UTILIZANDO O HARMONY SEARCH E CLUSTER SEARCH PARA RESOLVER O PROBLEMA DE BALANCEAMENTO DE LINHA DE PRODUÇÃO PARA CENTROS DE TRABALHO DE PESSOAS COM DEFICIÊNCIA

Renato Teixeira da Silva
Universidade Estadual Paulista "Júlio de Mesquita Filho" (UNESP)
Av. Dr. Ariberto Pereira da Cunha, 333, Guaratinguetá – SP – Brasil
renatovr@gmail.com

Antônio Augusto Chaves
Universidade Federal de São Paulo (UNIFESP)
Rua Talim, 330, Vila Nair, São José dos Campos – SP – Brasil
antonio.chaves@unifesp.br

Galeno José de Sena
Universidade Estadual Paulista "Júlio de Mesquita Filho" (UNESP)
Av. Dr. Ariberto Pereira da Cunha, 333, Guaratinguetá – SP – Brasil
gsena@feg.unesp.br

Jorge Muniz Junior
Universidade Estadual Paulista "Júlio de Mesquita Filho" (UNESP)
Av. Dr. Ariberto Pereira da Cunha, 333, Guaratinguetá – SP – Brasil
jorgemuniz@feg.unesp.br

RESUMO

O problema de balanceamento e designação de trabalhadores em linha de produção (ALWABP) consiste em otimizar a alocação de trabalhadores às estações de trabalho em uma linha de produção com pessoas com deficiência, objetivando organizar as linhas de produção de forma a aumentar a produtividade e ocultar as deficiências individuais. Este trabalho objetiva comparar as meta-heurísticas *Harmony Seach* (HS) e *Cluster Search* com meta-heurística geradora de soluções HS para o ALWABP. Os testes computacionais constatam a eficiência do CS, obtendo boas soluções em tempos computacionais aceitáveis.

PALAVARAS CHAVE. ALWABP, Harmony Search, Cluster Search.

MH – Metaheurísticas

ABSTRACT

The assembly line worker assignment and balancing problem (ALWABP) consists in optimize the allocation of workers in the workstations on an assembly line with disabled people, aiming to organize the assembly line to improve the productivity and to make disappear the individual disabilities. This work aims compare the metaheuristics Harmony Search (HS) and Cluster Search (CS) with HS as metaheuristic solution generator to ALWAPB. The computational tests show the efficiency of CS, obtaining good solutions in acceptable computational times.

KEYWORDS. ALWABP, Harmony Search, Cluster Search.

MH – Metaheuristics

1. Introdução

A Organização Internacional do Trabalho (ILO, do inglês *International Labour Organization*) estima que cerca de 72% da população mundial de pessoas com deficiência (em torno de 650 milhões de pessoas) estão em idade produtiva (Murray, 2010). No entanto, esta população possui altas taxas de desemprego: cerca de 50 a 70% em países industrializados e de 80 a 90% dessa população em países em desenvolvimento (United Nations, 2008). Segundo Tanaka e Manzini (2005), uma das dificuldades que geram estas altas taxas de desemprego apresentadas é a falta de preparação profissional e social das pessoas com deficiência.

Alguns países adotaram como estratégia de facilitar a inclusão dessas pessoas no mercado de trabalho a criação de Centros de Trabalho para Pessoas com Deficiências (CTD's). Eles são organizações sem fins lucrativos onde os trabalhadores com deficiência passam pela etapa de integração com o mercado de trabalho, sendo absorvidos posteriormente pelo mercado "normal" de trabalho. Um de seus princípios é afastar-se do estereótipo tradicional que considera as pessoas com deficiência incapazes de desenvolver um trabalho profissional contínuo, tendo em vista que os CTD's visam atender mercados reais, necessitando ser suficientemente flexíveis e eficientes para absorver suas variações.

Miralles *et al.* (2007) descreve como a aplicação de linhas de produção em CTD's provê muitas vantagens, sendo que, a divisão tradicional do trabalho em tarefas únicas é capaz de tornar imperceptíveis determinadas deficiências do trabalhador. Outra atribuição da tarefa é servir de método terapêutico para reabilitação das deficiências. Entretanto, algumas restrições específicas relacionadas à variabilidade dos tempos surgem neste ambiente, devendo o balanceamento conciliar os seguintes objetivos:

- Maximizar a eficiência da linha de produção, balanceando a carga de trabalho atribuída a cada trabalhador disponível em cada estação;
- Satisfazer e respeitar as restrições existentes neste ambiente devido aos fatores humanos ao atribuir tarefas aos trabalhadores.

Após analisar alguns CTD's, Miralles *et al.* (2007) observaram algumas características que podem ser encontradas neste ambiente, que serviram de motivação para a definição do Problema de Balanceamento de Linha e Designação de Trabalhadores (ALWABP, do inglês *Assembly Line Worker Assignment and Balancing Problem*).

O ALWABP pode ser classificado como um problema NP-difícil, sendo uma generalização do Problema Simples de Balanceamento de Linhas de Produção (SALBP, do inglês *Simple Assembly Line Balancing Problem*) (Scholl e Becker, 2006). Para problemas como este a aplicação de meta-heurísticas é oportuna, buscando obter bons resultados em um tempo computacional competitivo.

Sendo assim, propõe-se utilizar neste trabalho as meta-heurísticas *Harmony Search* (HS) e *Cluster Search* (CS) para a resolução o ALWABP. O objetivo é comparar o desempenho das duas meta-heurísticas entre si e com as melhores soluções encontradas na literatura. O HS possui método de otimização inspirado no processo musical de busca por um estado perfeito de harmonia, como é feito durante a improvisação do jazz. Já o CS visa detectar regiões promissoras no espaço de buscas e explorá-las por meio de um método de busca local.

