Um Método Para Planejamento de Produção em Sistemas de Manufatura Flexível

Gustavo Silva Paiva
Departamento de Ciência da Computação, Universidade Federal de Ouro Preto
Campus Morro do Cruzeiro, Ouro Preto, Minas Gerais, 35400-000, Brasil.
guustavo.paiva@gmail.com

Marco Antonio Moreira de Carvalho
Departamento de Ciência da Computação, Universidade Federal de Ouro Preto
Campus Morro do Cruzeiro, Ouro Preto, Minas Gerais, 35400-000, Brasil.
mamc@iceb.ufop.br

RESUMO

Em sistemas de manufatura flexível, uma mesma máquina pode ser configurada com diferentes ferramentas para processar diferentes tarefas, cada uma exigindo um conjunto específico de ferramentas. Há um limite para o número máximo de ferramentas instaladas simultaneamente na máquina, e entre o processamento de duas tarefas diferentes pode ser necessária a troca destas ferramentas, implicando na interrupção da produção para este fim. O problema de Minimização de Trocas de Ferramentas visa sequenciar o processamento das tarefas no intuito de minimizar estas trocas. Neste trabalho são propostos uma nova representação em grafos para o problema, uma nova heurística e um novo método de busca local. Estes métodos foram combinados em uma busca local iterada e comparados ao estado da arte em relação ao problema tratado. Experimentos computacionais extensos demonstram que o método proposto é competitivo e obteve novas melhores soluções para instâncias da literatura, superando o estado da arte atual.

PALAVRAS CHAVE. Escalonamento, Minimização de Troca de Ferramentas, Busca Local Iterada.

Tópicos: IND, MH, OC.

ABSTRACT

In flexible manufacturing systems, a single machine can be configured with different tools for processing different tasks, each requiring a specific set of tools. There is a limit to the maximum number of tools installed simultaneously on the machine and between the processing of two different tasks it may be necessary to switch these tools, causing the interruption of the production line. The Minimization of Tool Switches problem aims to sequence the processing of tasks in order to minimize these switches. This paper proposes a new representation in graphs, a new heuristic and a new local search method. These methods were combined in an iterated local search and compared to state of the art related to the problem. Extensive computational experiments show that the proposed method is competitive and obtained new best solutions for literature instances, surpassing the current state of the art.

KEYWORDS. Scheduling, Minimization of Tool Switches, Iterated Local Search.

Topics: IND, MH, OC.

1. Introdução

Um sistema de manufatura flexível é caracterizado por permitir uma maior versatilidade no planejamento da produção no tocante a variedade de produtos fabricados e rápida adequação em caso de imprevistos. Um tipo muito comum deste sistema, principalmente empregado por empresas metalúrgicas, utiliza *máquinas flexíveis*.

Uma máquina flexível possui a capacidade de efetuar diferentes tipos de operações (corte, perfuração, etc) sem que haja uma brusca troca de contexto entre uma operação e outra, tornando a produção mais dinâmica. Para isto, este tipo de máquina possui um compartimento de capacidade fixa em que *ferramentas* são carregadas. Cada produto a ser fabricado requer que um conjunto de ferramentas (e.g., lâminas de corte, brocas de perfuração, etc) específico seja carregado na máquina flexível para sua produção. O compartimento de ferramentas é suficiente para armazenar todas as ferramentas necessárias para fabricação dos produtos isoladamente, porém, geralmente não é suficiente para armazenar todas as ferramentas existentes simultaneamente. A fabricação de um tipo de produto específico pode ser entendida como uma *tarefa* a ser executada.

Para executar diferentes tarefas em sequência em uma mesma linha de produção, tornam-se obrigatórias as trocas de ferramentas quando o número total de ferramentas necessárias para realizar as tarefas for maior do que a capacidade do respectivo compartimento – o que é o caso comum. Para realizar estas trocas, as máquinas precisam ser desligadas, interrompendo a linha de produção. A quantidade destas trocas deve ser minimizada de forma a aumentar a produtividade pela diminuição das interrupções na linha de produção e consequentemente o tempo ocioso da máquina de produção.

A partir de uma demanda por produtos, predeterminada, é necessário a criação de um plano de produção para que uma máquina cumpra esta demanda. Este plano é dividido em tarefas e tem como objetivos maximizar a produtividade e diminuir os custos relacionados. Um plano de produção consiste em: (i) determinar a ordem em que as tarefas serão executadas; e (ii) decidir quando realizar cada troca de ferramentas e quais ferramentas serão trocadas.

