PROBLEMA DE BALANCEAMENTO E DESIGNAÇÃO DE TRABALHADORES EM LINHA DE PRODUÇÃO: UMA ABORDAGEM VIA O MÉTODO HÍBRIDO BUSCA POR AGRUPAMENTOS (CS)

Antonio Augusto Chaves
Instituto Nacional de Pesquisas Espaciais (INPE)
Av. dos Astronautas, 1758 São José dos Campos – SP – Brasil
chaves@lac.inpe.br

Cristóbal Miralles
ROGLE-Dpto. Organización de Empresas. Universidad Politécnica de Valencia (UPV)
Camí Vera s/n, 46022, Valencia – Espanha
cmiralles@omp.upv.es

Luiz Antonio Nogueira Lorena
Instituto Nacional de Pesquisas Espaciais (INPE)
Av. dos Astronautas, 1758 São José dos Campos – SP – Brasil
lorena@lac.inpe.br

RESUMO

Este trabalho aborda o Problema de Balanceamento e Designação de Trabalhadores em Linha de Produção (ALWABP), que consiste em otimizar a alocação de trabalhadores e tarefas às estações de trabalho em uma linha de produção com pessoas deficientes. O objetivo é aumentar a produtividade de tais linhas e tornar as deficiências individuais tão transparentes quanto possível. Para tal é proposto resolver esse problema de forma heurística por meio do método híbrido Busca por Agrupamentos (CS). Neste trabalho também são propostas algumas melhorias para o CS, tornando-o mais eficiente e robusto. Os testes computacionais mostram o potencial do CS para resolução do ALWABP. Conclusões a respeito desse trabalho também são apresentadas.

PALAVRAS CHAVE. Meta-heurística, Problema de Balanceamento de Linha de Produção, Busca por Agrupamentos. Área principal (Meta-heurística).

ABSTRACT

This paper approaches the Assembly Line Worker Assignment and Balancing Problem (ALWABP), which consists of optimizing the allocation of workers and tasks to stations in an assembly line with disabled people. The goal is to increase the productivity of the assembly line but always respecting the worker limitations; trying to make transparent the individual disabilities. This paper proposes to solve this problem by the hybrid method Clustering Search (CS). It is also proposed some improvements for the CS, becoming it more efficient and robust. The computational tests show the potential of CS to solve the ALWABP. Conclusions regarding this paper are also presented.

KEYWORDS. Metaheuristic, Assembly Line Balancing Problem, Clustering Search. Main area (Metaheuristic).

1. Introdução

A Organização Mundial de Saúde (OMS) estima que 10% da população mundial, cerca de 610 milhões de pessoas, são portadores de algum tipo de deficiência. Destas, mais da metade estão na idade ativa de trabalho e sofrem com as altas taxas de desemprego.

No Brasil, o Censo 2000 divulgou que 14,5% da população são portadores de algum tipo de deficiência, o que corresponde a 24,6 milhões de pessoas. Das quais 15,14 milhões têm idade e condições de integrarem o mercado formal de trabalho. Contudo, apesar de não haver estudos sobre a proporção de pessoas com deficiência que exercem alguma atividade produtiva, Organizações Não-Governamentais (ONG's), que trabalham com a inclusão social desses cidadãos, estimam que a taxa de desemprego entre os deficientes é muito alta no Brasil.

Visando uma maior inclusão dos deficientes ao mercado de trabalho, desde 1991 existe no Brasil uma lei que obriga as empresas com mais de 100 funcionários a contratarem pessoas portadoras de deficiências (art. 93 da Lei n. 8.213/91). A lei prevê que uma determinada quantidade de vagas, que varia de 2% a 5% do número total de funcionários, deve ser reservada para pessoas deficientes. Apesar disso, a integração de pessoas com deficiências à sociedade e, em particular, ao mercado de trabalho, encontra ainda várias dificuldades e esbarra diversas vezes em discriminações do passado (Jaime e Carmo, 2005).

Como forma de facilitar a inclusão desses trabalhadores no mercado de trabalho, alguns países (por exemplo, Espanha e Japão) adotaram a estratégia de criar Centros de Trabalho para Deficientes (CTD's). Esses centros funcionam como uma primeira etapa na integração dessas pessoas que, eventualmente, serão absorvidas pelo mercado formal de trabalho.

Esse modelo de integração sócio-trabalho procura afastar-se do estereótipo tradicional que considera as pessoas deficientes como incapazes de desenvolver um trabalho profissional contínuo. Da mesma maneira que em qualquer outra empresa, um CTD compete em mercados reais e precisa ser suficientemente flexível e eficiente para adaptar-se às variações do mercado, sendo que a única diferença é que um CTD é uma organização sem fins lucrativos.

Sendo assim, o benefício que geralmente pode ser obtido com a melhora de eficiência é o crescimento do CTD, ou seja, mais trabalhos para as pessoas deficientes, gradualmente integrando pessoas com níveis mais elevados de inaptidão, o que é de fato o objetivo principal de todo CTD.

Em Miralles et al. (2007) é mostrado como a utilização de linhas de produção nesses centros provê muitas vantagens, sendo que, a divisão tradicional do trabalho em tarefas únicas pode se tornar uma ferramenta perfeita para fazer invisíveis determinadas deficiências do trabalhador. Além disso, uma atribuição apropriada da tarefa também pode transformar-se em um bom método terapêutico para reabilitação das deficiências. Mas algumas restrições específicas relacionadas à variabilidade dos tempos surgem neste ambiente e, então, o procedimento de balanceamento aplicado deveria conciliar os seguintes objetivos:

(1) maximizar a eficiência da linha de produção balanceando a carga de trabalho atribuída a cada trabalhador disponível em cada estação;

(2) satisfazer e respeitar as restrições existentes nesse ambiente devido aos fatores humanos ao atribuir tarefas aos trabalhadores.

Após analisar alguns CTD's, Miralles et al. (2007) estabeleceram algumas características específicas que podem ser encontradas nesses ambientes, as quais serviram de motivação para definir um novo problema de linha de produção, chamado Problema de Balanceamento e Designação de Trabalhadores em Linhas de Produção (ALWABP, do inglês *Assembly Line Worker Assignment and Balancing Problem*).