O restante deste trabalho está organizado da seguinte forma. Na seção 2 faz-se uma revisão bibliográfica do ALWABP. Na seção 3 descrevem-se as meta-heurísticas HS e CS. Na seção 4 apresentam-se as implementações das meta-heurísticas aplicadas ao ALWABP. Na seção 5 apresentam-se os resultados computacionais obtidos. Na seção 6 são apresentadas algumas conclusões deste trabalho.

2. Revisão da literatura

Uma linha de produção é um sistema produtivo orientado pelo fluxo e organizado em estações de trabalho em série. A ideia é que cada unidade de produção passe pelas várias estações de trabalho através de um sistema de transporte, por exemplo, uma esteira ou um teleférico, para ser processada. Seu desenvolvimento possibilitou às empresas produzir em larga escala de forma padronizada e com custos eficientes. Este sistema vem ganhando força também na produção de baixo volume de produtos customizados. Levando em conta decisões relacionadas ao gerenciamento destes sistemas, os

problemas de balanceamento de linha de produção auxiliam de forma importante as decisões no nível tático e operacional das empresas. (Scholl e Becker, 2006).

Ao se trabalhar com CTD's (Centros de Trabalho de Pessoas com Deficiências), a utilização de uma abordagem simples de designação de trabalhadores é inviável, devido às características singulares de cada trabalhador. A partir desta motivação, Miralles *et al.* (2007) propuseram o Problema de Balanceamento de Linha e Designação de Trabalhadores (ALWABP, do inglês *Assembly Line Worker Assignment and Balancing Problem*). Neste problema, o objetivo consiste em alocar tanto as tarefas em estações de trabalho quanto os trabalhadores disponíveis às estações, tendo em vista que alguns trabalhadores podem ser muito lentos para executar certas tarefas ou até incapazes, e muito eficientes na execução de outras, devido à deficiência que eles apresentam.

Ao se trabalhar com o ALWABP, devem-se levar em conta as seguintes particularidades (Chaves *et al.*, 2009a):

- 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 as tarefas (varia conforme a habilidade e capacidade de cada trabalhador);
- 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;
- 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.

O ALWABP possui quatro diferentes abordagens. Dentre elas, a mais indicada para se aplicar aos CTD's é a ALWABP-2, que minimiza o tempo de ciclo dado o número de trabalhadores. Isto porque a finalidade desses centros é promover a inclusão das pessoas com deficiência no mercado de trabalho. Uma outra abordagem, o ALWABP-1, que minimiza número de trabalhadores dado um tempo de ciclo *c* não é muito usual. Sendo assim, este artigo delimita-se em tratar o ALWABP-2.

Boysen *et al.* (2007), na busca por diminuir a lacuna existente entre a academia e a prática, propuseram uma classificação para os problemas de balanceamento de linhas a partir de uma notação de três elementos [α | β | γ], sendo:

- α são possíveis características relacionadas ao grafo de precedências;
- β são possíveis características relacionadas às estações ou à linha de produção;
- γ indica os objetivos de otimização.

O ALWABP-2, segundo esta nomenclatura, classifica-se como [*pa, link, cum*|*equip*|*c*]. Isto porque o objetivo deste problema é minimizar o tempo de ciclo (γ = *c*), cada trabalhador possui características únicas (β = *equip*), o tempo de execução das tarefas varia de acordo com o trabalhador (α = *cum*), estas variações influenciam e devem ser consideradas no cálculo da função objetivo (α = *pa*) e existem conjuntos de tarefas encadeadas, que só podem ser alocadas para um mesmo trabalhador (α = *link*).

3. Meta-heurísticas

3.1. Harmony Search

Na literatura existem várias meta-heurísticas inspiradas em fenômenos naturais, por exemplo, recozimento físico no *Simulated Annealing* (Kirkpatrick, 1984), a memória humana na Busca Tabu (Glover, 1986), o processo de evolução no Algoritmo Genético (Goldberg e Holland, 1988), entre outras. Uma nova meta-heurística, introduzida por Geem *et al.* (2001), chamada Busca Harmônica (HS, do inglês *Harmony Search*) é inspirada no processo musical de busca por um estado perfeito de harmonia, como é feito durante a improvisação do jazz. Cada músico de jazz toca um possível acorde musical, que juntos, formarão uma harmonia. Caso a harmonia gerada seja boa, esta experiência é guardada em uma memória do músico, para ser usada futuramente, aumentando as chances de se melhorar a harmonia gerada em uma próxima rodada. Analogamente, no processo de otimização cada

variável de decisão é inicialmente gerada de forma aleatória dentro de um intervalo possível, e combinadas, geram uma solução com um dado valor de função objetivo. Caso esta seja uma boa solução, estas variáveis são guardadas em uma memória, para que sejam usadas na geração de novas soluções, aumentando as possibilidades de melhorar os resultados.

