UM ALGORITMO HEURISTICO PARA O *ORDER ACCEPTANCE AND SCHEDULING PROBLEM*

Yuri Laio Teixeira, Arthur Kramer, Anand Subramanian
Departamento de Engenharia de Produção - Universidade Federal da Paraíba
Centro de Tecnologia, Campus I - Bloco G, Cidade Universitária, 58051-970, João Pessoa, PB
{yurilaio, arthurhfrk}@gmail.com, anand@ct.ufpb.br

RESUMO

Este artigo trata do *Order Acceptance and Scheduling Problem* (OASP), que pode ser visto como uma variante do problema de sequenciamento da produção em uma máquina com tempo de *setup* dependente da sequência e objetivo de minimizar a soma dos atrasos ponderados. A principal diferença é que não é necessário sequenciar todas as tarefas, mas apenas um subconjunto delas. Cada tarefa tem uma receita associada e um peso devido ao atraso, assim como no problema original. O objetivo é maximizar a receita total, a qual pode ser vista como a soma da receita menos o atraso ponderado de cada tarefa sequenciada. Outras características como *release date* e *deadlines* também são consideradas no OASP. Para a resolução deste problema é proposto um algoritmo baseado em *Iterated Local Search* (ILS). Experimentos computacionais foram realizados em 1500 instâncias disponíveis na literatura e um grande número de soluções aprimorantes foram obtidas.

PALAVRAS CHAVE. ***Order Acceptance and Scheduling*, *Iterated Local Search*, Meta-heurística.**

Áreas Principais: PO na Administração e Gestão da Produção, Meta-heurística. Otimização Combinatória

ABSTRACT

This paper concerns the Order Acceptance and Scheduling Problem (OASP), which can be seen as a variant of the the single machine total weighted tardiness scheduling problem with sequence-dependent setup times. The main difference is that it is not necessary to schedule all jobs, but only a subset of them. Each job has an associate revenue and a weight due to tardiness as in the original problem. The objective is to maximize the total revenue which can be seen as the sum of the revenue minus the weighted tardiness of each scheduled job. Other characteristics such as released dates and deadlines are also considered in the OASP. To solve this problem, an Iterated Local Search (ILS) based algorithm is proposed. Computational experiments were performed in 1500 instances available in the literature and a large number of improved solutions were found.

KEY WORDS. Order Acceptance and Scheduling, Iterated Local Search, Metaheuristic.

Main areas: OR in Administration and Production Management. Metaheuristic. Combinatorial Optimization

1 Introdução

O ambiente atual das empresas é caracterizado por mercados que enfrentam uma concorrência acirrada e que as expectativas dos clientes estão se tornando cada vez mais elevadas, em termos de qualidade, custo e prazos de entregas (Lopez e Roubellat, 2008). Deste modo, as organizações buscam o melhor aproveitamento dos recursos. Com base nisto, é natural pensar que as empresas desejem entregar os pedidos de seus clientes dentro dos prazos estipulados. Esse é o caso dos problemas de sequenciamento da produção que objetivam minimizar a soma dos atrasos ponderados, como o $1|s_{jk}|\sum(w_jT_j)$ por exemplo. O $1|s_{jk}|\sum(w_jT_j)$ é um problema bastante abordado na literatura, tanto por métodos exatos (Tanaka e Araki, 2013), quanto por métodos heurísticos (Anghinolfi e Paolucci, 2008, 2009; Tasgetiren *et al*., 2009; Sioud *et al*., 2012; Kirlik e Oguz, 2012; Xu *et al*., 2013; Subramanian *et al*., 2014).

Por outro lado, em virtude das restrições de capacidade, as empresas muitas vezes devem analisar os pedidos com o intuito de decidir quais devem ser aceitos e em qual ordem eles devem ser processados. Tal decisão é feita com o objetivo de proporcionar a maior receita para a empresa. Esta problemática motivou a proposição do *Order Acceptance and Scheduling Problem* (OASP), recentemente sugerido na literatura por Oğuz *et al*. (2010). O OASP, como definido em Oğuz *et al*. (2010), considera, além de outras características, tempo de *setup* dependente da sequência, *release dates* e *deadlines*.

Antes da proposição do OASP, como apresentado por Oğuz *et al*. (2010), já era possível perceber o interesse no problema de decidir quais ordem aceitar e como sequenciá-las. Slotnick e Morton (2007) e Rom e Slotnick (2009) abordaram o OASP sem a presença de tempos de *setup* e de *deadlines*. O primeiro trabalho modela o problema como um problema de programação inteira e propõe um algoritmo *Branch-and-Bound* (B&B) com o uso de relaxação linear para a resolução do problema. Contudo, tal método é capaz de resolver apenas problemas de pequenas dimensões. Assim, os autores propõem também métodos heurísticos para o problema. O segundo por sua vez, propõe um algoritmo genético para o mesmo problema. O trabalho de Nobibon *et al*. (2009) fornece uma rápida visão dos métodos até então utilizados para este problema. Recentemente, Nobibon e Leus (2011) propuseram duas formulações matemáticas baseadas em programação linear e dois algoritmos B&B, capazes de resolver instâncias com até 50 tarefas em tempo aceitável, para o OASP sem tempos *setup* e sem *deadlines*.

Wang *et al*. (2013a,b) tratam do OASP num ambiente *flow shop*. Os autores apresentaram duas formulações matemáticas baseadas em programação linear inteira mista para o problema, capazes de resolver apenas problemas de pequenas dimensões (Wang *et al*., 2013a). Então, os autores desenvolveram um algoritmo B&B para o problema. Já em Wang *et al*. (2013b) os autores atacaram o problema por meio de meta-heurísticas, mais especificamente um algoritmo de colônia de abelhas.

Para o OASP, na configuração tratada neste trabalho, é possível encontrar, além do trabalho de Oğuz *et al*. (2010), o trabalho de Cesaret *et al*. (2012). Em Oğuz *et al*. (2010), os autores apresentam uma formulação matemática baseada em programação linear inteira mista, capaz de resolver problemas com até 15 ordens e três algoritmos heurísticos para os problemas de dimensão mais elevada. Cesaret *et al*. (2012) por sua vez desenvolveram um algoritmo baseado na meta-heurística Busca Tabu para este problema.

Uma visão geral do OASP e suas variantes podem ser encontradas em (Zhong *et al*., 2014; Wei e Ma, 2014; Slotnick, 2011; Shabtay *et al*., 2013).

É notório que o OASP se assemelha bastante ao $1|s_{jk}|\sum(w_jT_j)$. A principal diferença é que não se faz estritamente necessário o processamento de todas as ordens pertencentes ao conjunto O, além da adição da noção de receita. Em virtude disto, e pelo fato do problema ser $\mathcal{NP}$–difícil, onde a utilização de métodos exatos para a resolução de problemas de dimensão elevada pode ser bastante custosa computacionalmente, este trabalho propõe um algoritmo heurístico para resolução

do OASP. O método desenvolvido extende o algoritmo sugerido em (Subramanian *et al.*, 2014) que apresentou resultados bastante competitivos para o $1|s_{jk}|\sum(w_jT_j)$.

