PRO5826 - Estudo de Heurísticas e Meta-heurísticas para Problemas de Produção

Implementação I

Jhonatan Albertini

NUSP: 11887309

Luiza Pellin Biasoto

NUSP: 6846703

Docente: Debora Pretti Ronconi

18 de julho de 2021

1 Introdução

Este relatório visa descrever as premissas, métodos e o passo-a-passo da elaboração de uma heurística construtiva para otimização do problema de planejamento da produção em uma única máquina com datas de entrega em comum, considerando penalidades de adiantamento e de atraso de entrega. A modelagem do problema considerado NP-difícil tem como base o modelo descrito por Biskup e Feldmann[1]. Os testes serão executados em 280 problemas distintos: 7 conjuntos de 10 instâncias, com dimensões de 10, 20, 50, 100, 200, 500 e 1000 tarefas cada, sendo otimizados sob 4 diferentes valores para o parâmetro h, que determina o quão restritiva é a data de entrega do problema. Os resultados das otimizações utilizando a heurística e métodos descritos neste trabalho serão então comparados com os resultados obtidos por Biskup e Feldmann. Nas conclusões, serão apresentadas as suas vantagens e limitações, além de direcionadores para os próximos passos deste trabalho.

Tabela 1: Resumo das complexidades dos problemas para datas de entrega comuns restritivas e não restritivas.

para i = 1, ..., n:	**d é não restritiva**	**d é restritiva**
$\mathbf{a_i = b_i = 1}$	Solucionável em tempo polinomial	NP-difícil, solucionável em tempo pseudo-polinomial
$\mathbf{a_i = a}$ **e** $\mathbf{b_i = b}$	Solucionável em tempo polinomial	NP-difícil, solucionável em tempo pseudo-polinomial
$\mathbf{a_i = b_i}$ **(problema balanceado)**	NP-difícil, solucionável em tempo pseudo-polinomial	NP-difícil, solucionável em tempo pseudo-polinomial
$\mathbf{a_i}$**,** $\mathbf{b_i}$ **(problema generalizado)**	NP-difícil, aberto	NP-difícil, aberto

1.1 Organização do texto

O problema de planejamento da produção em uma única máquina com data de entrega em comum será definido na Seção 2. A elaboração da heurística e suas premissas serão descritas na Seção 3. Resultados computacionais comparados com os do artigo do Biskup serão apresentados na Seção 4 e as conclusões mostradas na Seção 5.

2 Descrição do problema

O problema de planejamento da produção em uma única máquina com datas de entrega em comum considera diferentes penalidades para adiantamento e atraso das tarefas executadas. O problema possui n tarefas que precisam ser processadas uma única vez e por uma única máquina. As i = 1, ..., n tarefas possuem tempos de processamento determinísticos e conhecidos, p_i, e cada uma possui penalidades por unidade de tempo para adiantamento a_i e atraso b_i distintos. A data de entrega de todas as tarefas é então definido como uma única data d em comum. Caso a data de término C_i de uma tarefa seja menor do que a data de entrega comum d, o adiantamento E_i é definido por: $E_i = d - C_i$. Por outro lado, caso a data de término de uma tarefa seja maior que d, o atraso desta é definido por: $T_i = C_i - d$.

Em problemas de planejamento com data de entrega em comum, d é uma informação determinística, fornecida ou calculada, que sensibiliza o quão restritivo é o problema. Uma data de entrega $d \geq \sum^{n}{}_{i=i} p_i$, i.e., igual ou maior que a soma de todos os tempos de processamento das tarefas, é considerada uma variável de decisão, não restritiva e permite o sequenciamento das tarefas sem considerá-la. Caso seja menor que o tempo total de processamento, ela se torna restritiva e passa a ser considerada para o sequenciamento das tarefas.

A complexidade do problema também depende dos termos de penalidades por unidade de tempo de atraso e adiantamento que, quando combinados com a data de entrega em comum restritiva ou não restritiva, determinam se o problema é solucionável em tempo polinomial ou NP-difícil. Biskup resumiu essas relações na Tabela 1.