O ALWABP pode ser classificado como um problema NP-*hard*, pois este é uma generalização do Problema Simples de Balanceamento da Linha de Produção (SALBP, do inglês *Simple Assembly Line Balancing Problem*) (Gutjahr e Nemhauser, 1964), no qual cada tarefa possui um tempo de execução fixo independente do trabalhador que a executa. E, o SALBP é classificado como NP-*hard* (Scholl e Becker, 2006). Portanto, a aplicação de métodos heurísticos é uma boa alternativa para obter bons resultados em pouco tempo computacional.

Em vista disso, este trabalho apresenta uma aplicação do método híbrido Busca por Agrupamentos (CS, do inglês *Clustering Search*) (Oliveira e Lorena, 2007) para resolver de forma aproximada o ALWABP. O CS visa detectar regiões promissoras no espaço de busca por meio de um processo de agrupamento de soluções geradas por uma meta-heurística. Essas regiões são exploradas tão logo sejam descobertas, por meio de heurísticas de busca local. A meta-heurística Busca Local Iterativa (ILS, do inglês *Iterated Local Search*) (Lourenço et al., 2003) é utilizada para gerar soluções para o processo de agrupamento do CS. Neste trabalho também são propostas algumas melhorias para o CS.

O restante do trabalho está organizado da seguinte forma. Na seção 2 tem-se uma descrição do ALWABP e uma revisão bibliográfica deste problema. A seção 3 descreve o método CS, e a seção 4 apresenta o CS aplicado ao ALWABP. A seção 5 apresenta os resultados computacionais. Na seção 6 são descritas algumas conclusões a respeito deste trabalho.

2. Definição e revisão bibliográfica do ALWABP

De uma forma geral, uma linha de produção consiste de um conjunto finito de tarefas, cada uma tendo um tempo de execução, e um conjunto de relações de precedência, as quais especificam a ordem de execução das tarefas. O problema do ALWABP é atribuir as tarefas para uma sequência de estações, tal que, as relações de precedências sejam satisfeitas e alguma medida de eficiência seja otimizada. Entretanto, como nos CTD's alguns trabalhadores podem ser muito lentos para executar certas tarefas ou até incapazes, e muito rápidos na execução de outras, o problema não consiste apenas em atribuir tarefas para estações, mas também trabalhadores disponíveis para estações, sempre respeitando as incompatibilidades quando atribuir tarefas para trabalhadores.

As principais características do ALWABP são:

- os tempos de processamento das tarefas e as relações de precedência são definidos previamente;
- existe um dado número de trabalhadores disponíveis, e o tempo de processamento das tarefas pode ser diferente dependendo de qual trabalhador executa a tarefa (uma vez que os trabalhadores têm diferentes habilidades e capacidades);
- não existem trabalhadores lentos ou rápidos. Ao invés disso, trabalhadores podem ser muito lentos, ou até incapazes, de executar algumas tarefas, mas muito eficientes quando executam outras tarefas;
- todo trabalhador é atribuído para somente uma estação de trabalho;
- toda tarefa é atribuída para somente uma estação de trabalho, contanto que o trabalhador selecionado para aquela estação seja capaz de realizar a tarefa, e que as relações de precedência sejam satisfeitas.

Analogamente ao SALBP, quando se deseja minimizar o número de estações, o problema é chamado ALWABP-1, e quando o objetivo é minimizar o tempo de ciclo (ou maximizar a taxa de produção), o problema é chamado ALWABP-2. Esta última é a situação mais comum nos CTD's, uma vez que, visa o aumento da eficiência da produção sem retirar nenhum posto de trabalho existente.

A situação mais típica em um CTD é ter um dado número de trabalhadores disponíveis (cada um deles com tempos de operação definidos para cada tarefa) e onde a eficiência da linha de produção deve ser maximizada. O tempo de ciclo (C_{tempo}) representa a quantidade de tempo que um produto pode gastar para ser processado por uma estação da linha de produção, tendo relação direta com a taxa de produção da linha. Portanto, maximizar a eficiência significa, neste ambiente, minimizar o tempo de ciclo.

Ao nosso conhecimento, apenas muito recentemente o problema encontrado nos CTD's começou a ser tratado na literatura. A maioria dos problemas sobre linha de produção encontrados na literatura trabalha com tempos de operação fixos, independentemente de qual trabalhador executa a tarefa. Essa simplificação é justificada pelo fato de que, na maioria dos casos de linhas de produção reais, a variação de tempo entre os trabalhadores é muito pequena.

Os primeiros estudos sobre as vantagens de utilizar linhas de produção nos CTD's foram realizados por Miralles et al. (2007), que introduziram o ALWABP e desenvolveram uma formulação matemática para o problema. Os autores estudaram um caso real de um CTD situado na cidade de Valência (Espanha), mostrando que a Pesquisa Operacional pode abranger, além de objetivos econômicos e produtivos, objetivos sociais. Posteriormente, Miralles et al. (2008) implementaram um algoritmo *Branch* e *Bound* com diferentes estratégias de busca para resolver o ALWABP.

Chaves et al. (2007 e 2008) propuseram a utilização do método híbrido CS para resolver de forma aproximada o ALWABP, no qual foi implementada a meta-heurística Recozimento Simulado (SA, do inglês *Simulated Annealing*) (Kirkpatrick et al., 1983) para gerar soluções para o processo de agrupamento. Neste trabalho também foi resolvida uma formulação matemática do ALWABP por meio de *software* CPLEX 10.

Costa e Miralles (2008) apresentam uma variante para o ALWABP, no qual é estudado como programar a rotação de tarefas neste problema. A rotação pode trazer diversos benefícios, tais como, aumentar a motivação dos trabalhadores, combater certas doenças do trabalho e, principalmente, auxiliar o tratamento terapêutico dos trabalhadores. Os autores propuseram uma formulação de programação inteira mista e um método de decomposição heurístico para resolução desse problema.