O trabalho está organizado como se segue. A Seção 2 descreve formalmente o problema. A Seção 3 mostra uma formulação matemática para o OASP, proposta por Oğuz *et al.* (2010). A Seção 4 apresenta o algoritmo proposto para a resolução do OASP. Os resultados obtidos a partir da aplicação do método proposto são ilustrados na Seção 5. Por fim, a Seção 6 traz as considerações finais e sugestões para trabalhos futuros.

2 Descrição do problema

O OASP é uma generalização do problema de sequenciamento da produção em uma máquina, objetivando a minimização da soma dos atrasos ponderados, também conhecido como $1||\sum(w_jT_j)$, de acordo com a classificação $(\alpha|\beta|\gamma)$ proposta por Graham *et al.* (1979). O $1||\sum(w_jT_j)$, por sua vez, foi provado ser $\mathcal{NP}$–difícil (Lenstra *et al.*, 1977). Desta forma, o OASP também está enquadrado na classe de problemas $\mathcal{NP}$–difícil.

O problema pode ser definido da seguinte forma: para um ambiente envolvendo uma máquina, é dado um conjunto de ordens de produção O, tal que, para cada ordem $i \in O$, é conhecido seu tempo de processamento p_i, sua *release date* r_i, data de entrega (*due date*) d_i, data limite (*deadline*) $\bar{d}_i$, tal que $d_i \leq \bar{d}_i$, receita máxima e_i, peso w_i, além dos tempos de *setup* s_{ij} para todas as outras ordens $j \in O$, de modo que s_{ij} pode ser diferente s_{ji}. Se uma ordem i for aceita para ser processada, ela deve ser, obrigatoriamente, finalizada antes do seu *deadline*. Caso essa ordem i seja finalizada entre sua *due date* e seu *deadline*, é aplicada, à receita máxima e_i, uma penalidade w_i por unidade de tempo passada da *due date*, de maneira que em $\bar{d}_i$ essa receita é zero.

O OASP busca definir quais ordens devem ser processadas e em qual sequência, com o objetivo de maximizar a lucro total. Para uma determinada sequência σ_S, é possível calcular, para cada $i \in S$, tal que $S \subseteq O$, seus respectivos tempos de términos C_i, de modo que $C_i = \sum_{k=1}^{i} p_k$. De posse dos tempos de términos e das *due dates* de cada ordem $i \in S$ é possível calcular seus atrasos T_i, definidos como $T_i = max\{0, C_i - d_i\}$. Os atrasos T_i, os pesos w_i e as receitas máximas e_i permitem o cálculo da receita obtida por cada ordem $i \in S$. Para uma dada sequência σ_S, a sua receita total é dada por: $R(\sigma_S) = \sum_{i \in S} max\{0, e_i - w_iT_i\}$.

3 Formulação matemática

Nesta seção é apresentada uma formulação matemática baseada em programação linear inteira para o OASP. A formulação aqui apresentada é a mesma proposta no trabalho de Oğuz *et al.* (2010). São definidas duas variáveis binárias para o problema, y_{ij} e I_i. A primeira relacionada com o sequenciamento e a segunda com a aceitação ou não da ordem, a saber:

$$y_{ij} = \begin{cases} 1, & \text{se a ordem } i \text{ precede a ordem } j \qquad i,j \in O, i \neq j \\ 0, & \text{caso contrário} \end{cases} \tag{1}$$

$$I_i = \begin{cases} 1, & \text{se a ordem } i \text{ for selecionada} \qquad i \in O \\ 0, & \text{caso contrário} \end{cases} \tag{2}$$

Para a implementação do modelo matemático foram definidas duas tarefas *dummy*, a tarefa 0 e a tarefa $n+1$, onde 0 representa a primeira tarefa a ser processada e $n+1$ representa a última tarefa da sequência a ser processada. Os dados associados às tarefas *dummy* foram definidos de forma a não interferir na na solução real do problema. A formulação encontra-se descrita a seguir.

$$z = max \sum_{i=1}^{n} R_i \tag{3}$$

Sujeito a:

$$\sum_{j=1,j\neq i}^{n+1} y_{ij} = I_i \qquad \forall i = 0,\ldots,n \qquad (4)$$

$$\sum_{j=0,j\neq i}^{n} y_{ij} = I_i \qquad \forall i = 1,\ldots,n+1 \qquad (5)$$

$$C_i + (s_{ij} + p_j)y_{ij} + \bar{d}_i(y_{ij} - 1) \leq C_j \qquad \forall i = 0,\ldots,n; \forall j = 1,\ldots,n+1; i \neq j \qquad (6)$$

$$(r_j + p_j)I_j + s_{ij}y_{ij} \leq C_j \qquad \forall i = 0,\ldots,n; \forall j = 1,\ldots,n+1; i \neq j \qquad (7)$$

$$C_i \leq \bar{d}_i I_i \qquad \forall i = 0,\ldots,n+1 \qquad (8)$$

$$T_i \geq C_i - d_i \qquad \forall i = 0,\ldots,n+1 \qquad (9)$$

$$T_i \leq (\bar{d}_i - d_i)I_i \qquad \forall i = 0,\ldots,n+1 \qquad (10)$$

$$T_i \geq 0 \qquad \forall i = 0,\ldots,n+1 \qquad (11)$$

$$R_i \leq e_i I_i - w_i T_i \qquad \forall i = 1,\ldots,n \qquad (12)$$

$$R_i \geq 0 \qquad \forall i = 0,\ldots,n \qquad (13)$$

$$C_0 = 0, C_{n+1} = \max_{i=1,\ldots n} \bar{d}_i \qquad (14)$$

$$I_0 = 1, I_{n+1} = 1 \qquad (15)$$

$$I_i \in \{0,1\}, y_{ij} \in \{0,1\} \qquad \forall i = 0,\ldots,n; \forall j = 1,\ldots,n+1, i \neq j \qquad (16)$$

A função objetivo (3) maximiza a receita total. As restrições (4) e (5) estabelecem que, se uma tarefa é processada, essa tarefa é precedida e sucedida por uma única tarefa. As restrições (6) indicam que, caso uma tarefa j seja precedida por uma i, então o tempo de conclusão da tarefa j deverá ser maior que o tempo de conclusão da tarefa i (C_i), somado o tempo de *setup* entre as tarefas i e j (s_{ij}) mais o tempo de processamento da tarefa j (p_j).

As restrições (7) asseguram que, se uma tarefa j é aceita e é precedida por uma tarefa i no sequenciamento, então o tempo de conclusão da tarefa j (C_j), deve ser, no mínimo, seu *release date* (r_j) somado ao tempo de *setup* entre as tarefas (s_{ij}) mais o tempo de processamento da tarefa j (p_j). No caso em que a tarefa i não preceda a tarefa j, o tempo de conclusão da tarefa j é dado pelo *release time* da tarefa j, somado ao seu tempo de processamento (p_j). Se a tarefa j não é aceita, o limite se reduz a $C_j \geq 0$. As restrições (8) garantem que qualquer tarefa que seja completada após sua data limite de entrega (*deadline*), não devem ser aceitas. As restrições (9) definem o atraso de cada tarefa, juntamente com os limites estabelecidos nas restrições (10) e (11). As restrições (12) calculam a receita associada à produção da tarefa i quando ela é aceita, levando-se em conta as penalidades por atraso, caso hajam. Por fim, as restrições (14) e (15) especificam os tempos de conclusão das tarefas *dummy* 0 e $n+1$, enquanto as restrições (16) definem as variáveis binárias do problema.