O *Problema de Minimização de Trocas de Ferramentas* (*Minimization of Tool Switches Problem* – MTSP) é definido como o problema de determinar um plano de produção, gerando a sequência em que as tarefas devem ser executadas de forma a minimizar o número de trocas de ferramentas necessário durante o processo de produção. A segunda parte do plano de produção é trivial e pode ser determinada em tempo determinístico polinomial pelo algoritmo *Keep Tool Needed Soonest*, vide [Tang e Denardo, 1988].

Uma instância do MTSP é composta pelo conjunto de tarefas que devem ser realizadas $T = \{1, \ldots, n\}$, o conjunto de ferramentas disponíveis $F = \{1, \ldots, m\}$, o conjunto de ferramentas F_j necessárias para executar a tarefa $j \in T$ e a capacidade C do compartimento de ferramentas da máquina. Uma solução do MTSP é representada pela permutação ϕ do conjunto T e também de um plano de trocas de ferramentas.

O plano de trocas de ferramentas MTSP é representado por uma matriz binária $A^{\phi}_{m,n}$ na qual as colunas obedecem a ordem das colunas em ϕ e cada entrada $a^{\phi}_{i,j} = 1$ caso a ferramenta i esteja na máquina durante a execução da tarefa j, ou $a^{\phi}_{i,j} = 0$ caso contrário. Logo, o número de trocas de ferramentas corresponde ao número de inversões em que $a^{\phi}_{i,j-1} = 0$ e $a^{\phi}_{i,j} = 1$ ($\forall i \in F,\ j \in \{2, \ldots n\}$), indicando que a ferramenta i não estava carregada na máquina durante a execução de uma tarefa $j-1$, porém, foi carregada antes da execução da tarefa j. O carregamento das ferramentas iniciais também são contabilizadas como trocas de ferramentas.

A Tabela 1 apresenta um exemplo numérico de uma instância do MTSP. A primeira linha representa as tarefas (enumeradas de 1 a 5) e as quatro próximas linhas apresentam as ferramentas necessárias para realizar a tarefa da respectiva coluna. Por fim a última linha apresenta a capacidade do compartimento de ferramentas da máquina.

Tabela 1: Exemplo de uma instância do MTSP.

Tarefas	1	2	3	4	5
Ferramentas	1	1	3	2	1
	2	3	4	3	4
	4		5	5	6
Capacidade do compartimento = 3					

A Tabela 2 apresenta duas possíveis soluções para a instância da Tabela 1. Em (a) a matriz A^{ϕ} se refere à solução $\phi = [1, 2, 3, 4, 5]$. A ferramenta 4, sublinhada, não é necessária para execução da tarefa 2, mas foi mantida no compartimento para evitar futuras trocas desnecessárias. A solução exige 9 trocas de ferramentas, sendo três para inserção das ferramentas iniciais (ferramentas 1, 2 e 4); uma troca para execução da tarefa 2 (ferramenta 2 pela ferramenta 3); uma troca para execução da tarefa 3 (ferramenta 1 pela ferramenta 5); uma troca na para execução da tarefa 4 (ferramenta 4 pela ferramenta 2) e três trocas para execução da tarefa 5 (ferramentas 2, 3 e 5 pelas ferramenta 1, 4 e 6, respectivamente). A Tabela 2(b), apresenta uma segunda solução, referente a $\phi = [3, 4, 1, 5, 2]$, que resulta em 8 trocas de ferramentas.

Tabela 2: Exemplo de representação em matriz binária e possível solução.

T	1	2	3	4	5
	1	1	0	0	1
	1	0	0	1	0
	0	1	1	1	0
	1	<u>1</u>	1	0	1
	0	0	1	1	0
	0	0	0	0	1

(a)

ϕ	3	4	1	5	2
	0	0	1	1	1
	0	1	1	0	0
	1	1	0	0	1
	1	0	1	1	<u>1</u>
	1	1	0	0	0
	0	0	0	1	0

(b)

O MTSP foi caracterizado $\mathcal{NP}$-Difícil por [Crama et al., 1994] e possui aplicação prática direta nas indústrias metalúrgica e microeletrônica, especificamente no processo de montagem de circuitos eletrônicos impressos.

