UM MÉTODO HEURÍSTICO INTEGRADO AO *SIMULATED ANNEALING* PARA A PROGRAMAÇÃO DE TAREFAS EM UMA MÁQUINA

Eder Oliveira Abensur

Centro de Engenharia, Modelagem e Ciências Sociais Aplicadas, UFABC
Av. dos Estados 5001 – CEP: 09210-580 - Santo Andre, SP – Brasil
e-mal: eder.abensur@ufabc.edu.br

RESUMO

O propósito desse estudo foi elaborar uma heurística construtiva prática e eficiente para o problema de programar um conjunto de *n* tarefas com tempos de processamento determinísticos e uma data de entrega comum em uma única máquina sendo que cada tarefa possui diferentes penalidades por atraso ou adiantamento em relação a data de entrega comum. Os resultados foram comparados contra o benchmark publicado por Biskup e Feldmann (1999) e, posteriormente, a metaheurística Simulated Annealing é aplicada para avaliação das possibilidades de melhoria.

Palavras-chave: **Heurística, Simulated Annealing, Single Machine Scheduling**

Área Principal: MH – Metaheurísticas

ABSTRACT

The purpose of this study was to develop a practical and efficient constructive heuristic to the problem of scheduling a set of *n* jobs with deterministic processing times and a common due date on a single machine and each task has different penalties for delay or advance for common due date. Results were compared against the benchmark published by Biskup and Feldmann (1999) and, subsequently, the metaheuristic Simulated Annealing is applied to assess the possibilities for improvement.

Keywords: Heuristic, Simulated Annealing, Single Machine Scheduling

Main Area: MH – Metaheuristics

1. Introdução

A heurística é um procedimento para resolver problemas através de uma abordagem intuitiva, em geral racional, no qual a estrutura do problema possa ser interpretada e explorada de forma a obter uma solução razoável. Como características principais dos procedimentos heurísticos destacam-se: o bom desempenho, rapidez e simplicidade.

A partir da década de 80 do século XX há o surgimento da Metaheurística que fortalece e revigora as técnicas heurísticas por mecanismos de busca inspirados em modelos do cotidiano humano ou da natureza. Fazem parte dessa relação:

a) Busca Tabu: faz analogia com a descoberta de uma trilha durante a escalada de uma montanha;
b) *Simulated Annealing*: faz analogia ao resfriamento de materiais;
c) Algoritmo Genético: faz analogia a reprodução humana com diversificação e mutação de cromossomos;
d) *Ant System*: analogia com um sistema de busca de comida por uma colônia de formigas.
e) Híbridos: combinação de duas ou mais técnicas.

As Metaheurísticas alargaram o campo de atuação e aumentaram as possibilidades de obtenção de melhores resultados em comparação às técnicas heurísticas anteriores. Em comum, elas surgiram para a solução de problemas específicos entre eles o tema desse trabalho: a programação de tarefas em uma máquina com diferentes penalidades para atraso ou adiantamento.

Há décadas que o problema de programação de tarefas em uma ou várias máquinas vem sendo pesquisado e a lista de propostas e contribuições sobre o assunto é vasta. Baker e Scuder (1990) fazem uma revisão cronológica sobre as variações do tema classificando-as em sete categorias diferentes de acordo com a função objetivo. Entre as referências primárias citadas nesse trabalho estão: Bagchi et al. (1987), Baker e Chadowitz (1989), Cheng (1987), Eilon e Chowdhurry (1977) e Gupta e Kypaparisis (1987).

Bector et al. (1988) faz um tratamento prévio de programação linear para definir a seqüência inicial de um método que minimiza as folgas das atividades. Entre as principais referências estão: Bagchi et al. (1986), Cheng (1987), Eilon e Chowdhurry (1977) e Gupta e Kypaparisis (1987).

Bagchi et al. (1987) propõe uma heurística que minimiza o desvio padrão das datas de conclusão das tarefas em relação à data de entrega comum. Ele consegue melhorar resultados para programações superiores a 20 tarefas a partir dos estudos de Eilon e Chowdhurry (1977).

