Abordagens GRASP Aplicadas ao Problema Quota CaRS

Matheus da Silva Menezes
CCEN – Universidade Federal Rural do Semiarido - UFERSA
Rio Grande do Norte / Mossoró / Brasil
matheus@ufersa.edu.br

Marco César Goldbarg
DiMAp - Universidade Federal do Rio Grande do Norte
Rio Grande do Norte / Mossoró / Brasil
gold@dimap.ufrn.br

Elizabeth Ferreira Goldbarg
DiMAp - Universidade Federal do Rio Grande do Norte
Rio Grande do Norte / Natal / Brasil
beth@dimap.ufrn.br

Vanessa Elionara Souza Ferreira
CCEN – Universidade Federal Rural do Semiarido - UFERSA
Rio Grande do Norte / Mossoró / Brasil
vanessaelionara@ufersa.edu.br

Gabriela Correia Colaço
CCEN – Universidade Federal Rural do Semiarido - UFERSA
Rio Grande do Norte / Mossoró / Brasil
gabicolco@yahoo.com

RESUMO

Este artigo apresenta um estudo de aplicação de abordagens GRASP hibridizadas com *Variable Neighood Search* e *Path Relinking*, bem como novos operadores para a montagem da Lista de Candidatos, aplicados e adaptados ao problema do Caixeiro Alugador com Quotas. Neste problema é disponibilizado um conjunto de vértices, cada um com um bônus associado e um conjunto de veículos. O objetivo do problema é determinar um ciclo que visite alguns vértices coletando, pelo menos, um bônus pré-definido e minimizando os custos de viagem através da rota, que pode ser feita com veículos de diferentes tipos. São apresentadas quatro versões GRASP produzindo resultados em oitenta instâncias. É realizada uma comparação com resultados do modelo exato do problema, executado no *solver* GLPK, onde as abordagens GRASP encontraram o melhor resultado em 62 instâncias, mostrando o potencial de aplicação ao problema.

PALAVRAS CHAVE. Otimização Combinatória. GRASP. Quota CaRS

Tópicos: Metaheurísticas, Otimização Combinatória, Logística e Transportes

ABSTRACT

This article presents an application study of GRASP approaches hybridized with Variable Neighood Search and Path Relinking, as well as new operators for the construction of the Candidate List, applied and adapted to the Quota Car Renter Problem. In this problem a set of vertices, each with an associated bonus and a set of vehicles, are made available. The goal of problem is to determine a cycle that visits some vertices by collecting at least a predefined bonus and minimizing travel costs through the route, which can be done with vehicles of different types. Four GRASP versions are presented, producing results in eighty instances. A comparison is made with results of the exact problem model, running in GLPK solver, and the GRASP approaches present best solutions in 62 instances, showing potential of solve the problem..

KEYWORDS. Combinatorial Optimization. GRASP. Quota CaRS.

Paper topics : Metaheuristics, Combinatorial Optimization. Logistics and Trasport

1. Introdução

A aplicação de métodos heurísticos e técnicas de modelagem matemática como meio de solucionar problemas logísticos, de bens e/ou pessoas, tem se tornado um tema cada vez mais recorrente na literatura. Tendo em vista que tais soluções impactam positivamente na tomada de decisão do cotidiano de diversas empresas e pessoas, [DA SILVA e OCHI, 2016] explicam que muitos pesquisadores tem se empenhado no desenvolvimento de algoritmos e modelos matemáticos para solucionar problemas de gerenciamento de frotas, onde cada uma dessas apresenta características, comportamento e objetivos distintos. De acordo com [BODIN, 1990] existem diversos métodos na literatura que realizam de forma eficiente a definição e o aprimoramento de rotas de transporte, porém a maior transformação em relação a esses métodos se desenvolveu no ambiente computacional, quando muitos algoritmos baseados em modelagem matemática atingiram resultados com baixo percentual de erro em relação às soluções exatas.

O Problema do Caixeiro Viajante (PCV) é um problema de otimização combinatória clássico e tema de diversos estudos e consiste em definir o menor custo associado a distância percorrida entre *n* vértices, iniciando e finalizando a roteirização no mesmo ponto, com cada vértice sendo visitado uma única vez. Conforme apresentado em [GOLDBARG; LUNA, 2005], o PCV tem aplicações em problemas de coleta e distribuição de produtos, sistemas de produção, roteamentos, seqüenciamento de tarefas, dentre outros. No trabalho de [GOLDBARG, ASCONAVIETA E GOLDBARG, 2012] foi proposta uma generalização do PCV, considerando como elemento central o cliente que pretende percorrer uma rota utilizando-se de veículos alugados, devendo-se agora considerar não apenas os custos associados às distancias percorridas, como também o custo associado ao aluguel de acordo com o modelo de carro utilizado, bem como suas taxas associadas, denominado Problema do Caixeiro Alugador ou *Traveling CarRenter Problem* (CaRS).