2. Contribuições Deste Trabalho

Neste trabalho propõem-se uma modificação da representação do MTSP por grafos, uma heurística e também um método de busca local. Ambos os métodos são combinados em uma estratégia de busca local iterada para solução do MTSP. As seções seguintes apresentam os detalhes.

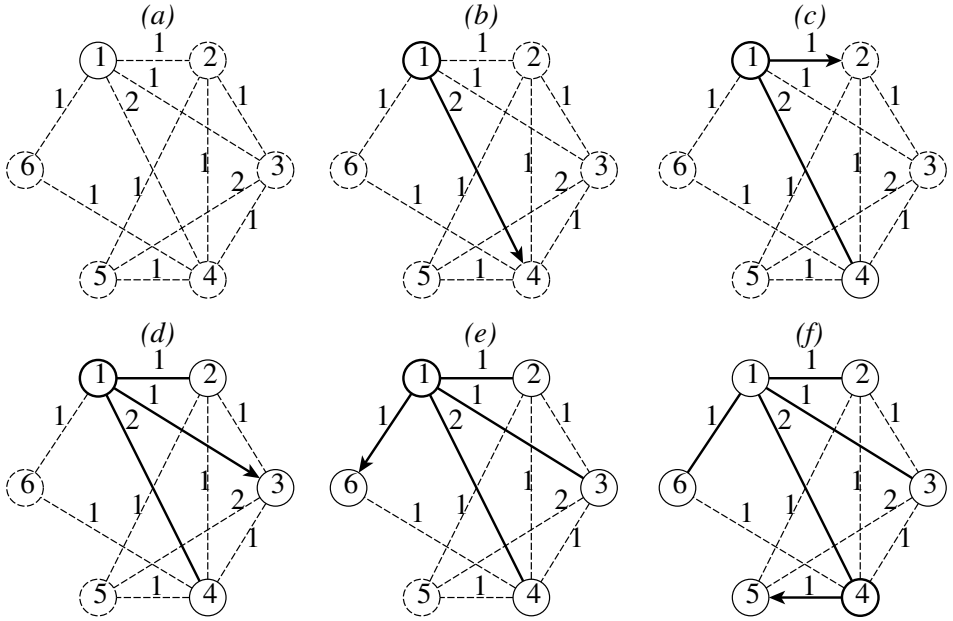
2.1. Uma Heurística Baseada em Busca em Grafos

Um *Grafo de Ferramentas* é definido como $G = (V, E)$, no qual V é o conjunto de vértices que representam as ferramentas e E é o conjunto das arestas $\{i, j\}$ que representam se uma ferramenta i é utilizada ao mesmo tempo da ferramenta j. Neste trabalho propõe-se que o peso de cada aresta $\{i, j\}$ represente a quantidade de vezes que a ferramenta i e a ferramenta j aparecem juntas em uma mesma tarefa. Salvo melhor juízo, não há registro de utilização desta estratégia para solução do MTSP.

No intuito de determinar a sequência das tarefas ϕ, inicialmente identifica-se quais ferramentas tendem a ser utilizadas em conjunto com maior frequência. A partir de um vértice inicial executa-se uma Busca em Largura (ou *Breadth-First Search* – BFS) com a adição de um critério baseado no peso das arestas para definição da ordem de exploração dos demais vértices. Arestas de maior peso indicam ferramentas que são utilizadas mais vezes em pares na máquina, logo devem estar próximas no sequenciamento das ferramentas. Desta forma, o critério de maior grau guia a execução da BFS. Ao final da execução da BFS a ordem de exploração dos vértices F_{ϕ} será a sequência de ferramentas.

Na Figura 1 é ilustrado o passo a passo da execução da *BFS* conforme descrita, usando o exemplo apresentado na Tabela 1.

Figura 1: Execução da BFS para o sequenciamento de ferramentas na instância apresentada na Tabela 1.



Considerando o vértice 1 como o inicial, sua vizinhança possui 4 vértices (2, 3, 4 e 6) que serão explorados na ordem [4, 2, 3, 6], conforme ilustra a sequência de (a) a (e). Na sequência (f) o vértice 4 tem sua vizinhança explorada e somente o vértice 5 ainda não o havia sido. Neste ponto, todos os vértices já foram explorados e obtemos o seguinte sequenciamento de ferramentas $F_\phi = [1, 4, 2, 3, 6, 5]$.

