5
Experimentos computacionais

A partir do *framework* desenvolvido, são propostas três aplicações para exemplificar a utilização de construção de vocabulário. Neste estudo de caso, são geradas três heurísticas para a resolução do problema de seqüenciamento de carros (PSC). O PSC, considerado nesta dissertação, foi proposto pela indústria automobilística Renault para o desafio Challenge ROADEF'2005, promovido pela Sociedade Francesa de Pesquisa Operacional. No concurso foram disponibilizadas 19 instâncias para o tipo de problema estudado, agrupadas em dois conjuntos de teste. Essas instâncias são descritas a seguir e podem ser obtidas no endereço eletrônico da competição [24].

A Tabela 5.1 apresenta as principais informações a respeito das instâncias utilizadas. Para cada instância, apresenta-se de qual conjunto de teste T ela faz parte, o seu nome ou identificação id, o número de carros a serem escalonados n, o número de restrições de capacidade de alta prioridade m_h e de baixa prioridade m_l, as taxas de utilização média t_h^{med} e máxima t_h^{max} das restrições de capacidade de alta prioridade e as taxas de utilização média t_l^{med} e máxima t_l^{max} das restrições de capacidade de baixa prioridade. Além disso, são apresentados o número de cores diferentes na instância, denotado por c, e o tamanho máximo permitido de grupos de carros de mesma cor g.

As heurísticas implementadas são obtidas com variações na utilização dos componentes do *framework*, explicadas nas seções a seguir. O repositório de soluções, necessário para a construção de vocabulário, é gerado através das soluções encontradas durante a busca pelo método proposto por Ribeiro et al. [26, 27, 29], chamada de heurística HPSC. Essa heurística foi desenvolvida para o Challenge ROADEF'2005 e projetada, conforme estabelecido no desafio, para que seu tempo de execução máximo seja de 10 minutos.

A representação das soluções do PSC utilizada é um vetor com a seqüência numerada dos carros que serão produzidos. Por exemplo, o vetor $[1, 3, 5, 4, 6, 2]$ representa uma solução de seis carros onde o primeiro veículo a entrar na linha de produção é o de identificador 1, seguido pelo 3, continuando assim até o último carro, o de identificador 2. As palavras são vetores incompletos de uma solução, ou seja, com lacunas na seqüência de carros.

T	id	n	m_h	m_l	t_h^{max}	t_h^{med}	t_l^{max}	t_l^{med}	c	g
A	024_38_3	1274	5	8	0.96	0.85	0.95	0.54	13	10
	024_38_5	1329	5	8	0.96	0.85	0.96	0.39	13	10
	025_38_1	1232	4	18	0.76	0.65	1.12	0.70	24	10
	048_39_1	618	5	12	1.00	0.89	0.96	0.67	12	10
B	022_S22_J1	540	2	7	0.85	0.45	1.00	0.23	13	500
	023_S23_J3	1130	9	8	1.04	0.57	0.55	0.26	13	25
	024_V2_S22_J1	1319	6	7	1.36	1.12	1.21	0.94	13	10
	025_S22_J3	1257	4	12	0.81	0.67	4.17	0.93	19	10
	028_ch1_S22_J2	385	9	11	1.14	0.60	0.95	0.68	20	15
	028_ch2_S23_J3	92	1	8	0.72	0.72	2.00	0.58	3	20
	029_S21_J6	773	4	3	1.05	0.58	1.09	0.42	12	15
	035_ch1_S22_J3	133	2	2	1.28	1.25	1.33	1.21	7	70
	035_ch2_S22_J3	239	3	2	2.10	1.45	1.39	1.72	7	150
	039_ch1_S22_J4	1263	2	9	0.58	0.29	0.85	0.77	15	25
	039_ch3_S22_J4	1119	2	9	0.68	0.57	0.88	0.68	17	20
	048_ch1_S22_J3	902	6	19	0.90	0.56	0.75	0.39	14	10
	048_ch2_S22_J3	595	7	16	0.89	0.54	0.58	0.30	12	10
	064_ch1_S22_J3	854	11	3	0.90	0.42	0.96	0.90	14	15
	064_ch2_S22_J4	451	4	1	0.85	0.34	1.16	1.16	10	15

Tabela 5.1: Dados das instâncias dos conjuntos de teste A e B

Nas palavras, os elementos '$*$' representam ausência de carro naquela posição. Por exemplo, o vetor $[1, *, 5, 4, *, *]$ representa uma palavra de tamanho 3.

