 52(4):1200–1214.

Magnanti, T. L. and Wong, R. T. (1981). Accelerating benders decomposition: Algorithmic enhancement and model selection criteria. *Operations research*, 29(3):464–484.

Manne, A. S. (1960). On the job-shop scheduling problem. *Operations Research*, 8(2):219–223.

Martins de Sá, E., Contreras, I., Cordeau, J.-F., Saraiva de Camargo, R., and de Miranda, G. (2015). The hub line location problem. *Transportation Science*.

Mercier, A. (2008). A theoretical comparison of feasibility cuts for the integrated aircraft-routing and crew-pairing problem. *Transportation Science*, 42(1):87–104.

Mercier, A., Cordeau, J.-F., and Soumis, F. (2005). A computational study of benders decomposition for the integrated aircraft routing and crew scheduling problem. *Computers & Operations Research*, 32(6):1451–1476.

Miller, C. E., Tucker, A. W., and Zemlin, R. A. (1960). Integer programming formulation of traveling salesman problems. *Journal of the ACM (JACM)*, 7(4):326–329.

Nemhauser, G. and Savelsbergh, M. (1992). *A cutting plane algorithm for the single machine scheduling problem with release times*. Springer.

Niu, Q., Zhou, F., and Zhou, T. J. (2011). An adaptive clonal selection algorithm with stage mutation operation for unrelated parallel machine scheduling problem with sequence-dependent setup times. In *Key Engineering Materials*, volume 467, pages 1967–1972. Trans Tech Publ.

Nogueira, T. H., de Carvalho, C., and Ravetti, M. G. (2014). Analysis of mixed integer programming formulations for single machine scheduling problems with sequence dependent setup times and release dates. *Submitted*.

Padberg, M. W. (1973). On the facial structure of set packing polyhedra. *Mathematical programming*, 5(1):199–215.

Papadakos, N. (2008). Practical enhancements to the magnanti–wong method. *Operations Research Letters*, 36(4):444–449.

Papadakos, N. (2009). Integrated airline scheduling. *Computers & Operations Research*, 36(1):176–195.

Pereira Lopes, M. J. and de Carvalho, J. (2007). A branch-and-price algorithm for scheduling parallel machines with sequence dependent setup times. *European journal of operational research*, 176(3):1508–1527.

Pessoa, A., Uchoa, E., de Aragão, M. P., and Rodrigues, R. (2010). Exact Algorithm Over an Arc-Time-Indexed Formulation for Parallel Machine Problems. *Mathematical Programming Computation*, 2(3-4):259–290.

Pinedo, M. L. (2008). *Scheduling: Theory, Algorithms, and Systems*. Springer.

Potts, C. (1980). An algorithm for the single machine sequencing problem with precedence constraints. In *Combinatorial Optimization II*, pages 78–87. Springer.

Queyranne, M. and Schulz, A. S. (1994). *Polyhedral approaches to machine scheduling*. Citeseer.

Rocha, P. L., Ravetti, M. G., Mateus, G. R., and Pardalos, P. M. (2008). Exact algorithms for a scheduling problem with unrelated parallel machines and sequence and machine-dependent setup times. *Computers & Operations Research*, 35(4):1250–1264.

Schulz, A. S. et al. (1996). *Polytopes and scheduling*. PhD thesis, Citeseer.

Şen, H. and Bülbül, K. (2015). A strong preemptive relaxation for weighted tardiness and earliness/tardiness problems on unrelated parallel machines. *INFORMS Journal on Computing*, 27(1):135–150.

Smith, W. E. (1956). Various optimizers for single-stage production. *Naval Research Logistics Quarterly*, 3(1-2):59–66.

Sousa, J. P. and Wolsey, L. A. (1992). A time indexed formulation of non-preemptive single machine scheduling problems. *Mathematical programming*, 54(1-3):353–367.

Tang, L., Jiang, W., and Saharidis, G. K. (2013). An improved benders decomposition algorithm for the logistics facility location problem with capacity expansions. *Annals of Operations Research*, 210(1):165–190.

Tran, T. T. and Beck, J. C. (2012). Logic-based benders decomposition for alternative resource scheduling with sequence dependent setups. In *ECAI*, pages 774–779.

Unlu, Y. and Mason, S. J. (2010). Evaluation of mixed integer programming formulations for non-preemptive parallel machine scheduling problems. *Computers & Industrial Engineering*, 58(4):785–800.

Üster, H. and Agrahari, H. (2011). A benders decomposition approach for a distribution network design problem with consolidation and capacity considerations. *Operations Research Letters*, 39(2):138–143.

Üster, H., Easwaran, G., Akçali, E., and Cetinkaya, S. (2007). Benders decomposition with alternative multiple cuts for a multi-product closed-loop supply chain network design model. *Naval Research Logistics (NRL)*, 54(8):890–907.

Vallada, E. and Ruiz, R. (2011). A genetic algorithm for the unrelated parallel machine scheduling problem with sequence dependent setup times. *European Journal of Operational Research*, 211(3):612–622.

van den Akker, J. M., Hoogeveen, J. A., and van de Velde, S. L. (1999). Parallel machine scheduling by column generation. *Operations Research*, 47(6):862–872.

Van Roy, T. J. (1986). A cross decomposition algorithm for capacitated facility location. *Operations Research*, 34(1):145–163.

Wagner, H. M. (1959). An integer linear-programming model for machine scheduling. *Naval Research Logistics Quarterly*, 6(2):131–140.

Weng, M. X., Lu, J., and Ren, H. (2001). Unrelated parallel machine scheduling with setup consideration and a total weighted completion time objective. *International Journal of Production Economics*, 70(3):215–226.

Wolsey, L. A. (1976). Technical note—facets and strong valid inequalities for integer programs. *Operations research*, 24(2):367–372.

Zhu, Z. and Heady, R. B. (2000). Minimizing the sum of earliness/tardiness in multi-machine scheduling: a mixed integer programming approach. *Computers & Industrial Engineering*, 38(2):297–305.