FORMULAÇÕES COM ARCOS INDEXADOS NO TEMPO PARA O PROBLEMA DE SEQUENCIAMENTO EM UMA MÁQUINA COM MANUTENÇÕES PERIÓDICAS E TEMPOS DE *SETUP*

Vitor Luis Nesello, Anand Subramanian
Depart. de Engenharia de Produção - Centro de Tecnologia - Universidade Federal da Paraíba
Via Expressa Padré Zé, SN, Bloco H, Jardim Cidade Universitária, 58059-900, João Pessoa, PB
vitornesello@gmail.com, anand@ct.ufpb.br

RESUMO

Esse artigo trata de um problema de sequenciamento em apenas uma máquina com manutenções periódicas e tempos de *setup* dependentes da sequência. Duas formulações matemáticas com arcos indexados no tempo são apresentadas para o problema, as quais são comparadas entre si e com dois métodos diferentes disponíveis na literatura, sendo um exato e um heurístico. Uma das formulações propostas foi capaz de obter as soluções ótimas de todas as instâncias disponíveis na literatura, envolvendo até 50 tarefas, apresentando desempenho superior aos demais métodos existentes, inclusive em termos de tempo computacional. O aluno bolsista de iniciação científica pesquisou sobre o tema sugerido pelo orientador e ambos desenvolveram as formulações apresentadas nesse artigo. O aluno implementou as formulações propostas e conduziu os experimentos computacionais. O presente documento foi redigido majoritariamente pelo aluno sempre sob supervisão do orientador.

PALAVRAS CHAVE. Sequenciamento, Manutenções periódicas, Formulação com arcos indexados no tempo.

Área Principal: PO na Administração e Gestão da Produção, Programação Matemática, Otimização Combinatória

ABSTRACT

This paper deals with a single-machine scheduling problem with periodic maintenances and sequence-dependent setup times. Two arc-time indexed formulations are proposed for the problem and they are compared between each other and also with the other two methods available in the literature, more precisely, one exact and one heuristic. One of the proposed formulations was capable of finding the optimal solutions of all instances available on the literature involving up to 50 jobs, with a performance superior than the other existing methods, even in terms of computational time. The student performed a research on the topic suggested by the advisor and both of them developed the formulations presented in this paper. The student implemented the proposed formulations and conducted the computational experiments. The present document was mostly written by the student always under the supervision of the advisor.

KEYWORDS. Scheduling, Periodic maintenance, Arc-time indexed formulation.

Main Area: OR in Production Management, Mathematical Programming, Combinatorial Optimization

1. Introdução

As atividades de manutenção têm por objetivo eliminar os desperdícios no processo de produção, melhorar o funcionamento dos equipamentos e evitar paradas decorrentes da quebra das máquinas. Assim, cada vez mais, as empresas têm buscado incorporar em suas atividades de planejamento da produção paradas para a realização de manutenções. Dessa forma, problemas de otimização que considerem essas características têm sido cada vez mais estudados.

Neste artigo, estuda-se o problema de sequenciamento de tarefas em uma máquina com manutenções periódicas, onde o tempo *setup* depende da sequencia de tarefas adotada. A máquina é submetida a períodos de indisponibilidade, pois manutenções periódicas, com tempo de duração pré-determinado, são realizadas. Uma máquina só pode processar uma tarefa por vez e preempções não são permitidas. Esse problema é considerado $\mathcal{NP}$-Difícil (Ángel Bello *et al*., 2011).

Algumas variantes desse problema já foram estudas: Ji *et al*. (2007) e Yu *et al*. (2014) desconsideram o *setup*; Yang *et al*. (2011) permitem manutenções em períodos flexíveis; e Lee e Kim (2012) utilizam uma função objetivo que minimiza o número de tarefas atrasadas. Apenas os trabalhos de Ángel Bello *et al*. (2011) e Pacheco *et al*. (2013) possuem as mesmas características do problema estudado neste artigo.

Este trabalho tem por objetivo propor formulações matemáticas para o problema em questão e compará-las às existentes na literatura. Pretende-se também mostrar que este problema pode ser formulado de uma forma mais eficiente como um problema de máquinas paralelas idênticas.

