Heurísticas Construtivas para o Problema de Balanceamento em Linhas de Produção com Trabalhadores Deficientes

Mayron César O. Moreira[1], Alysson M. Costa[1]

[1]Instituto de Ciências Matemáticas e de Computação – Universidade de São Paulo (USP)
Caixa Postal 668 – 13.560-970 – São Carlos – SP – Brasil

{mayron,alysson}@icmc.usp.br

Abstract. *We study constructive heuristics for the Assembly Line Worker and Assignment Balancing Problem (ALWABP). This problem typically occurs in assembly lines in sheltered work centers for the disabled. In this situation, the task execution times are heterogeneous. The development of efficient and fast heuristics procedures is important for the obtained solutions can be used both as a final planning procedures as well as initial solutions for more elaborated methods. In this short article, we propose four constructive heuristics for the ALWABP. Computational tests are done in order to compare these solutions both in terms of their qualities and in terms of their ability to accelerate the convergence times of a tabu search algorithm.*

Resumo. *Neste trabalho, estudam-se heurísticas construtivas para o Problema de Balanceamento e Designação de Trabalhadores em Linhas de Produção, conhecido na literatura como ALWABP (do inglês:* Assembly Line Worker Assignment and Balancing Problem*). Este problema é típico de linhas de produção com trabalhadores deficientes, onde os tempos de execução de cada tarefa são diferentes para cada trabalhador. O desenvolvimento de procedimentos heurísticos para geração de soluções iniciais de boa qualidade são de fundamental importância para o uso em métodos aproximativos mais elaborados dado que um bom ponto de partida tende a diminuir o tempo de convergênc desses métodos. Neste trabalho, propomos quatro heurísticas construtivas p o problema de balanceamento em linhas de produção com trabalhadores cientes, um estudo ainda inédito na literatura. Testes computacionais são ados para comparar os quatro procedimentos entre si e como solução ini um algoritmo de busca tabu. Os resultados indicam uma boa performc heurísticas nas duas situações.*

1. Introdução

O problema de alocação de trabalhadores com deficiências em li começou a ser abordado bem recentemente. [Miralles et al. 2007] i lema e o nomearam Problema de Balanceamento de Linhas de Pr de Trabalhadores (ALWABP, do inglês *Assembly Line Worker As ing Problem*). Este problema tem sua relevância técnica realçac prática e social, dados 1) a grande quantidade de pessoas pr - 10% da população mundial, segundo a OMS e 2) a grand do mercado de trabalho em países como o Brasil - cerca

deficiência em idade ativa do município de São Paulo se encontravam desempregados em 2004 [SERPRO 2004].

Os autores realçam a diferença entre o ALWABP e o problema clássico de balanceamento de linhas de produção: no caso do ALWABP, cada trabalhador executa uma tarefa com uma velocidade diferente, diferentemente do caso clássico onde os trabalhadores podem ser considerados iguais. Os autores propõem uma formulação matemática para o problema e um estudo de caso [Miralles et al. 2007], além de um algoritmo de *branch-and-bound* [Miralles et al. 2008]. [Chaves et al. 2009] resolve o mesmo problema com o auxílio de uma heurística de busca por agrupamento (*clustering search*). [Moreira and Costa 2009] apresentam um algoritmo de Busca Tabu que, apesar de intencionalmente simplificado, apresenta bons resultados para o ALWABP.

Neste trabalho, propomos quatro heurísticas construtivas para o problema de balanceamento de linhas de produção com trabalhadores deficientes. A idéia é utilizar conceitos apresentados para problemas similares, como o problema relacionado com o caso em que os trabalhadores executam as tarefas em tempos iguais - O Problema Simples de Balanceamento de Linhas (SALBP, do inglês, Simple Assembly Line Balancing Problem). De fato, existem diversos trabalhos com proposições de heurísticas para o SALBP. Essas abordagens heurísticas consistem em métodos baseados em prioridades e procedimentos de enumeração restrita [Talbot et al. 1986]. [Scholl and Voß 1996] apresentam várias heurísticas para distintas versões do SALBP e nossa idéia neste artigo é justamente estender as idéias destes autores para o ALWABP.

O restante deste documento está organizado da seguinte maneira: na seção seguinte apresentam-se as heurísticas propostas. Em seguida, resultados comparativos entre as quatro abordagens são expostos e analisados na seção 3. Além disso, cada solução gerada por cada heurística é utilizada como ponto de partida do algoritmo de busca tabu minimalista proposto por [Moreira and Costa 2009]. Conclusões são apresentadas na seção 4.