Na formação de frases pelo método `EIntOpt`, descrito na Seção 3.5, pode-se obter frases incompletas, ou seja, um vetor com elementos '$*$'. Nas frases incompletas aplica-se a heurística construtiva, apresentada na Seção 4.2.1, para inserir os carros ausentes nas soluções parciais.

Todos os algoritmos foram implementados na linguagem de programação C++, utilizando-se a versão 3.2.2 do compilador gcc. Os experimentos foram executados em um computador equipado com processador Pentium IV 1.7GH, memória RAM de 249 MB e sistema operacional Red Hat Linux versão 9 kernel 2.4.20-31.9. Os algoritmos foram executados compartilhando-se o processamento da máquina com aplicações do sistema ou de outros usuários.

Com estes experimentos, pretende-se exemplificar a utilização do *framework* e avaliar se acrescentar heurísticas de construção de vocabulário com os operadores clássicos à estratégia definida por HPSC produz um método melhor para a resolução do problema de seqüenciamento de carros. As definições dos pontos flexíveis do *framework* são apresentadas para cada heurística desenvolvida. No Apêndice A é apresentado o código utilizado na implementação das heurísticas e as modificações realizadas em HPSC.

5.1 Heurística HCV1

A heurística HCV1 é uma construção de vocabulário que busca palavras mais consistentes. Essa heurística é aplicada como uma técnica de pós-otimização e possui como entrada de dados um repositório de boas soluções. A execução consiste em extrair fragmentos de boas soluções do repositório e combiná-los em fragmentos maiores até obter soluções completas ou não ser mais possível combiná-los. A execução termina quando todos os fragmentos identificados forem avaliados em combinações, conforme a estratégia escolhida. O método para encontrar palavras é o algoritmo `IntOpt1`, descrito na Seção 3.5, que está disponível na classe `DecomposeVetInt1` do *framework*. Para combinar as palavras é utilizado o algoritmo `EIntOpt`, descrito na mesma seção do algoritmo anterior, implementado na classe `GrowVetInt`.

Nos testes realizados foram consideradas duas execuções da heurística HPSC como o método gerador do repositório de soluções, uma com 5 minutos e a outra com 10 minutos. As soluções candidatas ao repositório são as melhores encontradas durante a busca. Ou seja, sempre que for encontrada uma solução melhor ou, pelo menos, de mesmo custo que a melhor solução conhecida, ela é enviada ao repositório. Uma solução é adicionada no repositório se satisfizer os critérios de aceitação. No caso desta heurística, é avaliada a similaridade da nova solução em relação às demais presentes no repositório. A similaridade tolerada para as soluções no repositório de soluções (*pool*) é de pelo menos 10 posições diferentes, valor que foi arbitrado por experimentação para evitar que o repositório de soluções ficasse cheio de soluções similares. A fim de analisar os parâmetros de entrada do método, foram testados três tamanhos mínimos para as palavras (w_{min}).

Foram realizados testes com as instâncias de teste A e B. A Tabela 5.2 mostra os resultados dos testes realizados com o conjunto A e a Tabela 5.3 mostra os resultados do conjunto B. Na coluna $T_1(s)$ está o tempo de execução em segundos dado para a heurística HPSC, em $|pool|$ está o número de soluções que estão no repositório utilizado na construção de vocabulário e em $T_{w_{min}}$ estão as taxas em porcentagens do tamanho de uma solução utilizadas para determinar os tamanhos mínimos para as palavras (w_{min}). Tal que, $w_{min} = n \times T_{w_{min}}$, onde n é o número de carros a serem escalonados apresentado na Tabela 5.1. Nas colunas $|p|$ e $|f|$ estão as quantidades de palavras e frases, respectivamente, encontradas pelo método de construção de vocabulário e em $T_2(s)$ o tempo em segundos gasto na execução do método. As colunas HPRC, LPRC, PCC possuem o número de violações das restrições de alta prioridade e de baixa prioridade e o número de troca de cores, respectivamente, das

melhores soluções obtidas pelos métodos. Na linha onde há valor para $T_1(s)$ os valores de HPRC, LPRC, PCC são relativos à melhor solução obtida pela heurística HPSC, enquanto as demais são relativas às melhores soluções obtidas pela construção de vocabulário.

