Um estudo sobre formulações matemáticas para um problema clássico de escalonamento com antecipação e atraso em máquinas paralelas

Rainer Xavier de Amorim, Rosiane de Freitas
Programa de Pós-Graduação em Informática (PPGI)
Instituto de Computação - Universidade Federal do Amazonas (IComp - UFAM)
Av. Rodrigo Otávio - Campus Universitário, nº 6.200, CEP 69.077-000, Manaus, AM
{rainer,rosiane}@icomp.ufam.edu.br

RESUMO

Este trabalho aborda formulações para um problema clássico de escalonamento com antecipação e atraso ponderados em máquinas paralelas idênticas, com tarefas de tempo de processamento arbitrários. Uma análise sobre as principais formulações matemáticas em programação inteira e mista é apresentada, bem como adaptações destas formulações para problemas de escalonamento com antecipação e atraso em máquinas paralelas idênticas.
PALAVRAS CHAVE: escalonamento com antecipação e atraso, formulações matemáticas, teoria de escalonamento.
Área principal: Otimização Combinatória.

ABSTRACT

This article addresses formulations for a classic weighted earliness-tardiness scheduling problem on identical parallel machines, with jobs containing arbitrary processing times. An analysis of the main integer programming and mixed mathematical formulations is presented as well as adaptations of these formulations for earliness-tardiness scheduling problems on identical parallel machines.
KEYWORDS: earliness and tardiness scheduling, mathematical formulations, scheduling theory.
Main area: Combinatorial Optimization.

1. Introdução

O problemas de escalonamento que envolvem antecipação e atraso (earliness-tardiness, E/T) possuem aplicações em diversos problemas do cotidiano. Uma aplicação bastante conhecida ocorre nas indústrias de processos produtivos, utilizando a ideia de produção sem folga (término da produção nem antes e nem depois da data de término sugerida), ou seja, a fabricação deve ser concluída a tempo da entrega ao cliente (do inglês, *Just-in-Time* (JIT)). Tal termo surgiu no Japão na década de 1970, na indústria automotiva, onde se buscava um sistema que atendesse a uma demanda específica com o menor tempo de atraso possível. Esse sistema tem como objetivo a melhoria do processo produtivo e fluxo de manufatura, eliminação de estoques e desperdícios [Leung (2004)].

De acordo com Xiao e Li (2002), decisões na determinação de datas de término sugeridas de tarefas são importantes no planejamento e controle de muitas operações de

uma organização. Uma cotação rápida de entrega de um determinado trabalho pode ser muito atrativa para o cliente, mas pode aumentar a chance de entregas atrasadas. Por outro lado, a demora na cotação de um trabalho pode diminuir a chance de o mesmo ser entregue com atraso, mas será penalizado com o risco de se perder um determinado negócio em potencial.

O conceito JIT também pode ser utilizado para gerir as atividades externas de uma empresa, como o processo de compra de insumos de outras fábricas, fabricação e montagem de novos produtos a partir desses insumos e distribuição ou exportação dos itens produzidos. JIT auxilia em todo esse processo, pois parte da ideia de 'conduzir' a produção a partir de uma determinada demanda, ou seja, a fábrica produz somente o necessário e nas quantidades necessárias quando requerido, evitando assim a quebra da cadeia de suprimentos (*supply chain*) da empresa e o acúmulo de itens produzidos no estoque, pois esses produtos são logo enviados ao mercado consumidor [Józefowska (2007)].

O problema pode ser formalizado como segue. Seja $J = \{1,...,n\}$ um conjunto de tarefas que podem ser processadas em um ambiente monoprocessado ou em um conjunto de máquinas paralelas idênticas $M = \{1,...,m\}$, sem preempção. Cada máquina pode processar pelo menos uma tarefa, sendo uma por vez, e todas as tarefas devem ser processadas. Cada tarefa j possui um tempo de processamento positivo (*processing time*) p_j, um prazo de término sugerido (*due date*) d_j e pesos positivos (*weight*) α_j e β_j. A antecipação de uma tarefa j é definida como $E_j = \max\{0, d_j - C_j\}$, enquanto o atraso de uma tarefa j é definido como $T_j = \max\{0, C_j - d_j\}$, onde C_j é o tempo de completude da tarefa j, que é obtido através da soma do instante de início da execução da tarefa mais o tempo de processamento da mesma [Brucker (2006)].

Na literatura existe uma ampla e crescente quantidade de trabalhos que consideram problemas de escalonamento com antecipação e atraso. Dentre eles destacam-se as revisões da literatura apresentadas por Baker e Scudder (1990), Gordon et al. (2002) e Shabtay e Steiner (2012). Dentre os trabalhos encontrados na literatura, apresentamos uma análise das formulações matemáticas juntamente com uma adaptação destas para o escalonamento E/T, considerando tarefas independentes, pesos e tempos de processamento arbitrários, sem preempção e nem tempo ocioso (ou seja, é permitido apenas no término de uma sequência de tarefas em cada máquina).