O restante deste trabalho é organizado como se segue: uma breve descrição do problema estudado é feita na Seção 2 e as formulações propostas são apresentadas na Seção 3; os resultados dos experimentos realizados são discutidos na Seção 4; por fim, a Seção 5 conclui o trabalho e apresenta propostas para trabalhos posteriores.

2. Descrição do problema

Seja J o conjunto de tarefas a serem sequenciadas. Cada tarefa $j \in J$ possui um tempo de processamento p_i e cada par de tarefas $i, j \in J$ possui um *setup* s_{ij} associado. O período P representa o tempo entre os finais de duas manutenções. Todas as manutenções possuem um tempo de duração fixo denotado por PM. O tempo disponível para a alocação das tarefas entre cada manutenção é dado por $P - PM$, e o *makespan* ($C_{\max}$) começa a ser computado a partir do final da manutenção inicial, como se pode observar na Figura 1.

Figura 1: Representação de uma solução

3. Formulações matemáticas

Nesta seção são descritas duas formulações para o problema de sequenciamento de tarefas em uma máquina sujeita à restrições de disponibilidade. A primeira formulação, chamada de `F1`, é baseada na formulação com arcos indexados no tempo descrita em Pessoa *et al*. (2010) e a segunda, `F2`, também com arcos indexados no tempo, trata o problema descrito como um problema de máquinas paralelas.

Nas formulações seguintes, além dos dados apresentados na Seção 2, considera-se: T_{ij} como sendo o conjunto dos tempos em que a tarefa i pode preceder a tarefa j sem infringir nenhuma restrição; manutenção *dummy*, representada por 0, indica o início do sequenciamento; m é o número de manutenções a serem realizadas; J_+ é definido como o conjunto de todas as tarefas e manutenções, incluindo as *dummies*.

3.1. Primeira formulação (F1)

A formulação F1 é apresentada como se segue.

Variáveis de decisão:

$$x_{ij}^t = \begin{cases} 1, & \text{se a tarefa } i \in J \text{ termina e a tarefa } j \in J \text{ começa no tempo } t \in T_{ij}. \\ 0, & \text{caso contrário.} \end{cases}$$

$$\min \sum_{i \in J} \sum_{t \in T_{i0}} (t - s_{i0}) x_{i0}^t \tag{1}$$

$$\text{Sujeito a} \quad \sum_{\substack{j \in J \\ t = s_{0j}}} x_{0j}^t = 1 \tag{2}$$

$$\sum_{\substack{i \in J_+ \\ i \neq j}} \sum_{t \in T_{ij}} x_{ij}^t = 1 \qquad \forall j \in J_+ \tag{3}$$

$$\sum_{\substack{j \in J_+ \\ t \in T_{ji}}} x_{ji}^t - \sum_{\substack{j \in J \\ t' \in T_{ij} \\ t' = t + p_i + s_{ij}}} x_{ij}^{t'} = 0 \qquad \forall i \in J_+, t = 0, \ldots, (m+1) \times P \tag{4}$$

$$x_{ij}^t \in \{0, 1\} \qquad \forall i, j \in J_+, t \in T_{ij} \tag{5}$$

A função objetivo (1) minimiza o *makespan*. A restrição (2) força alguma tarefa a sair da tarefa *dummy* no tempo igual ao *setup* entre uma manutenção e a tarefa seguinte. As restrições (3) obrigam que sempre alguma tarefa preceda a tarefa j, assim sendo, essas restrições obrigam que todas as tarefas $j \in J$ sejam processadas exatamente uma vez. As restrições (4) representam as restrições de fluxo. Finalmente, as restrições (5) mostram o domínio das variáveis. Note que a formulação F1 permite que variáveis do tipo x_{jj}^t existam. Isso acontece devido a existência de tempos ociosos. A Figura 2 mostra um solução viável associada a um exemplo de instância de apenas quatro tarefas. Os dados dessa instância estão descritos na Tabela 1.