4 Algoritmo Proposto

Esta seção descreve o algoritmo *Iterated Local Search* proposto. Para um maior entendimento da meta-heurística ILS ver Lourenço *et al.* (2003). A ideia principal do método é combinar um estágio de intensificação, ou seja, uma fase de busca local que é responsável por encontrar um ótimo local, e um estágio de diversificação, que tem como função reconstruir, aleatoriamente, parte da melhor solução encontrada. Apesar de ser um algoritmo simples, implementações bem sucedidas em uma grande variedade de problemas possibilitou o alcanço de um balanço entre as fases de intensificação e diversificação, resultando em soluções de alta qualidade.

O método utilizado neste trabalho é referido aqui como ILS-OAS, foi baseado no método utilizado por Subramanian *et al.* (2014) para o $1|s_{jk}|\sum(w_jT_j)$ e é apresentado posteriormente na Subseção 4.2.

4.1 Representação da solução

Uma solução para o OASP consiste em um conjunto de ordens selecionadas dentre o conjunto de todas as ordens e dispostas na sequência em que estas ordens serão processadas. A solução é aqui representada por um vetor de $n + 2$ posições, onde $n = |O|$ representa o número total de ordens.

Devido a presença de tempos de *setup*, optou-se pela inserção de duas ordens auxiliares, cujos tempos de processamento de *setup* para todas as outras ordens é zero. Tais ordens ocupam, obrigatoriamente a primeira e a última posição do vetor. Logo, cada posição no vetor representa uma ordem. Como o problema envolve a decisão de quais ordens aceitar, as ordens não processadas são aquelas em que seus tempos de término ultrapassam seus respectivos *deadlines*. Usualmente, estas ordens estão nas posições finais do vetor, de modo que a partir da primeira ordem não aceita, as subsequentes também não serão aceitas para o processamento, e com isso não contribuindo com a função objetivo do problema.

No exemplo a seguir, percebe-se que a ordem 2 é finalizada entre a sua *due date* e seu *deadline*, logo sua receita total será penalizada em $w_2(C_2 - d_2)$, onde C_2 é o tempo de término (*completion time*) da ordem 2. No mesmo exemplo, a ordem 1 é finalizada após seu *deadline*, ou seja, admite-se que esta ordem não foi selecionada para ser processada.

j	r_j	p_j	d_j	$\bar{d}_j$	w_j	e_j
1	3	3	8	10	2	4
2	5	2	7	9	1	2
3	0	2	7	9	2	4
4	0	3	9	10	3	3
5	1	1	6	7	1	1

Figura 1: Exemplo – representação da solução

4.2 ILS-OAS

A heurística *multi-start* ILS-OAS combina características da meta-heurística ILS com um procedimento de busca local baseado no método *Randomized Variable Neighborhood Descent* (RVND) (Mladenović e Hansen, 1997; Subramanian *et al.*, 2010). Dois parâmetros são requeridos pelo ILS-OAS, sendo eles o $MaxIter$, que representa o número de reinicializações do método, e o $MaxIterILS$ que determina o número de iterações sem melhoras do ILS que define quando a iteração corrente do ILS termina. As principais etapas do ILS-OAS são descritas a seguir:

Passo 0: Seja $iter$ a iteração atual. Se $iter \leq MaxIter$, então gere uma solução inicial pela escolha de uma estratégia e um critério de inserção aleatoriamente e execute a busca local usando o método RVND considerando todas as estruturas de vizinhanças. Caso contrário, pare.

Passo 1: Seja $iterILS$ o número de perturbações sem melhoras na solução. Enquanto $iterILS \leq MaxIterILS$, execute o mecanismo de perturbação e a busca local RVND. Caso a solução seja melhor que a melhor solução do ILS até então, atualize-a e faça $iterILS = 0$. Caso contrário, incremente $iterILS$.

Passo 2: Caso a melhor solução do passo anterior for melhor que a melhor solução global, atualize-a. Caso contrário, descarte-a. Retorne ao Passo 0.

4.3 Solução Inicial

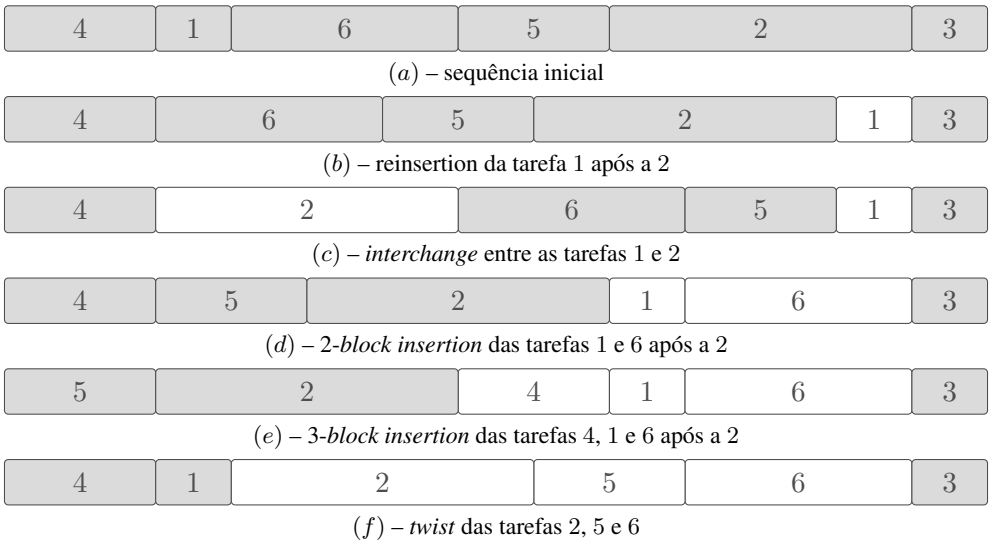
Cada iteração do ILS-OAS começa pela construção da solução inicial, gerada por meio de uma heurística baseada em um critério guloso de inserção que funciona da forma que se segue.

Inicialmente são selecionados, aleatoriamente, 10% das ordens que formarão uma sequência de ordem arbitrária. Em seguida, a esta sequência é adicionada, a cada iteração, uma nova ordem usando uma regra em que a ordem escolhida para ser adicionada na sequência é aquela em que o aumento no *setup* total da sequência é minimizado. Finalmente, uma solução real do problema pode ser obtida conforme descrito na Subseção 4.1. O processo é repetido até que a sequência atual contenha todas as n ordens. Tal processo de geração da solução inicial tem complexidade $O(n^2)$.

4.4 Busca Local

O algoritmo ILS-OAS aplica a busca local na solução inicial e em cada solução obtida pelo processo de diversificação, aqui representado pelo método de perturbação. A busca local realizada pelo método RVND procura, a cada busca na vizinhança, o melhor vizinho. Ao invés de definir uma ordem de investigação das vizinhanças, o RVND proporciona uma escolha aleatória dentro de um conjunto de vizinhanças a ser explorada. Estas vizinhanças são as clássicas e bem conhecidas da literatura dos problemas de sequenciamento da produção. Tais vizinhanças são:

- insertion – Uma tarefa é removida de sua posição atual e inserida em outra posição da mesma sequência;
- *interchange* – Duas tarefas tem suas posições trocadas, uma pela outra.
- *2-block insertion* – Um bloco de duas tarefas consecutivas são removidas e inseridas em outra posição da mesma sequência.
- *3-block insertion* – Um bloco de três tarefas consecutivas são removidas e inseridas em outra posição da mesma sequência.
- *twist* – A ordem das tarefas pertencentes a um bloco de tarefas é invertida. Considerou-se apenas movimentos envolvendo até 10 tarefas.



