Pesquisa Operacional para o Desenvolvimento
Vol. 11, n. 3, p. 151–158, 2019

ISSN 1984-3534

O PROBLEMA DO CAIXEIRO ALUGADOR COM RESTRIÇÕES DE ALUGUEL E DEVOLUÇÃO[1]

José Victor Dias Pereira[a], André Renato Villela da Silva[a]*

[a] Instituto de Ciência e Tecnologia
Universidade Federal Fluminense, Rio das Ostras-RJ, Brasil

Artigo premiado SBPO 2019: Trabalho de Iniciação Científica - 2o Lugar
Recebido 25/09/2019, aceito 17/10/2019

RESUMO

O Problema do Caixeiro Alugador é uma variante do Problema do Caixeiro Viajante onde o cliente deseja percorrer algumas cidades utilizando um carro alugado. Neste problema o cliente pode trocar de carro em qualquer cidade da sua rota, pois cada carro apresenta um custo operacional diferente para cada trecho entre as cidades. O objetivo do Problema do Caixeiro Alugador é encontrar um ciclo hamiltoniano que minimiza os custos de deslocamentos entre as cidades e as taxas de devoluções dos carros. Este trabalho propõe um modelo mais realista para o problema, adicionando algumas restrições para a devolução dos carros e observando seu impacto nas soluções finais.

Palavras-chave: Problema do caixeiro alugador, Algoritmo genético, Meta-heurística.

ABSTRACT

The Traveling Car Renter Problem is a variant of the Traveling Salesman Problem where the customer wants to travel through some cities using a rental car. In this problem the customer can change vehicles in any city of the route, because each vehicle presents a different operational cost for each path between cities. The objective of the Traveling Car Renter Problem is to find a Hamiltonian cycle that minimizes transportation costs among cities and return fees. This paper proposes a more realistic model for the problem, adding some restrictions for the return of the vehicles and observing their impact on the final solutions.

Keywords: Traveling car renter problem, Genetic algorithm, Metaheuristics.

* Autor para correspondência. E-mail: arvsilva@id.uff.br
DOI: 10.4322/PODes.2019.011

[1] *Todos os autores assumem a responsabilidade pelo conteúdo do artigo.*

1. Introdução

No setor de turismo, é comum que o cliente alugue um veículo para viajar entre pontos turísticos ou cidades específicas. Surge, então, o interesse e a necessidade de otimizar a rota turística para diminuir os custos do cliente durante sua viagem. Diante desta necessidade, pesquisadores desenvolveram diversos algoritmos e modelos matemáticos para lidar com o problema, que é comumente chamado de Problema do Caixeiro Alugador (CaRS), uma variação do Problema do Caixeiro Viajante.

O CaRS pode ser definido como um problema onde um cliente deseja visitar um grupo de cidades (uma única vez) e para isso dispõe de alguns tipos de carros que podem ser utilizados e devolvidos em qualquer cidade. O custo do deslocamento entre duas cidades depende do tipo do carro e a devolução do carro alugado pode gerar uma taxa de devolução (caso o carro seja devolvido em uma cidade diferente da que foi alugado). Ou seja, o cliente dispõe de diferentes carros para percorrer diferentes trechos da sua viagem. Ao final ele deseja ter o menor custo possível, que inclui o deslocamento entre as cidades e as taxas respectivas às devoluções dos carros.

Os artigos da literatura modelam o problema oferecendo a possibilidade de se alugar e devolver qualquer tipo de carro em qualquer cidade. Além disso, cada tipo de veículo só pode ser alugado uma única vez. Por não considerar essas características realistas, o objetivo deste trabalho é estudar como a limitação das opções de aluguel e devolução pode afetar o encontro de soluções de boa qualidade, bem como a liberação para que o mesmo tipo de veículo seja utilizado uma segunda vez.

O restante do artigo está organizado da seguinte forma: a Seção 2 traz uma breve revisão da literatura, a Seção 3 apresenta a metodologia proposta, as Seções 4 e 5 discutem os resultados obtidos e as conclusões do trabalho, respectivamente.

2. Revisão Bibliográfica

O primeiro trabalho sobre o CaRS foi feito por (Goldbarg et al., 2012). Quatro meta-heurísticas foram apresentadas, e após vários testes e comparações, o Algoritmo Memético MA2 superou os outros métodos. O ponto forte de MA2 foi a utilização de uma versão da heurística construtiva baseada no critério de escolha do vizinho mais próximo. Esse trabalho apresentou uma diversidade de possíveis variantes do problema, mas tratou de apenas uma delas.