Tabela 1: Dados do exemplo

Setup

	0	1	2	3	4
0	0	2	1	2	1
1	1	0	2	3	1
2	1	1	0	3	3
3	2	2	1	0	2
4	1	2	4	2	0

Processamento

j	p_j
0	1
1	1
2	1
3	2
4	1

3.2. Segunda formulação (F2)

É possível observar que F1 possui um número pseudo-polinomial de variáveis e restrições proporcional ao horizonte de tempo estimado ($((m+1) \times P)$), o que torna seu uso proibitivo na prática, sobretudo para instância de dimensão elevada. A segunda formulação proposta neste trabalho (F2) também possui um número pseudo-polinomial de variáveis e restrições, porém proporcional a P, ou seja, tornando-a mais escalável que F1. Nessa segunda proposta, a ideia é tratar cada período P entre as manutenções como uma máquina independente. Dessa maneira, o número de variáveis do problema é reduzido, pois a dimensão de tempo da variável de decisão diminui, tornando-o mais fácil ser resolvido. A função objetivo foi modificada, passando a minimizar o *makespan* de apenas um dos períodos ou "máquinas". Para os demais períodos, se faz necessário obter apenas uma configuração viável, conforme demonstrado na Figura 3. O *makespan* passa a ser calculado da seguinte maneira: $C_{\max} = 2P + C_3$.

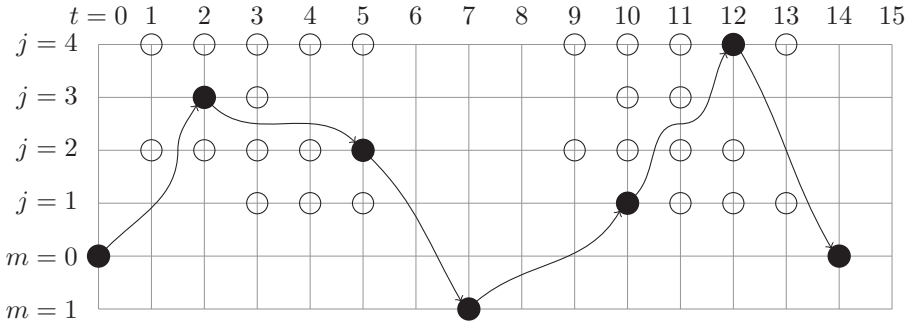

Figura 2: Solução viável usando `F1`

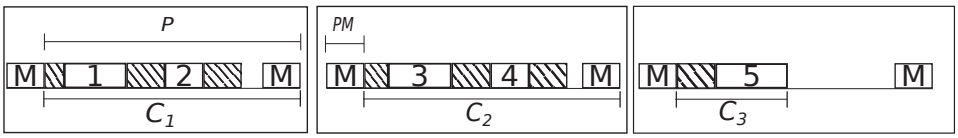

Figura 3: Representação da solução seguindo a segunda abordagem

Para tornar essa abordagem viável, foi necessário adicionar uma outra manutenção *dummy* $0'$ de forma a representar o último período, ou seja, o período cujo *makespan* deve ser minimizado. São criadas m sequências de tarefas saindo da manutenção 0 e apenas uma sequência saindo de $0'$. Para respeitar as restrições de fluxo, o mesmo número de sequencias que saem de cada manutenção *dummy* devem chegar a elas. Note que com essa formulação, não necessariamente a sequência que saiu da manutenção $0'$ chegará também em $0'$, mas isso não interfere nos resultados práticos. A nova função objetivo deve minimizar apenas o tempo em que uma das sequências chega à manutenção *dummy* $0'$, uma vez que não importa o momento em que as outras sequências terminam, pois as manutenções têm tempos de início pré-definidos. A formulação `F2` fica da seguinte forma:

$$\min \sum_{i \in J} \sum_{t \in T_{i0'}} (t - s_{i0'}) x^t_{i0'} \tag{6}$$

Sujeito a (5)

$$\sum_{\substack{j \in J \\ t = s_{0j}}} x^t_{0j} = m \tag{7}$$

$$\sum_{\substack{j \in J \\ t = s_{0'j}}} x^t_{0'j} = 1 \tag{8}$$

$$\sum_{\substack{i \in J_+ \\ i \neq j}} \sum_{t \in T_{ij}} x^t_{ij} = 1 \qquad \forall j \in J \tag{9}$$

$$\sum_{\substack{j \in J_+ \\ t \in T_{ji}}} x^t_{ji} - \sum_{\substack{j \in J \\ t' \in T_{ij} \\ t' = t + p_i + s_{ij}}} x^{t'}_{ij} = 0 \qquad \forall i \in J_+, t = 0, \ldots, P \tag{10}$$

A função objetivo (6) minimiza o *makespan* apenas do período da manutenção $0'$. A restrição (7) obriga que m tarefas saiam da manutenção *dummy* 0. A restrição (8) obriga que apenas

uma tarefa saia da manutenção *dummy* $0'$. Finalmente, as restrições (9) obrigam que apenas uma tarefa preceda as tarefas $j \neq 0$, além de garantir que todas as tarefas sejam processadas exatamente uma vez. A Figura 4 representa uma solução viável do exemplo mostrado na Tabela 1 usando a formulação `F2`.