(a) – sequência inicial

(b) – reinsertion da tarefa 1 após a 2

(c) – *interchange* entre as tarefas 1 e 2

(d) – 2-*block insertion* das tarefas 1 e 6 após a 2

(e) – 3-*block insertion* das tarefas 4, 1 e 6 após a 2

(f) – *twist* das tarefas 2, 5 e 6

Figura 2: Vizinhanças

As estruturas de vizinhanças utilizadas na fase de busca local geram, na maiorias delas, $O(n^2)$ vizinhos, de modo que a complexidade total para a busca na vizinhança é de $O(n^3)$, dada a necessidade de $O(n)$ para a avaliação da solução. Quando uma solução melhor é encontrada na fase de busca local ela se torna a nova solução atual, então esta solução (s') passa pelo processo de diversificação na próxima iteração. Desta maneira, a melhor solução ao fim de todas as iterações (s^*) é retornada como a solução encontrada pelo algoritmo ILS-OAS.

4.5 Diversificação

Uma vez obtida uma solução ótima local, com relação às vizinhanças aqui consideradas, um processo de diversificação é então aplicado. No ILS-OAS foi adotado um mecanismo denominado *Double-Bridge*, proposto por Martin *et al.* (1991), amplamente empregado no Problema do Caixeiro Viajante (PCV). No *Double-Bridge*, quatro "arcos" aleatórias de uma dada solução são deletadas e então reconectadas de forma a gerar uma nova sequência. No ILS-OAS, a perturbação consiste em um simples movimento, aleatório, na vizinhança *Double-Bridge*, o que é suficiente para garantir um grau de diversificação adequado.

5 Resultados

O algoritmo ILS-OAS foi implementado em linguagem de programação C++ e executado em um PC Intel Core i7 3.40 GHz com 16 GB de memória RAM e sistema operacional Linux Mint 13. Apenas uma *thread* foi utilizada nos experimentos e o algoritmo foi executado 10 vezes em cada instância. Com base no trabalho de Subramanian *et al.* (2014), os parâmetros do ILS-OAS foram definidos como: $MaxIter = 10$ e $MaxIterILS = 4n$.

As instâncias do problema foram propostas por Oğuz *et al.* (2010) e estão disponibilizadas em `http://home.ku.edu.tr/~coguz/`. Para a geração das instâncias, foram determinados os tamanhos de instâncias, em particular, $n = 10, 15, 20, 25, 50, 100$, e utilizados dois parâmetros para variar as características de cada um dos conjuntos de instâncias. O primeiro, *R*, está associado à variação das *due dates*, enquanto o segundo, τ, indica o quanto o *deadline* está afastado da *due date*. Foram adotados $\tau = 0.1, 0.3, 0.5, 0.7, 0.9$ e $R = 0.1, 0.3, 0.5, 0.7, 0.9$, resultando em 25 combinações de τ e *R*, contendo, cada uma, um conjunto de 10 instâncias, totalizando 1500 instâncias ao todo.

Os resultados apresentados nas tabelas a seguir ilustram o desempenho do algoritmo proposto. As tabelas apresentam os resultados de todas os grupos instâncias por tamanho (n), τ e R. Para melhor comparação dos resultados obtidos, foram considerados dois resultados para o algoritmo proposto. O primeiro, o melhor resultado dentre as 10 execuções, e o segundo, dado pela média das 10 execuções. Devido a grande quantidade de instâncias, os resultados foram apresentados por grupo de instâncias de maneira similar a Cesaret *et al.* (2012), indicando o menor e maior *Gap* do grupo, além do *Gap* médio entre elas. Por exemplo, para o conjunto de 10 instâncias de tamanho 20, $\tau = 0.1$ e $R = 0.1$, foram reportados o menor e o maior *Gap* do grupo, além do *Gap* médio do conjunto. Vale ressaltar que o *Gap* representa desvio do valor das soluções obtidas em relação ao limite superior reportado em `http://home.ku.edu.tr/~coguz/`. É importante frisar que Cesaret *et al.* (2012) compilaram seus resultados a partir de apenas uma execução por instância em um *workstation* Intel Xeon 3.00 GHz e 4 GB de memória RAM. Assim sendo, torna-se difícil realmente avaliar a robustez do algoritmo desenvolvido pelos autores em cada grupo, uma vez que este possui componentes aleatórios.

Observando as Tabelas 1-6 é possível verificar que o algoritmo proposto obteve, em média, um resultado superior ao de Cesaret *et al.* (2012), no que diz respeito a qualidade das soluções encontradas. Um total de 405 soluções ótimas foram obtidas, contra 367 de Cesaret *et al.* (2012). Além disso, o ILS-OAS foi capaz de melhorar o resultado de 779 instâncias e igualar o resultado de outras 554. Por outro lado, o tempo médio de execução do ILS-OAS foi bastante elevado quando comparado ao algoritmo Busca Tabu de Cesaret *et al.* (2012).

6 Conclusões e trabalhos futuros

Este trabalho abordou o *Order Acceptance and Scheduling Problem* (OASP). Para resolvê-lo foi proposto um algoritmo heurístico baseado em *Iterated Local Search* (ILS). Os resultados alcançados pelo algoritmo desenvolvido, denominado ILS-OAS, mostraram-se competitivos quando comparado ao melhor algoritmo conhecido para o problema. Resultados de qualidade superior foram encontrados, porém com custo computacional bastante elevado, sobretudo nas

instâncias de 100 tarefas. Como trabalhos futuros, pretende-se utilizar estruturas de dados mais avançadas com o intuito de reduzir a complexidade da busca local, além de examinar estratégias exatas para resolução do problema.