O processo de otimização do algoritmo HS consiste em cinco passos (Geem *et al.*, 2001; Lee e Geem, 2005; Coelho e Bernert, 2009):

Passo 1: Inicialização do problema de otimização e dos parâmetros do algoritmo

Primeiramente, o problema de otimização é especificado conforme abaixo:

Minimizar F sujeito a $x_i \in X_i$, $i = 1,..., N$

onde F é a função objetivo, x é o conjunto das variáveis de decisão (x_i); X_i é o conjunto de possíveis valores para cada variável de decisão, sendo para variáveis contínuas $x_{i,lower} < X_i < x_{i,upper}$, onde $x_{i,lower}$ e $x_{i,upper}$ os limites inferior e superior ou, $X_i = \{ x_i(1), x_i(2), ..., x_i(K)\}$ para variáveis discretas. Neste contexto, os parâmetros usados para resolver o algoritmo HS são especificados neste passo. O número de soluções na memória harmônica (HMS, em inglês *Harmony Memory Size*), que é o tamanho da matriz da memória harmônica, a taxa de consideração da memória harmônica (HMCR, em inglês *Harmony Memory Considering Rate*), taxa de afinação (PAR, em inglês Pitch Adjusting Rate), e o número máximo de buscas (critério de parada) são selecionados neste passo. Os parâmetros HMCR e PAR serão definidos no passo 3, onde são utilizados.

Passo 2: Inicializar a memória harmônica (HM, do inglês *Harmony Memory*)

Na HM ficam alocados os vetores de solução. Neste passo, a matriz HM, apresentada na equação (10), é preenchida com vetores de soluções gerados aleatoriamente utilizando uma distribuição uniforme.

$$HM = \begin{bmatrix} x_1^1 & \cdots & x_N^1 \\ \vdots & \ddots & \vdots \\ x_1^{HMS} & \cdots & x_N^{HMS} \end{bmatrix} \quad (1)$$

Passo 3: Improvisar uma nova harmonia a partir da HM.

Um novo vetor de harmonia, $x' = (x_1', x_2', \dots, x_N')$, é gerado com base na consideração da memória, ajustes finos e seleção aleatória. Em analogia ao processo musical, a criação de um novo vetor de solução, uma nova harmonia, é denominada improvisação. Na consideração da memória, cada valor de uma variável (x_i') da nova solução é definido a partir de qualquer valor respectivo da HM ($x_i^1 - x_i^{HMS}$). A HMCR, variando de 0 a 1, é a taxa de consideração de um valor dos valores históricos contidos na HM, enquanto (1 – HMCR) é a taxa de escolha aleatória de um valor dentro do intervalo de possíveis valores.

$$x_i' \leftarrow \begin{cases} x_i' \in \{x_i^1, x_i^2, \dots, x_i^{HMS}\} & \text{com a probabilidade } HMCR \\ x_i' \in X_i & \text{com a probabilidade } (1 - PAR) \end{cases} \quad (2)$$

Em seguida, todo componente obtido a partir da consideração da memória é examinado para determinar quando ele deve ter seu tom ajustado. O PAR, que também varia de 0 a 1, é a taxa de ajuste do valor obtido na memória, enquanto (1-PAR) é a taxa onde o valor obtido da memória é considerado sem alterações.

$$\text{Decisão de ajuste de tom para } x_i' \leftarrow \begin{cases} Sim & \text{com a probalilidade } PAR \\ Não & \text{com a probabilidade } (1 - PAR) \end{cases} \quad (3)$$

Caso haja o ajuste para o tom de x_i', este valor deve ser substituído por;

$$x_i^{'} \leftarrow x_i^{'} \pm r.bw \quad (4)$$

onde r é um valor aleatório gerado a partir de uma distribuição uniforme entre 0 e 1 e bw é o intervalo arbitrário máximo de variação do tom.

Passo 4: Atualizar a HM

Em geral, caso o novo vetor de harmonia $x' = (x'_1, x'_2, \dots, x'_N)$ possua um valor da função objetivo menor que o pior vetor existente na HM, substitui-se o pior vetor pelo novo vetor de harmonia gerado. Contudo, outros aspectos podem ser também avaliados, por exemplo, semelhança entre vetores, evitando uma convergência a um ótimo local.

Passo 5: Checar critério de parada

Caso o critério de parada definido seja atingido, a execução do algoritmo é finalizada. Caso contrário, executa-se novamente os passos 3 e 4.

3.2. Busca por Agrupamentos (CS)

O método híbrido Busca por Agrupamentos (do inglês *Cluster Search*) de Chaves *et al.* (2007), baseado no *Evolutionary Clustering Search* (ECS) de Oliveira e Lorena (2007), visa encontrar boas soluções por meio da intensificação de buscas em regiões promissoras do espaço de soluções. O espaço de soluções é divido em regiões ou *clusters* que são atualizados dinamicamente. Este processo de atualização é feito a cada iteração do algoritmo alimentado pelas informações geradas por uma meta-heurística.

No CS, três atributos definem um *cluster* $C = (c, v, r)$. Sua localização dentro do espaço de busca é representada pelo centro c. Cada cento contém parte das características das soluções agrupadas no respectivo *cluster*. Com isso, elimina-se a necessidade de armazenar todas as suas soluções. A quantidade de soluções agrupadas no *cluster* é definida pelo atributo v e através da comparação deste com um limitante λ, isto é, quando v atingir este valor um *cluster* passa a ser considerado promissor. Visando controlar e adicionar "inteligência" ao método de busca local evitando que o mesmo não seja executado por mais de r_{max} vezes em regiões já exploradas de forma satisfatória anteriormente ou em regiões ruins, o índice de ineficácia r é acompanhado. Ele apresenta o número de vezes consecutivas em que a busca local foi aplicada no *cluster* sem melhorar a solução.

Como artifício para conseguir distribuir as soluções em *clusters* define-se uma função de medida de distância $d(i, j)$, que calcula a distância entre duas soluções *i* e *j* como um número positivo. Quanto maior a similaridade entre duas soluções menor será a função de medida de distância.