Um exemplo para o escalonamento com penalidades de antecipação e atraso em ambiente de máquinas paralelas idênticas é apresentado na Figura 1 onde, em (a), é apresentado um exemplo de instância com 5 tarefas para o problema com seus respectivos valores para tempos de processamento (p_j), data de término sugerida (d_j), penalidades de antecipação (α_j) e atraso (β_j). Já em (b) tem-se a solução ótima para a instância dada em duas máquinas paralelas idênticas.

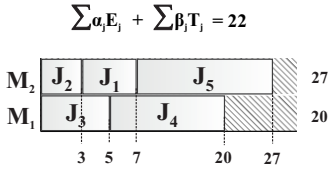
Este artigo é organizado como a seguir. A seção 2 apresenta os trabalhos relacionados com formulações de programação inteira e mista em máquinas paralelas, também é apresentada uma adaptação destas formulações para a função de escalonamento E/T nas subseções 2.2, 2.3, 2.4, 2.5 e 2.6. A seção 3 conclui o artigo.

2. Formulações matemáticas

Nesta seção serão detalhadas as principais formulações matemáticas existentes na literatura para o problema de escalonamento clássico com penalidades de antecipação e

J_j	p_j	d_j	α_j	β_j
J_1	4	7	3	5
J_2	3	4	4	5
J_3	5	5	8	8
J_4	15	17	6	4
J_5	20	25	7	3

(a)



(b)

Figura 1. (a) Exemplo de uma instância de 5 tarefas para o problema de escalonamento com antecipação e atraso (b) solução ótima.

atraso (tarefas independentes, com tempos de processamento arbitrários, sem preempção permitida, com datas de chegada iguais e datas de término distintas). Sendo assim, duas questões ajudam na classificação das formulações matemáticas existentes:

1. Permitem tempo ocioso no escalonamento ou não.
2. Consideram escalonamento em máquinas paralelas ou somente em uma única máquina.

Kanet e Sridharan (2000) apresentaram uma revisão da literatura sobre problemas de escalonamento com tempos ociosos inseridos, onde são considerados problemas de escalonamento envolvendo penalidades de antecipação e atraso. Os autores resumiram os resultados relacionados ao uso ou não de tempo ocioso em problemas gerais de escalonamento. Assim, afirmam que não é necessário considerar ou tratar tempo ocioso nos dois seguintes casos:

1. Problemas de escalonamento em ambiente monoprocessado.
2. Problemas de escalonamento em máquinas paralelas, quando todas as tarefas do problema estão simultaneamente disponíveis (quando se tem datas de chegadas iguais) e quando o problema de escalonamento apresentar medidas regulares de performance.

2.1. Tempo ocioso (TO) × não-ocioso (STO)

Quando se trabalha com tarefas com datas de término distintas, pode ser vantajoso segurar a produção por algum tempo a fim de reduzir os custos decorrentes da antecipação de tarefas. Assim, é possível haver tempo ocioso no escalonamento [Józefowska (2007)].

A inserção de tempo ocioso no escalonamento resulta em uma subutilização da máquina. A maximização da utilização da máquina geralmente entra em conflito com o valor de função objetivo JIT, o que pode impactar diretamente nos custos de estoque de uma empresa. Apesar do custo decorrente da existência de tempo ocioso na máquina não ser desprezível, ele é compatível com o conceito JIT quando se quer obter o menor valor de antecipação e atraso possível. Mas se o custo do tempo ocioso na máquina for elevado, pode ser necessária a aplicação de uma restrição adicional a fim de evitar tempo ocioso no escalonamento. Assim, tem-se dois grupos principais de problemas de escalonamento: problemas com tempo ocioso permitido e problemas sem tempo ocioso. O primeiro caso é mais consistente com a ideia do escalonamento JIT, enquanto que a versão sem tempo ocioso pode ter seu uso justificado em outras aplicações.

Dependendo do problema de antecipação e atraso considerado, existem problemas de escalonamento que não permitem tempo ocioso entre as tarefas devido as restrições do problema, como as linhas de produção de uma fábrica, por exemplo, que não podem

parar e devem seguir um fluxo contínuo de produção. Outro exemplo é quando se tem máquinas muito caras, que precisam ser utilizadas em uma determinada aplicação, logo, a sua utilização deve ser maximizada. Dessa forma, a restrição que evita tempo ocioso, apesar de não ser muito compatível com o conceito JIT, é também considerada na literatura.

Por outro lado, há problemas que permitem tempo ocioso no escalonamento, também devido às restrições do problema, ou seja, existem problemas que requerem que as tarefas terminem mais próximas de suas respectivas datas de término sugeridas, de forma a atender a um determinado cronograma de produção ou atender às necessidades de um cliente, por exemplo. Dessa forma, com um número razoável de tarefas terminando na data de término sugerida, os custos das penalidades de antecipação e atraso tendem a ser reduzidos. Assim, pode ser mais lucrativo considerar tempo ocioso em alguns problemas.