Uma abordagem transgenética foi aplicada para o problema por (Goldbarg et al., 2013). Algoritmos transgenéticos são meta-heurísticas inspiradas na evolução endossimbiótica intracelular (Shin et al., 2011; Moscato e Cotta, 2010), um processo evolutivo em que uma célula hospedeira e um endossimbionte cooperam para a otimização do problema. O Algoritmo Transgenético TA foi comparado com MA2 e obteve melhores resultados em muitos casos. O algoritmo TA é bastante complexo, com alguns operadores pouco comuns na literatura sobre algoritmos evolutivos, embora seus princípios não sejam novos.

Em (Silva e Ochi, 2016), foram propostos diversos métodos para a resolução do problema, como: um Algoritmo Evolutivo (EA), um método exato por meio da execução de um formulação de um Programa Inteiro Misto, e um método híbrido juntando os dois anteriores. Os resultados obtidos pelo método híbrido foram, em algumas instâncias, até 8% melhor do que os resultados da literatura até então. No entanto, apenas instâncias não-euclidianas foram analisadas.

Em (Dias et al., 2016), foi proposta uma versão híbrida de ILS (Iterated Local Search) com métodos de busca baseados em RVND (Random Variable Neighborhood Descent). Esta versão possuía um mecanismo *multi-start* e amplo conjunto de buscas locais distintas, bastante utilizadas em problemas de roteamento de veículos. A meta-heurística apresentou tempos computacionais relativamente baixos e conseguiu diversos resultados melhores do que a literatura para instâncias euclidianas.

Em (Dias, 2017), novas formulações matemáticas para programas inteiros mistos e para

métodos *branch-and-cut* foram propostas. Diversas soluções ótimas foram encontradas devido a essas contribuições, além de serem obtidos melhores limites superiores e inferiores para as instâncias maiores e mais difíceis. Esta trabalho também apresentou uma vasta comparação entre as formulações matemáticas existentes na literatura do problema. O grande problema das formulações apresentadas neste artigo foi o consumo de memória ainda maior do que na formulação de (Silva e Ochi, 2016). Em muitas instâncias, a otimização foi encerrada antecipadamente por falta de memória disponível no sistema computacional empregado.

Por fim, em (De Moraes et al., 2018), uma outra versão híbrida de ILS, VNS (Variable Neighborhood Search) e RVND foi proposta, obtendo excelentes resultados estatisticamente comprovados para instâncias não-euclidianas. Esta versão também utilizou pouco tempo computacional para obter tais resultados.

3. Métodos Propostos

O problema original do Caixeiro Alugador possui uma característica muito pouco realista onde qualquer carro pode ser devolvido/alugado em qualquer cidade. Essa liberdade de escolha pode gerar rotas muito boas, porém impossíveis de se obter na prática, visto que nem todas as cidades possuem uma concessionária e as concessionárias não dispõem de todos os tipos de carros. Diante desta impossibilidade, foram adicionadas restrições no conjunto de carros que podem ser devolvidos/alugados em cada cidade, para melhor representar a realidade do problema. As restrições adicionadas seguiram duas estratégias básicas: (a) apenas algumas cidades possuem a opção de alugar/devolver os carros: essa estratégia visa contemplar os casos reais nos quais algumas cidades não possuem uma concessionária disponível; (b) todas as cidades possuem a opção alugar/devolver alguns carros (mas não todos): essa estratégia visa contemplar os casos reais nos quais as concessionárias não possuem todos os tipos de carros.

Para executar essas estratégias foram criadas 3 novas classes de instâncias. Na Classe 1, 20% das cidades possuem a opção de alugar/devolver todos os tipos de carros. Na Classe 2, 50% das cidades possuem a opção de alugar/devolver todos os tipos de carros. Na Classe 3, todas as cidades possuem a opção de alugar/devolver 50% dos tipos dos carros.