Quaddus (1987) propõe um modelo de programação linear para determinação da *constant-allowance due-date* (COM) minimizando a somatória dos custos das tarefas com diferentes penalidades para término antes, depois e na própria data de entrega comum. As referências destacadas são: Eilon e Chowdhurry (1977) e Weeks e Fryer (1977).

Eilon e Chowdhury (1977) elaboram e definem a eficiência de quatro diferentes heurísticas para determinar uma seqüência ótima que minimize a variância do tempo de espera. Curiosamente, a heurística mais simples é a mais eficiente.

Entre muitos possíveis métodos heurísticos, Eilon e Chowdhurry (1977) destacaram quatro heurísticas que foram avaliadas de acordo com seu desempenho e eficiência para a elaboração de seqüências que minimizassem a variância do tempo de espera para problemas com até 100 tarefas. Os resultados obtidos mostraram que o método 1.2 proposto obteve o melhor desempenho.

As etapas desse método são as seguintes:

a) O conjunto de N tarefas é classificada em ordem decrescente por tempo de processamento;
b) A tarefa mais longa é programada por último, a segunda mais longa em primeiro, a terceira e quarta tarefas mais longas são respectivamente a penúltima e antepenúltima da lista. As demais tarefas são posicionadas alternadamente no fim e no começo da lista;
c) O resultado é uma seqüência em forma de V com igual ou quase o mesmo número de tarefas em cada lado.

Conforme Baker e Scuder (1990), seja n o número de tarefas sendo que cada uma possui um tempo de processamento p_j e uma data prevista de conclusão d_j. Como resultado das decisões de programação, a tarefa j terá um tempo de conclusão C_j. Associado a cada uma há uma penalidade por adiantamento $\alpha_j > 0$ e uma penalidade de atraso $\beta_j > 0$. Assumindo-se que as funções de penalidade são lineares, a função objetivo para uma seqüência S pode ser escrita como $f(S)$, onde:

$$f(S) = \sum_{j=1}^{n} (\alpha j Ej + \beta j Tj)$$

sendo:

$$Ej = \max(0, d_j - C_j)$$
$$Tj = \max(0, C_j - d_j)$$

Ainda segundo Baker e Scuder (1990), a solução ótima apresenta-se no formato "V", sendo que no primeiro ramo as tarefas estão em ordem decrescente de p_j / α_j e no segundo ramo em ordem crescente de p_j / β_j. As ordens decrescentes em p_j / α_j posicionam-se antes da data de entrega comum (bico do "V") e as tarefas em ordem crescente de p_j / β_j após a mesma.

O trabalho de Biskup e Feldmann (1999) trouxe uma grande contribuição ao problema citado analisando as suas propriedades quando submetido a uma restrição adicional: uma data de entrega comum (*due date*) para todas as tarefas. As propriedades são:
a) Há uma seqüência ótima no formato "*V*";
b) De duas uma: a primeira ordem começa em t=0 (momento inicial para contagem do tempo) ou uma tarefa é concluída na data de entrega comum.

De acordo com a revisão bibliográfica realizada nota-se que o trabalho de Eilon e Chowdhury (1977) representa uma intersecção entre vários tipos de abordagem sobre o tema. Desse modo surgiu a motivação dessa pesquisa baseada na possibilidade de adequação das suas heurísticas para o problema analisado.

Este estudo visa desenvolver uma heurística simples, rápida e eficiente ajustada às características do problema de processar um conjunto de *n* tarefas em uma mesma máquina sendo que cada tarefa possui diferentes penalidades de atraso e adiantamento em relação a uma data de entrega comum pré-definida. A Metaheurística *Simulated Annealing* é usada para complementar o estudo e avaliar o seu desempenho de refinamento em relação à heurística proposta.

A heurística denominada como 1.2, proposta por Eilon e Chowdhury (1977), foi adaptada às características do problema e adotada como heurística construtiva para geração da seqüência inicial. O ciclo heurístico é encerrado com o uso da técnica de Troca de Pares (a*ll pairs*) como melhoria da solução inicial. Os resultados foram comparados contra o benchmark publicado por Biskup e Feldmann (1999) para comparação de desempenho para 10 problemas com 10, 20, 50, 100, 200, 500 e 1000 tarefas cada um submetidos a quatro diferentes datas de entrega comum totalizando 280 avaliações. A parte final do trabalho é o uso do *Simulated Annealing* para avaliação da sua eficiência na busca de melhores resultados. A programação foi feita com a linguagem *Visual Basic for Apliccations* (VBA).