Os problemas de escalonamento com e sem tempo ocioso podem ser encontrados na literatura com as seguintes variações de datas de término sugeridas:

- Escalonamento com tempo ocioso:
  a) Pesos arbitrários (*arbitrary weights*): $\alpha \neq \beta$.
  b) Pesos proporcionais (*proportional weights*): $\alpha_i = \alpha p_i$ e $\beta_i = \beta p_i$, $i = 1, ..., n$, $\alpha, \beta \geq 0$.
- Escalonamento sem tempo ocioso:
  a) Pesos arbitrários (*arbitrary weights*): $\alpha \neq \beta$.
  b) Pesos independentes de tarefa (*job independent weights*): $\alpha_i = \alpha$ e $\beta_i = \beta$, $i = 1, ..., n$.

2.2. Formulação de programação linear inteira mista para o escalonamento em máquinas paralelas com tempo ocioso (PIM-TO)

A primeira formulação estudada é o modelo de programação linear inteira mista (PIM) proposto por Arenales et al. (2007) para o problema de escalonamento em máquinas paralelas. A formulação proposta assume que: todas as tarefas devem estar disponíveis no instante zero (datas de chegadas iguais, $r_j = 0$, sem perda de generalidade), cada tarefa possui a sua própria data de término sugerida (d_j distintos), qualquer máquina pode processar qualquer tarefa, cada tarefa pode ser executada somente uma vez, a preempção de uma tarefa não é permitida, cada máquina pode processar exatamente uma tarefa em um dado instante de tempo, o número de tarefas e máquinas são fixos e os tempos de processamento também são fixos, e apresenta tempos de preparação (*setup*) relacionadas a máquina e tarefas consecutivamente escalonadas.

A formulação utiliza quatro conjuntos de variáveis. O primeiro é composto de variáveis binárias tri-indexadas x_{ijk}, com valor 1 se a tarefa i precede imediatamente a tarefa j na máquina k, e 0 caso contrário. O segundo conjunto C_{ik}, de variáveis reais, consiste no instante de término de processamento de uma tarefa i na máquina k. Por fim, os conjuntos T_i e E_i, de variáveis também reais, representam, respectivamente, o atraso e a antecipação da tarefa i.

A formulação matemática também apresenta os seguintes parâmetros não-negativos:

- p_{ik} é o tempo de processamento da tarefa i na máquina k (para máquinas paralelas idênticas a notação muda para p_j).

- s_{ijk} é o tempo de preparação da máquina k para processar a tarefa j imediatamente após a tarefa i (caso o problema não tenha tempos de preparação, assume-se o valor zero para o tempo de preparação da máquina).
- d_j é a data de entrega da tarefa j.
- M é uma constante suficientemente grande (conhecida pelo termo em inglês, *big M*).

A seguir é apresentada uma adaptação da formulação de Arenales et al. (2007) envolvendo máquinas paralelas idênticas, com restrições e função objetivo lineares.

$$\text{Min} \sum_{j=1}^{n} \alpha_j E_j + \sum_{j=1}^{n} \beta_j T_j \tag{1}$$

$$\text{S. a.} \sum_{k=1}^{m} \sum_{i=0}^{n} x_{ijk} = 1 \qquad j = 1,2,...,n. \tag{2}$$

$$\sum_{j=1}^{n} x_{0jk} \leq 1 \qquad k = 1,2,...,m. \tag{3}$$

$$\sum_{i=0,\ i\neq h}^{n} x_{ihk} - \sum_{j=0,\ j\neq h}^{n} x_{hjk} = 0 \qquad h = 1,...,n \text{ e } k = 1,...,m. \tag{4}$$

$$C_{0k} = 0 \qquad k = 1,...,m. \tag{5}$$

$$C_{jk} \geq C_{ik} - M + (p_j + s_{ijk} + M)x_{ijk} \qquad i = 0,...,n; j = 1,...,n \text{ e } k = 1,...,m. \tag{6}$$

$$E_i \geq d_i - C_{ik} \qquad i = 1,...,n \text{ e } k = 1,...,m. \tag{7}$$

$$T_i \geq C_{ik} - d_i \qquad i = 1,...,n \text{ e } k = 1,...,m. \tag{8}$$

$$T_i \geq 0, E_i \geq 0 \qquad i = 1,...,n. \tag{9}$$

$$x_{ijk} \in \{0,1\} \qquad i,j = 0,...,n \text{ e } k = 1,...,m. \tag{10}$$

Na formulação acima, a função objetivo (1) minimiza a antecipação e o atraso das tarefas. Os conjuntos de restrições (2), (3), (4) asseguram uma sequência exclusiva de tarefas para cada uma das máquinas. Sendo que o conjunto de restrições (2) indica que cada tarefa j na máquina k tenha apenas uma tarefa precedente. O conjunto de restrições (3) garante que cada máquina k, se usada, tenha uma sequência exclusiva de tarefas. O conjunto de restrições (4) define que cada tarefa j tenha uma única tarefa imediata sucessora, com exceção da tarefa 0 que estabelece o início e o fim de uma sequência de tarefas na máquina k. Para a tarefa 0, o conjunto de restrições (5) estabelece que o instante de termino das tarefas nas máquinas devem ser iguais a zero.