Tabela 1: Resultados para $n = 10$

n =10		Gap (%)									# soluções ótimas			tempo (s)	
		TS (1 exec.)			ILS-OAS (Melhor)			ILS-OAS (Média)			TS	ILS-OAS	ILS-OAS		
τ	R	Min (%)	Média (%)	Max (%)	Min (%)	Média (%)	Max (%)	Min (%)	Média (%)	Max (%)	(1 exec.)	(Melhor)	(Média)	TS	ILS-OAS
0,1	0,1	0,09	0,84	2,61	0,09	0,34	1,68	0,09	0,34	1,68	6	8	8	<0,01	0,01
	0,3	0,09	1,31	3,43	0,09	0,12	0,36	0,09	0,12	0,36	5	9	9	<0,01	0,01
	0,5	0,00	0,84	7,50	0,00	0,84	7,50	0,00	0,84	7,50	8	8	8	<0,01	0,01
	0,7	0,00	0,35	2,68	0,00	0,43	3,74	0,00	0,43	3,74	8	9	9	0,01	0,02
	0,9	0,00	0,24	1,24	0,00	0,31	1,79	0,00	0,31	1,79	7	8	8	<0,01	0,01
0,3	0,1	0,08	0,25	1,68	0,08	0,09	0,10	0,08	0,09	0,10	9	10	10	<0,01	0,01
	0,3	0,09	0,76	2,13	0,09	0,09	0,12	0,09	0,09	0,12	6	10	10	<0,01	0,01
	0,5	0,03	2,11	7,89	0,03	0,27	1,98	0,03	0,27	1,98	6	9	9	0,01	0,01
	0,7	0,08	0,63	2,39	0,08	0,41	1,27	0,08	0,44	1,27	6	7	6	<0,01	0,02
	0,9	0,08	0,86	3,39	0,00	0,34	1,40	0,00	0,37	1,40	4	6	5	<0,01	0,02
0,5	0,1	0,00	0,65	6,06	0,00	0,06	0,09	0,00	0,06	0,09	8	10	10	<0,01	0,01
	0,3	0,01	0,95	5,07	0,01	0,45	3,85	0,01	0,45	3,85	8	9	9	<0,01	0,02
	0,5	0,00	0,08	0,27	0,02	0,27	1,92	0,02	0,27	1,92	9	8	8	<0,01	0,02
	0,7	0,00	0,27	2,15	0,00	1,31	5,39	0,00	1,31	5,39	4	4	4	<0,01	0,02
	0,9	0,00	0,23	1,62	0,00	0,51	4,42	0,00	0,51	4,42	7	9	9	<0,01	0,02
0,7	0,1	0,00	0,43	4,26	0,00	0,12	1,20	0,00	0,12	1,20	8	9	9	<0,01	0,02
	0,3	0,00	0,00	0,01	0,00	0,00	0,01	0,00	0,00	0,01	10	10	10	<0,01	0,01
	0,5	0,00	0,13	1,26	0,00	0,00	0,04	0,00	0,00	0,04	9	10	10	<0,01	0,02
	0,7	0,00	0,01	0,08	0,00	0,25	2,36	0,00	0,25	2,36	8	9	9	<0,01	0,02
	0,9	0,00	0,37	3,60	0,00	0,16	1,40	0,00	0,16	1,40	6	9	9	<0,01	0,02
0,9	0,1	0,00	0,00	0,00	0,00	0,50	5,00	0,00	0,50	5,00	10	9	9	<0,01	0,01
	0,3	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	9	10	10	<0,01	0,02
	0,5	0,00	0,00	0,01	0,00	0,00	0,01	0,00	0,00	0,01	8	10	10	<0,01	0,02
	0,7	0,00	0,00	0,00	0,00	0,23	2,31	0,00	0,23	2,31	9	9	9	<0,01	0,02
	0,9	0,00	0,00	0,00	0,00	0,15	1,46	0,00	0,15	1,46	8	9	9	<0,01	0,02

Tabela 2: Resultados para $n = 15$

n =15		Gap (%)									# soluções ótimas			tempo (s)	
		TS (1 exec.)			ILS-OAS (Melhor)			ILS-OAS (Média)			TS	ILS-OAS	ILS-OAS		
τ	R	Min (%)	Média (%)	Max (%)	Min (%)	Média (%)	Max (%)	Min (%)	Média (%)	Max (%)	(1 exec.)	(Melhor)	(Média)	TS	ILS-OAS
0,1	0,1	1,19	2,70	3,57	0,00	2,37	3,55	0,00	2,37	3,55	0	1	1	0,04	0,05
	0,3	0,00	3,19	8,50	1,26	3,20	6,54	1,26	3,20	6,54	1	0	0	0,01	0,05
	0,5	0,00	1,71	3,65	1,00	2,02	4,38	1,00	2,04	4,38	1	0	0	0,01	0,06
	0,7	0,00	1,77	4,45	0,00	1,79	4,86	0,00	1,79	4,86	1	2	2	0,01	0,06
	0,9	0,00	1,32	6,08	0,00	1,16	6,08	0,00	1,16	6,08	4	5	5	0,01	0,06
0,3	0,1	3,36	4,42	8,05	0,00	3,08	5,75	0,84	3,20	6,09	0	1	0	0,01	0,06
	0,3	1,57	4,92	11,33	0,59	3,95	11,33	0,59	3,95	11,33	0	0	0	0,01	0,07
	0,5	1,26	4,61	8,04	1,26	4,52	7,76	1,26	4,57	7,76	0	0	0	0,01	0,07
	0,7	1,55	3,97	7,57	1,17	3,07	7,57	1,17	3,22	7,57	0	0	0	0,01	0,08
	0,9	0,00	3,79	7,10	0,00	3,17	7,10	0,00	3,37	7,74	2	2	2	0,01	0,08
0,5	0,1	2,47	7,48	12,56	2,47	7,00	12,56	2,47	7,00	12,56	0	0	0	0,01	0,08
	0,3	4,26	7,68	13,73	3,05	7,29	13,73	3,05	7,39	13,73	0	0	0	0,01	0,08
	0,5	5,81	9,35	15,29	4,07	9,17	15,29	4,07	9,30	15,29	0	0	0	0,01	0,09
	0,7	1,16	6,47	11,00	1,16	6,20	10,05	1,16	6,38	10,81	0	0	0	0,01	0,09
	0,9	0,10	6,36	18,49	0,10	6,65	18,39	0,10	6,77	18,39	1	1	1	0,01	0,10
0,7	0,1	0,00	0,70	4,06	0,00	0,23	1,65	0,00	0,23	1,65	7	9	9	0,01	0,08
	0,3	0,07	0,57	2,78	0,06	0,09	0,10	0,06	0,14	0,63	7	10	9	0,01	0,08
	0,5	0,07	0,43	2,40	0,07	0,34	2,00	0,07	0,42	2,00	7	8	6	0,01	0,09
	0,7	0,08	2,05	8,28	0,06	1,57	8,28	0,09	1,90	8,28	3	7	4	0,01	0,09
	0,9	0,08	0,11	0,27	0,08	0,11	0,27	0,08	0,19	0,71	9	9	8	0,01	0,10
0,9	0,1	0,00	0,27	2,70	0,00	0,21	2,12	0,00	0,63	4,00	9	9	8	0,01	0,07
	0,3	0,00	0,47	4,59	0,00	0,21	1,38	0,00	0,27	1,38	9	8	8	0,01	0,08
	0,5	0,00	0,03	0,08	0,00	0,08	0,51	0,00	0,36	1,48	10	9	6	0,01	0,10
	0,7	0,00	0,03	0,10	0,00	0,22	1,63	0,00	0,43	2,27	10	8	8	0,01	0,09
	0,9	0,00	0,05	0,10	0,00	0,05	0,10	0,00	0,31	1,36	10	10	7	0,01	0,10