O sequenciamento das tarefas ϕ é obtido a partir de F_ϕ da seguinte maneira: simula-se a disponibilização sucessiva das ferramentas em F_ϕ, uma a cada instante, verificando-se a composição das tarefas do problema, de modo que, caso a composição de uma tarefa seja o conjunto ou subconjunto das ferramentas disponíveis em determinado instante, esta tarefa é imediatamente inserida ao final da solução ϕ. Para o exemplo da Tabela 1 e $F_\phi = [1, 4, 2, 3, 6, 5]$, tem-se $\phi = [1, 2, 5, 3, 4]$.

2.2. Um Novo Método de Busca Local

[Crama et al., 1994] apresentaram uma definição de *1-blocks*: um conjunto de entradas consecutivas de uma linha de uma matriz com o valor igual a 1. Desconsiderando-se a inserção das ferramentas iniciais, uma troca de ferramenta é caracterizada pela existência de dois *1-blocks* diferentes em uma mesma linha na matriz A^ϕ. Com base nesta definição, propõe-se uma busca local que tem como objetivo diminuir o número de *1-blocks* em cada linha da matriz A^ϕ.

Esta busca local consiste em examinar iterativamente a matriz A^ϕ em procura por dois ou mais *1-blocks* em uma mesma linha. Ao encontrá-los, tenta-se agrupá-los contiguamente dois a dois pela movimentação das colunas relacionadas ao primeiro *1-block*, uma a uma, para antes ou depois das colunas relacionadas ao segundo *1-block*, o que for melhor em relação ao número de trocas de ferramentas correspondente. Caso ambos os movimentos de uma coluna piorem a solução, esta coluna não é movimentada. O Algoritmo 1 apresenta o pseudo-código da busca local proposta.

Para ilustrar o funcionamento do agrupamento de *1-blocks*, considere a instância do MTSP apresentada na Tabela 1 e a solução inicial $\phi = [1, 2, 3, 4, 5]$, mostrada na tabela 2(a), cujo número de trocas de ferramentas é 9. Inicialmente, examina-se a primeira linha da matriz e encontra-se o *1-block* j, formado pelas colunas 1 e 2. Em seguida encontra-se o segundo *1-block* k, composto apenas pela coluna 5. Avalia-se então as possíveis realocações das colunas de j. Primeiro considera-se

Algoritmo 1: Agrupamento de *1-blocks*

Dados: Matriz A^{ϕ}
para cada *linha* l *da matriz* A^{ϕ} **faça**
  Examine a linha l até encontrar um primeiro *1-block* j;
  enquanto *Existir um proximo 1-block* k **faça**
    Para cada coluna de j decida com base com o número de troca se esta deve ser inserida antes ou depois de k ou ser mantida em j;
    $j \leftarrow k$;
  fim
fim
retorna A^{ϕ};

a coluna 1: as possíveis movimentações geram as sequências $[2, 3, 4, 1, 5]$ e $[2, 3, 4, 5, 1]$, resultando em 8 e 9 trocas respectivamente. Como há um movimento de melhora, ele é executado e a solução parcial é atualizada para $[2, 3, 4, 1, 5]$. Em seguida, examina-se a segunda coluna do primeiro *1-block*. As sequências decorrentes dos dois movimentos são $[3, 4, 1, 2, 5]$ e $[3, 4, 1, 5, 2]$, resultando em 8 trocas ambos. Não havendo piora do valor da solução, o segundo movimento é realizado, resultando na solução apresentada na Tabela 2(*b*). Na sequência, o método segue examinando as demais linhas da matriz, até ter examinado todas.

3. Busca Local Iterada

Após a geração de uma solução inicial pela aplicação da heurística proposta, optou-se por realizar o aprimoramento da mesma usando a estratégia de *Busca Local Iterada* [Lourenço et al., 2003] (ou *Iterated Local Search*, ILS), que consiste em aplicar iteradamente métodos de busca local e perturbar uma solução até que atinja uma condição de parada.

Para compor este método, utilizou-se a solução inicial gerada pela heurística proposta na Seção 2.1, a busca local por agrupamento de *1-blocks* proposta na Seção 2.2 e o método 2-*opt* tanto como mecanismo de busca local quanto mecanismo de perturbação. O algoritmo *2-opt* é um método utilizado amplamente na literatura e consiste na realização de trocas de posições entre pares de elementos, gerando soluções que diferem em exatamente 2 elementos da configuração inicial. O pseudo-código do método ILS composto pelos algoritmos propostos é apresentado no Algoritmo 2.