No conjunto de restrições (6) são verificados os instantes de conclusão das tarefas nas máquinas onde são executadas. Nos conjuntos de restrições (7) e (8), são definidos, respectivamente, a antecipação e o atraso associado a cada uma das tarefas. Os conjuntos de restrições 9 e 10 indicam o domínio das variáveis utilizadas na formulação. Os conjuntos de restrições acima descritos não excluem a ocorrência de tempo ocioso entre as tarefas.

2.3. Formulação de programação linear inteira mista para o escalonamento em máquinas paralelas sem tempo ocioso (PIM-STO)

A segunda formulação estudada neste trabalho é o modelo de PIM proposto por Tsai e Wang (2012) para máquinas paralelas não-relacionadas ($R||\sum_{i=1}^{n} T_j + \sum_{i=1}^{n} E_j$) baseado na formulação apresentada por Rabadi et al. (2006) para a minimização do *makespan*

em máquinas paralelas não-relacionadas. A formulação proposta por Tsai e Wang (2012) também assume que: todas as tarefas devem estar disponíveis no instante zero (datas de chegadas iguais, $r_j = 0$, sem perda de generalidade), cada tarefa possui a sua própria data de término sugerida (d_j distintos), qualquer máquina pode processar qualquer tarefa, cada tarefa pode ser executada somente uma vez, a preempção de uma tarefa não é permitida, cada máquina pode processar exatamente uma tarefa em um dado instante de tempo, o número de tarefas e máquinas são fixos e os tempos de processamento também são fixos e apresenta tempos de preparação relacionadas a máquina e tarefas consecutivamente escalonadas.

As variáveis utilizadas nesta formulação são as mesmas da formulação anterior (com os casos especiais para: x_{0jk}, variável binária que terá valor 1 se a tarefa j é a primeira a ser processada na máquina k, 0 caso contrário, e x_{i0k}, variável binária que terá valor 1 se a tarefa j é a última a ser processada na máquina k, 0 caso contrário).

A seguir é apresentada uma adaptação da formulação matemática em programação inteira mista de Tsai e Wang (2012), envolvendo máquinas paralelas idênticas, com restrições e função objetivo lineares:

$$\text{Min} \sum_{j=1}^{n} \alpha_j E_j + \sum_{j=1}^{n} \beta_j T_j \tag{11}$$

$$\text{S. a.} \sum_{i=0,\ i\neq j}^{n} \sum_{k=1}^{m} x_{ijk} = 1 \qquad j = 1,2,...,n. \tag{12}$$

$$\sum_{i=0,\ i\neq h}^{n} x_{ihk} - \sum_{j=1,\ j\neq h}^{n} x_{hjk} = 0 \qquad h = 1,2,...,n;\ \ k = 1,2,...,m. \tag{13}$$

$$C_i + \sum_{k=1}^{m} x_{ijk}(p_j) + M(\sum_{k=1}^{m} x_{ijk} - 1) \leq C_j \qquad i = 0,1,...,n;\ \ j = 1,2,...,n. \tag{14}$$

$$\sum_{j=0}^{n} x_{0jk} = 1 \qquad k = 1,2,...,m. \tag{15}$$

$$T_j = C_j - d_j \qquad j = 1,2,...,n. \tag{16}$$

$$E_j = d_j - C_j \qquad j = 1,2,...,n. \tag{17}$$

$$C_0 = 0 \tag{18}$$

$$C_j, E_j, T_j \geq 0 \qquad j = 1,2,...,n. \tag{19}$$

$$x_{ijk} \in \{0,1\} \quad i,j = 1,2,...,n;\ \ k = 1,2,...,m. \tag{20}$$

$$C_0 + \sum_{k=1}^{m} x_{0jk}(p_j) \geq C_j + M(\sum_{k=1}^{m} x_{0jk} - 1) \qquad j = 1,2,...,n. \tag{21}$$

$$C_i + \sum_{k=1}^{m} x_{ijk}(p_j) \geq C_j + M(\sum_{k=1}^{m} x_{ijk} - 1) \qquad i = 0,1,...,n;\ \ j = 1,2,...,n. \tag{22}$$

Na formulação acima, a função objetivo (11) minimiza a antecipação e o atraso das tarefas com pesos distintos. O conjunto de restrições (12) assegura que uma tarefa seja escalonada somente uma vez e processada por somente uma máquina. O conjunto de restrições (13) assegura que cada tarefa i não seja nem precedida e nem sucedida por mais de uma tarefa j. O conjunto de restrições (14) é utilizado para calcular o tempo de completude das tarefas e assegurar que nenhuma tarefa i deve preceder e suceder ao mesmo tempo uma determinada tarefa j no escalonamento.

O conjunto de restrições (15) assegura que não mais que uma tarefa seja escalonada primeiro em uma máquina. O valores para o atraso e antecipação das tarefas são calculados nos conjuntos de restrições (16) e (17), respectivamente. O conjunto de restrições (18) define o tempo de completude da tarefa fictícia 0 seja zero. O conjunto de restrições (19) assegura que os valores para tempos de completude, antecipação e atraso sejam não-negativos. O conjunto de restrições (20) assegura que a variável de decisão x_{ijk} seja binária. Os conjuntos de restrições (21) e (22) asseguram que não haja tempo ocioso no escalonamento.