2. Heurística Construtiva Proposta

Neste trabalho, as heurísticas propostas possuem um caráter guloso, isto é, fazem sempre escolhas sequenciais, localmente - mas não necessariamente globalmente - ótimas. A idéia básica por trás do algoritmo é tentar, em cada máquina, alocar o trabalhador que consiga executar o maior número tarefas cujo tempo total de execução não ultrapasse o valor de tempo de ciclo pré-estabelecido. Assim, ao fim dessa escolha gulosa, aloca-se o melhor dos trabalhadores e o processo continua até que todos os trabalhadores estejam alocados nas máquinas. O algoritmo 1 apresenta o pseudocódigo da heurística construtiva proposta.

As linhas 1 e 2 atribuem os valores de limite inferior (LB, do inglês *lower bound*) e superior (UB, do inglês *upper bound*) de uma dada instância às variáveis *LB* e *UB*, respectivamente. Neste trabalho, utilizou-se a seguinte estimativa para o limite inferior: $\max\{\sum_i t_i^*/|W|, \max\{t_i^*\}\}$, onde t_i^* é o menor tempo de execução da tarefa i considerando todos os trabalhadores. Para uma estimativa do limite superior, foi utilizado o valor constante 200, obtido através de uma majoração dos melhores resultados apresentados até hoje na literatura.

As linhas 4-33 se referem ao laço principal do algoritmo. Assim, partindo de um

Algoritmo 1 Heurística Construtiva para o ALWABP

```
1: LB: Limitante inferior para a determinada instância;
2: UB: Limitante superior para a determinada instância;
3: melhorTarefasAlocadas ← 0; /*total de tarefas na melhor alocação*/
4: para C ←LB até UB faça
5:   totTarefasAlocadasIter ← 0; /*total de tarefas alocadas em cada iteração*/
6:   para cada s ∈ S faça
7:     melhorAlocacaoMaquina ← 0;
8:     para cada w ∈ W faça
9:       se w não foi alocado então
10:        melhorAlocacaoTemp ← 0;
11:        enquanto o tempo total das tarefas alocadas para o trabalhador w não ul-
           trapassar C faça
12:          Escolha a melhor tarefa i não alocada de acordo com o critério adotado;
13:          Aloque i para o trabalhador w; /*não alocamos a tarefa de fato, apenas
             simulamos que a mesma ficaria alocada na máquina s e com o trabalhador
             w*/
14:          melhorAlocacaoTemp ← melhorAlocacaoTemp + 1;
15:        fim
16:        se melhorAlocacaoTemp > melhorAlocacaoMaquina então
17:          Guarde o trabalhador w e as tarefas que serão alocadas a ele;
18:          melhorAlocacaoMaquina ← melhorAlocacaoTemp;
19:        fim
20:      fim
21:    fim
22:    Aloque o melhor trabalhador de W na máquina s com suas respectivas tarefas;
23:    totTarefasAlocadasIter ← totTarefasAlocadasIter +
       melhorAlocacaoMaquina;
24:  fim
25:  se totTarefasAlocadasIter = |N| então
26:    Retorne a solução factível;
27:  senão
28:    se melhorTarefasAlocadas < totTarefasAlocadasIter então
29:      Guarde a solução infactível que aloque mais tarefas;
30:      melhorTarefasAlocadas ← totTarefasAlocadasIter;
31:    fim
32:  fim
33: fim
```

valor de tempo de ciclo igual a LB, tenta-se encontrar uma solução que possui o menor valor de C ou uma solução que alocasse o maior número de tarefas, caso não se consiga uma solução factível. As linhas 6-24 correspondem ao laço em que ocorre a tentativa de alocação para cada máquina do conjunto S. Essa alocação involve tanto a de trabalhadores quanto a de tarefas ao mesmo.

Os trechos correspondentes a 8-21 estão associados a verificação do melhor trabalhador a ser alocado na máquina s, levando-se em conta a quantidade de tarefas que w

consegue executar em um tempo de ciclo C (4-33). Note que na linha 13, são "alocadas" as melhores tarefas de acordo com o critério estabelecido. Neste trabalho, utilizamos quatro critérios para auxiliar na escolha de qual tarefa deve ser alocada a uma máquina para que um trabalhador a execute. A determinação dos critérios promove uma extensão do trabalho de [Scholl and Voß 1996]. Assim, os critérios considerados neste trabalho são os seguintes:

- **Critério 1**: $t[i][w] - t[i][\bar{w}]$, onde: $t[i][w]$ é o tempo de execução da tarefa i pelo trabalhador w e $\bar{w} \neq w$ é um outro trabalhador de W que executa a tarefa i com menor tempo;
- **Critério 2**: $\frac{t[i][w]}{t[i][\bar{w}]}$;
- **Critério 3**: $\frac{n_{succ}[i]}{t[i][w]}$, onde $n_{succ}[i]$ é o número de tarefas sucessoras de i que ainda não foram alocadas;
- **Critério 4**: $\frac{n_{imedSucc}[i]}{t[i][w]}$, onde $n_{imedSucc}[i]$ é o número de tarefas sucessoras imediatas de i que ainda não foram alocadas.