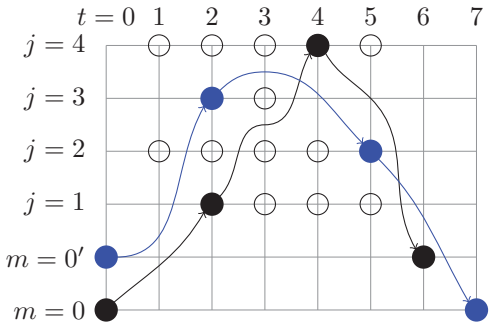

Figura 4: Solução viável usando `F2`

4. Resultados computacionais

Todas as instâncias foram resolvidas em um computador usando apenas um núcleo do processador Intel Core i7 com 16 GB de RAM e sistema operacional Ubuntu 14.04. As formulações propostas foram implementadas na linguagem de programação C++ e o *solver* de programação linear inteira adotado foi o CPLEX versão 12.6.

Nesta seção é apresentado o formato das instâncias, bem como os resultados computacionais obtidos por meio de ambas as formulações. São feitas comparações entre elas e com os métodos da literatura.

A primeira formulação apresentada neste documento só é utilizada para as instancias de menor porte, tendo em vista a grande demanda de recursos computacionais, já a segunda proposta é utilizada para todas as instâncias deste trabalho, pois os recursos computacionais necessários não são tão elevados.

4.1. Instâncias

As instâncias usadas na realização dos testes foram aquelas sugeridas por Pacheco *et al.* (2013) e são obtidas da seguinte maneira: Os valores de $p_i + s_{ij}$ foram obtidos usando distribuição uniforme entre três intervalos diferentes: [2, 8], [4, 12] e [5, 20]. Todos os elementos da matriz de custo foram transformados para que respeitassem a desigualdade triangular. Para cada instância, P contempla quatro valores diferentes: $2.25dm$, $2.5dm$, $3dm$ e $4dm$, onde $dm = \max_i\{(s_{i0}+s_{0i})/2\}$. Foram geradas instâncias com 10, 12, 15, 20, 30, 40 e 50 tarefas.

As instâncias foram agrupadas de acordo com o intervalo de $p_i + s_{ij}$ da seguinte maneira. Grupo I se refere ao intervalo [2, 8], grupo II ao intervalo [4, 12] e o grupo III ao intervalo [5, 20].

4.2. Algoritmo iterativo

Antes de resolver o problema propriamente dito, deve-se determinar o número mínimo necessário de manutenções m. Para tanto, um algoritmo iterativo é proposto. Na primeira iteração, m é estimado da seguinte maneira: $m = (|J| - 1 \times c_{min})/(P - PM)$, onde $c_{min} = \min_{i,j \in J, i \neq j}\{p_i + s_{ij}\}$. Essa estimativa é então utilizada para criar o conjunto T_{ij}. Nas iterações subsequentes, caso o resolvedor detecte inviabilidade, uma manutenção é adicionada ao parâmetro m e outra tentativa de resolução é efetuada. De acordo com os experimentos realizados, foi possível perceber que o tempo gasto para provar inviabilidade não é muito representativo e não implica em um aspecto negativo significativo para a formulação.

4.3. Comparações computacionais

Na Tabela 2, as duas formulações com número pseudo-polinomial de variáveis e restrições propostas, bem como a formulação compacta apresentada por Pacheco *et al*. (2013) são comparadas entre si.

Os dados reportados na Tabela 2 se referem a média das soluções ótimas (opt_{avg}) de cada grupo de instâncias, a média do limite inferior na raiz (LB_{root}), a média do tempo total de execução em segundos (t(s)) e a média do número de nós das árvores (#nodes). Já para a formulação comparada, apenas o tempo de execução é reportado, que foi extraído do artigo Pacheco *et al*. (2013), cujo ambiente de testes foi o seguinte: processador Intel Core 8400 de 2,99 GHz e 3,21 GB de RAM utilizando o CPLEX 11.1.