2.4. Formulação indexada pelo tempo para o escalonamento em ambiente monoprocessado (PI-STO-IT)

A terceira formulação estudada neste trabalho é um modelo de programação linear inteira (PI) proposto por Tanaka et al. (2009) para a minimização das penalidades de antecipação e atraso em ambiente monoprocessado (baseado nas formulações propostas por Pritsker et al. (1968), Dyer e Wolsey (1990), Sousa and Wolsey (1992) e Van den Akker et al. (1999)).

As variáveis de decisão são binárias bi-indexadas, x_{it} $(i \in N, 1 \leq t \leq T)$, tal que x_{it} é igual a 1 se $t \geq p_i$ e se a tarefa i é completada no instante t $(t = C_i)$, e é igual a 0, caso contrário. Além disso, possui os seguintes parâmetros não-negativos: p_j, indicando o tempo de processamento da tarefa j e T uma constante indicando o limite superior para o intervalo de tempo em que as tarefas devem ser escalonadas (intervalo de tempo entre 1 e T).

Desta forma, o problema pode ser formulado como segue:

$$\text{Min} \sum_{i=1}^{n} \sum_{t=1}^{T} f_i(t) x_{it} \tag{23}$$

$$\text{S. a.} \sum_{i=1}^{n} \sum_{s=t}^{min(T,\ t+p_i-1)} x_{is} = 1 \quad 1 \leq t \leq T \tag{24}$$

$$\sum_{t=1}^{T} x_{it} = 1 \quad i \in N \tag{25}$$

$$x_{it} \in \{0,1\}, \quad i \in N, \quad 1 \leq t \leq T \tag{26}$$

O conjunto de restrições (24) define a restrição de capacidade da máquina, onde cada máquina deve processar exatamente uma tarefa em um determinado instante de tempo. O conjunto de restrições (25) indica o número de ocorrências das tarefas no escalonamento. Para a função objetivo (23), minimização da antecipação e atraso, tem-se: $f_j(t) = \alpha_i \cdot max\{d_i - t, 0\} + \beta_i \cdot max\{t - d_i, 0\}$.

2.5. Formulação clássica geral indexada pelo tempo para o escalonamento em ambiente monoprocessado (PI-TO-IT)

A quarta formulação baseia-se na formulação geral proposta para problemas de escalonamento com funções objetivo regulares (não-decrescentes em relação aos tempos de completude das tarefas) proposta por Dyer e Wolsey (1990).

As variáveis de decisão são binárias bi-indexadas, y_j^t, indicando que uma tarefa j inicia o seu processamento no instante t em alguma máquina, com intervalo de tempo entre 0 e T.

Desta forma, o problema pode ser formulado como segue:

$$\text{Min} \sum_{j \in J} \sum_{t=0}^{T-p_j} f_j(t+p_j) y^t{}_j \tag{27}$$

$$\text{S. a.} \sum_{t=0}^{T-p_j} y^t{}_j = 1 \quad (\forall j \in J) \tag{28}$$

$$\sum_{\substack{j \in J, \\ t+p_j \leq T}} \sum_{s=max\{0,t-p_j+1\}}^{t} y^s{}_j \leq 1 \quad (t = 0, ..., T-1) \tag{29}$$

$$y^t{}_j \in \{0,1\} \quad (\forall j \in J; t = 0, ..., T-1) \tag{30}$$

Na função objetivo (27), a função $f_j(x)$ é definida para cada tarefa, onde x é o instante onde cada tarefa finaliza o seu processamento na linha do tempo. Se a função objetivo é para a minimização do atraso ponderado, por exemplo, $f_j(x)$ é igual a $\beta_j \times max\{0, x - d_j\}$. O conjunto de restrições (28) determina que a tarefa deve ser processada exatamente uma vez. O conjunto de restrições (29) determina que a máquina pode processar no máximo uma tarefa em um dado instante de tempo, sendo que a soma de tal instante com o seu respectivo tempo de processamento não pode ser maior que o máximo tempo de execução. Somente 1 tarefa pode ser executada ao mesmo tempo (na única máquina disponível). Esta formulação pode ser adaptada para ambiente multiprocessado, trocando-se para m o lado direito desta restrição, indicando que no pior caso, somente m tarefas podem ser executadas ao mesmo tempo, onde m é o número de máquinas disponíveis. O conjunto de restrições (30) define os tipos de variáveis utilizadas.

Amorim et al. (2013) propuseram uma formulação indexada pelo tempo baseada na formulação clássica geral de escalonamento e fluxo em redes. Onde foram adicionadas restrições para evitar tempo ocioso na minimização da antecipação e atraso ponderados (funções não-regulares).