Os critérios 1 e 2 são considerados em prioridade crescente enquanto 3 e 4 em prioridade decrescente.

As linhas 26-32 correspondem a fase de verificação se uma solução factível para a alocação de tarefas aos trabalhadores foi satisfeita, dado um valor de tempo de ciclo C. Note que a linha 25 testa se todas as tarefas foram alocadas. Caso seja verdadeiro, retornamos para o procedimento invocador essa solução obtida e terminamos o algoritmo. Caso contrário (linha 27), chegamos a conclusão que não conseguimos gerar uma solução inicial factível, ou seja, nem todas as tarefas foram alocadas; assim sendo, guardamos essa solução infactível até encontrarmos uma solução factível ou até encontrar uma solução infactível que aloque mais tarefas que a atual, para outro valor de C. Observe que, agindo desta forma para o caso de infactibilidade, o algoritmo dá maior prioridade a uma solução que aloque mais tarefas do que uma solução que possua um menor tempo de ciclo.

3. Resultados Computacionais

As heurísticas descritas na seção 2 foram implementadas na linguagem C++, utilizando o compilador g++ 4.3, no sistema operacional Ubuntu Linux 8.10. Os testes foram feitos em um computador Core 2 Duo com processador de 2.2 GHz e 3.0 Gb de RAM. Assim, os algoritmos propostos foram submetidos a testes computacionais para a comparação das suas eficiências e aplicabilidades. Para os testes, utilizou-se um conjunto de instâncias geradas a partir de instâncias clássicas para o SALBP. Detalhes da geração das instâncias podem ser encontrados em [Chaves et al. 2007]. As instâncias são agrupadas em quatro famílias: Roszieg, Heskia, Tonge e Wee-Mag.

Todas as instâncias foram testadas para cada heurística proposta. As heurísticas construtivas foram denominadas de acordo com o seu critério, ou seja, HC1, HC2, HC3 e HC4 são heurísticas construtivas que utilizam os critérios 1, 2, 3 e 4, respectivamente. Como primeiro critério, foi comparado o número de soluções factíveis geradas por cada heurística em cada família de instâncias. A Tabela 1 apresenta estes resultados.

	HC1	HC2	HC3	HC4
Roszieg	80	80	80	80
Heskia	66	66	62	62
Tonge	61	62	61	60
Wee-Mag	80	80	80	80
Total	287	288	283	282

Tabela 1. Número de soluções factíveis obtidas por cada heurística

	HC1	HC2	HC3	HC4
Roszieg	24,89	23,81	24,84	24,66
Heskia	88,62	88,35	95,24	95,61
Tonge	97,62	95,53	101,43	99,17
Wee-mag	39,41	39,00	47,89	44,33

Tabela 2. Média dos tempos de ciclo encontrados para cada família de instâncias

Como pode-se observar na Tabela 1, as heurísticas se comportaram bem similares quanto ao número de soluções factíveis encontradas; as heurísticas HC1 e HC2 foram ligeiramente melhores que as heurísticas HC3 e HC4, obtendo aproximadamente 90% de factibilidade nas instâncias das quatro famílias consideradas. Com o objetivo de avaliar a qualidade das soluções das quatro abordagens propostas, foram selecionadas as instâncias às quais todas as heurísticas geraram soluções factíveis e a média dos tempos de ciclo das soluções obtidas são comparadas na Tabela 2, onde as colunas S e T(s) representam a média das soluções encontradas pela heurística e o tempo total de execução da heurística, respectivamente.

Os resultados da Tabela 2 mostram que as heurísticas HC1 e HC2 apresentaram melhores valores, em média, de tempos de ciclo em relação às soluções obtidas pelas heurísticas HC3 e HC4, apesar dos resultados não estarem tão distantes entre si. O tempo de execução das heurísticas pode ser considerado igual, já que em média, para todas as famílias, cada heurística gerou uma solução em um tempo menor que duas casas decimais depois de zero, o que permite dizer que todos os algoritmos são executados de forma bastante eficiente. Para se ter uma idéia da relevância deste fato, [Chaves 2009] utiliza o *software* comercial CPLEX versão 10.1 para encontrar limitantes para as 320 instâncias. Devido à natureza NP-Difícil, o CPLEX encontrou grandes dificuldades para resolver as instância maiores. Para as instâncias Tonge, o CPLEX termina a execução em razão do estouro de memória do computador em 79 das 80 instâncias testadas. O mesmo problema ocorre nas instâncias da família Wee-Mag, só que em 68 das 80 instâncias.