No CaRS, existem diversas possibilidades de escolhas de veículo para o cliente, que deve buscar em cada ponto obter a opção de carro com melhor combinação possível no que se refere aos custos associados À distancia percorrida, valor de aluguel e taxa de retorno, onde este último é um custo associado a entrega do carro em um ponto diferente do de origem de aluguel de tal veículo. Admitindo-se que o CaRS apresenta uma maior quantidade de variantes a serem consideradas, e o PCV é reconhecido como um subcaso do mesmo e é NP-difícil [GUTIN; PUNNEN, 2002], o CaRS também é um problema NP-difícil.

De acordo com [GOLDBARG et al. 2011] o CaRS possui grande aplicação no segmento de transporte turístico, segmento este que vem apresentando um crescente volume de estudos. Além disso, diversos estudos, tais como [VANSTEENWEGEN, SOUFFRIAU e OUDHEUSDEN, 2011], estão direcionados para a otimização de definição de rotas turísticas, buscando obter o máximo de satisfação de um indivíduo que apresenta diferentes restrições em visitar variados pontos de turísticos de uma cidade. Este artigo trata de uma variante do problema CaRS, denominada Caixeiro Alugador com Quota ou *Quota CaRS* (QCaRS) que foi introduzido em [MENEZES et al, 2014] e reformulado em [GOLDBARG et al, 2016]. Este problema é uma generalização do Problema do Caixeiro Viajante com Quota ou *Quota TSP* (QTSP). O Problema QCaRS é uma variante do CaRS que, assim como no QTSP, uma quota é associada a cada cidade. Esta quota define um nível de satisfação em visitar a cidade associada. Esse parâmetro é relevante, pois em muitos casos o turista não pode visitar todas as atrações existentes durante uma viagem, devendo escolher as que possuam maior interesse.

No problema QCaRS uma satisfação presumida de visita é atribuída a cada cidade e uma satisfação cumulativa pré-definida mínima deve ser atendida. Além disso, um vértice é escolhido para ser a cidade onde o passeio começa e termina, denominada cidade base. O objetivo é selecionar um subconjunto de cidades (vértices) a serem visitados, visando à minimização do custo total da viagem, incluindo o aluguel dos veículos e que a satisfação mínima de visitar as cidades, representada pelo parâmetro ω seja atendida. Nesses casos, é interessante tentar maximizar a satisfação visitando os pontos mais atraentes.

Como o problema QCaRS é relativamente novo na literatura, poucas metaheurísticas foram aplicadas ao mesmo. Em [MENEZES et AL, 2014] foi apresentado um algoritmo memético e em [GOLDBARG et al, 2016] um algoritmo evolucionário híbrido para a resolução do problema. O presente artigo propõe a implementação de estratégias GRASP adaptadas ao problema, e hibridizadas com *Variable Neighborhood Search* (VNS) e *Path Relinking*, ainda não relatadas na literatura.

Os demais tópicos do artigo estão organizados da seguinte forma: Na seção 2 é definido o problema QCaRS. Na seção 3, as estratégias GRASP e suas hibridizações são relatadas em detalhes. Já na seção 4, temos os resultados dos experimentos computacionais enquanto na seção 5 temos as considerações finais sobre o estudo realizado.

2. O Problema Quota-CaRS

Seja $G = (V, A)$ um grafo completo, onde V é um conjunto com n nós (cidades) e A é um conjunto de arcos (estradas entre as cidades). Um bônus SV_i, $i = 1,...,n$, é atribuído a cada cidade $i \in V$. Neste problema, um conjunto C de veículos está disponível para aluguel. Custos operacionais específicos estão associados a cada carro, incluindo o consumo de combustível, pagamento de pedágio e custos de aluguel. Uma vez que as taxas de pedágio normalmente dependem do tipo de veículo e do comprimento do caminho percorrido, é possível assumir, sem perda de generalidade, que o custo operacional de cada carro para atravessar aresta $(i,j) \in A$ é uma função daquele carro. Além disso, o QCaRS considera que o bônus mínimo a ser recolhido é dado pelo parâmetro ω, e nesse estudo tem valor igual $0.8ST$, onde, $ST = \Sigma SV_i$, $i = 1,...,n$.