Algoritmo 2: Busca Local Iterada

Dados: ϕ, α, δ
Aplique a busca local de agrupamento de *1-blocks* em A^{ϕ};
enquanto *critério de parada não atendido* **faça**
  $\phi' \leftarrow \phi$;
  Perturbe a solução ϕ' em uma proporção α;
  Aplique a busca local *2-opt* em ϕ' em uma proporção δ;
  Aplique a busca local de agrupamento de *1-blocks* em ϕ';
  se ϕ' *atender o critério de aceitação* **então**
    $\phi \leftarrow \phi'$;
  fim
fim
retorna ϕ;

A perturbação de uma solução é feita através de sucessivas trocas de posições entre duas tarefas selecionadas aleatoriamente. O número de vezes em que estas trocas ocorrerão é definido como uma porcentagem do número de tarefas do problema indicado pelo parâmetro α. A aplicação do método *2-opt* como mecanismo de perturbação visa evitar que a busca local fique estagnada em ótimos locais. A aplicação do mesmo método como mecanismo de busca local se assemelha à

aplicação anterior, entretanto, esta possui a característica de somente serem realizadas alterações na solução que resultem em diminuição do valor da mesma. No intuito de redução do tempo de execução deste método, optou-se por aplicá-lo em uma proporção δ do espaço de busca, selecionando aleatoriamente os pares de tarefas para troca de posição.

Após experimentos preliminares, definiram-se os valores de parâmetros $\alpha = \lfloor 0,2 \times |T| \rfloor$ e $\delta = \left\lfloor 0,25 \times \binom{|T|}{2} \right\rfloor$. O critério de aceitação utilizado é a diminuição no valor da solução, e o critério de parada é definido como 200 iterações.

4. Experimentos Computacionais

O método proposto foi implementado utilizando a linguagem C++, compilado utilizando g++ 4.4.1 e a opção de otimização -O3. Os experimentos foram realizados em um computador com processador *Intel i5 Quad Core* de 3.2GHz, 8 GB de RAM, e sistema operacional Ubuntu 15.10.

Os resultados obtidos foram comparados com uma implementação combinada de *Clustering Search*, *Descida em Vizinhança Variável* e *Algoritmo Genético de Chaves Aleatórias Viciadas*, denominada *CS+BRKGA* [Chaves et al., 2016], cujos resultados são os melhores para as instâncias da literatura e representam o estado da arte atualmente. Os resultados deste método foram obtidos em um computador com processador *Intel Core i7* de 3.4 GHz e 16GB de RAM. É importante notar que, apesar de diferentes, as arquiteturas originais de ambos os métodos utilizados neste experimentos possuem poder de processamento comparáveis.

Nas tabelas a seguir são apresentados o número de tarefas (n), o número de ferramentas (m), a capacidade da máquina (C), número de instâncias (e), a melhor solução encontrada por cada método (S^*), a solução média encontrada por cada método (S), o tempo médio de execução de cada método em segundos (T) e também o *gap* (ou distância percentual) em relação ao melhor resultado conhecido, calculado como *100 × (valor_obtido - valor_referência) / valor_referência*. Devido à utilização de componentes de aleatoriedade pelos métodos, foram realizadas 20 execuções independentes por instância, avaliando-se também o desvio padrão das soluções (denotado por σ) obtidas para cada instância.

4.1. Instâncias de Yanasse et al. [2009]

As 1510 instâncias propostas por [Yanasse et al., 2009] foram divididas em 5 grupos (A, B, C, D e E). Os grupos A, B, C e E são compostos em sua grande maioria por instâncias triviais e ambos métodos conseguiram obter todas as soluções ótimas (provadas por [Yanasse et al., 2009]). Por restrição de espaço, os resultados detalhados referentes a estes grupos serão omitidos, entretanto, estes podem ser obtidos diretamente com os autores. A Tabela 3 apresenta o sumário dos resultados para estes grupos.

Tabela 3: Sumário dos resultados para os grupos A, B, C e E.

Conjunto	e	*OPT*	*CS+BRKGA*		Método Proposto		
			S	T	S	T	σ
A	450	24,544	24,544	3,71	24,544	0,11	0,000
B	330	25,216	25,218	4,05	25,217	0,18	0,0026
C	340	28,925	29.079	9,83	28.926	1,67	0,0034
E	80	16,888	16,949	6,54	16,890	0,51	0,0051

Das 260 instâncias do grupo D, apenas para 189 se conhece as soluções comprovadamente ótimas. O *ILS* foi capaz de igualar todas as melhores soluções conhecidas e também estabelecer novas melhores soluções, conforme apresentado na Tabela 4.