3. Busca por Agrupamentos (CS)

Busca por Agrupamentos (CS) é um método híbrido que objetiva combinar adequadamente meta-heurísticas e heurísticas de busca local, no qual a busca é intensificada somente em regiões do espaço de busca que tenham um potencial de melhoria da solução. O CS acrescenta uma "inteligência" para auxiliar a escolha de em quais soluções aplicar busca local, ao invés de escolher aleatoriamente ou aplicar busca local em todas as soluções. Assim, espera-se uma melhoria no processo de convergência do CS associada a uma diminuição no esforço computacional em virtude do emprego mais direcionado dos métodos de busca local.

Neste trabalho propõe-se algumas melhorias para o método CS, tornando-o mais eficiente, robusto e amigável. Para tal, foi adicionada uma perturbação aleatória e uma variável para controlar a eficiência do método de busca local. Além disso, define-se com maior clareza as funções de cada componente do CS, redesenhando sua estrutura. Essas modificações propostas fazem do CS um método de aprendizado e uso fáceis e intuitivos.

O CS procura dividir o espaço de busca e localizar regiões promissoras por meio do enquadramento destas em *clusters*. Para os fins do CS, um *cluster* pode ser definido por três atributos $C = (c, v, r)$. O centro c_i é uma solução que representa o *cluster* C_i, identificando a sua localização dentro do espaço de busca. Ao invés de armazenar todas as soluções agrupadas no *cluster*, apenas parte das características destas soluções são inseridas no centro do *cluster*. O volume v_i é a quantidade de soluções agrupadas no *cluster* C_i. Um *cluster* se torna promissor quando o volume atinge certo limitante λ, definido *a priori*. O índice de ineficácia r_i é uma variável de controle para identificar se a busca local está ou não melhorando o centro do *cluster* C_i. O valor de r_i indica o número de vezes consecutivas que a busca local foi aplicada no *cluster* C_i e não melhorou a solução. Este atributo tem como objetivo evitar que o método de busca local fique sendo executado por mais de r_{max} vezes em regiões ruins ou regiões que já tenham sido suficientemente exploradas por este em iterações anteriores.

Para que o algoritmo possa agrupar soluções em *clusters* é necessário definir alguma forma de medir a distância entre duas soluções. Sendo assim, uma função de medida de distância $d(i, j)$ é definida, *a priori*, para calcular a distância entre duas soluções como um número positivo, o qual é maior dependendo de quão mais distante estão as duas soluções.

O CS é um método iterativo que possui três componentes principais: uma meta-heurística, um processo de agrupamento e um método de busca local. A estratégia híbrida do CS pode ser descrita pelo fluxograma ilustrado na Figura 1.

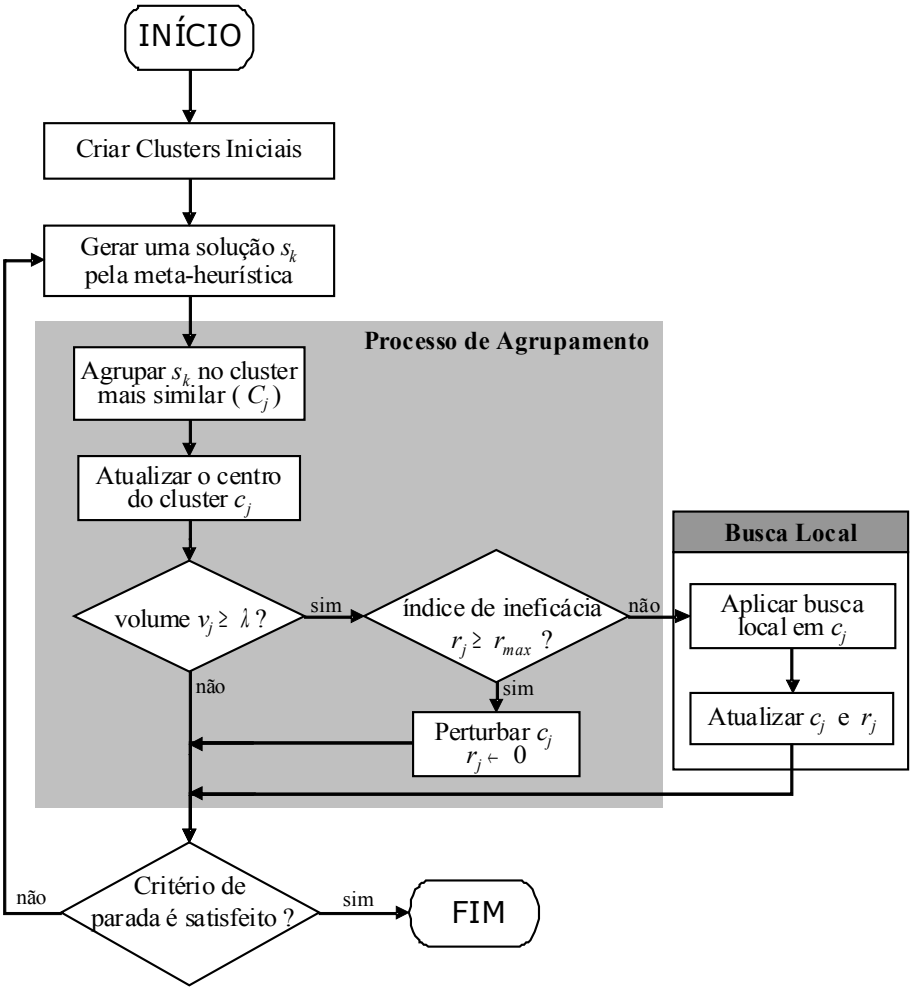

Figura 1 – Fluxograma do CS

Primeiramente, é necessário definir o número de *clusters* do CS, ou seja, a quantidade de regiões na qual o espaço de busca será dividido. Os centros dos *clusters* devem ser gerados de forma a representar diferentes regiões do espaço de busca. Para tal propõe-se um método baseado na diversidade máxima, no qual são gerados *n* soluções aleatórias e selecionam-se as *m* soluções mais distantes entre si. Estas soluções serão os centros dos *clusters* iniciais.