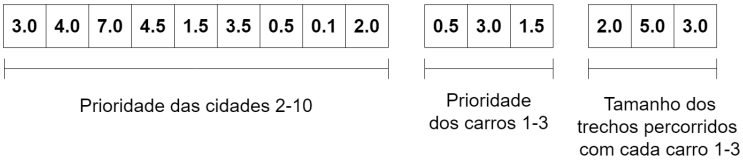
Além disso, o problema original também foi modelado de forma a impedir que um carro que tenha sido devolvido possa ser utilizado novamente em outro trecho posterior. Essa também é uma situação irreal, pois nada impede um cliente de alugar um mesmo tipo de veículo uma segunda vez, se ele considerar que isso lhe é favorável. No presente trabalho, essa limitação também foi removida. No entanto, após a realização de testes preliminares, foi possível perceber que a solução ótima da versão original do problema foi idêntica à solução ótima da versão relaxada (com possibilidade de realugar o mesmo carro) para todas as instâncias testadas. Embora, na teoria, o realuguel de um carro possa oferecer uma solução com custo final menor, na prática isso não foi percebido para nenhuma instância.

O algoritmo utilizado para a resolução do problema foi adaptado de (Silva e Ochi, 2016), onde utiliza-se uma meta-heurística do tipo Algoritmo Evolutivo que trata a versão original do CaRS. A escolha deste algoritmo se deveu ao fato dele ser bastante competitivo em relação aos demais e por ser aquele que propicia a mais fácil adaptação para tratar as alterações propostas neste trabalho. Como o realuguel do carro ocorreu na solução ótima da versão relaxada do problema, essa possibilidade não foi considerada na adaptação do algoritmo, a fim de que a comparação da versão adaptada com a versão original do algoritmo seja a mais justa possível.

O Algoritmo Evolutivo de (Silva e Ochi, 2016) é uma versão mais complexa do clássico Algoritmo Genético, onde os indivíduos são representados por vetores de valores reais que definem a ordem de visitação das cidades, a ordem de utilização dos carros e quantas cidades serão visitadas utilizando cada carro. O Caixeiro, por definição, sempre deve iniciar sua rota pela cidade 1, portanto a prioridade dessa cidade é infinita e não será exibida nos exemplos a seguir. A Figura 1 ilustra um exemplo arbitrário de um indivíduo (uma possível rota/solução do problema) em uma

instância com 10 cidades e 3 tipos de carros disponíveis.

Figura 1: Representação de um indivíduo.



Fonte: Elaborada pelos autores.

Na Figura 1, pode-se observar que a cidade de índice 4 tem a maior prioridade (7.0), portanto deve ser imediatamente visitada após a partida da cidade 1. O carro de índice 2 possui também a maior prioridade (3.0) e portanto, será o primeiro tipo de carro alugado e percorrerá os primeiros 50% de todas as cidades, de acordo com o tamanho do seu trecho, armazenado na segunda posição (5.0) do último vetor. Esses valores da última parte da representação precisam ser normalizadas ao se estabelecer a solução em si.

Após ordenar as cidades e os carros de acordo com suas respectivas prioridades, monta-se a solução final: $(1 \xrightarrow{2} 4 \xrightarrow{2} 5 \xrightarrow{2} 3 \xrightarrow{2} 7 \xrightarrow{2} 2 \xrightarrow{3} 10 \xrightarrow{3} 6 \xrightarrow{3} 8 \xrightarrow{1} 9 \xrightarrow{1} 1)$, onde $a \xrightarrow{c} b$ indica que o Caixeiro viajará da cidade *a* para a cidade *b* utilizando o carro *c*. Um pseudo-código simplificado do Algoritmo Evolutivo, proposto em (Silva e Ochi, 2016) pode ser visto no Algoritmo 1 a seguir.

Algoritmo 1: Pseudo-código do Algoritmo EA

```
1:  pop ← PopulacaoInicial(totalIndiv)
2:  geracao ← 1
3:  melhorSol ← MelhorIndivIduo(pop)
4:  while tempo total não excedido do
5:      filhos ← Recombinacao(pop)
6:      pop ← SelecaoNatural(pop+filhos)
7:      melhorSol ← MelhorIndividuo(pop)
8:      geracao ← geracao + 1
9:      if geracao mod mesmoLambda = 0 then
10:         lambda = lambda * incLambda
11:         if geracao mod intGer = 0 then
12:             Intensificacao(pop)
13:         end if
14:     end if
15: end while
16: return melhorSol
```