A estratégia híbrida do CS é descrita nos cinco passos a seguir:

Passo 1: Criação dos Clusters Iniciais

A partir de um número definido de *clusters*, devem-se gerar os seus centros de forma a representar diferentes regiões do espaço de busca. Propõe-se utilizar um método baseado na diversidade máxima, onde as *m* soluções mais distantes entre si são selecionadas a partir de *n* soluções geradas aleatoriamente, tal que $n > m$.

Passo 2: Gerar solução s_k pela meta-heurística

Através da utilização de qualquer meta-heurística, gerar uma solução s_k. Esta solução será a informação que alimentará todo processo de agrupamento. Deve-se preferencialmente utilizar uma meta-heurística capaz de gerar soluções com grande diversidade, possibilitando ao CS uma ampla análise do espaço de soluções.

Passo 3: Agrupar solução s_k ao cluster C_j mais similar

Ao receber a solução s_k gerada pela meta-heurística, o CS a compara com todos os centros dos *clusters*, visando encontrar o de menor distância com a solução, considerando C_j o mais similar. A intenção é reunir soluções similares dentro dos *clusters* e direcionar a busca para

as regiões supostamente promissoras. Ao inserir uma nova solução no *cluster* C_j, deve-se primeiramente atualizar seu centro, para que também conste traços da característica desta nova solução. Para isto, é utilizado o método *Path-Relinking* (Glover (1996), utilizando como novo centro do *cluster* a melhor solução encontrada dentro da trajetória que interconecta o centro c_j e a solução s_k. O volume do *cluster* v_j também é alterado, sofrendo um incremento.

Passo 4: Realizar busca local no cluster C_j

Para que o *cluster* C_j seja elegível para se aplicar busca local, ele deve ser considerado promissor. Isso se dá quando o volume v_j atinge o limitante λ.

$$v_j = \lambda \leftarrow \begin{cases} \text{Sim} & \text{analisar eficiência da busca em C}_\text{j} \\ \text{Não} & \text{passar para próximo passo} \end{cases} \quad (5)$$

Em seguida, analisa-se o sucesso do método de busca local através do atributo r_j. Caso o método não tenha sucesso nas últimas r_{max} aplicações neste *cluster* ($r_j = r_{max}$) é aplicado em seu centro c_j uma perturbação aleatória, objetivando escapar desta região do espaço de busca, e reiniciando o valor de r_j. Caso contrário, uma busca local é aplicada neste centro, a fim de intensificar a busca na vizinhança de C_j.

$$r_j = r_{max} \leftarrow \begin{cases} \text{Sim} & \text{perturbar c}_\text{j}\text{, reiniciar r}_\text{j}\text{ e ir para próximo passo} \\ \text{Não} & \text{realizar busca local} \end{cases} \quad (6)$$

Considera-se de sucesso a busca local que consegue encontrar uma solução que seja a melhor obtida neste *cluster* até o momento (c_j^*). Quando isto acontece o centro do *cluster* é novamente atualizado e o atributo de ineficiência da busca r_j é reiniciado. Caso contrário, somente incrementa-se r_j, registrando a ineficiência da busca nesta aplicação.

$$Busca\ local(c_j) = c_j^* \leftarrow \begin{cases} \text{Verdadeiro} & \text{atualizar c}_\text{j}\text{ e reiniciar r}_\text{j} \\ \text{Falso} & \text{incrementar r}_\text{j} \end{cases} \quad (7)$$

Passo 5: Checar critério de parada

Caso o critério de parada definido seja atingido, a execução do algoritmo é finalizada. Caso contrário, executa-se novamente os passos 2, 3 e 4.

4. Meta-heurísticas aplicadas ao ALWABP

Inspirado em Chaves *et al.* (2009), uma solução do ALWABP é representado por dois vetores: um representa a alocação trabalhador/estação de trabalho; e o outro representa a designação tarefa/estação de trabalho, como mostrado na expressão (8). Um exemplo de uma solução com 10 tarefas, 4 trabalhadores e 4 estações de trabalho pode ser vista na Figura 1.

$$s = \{x', x''\} \quad (8)$$

Tarefas	*n1*	*n2*	*n3*	*n4*	*n5*	*n6*	*n7*	*n8*	*n9*	*n10*
Estação de Trabalho	*s1*	*s2*	*s1*	*s2*	*s2*	*s3*	*s4*	*s3*	*s4*	*s4*

Trabalhadores	*h1*	*h2*	*h3*	*h4*
Estação de Trabalho	*s2*	*s3*	*s4*	*s1*

Figura 1 - Exemplo de solução do ALWABP. Adaptado de Chaves *et al.* (2009).

Como visto na seção 2, o valor da função objetivo do ALWABP-2 é igual ao tempo de ciclo *C*. Como em Chaves *et al.* (2009), será acrescentado um complemento, responsável por penalizar as soluções inviáveis geradas, como visto na equação (9). No ALWABP-2, dois tipos de violações podem ocorrer e inviabilizar a solução gerada. A primeira violação é o desrespeito às regras de precedência, mensurada em f_p. A segunda é a alocação de um trabalhador a uma tarefa que ele não é capaz de realizar, mensurada em f_t. Estas variáveis, f_p e f_t, são acompanhadas por fatores multiplicador ω e δ, respectivamente, responsável por diferenciar a penalização de acordo com a violação ocorrida.

$$f(s) = C + (\omega \cdot f_p + \delta \cdot f_t) \tag{9}$$

4.1. Desenvolvendo o HS

Esta seção traz uma descrição dos elementos pertencentes ao HS. Seu pseudo-código pode ser visto na Figura 2.