O objetivo do QCaRS é completar um caminho Hamiltoniano em G pagando o mínimo possível. Como o QCaRS é uma variante do Quota TSP quando apenas um carro é utilizado, e o Quota TSP é NP-Difícil, então o QCaRS também é.

Neste estudo, investigamos a mesma variante do problema QCaRS relatada em outros estudos: o grafo que modela o problema possui todas as ligações (**completo)**; qualquer carro pode ser alugado em qualquer cidade (**total)**; o carro alugado pode ser devolvido em qualquer cidade (**irrestrito)**; cada carro só pode ser locado uma vez (**sem repetição)**; os custos de retorno independem da topologia das estradas ou restrições de rede (**livre)**; os custos de utilizar um carro *c*, para ir de uma cidade *i* para uma cidade *j* são simétricos aos de *j* para *i* utilizando *c* (**simétrico**)

Uma definição formal detalhada e um modelo matemático para o problema foram apresentados e implementados no *solver* GLPK em [MENEZES et AL, 2014] e atualizado com novas linearizações em [GOLDBARG et al, 2016].

3. Estratégias GRASP e VNS aplicadas ao Quota CaRS

GRASP é a abreviação da nomenclatura Greedy Randomized Adaptative Search Procedure e proposto por [FEO; RESENDE, 1989] e [FEO; RESENDE, 1995]. Seu diferencial está na geração da solução inicial, cuja abordagem está referenciada nas três primeiras iniciais de sua sigla: gulosa (*Greedy*), aleatória (*Randomized*) e adaptativa (*Adaptive*). Essa estratégia tem sido amplamente utilizada em problemas relacionados ao roteamento de veículos, a exemplo de [LABADIE; MELECHOVSKý; CALVO, 2011], [SOUFFRIAU et al., 2008], [CHAOVALITWONGSE; KIM; PARDALOS, 2003], [CHAVES et al., 2007], [MARINAKIS; MIGDALAS; PARDALOS, 2005].

Essa metaheurística busca manter um equilíbrio entre ser guloso e aleatório, e envolve duas fases distintas: a Fase Construtiva, onde se busca construir boas soluções, utilizando-se para isso uma lista de candidatos, listando todos os vizinhos, a partir da qual é gerada a Lista Restrita de Candidatos - LRC , onde sua composição é formada pelos α % melhores vizinhos disponíveis, onde $0 \leq \alpha \leq 1$. Valores de α próximos a zero conduzem a soluções muito próximas aquela obtida pela escolha gulosa, enquanto valores de α próximos a 1 conduzem a soluções praticamente aleatórias. A heurística é adaptativa porque os benefícios associados com a escolha de cada elemento são atualizados em cada iteração da fase de construção, de forma a refletir as mudanças

oriundas da seleção do elemento anterior. O componente probabilístico do procedimento reside no fato de que cada elemento é selecionado de forma aleatória, a partir de um subconjunto restrito formado pelos melhores elementos que compõem a LRC.

Na segunda fase aplica-se a Busca Local. Procura-se com ela melhorar as soluções encontradas, através de um método de exploração da vizinhança. Um algoritmo de busca local define uma vizinhança para cada solução, fazendo uma transformação da solução atual através de uma técnica ou heurística, gerando um conjunto de soluções com características próximas.

O principal desafio entre os métodos heurísticos que usam a busca local é definir uma estratégia eficiente para cobrir o espaço de busca, explorando principalmente regiões promissoras [VANSTEENWEGEN et al., 2011].

Muitas vezes temos o GRASP hibridizado com outra metaheurística. A heurística *Variable Neighborhood Search* (VNS) foi proposta por [MLADENOVIC, 1995] e consiste na ideia de mudar sistematicamente a vizinhança de busca. O método VNS funciona de forma pouco convencional se comparado a outras metaheurísticas [CHAVES et al., 2007]. Ele explora vizinhanças mais distantes da solução corrente e concentra a busca em torno de uma nova solução. A solução é atualizada apenas caso um movimento de melhora seja realizado. O método pode incluir também rotinas de busca local que utilizam diferentes estruturas de vizinhança.