A data de entrega em comum para os problemas considerados neste trabalho é restritiva e definida por:

$$d = h * \sum_{i=1}^{n} p_i \tag{1}$$

sendo que h pode assumir o valor de 0,2, 0,4, 0,6 ou 0,8 dependendo do nível de restrição do problema. As penalidades por unidade de tempo de atraso e adiantamento também são determinísticas e distintas para cada um dos problemas. Portanto, este é definido como NP-difícil e aberto, i.e., torna improvável encontrar algoritmos eficientes capazes de obter a solução ótima do problema.

2.1 Propriedades

Biskup descreve duas importantes propriedades do problema considerado e que são ilustradas na Figura 1.

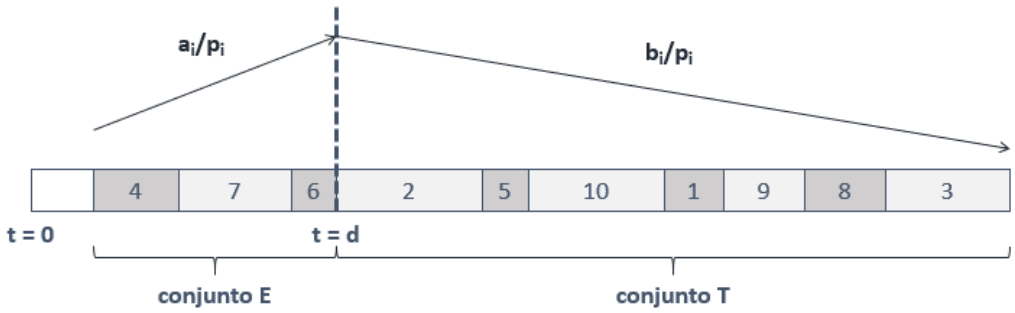

Figura 1: Exemplo de um sequenciamento ótimo de 10 tarefas, ilustrando as Propriedades I e II.

2.1.1 Propriedade I: Não há tempo ocioso entre tarefas sucessivas em um planejamento ótimo

É possível que, em um planejamento ótimo, a primeira tarefa não inicie no instante 0. Porém, não há intervalos ociosos entre tarefas sucessivas.

Cada sequência de tarefas pode ser dividida em 3 subconjuntos E, D e T, com $|D| \leq 1$. O subconjunto E possui as tarefas que terminam antes da (ou exatamente na) data de entrega comum d, o subconjunto T possui as tarefas que iniciam depois da (ou exatamente na) data de entrega comum, e o conjunto D possui no máximo uma tarefa que inicia estritamente antes de d e finaliza após d.

2.1.2 Propriedade II: Todo planejamento ótimo possui uma propriedade em forma de V

As tarefas de E são ordenadas em ordem crescente de acordo com a relação a_i/p_i, e as tarefas de T são ordenadas em ordem decrescente de acordo com b_i/p_i.

2.2 Formulação matemática

Para instâncias menores, é possível encontrar o planejamento ótimo S* através da formulação com programação inteira mista descrita nesta seção. s_i e x_{ik} são as variáveis de decisão que, juntas com d, definem S*. s_i é o tempo de início de uma tarefa, x_{ik} recebe 1 se a tarefa i iniciar antes de k (senão, 0) e R é um número suficientemente grande (big-M). A soma das penalidades de atrasos e adiantamentos das tarefas do problema é definida conforme a equação (2) a seguir:

$$f(S) = \sum_{i=1}^{n} a_i E_i + \sum_{i=1}^{n} b_i T_i \tag{2}$$

sendo S uma solução factível para o problema. O objetivo é encontrar um planejamento S^* que minimize a equação (2) sujeito a:

$$T_i \geq s_i + p_i - d, i = 1, ..., n \tag{3}$$

$$E_i \geq d - s_i - p_i, i = 1, ..., n \tag{4}$$

$$s_i + p_i \leq s_k + R(1 - x_{ik}), i = 1, ..., n-1, k = i+1, ..., n \tag{5}$$

$$s_k + p_k \leq s_i + Rx_{ik}, i = 1, ..., n-1, k = i+1, ..., n \tag{6}$$

$$T_i, E_i, s_i \geq 0, i = 1, ..., n \tag{7}$$

$$x_{ik} \in \{0, 1\}, i = 1, ..., n-1, k = i+1, ..., n \tag{8}$$

As restrições (3) e (4) determinam a quantidade de atraso e adiantamento, respectivamente, de cada tarefa i. As restrições (5) e (6) determinam o instante do processamento das tarefas: se i for executada antes de k, a equação (5) é restritiva para s_i, pois $x_{ik} = 1$ (sendo p_i maior que 0, s_k obrigatoriamente será maior que s_i, deixando a equação (6) irrestrita para s_k); caso k seja executada antes de i, $x_{ik} = 0$ e o inverso ocorre: a equação (6) passa a ser restritiva para s_k e a (5) passa a ser irrestrita para s_i.