Tabela 3: Resultados para $n = 20$

n =20		Gap (%)									# soluções ótimas			tempo (s)	
		TS (1 exec.)			ILS-OAS (Melhor)			ILS-OAS (Média)			TS	ILS-OAS	ILS-OAS		
τ	R	Min (%)	Média (%)	Max (%)	Min (%)	Média (%)	Max (%)	Min (%)	Média (%)	Max (%)	(1 exec.)	(Melhor)	(Média)	TS	ILS-OAS
0,1	0,1	1,45	3,16	4,57	0,44	2,31	4,11	0,71	2,67	4,47	0	0	0	0,1	0,17
	0,3	0,82	3,27	5,02	0,82	2,02	2,82	0,82	2,19	2,88	0	0	0	0,04	0,18
	0,5	0,82	1,91	3,29	0,77	1,53	3,04	0,77	1,63	3,04	0	0	0	0,03	0,19
	0,7	0,00	0,64	1,92	0,00	0,46	1,32	0,00	0,46	1,32	5	6	6	0,04	0,18
	0,9	0,00	1,16	3,91	0,00	0,54	3,04	0,00	0,65	3,65	3	7	6	0,03	0,18
0,3	0,1	1,65	4,65	7,65	0,00	3,63	7,45	1,89	4,12	7,45	0	1	0	0,03	0,20
	0,3	3,17	4,75	6,22	1,44	3,60	5,33	1,59	3,99	6,11	0	0	0	0,04	0,21
	0,5	1,93	5,21	7,61	0,81	3,89	7,00	1,47	4,38	7,00	0	0	0	0,03	0,24
	0,7	1,29	3,27	6,96	0,00	2,35	6,96	0,94	2,67	6,96	0	1	0	0,04	0,25
	0,9	0,00	2,13	4,69	0,00	1,84	4,69	0,00	2,15	5,25	2	3	2	0,04	0,25
0,5	0,1	3,61	5,75	7,69	3,61	4,35	5,53	3,61	4,98	6,24	0	0	0	0,03	0,24
	0,3	2,83	6,32	9,16	1,42	5,40	9,16	1,65	5,74	9,52	0	0	0	0,07	0,27
	0,5	3,09	6,91	10,09	2,88	5,65	9,21	3,32	6,11	9,87	0	0	0	0,04	0,27
	0,7	1,21	5,32	10,70	1,16	4,75	10,70	1,39	5,13	10,70	0	0	0	0,03	0,31
	0,9	0,00	6,25	10,57	0,00	5,80	10,57	0,00	6,17	10,90	1	1	1	0,05	0,32
0,7	0,1	5,47	9,50	14,85	5,47	8,89	13,54	5,47	9,16	14,06	0	0	0	0,05	0,24
	0,3	5,94	9,21	14,29	5,94	8,36	11,76	5,94	8,70	12,18	0	0	0	0,04	0,27
	0,5	0,09	9,13	20,18	0,09	8,67	20,18	0,09	8,91	20,18	2	2	1	0,04	0,29
	0,7	0,09	6,71	12,17	0,09	6,72	12,17	0,09	7,05	12,87	3	2	2	0,07	0,31
	0,9	0,00	7,90	12,83	0,00	7,16	12,50	0,00	7,51	12,69	1	1	1	0,04	0,35
0,9	0,1	0,00	0,12	1,21	0,00	0,00	0,01	0,00	0,43	1,67	0	10	4	0,06	0,26
	0,3	0,00	0,12	0,74	0,00	0,58	3,12	0,00	0,86	3,30	8	8	5	0,03	0,28
	0,5	0,00	0,48	2,33	0,00	0,15	0,74	0,00	0,26	0,99	8	9	8	0,03	0,29
	0,7	0,00	0,44	1,48	0,00	0,11	0,29	0,00	0,31	1,32	8	8	7	0,03	0,33
	0,9	0,00	1,94	13,21	0,00	1,63	13,21	0,09	2,02	13,21	6	6	3	0,03	0,33

Tabela 4: Resultados para $n = 25$

n =25		Gap (%)									# soluções ótimas			tempo (s)	
		TS (1 exec.)			ILS-OAS (Melhor)			ILS-OAS (Média)			TS	ILS-OAS	ILS-OAS		
τ	R	Min (%)	Média (%)	Max (%)	Min (%)	Média (%)	Max (%)	Min (%)	Média (%)	Max (%)	(1 exec.)	(Melhor)	(Média)	TS	ILS-OAS
0,1	0,1	1,09	4,02	6,28	1,09	2,24	3,51	1,31	2,81	3,52	0	0	0	0,08	0,40
	0,3	1,61	3,31	6,41	0,69	2,07	5,10	1,27	2,41	5,58	0	0	0	0,06	0,38
	0,5	0,87	2,25	4,29	0,00	1,11	1,90	0,63	1,37	2,02	0	1	0	0,06	0,40
	0,7	0,00	1,19	4,07	0,00	0,53	1,67	0,00	0,69	2,17	4	6	4	0,06	0,37
	0,9	0,00	0,94	2,11	0,00	0,28	2,09	0,00	0,49	2,09	4	8	5	0,05	0,39
0,3	0,1	1,71	3,66	5,24	1,71	3,09	4,66	1,92	3,47	4,79	0	0	0	0,1	0,48
	0,3	2,61	4,89	7,33	2,15	3,25	5,09	2,15	3,47	5,21	0	0	0	0,09	0,55
	0,5	1,74	2,99	5,59	0,89	2,03	3,50	1,34	2,42	4,34	0	0	0	0,08	0,50
	0,7	1,06	2,48	5,69	0,74	2,08	4,96	0,92	2,32	5,28	0	0	0	0,08	0,53
	0,9	0,00	1,85	4,05	0,00	1,34	2,96	0,00	1,48	3,30	2	2	2	0,07	0,57
0,5	0,1	2,80	6,35	7,37	1,60	4,64	6,76	2,44	5,32	6,76	0	0	0	0,07	0,56
	0,3	3,17	5,49	9,24	2,85	4,46	7,29	3,03	5,02	8,02	0	0	0	0,08	0,61
	0,5	1,94	5,28	7,89	0,97	4,84	7,52	1,17	5,22	7,91	0	0	0	0,09	0,67
	0,7	1,60	5,70	10,89	0,64	4,35	7,97	1,15	4,98	8,29	0	0	0	0,08	0,70
	0,9	1,03	4,32	7,45	1,03	3,23	6,67	1,10	3,88	8,17	0	0	0	0,08	0,76
0,7	0,1	3,24	9,28	18,24	2,31	8,10	16,22	2,82	8,87	18,14	0	0	0	0,06	0,60
	0,3	7,22	9,74	14,11	5,78	8,80	12,40	6,35	9,57	12,86	0	0	0	0,08	0,67
	0,5	6,61	11,51	14,75	5,37	10,35	14,03	6,43	11,05	14,34	0	0	0	0,08	0,74
	0,7	1,69	7,86	13,75	1,58	6,58	12,08	1,69	7,56	13,02	0	0	0	0,07	0,77
	0,9	2,55	10,24	14,64	1,07	8,48	13,64	2,45	9,27	14,44	0	0	0	0,08	0,87
0,9	0,1	0,00	1,28	6,40	0,00	0,63	3,45	0,00	0,98	5,02	5	6	4	0,06	0,64
	0,3	0,00	0,07	0,44	0,00	0,01	0,10	0,00	0,41	1,10	8	10	3	0,06	0,68
	0,5	0,00	3,51	12,43	0,00	2,49	12,30	0,00	3,15	12,30	3	6	1	0,07	0,71
	0,7	0,00	7,53	24,81	0,00	6,88	20,99	0,01	7,26	21,90	4	4	3	0,08	0,72
	0,9	0,00	6,77	22,27	0,00	6,07	19,10	0,30	6,69	20,96	4	2	0	0,09	0,81