No conjunto D as diferenças se tornam mais claras entre *ILS* ($\sigma = 0,11$ e $T = 6,05$s) e *CS+BRKGA* ($\sigma = 2,15$ e $T = 27,66$s). Não é possível determinar em quantas instâncias o resultado foi melhorado, dado que apenas os resultados médios são reportados na literatura.

Tabela 4: Resultados do Grupo D. Valores em negrito indicam que as melhores soluções foram atingidas.

n	m	C	e	*CS+BRKGA*				Método Proposto			
				S^*	S	*gap*	T	S^*	S	*gap*	T
20	15	5	10	26,10	26,58	0,77%	10,78	**25,90**	25,90	0,00%	1,61
20	15	10	20	**18,20**	18,44	0,00%	12,34	**18,20**	18,21	0,00%	2,70
20	20	5	10	29,30	29,93	0,17%	14,84	**29,20**	29,25	0,00%	2,14
20	20	10	10	**20,60**	20,76	0,00%	16,08	**20,60**	20,60	0,00%	3,22
20	20	15	30	**21,67**	21,79	0,00%	24,66	**21,67**	21,67	0,00%	4,16
20	25	5	10	**35,10**	35,74	0,00%	19,16	**35,10**	35,14	0,00%	2,35
20	25	10	10	**25,40**	25,47	0,00%	21,49	**25,40**	25,40	0,00%	4,10
20	25	15	40	**36,25**	36,75	0,00%	28,11	**36,25**	36,26	0,00%	9,35
20	25	20	40	**26,15**	26,28	0,00%	35,53	**26,15**	26,15	0,00%	4,17
25	15	10	10	**15,90**	16,00	0,00%	21,14	**15,90**	15,90	0,00%	4,56
25	20	10	10	**21,60**	22,05	0,00%	27,48	**21,60**	21,60	0,00%	11,38
25	20	15	10	**22,60**	22,82	0,00%	25,66	**22,60**	22,60	0,00%	12,50
25	25	10	10	**26,60**	27,06	0,00%	36,88	**26,60**	26,70	0,00%	13,46
25	25	15	10	**25,00**	25,00	0,00%	54,70	**25,00**	25,00	0,00%	7,41
25	25	20	30	**25,50**	25,59	0,00%	66,10	**25,50**	25,50	0,00%	7,67

4.2. Instâncias de Crama et al. [1994]

As 160 instâncias propostas por [Crama et al., 1994] foram divididas em 4 grupos (C_1, C_2, C_3 e C_4). Os grupos C_1 e C_2 são considerados triviais e de pequenas dimensões (10 e 15 tarefas respectivamente). Os resultados para os grupos C_1 e C_2 são sumarizados na Tabela 5.

Tabela 5: Sumário dos resultados para os grupos C_1 e C_2.

Conjunto	e	*OPT*	*CS+BRKGA*		Método Proposto		
			S	T	S	T	σ
C_1	450	11,175	11,178	3,71	11,175	0,11	0,000
C_2	330	22,000	22,073	4,05	22,025	0,18	0,056

Os grupos C_3 e C_4 exigiram maior esforço dos métodos comparados em relação aos anteriores. Não se conhece as soluções ótimas destas instâncias, entretanto, o método proposto foi capaz de determinar novos melhores resultados. Novamente, não é possível determinar em quantas instâncias o resultado foi melhorado, dado que se conhece apenas os resultados médios. Os resultados são detalhados na Tabela 6.

Tabela 6: Resultados dos Grupos C_3 e C_4. Valores em negrito indicam que as melhores soluções foram atingidas.

n	m	C	e	*CS+BRKGA*				Método Proposto			
				S^*	S	*gap*	T	S^*	S	*gap*	T
30	40	15	10	106,80	108,03	0,25%	140,05	**106,40**	106,82	0,00%	104,03
30	40	17	10	88,70	89,98	0,23%	127,09	**88,50**	88,78	0,00%	160,88
30	40	20	10	70,70	71,85	0,28%	121,43	**70,50**	70,80	0,00%	228,83
30	40	25	10	53,10	53,97	0,38%	104,02	**52,90**	53,15	0,00%	206,65
40	60	20	10	199,80	202,25	0,55%	599,34	**198,70**	199,27	0,00%	533,78
40	60	22	10	175,30	177,28	0,75%	557,97	**174,00**	174,40	0,00%	801,76
40	60	25	10	147,50	149,35	0,68%	533,52	**146,50**	146,83	0,00%	1144,95
40	60	30	10	114,50	116,94	0,44%	473,56	**114,00**	114,39	0,00%	1908,61