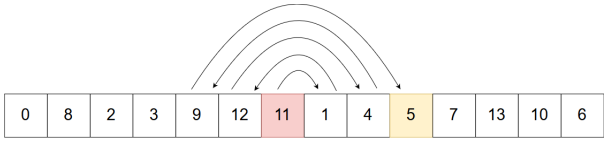
Na população inicial, os indivíduos são gerados de forma estocástica, tanto nas prioridades das cidades, quanto nos carros e até mesmo no tamanho dos trechos a serem percorridos. A recombinação gera novos indivíduos calculando-se a média dos valores dos pais que forem escolhidos. Os melhores indivíduos (entre pais e filhos) sobrevivem para a geração seguinte. Conforme o número de gerações passa (a cada *mesmoLambda* gerações), um fator λ aumenta. Esse fator faz com que os filhos possam se tornar mais ou menos diferentes das médias originalmente calculadas dos valores dos pais. A cada grupo de *indGer* gerações, a população sobrevivente sofre uma intensificação. Nessa etapa, os piores indivíduos passam por um procedimento de busca local *3-swap*, onde as prioridades das cidades são trocadas de 3 em 3. Se a troca resultar uma solução melhor, ela é mantida; caso contrário, a troca é desfeita.

A modificação proposta por este trabalho inicia-se no momento da montagem de cada solução, a partir da lista de prioridades, e tem como objetivo adaptar o algoritmo às novas instâncias. A adaptação no algoritmo consiste em verificar a disponibilidade do carro (que está sendo devolvido e do próximo a ser alugado) na cidade em que será devolvido/alugado ao final de cada trecho. Caso a cidade originalmente escolhida não disponha dos dois tipos de carros simultanea-

mente, procura-se nas cidades vizinhas até que se encontre uma cidade que contenha os dois tipos de carros. Deste modo, o trecho com o carro em questão pode ser reduzido ou estendido, de acordo com a necessidade.

A busca por uma cidade vizinha que satisfaça a condição inicia-se verificando as cidades imediatamente adjacentes à cidade originalmente selecionada. Caso não seja suficiente, aumenta-se o raio de busca enquanto necessário. A Figura 2 ilustra a lógica de busca por uma cidade que contenha os carros.

Figura 2: Dada uma rota arbitrária, pré-estabelecida, com 14 cidades, o bloco vermelho(11) representa a cidade final de um trecho e que não possui os dois carros requeridos, as setas indicam a ordem em que foram verificadas as cidades vizinhas, o bloco amarelo(5) representa uma cidade que satisfez a condição.



Fonte: Elaborada pelos autores.

Na Figura 2, a cidade 11 foi escolhida como a ultima cidade de um trecho percorrido com um determinado carro, porém, esta cidade não possui o próximo carro que o cliente deseja alugar e/ou não aceita a devolução do carro que o cliente utiliza no momento. Desta forma, verifica-se as cidades adjacentes sempre priorizando as mais próximas da cidade originalmente escolhida.

Essa lógica de busca foi definida com o objetivo de minimizar a distância entre a cidade originalmente escolhida e a cidade que satisfaça a condição, para que a interferência no comprimento dos outros trechos seja a menor possível.

A formulação matemática de (Silva e Ochi, 2016) também foi alterada para conter as novas restrições e instâncias do problema. Esta formulação apresenta bons limites primais e duais, mas não sofre do problema do consumo excessivo de memória daquela proposta em (Dias, 2017).

4. Resultados Computacionais

As novas classes de instâncias foram geradas utilizando como base as instâncias originais presentes em (Goldbarg et al., 2012). Na geração das novas instâncias, a decisão de quais carros estariam disponíveis em quais cidades foi tomada aleatoriamente de acordo com as estratégias aplicadas em cada instância. Vale ressaltar que as cidades escolhidas (e seus respectivos carros) nas instâncias de Classe 1 também estão contidas nas instâncias de Classe 2, desta forma, a Classe 2 é uma versão menos restritiva da Classe 1, o que permite uma comparação direta dos resultados.

Foram realizadas 15 execuções do Algoritmo Evolutivo adaptado para cada instância de cada classe, totalizando 45 execuções por instância. Os testes foram feitos em um notebook i5-6200U 2.3GHz, 8GB RAM e Windows 10.0 64-bits. Em relação à formulação matemática, esta foi executada pelo otimizador CPLEX, versão 12.8. Foi estabelecido um limite máximo de tempo para estas execuções: as instâncias com mais de 160 cidades receberam 4 horas (14400 segundos) e a demais instâncias receberam apenas 1 hora (3660 segundos).