Os resultados mostram o impacto do parâmetro $T_{w_{min}}$ sobre a construção de vocabulário. Quanto menos restritivo, aceitando palavras menores, o método realiza interseções com mais soluções para a formação das palavras. Quanto mais restritivo, menos soluções são utilizadas para se obter palavras de maior comprimento. Nos testes da instância 025_38_1 com $T_{w_{min}} = 90\%$, a restrição é tão forte que até inviabiliza a formação de palavras. Um bom valor para $T_{w_{min}}$ é aquele que busca a formação de palavras com equilíbrio entre mais soluções na composição, tornando-as mais consistentes, e palavras mais extensas, mais próximas de uma solução completa. O valor $T_{w_{min}} = 50\%$ obteve os melhores resultados na média considerando todas as instâncias. Por isso, só são exibidos os resultados dos testes com as instâncias do tipo B para as execuções com $T_{w_{min}} = 50\%$.

Pelo fato de não aceitar inconsistências (carros diferentes para uma mesma posição), em nenhum teste foi possível formar uma frase com mais de uma palavra. A geração de palavras mais extensas se beneficia disso, e possibilitou até o caso com a instância 024_38_3, $T_1(s) = 300$ e $T_{w_{min}} = 90\%$, em que a construção de vocabulário produziu uma solução melhor que a melhor encontrada até então na busca. Quanto mais próxima uma palavra está de uma solução completa há menos carros para serem inseridos. Por isso, as soluções geradas pelo método com $T_{w_{min}} = 20\%$ são piores que as demais, já que é necessário inserir muitos carros considerando apenas os critérios gulosos. Os testes com $T_{w_{min}} = 50\%$ na instância 024_38_5 mostram que determinar palavras mais consistentes pode ser uma boa estratégia, visto que possibilitou obter resultados melhores que os obtidos com palavras mais extensas.

A variação do tempo de execução da heurística HPSC mostra que o método HCV1 é sensível à qualidade e ao tamanho do repositório de soluções. Embora, com a instância 024_38_3 foi observado o contrário, na qual um repositório menor produziu soluções melhores. Nas demais instâncias, foi observado o que era esperado.

O tempo de execução de HCV1 está relacionado com o tamanho e a quantidade de palavras. Quanto menor o tamanho das palavras e maior o número de palavras encontradas, verifica-se que é maior o tempo gasto pelo método. Nas instâncias 024_38_3 e 024_38_5 com $T_{w_{min}} = 50\%$ o tempo de execução de HCV1 é aproximadamente 1/3 do tempo da heurística HPSC e menor nas demais instâncias. Essa relação sinaliza ser possível utilizar o

Instância	$T_1(s)$	$\|pool\|$	$T_{w_{min}}$	$\|p\|$	$\|f\|$	$T_2(s)$	HPRC	LPRC	PCC
024_38_3	300						4	1248	1115
		141	20%	28	28	124	6	1237	1094
			50%	36	36	107	5	1161	1081
			90%	30	30	41	4	1184	1091
	600						4	15	1080
		179	20%	39	39	188	7	1079	1103
			50%	45	45	140	6	1175	1094
			90%	36	36	55	6	114	1095
024_38_5	300						4	650	1159
		144	20%	30	30	169	6	644	1161
			50%	32	32	105	5	683	1151
			90%	33	33	41	6	631	1162
	600						4	80	538
		239	20%	49	49	267	6	619	1164
			50%	59	59	197	4	688	1160
			90%	55	55	81	5	634	1160
025_38_1	300						0	240	867
		20	20%	2	2	14	0	673	766
			50%	7	7	35	0	464	803
			90%	0	0	< 0	-	-	-
	600						0	215	795
		33	20%	5	5	35	0	647	761
			50%	11	11	53	0	409	843
			90%	0	0	1	-	-	-
048_39_1	300						0	79	522
		212	20%	33	33	45	3	778	506
			50%	40	40	44	6	771	505
			90%	50	50	27	2	825	519
	600						0	77	322
		220	20%	28	28	44	4	776	509
			50%	42	42	47	4	893	516
			90%	51	51	29	2	819	521