3 Heurísticas propostas

3.1 Ideia geral

As heurísticas propostas neste trabalho se baseiam na heurística definida por Jarboui et al.[2], denominada Hte. Considere dois conjuntos de tarefas, E e T. Ambos são divididos pela data de entrega em comum d. O tempo de processamento total de E é limitado por d, enquanto a primeira tarefa de T tem $s_i = d$. Seu princípio consiste de inserir no conjunto

T, a cada iteração, a tarefa que, *a priori*, é menos interessante quando consideramos as primeiras tarefas a serem executadas, e reciprocamente para o conjunto E. As tarefas são ordenadas de forma decrescente em a_i/p_i (respectivamente b_i/p_i) para serem adicionadas ao conjunto T (respectivamente E). Quando não for mais possível alocar tarefas em E (no caso de E atingir o limite de processamento d) ou não houver mais tarefas a serem alocadas, o processamento para. No limite, será possível alocar no máximo $n/2$ tarefas ao conjunto E, e por isso é necessária uma fase de busca local para refinar os resultados da heurística.

A Hte considera que todo planejamento ótimo do problema considerado tem uma propriedade em formato de **V**, conforme descrito na Seção 2.1 deste trabalho. As tarefas i do conjunto E são ordenadas crescentemente em relação a a_i/p_i, e as tarefas i no conjunto T são ordenadas decrescentemente em relação a b_i/p_i.

Jarboui et al. demonstram que essa heurística Hte apresenta resultados satisfatórios e analisando como ela se comportava na prática, pode-se observar que algumas tarefas com alto b_i/a_i, indicando afinidade pelo conjunto T, eventualmente acabava vendo alocada no conjunto E. Considerando que tanto a heurística Hte quanto a heurística II proposta por Biskup produzem resultados satisfatórios, foi elaborada uma heurística que aproveitasse o melhor de cada. Considerando que uma tarefa com relação b_i/a_i alta, é esperado que ela seja alocada no conjunto T.

3.2 Detalhamento do método

A heurística desenvolvida neste trabalho consiste de duas etapas: uma alocação inicial de tarefas em dois conjuntos com base em um valor de corte para um indicador z, definido a seguir, e uma segunda etapa de alocação baseada na heurística Hte.

Figura 2: Fluxo da heurística proposta.

3.2.1 Etapa I: Pré-alocação baseada em z_{corte}

Na primeira etapa, é escolhido um valor de corte para a relação b_i/a_i e todos serviços que excedem esse valor são alocados neste conjunto. De forma a uniformizar essa medida, em vez de utilizar a relação b_i/a_i, é proposto um indicador equivalente $z = (b_i - a_i)/(b_i+a_i)$, que pode variar entre -1 e 1, dependendo do conjunto que a tarefa tem mais afinidade. O valor de corte é chamado de z_{corte}. As tarefas com $z > z_{corte}$, indicando $b_i \gg a_i$ são alocados no conjunto E, e os itens com $z < -z_{corte}$ são alocados no conjunto T, sempre observando se o conjunto E ainda possui capacidade para mais tarefas.

3.2.2 Etapa II: Alocação baseada em Hte e busca determinística

Na segunda etapa, os itens que restaram no grupo intermediário, chamado de **grupo sem preferência**, são alocados conforme preconiza a heurística Hte e reordenados da forma ótima conforme a propriedade em V. Vale ressaltar que o valor z de cada serviço permanece como critério de desempate entre tarefas com relações a_i/p_i e b_i/p_i idênticas.

Uma busca determinística também foi inserida nesta etapa, de forma a procurar em E tarefas interessantes de se enviar para T. Com base no maior a_i/p_i das tarefas do conjunto E, a busca inicia removendo a tarefa mais interessante de E e inserindo em T. As tarefas são então reordenadas de acordo com a propriedade em V e o objetivo é calculado. Caso este melhore em relação ao anterior, a busca continua. Senão, o penúltimo objetivo é utilizado.