A meta-heurística é um gerador de soluções para o processo de agrupamento. Pode-se utilizar qualquer meta-heurística. Porém, é importante que esta seja capaz de gerar um grande número de soluções diferentes, possibilitando uma análise ampla do espaço de busca.

A cada iteração do CS, uma solução s_k é gerada pela meta-heurística e enviada para o processo de agrupamento, que objetiva reunir soluções similares dentro de *clusters* e direcionar a busca para regiões supostamente promissoras. Sendo assim, a solução s_k é agrupada no *cluster* mais similar C_j, que é o *cluster* com a menor distância entre centro c_j e a solução s_k.

A "inserção" de uma nova solução em um *cluster* deve causar uma perturbação em seu centro. Tal perturbação é chamada de assimilação e consiste basicamente em atualizar o centro com características da solução s_k. Neste processo é utilizado o método Reconexão por Caminhos (PR, do inglês *Path-Relinking*) (Glover, 1996), que realiza movimentos exploratórios na trajetória que interconecta o centro c_j e a solução s_k. A melhor solução neste caminho será o novo centro do *cluster*.

Em seguida é analisado o volume v_j do *cluster*. Caso esse volume tenha atingido o limitante λ, esse *cluster* pode estar em uma região de busca promissora. Porém, se o método de busca local não tiver obtido sucesso nas últimas r_{max} aplicações neste *cluster* promissor (índice de ineficácia $r_j \geq r_{max}$) é aplicada uma perturbação aleatória no centro c_j, objetivando escapar desta região do

espaço de busca. Por outro lado, se r_j for menor que r_{max}, uma busca local é aplicada no centro c_j intensificando a busca na vizinhança deste *cluster*. A busca obtém sucesso em um *cluster* quando encontra uma solução que seja a melhor obtida neste *cluster* até o momento (c_j*). Encerrado o processo de agrupamento, retorna-se para a meta-heurística que irá gerar outra solução.

O componente de busca local é um módulo de busca que provê a exploração de uma suposta região promissora, delimitada pelo *cluster*, por meio de heurísticas de busca local específicas para o problema abordado. O método Descida em Vizinhança Variável (VND, do inglês *Variable Neighborhood Descent*) (Mladenovic e Hansen, 1997) pode ser utilizado neste componente, permitindo analisar um número maior de soluções ao redor do *cluster*.

4. CS Aplicado ao ALWABP

Uma solução do ALWABP é composta por dois vetores. O primeiro vetor representa a alocação tarefa/estação. O segundo vetor representa a alocação trabalhador/estação. A Figura 2 ilustra a representação de uma solução com 11 tarefas, 5 trabalhadores e 5 estações.

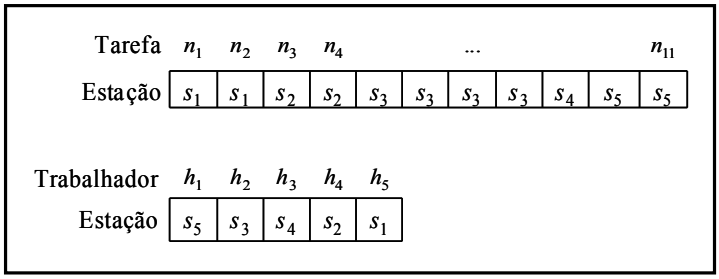

Figura 2 - Representação de uma solução do ALWABP

O cálculo da função objetivo do ALWABP computa os tempos de operação de cada estação, considerando as tarefas e o trabalhador alocados na estação. A função objetivo corresponde ao tempo de ciclo da linha de produção (C_{tempo}), que é o maior tempo de operação das estações. Soluções que não satisfaçam as restrições da rede de precedências ou aloquem alguma tarefa incompatível com um trabalhador precisam ser penalizadas. Desta forma, uma solução é avaliada pela seguinte função objetivo:

$$f(s) = C_{tempo} + \sigma * f_r(s) + \varpi * f_t(s) \quad (1)$$

na qual, os componentes f_r e f_t mensuram, respectivamente, a inviabilidade da rede de precedências e da alocação tarefa/trabalhador incompatíveis. σ e ϖ são os pesos que refletem a importância relativa de cada um dos componentes de *f*. Neste trabalho definiu-se empiricamente σ = 1000 e ϖ = 500, privilegiando as soluções que satisfaçam as restrições da rede de precedências, pois essas inviabilidades são mais difíceis de serem eliminadas.

A medida de distância $d(i, j)$ é definida como o número de tarefas atribuídas em estações diferentes entre duas soluções. Portanto, quanto maior o número de tarefas alocadas em estações diferentes entre duas soluções, maior será a distância entre elas e menor será a similaridade.

Neste trabalho propõe-se um método para gerar uma solução inicial sem violar as precedências entre tarefas. Para tal, esse método leva em consideração o número de tarefas que precedem uma determinada tarefa. Desta forma, constrói-se uma lista em ordem crescente das tarefas com menor número de tarefas antecessoras. No caso de duas ou mais tarefas com o mesmo número de antecessoras, a ordem é escolhida aleatoriamente. As estações são preenchidas da primeira para a última com as tarefas seguindo a ordem da lista construída, procurando alocar o mesmo número de tarefas a todas as estações. Após alocar as tarefas, os trabalhadores são alocados aleatoriamente às estações. Sendo assim, a solução construída tende a ter somente inviabilidades entre trabalhadores e tarefas incompatíveis.

4.1 Meta-heurística

Neste trabalho optou-se pela meta-heurística Busca Local Iterativa (ILS) que consiste em um processo iterativo, no qual uma solução é perturbada, gerando novas soluções de partida para um método de busca local.

O algoritmo ILS inicializa gerando uma solução inicial para o ALWABP (s_0). Em seguida, é aplicada uma heurística de busca local obtendo uma solução ótima local ($\hat{s}$). A partir dessa solução inicializa-se um processo iterativo até que o critério de parada seja satisfeito. A cada iteração realiza-se um procedimento de perturbação aleatória da solução $\hat{s}$ produzindo um novo ponto de partida para a busca local (s'). A heurística de busca local é então aplicada sobre s' encontrando a solução ótima local desta região ($\hat{s}'$).