Tabela 5.2: Instâncias do teste A resolvidas pela heurística HCV1

Instância	$T_1(s)$	$\|pool\|$	$T_{w_{min}}$	$\|p\|$	$\|f\|$	$T_2(s)$	HPRC	LPRC	PCC
022_S22_J1	600						0	3	144
		35	50%	7	7	4	0	16	211
023_S23_J3	600						48	0	431
		563	50%	51	51	108	48	92	947
024_V2_S22_J1	600						1082	1327	1104
		153	50%	6	6	49	1195	2038	1035
025_S22_J3	600						0	3912	867
		24	50%	5	5	25	0	4011	868
028_ch1_S22_J2	600						54	7	110
		617	50%	1	1	90	54	165	154
028_ch2_S23_J3	600						0	70	6
		6	50%	1	1	1	0	72	10
029_S21_J6	600						35	2150	183
		620	50%	109	109	111	35	2150	281
035_ch1_S22_J3	600						67	52	49
		603	50%	161	161	25	67	61	81
035_ch2_S22_J3	600						386	342	204
		600	50%	0	0	65	-	-	-
039_ch1_S22_J4	600						0	29	262
		19	50%	4	4	9	0	42	312
039_ch3_S22_J4	600						0	0	231
		14	50%	3	3	6	0	0	275
048_ch1_S22_J3	600						0	0	216
		32	50%	7	7	17	0	23	384
048_ch2_S22_J3	600						3	0	344
		50	50%	3	3	3	5	10	456
064_ch1_S22_J3	600						0	0	215
		48	50%	10	10	15	0	14	231
064_ch2_S22_J4	600						0	69	130
		15	50%	3	3	1	0	69	169

Tabela 5.3: Instâncias do teste B resolvidas pela heurística HCV1

método durante uma busca para diversificar as soluções. A heurística HCV3, descrita na Seção 5.3, explora essa possibilidade de composição das heurísticas.

5.2 Heurística HCV2

A heurística HCV2 é uma construção de vocabulário que busca palavras mais extensas. Para isso, ela considera um número reduzido de soluções na formação das palavras. O método para encontrar palavras é o algoritmo `IntOpt2`, descrito na Seção 3.5, que está disponível na classe `DecomposeVetInt2` do *framework*. Para combinar as palavras é utilizado o algoritmo `EIntOpt`, descrito na mesma seção do algoritmo anterior, implementado na classe `GrowVetInt`. A heurística HCV2 é aplicada como uma técnica

de pós-otimização.

Nos testes realizados são consideradas duas execuções da heurística HPSC como o método gerador do repositório de soluções, uma com 5 minutos e outra com 10 minutos. As soluções candidatas ao repositório são as melhores encontradas durante a busca. Ou seja, sempre que for encontrada uma solução melhor ou, pelo menos, de mesmo custo que a melhor solução conhecida ela é enviada ao repositório. Uma solução é adicionada no repositório se satisfizer aos critérios de aceitação. No caso desta heurística, é avaliada a similaridade da nova solução em relação às demais presentes no repositório. A similaridade tolerada para as soluções no repositório de soluções (*pool*) é de pelo menos 10 posições diferentes. A fim de analisar os parâmetros de entrada do método, foram testados três quantidades mínimas de palavras que compõe uma palavra (s_{min}).

Foram realizados testes com as instâncias de teste A e B. A Tabela 5.4 mostra os resultados dos testes realizados com o conjunto A e a Tabela 5.5 mostra os resultados do conjunto B. Na coluna $T_1(s)$ está o tempo de execução em segundos dado para a heurística HPSC, em $|pool|$ está o número de soluções que estão no repositório utilizado na construção de vocabulário e em s_{min} estão os números de soluções que participam na formação das palavras. Nas colunas $|p|$ e $|f|$ estão as quantidades de palavras e frases, respectivamente, encontradas pelo método de construção de vocabulário e em $T_2(s)$ o tempo em segundos gasto na execução do método. As colunas HPRC, LPRC, PCC possuem o número de violações das restrições de alta prioridade e de baixa prioridade e o número de troca de cores, respectivamente, das melhores soluções obtidas pelos métodos. Na linha onde há valor para $T_1(s)$ os valores de HPRC, LPRC, PCC são relativos à melhor solução obtida pela heurística HPSC, as demais são relativas às melhores soluções obtidas pela construção de vocabulário.

Pelo fato da heurística HCV2 buscar por palavras maiores pela combinação de um número reduzido de soluções, o maior número de palavras encontradas eleva o tempo de execução do método. O tempo de execução de HCV2 chega a ser superior ao da heurística HPSC em casos de testes com as instâncias 024_38_3 e 024_38_5. O parâmetro s_{min} varia de 2 a 4, implicando diretamente no número de palavras geradas ($|p|$) e por conseqüência no tempo de execução ($T_2(s)$) do método .