Tabela 5: Resultados para $n = 50$

n =50		Gap (%)									# soluções ótimas			tempo (s)	
τ	R	TS (1 exec.)			ILS-OAS (Melhor)			ILS-OAS (Média)			TS	ILS-OAS	ILS-OAS	TS	ILS-OAS
		Min (%)	Média (%)	Max (%)	Min (%)	Média (%)	Max (%)	Min (%)	Média (%)	Max (%)	(1 exec.)	(Melhor)	(Média)		
0,1	0,1	1,20	2,23	3,00	1,03	1,65	2,40	1,50	1,97	2,83	0	0	0	1,19	6,65
	0,3	1,39	2,44	3,92	1,06	1,49	1,79	1,16	1,84	2,18	0	0	0	0,98	6,64
	0,5	1,12	1,61	2,10	0,34	0,75	1,20	0,38	1,04	1,64	0	0	0	0,89	6,13
	0,7	0,00	2,45	16,39	0,00	1,87	16,39	0,00	1,99	16,39	1	4	2	0,58	5,26
	0,9	0,00	0,54	1,90	0,00	0,00	0,00	0,00	0,01	0,10	2	10	10	0,46	4,42
0,3	0,1	1,58	2,63	3,48	1,38	2,49	3,82	1,97	2,99	4,16	0	0	0	1,61	7,77
	0,3	2,66	3,54	5,05	1,55	2,43	3,23	2,31	3,11	3,99	0	0	0	1,49	8,24
	0,5	1,33	2,56	4,84	0,75	2,00	4,26	1,03	2,31	4,83	0	0	0	1,16	8,12
	0,7	0,00	1,40	3,44	0,00	0,76	2,25	0,00	1,09	2,62	1	2	1	0,85	7,98
	0,9	0,40	1,35	2,85	0,00	0,46	1,83	0,00	0,64	2,02	0	5	3	0,68	7,02
0,5	0,1	2,82	4,15	5,34	2,48	3,41	4,20	2,81	3,87	4,76	0	0	0	1,36	9,24
	0,3	3,08	5,69	8,02	2,67	4,44	6,87	3,24	5,12	7,29	0	0	0	1,18	10,56
	0,5	1,55	4,42	8,01	1,74	3,96	7,14	2,00	4,52	7,95	0	0	0	1,35	10,71
	0,7	1,51	3,10	5,16	0,95	2,54	4,60	1,51	3,11	5,19	0	0	0	1,22	10,77
	0,9	0,00	3,17	5,83	0.00	2,19	4,85	0.00	2,83	5,66	0	1	1	1,16	12,17
0,7	0,1	3,72	6,61	8,61	3,35	5,57	6,69	4,42	6,38	7,32	0	0	0	1,44	10,02
	0,3	4,25	7,26	10,77	3,17	5,93	9,36	4,41	7,03	10,14	0	0	0	1,52	11,22
	0,5	6,99	8,94	13,17	6,09	7,91	11,74	6,70	8,64	12,49	0	0	0	1,59	12,39
	0,7	2,41	9,23	17,72	2,41	8,73	18,23	4,05	9,83	19,15	0	0	0	1,27	14,82
	0,9	4,27	10,32	17,66	4,27	8,86	14,72	5,82	10,04	15,58	0	0	0	1,15	15,38
0,9	0,1	8,05	12,67	18,49	7,68	12,24	18,49	9,10	13,54	19,23	0	0	0	1,25	11,99
	0,3	11,62	16,33	20,66	11,74	15,18	20,20	13,37	16,96	22,16	0	0	0	1,26	13,38
	0,5	7,21	15,50	20,26	6,11	13,54	17,21	8,27	15,42	19,51	0	0	0	1,42	14,56
	0,7	7,75	13,85	21,05	6,20	12,53	19,01	7,23	14,13	21,08	0	0	0	1,43	15,67
	0,9	10,75	14,37	18,74	10,02	13,70	18,08	11,74	15,33	19,85	0	0	0	1,37	15,85

Tabela 6: Resultados para n =100

n =100		Gap (%)									# soluções ótimas			tempo (s)	
τ	R	TS (1 exec.)			ILS-OAS (Melhor)			ILS-OAS (Média)			TS	ILS-OAS	ILS-OAS	TS	ILS-OAS
		Min (%)	Média (%)	Max (%)	Min (%)	Média (%)	Max (%)	Min (%)	Média (%)	Max (%)	(1 exec.)	(Melhor)	(Média)		
0,1	0,1	1,18	2,44	3,28	1,21	1,46	1,60	1,49	1,72	2,11	0	0	0	16,76	120,20
	0,3	1,51	2,10	3,06	1,02	1,19	1,53	1,23	1,45	1,83	0	0	0	17,26	124,80
	0,5	0,36	1,39	2,74	0,18	0,56	0,80	0,56	0,80	1,03	0	0	0	10,87	107,47
	0,7	0,00	0,43	0,80	0,00	0,02	0,17	0,00	0,08	0,31	1	9	7	6,21	88,05
	0,9	0,00	0,22	0,89	0,00	0,00	0,00	0,00	0,01	0,14	4	10	10	3,53	62,32
0,3	0,1	1,47	2,65	3,96	0,98	1,99	2,71	1,49	2,38	2,98	0	0	0	22,37	142,38
	0,3	1,94	2,94	5,39	1,40	2,03	3,26	1,70	2,39	3,81	0	0	0	20,1	149,38
	0,5	1,14	2,21	3,81	1,27	1,76	2,54	1,49	2,08	2,73	0	0	0	16,77	154,77
	0,7	0,37	1,65	2,75	0,00	0,57	1,59	0,39	0,88	1,72	0	2	0	9,48	140,75
	0,9	0,19	0,91	2,41	0,00	0,37	1,02	0,00	0,58	1,37	0	4	3	7,58	129,25
0,5	0,1	2,49	3,96	5,27	2,69	3,40	4,10	2,99	3,94	4,88	0	0	0	25,96	173,08
	0,3	2,68	3,95	5,99	2,40	3,48	4,29	3,12	4,01	4,87	0	0	0	28,87	184,05
	0,5	2,57	3,54	4,53	1,83	3,46	4,56	2,51	3,94	4,92	0	0	0	20,56	191,80
	0,7	1,61	2,71	4,04	1,02	2,18	3,23	1,21	2,62	3,74	0	0	0	15,57	197,41
	0,9	0,67	2,10	4,65	0,87	1,50	2,86	1,00	1,94	3,76	0	0	0	12,15	203,76
0,7	0,1	2,70	4,98	6,42	4,02	4,82	5,64	4,49	5,42	6,62	0	0	0	33,6	203,95
	0,3	3,93	6,66	10,83	2,75	5,61	8,44	3,30	6,34	9,43	0	0	0	26,62	222,86
	0,5	4,04	6,45	12,85	3,94	6,14	12,77	4,29	6,75	13,16	0	0	0	22,3	243,96
	0,7	3,36	7,36	13,20	3,53	7,59	12,48	4,05	8,92	13,82	0	0	0	26,36	302,04
	0,9	5,30	8,40	12,95	4,76	8,20	10,36	5,43	9,55	12,36	0	0	0	17,84	315,28
0,9	0,1	7,30	9,03	11,95	7,46	8,79	12,52	7,95	10,09	13,53	0	0	0	29,46	286,46
	0,3	7,32	13,61	17,41	10,25	13,39	16,11	11,87	15,58	17,99	0	0	0	26,32	315,33
	0,5	10,80	15,68	18,38	10,92	15,55	20,08	12,49	17,82	21,70	0	0	0	21,51	327,52
	0,7	10,70	15,22	19,95	6,34	14,18	20,60	8,27	15,54	22,50	0	0	0	22,7	200,12
	0,9	11,23	15,50	22,21	8,23	15,41	21,64	10,63	17,09	22,52	0	0	0	17,48	200,79

Referências

Anghinolfi, D. e Paolucci, M. (2008), A new ant colony optimization approach for the single machine total weighted tardiness scheduling problem. *International Journal of Operations Research*, v. 5, p. 44–60.