As tarefas devem ser processadas no intervalo de tempo $[0, T]$, onde $T = \sum_{j=1}^{n} p_j$, onde m é o número de máquinas. As variáveis de decisão binárias y_j^t indicam que a tarefa j inicia o seu processamento no instante t em alguma máquina. Assim, o problema foi formulado como segue (Formulação indexada pelo tempo baseada no modelo de fluxo em redes para o escalonamento em máquinas paralelas - PI-STO-JIT):

$$\text{Min} \sum_{t=0}^{T} \sum_{j=1}^{n} f_j(C_j) y^t{}_j \tag{31}$$

$$\text{S. a.} \sum_{t=0}^{T} y^t{}_j = 1 \quad (j = 1, 2, ..., n) \tag{32}$$

$$\sum_{j=1}^{n} y_j^{t-p_j} - \sum_{j=0}^{n} y_j^t = 0 \quad (t = 1, ..., T) \tag{33}$$

$$\sum_{j=1}^{n} y_j^0 = m \tag{34}$$

$$y^t{}_j \in \{0,1\} \quad (j = 1, 2, ..., n; t = 0, ..., T) \tag{35}$$

A função objetivo $f_j(C_j)$ (31) desta formulação é baseada no problema $P||\sum \alpha_j E_j + \sum \beta_j T_j$. Assim, o valor de função objetivo pode ser calculado como segue: $f_j(C_j) =$

$\sum \alpha_j \cdot max\{0, d_j - C_j\} + \sum \beta_j \cdot max\{0, C_j - d_j\}$. O conjunto de restrições (32) determina que a tarefa deve ser processada exatamente uma vez. O conjunto de restrições (33) determina a representação do escalonamento através de fluxo em redes, o que garante que não se tenha tempo ocioso entre as tarefas no escalonamento, neste caso, o tempo ocioso é permitido somente quando todas as tarefas em cada máquina são processadas, o tempo ocioso é representado pela tarefa fictícia 0 (zero), que é utilizada no final da sequência de tarefas em cada máquina, como esta tarefa fictícia é a última da sequência de tarefas, ela é direcionada ao instante final do escalonamento, como pode ser observado na Figura 2 (b). O conjunto de restrições (34) define o instante de início do escalonamento, no início do escalonamento é assegurado que a primeira tarefa da sequência em cada máquina comece no instante 0 (zero).

A Figura 2 (b) apresenta o escalonamento utilizando a representação de fluxo em redes para a solução apresentada na Figura 2 (a), onde cada caminho é representado por diferentes setas na rede da Figura 2 (b), que indica uma sequência de tarefas em cada máquina.

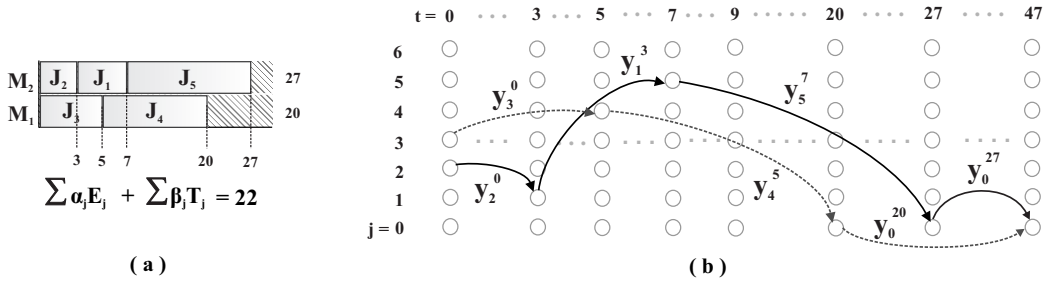


Figura 2. (a) Solução utilizando o gráfico de Gannt. (b) representação do escalonamento no modelo de fluxo em redes para a formulação clássica do problema.

Uma vantagem importante da formulação indexada pelo tempo é que esta formulação pode ser utilizada para modelar vários critérios de desempenho, do tipo regulares (funções objetivo regulares) para problemas de escalonamento. A mudança na formulação se dá através da mudança dos custos da função objetivo. A desvantagem na utilização deste modelo é no tamanho, onde existem $n+T$ restrições e aproximadamente nT variáveis, onde T é a soma de todos os tempos de processamento $\sum_{j=1}^{n} p_j$ (para ambiente monoprocessado). Desta forma, instâncias com muitas tarefas, ou tarefas com um tempo de processamento grande, demandam muito tempo para solução.

2.6. Formulação baseada no modelo de fluxo em redes e roteamento de veículos para o o escalonamento em máquinas paralelas - *Arc-time indexed formulation* (PI-STO-ArcTime)

A última formulação, proposta por Pessoa et al. (2010) para a minimização do atraso ponderado de tarefas em ambiente mono e multiprocessado, baseia-se no modelo de fluxo em redes e roteamento de veículos (*Arc-time indexed formulation*). Esta formulação considera o escalonamento no intervalo de tempo de 0 a T, onde as máquinas estão ociosas no instante 0 e devem estar novamente ociosas no instante T. As variáveis binárias x_{ij}^{t}, $i \neq j$, indicam que a tarefa i termina sua execução e a tarefa j inicia a sua execução no instante t, em alguma máquina. As variáveis x_{0j}^{t} indicam que a tarefa j inicia seu processamento no instante t em alguma máquina que estava ociosa no intervalo $t-1$ a t, em particular, as variáveis x_{0j}^{0} indicam que a tarefa j inicia em alguma máquina no instante 0. As variáveis

x^t_{i0} indicam que a tarefa i finaliza o seu processamento no instante t em uma máquina que ficará ociosa nos instantes t a $t+1$, as variáveis x^T_{i0} indicam que a tarefa i será finalizada no instante final do intervalo de tempo. As variáveis x^t_{00} indicam que o número de máquinas que estavam livres no intervalo $t-1$ a t e irá permanecer ociosa no intervalo t a $t+1$.