Outra estratégia de uso comum em conjunto com o GRASP é o *Path-Relinking*. proposto de forma pioneira por [GLOVER, 1996] e é uma estratégia de intensificação de soluções que possui o objetivo de gerar novas soluções a partir da utilização de trechos de soluções de elite que são inseridos nas soluções do sistema e assim encontrar soluções de melhor qualidade, que possuam as características relevantes das soluções de elite. Para gerar os caminhos, os movimentos são selecionados para apresentar os atributos que estão presentes na solução de elite, passando-os para a solução atual. O *Path Relinking* pode ser visto como uma estratégia que pretende incorporar atributos de soluções de elevada qualidade, ao favorecer estes atributos nas novas soluções através de movimentos selecionados. Em [RESENDE; RIBEIRO, 2003], é feita uma revisão das várias alternativas consideradas para o Path-Relinking.

De posse dos comentários acima, foram propostos no presente trabalho quatro algoritmos com abordagem GRASP, sendo o primeiro em sua formulação tradicional hibridizado com VNS, o segundo hibridizado com VNS e *Path-Relinking*. Os outros dois algoritmos utilizarão um novo operador de intensificação que será descrito em detalhes posteriormente. Os algoritmos de abordagem GRASP utilizados no presente trabalho são descritos a seguir

3.1. GRASP + VNS

Nesta seção, é apresentado o algoritmo GRASP hibridizado com VNS, proposto para a resolução do problema pCaRS. A fase inicial do algoritmo corresponde a fase construtiva do GRASP. A segunda fase engloba a aplicação de técnicas de busca local com abordagem VNS nas soluções encontradas.

O pseudo-código do algoritmo é apresentado na Figura 1, onde suas partes constitutivas são explicadas nos itens subsequentes. Os parâmetros de entrada para o algoritmo são: `nomeInstancia`, que identifica o problema que será executado, `imax`, que representa o número de iterações que o algoritmo vai executar, e α que é o tamanho da Lista Restrita de Candidatos. O conjunto de soluções encontrados é representado por $X = \{x_0, x_1, \ldots, x_n\}$ e a melhor solução encontrada por x^*. A função que avalia o custo de cada solução é representada por `f` e `numSol` representa o número de soluções que serão geradas na fase construtiva do algoritmo.

A fase construtiva está implementada no método *geraSolConstrutiva*(α); que recebe o tamanho da lista restrita de candidatos como parâmetro de entrada. A cidade inicial escolhida no algoritmo é a cidade base (nó-1). Um carro c_1 é escolhido aleatoriamente para a primeira cidade. Em seguida, a cidade *i* onde o carro c_1 será entregue é também escolhida de forma aleatória. Um caminho é criado entre o nó-1 e a cidade *i* através da metodologia construtiva do GRASP, considerando os custos de viagem com o carro c_1 para montar a LRC. A cidade *i* é a cidade inicial da próxima parte da rota. Em seguida, um novo carro c_2 e uma nova cidade *j* são

escolhidos aleatoriamente. Obviamente, a cidade *j* não pode ser uma cidade visitada anteriormente. Um caminho é construído entre as cidades *i* e *j* com as cidades não visitadas, utilizando a mesma abordagem. Caso não existam mais carros disponíveis, utiliza-se o último carro sorteado. O procedimento continua sorteando outra cidade aleatoriamente e construindo o caminho até que o nível mínimo de satisfação seja alcançado, ou seja, 80% da satisfação total disponível. Então o ciclo é fechado, considerando a ligação com a cidade de partida.

Algoritmo 1: GRASP + VNS proposto para o QCaRS

Entrada: `nomeInstancia, imax, α`
Saída: x^*
  $f(x^*) \leftarrow \infty$
  para $i=1 \ldots i_{max}$ **faça**
    $x \leftarrow$ *geraSolConstrutiva*(α);
    $x \leftarrow$ *buscaLocal*(x);
    se $f(x) < f(x^*)$ **então**
      $f(x^*) \leftarrow f(x)$
      $x^* \leftarrow x$
    fim
  fim
retorne x^*

Figura 1. Algoritmo GRASP +VNS

O VNS corresponde à fase de busca local, que no caso do problema QCaRS, pode atuar em três áreas distintas: a melhor rota,a ordem de uso dos carros e o nível de satisfação. A fase de busca local do algoritmo proposto visa lidar com estas três áreas, afim de melhorar os candidatos a solução. A busca local é implantada no procedimento *buscaLocal()* apresentado na Figura 2 e é similar ao proposto em [MENEZES et al, 2014], sendo composto por seis buscas locais aplicadas em seqüência.