Anghinolfi, D. e Paolucci, M. (2009), A new discrete particle swarm optimization approach for the single-machine total weighted tardiness scheduling problem with sequence-dependent setup times. *European Journal of Operational Research*, v. 193, n. 1, p. 73 – 85.

Cesaret, B., Oğuz, C. e Salman, F. S. (2012), A tabu search algorithm for order acceptance and scheduling. *Computers & Operations Research*, v. 39, n. 6, p. 1197 – 1205. Special Issue on Scheduling in Manufacturing Systems.

Graham, R., Lawler, E., Lenstra, J. e Kan, A. Optimization and approximation in deterministic sequencing and scheduling: a survey. P.L. Hammer, E. J. e Korte, B. (Eds.), *Discrete Optimization II Proceedings of the Advanced Research Institute on Discrete Optimization and Systems Applications of the Systems Science Panel of NATO and of the Discrete Optimization Symposium co-sponsored by IBM Canada and SIAM Banff, Aha. and Vancouver*, volume 5 of *Annals of Discrete Mathematics*, p. 287 – 326. Elsevier, 1979.

Kirlik, G. e Oguz, C. (2012), A variable neighborhood search for minimizing total weighted tardiness with sequence dependent setup times on a single machine. *Computers & Operations Research*, v. 39, n. 7, p. 1506 – 1520.

Lenstra, J., Kan, A. R. e Brucker, P. Complexity of machine scheduling problems. P.L. Hammer, E.L. Johnson, B. K. e Nemhauser, G. (Eds.), *Studies in Integer Programming*, volume 1 of *Annals of Discrete Mathematics*, p. 343 – 362. Elsevier, 1977.

Lopez, P. e Roubellat, F. *Production Scheduling*. ISTE. Wiley, 2008.

Lourenço, H. R., Martin, O. C. e Stützle, T. Iterated local search. Glover, F. e Kochenberger, G. A. (Eds.), *Handbook of Metaheuristics*, volume 57 of *International Series in Operations Research & Management Science*, p. 320–353. Springer US, 2003.

Martin, O., Otto, S. W. e Felten, E. W. (1991), Large-step markov chains for the traveling salesman problem. *Complex Systems*, v. 5, p. 299–326.

Mladenović, N. e Hansen, P. (1997), Variable neighborhood search. *Computers & Operations Research*, v. 24, n. 11, p. 1097 – 1100.

Nobibon, F. T., Herbots, J. e Leus, R. Order acceptance and scheduling in a single-machine environment: exact and heuristic algorithms. Blazewicz, J., Drozdowski, M., Kendall, G. e McCollum, B. (Eds.), *Proceedings of the 4th Multidisciplinary International Scheduling Conference: Theory and Applications (MISTA 2009), 10-12 Aug 2009, Dublin, Ireland*, p. 772–774. Abstract, 2009.

Nobibon, F. T. e Leus, R. (2011), Exact algorithms for a generalization of the order acceptance and scheduling problem in a single-machine environment. *Computers & Operations Research*, v. 38, n. 1, p. 367 – 378. Project Management and Scheduling.

Oğuz, C., Salman, F. S. e Yalçın, Z. B. (2010), Order acceptance and scheduling decisions in make-to-order systems. *International Journal of Production Economics*, v. 125, n. 1, p. 200 – 211.

Rom, W. O. e Slotnick, S. A. (2009), Order acceptance using genetic algorithms. *Computers & Operations Research*, v. 36, n. 6, p. 1758 – 1767.

Shabtay, D., Gaspar, N. e Kaspi, M. (2013), A survey on offline scheduling with rejection. *Journal of Scheduling*, v. 16, n. 1, p. 3–28.

Sioud, A., Gravel, M. e Gagné, C. (2012), A hybrid genetic algorithm for the single machine scheduling problem with sequence-dependent setup times. *Computers & Operations Research*, v. 39, n. 10, p. 2415 – 2424.

Slotnick, S. A. (2011), Order acceptance and scheduling: A taxonomy and review. *European Journal of Operational Research*, v. 212, n. 1, p. 1 – 11.

Slotnick, S. A. e Morton, T. E. (2007), Order acceptance with weighted tardiness. *Computers & Operations Research*, v. 34, n. 10, p. 3029 – 3042.

Subramanian, A., Drummond, L., Bentes, C., Ochi, L. e Farias, R. (2010), A parallel heuristic for the vehicle routing problem with simultaneous pickup and delivery. *Computers & Operations Research*, v. 37, n. 11, p. 1899 – 1911. Metaheuristics for Logistics and Vehicle Routing.

Subramanian, A., Battarra, M. e Potts, C. N. (2014), An iterated local search heuristic for the single machine total weighted tardiness scheduling problem with sequence-dependent setup times. *International Journal of Production Research*, v. 52, n. 9, p. 2729–2742.

Tanaka, S. e Araki, M. (2013), An exact algorithm for the single-machine total weighted tardiness problem with sequence-dependent setup times. *Computers & Operations Research*, v. 40, n. 1, p. 344 – 352.

Tasgetiren, M. F., Pan, Q.-K. e Liang, Y.-C. (2009), A discrete differential evolution algorithm for the single machine total weighted tardiness problem with sequence dependent setup times. *Computers & Operations Research*, v. 36, n. 6, p. 1900 – 1915.

Wang, X., Xie, X. e Cheng, T. (2013a), A modified artificial bee colony algorithm for order acceptance in two-machine flow shops. *International Journal of Production Economics*, v. 141, n. 1, p. 14 – 23. Meta-heuristics for manufacturing scheduling and logistics problems.

Wang, X., Xie, X. e Cheng, T. (2013b), Order acceptance and scheduling in a two-machine flowshop. *International Journal of Production Economics*, v. 141, n. 1, p. 366 – 376. Meta-heuristics for manufacturing scheduling and logistics problems.

Wei, J. e Ma, Y.-S. (2014), Design of a feature-based order acceptance and scheduling module in an ERP system. *Computers in Industry*, v. 65, n. 1, p. 64 – 78.

Xu, H., Lü, Z. e Cheng, T. (2013), Iterated local search for single-machine scheduling with sequence-dependent setup times to minimize total weighted tardiness. *Journal of Scheduling*, p. 1–17.

Zhong, X., Ou, J. e Wang, G. (2014), Order acceptance and scheduling with machine availability constraints. *European Journal of Operational Research*, v. 232, n. 3, p. 435 – 441.