$$\text{Minimizar} \sum_{j\in J_+}\sum_{j\in J\setminus\{i\}}\sum_{t=p_i}^{T-p_j} f_j(t+p_j)x^t_{ij} \tag{36}$$

$$\text{Sujeito a} \sum_{i\in J_+\setminus\{j\}}\sum_{t=p_i}^{T-p_j} x^t_{ij} = 1 \quad (\forall j \in J) \tag{37}$$

$$\sum_{j\in J_+\setminus\{i\},\ t-p_j\geq 0} x^t_{ji} - \sum_{j\in J_+\setminus\{i\},\ t+p_i+p_j\leq T} x^{t+p_i}_{ij} = 0 \quad (\forall j \in J; \tag{38}$$
$$t = 0,...,T-p_i)$$

$$\sum_{j\in J_+,\ t-p_j\geq 0} x^t_{j0} - \sum_{j\in J_+,\ t+p_j+1\leq T} x^{t+1}_{0j} = 0 \quad (t = 0,...,T-1) \tag{39}$$

$$\sum_{i\in J_+} x^0_{0j} = m \tag{40}$$

$$x^t_{ij} \in Z_+ \quad (\forall i \in J_+; \forall j \in J_+\setminus\{j\}; t = p_i,...,T-p_j) \tag{41}$$

$$x^t_{00} \in Z_+ \quad (t = 0,...,T-1) \tag{42}$$

Os conjuntos de restrições (38), (39) e (40) definem o fluxo em redes de m unidades (ou m máquinas) em um grafo acíclico $G = (V,A)$. Neste fluxo existem apenas uma origem e um destino, qualquer solução pode ser decomposta em um conjunto de m caminhos correspondentes a cada escalonamento (sequência de tarefas com seus tempos ociosos) de cada máquina. O conjunto de restrições (37) define que cada tarefa deve estar em exatamente um caminho, e por isso, processada em somente uma máquina.

A Figura 3 apresenta o escalonamento utilizando a representação de fluxo em redes e roteamento de veículos, para a instância apresentada na Figura 1 (a), onde cada caminho na rede representa o escalonamento em uma máquina.

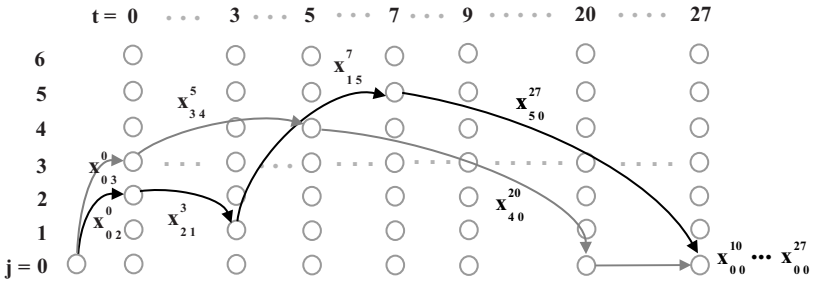


Figura 3. Representação do escalonamento no modelo de fluxo em redes.

Esta formulação foi adaptada para a minimização conjunta das penalidades de antecipação e atraso, sem tempo ocioso entre as tarefas, o tempo ocioso neste caso seria considerado somente após o fim da execução das tarefas nas máquinas.

Na Tabela 1 são resumidas as principais características das formulações estudadas. A Tabela 1 apresenta sete colunas: formulação matemática, número de restrições, número de variáveis, tipos de variáveis, ambiente de processamento, tipo de formulação e referência.

Tabela 1. Dimensão das formulações estudadas

Formulação matemática	Número de restrições	Número de variáveis	Tipos de variáveis	Ambiente de processamento	Tipo de formulação	Referência
Com tempo Ocioso						
PIM-TO	$O(n^2m)$	$O(n^2m)$	mono, bi e tri-indexadas	mono e multi-processado	PIM	Arenales et al. (2007)
PI-TO-IT	$O(n+T)$	$O(nT)$	bi-indexadas	monoprocessado	PI	Dyer e Wolsey (1990), Pessoa et al. (2010)
Sem Tempo Ocioso						
PIM-STO	$O(n^2)$	$O(n^2m)$	mono, bi e tri-indexadas	mono e multi-processado	PIM	Tsai e Wang (2012)
PI-STO-IT	$O(n+T)$	$O(nT)$	bi-indexadas	monoprocessado	PI	Tanaka et al. (2009)
PI-STO-JIT	$O(n+T)$	$O(nT)$	bi-indexadas	mono e multi-processado	PI	Amorim et al. (2013)
PI-STO-ArcTime	$O(nT)$	$O(n^2T)$	tri-indexadas	mono e multi-processado	PI	Pessoa et al. (2010)