Algoritmo HS
criar HM
enquanto critério de parada não satisfeito **faça** // *Geração do primeiro vetor de solução*
  para (*i*=1 até *n*)
    se (*HMCR < aleatório [0,1])* **então** // *Consideração da memória*
      definir tamanho do bloco *k*
      para (j=i até *k*)
        $x_j' \leftarrow aleatório\{x_j^1, x_j^2, \dots, x_j^{HMS}\}$
        se (HMCR < aleatório [0,1]) **então** // *Ajuste de tom*
          $x_j' \leftarrow x_j' \pm aleatório[0,1] \cdot bw$
        fim se
      fim para
      atualizar i ← k
    senão
      $x_i' \leftarrow aleatório\ X_i$
    fim se
  fim para
  $S_{temp} \leftarrow S$ // *Geração do segundo vetor de solução*
  para (*i*=1 até *m*)
    $x_i'' \leftarrow aleatório\ S_{temp}$
    retirar x_i'' de S_{temp}
  fim para
  se (f_p=*0*) **então** // *Busca local no segundo vetor*
    x'' ← heurística busca local (x'')
  fim se
  Atualizar Memória (x', x'') // *Atualização da memória*
fim enquanto

Figura 2 - Pseudo-código algoritmo HS

A memória harmônica será formada por *HMS* conjuntos de dois vetores, representada na expressão (10). Nesta aplicação utilizaremos uma memória com 100 harmonias.

$$HM = \begin{bmatrix} x_1' & x_1'' \\ x_2' & x_2'' \\ & \vdots \\ x_{HMS}' & x_{HMS}'' \end{bmatrix} \tag{10}$$

A memória harmônica inicial é constituída de 80% de soluções aleatórias e o seu complemento de soluções que não violam as precedências entre tarefas. Para estas, utilizou-se um método adaptado de Chaves *et al.* (2009). Define-se de forma aleatória a quantidade de tarefas que

devem ser alocadas em cada estação de trabalho. É permitido que uma estação não possua tarefas alocadas. Constrói-se uma lista em ordem crescente das tarefas com o menor número de tarefas antecessoras. Quando há 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. Após alocar as tarefas, os trabalhadores são alocados aleatoriamente às estações. Com isso, a única inviabilidade que pode ocorrer é a entre trabalhadores e tarefas incompatíveis.

Como pode ser visto em seu pseudo-código, o HS foi aplicado somente ao primeiro vetor de solução x' , que aloca as estações de trabalho às tarefas. A taxa de consideração da memória de 90% foi aplicada a blocos de variáveis da solução. O tamanho destes blocos é definido aleatoriamente, limitado em 25% do tamanho de uma solução. Apesar de trabalhar com blocos, o ajuste de tom foi aplicado a cada uma das variáveis do bloco. Sua taxa variou linearmente iniciando em 25% e terminando em 10%, de acordo com a iteração.

Para realizar a otimização no segundo vetor de solução x'' , onde os trabalhadores são alocados nas estações de trabalho, foi utilizada uma heurística de busca local após uma alocação aleatória dos trabalhadores, visando resultados eficientes. A heurística consiste em buscar o melhor movimento ao se trocar a alocação em estações de trabalho de dois trabalhadores.

A atualização da memória harmônica verifica inicialmente se a solução gerada na iteração é melhor que a pior solução da memória. Caso não seja, ela pode ser aceita com uma probabilidade de 5% para diversificar a memória. Sendo melhor, primeiramente é verificado se há alguma solução semelhante na memória. Considera-se semelhante toda solução que possua uma diferença menor que 30% em relação à nova solução. Quando isto acontece, caso a nova solução seja melhor que a solução mais semelhante, substitui-se esta solução. Caso contrário, parte-se para uma nova iteração. Quando não há solução semelhante, com uma probabilidade de 95% substitui-se o pior indivíduo. Com 5% de probabilidade, substitui-se o indivíduo mais semelhante, mesmo a diferença entre as duas soluções sendo maior que 30%. O objetivo é não perder a diversidade da memória harmônica, importante para possibilitar a busca em todo o espaço de soluções.

Como critério de parada, foi utilizado o valor de 100000 iterações.

4.2. Desenvolvendo o CS

Esta seção traz uma descrição dos elementos pertencentes ao CS. Seu pseudo-código pode ser visto na Figura 3.

Após identificar o *cluster* C_j mais similar a solução corrente $\hat{s}$, atualiza-se o centro c_j com as características da solução $\hat{s}$. Para este procedimento utiliza-se o *Path-Relinking* partindo do centro c_j em direção a solução $\hat{s}$. Contudo, a análise foi limitada a 30% do caminho, evitando que o centro se mova para regiões muito distantes da atual.

A definição de um *cluster* promissor é dada a partir da análise de seu volume v_j, ou seja, quando este atinge o limitante λ. Neste trabalho utiliza-se $\lambda = 10$. Ao verificar que um cluster C_j é promissor, antes de aplicar um procedimento de busca local, é analisado se a heurística vem obtendo sucesso neste *cluster* através do r_j. Caso r_j atinja $r_{max} = 5$, o centro c_j sofre uma perturbação, trocando 30% da alocação das tarefas aleatoriamente. Sendo $r_j < r_{max}$, intensifica-se a busca por meio da heurística de busca local.