O trabalho está estruturado como segue. As seções 2, 3 e 4 descrevem, respectivamente, as heurísticas construtivas, heurísticas de melhoria e o *Simulated Annealing*. A seção 5 apresenta os testes aplicados e os resultados obtidos. A seção 6 mostra as conclusões do estudo.

2. Heurística Construtiva

As heurísticas de construção geram uma solução através da adição de componentes individuais, um de cada vez, até achar uma solução factível. Devido a sua simplicidade e rapidez são empregadas para gerar uma solução inicial.

A heurística construtiva proposta parte da idéia central do método 1.2 de Eilon e Chowdhurry (1977) ajustada pelas propriedades do problema. A descrição do método é a seguinte:

Para problemas com data de entrega comum inferior a 60% da somatória dos tempos das tarefas:

a) O conjunto de *N* tarefas são classificadas em ordem crescente de β_j / α_j ;
b) As tarefas iniciais da seqüência anterior são colocadas por último na lista até totalizarem 30% das *n* tarefas a serem programadas. As demais tarefas são posicionadas alternadamente no fim e no começo da lista;
c) O resultado é uma seqüência em forma de *V*;
d) As tarefas anteriores a data de entrega comum são classificadas em ordem decrescente de p_j / α_j e as posteriores em ordem crescente de p_j / β_j .
e) A seqüência inicial definida é submetida a atrasos de acordo com a diferença entre as datas de conclusão das ordens adiantadas e a data de entrega comum, assim, obrigatoriamente, uma ordem termina na "*due date*" respeitando a segunda propriedade de Biskup e Feldmann (1999) citada anteriormente.

Para problemas com data de entrega comum igual a 80% da somatória dos tempos das tarefas

a) O conjunto de N tarefas são classificadas em ordem decrescente de β_j / α_j ;
b) As tarefas da seqüência anterior são colocadas na mesma ordem para serem programadas;
c) As tarefas anteriores a data de entrega comum são classificadas em ordem decrescente de p_j / α_j e as posteriores em ordem crescente de p_j / β_j ;
d) A seqüência inicial definida é submetida a atrasos de acordo com a diferença entre as datas de conclusão das ordens adiantadas e a data de entrega comum, assim, obrigatoriamente, uma ordem termina na "*due date*" respeitando a segunda propriedade de Biskup e Feldmann (1999) citada anteriormente.

Como exemplo, para um conjunto de 10 tarefas com data de entrega comum inferior a 60% da somatória dos tempos das tarefas, sendo *S* a posição da tarefa na seqüência e *N* a sua posição na classificação crescente de β_j / α, teríamos a seguinte seqüência inicial: $S_{10} = N_1$; $S_9 = N_2$; $S_8 = N_3$; $S_1 = N_4$; $S_7 = N_5$; $S_2 = N_6$; $S_6 = N_7$; $S_3 = N_8$; $S_5 = N_9$; $S_4 = N_{10}$.

3. Heurística de Melhoria

As heurísticas construtivas geram uma solução inicial. Caso ela não seja satisfatória, empregam-se procedimentos de melhoria sobre a solução obtida. O procedimento escolhido para refinar a solução inicial foi o de troca de pares (a*ll pairs*). Por esse método cada tarefa localizada no ramo alfa da seqüência (adiantadas) era trocada por uma tarefa do ramo beta (atrasadas). A cada permutação de tarefas o valor da função objetivo é calculado. O procedimento é encerrado quando não houver mais trocas a fazer.

Por uma questão de eficiência e desempenho refletido em tempo de processamento, adotou-se um percentual aplicado sobre o número máximo de trocas possíveis como critério de parada da heurística de melhoria. Os percentuais utilizados foram os seguintes:

a) 100% para N = 10;
b) 100% para N = 20;
c) 100% para N = 50;
d) 50% para N = 100;
e) 25% para N = 200;
f) 10% para N = 500;
g) 10% para N=1000.