1. *buscaLocal*(*x*)
2. *x*← removeCidade(*x*)
3. *x*← inverteSol (*x*)
4. *x*← insereCidade(*x*)
5. *x*← substituiCidade(*x*)
6. *x*← substituiCarro(*x*)
7. *x* ← 2Opt(*x*)
8. fim

Figura 2. Procedimento *buscaMultiOperadores*

A cada aplicação de procedimentos constantes na busca local, a solução `x` é atualizada com o melhor valor encontrado no procedimento. Dessa forma, se a solução de entrada é melhorada durante a aplicação de um procedimento da fase de busca local, então a nova solução substitui a antiga. Ao final do procedimento de busca local é retornada a melhor solução da vizinhança da solução de entrada `x`. Os procedimentos de busca possuem o seguinte funcionamento básico:

- ***removeCidade*:** Este método consiste em remover da solução as cidades com os menores índices de satisfação. As cidades vão sendo removidas enquanto a restrição referente a satisfação mínima é atendida.
- ***InverteSol*:** Este método inverte a ordem de visita das cidades na solução. Os mesmos carros estão associados com as mesmas cidades, mas os pontos de aluguel e entrega são trocados.
- ***substituiCarro***: Este procedimento concentra-se em veículos que não estão ainda na solução a examinando a inserção de um novo carro, se possível, substituindo outro na solução.

- ***substituiCidade***: Este procedimento examina a substituição de cidades na solução por cidades que não estão presentes na solução. Todas as cidades que estão fora da solução de entrada são consideradas para a inserção.
- ***insereCidade***: Este procedimento testa a inserção de uma nova cidade entre cada par de cidades da solução de entrada. Ele também foca em cidades que ainda não estão na solução de entrada. O carro atribuído a nova cidade é o mesmo da cidade imediatamente anterior. .
- ***2-opt*****:** é um algoritmo de busca local simples proposto por Croes (1958) para o PCV. Um movimento 2-opt consiste em eliminar duas arestas e religar os dois caminhos, resultando em uma maneira diferente de obter uma nova rota. Entre todos os pares de arestas cuja troca *2-opt* atua, a que resultar no menor custo de rota é escolhido. Este procedimento é então iterado até que o par de arestas mais econômico seja encontrado. A rota resultante é chamada de 2-ótima. Neste caso, a seqüência de carros associados permanece a mesma

Durante a aplicação dos procedimentos de busca local e Path Relinking, podemos ter a geração de soluções inviáveis. Tais soluções são reparadas através do mesmo procedimento adotado em [MENEZES et al, 2014].

3.2.. GRASP + VNS + *Path Relinking*

O procedimento *Path Relinking* é bastante utilizado em conjunto com a metaheurística GRASP, como estratégia de intensificação de soluções [RESENDE; RIBEIRO, 2003]. Duas estratégias básicas de utilização *do Path-Relinking* são:a aplicação a todos os pares de soluções elite, ou periodicamente, durante as iterações do GRASP, ou após todas as iterações GRASP em uma etapa de pós-otimização.

O *Path Relinking* proposto considera que uma quantidade pré-definida das melhores soluções geradas pelo algoritmo GRASP formarão o conjunto de elite. Este conjunto é atualizado e ordenado a cada iteração até atingir o número máximo de soluções pré-determinado. Se uma nova solução for melhor do que a última solução do conjunto (e conseqüentemente a pior solução deste conjunto), então a nova boa solução é adicionada ao conjunto e a última é removida, reordenando o conjunto.

Algoritmo 2: GRASP + VNS + *Path Relinking* proposto para o QCaRS

Entrada: `nomeInstancia, imax, α`
Saída: x^*
  $f(x^*) \leftarrow \infty$
  para $i=1 \ldots i_{max}$ **faça**
    $x \leftarrow$ *geraSolConstrutiva*(α);
    $x \leftarrow$ *buscaLocal*(x);
    se $f(x) < f(x^*)$ **então**
      $f(x^*) \leftarrow f(x)$
      $x^* \leftarrow x$
    fim
    $Elite \leftarrow$ *montaElite*(x);
  fim
  $x \leftarrow$ *PathRelinking*($Elite$);
retorne x^*