Assim como em Chaves *et al.* (2009), a heurística de busca local utilizada foi o *Variable Neighborhood Descent*. Ele foi modelado com três heurísticas com base em movimentos relevantes para o ALWABP:

- Trocar tarefa: realizar o melhor movimento de trocar duas tarefas que estão atribuídas a estações diferentes;
- Transferir tarefa: realizar o melhor movimento de transferir uma tarefa de uma estação para outra;
- Trocar trabalhador: realizar o melhor movimento de trocar a posição de dois trabalhadores.

Algoritmo CS

criar clusters iniciais
$\hat{s} \leftarrow$ {meta-heurística – HS}
enquanto critério de parada não satisfeito **faça**
  encontrar cluster C_j mais similar a $\hat{s}$
  $v_j \leftarrow v_j + 1$ // agrupar $\hat{s}$ em C_j
  $c_j = Path\ Relinking(c_j, \hat{s}')$ // atualizar centro do cluster
  se $(v_j = \lambda)$ **então**
    $v_j \leftarrow 0$
    se $(r_j = r_{max})$ **então**
      $c_j \leftarrow PerturbaçãoAleatória(c_j)$
      $r_j \leftarrow 0$
    senão
      $\hat{c}_j \leftarrow BuscaLocal\ (c_j)$
      se $f(\hat{c}_j) < f(c_j)$ **então**
        $c_j \leftarrow \hat{c}_j$
      fim se
      se $f(\hat{c}_j) < f(c_j^*)$ **então**
        $c_j^* \leftarrow \hat{c}_j$
        $r_j \leftarrow 0$
      senão
        $r_j \leftarrow r_j + 1$
      fim se
    fim se
  fim se
fim enquanto

Figura 3 - Pseudo-código algoritmo CS
Adaptado de Chaves *et al.* (2009)

As heurísticas são executadas na sequência apresentada. Caso haja melhora na solução, o VND regressa a primeira heurística, parando somente quando não existam mais melhoras para a solução corrente. Caso a solução encontrada seja melhor que o centro c_j, o mesmo é atualizado. Contudo, o índice de ineficácia r_j só é reiniciado caso a solução encontrada seja a melhor encontrada até o momento neste *cluster*.

O critério de parada utilizado é o critério do HS, ou seja, 100000 iterações.

5. Resultados computacionais

O HS e o CS foram codificados em C++ e os testes computacionais foram realizados em um PC com processador Pentium Dual Core 1,60 Ghz e 2 GB de memória RAM. O objetivo dos experimentos é evidenciar a qualidade dos resultados do CS, em comparação a utilização somente do HS, monstrando que o CS pode ser competitivo para resolver o ALWABP.

Ao todo, foram testadas 320 instâncias dividias em 4 conjuntos de problemas teste: *Roszieg*, *Heskia*, *Tonge* e *Wee-Mag*. Estas instâncias, propostas por Chaves *et al.* (2007), foram baseadas na coleção de problemas clássicos do SALBP (Hoffmann, 1990). Suas principais características (Tar – número de tarefas, Trab – número de trabalhadores e OS – característica da rede de precedências) podem ser vistas na Tabela 1. Elas estão disponíveis no endereço http://www.feg.unesp.br/~chaves/instancias.html.

Tabela 1 – Características das Instâncias

Família	Tar	Trab		OS
		Grupos 1-4	Grupos 5-8	
Roszieg	25	4	6	71,67
Heskia	28	4	6	22,49
Wee-mag	70	10	17	11,67
Tonge	75	11	19	59,42

Durante os testes, executa-se os algoritmos 20 vezes cada uma das instâncias. Os resultados obtidos podem ser vistos nas tabelas a seguir. Eles também foram comparados com os melhores resultados encontrados na literatura até o momento, gerados a partir da utilização do *Beam Search* (IBS) por Blum e Miralles (2011), executados em um PC com AMD64X2 e 4 GB de memória.

Tabela 2 - Resultados da Família Roszieg

Roszieg		IBS			HS_ALWABP			CSHS_ALWABP		
Grupo	Melhor	Melhor	Média	t (s)	Melhor	Média	t (s)	Melhor	Média	t (s)
1	20,1	***20,1***	20,1	0,1	***20,1***	20,4	87,5	***20,1***	20,2	151,8
2	31,5	***31,5***	31,5	0,9	***31,5***	42,5	72,1	***31,5***	31,5	150,5
3	28,1	***28,1***	28,1	1,3	***28,1***	28,4	69,2	***28,1***	28,1	153,0
4	28,0	***28,0***	28,0	0,0	***28,0***	28,1	69,8	***28,0***	28,0	149,7
5	9,7	***9,7***	9,7	0,1	***9,7***	10,5	74,0	***9,7***	10,0	173,8
6	11,0	***11,0***	11,0	0,1	11,3	12,5	77,6	***11,0***	11,6	179,4
7	16,0	***16,0***	16,0	0,1	16,1	17,1	73,9	***16,0***	16,2	178,9
8	15,1	***15,1***	15,1	0,2	15,2	16,0	74,3	***15,1***	15,4	178,6
Média	**19,9**	**19,9**	**19,9**	**0,4**	**20,0**	**21,9**	**74,8**	**19,9**	**20,1**	**164,4**