3.2.3 Etapa III: Busca em z's

Foi observado que melhores objetivos de algumas instâncias eram obtidos com z_{corte} mais apertados (entre 0,75 e 0,9), e para outras com valores menos restritivos ($z_{corte} = 0{,}5$). Por esse motivo foi inserida uma terceira etapa, que resolve os problemas com z_{corte} de 0,25, 0,50, 0,75, 0,9 e 2,0, e consolida os menores valores encontrados para cada problema.

No caso de z=2,0, não existe a pré-alocação na Etapa I, pois $z \in \{0, 1\}$. Neste caso, a heurística passa a funcionar de forma similar à Hte, com a adição de uma ordenação baseada no indicador z, que passa a funcionar como critério de desempate nos casos em que a_i/p_i (ou b_i/p_i) de tarefas sucessivas das listas ordenadas são iguais.

4 Resultados computacionais

Na Tabela 2 são apresentados os resultados dos experimentos dos 280 problemas em comparação aos resultados dos ótimos/limites obtidos por Biskup. Nesta, conseguimos observar que, para problemas mais restritivos (h = 0,2 ou h = 0,4), a heurística é efetiva e apresentou um melhor resultado para os problemas com pelo menos 50 tarefas. Além disso, há uma melhora gradual conforme o aumento do número de tarefas, sendo o pior desempenho para problemas de 10 tarefas e o melhor para 1000 tarefas.

Se forem desconsiderados os problemas pequenos (até 20 tarefas) e menos restritivos (h maior ou igual a 0,6), esta encontrou objetivos entre 1,28% e 5,84% menores do

Tabela 2: Comparação dos objetivos obtidos pelos experimentos com os apresentados por Biskup e Feldmann[1]. Cada célula representa a variação (em porcentagem) da média dos 10 problemas de cada experimento em relação aos valores apresentados por Biskup para os conjuntos n e parâmetros h após a etapa de busca em z. Na última coluna estão os resultados da Hte por Jarboui et al.[2].

Dif.(%) **h** \ **n**	**10**	**20**	**50**	**100**	**200**	**500**	**1000**	**Média**	**Hte**
0,2	6,90	2,04	-3,46	-4,64	-4,80	-5,25	-5,84	**-2,15**	**3,89**
0,4	14,25	3,41	-1,28	-2,86	-2,30	-2,05	-2,97	**0,89**	**0,26**
0,6	4,82	1,83	0,48	0,27	0,06	0,02	0,04	**1,08**	**-1,63**
0,8	4,77	1,42	0,08	0,25	0,06	0,02	0,04	**0,95**	**-0,86**
Média	**7,69**	**2,18**	**-1,04**	**-1,74**	**-1,74**	**-1,81**	**-2,18**	**0,19**	**0,42**

Tabela 3: Tempos médios de processamento dos experimentos para as dimensões n.

n	**10**	**20**	**50**	**100**	**200**	**500**	**1000**
t (s)	0,003	0,005	0,011	0,021	0,042	0,112	0,237

que Biskup. Considerando todos os problemas, a heurística deste trabalho obteve um resultado em média 0,19% maior.

Os melhores resultados obtidos para h a partir de 0,6 são provenientes de $z_{corte} = 0{,}5$, enquanto os melhores resultados obtidos para h até 0,4 são provenientes de $z_{corte} = 0{,}9$ e $z_{corte} = 2{,}0$ (dados completos por z_{corte} disponíveis no arquivo **export.xlsx** anexo). Isso mostra que a busca em z_{corte} foi essencial para a obtenção dos melhores resultados consolidados, e que a elaboração de uma busca local mais robusta pode melhorar os resultados obtidos para h a partir de 0,6.

Na última coluna da Tabela 2 estão os resultados disponíveis da heurística original Hte na literatura. No geral, a Hte teve um desempenho melhor para cenários menos restritivos (h a partir de 0,6), mas a melhoria proposta neste trabalho trouxe melhores resultados consolidados (média de 0,19% contra 0,42% do original).