Tabela 2: Comparação entre os métodos exatos

N	Grupo	P	opt	F2			F1			Pacheco *et al*. (2013)
				LB_{root}	#nodes	t(s)	LB_{root}	#nodes	t(s)	t(s)
10	I	2,25	39,000	39,000	0,0	0,02	39,000	0,0	0,62	15,30
		2,50	35,200	35,200	0,0	0,03	35,200	0,0	0,65	2,57
		3,00	28,700	28,700	0,0	0,04	28,700	0,0	0,64	0,16
		4,00	28,000	28,000	0,0	0,19	28,000	0,0	1,16	0,15
	II	2,25	82,850	82,850	0,0	0,02	82,850	0,0	2,00	8,94
		2,50	66,900	66,900	0,0	0,03	66,900	0,0	1,08	8,03
		3,00	61,300	61,300	0,0	0,06	60,800	3,2	1,16	1,99
		4,00	55,400	55,400	0,0	0,58	55,400	0,0	2,80	0,14
	III	2,25	106,600	106,600	0,0	0,04	106,600	0,0	2,68	7,50
		2,50	91,250	91,250	0,0	0,03	91,250	0,0	2,04	1,91
		3,00	82,800	82,800	0,0	0,10	82,800	0,0	1,78	0,42
		4,00	76,000	76,000	0,0	0,53	75,353	4,6	3,46	0,12
12	I	2,25	46,125	46,125	0,0	0,03	46,125	0,0	1,27	64,22
		2,50	41,450	41,450	0,0	0,05	41,450	0,0	0,87	40,37
		3,00	35,000	35,000	0,0	0,09	35,000	0,0	0,93	2,39
		4,00	32,200	32,200	0,0	0,33	32,200	0,0	1,09	0,22
	II	2,25	88,100	88,100	0,0	0,03	88,100	0,0	3,74	35,46
		2,50	79,350	79,350	0,0	0,05	79,350	0,0	2,46	17,31
		3,00	69,000	69,000	0,0	0,12	69,000	0,0	2,13	2,72
		4,00	62,400	62,400	0,0	0,55	62,200	3,8	2,43	0,35
	III	2,25	115,300	115,300	0,0	0,06	115,300	0,0	6,07	28,92
		2,50	100,200	100,200	0,0	0,08	100,200	0,0	4,49	3,79
		3,00	100,400	100,400	0,0	0,21	97,720	37,6	8,80	6,73
		4,00	84,400	84,400	0,0	1,25	83,400	7,2	4,66	0,25
15	I	2,25	57,200	57,200	0,0	0,05	56,750	8,4	2,72	26,53
		2,50	47,800	47,800	0,0	0,08	47,800	0,0	2,08	6,29
		3,00	41,800	41,800	0,0	0,14	33,729	4,8	2,64	2,59
		4,00	37,400	37,326	1,2	0,37	37,350	2,8	1,61	0,70
	II	2,25	98,700	98,700	0,0	0,07	98,700	0,0	7,34	124,11
		2,50	91,100	91,100	0,0	0,12	91,100	0,0	7,80	120,44
		3,00	81,600	81,600	0,0	0,23	80,295	22,2	11,13	9,13
		4,00	75,000	75,000	0,0	0,65	73,745	27,8	8,53	1,81
	III	2,25	138,250	138,250	0,0	0,08	138,250	0,0	14,27	50,91
		2,50	124,400	124,400	0,0	0,12	124,400	0,0	11,67	11,09
		3,00	111,800	111,800	0,0	0,43	109,554	24,6	10,66	5,43
		4,00	98,000	98,000	0,0	0,81	77,278	44,0	10,80	0,78

É possível perceber que ambas as formulações propostas neste artigo apresentam melhor desempenho que os métodos de Pacheco *et al*. (2013). Já analisando a comparação entre as

formulações propostas neste estudo, a segunda formulação é mais eficiente que a primeira, pois, além de ser mais rápida, foi capax de obter, em média, melhores limites inferiores no nó raiz.