Figura 3. Algoritmo GRASP +VNS + Path Relinking

De posse dessas soluções, o procedimento funciona utilizando a melhor solução do conjunto como solução base. A solução de destino é escolhida como sendo a segunda melhor solução do conjunto. Em seguida busca-se inserir cada cidade, com seu respectivo carro na solução de destino, sobrepondo o par cidade-carro anteriormente existente. O algoritmo considera a inserção acumulativa das cidades constantes na solução base na solução destino, na mesma ordem em que estão presentes na solução base. Ao final desse processo, a melhor solução obtida permanece e será a nova solução base. A terceira melhor solução do conjunto é então escolhida

como solução de destino, repetindo todo o processo. O procedimento continua de forma semelhante até que não existam mais soluções destino no conjunto de elite. A Figura 4 a seguir ilustra o funcionamento da escolha das soluções base e destino e do operador de *Path Relinking* proposto.

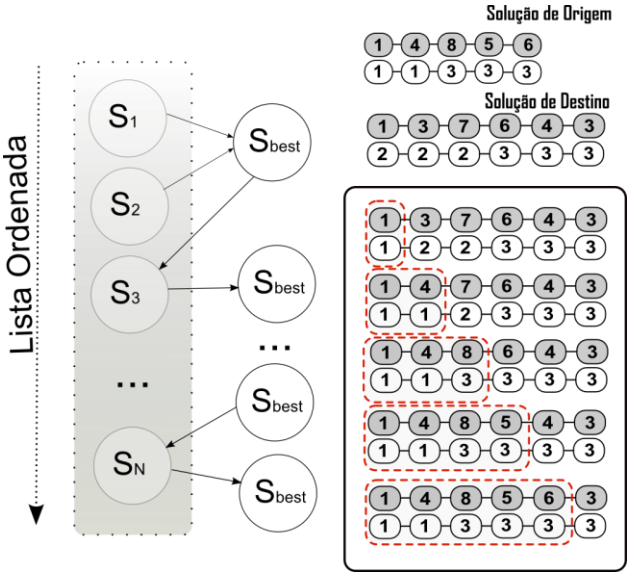

Figura 4. Exemplo de aplicação do Procedimento *Path Relinking*

3.3. GRASP 2-Potencial + VNS + *Path Relinking*

Este terceiro experimento com o GRASP considerou a utilização de uma nova abordagem na construção da lista de candidatos. Para cada candidato, verificou-se a adequação para inserção considerando dois passos adiante, considerando as três cidades com menor custo de inserção em cada passo. A intenção dessa abordagem é de tentar evitar que seja efetuada uma inserção que é boa considerando a LRC atual, mas que não será uma boa opção para o restante da solução em longo prazo.

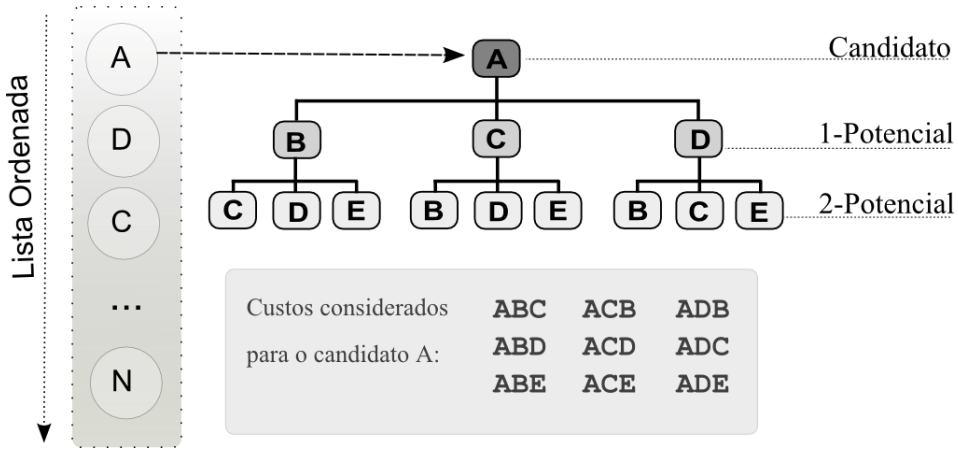

Figura 5. Exemplo de aplicação do Procedimento *2Potencial*

Para cada cidade na Lista de Candidatos, são consideradas as próximas três melhores cidades (com menor custo), e a partir de cada uma dessas cidades, as três próximas de menor custo. A montagem da lista é feita considerando o somatório cumulativo de cada caminho com todos os carros disponíveis. Cada cidade da lista de candidatos terá (*nCar* 9*) opções geradas para a lista. A LRC é montada a partir da lista de candidatos, considerando os α % melhores. A Figura 5 ilustra o funcionamento dessa abordagem a partir de uma entrada de cidade A da lista de candidatos da abordagem tradicional.