4. *Simulated Annealing*

Entre as muitas metaheurísticas escolheu-se o *Simulated Annealing* pela sua facilidade de implementação. Essa técnica criada por Kirkpatrick et al (1983), faz uma analogia ao processo térmico de resfriamento de um material a alta temperatura. Há uma avaliação do nível de energia a cada diminuição de temperatura e o processo é encerrado quando o ponto de solidificação ou de energia mínima é atingido. Os parâmetros desse método são:

a) Temperatura inicial (T_0);
b) Temperatura final (T_f);
c) Número de iterações para atingir o equilíbrio a uma dada temperatura (K);
d) Variação do nível de energia ou função objetivo (Δ);
e) Proporção de redução da temperatura (β);
f) Probabilidade de certas configurações terem sua energia aumentada de ΔE é: $p(\Delta E) = e^{-\Delta E/T}$.

Sendo uma função decrescente esse processo assemelha-se aos métodos de descida (*Descent Methods*) que procuram construir uma seqüência melhorada a partir de uma seqüência inicial estudando-se uma vizinhança que representa o conjunto de seqüências que podem ser geradas a partir da seqüência vigente.

Como as outras metaheurísticas ela pretende alargar o horizonte de alternativas analisadas. A questão do horizonte de alternativas pode ser entendido através de um exemplo cotidiano. Imagine que uma pessoa queira comprar uma pizza e resolva fazer uma pesquisa a pé pelo

seu bairro. Apesar do interesse, essa pessoa não estaria inclinada a fazer uma caminhada muito distante de sua casa e provavelmente limitaria a sua busca a poucos quarteirões de distância da sua residência. Além disso, alguns trajetos mais íngremes como subidas e áreas inseguras seriam evitados. Após algum tempo essa pessoa faria a melhor escolha sobre o número de alternativas encontradas. No entanto, poderia existir alguma pizzaria com uma melhor oferta do produto um pouco além do limite estabelecido ou talvez num dos trajetos evitados pelo consumidor.

O *Simulated Annealing* tenta evitar essa desvantagem inerente aos métodos de descida aceitando, de forma criteriosa, algumas seqüências que produzam um resultado inferior à seqüência vigente. Especificamente, quando $\Delta \leq 0$ a seqüência é aceita, no entanto, seqüências com $\Delta > 0$ poderão ser admitidas desde que $R \leq e^{-\Delta E/T}$.

Osman e Potts (1989) fazem uma aplicação do *Simulated Annealing* para o problema de n tarefas em m máquinas. Fazendo m igual a 1 reduz-se o problema para o caso analisado neste trabalho. De acordo com Osman e Potts (1989) os parâmetros do *Simulated Annealing* para essa situação são:

a) Temperatura inicial (T_0)

$$\sum_{i=1}^{n} pi / 5n$$

sendo:
n = número de tarefas
p = tempo de processamento

b) Temperatura final (T_f);
$T_f = 1$

c) Número de iterações para atingir o equilíbrio a uma dada temperatura (K);
K = max((3300 ln (n) + 7500 ln (n) -18250), 2000)

sendo:
m=1
n = 10, 20, 50, 100, 200, 500 e 1000
Tem-se:
K= 2.000 iterações para $10 \leq n \leq 200$
K = 2.258 iterações para n = 500 (adotou-se 2.000)
K = 4.545 iterações para n = 1000 (adotou-se 4.000)

d) Proporção de redução da temperatura (β);
$\beta = (T_0 - T_f) / ((K-1)T_0T_f)$

e) Temperatura para uma dada iteração;
$T_{k+1} = T_k / (1 + \beta T_k)$

Ainda segundo Osman e Potts (1989) os melhores resultados foram obtidos a partir de trocas de pares escolhidos de forma aleatória. Esta foi a opção adotada para a construção do algoritmo. O pseudo-código do algoritmo desenvolvido para o *Simulated Annealing* é mostrado a seguir:

Selecionar uma seqüência inicial factível (heurística construtiva)
Definir a Temperatura inicial (T_0)
Determinar o número máximo de iterações (K)
Para K=1 até $Kmax$ fazer
    Para cada Temperatura (T_k)
    Fazer troca aleatória entre as tarefas dos ramos Alfa (adiantadas) e Beta (atrasadas)
    Guardar a seqüência gerada
    Colocar em formato "V"para início em t=0 e/ou t≠0
    Calcular Δ (Função Objetivo atual – Função Objetivo anterior)
    Se $\Delta \leq 0$ então
        Aceitar a nova seqüência
        Atualizar o valor da Função Objetivo
        Comparar com a função incumbente
        Se melhor que a função incumbente então
            Atualizar o valor da função incumbente
        Fim Se
    Senão
        Se Randômico [0,1] $\leq e^{-\Delta E/T}$
            Aceitar a nova seqüência
            Guardar o valor da nova seqüência
        Fim Se
    Fim Se
$K = K + 1$

5. Resultados

Os resultados obtidos são mostrados a seguir. Foram simulados dez problemas diferentes propostos por Biskup e Feldmann (1999) para cada grupo de tarefas. A tabela 1 mostra a diferença percentual dos resultados encontrados para cada tipo de problema por tipo de método comparados com o benchmark de Biskup e Feldmann (1999). Percebe-se que o *Simulated Annealing* consegue resultados melhores. As diferentes datas de entrega comum são apresentadas como um percentual (20%, 40%, 60% e 80%) em função da soma dos tempos das tarefas.

A tabela 2 resume e compara as médias gerais dos métodos e seus respectivos tempos de processamento. A heurística construtiva não apresenta tempos, pois está incorporada no programa que efetua a heurística de melhoria. O tempo médio do *Simulated Annealing* é superior, porém com resultados bem melhores.

Tabela 1 – Diferença percentual dos resultados encontrados contra o Benchmark de Biskup **e** Feldmann (1999) para cada tipo de data de entrega comum

	Heurística Construtiva					**Heurística de Melhoria**					**Simulated Annealing**				
N	20%	40%	60%	80%	**média**	20%	40%	60%	80%	**média**	20%	40%	60%	80%	**Méd.**
10	11	17,9	6,1	7,5	**10,6**	1,8	1,9	8,9	36,6	**12,3**	0,2	0,2	4,3	2,6	**1,8**
20	11,9	16,6	4	9	**10,4**	2,3	13	11,2	39,8	**16,6**	-3,6	-1,5	2,3	1,9	**-0,2**
50	9	10,4	2,8	11,2	**8,4**	3	4,5	1,5	28,6	**9,4**	-5,7	-4,6	0,4	0,4	**-2,4**
100	11,6	10,6	1,9	12,8	**9,2**	7,9	6,7	2,2	44,3	**15,3**	-6,2	-4,7	1,3	0,2	**-2,4**
200	9,9	8,1	4,8	15,5	**9,6**	8,1	14,9	13,7	56,9	**23,4**	-5,7	-3,5	1,3	0,1	**-1,9**
500	10,7	9,5	2,3	15,5	**9,5**	9,6	8,8	3,8	54,6	**19,2**	-6,2	-3	2	0	**-1,8**
1000	10,8	9,4	5,6	17	**10,7**	10,3	17,7	16,5	59	**25,9**	-6,5	-3,9	1,6	0,1	**-2,2**
Média	**10,7**	**11,8**	**3,9**	**12,6**		**6,1**	**9,6**	**8,3**	**45,6**		**-4,8**	**-3**	**1,9**	**0,8**	

Tabela 2 – Quadro sintético da diferença percentual dos resultados encontrados

	Heurística Construtiva		**Heurística de Melhoria**		**Simulated Annealing**	
N	% Difer.	Tempo médio (s)	% Difer.	Tempo médio (s)	% Difer.	Tempo médio (s)
10	10,6	-	12,3	1,4	1,8	104,7
20	10,4	-	16,6	4,5	-0,2	119,3
50	8,4	-	9,4	32,7	-2,4	132,7
100	9,2	-	15,3	56,3	-2,4	240,5
200	9,6	-	23,4	84,1	-1,9	275,5
500	9,5		19,2	274,9	-1,8	822,3
1000	10,7		25,9	1347,5	-2,2	2012,7
Média	**9,8**		**17,4**	**257,3**	**-1,3**	**529,7**

6. Conclusão

De acordo com os resultados obtidos na seção anterior comprovou-se que a heurística 1.2 proposta por Eilon e Chowdhurry (1977) e adaptada às características do problema é viável, rápida e eficiente para a programação de *n* tarefas em uma única máquina. Essa mesma heurística construtiva integrada a uma metaheurística como o *Simulated Annealing* produzem excelentes resultados levando-se em consideração o seu desempenho em termos de desvio percentual e tempo médio de processamento. Os resultados inferiores encontrados para a busca local para 500 e 1000 tarefas foram conseqüência da limitação do número de trocas imposto para privilegiar o tempo de execução.