As tabelas a seguir resumem os resultados encontrados nos testes. As primeiras colunas indicam o nome da instância, a quantidade de cidades (N) e a quantidade de tipos de veículos disponíveis (C). As três colunas seguintes indicam a melhor solução encontrada pela otimização da formulação, o tempo consumido e o limite dual (inferior) ao término da otimização. As três colunas seguintes indicam a média das soluções, a melhor solução gerada pela meta-heurística e o tempo máximo que o algoritmo ficou em execução. Esse tempo foi o mesmo utilizado em (Silva e Ochi, 2016). A última coluna indica a diferença percentual da melhor solução encontrada pela meta-heurística em relação à melhor ao valor da quarta coluna.

Tabela 1: Comparação das soluções obtidas pelo Algoritmo Evolutivo e dos valores da formulação matemática para as instâncias contendo 20% das cidades com disponibilidade de aluguel e devolução de veículos. *Célula vazia indica que o limite inferior é igual ao superior, ou seja, a solução ótima foi provada dentro do tempo limite.

Instância	N	C	L.Sup	Tempo(s)	L.Inf*	Média	Melhor	Tempo(s)	Melhor(Erro%)
brasilne50n	50	5	634	5,7		667,1	640	98,0	0,9%
santos50n	50	5	411	90,7		435,0	426	102,0	3,6%
berlin52na	52	3	1356	9,0		1394,3	1386	120,6	2,2%
macapa80n	80	5	604	238,4		635,9	625	416,0	3,5%
rat99nb	99	5	1355	562,1		1434,5	1395	873,3	3,0%
st70nb	70	4	926	40,1		963,5	947	276,6	2,3%
londrina100n	100	3	1152	122,1		1189,7	1179	738,0	2,3%
osasco100n	100	4	981	422,4		1021,8	1004	666,0	2,3%
rd100nb	100	4	1368	781,4		1448,8	1415	834,0	3,4%
w100nb	100	4	1624	3600,0	1621,1	1704,0	1691	777,3	4,1%
pr107n	107	5	1645	511,0		1742,9	1705	1054,3	3,6%
ch130n	130	5	1649	955,2		1754,3	1725	1200,0	4,6%
cuiaba140n	140	4	1297	1090,1		1354,3	1340	1200,0	3,3%
krob150n	150	3	2863	689,4		3000,5	2984	1200,0	4,2%
portovelho160n	160	3	1391	1865,9		1450,9	1438	1200,0	3,4%
d198n	198	4	3031	5746,5		3244,1	3206	2400,0	5,8%
teresina200n	200	5	1348	14400,0	1341,4	1428,0	1403	2400,0	4,1%
aracaju200n	200	3	1843	13792,5		1945,2	1927	2400,0	4,6%

Fonte: Elaborada pelos autores.

Para a classe de instâncias mostrada na Tabela 1, a formulação matemática obteve a solução ótima em grande parte dos casos, pois as opções de aluguel e devolução são menores. Vale observar que a diferença entre a solução ótima e a melhor solução encontrada pela meta-heurística (última coluna) é muito pequena.

Para a classe de instâncias mostradas na Tabela 2, a quantidade de cidades onde pode ser feito aluguel e devolução dos carros é maior, permitindo a existência de soluções com menor custo que nas instâncias de Classe 1. Ainda pode-se observar que a diferença entre a melhor solução da meta-heurística e a solução ótima (última coluna) continua pequena. Nessa classe de instâncias, onde há um maior número de opções em relação à classe anterior, o tempo computacional consumido pelo Algoritmo Evolutivo começa a ser menor do que o tempo da formulação matemática em uma quantidade significativa de instâncias.

Os resultados encontrados para a classe de instâncias mostradas na Tabela 3 não podem ser diretamente comparados aos resultados encontrados nas outras classes, pois, naquela classe, todas as cidades contêm a opção de alugar ou devolver alguns tipos de carros.