Tabela 3 - Resultados da Família Heskia

Heskia		IBS			HS_ALWABP			CSHS_ALWABP		
Grupo	Melhor	Melhor	Média	t (s)	Melhor	Média	t (s)	Melhor	Média	t (s)
1	102,3	*102,3*	102,3	81,6	103,7	108,5	80,2	*102,3*	103,8	204,7
2	122,6	*122,6*	122,6	29,8	125,6	130,0	78,4	*122,6*	124,2	204,5
3	172,5	*172,5*	172,5	56,3	176,6	180,9	78,7	*172,5*	174,3	206,3
4	171,2	*171,2*	171,3	52,1	172,1	176,1	79,4	*171,2*	171,6	206,3
5	34,9	*34,9*	34,9	10,9	38,7	43,1	94,3	35,1	37,6	260,7
6	42,6	*42,6*	42,6	25,4	45,7	49,4	94,7	43,1	44,2	270,3
7	75,2	*75,2*	75,2	16,7	78,9	83,9	92,8	75,4	77,7	262,7
8	67,2	*67,2*	67,2	25,1	73,9	77,9	93,7	67,3	70,8	262,4
Média	**98,6**	**98,6**	**98,6**	**37,2**	**101,9**	**106,2**	**86,5**	**98,7**	**100,5**	**234,7**

Tabela 4 - Resultados da Família Wee-mag

Wee-mag		IBS			HS_ALWABP			CSHS_ALWABP			CS			BT		
Grupo	Melhor	Melhor	Média	t (s)	Melhor	Média	t (s)	Melhor	Média	t (s)	Melhor	Média	t (s)	Melhor	Média	t (s)
1	28,7	***28,7***	29,7	1049,1	45,8	52,4	1183,7	33,3	36,7	4464,2	29,0	32,7	94,3	35,3	38,4	57,9
2	33,6	***33,6***	34,9	849,3	51,4	62,0	1195,0	37,4	41,8	4635,3	34,6	38,4	91,4	41,1	45,3	58,2
3	50,1	***50,1***	51,6	1603,3	79,3	91,9	1231,9	56,4	62,6	4694,1	50,8	56,7	96,0	58,4	64,8	58,4
4	48,6	***48,6***	50,4	1433,4	76,9	91,9	1195,2	55,5	61,8	4729,9	49,6	55,6	103,9	56,1	61,3	57,9
5	10,3	***10,3***	10,7	570,5	23,3	27,5	1441,5	15,4	18,0	5445,0	13,1	20,9	141,2	19,9	23,8	71,7
6	11,9	***11,9***	12,4	602,4	29,5	35,4	1441,4	18,4	21,0	5542,7	14,6	18,2	155,2	23,3	28,6	71,3
7	18,2	***18,2***	19,0	713,6	42,0	50,1	1471,1	26,4	30,1	5693,2	21,2	27,1	148,0	32,6	39,2	71,1
8	18,1	***18,1***	18,9	900,0	41,9	50,6	1485,6	26,3	30,4	6313,4	21,6	26,8	140,6	31,7	39,1	71,2
Média	**27,4**	**27,4**	**28,4**	**965,2**	**48,8**	**57,7**	**1330,7**	**33,6**	**37,8**	**5189,7**	**29,3**	**34,6**	**121,3**	**37,3**	**42,6**	**64,7**

Os resultados apresentados mostram a eficácia da meta-heurística CS em comparação ao HS, sendo superior em todos os conjuntos de problema. Para os conjuntos Roszieg e Heskia, o CS obteve resultados semelhantes aos do IBS, tendo uma variação de 0,1% em relação ao conjunto Heskia.

Para as instâncias do conjunto Wee-mag, o IBS conseguiu obter soluções em média 18% melhores que o CS e 43% melhor que o HS. Contudo, ao se comparar com o CS proposto por Chaves *et al.* (2009) e a Busca Tabu proposta por Moreira e Costa (2009) para resolver o ALWABP, houve uma melhora de 1,1% em relação ao CS e de 8% em relação a Busca Tabu.

Tabela 5 - Resultados da Família Tonge

Tonge		IBS			HS_ALWABP			CSHS_ALWABP		
Grupo	Melhor	Melhor	Média	t (s)	Melhor	Média	t (s)	Melhor	Média	t (s)
1	93,7	94,9	96,7	863,8	112,9	135,7	944,1	***93,7***	101,3	3592,6
2	110,2	***110,2***	111,5	921,7	137,1	168,6	1038,3	111,1	118,8	3684,5
3	162,3	165,0	168,0	1502,8	198,5	229,2	964,6	***162,3***	172,1	4582,9
4	169,0	170,0	171,4	1495,0	211,8	247,0	964,1	***169,0***	182,0	3727,4
5	33,1	***33,1***	34,2	879,8	54,2	66,7	1157,4	39,2	43,1	4414,7
6	40,0	***40,0***	41,0	705,0	69,7	87,6	1167,7	44,6	50,3	4569,9
7	66,4	***66,4***	67,9	1242,8	103,2	124,8	1101,1	74,0	80,3	4623,0
8	64,7	***64,7***	66,6	1564,2	105,9	128,2	1098,9	72,2	81,1	4631,1
Média	**92,4**	**93,0**	**94,7**	**1146,9**	**124,2**	**148,5**	**1054,5**	**95,8**	**103,6**	**4228,3**

Para o conjunto Tonge, que possui as maiores instâncias, o resultado global do CS foi 3% pior que o IBS e o HS 25% pior. Entretanto, em três dos oito grupos existentes o CS apesentou um resultado melhor que o IBS. Com isso, houve uma redução de 0,7% no melhor resultado já encontrado do conjunto Tonge, saindo de 93,0, resultado obtido pelo IBS, para 92,4.