Na Tabela 3, a comparação feita é entre `F2` e o método heurístico proposto por Pacheco *et al*. (2013). Para a formulação matemática, as informações descritas são as mesmas da Tabela 2, e, além disso, os tempos parciais (*t*'(s)) para cada grupo de instância também são reportados, que são os tempos associados à última iteração do método, ou seja, aquela em que o número de manutenções foi estimado corretamente. Para o método heurístico, são apresentados os tempos reportados por Pacheco *et al*. (2013) e a média do *makespan* (UB) para cada grupo. Também é mostrado quantas instâncias foram resolvidas na otimalidade pela heurística (#ótimos).

Tabela 3: Comparação entre a formulação proposta e o método heuriístico de Pacheco *et al*.

N	Grupo	P	opt	F2				Pacheco *et al*. (2013)		
				LB_{root}	*t*(s)	*t*'(s)	#nodes	#ótimos	UB	*t*(s)
20	I	2,25	63,900	63,900	0,10	0,04	0,0	5	63,900	3,60
		2,50	58,100	58,100	0,18	0,10	0,0	5	58,100	3,35
		3,00	51,400	51,400	0,33	0,25	0,0	3	52,400	3,79
		4,00	47,000	46,800	0,63	0,56	3,4	3	49,000	6,79
	II	2,25	147,950	147,950	0,14	0,03	0,0	4	148,550	3,25
		2,50	123,200	122,866	0,35	0,21	2,2	5	123,200	5,22
		3,00	106,600	106,400	0,74	0,52	3,0	2	108,200	4,11
		4,00	96,400	95,587	2,02	1,80	14,8	2	97,200	5,76
	III	2,25	188,250	188,250	0,17	0,04	0,0	3	189,850	2,84
		2,50	165,900	165,900	0,48	0,26	0,0	5	165,900	3,71
		3,00	146,400	146,400	1,06	0,76	0,0	3	150,000	4,54
		4,00	134,200	132,640	3,77	2,86	47,0	1	136,800	5,22
30	I	2,25	87,275	87,100	0,53	0,35	2,0	2	88,726	15,18
		2,50	80,500	79,768	0,88	0,71	14,8	1	82,600	28,36
		3,00	71,400	71,400	0,99	0,84	11,2	0	75,800	25,78
		4,00	065,600	65,400	1,32	1,29	4,2	2	67,600	41,70
	II	2,25	190,800	190,800	0,54	0,25	0,0	2	191,800	16,15
		2,50	173,550	173,167	0,74	0,42	1,8	2	174,150	18,65
		3,00	154,300	154,153	1,96	1,49	57,2	1	158,300	23,58
		4,00	138,400	138,200	2,83	2,78	8,4	2	140,000	28,78
	III	2,25	260,100	259,986	0,51	0,17	1,4	2	268,750	20,36
		2,50	235,550	234,633	1,19	0,67	11,4	1	238,350	22,29
		3,00	206,000	202,869	3,97	3,15	86,4	1	212,200	31,89
		4,00	187,400	184,592	22,70	21,26	469,4	0	195,000	34,01
40	I	2,25	119,725	119,725	0,52	0,18	0,0	3	120,126	43,98
		2,50	112,500	112,500	0,57	0,27	0,0	0	116,150	51,30
		3,00	99,800	99,800	1,61	0,80	1,0	3	100,800	68,07
		4,00	89,200	89,200	2,13	1,90	0,0	3	91,000	77,61
	II	2,25	243,000	243,000	0,63	0,17	0,0	0	250,702	47,48
		2,50	227,200	227,200	0,78	0,21	4,0	1	233,250	51,06
		3,00	208,200	207,800	3,28	2,00	13,8	3	209,800	55,28
		4,00	183,400	183,045	15,77	14,97	182,6	0	193,600	80,32
	III	2,25	328,200	327,564	1,21	0,40	3,4	1	331,000	39,48
		2,50	298,050	295,414	7,36	6,22	250,6	0	303,050	66,99
		3,00	262,100	259,756	11,05	8,93	791,8	0	276,700	63,16
		4,00	238,000	235,499	89,58	86,80	745,6	0	254,400	71,43
50	I	2,25	141,975	141,975	0,76	0,26	0,0	1	146,302	101,71
		2,50	139,850	139,850	1,09	0,38	0,0	3	141,400	79,45
		3,00	123,800	123,800	2,51	1,58	0,0	3	124,400	141,08
		4,00	111,600	111,600	5,64	5,30	0,0	3	113,000	129,50
	II	2,25	170,575	170,575	5,52	0,62	0,0	4	170,576	117,25
		2,50	142,700	142,700	7,36	0,91	0,0	4	142,900	208,80
		3,00	117,400	117,052	11,24	2,74	13,6	0	123,800	233,46
		4,00	102,400	101,598	29,30	17,41	8,8	0	108,800	286,48