Pode-se observar que a formulação matemática encontrou maiores dificuldades na busca pela solução ótima, por causa do limite de tempo estabelecido. Contudo, o Algoritmo Evolutivo continuou encontrando soluções muito próximas do limite primal. Nessa classe de instâncias, o Algoritmo Evolutivo precisou de um tempo computacional bem menor (na maioria das instâncias) para obter soluções com qualidade muito próxima, o que mostra que essa abordagem heurística tem um desempenho muito competitivo. Vale ressaltar que na instância d198n a solução encontrada pela meta-heurística foi superior à solução encontrada pela formulação matemática.

Ao comparar os melhores resultados obtidos neste trabalho com os melhores resultados obtidos por (Silva e Ochi, 2016) pode-se perceber que a inclusão das restrições no aluguel e devolução dos carros impactou relativamente o custo total da rota, sendo em torno de 10,9% mais caro para instâncias pequenas como a santos50n e em torno de 2,2% mais caro para instâncias grandes como aracaju200n.

Tabela 2: Comparação das soluções obtidas pelo Algoritmo Evolutivo e dos valores da formulação matemática para as instâncias contendo 50% das cidades com disponibilidade de aluguel e devolução de veículos. *Célula vazia indica que o limite inferior é igual ao superior, ou seja, a solução ótima foi provada dentro do tempo limite. "N.D." indica que não foi encontrada nenhuma solução.

Instância	N	C	L.Sup	Tempo(s)	L.Inf*	Média	Melhor	Tempo(s)	Melhor(Erro%)
brasilne50n	50	5	615	15,7		638,1	624	98,0	1,5%
santos50n	50	5	395	84,3		414,5	404	102,0	2,3%
berlin52na	52	3	1335	13,4		1368,0	1354	120,6	1,4%
macapa80n	80	5	600	649,0		629,1	617	416,0	2,8%
rat99nb	99	5	1352	2104,4		1420,0	1403	873,3	3,8%
st70nb	70	4	910	139,4		946,5	939	276,6	3,2%
londrina100n	100	3	1146	479,2		1186,0	1179	738,0	2,9%
osasco100n	100	4	967	475,3		1003,5	990	666,0	2,4%
rd100nb	100	4	1364	3600,0	1360,0	1436,8	1420	834,0	4,1%
w100nb	100	4	1617	3600,0	1613,0	1676,9	1654	777,3	2,3%
pr107n	107	5	1640	3600,0	1655,3	1736,6	1712	1054,3	4,4%
ch130n	130	5	1640	3563,0		1718,6	1699	1200,0	3,6%
cuiaba140n	140	4	1293	2291,5		1344,9	1331	1200,0	2,9%
krob150n	150	3	2846	3,1		2991,8	2966	1200,0	4,2%
portovelho160n	160	3	1385	2254,7		1442,9	1435	1200,0	3,6%
d198n	198	4	3024	10166,7		3224,1	3197	2400,0	5,7%
teresina200n	200	5	N.D.	14400,0	1334,3	1417,6	1402	2400,0	-
aracaju200n	200	3	1837	5024,5		1937,0	1923	2400,0	4,7%

Fonte: Elaborada pelos autores.

Tabela 3: Comparação das soluções obtidas pelo Algoritmo Evolutivo e dos valores da formulação matemática para as instâncias contendo todas as cidades com apenas metade (arredondado para cima) dos veículos disponíveis. *Célula vazia indica que o limite inferior é igual ao superior, ou seja, a solução ótima foi provada dentro do tempo limite. "N.D." indica que não foi encontrada nenhuma solução.

Instância	N	C	L.Sup	Tempo(s)	L.Inf*	Média	Melhor	Tempo(s)	Melhor(Erro%)
brasilne50n	50	5	629	34,1		663,4	643	98,0	2,2%
santos50n	50	5	404	390,6		430,1	414	102,0	2,5%
berlin52na	52	3	1342	114,8		1379,9	1367	120,6	1,9%
macapa80n	80	5	613	2322,1		640,1	628	416,0	2,4%
rat99nb	99	5	1420	3600,0	1367,5	1465,1	1443	873,3	1,6%
st70nb	70	4	923	396,4		969,7	957	276,6	3,7%
londrina100n	100	3	1146	347,9		1186,3	1179	738,0	2,9%
osasco100n	100	4	979	1389,7		1017,9	1007	666,0	2,9%
rd100nb	100	4	1374	3600,0	1359,8	1448,6	1417	834,0	3,1%
w100nb	100	4	1635	3600,0	1630,7	1704,1	1680	777,3	2,8%
pr107n	107	5	1681	3600,0	1677,0	1801,4	1759	1054,3	4,6%
ch130n	130	5	1670	3600,0	1645,5	1744,3	1734	1200,0	3,8%
cuiaba140n	140	4	1296	2728,1		1367,6	1340	1200,0	3,4%
krob150n	150	3	2862	3600,0	2842,9	2992,8	2968	1200,0	3,7%
portovelho160n	160	3	1392	3600,0	1383,0	1435,1	1425	1200,0	2,4%
d198n	198	4	3292	14400,0	3019,4	3232,3	3190	2400,0	-3,1%
teresina200n	200	5	N.D.	14400,0	1339,2	1428,6	1408	2400,0	-
aracaju200n	200	3	1837	7990,8		1932,1	1902	2400,0	3,5%