4.1 Tempos de processamento

Na Tabela 3 estão disponíveis os tempos médios de execução para cada conjunto de problemas de dimensões n. Os tempos de processamento aumentam linearmente conforme a dimensão do problema aumenta, indicando uma complexidade O(n) para o algoritmo proposto. A Figura 3 indica a relação linear entre os tempos de execução e as dimensões do problema: a regressão linear entre ambos apresentou um R^2 de 99,9%.

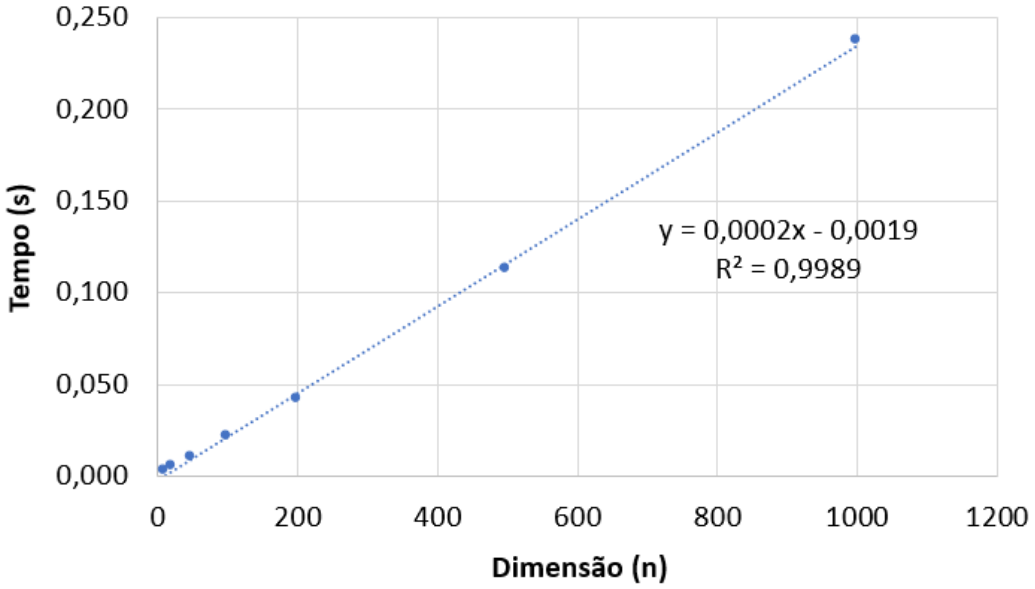

Figura 3: Tempos de execução em segundos vs. dimensões n.

Os experimentos foram realizados com programação na linguagem Python e em uma máquina com as seguintes configurações:

- Processador: Intel(R) Core(TM) i7-8750H CPU @ 2.20GHz.
- Memória RAM: 8GB.

5 Conclusão

Os resultados obtidos demonstram que uma abordagem mista que a princípio priorize os serviços pela diferença entre os seus coeficientes a_i e b_i em seguida por quanto os serviços remanescentes penalizarão a função objetivo é uma estratégia válida, principalmente em problemas com grande número de itens e restrição na capacidade do conjunto E, visto que a heurística de referência H_{te} apresentava resultados melhores justamente nos casos com h mais alto.

Os próximos passos para continuar aprimorarando a heurística pode envolver uma limitação na quantidade de serviços que seriam pré-alocados, seja esse limitante uma parcela do número de serviços do problema ou como uma parcela do tempo disponível no conjunto E.

Devido ao desempenho abaixo do esperado nas instâncias com h a partir de 0,6, sugere-se também que a etapa de busca ao fim da heurística H_{te} pode ter um potencial maior do que o explorado na implementação aqui proposta.

O conceito do z_{corte} também poderia ter sido implementado em conjunto de outras heurísticas para avaliar seu potencial.

6 Anexos

Aqui estão descritos os arquivos que seguem anexos a este relatório:

- **export.xlsx**: planilha Excel com os resultados dos experimentos.
- **codigo_final.py**: código Python completo.

Referências

[1] Dirk Biskup and Martin Feldmann. Benchmarks for scheduling on a single machine against restrictive and unrestrictive common due dates. *Computers & Operations Research*, 28(8):787–801, 2001.

[2] Bassem Jarboui, Patrick Siarry, and Jacques Teghem. *Metaheuristics for Production Scheduling (ISTE)*. Wiley-IEEE Press, 1st edition, 2013.