III	2,25	397,375	396,401	2,02	0,48	11,0	0	400,578	72,53
	2,50	359,500	358,769	3,60	1,88	48,0	0	373,800	89,76
	3,00	322,200	320,932	40,30	23,93	811,8	0	342,900	127,48
	4,00	293,200	292,717	96,92	93,66	717,6	0	307,600	141,77

Como é possível observar na Tabela 3, o método proposto neste trabalho é mais eficiente que o método heurístico proposto por Pacheco *et al*. (2013), pois foi capaz de determinar as soluções ótimas de todas as instâncias e, na maioria dos casos, em um tempo computacional inferior. Já a heurística não foi capaz de achar as soluções ótimas de todas as instâncias.

Note que o tamanho médio das árvores de enumeração é pequeno, pois na maioria das instâncias a formulação apresentada resolve o problema na raiz. Além disso, a diferença entre os tempos parciais e totais são geralmente pequenas, o que indica que o tempo gasto até achar o número mínimo de manutenções não é muito grande e torna viável a abordagem iterativa.

5. Conclusões

Como é possível observar pelas comparações feitas na seção anterior, é vantajoso modelar problemas dessa natureza como problemas de máquinas paralelas, pois o número de variáveis e restrições, ainda que pseudo-polinomiais, diminuem consideravelmente. Além disso, é notória a superioridade do modelo proposto em relação àqueles disponíveis na literatura, bem como para as instâncias do porte analisado. Também se percebe que não é vantajoso utilizar o método heurístico proposto por Pacheco *et al*. (2013), tendo em vista que é possível resolver o problema de maneira exata e de forma mais rápida, mesmo sendo um processo iterativo.

Para trabalhos futuros pode-se propor um método de estimação mais eficaz, para que menos tempo seja desperdiçado até que o algoritmo chegue no número mínimo de manutenções. É possível também desenvolver métodos exatos mais eficientes, tendo como base `F2`, para a resolução de instâncias de porte mais elevado, seguindo as ideias propostas por Pessoa *et al*. (2010).

REFERÊNCIAS

Ji, M., He, Y. e Cheng, T. (2007), Single-machine scheduling with periodic maintenance to minimize makespan. *Computers & Operations Research*, v. 34, n. 6, p. 1764 – 1770.

Lee, J.-Y. e Kim, Y.-D. (2012), Minimizing the number of tardy jobs in a single-machine scheduling problem with periodic maintenance. *Computers & Operations Research*, v. 39, n. 9, p. 2196 – 2205.

Pacheco, J., Ángel-Bello, F. e Ávarez, A. (2013), A multi-start tabu search method for a single-machine scheduling problem with periodic maintenance and sequence-dependent set-up times. *Journal of Scheduling*, v. 16, n. 6, p. 661–673.

Pessoa, A., Uchoa, E., de Aragão, M. P. e Rodrigues, R. (2010), Exact algorithm over an arc-time-indexed formulation for parallel machine scheduling problems. *Mathematical Programming Computation*, v. 2, n. 3-4, p. 259–290.

Yang, S.-l., Ma, Y., Xu, D.-l. e Yang, J.-b. (2011), Minimizing total completion time on a single machine with a flexible maintenance activity. *Computers & Operations Research*, v. 38, n. 4, p. 755 – 770.

Yu, X., Zhang, Y. e Steiner, G. (2014), Single-machine scheduling with periodic maintenance to minimize makespan revisited. *Journal of Scheduling*, v. 17, n. 3, p. 263–270.

Ángel Bello, F., Álvarez, A., Pacheco, J. e Martínez, I. (2011), A single machine scheduling problem with availability constraints and sequence-dependent setup costs. *Applied Mathematical Modelling*, v. 35, n. 4, p. 2041 – 2050.