Para realizar a perturbação são definidos três movimentos:

- m_1: trocar a alocação de dois trabalhadores de estações;
- m_2: trocar a alocação de duas tarefas de estações diferentes;
- m_3: transferir uma tarefa de uma estação para outra estação.

A intensidade da perturbação varia a cada iteração. Uma porcentagem β relacionada com o número de tarefas do problema tratado define a quantidade de movimentos realizados. A porcentagem β varia de 25% a 75%. Os três movimentos são executados aleatoriamente.

As heurísticas Trocar Trabalhador e Trocar Tarefa foram aplicadas para obter um ótimo local para a solução s'. Essas heurísticas serão explicadas na seção 3.3, pois elas também foram utilizadas no componente de busca local do CS. A diferença é que no ILS estas heurísticas foram implementadas baseadas no conceito de *first improvement*, ou seja, sempre que uma solução melhor é encontrada esta passa a ser a solução corrente e continua-se a busca a partir desta.

O critério de aceitação irá determinar se $\hat{s}'$ é aceita ou não como a nova solução corrente, aceitando-a quando esta for melhor que a solução $\hat{s}$. Com objetivo de favorecer a diversificação, foi adotada uma probabilidade de 5% de aceitar um ótimo local que piore a solução corrente.

O critério de parada do CS foi definido como o número máximo de iterações do ILS, gerando 5000 soluções para o processo de agrupamento.

4.2 Processo de Agrupamento

As soluções $\hat{s}'$ geradas são enviadas para o processo de agrupamento que procura agrupá-las como uma solução conhecida, de acordo com a métrica de distância *d*. O *cluster* que possuir a menor distância em relação à solução $\hat{s}'$ é considerado o *cluster* mais similar (C_j). O centro deste *cluster* deve ser atualizado (assimilação) com novas características contidas na solução $\hat{s}'$.

No processo de assimilação foi utilizado o método Reconexão por Caminho (PR), o qual parte do centro c_j em direção à solução $\hat{s}'$, analisando o caminho que interconecta essas soluções. Para gerar esses caminhos, movimentos são selecionados por meio da troca da alocação de uma tarefa *i* no centro c_j pela alocação da tarefa *i* na solução $\hat{s}'$, mudando a alocação tarefa/estação da solução corrente. O método termina quando 30% do caminho for analisado. A melhor solução encontrada no caminho será o novo centro c_j. Tal estratégia permite que o centro se desloque no espaço de busca, mas evita que este se mova para regiões muito distantes da atual.

Após atualizar o centro c_j, é preciso conduzir uma análise do volume v_j, verificando se este *cluster* pode ser considerado promissor, ou seja, se o número de soluções agrupadas atingiu o limitante λ ($v_j \geq \lambda$). Neste trabalho utilizou-se $\lambda = 10$.

Então, se o volume v_j atingir λ e a heurística de busca local estiver obtendo sucesso nesse *cluster* ($r_j < r_{max}$; $r_{max} = 5$), uma intensificação é aplicada no centro do *cluster* por meio das heurísticas de busca local. Caso contrário, se o índice de ineficácia $r_j \geq 5$, é necessário aplicar uma perturbação no centro c_j, trocando aleatoriamente a alocação de 30% das tarefas.

4.3 Heurísticas de Busca Local

A busca local é ativada quando um *cluster* promissor é identificado, intensificando a busca na vizinhança do *cluster* por meio do método Descida em Vizinhança Variável (VND).

Os três tipos de movimentos definidos anteriormente são relevantes em um método de busca local para o ALWABP. Sendo assim, o VND utiliza três heurísticas baseadas nesses movimentos. As heurísticas do VND são:

[1] *Trocar Tarefa*: realizar o melhor movimento de trocar duas tarefas que estão atribuídas a diferentes estações;

[2] *Transferir Tarefa*: realizar o melhor movimento de transferir uma tarefa de uma estação para outra;

[3] *Trocar Trabalhador*: realizar o melhor movimento de trocar a posição de dois trabalhadores.

As heurísticas são executadas em sequência e se uma solução melhor for obtida, o VND retorna para a primeira heurística continuando a busca da melhor solução. A condição de parada do VND é que não existam mais melhoras para a solução corrente. O centro do *cluster* é atualizado se a solução encontrada pelo VND for melhor que o centro atual. Mas, o índice de ineficácia só será zerado caso o VND encontre uma solução que seja a melhor até o momento neste *cluster*.

O pseudocódigo do CS aplicado ao ALWABP é apresentado na Figura 3.

```
algoritmo CS
    crie os clusters iniciais
    { meta-heurística – ILS}
    gere uma solução inicial s_0
    ŝ ← BuscaLocal ( s_0 )
    enquanto ( critério de parada não for satisfeito ) faça
        s' ← Perturbação ( ŝ )
        ŝ' ← BuscaLocal ( s' )
        ŝ ← CritériodeAceitação ( ŝ , ŝ' )
        { processo de agrupamento }
        encontre o cluster mais similar a ŝ' ( C_j | d( ŝ',c_j) ≡ min { d( ŝ',c) } )
        agrupe ŝ' no cluster mais similar C_j ( v_j ← v_j + 1 )
        atualize o centro do cluster ( c_j ← PR (c_j, ŝ') )
        se ( v_j ≥ λ ) então
            reduza o volume v_j ← 0
            se ( r_j ? r_max ) então
                aplique uma perturbação aleatória em c_j
                zere o índice de ineficácia r_j ← 0
            senão
                { busca local }
                ĉ_j = VND ( c_j )
                se ( f( ĉ_j ) < f( c_j ) ) então
                    atualize o centro c_j ← ĉ_j
                se ( f( ĉ_j ) < f( c*_j ) ) então
                    zere o índice de ineficácia r_j ← 0
                    atualize o melhor centro c*_j ← ĉ_j
                senão
                    aumente o índice de ineficácia r_j ← r_j + 1
            fim-se
        fim-se
    fim-enquanto
fim-algoritmo
```

Figura 3 – Pseudocódigo do CS

5. Resultados Computacionais

O CS para o ALWABP foi codificado em C++ e os testes computacionais foram executados em um PC com processador Pentium 4 2,6 GHz e memória de 1 GB. Os experimentos foram empreendidos com o objetivo de evidenciar a qualidade dos resultados do CS, mostrando que este método pode ser competitivo para resolver o ALWABP.