Essa conclusão é reforçada pelos resultados obtidos para a programação de 10 tarefas. O benchmark de Biskup **e** Feldmann (1999) para esse conjunto apresenta solução ótima. A heurística desenvolvida obtém um desvio percentual de 3,2% em 1,4s de execução com

apenas 25 permutações de tarefas. Para um conjunto de 10 ordens há 10! permutações possíveis ou 3.628.800 possibilidades. Graças à adaptação das propriedades levantadas na revisão bibliográfica conseguiu-se resultados excelentes com no máximo 2.000 iterações ou 0,055% do total de alternativas.

Conforme esperado a metaheurística produziu melhores resultados do que a busca local. Em parte, essa melhoria foi obtida devido a uma adaptação do modelo de Osman e Potts (1989) para a situação estudada. Esse modelo foi concebido para um problema de *n* tarefas em *m* máquinas e não previa a possibilidade de inserção de um tempo ocioso para o início da programação. Esse ajuste foi incorporado no *Simulated Annealing* conseguindo ótimos resultados.

Conforme citado anteriormente, a heurística de Eilon e Chowdhurry (1977) foi mantida integralmente até as simulações com data de entrega comum representando 60% da soma dos tempos individuais das tarefas. Para a data de entrega comum igual a 80% foi feita uma segunda adaptação classificando as tarefas em ordem decrescente de β_j / α_j e aumentando-se de 30% para 100% o número de tarefas adiantadas. Esse procedimento ofereceu excelentes resultados contribuindo para a sugestão de pesquisar a personalização das heurísticas construtivas para cada tipo de data de entrega comum.

Referências

Bagchi, U.; Sullivan, R.; Chang, Y. (1986), Minimizing mean absolute deviation of completion times about a common due date. *Naval Research Logistical Quarterly*, 34, 227-240.

Bagchi, U.; Sullivan, R.; Chang, Y. (1987), Minimizing mean squared deviation of completion times about a common due date. *Management Science*, 33, 894-906.

Baker, Kenneth R.; Chadowitz, A. (1989), Algorithms for minimizing earliness and tardiness penalties with a common due date. Working paper n. 240, *Amos Tuck School of Business Administration*, Dartmounth College, Hanover, N.H.

Baker, Kenneth R.; Scuder, Gary D. (1990), Sequencing with earliness and tardiness penalties: a review. *Operations Research Society of America*, 38, 22-36,

Bector, C. Y; Gupta, Y; Gupta, M. (1988), Determination of an optimal common due date and optimal sequence in a single machine job shop. *International Journal of Production Research*, 26, 613-628.

Biskup, Dirk; Feldmann, Martin. (1999), Benchmarks for scheduling on a single machine against restrictive and unrestrictive common due dates. *Computers & Operations Research*, 28, 787-801.

Cheng, T. (1987), An algorithm for the CON due date determination and sequencing problem. *Computers & Operations Research*, 14, 537-542.

Eilon, S; Chowdhurry, I. (1977), Minimizing waiting time variance in the single machine problem. *Management Science*, 23, 567-575.

Gupta, S.; Kypaparisis, J. (1987), Single machine scheduling research. *OMEGA*, 15, 207-227.

Kirkpatrick S.; Gelatt C.; Vecchi P. (1983), Optimization by simulated annealing. *Science*, 220, 671-679.

Osman, I.H..; Potts, C.N. (1989), Simulated Annealing for permutation flow-shop scheduling. *OMEGA*, 17, 551-557.

Quaddus, M. (1987), A generalized model of optimal due-date assignment by linear programming. *Journal of the Operational Research Society*, 38, 353-359.

Weeks, J.K.; Fryer, J.S. (1977), A methodology for assigning minimum cost due dates. *Management Science*, 23, 872-881.