6. Conclusões

Este artigo apresenta a resolução do Problema de Balanceamento de Linha e Designação de Trabalhadores (ALWABP) utilizando duas abordagens: o *Harmony Search* e o *Cluster Search* utilizando o *Harmony Search* como meta-heurística geradora de soluções.

A resolução deste problema tem grande importância para a sociedade, pois uma considerável parte da população mundial apresenta algum tipo de deficiência e há uma grande dificuldade em incluir estas pessoas no mercado de trabalho. Para que a utilização do modelo de CTD's seja viável, é necessário conseguir balancear a linha de forma rápida, tendo em vista que os trabalhadores possuem altas taxas de absenteísmo e exames periódicos de saúde física e psicológica frequentes, sendo possível aos gestores somente no início do dia saber quais são os trabalhadores disponíveis.

O CS apresenta em todos os conjuntos de problemas testados resultados melhores que o HS. Para três grupos do conjunto Tonge, o CS encontrou resultados melhores que o *Beam Search*, diminuindo os valores das melhores soluções conhecidas na literatura. Além disso, os resultados encontrados apresentaram uma boa robustez e tempo computacional razoável, viabilizando sua aplicação em CTD's.

Para estudos futuros, sugere-se a utilização de extensões do *Harmony Search* visando melhorar a geração de soluções para o *Cluster Search*. Esta alteração pode permitir ao HS indicar melhor as regiões promissores e fazer com que o algoritmo atinja os resultados em um tempo computacional menor. Outro caminho é realizar uma aplicação prática deste problema, que auxiliaria a enxergar novas restrições do problema e viabilizar um melhor atendimento dos CTD's.

Referências

Blum, C.; Miralles, C. (2011). On solving the assembly line worker assignment and balancing problem via beam search. *Computers & Operations Research***,** v. 38, n. 1, p. 328-339, Jan ISSN 0305-0548.

Boysen, N.; Fliedner, M.; Scholl, A. (2007). A classification of assembly line balancing problems. *European Journal of Operational Research***,** v. 183, n. 2, p. 674-693, ISSN 0377-2217.

Chaves, A.; Lorena, L.; Miralles, C. (2009). Hybrid metaheuristic for the assembly line worker assignment and balancing problem. *Lecture Notes in Computer Science (Hybrid Metaheuristics)***,** v. 5818, p. 1-14,

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

Coelho, L. D.; Bernert, D. L. D. (2009). An improved harmony search algorithm for synchronization of discrete-time chaotic systems. *Chaos Solitons & Fractals***,** v. 41, n. 5, p. 2526-2532, Sep ISSN 0960-0779.

Geem, Z. W.; Kim, J. H.; Loganathan, G. V. (2001). A new heuristic optimization algorithm: Harmony search. *Simulation***,** v. 76, n. 2, p. 60-68, ISSN 00375497 (ISSN).

Glover, F. (1986). Future paths for integer programming and links to artificial intelligence. *Computers and Operations Research***,** v. 13, n. 5, p. 533-549, ISSN 03050548 (ISSN).
Glover, F., Tabu search and adaptive memory programming-advances, applications and challenges. Interfaces in computer science and operations research: Kluwer Academic Publishers, v.1, p.1-75, 1 ed. 1996.
Goldberg, D.; Holland, J. (1988). Genetic algorithms and machine learning. *Machine Learning***,** v. 3, n. 2, p. 95-99,
Hoffmann, T. R. (1990). Assembly line balancing: a set of challenging problems. *International Journal of Production Research***,** v. 28, n. 10, p. 1807 - 1815, ISSN 0020-7543. Acesso em: October 17, 2010.
Kirkpatrick, S. (1983). Optimization by simulated annealing: Quantitative studies. *Journal of Statistical Physics***,** v. 34, n. 5, p. 975-986,
Lee, K. S.; Geem, Z. W. (2005). A new meta-heuristic algorithm for continuous engineering optimization: harmony search theory and practice. *Computer Methods in Applied Mechanics and Engineering***,** v. 194, n. 36-38, p. 3902-3933, ISSN 0045-7825.
Miralles, C.; Garcia-Sabater, J.; Andres, C.; Cardos, M. (2007). Advantages of assembly lines in Sheltered Work Centres for Disabled. A case study. *International Journal of Production Economics***,** v. 110, n. 1-2, p. 187-197,
Moreira, M.; Costa, A. (2009). A minimalist yet efficient tabu search algorithm for balancing assembly lines with disabled workers.
Murray, B. Employment for social justice and a fair globalization. 2010. Disponível em: < http://www.ilo.org/wcmsp5/groups/public/---ed_emp/documents/publication/wcms_140958.pdf >. Acesso em: 03 out. 2010.
Oliveira, A.; Lorena, L. (2007). Hybrid evolutionary algorithms and clustering search. *Hybrid Evolutionary Algorithms*, p. 77-99,
Scholl, A.; Becker, C. (2006). State-of-the-art exact and heuristic solution procedures for simple assembly line balancing. *European Journal of Operational Research***,** v. 168, n. 3, p. 666-693, Feb ISSN 0377-2217.
Tanaka, E. D. O.; Manzini, E. J. (2005). O que os empregadores pensam sobre o trabalho da pessoa com deficiência? *Revista Brasileira de Educação Especial***,** v. 11, p. 273-294, ISSN 1413-6538.
United Nations. Mainstreaming Disability in the Development Agenda. 2008. Disponível em: < http://www.un.org/disabilities/documents/reports/e-cn5-2008-6.doc >. Acesso em: 03 out. 2010.