Quatro conjuntos de problemas foram usados nos testes: *Roszieg*, *Heskia*, *Tonge* e *Wee-Mag*. As instâncias desses problemas foram propostas em Chaves et al. (2007) e são baseadas em instâncias de uma coleção de problemas clássicos do SALBP (Hoffmann, 1990). Nestes conjuntos têm-se um total de 320 instâncias para o ALWABP, que permitem extrair conclusões sobre o comportamento do CS diante de diferentes tipos de variações encontradas no ALWABP. Essas instâncias estão disponíveis em http://www.lac.inpe.br/~lorena/intancias.html.

As instâncias testadas levam em consideração cinco fatores: número de tarefas (*Tam*); complexidade da rede de precedência (*OS*); número de trabalhadores (*NrT*); variabilidade dos tempos de operação (*Var*); e porcentagem de incompatibilidades tarefa/trabalhador (*Inc*).

As tabelas a seguir apresentam os resultados dos testes computacionais realizados neste trabalho. Em razão do número elevado de instâncias, optou-se por apresentar os valores médios de cada 10 instâncias com as mesmas características.

A Tabela 1 apresenta os resultados do CS aplicado ao ALWABP, e possui as seguintes colunas:

- ***melhor***: melhor solução conhecida;
- ***sol****: melhor solução encontrada pelo método;
- ***sol***: solução média encontrada em 20 execuções do método;
- ***gap***: erro relativo entre a melhor solução do método e a melhor solução conhecida;
- ***T****: tempo médio gasto pelo método para encontrar a melhor solução (em segundos);
- ***T***: tempo médio de execução do método (em segundos);
- ***delta***: erro relativo entre a melhor solução do CS e a melhor solução do ILS.

Chaves et al. (2008) encontraram as soluções ótimas para as instâncias *Roszieg* e *Heskia* por meio do *software* CPLEX. Para as instâncias *Tonge* e *Wee-Mag* não foi possível obter as soluções ótimas devido ao estouro de memória do computador, e os *gaps* entre os limitantes inferior e superior ficaram acima de 90%. Observamos, entretanto, que as soluções obtidas pelo CS são muito melhores que as obtidas pelo CPLEX para estas instâncias.

Observando a Tabela 1 nota-se que o CS encontrou a solução ótima para todas as instâncias *Roszieg* e *Heskia*, Para as instâncias *Tonge* e *Wee-Mag*, o CS encontrou a melhor solução conhecida para a maioria das instâncias, tendo *gaps* bem próximos de zero. Por meio da solução média é possível perceber que o CS se mostrou um método robusto, uma vez que, o CS apresenta valores de solução média próximos à melhor solução encontrada.

Outra observação que pode ser feita sobre os dados dessa tabela diz respeito à melhora produzida pela utilização do processo de agrupamento. Por meio da coluna *delta*, pode-se notar que o CS obtém uma melhora significativa, principalmente para as instâncias *Tonge* e *Wee-Mag*, em relação às soluções encontradas pela meta-heurística ILS.

Os tempos computacionais do CS foram competitivos, sendo possível obter boas soluções em alguns segundos de execução do método. Em média, o CS converge para a melhor solução em aproximadamente 60% do tempo total de execução. Esses dados mostram que o processo de agrupamento do CS melhora o processo de convergência da meta-heurística.

A Tabela 2 compara os resultados do CS apresentado neste trabalho com a versão deste método apresentada em Chaves et al. (2008), ressaltando que foram propostas melhorias para o CS e também a troca da meta-heurística utilizada para gerar soluções para o processo de agrupamento. Chaves et al. (2008) utilizaram Recozimento Simulado (SA) e neste trabalho é utilizada Busca Local Iterativa (ILS).

Os resultados obtidos neste trabalho com o método CS são melhores em termos de qualidade da solução se comparados com os obtidos em Chaves et al. (2008), obtendo uma melhora de 16,78% (coluna *gap*). Além disso, esse método apresentou soluções médias próximas das melhores soluções encontradas, diferentemente do ocorrido em Chaves et al. (2008).