Fonte: Elaborada pelos autores.

5. Conclusões

Este trabalho abordou uma variante do clássico Problema do Caixeiro Viajante, chamada de Problema do Caixeiro Alugador. Neste problema, o cliente deseja percorrer uma série de

cidades com carros alugados, podendo trocar de carro em qualquer cidade da sua trajetória. Neste trabalho, as opções de aluguel e devolução dos carros foram limitadas para melhor representar características de uma situação real. O objetivo do problema é encontrar a rota que minimiza o custo de deslocamento entre as cidades, juntamente com as taxas de devolução dos carros que forem alugados.

Foi proposto um Algoritmo Evolutivo adaptado de (Silva e Ochi, 2016) para a resolução do problema. Ao perceber que uma rota pré-estabelecida não poderia ser realizada por não haver opção de devolução ou aluguel de outro veículo nas cidades indicadas, o Algoritmo recalcula os trechos das rotas de forma a encontrar cidades próximas que permitam a devolução e o aluguel de outro veículo. Após alguns experimentos computacionais pôde-se notar que o Algoritmo Evolutivo conseguiu encontrar soluções de boa qualidade em um tempo computacional significativamente menor do que a formulação matemática testada.

Agradecimentos. Os autores agradecem à FAPERJ pela bolsa de IC (200.243/2018) concedida ao aluno José Victor Dias Pereira.

Referências

De Moraes, R. F., Villela da Silva, A. R., Ochi, L. S. e Martí, L. Implementation of a RVND, VNS, ILS heuristic for the traveling car renter problem. In: *2018 IEEE Congress on Evolutionary Computation (CEC)*, Rio de Janeiro. IEEE, 2018. p. 1–8.

Dias, S. S. *Abordagens para resolução do Problema do Caixeiro Alugador*. Dissertação (Mestrado em Computação) – Instituto de Computação, Universidade Federal Fluminense, Niterói-RJ, 2017.

Dias, S. S., Ochi, L. S., Machado, V. M. C., Simonetti, L. G. e Silva, A. R. V. Uma heurística baseada em ILS para o problema do caixeiro alugador. In: *Anais do XLVIII Simpósio Brasileiro de Pesquisa Operacional*, Vitória. SOBRAPO, 2016. p. 1625–1636.

Goldbarg, M. C., Asconavieta, P. H. e Goldbarg, E. F. G. Memetic algorithm for the traveling car renter problem: An experimental investigation. *Memetic Computing*, v. 4, p. 89–108, 2012.

Goldbarg, M. C., Goldbarg, E. F. G., Asconavieta, P. H., da S. Menezes, M. e Luna, H. P. L. A transgenetic algorithm applied to the traveling car renter problem. *Expert Systems with Applications*, v. 40, n. 16, p. 6298–6310, 2013.

Moscato, P. e Cotta, C. A modern introduction to memetic algorithms. In: Gendreau, M. e Potvin, J.-Y. (ee.), *Handbook of Metaheuristics* volume 146 of *International Series in Operations Research & Management Science*, p. 141–183. Springer US, 2010.

Shin, K. S., Park, J. O. e Kim, Y. K. Multi-objective FMS process planning with various flexibilities using a symbiotic evolutionary algorithm. *Computers & Operations Research*, v. 38, n.3, p. 702–712, 2011.

Silva, A. R. V. e Ochi, L. S. An efficient hybrid algorithm for the traveling car renter problem. *Expert Systems with Applications*, v. 64, p. 132–140, 2016.