Em alguns casos, como os testes com a instância 024_29_3 e $T_1(s) = 600$, foi possível combinar duas palavras. Isso fica evidenciado pelo diferente número de palavras e frases encontradas. No teste com a instância 024_29_5, $T_1(s) = 300$ e $s_{min} = 3$, três palavras foram combinadas para formar uma frase, ou foram geradas 2 frases com a combinação de 2 palavras cada. A diversidade

Instância	$T_1(s)$	$\|pool\|$	s_{min}	$\|p\|$	$\|f\|$	$T_2(s)$	HPRC	LPRC	PCC
024_38_3	300						4	1248	1115
		141	2	70	70	398	9	1195	1112
			3	47	46	281	65	426	999
			4	36	35	216	75	470	961
	600						4	15	1080
		180	2	90	89	531	46	785	1050
			3	60	59	361	73	440	990
			4	45	44	266	75	455	983
024_38_5	300						4	650	1159
		144	2	72	72	463	70	385	1102
			3	48	46	302	90	265	975
			4	36	35	231	98	237	966
	600						4	78	470
		243	2	122	122	792	9	657	1188
			3	82	81	534	91	278	1009
			4	61	60	393	94	273	960
025_38_1	300						0	240	867
		20	2	10	10	66	0	485	779
			3	7	7	51	0	675	779
			4	6	5	35	0	524	777
	600						0	216	841
		32	2	17	17	114	0	397	828
			3	11	11	79	0	568	769
			4	9	8	59	0	657	778
048_39_1	300						0	79	522
		212	2	106	106	180	7	781	495
			3	71	70	122	20	516	476
			4	54	53	95	25	495	497
	600						0	77	320
		221	2	111	111	188	9	884	519
			3	74	74	128	20	451	496
			4	56	55	96	21	444	507

Tabela 5.4: Instâncias do teste A resolvidas pela heurística HCV2

das soluções inviabiliza a composição de mais palavras em frases, por gerarem inconsistências.

O tempo de execução é muito variável, e no caso de instâncias grandes gasta-se muito mais tempo do que o disponível para a busca. O exemplo com a instância 023_S23_J3 ilustra bem o tempo necessário para executar completamente HCV2 em instâncias grandes, precisando de $1952s$ para concluir a execução. Na instância 028_ch2_S23_J3 foi necessário apenas $1s$ para concluir a execução da heurística.

Os testes com $s_{min} = 2$ geraram palavras mais extensas e, como já observado nos testes de HCV1, essas palavras produziram as melhores soluções. Entretanto, a qualidade das soluções é pior do que as geradas pela heurística HCV1. Ao analisar os experimentos chegamos à conclusão que combinar soluções aleatoriamente para gerar palavras extensas não é uma boa estratégia para este problema.

5.3 Heurística HCV3

A heurística HCV3 é uma construção de vocabulário com o mesmo propósito da HCV1, mas utilizada durante a busca e não como pós-otimização. O método para encontrar palavras é o algoritmo `IntOpt1`, descrito na Seção 3.5, que está disponível na classe `DecomposeVetInt1` do *framework*. Para combinar as palavras é utilizado o algoritmo `EIntOpt`, descrito na mesma seção do algoritmo anterior, implementado na classe `GrowVetInt`.

Nos testes realizados, a heurística HPSC foi executada como método gerador do repositório de soluções durante 5 minutos. Em seguida, executou-se a heurística de construção de vocabulário para gerar uma nova solução, em que foi considerada a melhor solução encontrada. A solução encontrada anteriormente foi melhorada pela heurística HPSC por mais 5 minutos, descontando-se o tempo gasto na execução da construção de vocabulário. No total, o método tem 10 minutos para execução.

As soluções candidatas ao repositório são as melhores encontradas durante a busca. Ou seja, sempre que for encontrada uma solução melhor ou, pelo menos, de mesmo custo que a melhor solução conhecida, ela é enviada ao repositório. Uma solução é adicionada no repositório se satisfizer aos critérios de aceitação. No caso desta heurística, é avaliada a similaridade da nova solução em relação às demais presentes no repositório. A similaridade tolerada para as soluções no repositório de soluções (*pool*) é de pelo menos 10 posições diferentes.