Para o grupo C_3, o *ILS* se mostra mais robusto ($\sigma = 0,63$ e $T = 175,10$s) do que o

CS+BRKGA ($\sigma = 2,26$ e $T = 123,15$s), porém, ao custo de maior tempo de execução. No grupo C_4 esta tendência se acentua: o *ILS* mantém baixo o erro em relação às melhores soluções ($\sigma = 0,85$), ao contrário do *CS+BRKGA* ($\sigma = 4,36$) que não foi capaz de obter as melhores soluções para nenhum dos subgrupos de instâncias. Entretanto, o tempo de execução médio do *ILS* ($T = 1097,28$s) é aproximadamente duas vezes o tempo de execução do *CS+BRKGA* ($T = 541,10$s), muito embora seja um tempo aceitável na prática.

5. Conclusões e Trabalhos Futuros

A Minimização de Trocas de Ferramentas é um problema combinatório $\mathcal{NP}$-Difícil de interesse prático por sua aplicabilidade industrial direta em sistemas de manufatura flexível.

Neste trabalho foram propostos uma nova representação em grafos para o problema, um novo método heurístico para geração de soluções e um novo método de busca local baseada no agrupamento de *1-blocks*. Estes métodos foram embutidos em uma estratégia de busca local iterada (ou *ILS*) juntamente com o conhecido método *2-opt*, utilizado como método de perturbação e de busca local.

Os experimentos computacionais consideraram 1510 instâncias da literatura (divididas em 9 grupos) e comparou o método proposto com o estado da arte, representado pela recente metaheurística *CS + BRKGA*. Ambos os métodos foram capazes de obter as soluções ótimas em baixo tempo de execução para grande parte das instâncias, porém, consideradas fáceis. Para os conjuntos de instâncias mais difíceis, a diferença de performance entre os métodos tornou-se clara. O *ILS* foi capaz de estabelecer novos melhores resultados para 3 diferentes grupos de instâncias. Adicionalmente, o *ILS*, que contém componentes de aleatoriedade, demonstrou maior robustez do que o estado da arte ao obter um baixo desvio padrão entre as soluções obtidas em execuções independentes. De uma maneira geral, o método proposto é competitivo e robusto. Embora o tempo de execução tenha aumentado consideravelmente para instâncias de maiores dimensões, ainda tratam-se de valores aceitáveis na prática.

Os trabalhos futuros se concentrarão na pesquisa de novas estruturas de vizinhança e aceleração das já existentes, seja por melhoria das técnicas empregadas ou pelo emprego de novas técnicas, como paralelismo.

6. Agradecimentos

Esta pesquisa foi apoiada pelo Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq), pela Fundação de Apoio à Pesquisa do Estado de Minas Gerais (FAPEMIG) e pela Universidade Federal de Ouro Preto.

Referências

Chaves, A. A., Lorena, L. A. N., Senne, E. L. F., e Resende, M. G. C. (2016). Hybrid method with CS and BRKGA applied to the minimization of tool switches problem. *Computers & Operations Research*, 67:174–183. ISSN 0305-0548.

Crama, Y., Kolen, A. W. J., Oerlemans, A. G., e Spieksma, F. C. R. (1994). Minimizing the number of tool switches on a flexible machine. *International Journal of Flexible Manufacturing Systems*, 6(1):33–54. ISSN 0920-6299, 1572-9370.

Lourenço, H. R., Martin, O. C., e Stützle, T. (2003). *Iterated local search*. Springer.

Tang, C. S. e Denardo, E. V. (1988). Models Arising from a Flexible Manufacturing Machine, Part I: Minimization of the Number of Tool Switches. *Operations Research*, 36(5):767–777. ISSN 0030-364X.

Yanasse, H. H., Rodrigues, R. d. C. M., e Senne, E. L. F. (2009). Um algoritmo enumerativo baseado em ordenamento parcial para resolução do problema de minimização de trocas de ferramentas. *Gestão & Produção*, 16(3):370–381. ISSN 0104-530X.