Tabela 1 – ALWABP: Resultados Computacionais do CS

					CS					ILS				
	NrT	Var	Inc	melhor	sol*	sol	gap	T*	T	sol*	sol	gap	T	delta
Roszieg (25 tarefas)	4	L1	I10	20,1	20,1	20,2	0,0	0,8	3,8	20,1	20,4	0,0	3,4	0,0
			I20	31,5	31,5	32,5	0,0	0,9	3,7	32,4	38,9	3,5	3,2	3,5
		H3	I10	28,1	28,1	28,5	0,0	0,6	3,8	28,2	28,7	0,5	3,3	0,5
			I20	28,0	28,0	28,0	0,0	0,2	3,8	28,0	28,1	0,0	3,3	0,0
	6	L1	I10	9,7	9,7	10,7	0,0	1,3	5,2	10,3	11,8	6,3	4,6	6,3
			I20	11,0	11,0	12,1	0,0	1,4	5,2	11,5	14,3	4,9	4,5	4,9
		H3	I10	16,0	16,0	16,9	0,0	1,5	5,2	16,5	18,7	3,2	4,6	3,2
			I20	15,1	15,1	15,6	0,0	1,9	5,2	15,3	17,7	1,4	4,6	1,4
Heskia (28 tarefas)	4	L1	I10	102,3	102,3	102,8	0,0	1,3	5,0	102,3	103,0	0,0	6,3	0,0
			I20	122,6	122,6	123,8	0,0	1,4	5,0	122,7	124,2	0,1	6,3	0,1
		H3	I10	172,5	172,5	175,5	0,0	1,7	5,0	172,6	176,4	0,1	6,4	0,1
			I20	171,2	171,2	171,7	0,0	1,4	5,1	171,3	171,8	0,0	6,4	0,0
	7	L1	I10	34,9	34,9	37,8	0,0	4,4	9,0	35,3	38,6	1,2	7,5	1,2
			I20	42,6	42,6	44,7	0,0	3,4	9,0	43,6	45,7	2,5	7,5	2,5
		H3	I10	75,2	75,2	77,7	0,0	2,9	9,0	76,7	78,6	2,3	7,5	2,3
			I20	67,2	67,2	70,7	0,0	3,6	9,0	68,1	72,4	1,3	7,5	1,3
Tonge (70 tarefas)	10	L1	I10	96,7	96,7	116,6	0,0	64,0	122,2	120,0	135,3	24,3	102,5	24,3
			I20	116,0	116,0	141,8	0,0	64,6	122,6	151,8	174,8	31,2	101,7	31,2
		H3	I10	167,1	167,9	199,4	0,5	66,2	122,8	214,6	236,5	28,7	102,6	28,1
			I20	174,0	174,0	206,0	0,0	65,7	123,0	220,7	244,3	27,1	102,6	27,1
	17	L1	I10	42,6	42,6	51,8	0,0	101,1	183,0	64,2	71,1	50,9	151,1	50,9
			I20	48,6	48,6	61,6	0,0	105,3	184,7	74,2	87,4	53,1	151,4	53,1
		H3	I10	78,6	78,6	94,0	0,0	100,1	184,5	113,7	129,3	47,0	151,4	47,0
			I20	78,2	78,6	95,6	0,6	100,3	184,9	116,1	131,3	49,4	151,5	48,6
Wee-Mag (75 tarefas)	11	L1	I10	29,0	29,0	32,7	0,0	94,3	163,2	31,2	35,5	7,6	51,0	7,6
			I20	34,6	34,6	38,4	0,0	91,4	160,8	37,4	41,0	8,2	51,2	8,2
		H3	I10	50,8	50,8	56,7	0,0	96,0	160,4	54,3	61,3	6,9	51,0	6,9
			I20	49,6	49,6	55,6	0,0	103,9	158,8	52,7	58,9	6,3	51,2	6,3
	19	L1	I10	13,1	13,1	20,9	0,0	141,2	248,6	16,7	45,4	27,4	64,8	27,4
			I20	14,6	14,6	18,2	0,0	155,2	249,0	18,7	23,7	28,8	65,0	28,8
		H3	I10	21,2	21,2	27,1	0,0	148,0	244,8	25,1	34,1	19,1	64,6	19,1
			I20	21,6	21,6	26,8	0,0	140,6	243,4	24,9	33,7	15,2	64,7	15,2
média				**62,01**	**62,05**	**69,14**	**0,03**	**52,08**	**92,15**	**71,60**	**79,15**	**14,33**	**48,91**	**14,29**

Tabela 2 – ALWABP: Comparação de Resultados

					CS					Chaves et al. (2008)				
	NrT	Var	Inc	melhor	sol*	sol	gap	T*	T	sol*	sol	gap	T*	T
Roszieg (25 tarefas)	4	L1	I10	20,1	20,1	20,2	0,0	0,8	3,8	20,1	20,2	0,0	2,2	5,2
			I20	31,5	31,5	32,5	0,0	0,9	3,7	31,5	34,3	0,0	2,0	5,1
		H3	I10	28,1	28,1	28,5	0,0	0,6	3,8	28,1	28,1	0,0	2,0	5,2
			I20	28,0	28,0	28,0	0,0	0,2	3,8	28,0	28,1	0,0	1,9	5,2
	6	L1	I10	9,7	9,7	10,7	0,0	1,3	5,2	9,7	10,2	0,0	3,5	6,0
			I20	11,0	11,0	12,1	0,0	1,4	5,2	11,0	11,9	0,0	3,6	6,0
		H3	I10	16,0	16,0	16,9	0,0	1,5	5,2	16,0	16,2	0,0	3,5	6,0
			I20	15,1	15,1	15,6	0,0	1,9	5,2	15,1	15,4	0,0	3,4	6,0
Heskia (28 tarefas)	4	L1	I10	102,3	102,3	102,8	0,0	1,3	5,0	102,3	103,5	0,0	3,0	5,8
			I20	122,6	122,6	123,8	0,0	1,4	5,0	122,6	123,7	0,0	2,5	5,7
		H3	I10	172,5	172,5	175,5	0,0	1,7	5,0	172,5	173,1	0,0	2,6	5,8
			I20	171,2	171,2	171,7	0,0	1,4	5,1	171,2	171,8	0,0	2,7	5,8
	7	L1	I10	34,9	34,9	37,8	0,0	4,4	9,0	34,9	36,4	0,0	4,6	7,4
			I20	42,6	42,6	44,7	0,0	3,4	9,0	42,6	44,3	0,0	4,2	7,4
		H3	I10	75,2	75,2	77,7	0,0	2,9	9,0	75,2	76,4	0,0	3,6	7,4
			I20	67,2	67,2	70,7	0,0	3,6	9,0	67,2	70,2	0,0	4,1	7,4
Tonge (70 tarefas)	10	L1	I10	96,7	96,7	116,6	0,0	64,0	122,2	107,5	732,8	11,5	40,6	58,0
			I20	116,0	116,0	141,8	0,0	64,6	122,6	141,8	826,0	22,6	41,8	59,3
		H3	I10	167,1	167,9	199,4	0,5	66,2	122,8	179,5	778,7	7,5	39,4	58,0
			I20	174,0	174,0	206,0	0,0	65,7	123,0	206,4	755,2	18,6	38,1	57,9
	17	L1	I10	42,6	42,6	51,8	0,0	101,1	183,0	71,9	877,7	68,9	55,7	83,9
			I20	48,6	48,6	61,6	0,0	105,3	184,7	83,9	913,2	70,4	56,3	86,0
		H3	I10	78,6	78,6	94,0	0,0	100,1	184,5	132,6	912,0	73,0	53,6	84,9
			I20	78,2	78,6	95,6	0,6	100,3	184,9	113,8	875,9	43,5	59,1	84,9
Wee-Mag (75 tarefas)	11	L1	I10	29,0	29,0	32,7	0,0	94,3	163,2	33,6	59,0	16,2	52,8	69,5
			I20	34,6	34,6	38,4	0,0	91,4	160,8	38,5	45,9	11,3	59,0	71,3
		H3	I10	50,8	50,8	56,7	0,0	96,0	160,4	56,2	67,1	10,9	57,3	69,9
			I20	49,6	49,6	55,6	0,0	103,9	158,8	55,2	73,5	11,5	53,1	69,8
	19	L1	I10	13,1	13,1	20,9	0,0	141,2	248,6	19,4	504,2	48,0	59,0	99,4
			I20	14,6	14,6	18,2	0,0	155,2	249,0	21,8	544,9	49,9	66,8	102,2
		H3	I10	21,2	21,2	27,1	0,0	148,0	244,8	30,6	370,0	44,6	69,7	100,9
			I20	21,6	21,6	26,8	0,0	140,6	243,4	27,9	443,6	29,6	68,9	101,2
média				**62,01**	**62,05**	**69,14**	**0,03**	**52,08**	**92,15**	**70,89**	**304,48**	**16,81**	**28,76**	**42,33**