Foi utilizado $T_{w_{min}} = 75\%$, que representa a fração do tamanho de

Instância	$T_1(s)$	$\|pool\|$	s_{min}	$\|p\|$	$\|f\|$	$T_2(s)$	HPRC	LPRC	PCC
022_S22_J1	600						0	3	144
		34	2	17	17	11	0	6	309
023_S23_J3	600						48	0	431
		246	2	123	123	1952	48	90	957
024_V2_S22_J1	600						1082	1327	1104
		152	2	76	76	544	1205	2110	1046
025_S22_J3	600						0	3912	867
		221	2	111	111	188	9	884	519
028_ch1_S22_J2	600						54	7	110
		614	2	307	307	233	54	309	152
028_ch2_S23_J3	600						0	70	6
		5	2	2	2	1	0	72	9
029_S21_J6	600						35	2150	183
		621	2	310	309	358	35	2150	379
035_ch1_S22_J3	600						67	52	49
		604	2	302	302	8	67	58	74
035_ch2_S22_J3	600						386	342	204
		601	2	300	299	40	388	341	213
039_ch1_S22_J4	600						0	29	262
		18	2	9	9	39	0	44	771
039_ch3_S22_J4	600						0	0	231
		13	2	6	6	23	0	0	341
048_ch1_S22_J3	600						0	0	216
		31	2	15	15	62	0	13	463
048_ch2_S22_J3	600						3	0	344
		50	2	25	25	50	3	53	423
064_ch1_S22_J3	600						0	0	215
		47	2	23	23	63	0	14	270
064_ch2_S22_J4	600						0	69	130
		14	2	7	7	2	0	69	162

Tabela 5.5: Instâncias do teste B resolvidas pela heurística HCV2

uma solução utilizada para determinar os tamanhos mínimos para as palavras (w_{min}), de modo que $w_{min} = n \times T_{w_{min}}$, onde n é o número de carros a serem escalonados apresentado na Tabela 5.1.

Foram realizados testes com as instâncias de teste A e B e os resultados estão, respectivamente, nas tabelas Tabela 5.6 e Tabela 5.7. O método foi executado cinco vezes e o resultado relativo à melhor solução encontrada para cada instância é apresentado. Na coluna $|pool|$ é informado o número de soluções que estão no repositório utilizado na construção de vocabulário, em $|p|$ e $|f|$ são informados as quantidades de palavras e frases, respectivamente, encontradas pelo método de construção de vocabulário e $T_{VB}(s)$ é o tempo em segundos gasto na execução do método. As colunas HPRC, LPRC e PCC informam o número de violações das restrições de alta prioridade e de baixa

prioridade e o número de troca de cores, respectivamente, das melhores soluções obtidas pelos métodos.

Instância	$\|pool\|$	$\|p\|$	$\|f\|$	$T_{VB}(s)$	HPRC	LPRC	PCC
024_38_3	147	38	38	80	4	16	519
024_38_5	105	21	21	39	4	63	478
025_38_1	15	0	0	0	0	199	848
048_39_1	213	44	44	35	0	72	416

Tabela 5.6: Instâncias do teste A resolvidas pela heurística HCV3

Instância	$\|pool\|$	$\|p\|$	$\|f\|$	$T_{VB}(s)$	HPRC	LPRC	PCC
022_S22_J1	31	6	6	2	0	3	117
023_S23_J3	426	48	48	78	48	0	349
024_V2_S22_J1	73	0	0	4	1081	896	540
025_S22_J3	21	5	5	15	0	3912	869
028_ch1_S22_J2	603	92	92	105	54	9	156
028_ch2_S23_J3	3	0	0	39	0	70	6
029_S21_J6	596	146	146	124	35	2150	321
035_ch1_S22_J3	602	1	1	29	67	52	50
035_ch2_S22_J3	594	0	0	187	385	341	205
039_ch1_S22_J4	9	2	2	14	0	29	362
039_ch3_S22_J4	13	2	2	1	0	0	247
048_ch1_S22_J3	36	6	6	10	0	0	219
048_ch2_S22_J3	48	3	3	3	3	0	346
064_ch1_S22_J3	38	10	10	12	0	0	220
064_ch2_S22_J4	10	2	2	1	0	69	130

Tabela 5.7: Instâncias do teste B resolvidas pela heurística HCV3

A heurística HCV3 obteve resultados superiores às heurísticas anteriores. Entretanto, não foi possível melhorar a resolução do problema de seqüenciamento de carros pela heurística HPSC. A heurística HPSC foi utilizada como um componente estanque, e não era possível otimizar seu funcionamento no contexto da aplicação proposta. Cabe ressaltar que essa heurística foi configurada para o desafio Challenge ROADEF'2005, para ser executada em 10 minutos contínuos, considerando esse tempo para avançar a execução dos algoritmos na otimização de cada objetivo.