3. Conclusões

Este trabalho apresentou um estudo das formulações matemáticas para problemas de escalonamento E/T, com tarefas independentes, de tempos de processamento arbitrários e pesos distintos. Também foram apresentadas adaptações das formulações estudadas para a função de escalonamento E/T, com e sem tratamento de tempo ocioso entre as tarefas. No caso do escalonamento sem tempo ocioso, só é permitido tempo ocioso no final de uma sequência de tarefas em cada máquina. Uma análise da dimensão das formulações nos permite afirmar que quanto maior for o tamanho da instância, ou tarefas com um grande tempo de processamento, maior será o tempo para a solução do problema. Assim, faz-se necessário considerar métodos mais eficientes de resolução de tais problemas.

Para trabalhos futuros envolvendo problemas de escalonamento com penalidades de antecipação e atraso, considera-se importante serem propostas formulações matemáticas em programação inteira mais robustas para tais problemas, de tal forma a serem desenvolvidos métodos exatos mais eficientes. Métodos híbridos, envolvendo a resolução de formulações relaxadas para o problema em conjunto com métodos heurísticos, é um caminho interessante a seguir, no intuito de se obter soluções ótimas com tempos de execução mais eficientes.

Agradecimentos

Agradecemos ao Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq), a Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES) e a Fundação de Amparo à Pesquisa do Estado do Amazonas (FAPEAM) pelo apoio financeiro.

Referências

Amorim, R. X. d., deFreitas, R., e Uchoa, E. (2013). A network flow IP formulation and exact/heuristics approaches for just-in-time scheduling problems on parallel machines without idle times. *Anais do XLV SBPO - Simpósio Brasileiro de Pesquisa Operacional*, 45:1835–1846.

Arenales, M., Morabito, R., Armentano, V. A., e Yanasse, H. (2007). *Pesquisa Operacional: As Disciplinas da Execução da Estratégia*. Elsevier.

Baker, K. R. e Scudder, G. D. (1990). Sequencing with earliness and tardiness penalties: a review. *Oper. Res.*, 38:22–36.

Brucker, P. (2006). Scheduling algorithms. *Springer Publishing Company, Incorporated*, 5a ed.:1–104.

Dyer, M. E. e Wolsey, L. A. (1990). Formulating the single machine sequencing problem with release dates as a mixed integer program. *Discrete Appl. Math.*, 26(2-3):255–270.

Gordon, V., Proth, J.-M., e Chu, C. (2002). A survey of the state-of-the-art of common due date assignment and scheduling research. *European Journal of Operational Research*, 139:1–25.

Józefowska, J. (2007). *Just-in-Time Scheduling: Models and Algorithms for Computer and Manufacturing Systems*. International Series in Operations Research & Management Science. Springer.

Kanet, J. J. e Sridharan, V. (2000). Scheduling with inserted idle time: problem taxonomy and literature review. *Operations Research*, 48:99–110.

Leung, J. Y.-T. e Anderson, J. H. (2004). *Handbook of scheduling: algorithms, models and performance analysis*. Chapman and Hall/CRC.

Pessoa, A., Uchoa, E., Poggi, M., e de Freitas, R. (2010). Exact algorithm over an arc-time-indexed formulation for parallel machine scheduling problems. *Mathematical Programming Computation*, 2:259–290.

Pritsker, A., Watters, L., e Wolfe, P. (1968). *Multiproject Scheduling with Limited Resources: A Zero-one Programming Approach*. Number Nº 3800 in P (Rand Corporation). Rand Corporation.

Rabadi, G., Moraga, R. J., e Al-Salem, A. (2006). Heuristics for the unrelated parallel machine scheduling problem with setup times. *Journal of Intelligent Manufacturing*, 17:85–97.

Shabtay, D. e Steiner, G. (2012). Scheduling to maximize the number of just-in-time jobs: A survey. *Springer Optimization and Its Applications*, 60:3–20.

Sousa, J. P. e Wolsey, L. A. (1992). A time indexed formulation of non-preemptive single machine scheduling problems. *Math. Program.*, 54:353–367.

Tanaka, S., Fujikuma, S., e Araki, M. (2009). An exact algorithm for single-machine scheduling without machine idle time. *J. of Scheduling*, 12(6):575–593.

Tsai, C.-Y. e Wang, Y.-C. (2012). The sum of earliness and tardiness minimization on unrelated parallel machines with inserted idle time. *Latest Advances in Information Science and Applications*, pages 125–130.

van den Akker, J. M., van Hoesel, C. P. M., e Savelsbergh, M. W. P. (1999). A polyhedral approach to single-machine scheduling problems. *Mathematical Programming*, 85(3):541.

Xiao, W.-Q. e Li, C.-L. (2002). Approximation algorithms for common due date assignment and job scheduling on parallel machines. *IIE Transactions*, 34:467–477.