6. Conclusões

Este trabalho apresentou uma solução para o Problema de Balanceamento e Designação de Trabalhadores em Linha de Produção (ALWABP) utilizando o método Busca por Agrupamentos (CS) tendo como gerador de soluções a meta-heurística Busca Local Iterativa (ILS).

Os experimentos computacionais mostram que o CS é competitivo para resolver esse problema em um tempo computacional razoável. O CS encontrou o ótimo global para o conjunto de instâncias menores considerado nos testes computacionais. Para as instâncias maiores são obtidas boas soluções em pouco tempo de execução. Os resultados da versão do CS apresentada neste trabalho foram melhores que os resultados apresentados em Chaves et al. (2008). Portanto, estes resultados validam as melhorias propostas e a aplicação do CS ao ALWABP.

Na prática, esses tempos computacionais pequenos tornam viável um rápido balanceamento da linha de produção. Isto é muito importante se levar em consideração que, por causa do alto absentismo e dos exames psicológicos e físicos periódicos dos trabalhadores, o gerente do CTD conhece apenas no início de cada dia quais trabalhadores estão disponíveis. Portanto, abordagens como o CS, que proveem boas soluções em pouco tempo computacional, são muito interessantes para realizar um balanceamento diário da linha de produção. Procedimentos de balanceamento ágeis também são necessários para rotinas de rotação de trabalho, as quais são necessárias neste ambiente, obtendo múltiplas combinações facilmente.

Uma vantagem adicional do CS, que pode facilitar uma implementação real nos CTD's, é o fato de não utilizar nenhum *software* comercial. Desta forma, não acarreta maiores custos aos CTD's que são entidades sem fins lucrativos.

Um possível alvo de estudos futuros é a utilização de penalidades variáveis no cálculo da função objetivo do ALWABP. Esta estratégia ajudaria o CS manter a diversidade nas soluções. Outro trabalho interessante seria realizar uma implementação real deste problema.

Agradecimentos

Os autores agradecem à FAPESP (processo 2008/09242-9) pelo suporte a esta pesquisa.

Referências

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

Chaves, A. A., Miralles, C. e Lorena, L. A. N. (2008), Uma Metaheurística Híbrida Aplicada ao Problema de Balanceamento e Designação de Trabalhadores em Linha de Produção, *Atas do XL Simpósio Brasileiro de Pesquisa Operacional*, 1479-1490.

Costa, A. M. e Miralles, C., Rotação de tarefas em linhas de produção com trabalhadores deficientes, *Atas do XL Simpósio Brasileiro de Pesquisa Operacional*, 143-152, 2008.

Glover, F., Tabu search and adaptive memory programming: Advances, applications and challenges, em Barr, R. S., Helgason, R. V. e Kennington, J. L. (Eds.), *Interfaces in Computer Science and Operations Research*, Kluwer, 1–75, 1996.

Guthahr, A. L. e Nemhauser, G. L. (1964), An algorithm for the line balancing problem, *Management Research*, 12, 450-459.

Hoffmann, T. R., Assembly line balancing: A set of challenging problems, *International Journal of Production Research*, 28, 1807–1815, 1990.

Jaime, L. R. e Carmo, J. C., *A inserção de uma pessoa com deficiência no mundo do trabalho: o resgate de uma cidadania*, Ed. dos Autores, São Paulo, 2005.

Kirkpatrick, S., Gellat, D. C. e Vecchi, M. P. (1983), Optimization by simulated annealing, *Science*, 220, 671-680.

Lourenço, H., Martin, O. e Stützle, T., Iterated Local Seach, em Glover, F. e Kochenberg, G. A. (eds.), *Handbook of Metaheuristics*, Springer, New York, 2003.

Miralles, C., García-Sabater, J., Andrés, C. e Cardós, M. (2007), Advantages of assembly lines in sheltered work centres for disabled: a case study, *International Journal of Production Economics*, 187-197.

Miralles, C., García-Sabater, J., Andrés, C. e Cardós, 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.

Mladenovic, N. e Hansen, P., Variable Neighborhood Search, *Computers and Operations Research*, 24 (11), 1097-1100, 1997.

Oliveira, A. C. M. e Lorena, L. A. N., Hybrid evolutionary algorithms and clustering search, em Grosan, C., Abraham, A. e Ishibuchi, H. (Eds.), *Hybrid Evolutionary Systems - Studies in Computational Intelligence*. Springer SCI Series, 81–102, 2007.

Scholl, A. e Becker, C. (2006), State-of-the-art exact and heuristic solution procedures for simple assembly line balancing, *European Journal of Operational Research*, 168, 666–693.