A fim de comparar o desempenho destas abordagens como solução inicial de uma metaheurística, foram testadas cada uma das soluções encontradas pelas heurísticas, tanto factíveis ou não, como ponto de partida do algoritmo de busca tabu minimalista (BT) proposto em [Moreira and Costa 2009].

A Tabela 3 apresenta a comparação das médias dos tempos de ciclo para cada família de instância do algoritmo de busca tabu minimalista iniciando com cada uma das soluções obtidas pelas heurística propostas (BT-H1, BT-H2, BT-H3 e BT-H4). As colunas S^*, $T^*(s)$ e T(s) representam a média das melhores soluções encontradas para cada família de instâncias, média do tempo de encontrar as melhores soluções e tempo total de execução dos algoritmos, respectivamente. Nota-se que os quatro algoritmos apresentaram desempenhos similares para as classes Heskia e Roszieg e que as heurísticas HC1 e HC2 tenderam a ser melhores para as classes de instâncias maiores (Tonge e Wee-mag), sendo que o algoritmo iniciado com HC1 foi ligeiramente superior.

4. Conclusões

Neste trabalho, propomos quatro heurísticas construtivas para um problema de balanceamento de linhas de produção que surge em linhas de produção com trabalhadores defi-

	BT-H1			BT-H2			BT-H3			BT-H4		
Família	S^*	$T^*(s)$	T(s)	S^*	$T^*(s)$	T(s)	S^*	$T^*(s)$	T(s)	S^*	$T^*(s)$	T(s)
Roszieg	19,95	0,15	2,67	19,94	0,14	2,55	19,95	0,21	2,54	19,95	0,14	2,54
Heskia	99,63	0,62	3,69	99,94	0,64	3,61	99,63	0,52	3,55	99,64	0,62	3,71
Tonge	109,56	12,56	33,56	108,36	13,76	33,55	110,36	12,69	33,49	110,46	12,92	33,55
Wee-mag	36,58	6,19	41,37	36,34	6,33	40,57	39,21	9,99	41,60	38,28	11,91	43,43

Tabela 3. Resultados para todas as famílias de instâncias

cientes. As heurísticas diferiam entre si nos critérios que definiam a ordem de escolha das tarefas e foram comparadas tanto em termos da qualidade das soluções obtidas, como em termos da eficácia das soluções construtivas de melhorarem o desempenho de um algoritmo mais sofisticado. Os resultados indicaram um bom desempenho dos métodos propostos nas duas situações.

Referências

Chaves, A. (2009). *Uma meta-heurística híbrida com busca por agrupamentos aplicada a problemas de otimização combinatória.* PhD thesis, Instituto Nacional de Pesquisas Espaciais - INPE.

Chaves, A. A., Lorena, L., and Miralles, C. (2009). Hybrid metaheuristic for the assembly line worker assignment and balancing problem. *Lecture Notes on Computer Science*, 5818:1–14.

Chaves, A. A., Miralles, C., and Lorena, L. A. N. (2007). Clustering search approach for the assembly line worker assignment and balancing problem. In *Proceedings of the 37th International Conference on Computers and Industrial Engineering, Alexandria, Egypt*, pages 1469–1478.

Miralles, C., Garcia-Sabater, J. P., Andres, C., and Cardos, M. (2007). Advantages of assembly lines in sheltered work centres for disabled: a case study. *International Journal of Production Economics*, 110:187–197.

Miralles, C., Garcia-Sabater, J. P., Andrés, C., and Cardos, M. (2008). Branch and bound procedures for solving the assembly line worker assignment and balancing problem: Application to sheltered work centres for disabled. *Discrete Applied Mathematics*, 156:352–367.

Moreira, M. and Costa, A. (2009). A minimalist yet efficient tabu search for balancing assembly lines with disabled workers. In *Anais do XLI Simpósio Brasileiro de Pesquisa Operacional.*

Scholl, A. and Voß, S. (1996). Simple assembly line balancing - heuristic approaches. *Journal of Heuristics*, 2:217–244.

SERPRO (2004). http://www.serpro.gov.br/noticias-antigas/noticias-2004/20041221_02 (Consultado em 23 de Março de 2009).

Talbot, F. B., Patterson, J. H., and Gehrlein, W. V. (1986). A comparative evaluation of heuristic line balancing techniques. *Management Science*, 32:430–454.