É importante ressaltar que as soluções podem ter tamanhos (quantidade de cidades) diferentes, o que pode tornar o procedimento apenas parcial, conforme ilustrado na Figura 5. Após cada inserção, pode ser necessário aplicar o procedimento de restauração de soluções, já descrito anteriormente. Foram consideradas para implementação duas variações dessa abordagem, com diferenças na forma como o sorteio na LRC é efetuado, sendo:

- **Versão 1**: O sorteio na LRC foi efetuado considerando igual probabilidade a todas os candidatos.
- **Versão 2:** O sorteio na LRC foi do tipo "roleta", onde a probabilidade foi proporcional ao valor de inserção de cada candidato.

4. Experimentos Computacionais

As metaheurísticas propostas e seus respectivos experimentos computacionais são apresentados com detalhes no presente capítulo. Com a finalidade de validar os algoritmos propostos, os mesmos foram implementados em um PC Intel Core i5, 16G de RAM, rodando Ubuntu 12.04 64bits, utilizando a linguagem C++ e executados trinta vezes para cada instância. Os testes preliminares para a definição de parâmetros do algoritmo, consideraram seis instâncias da biblioteca QCaRSLib, disponível em <http://www.dimap.ufrn.br/lae/en/projects/CaRS.php> . com um número de cidades na faixa de 9 a 25 e 2 a 4 veículos, onde foram executadas 20 vezes cada uma. Dessa forma, os parâmetros adotados no GRASP foram uma lista LCR de tamanho α=0.25 e como critério de parada, foi estabelecido um limite de 5000 iterações. Para o modelo exato foi utilizado o solver GLPK versão 4.47, limitado a 80.000 segundos de execução.

As tabelas I e II apresentam os resultados obtidos através dos experimentos realizados. Os valores com asterisco indicam que o solver GLPK não alcançou garantias de solução ótima e parou devido a limitações de tempo. Os valores sombreados referem-se à melhor solução encontrada para cada instância e os valores em negrito e sem sombreado referem-se ao melhor valor encontrado entre os algoritmos GRASP propostos.

Os algoritmos com abordagem GRASP propostos obtiveram uma boa adaptação aos problemas analisados. Os algoritmos variaram na fase de pós-otimização, onde tivemos a proposta com e sem *Path-Relinking*. Além disso, foi proposta a abordagem 2-Potencial com duas formas de escolha do candidato na LCR: através de sorteio e através de roleta. Os resultados são exibidos nas tabelas 1 e 2. O menor tempo de processamento em todas as instâncias provém da abordagem GRASP sem estratégias de pós-otimização, representado pela sigla GRASP1. A abordagem GRASPPR também obteve um tempo de processamento ligeiramente superior ao da abordagem GRASP1, indicando que o operador de *Path Relinking* é leve e demanda pouco esforço computacional. A abordagem 2-Potencial obteve um tempo de processamento pouco maior que as outras duas abordagens.

O menor valor encontrado para o problema, incluindo os resultados do modelo exato obtidos em [GOLDBARG et al, 2016] são tidos como valor de referência para o problema. Como o problema considerado é de minimização, quanto maior o resultado desta operação, pior é a solução encontrada pelo algoritmo. Em relação a essa variável, verifica-se que a versão GRASPPR foi mais eficiente, encontrando, de forma geral, os menores valores da abordagem considerada. O GRASP1 também obteve um bom desempenho, com soluções próximas ao mínimo encontrado. A abordagem 2-Potencial obteve desempenho pior do que o algoritmo com a abordagem tradicional. Os menores valores médios foram encontrados pela abordagem GRASPPR, que utiliza o operador de *Path-Relinking* como pós-otimização. As abordagens de GRASPPR e GRASP1 obtiveram soluções médias muito próximas, e com comportamento semelhante. Observe que estas duas abordagens se destacaram em relação à utilização da metaheurística GRASP com abordagem 2-Potencial. É importante ressaltar que quanto maior o número de cidades na instância, maior é a diferença entre a solução mínima encontrada e a solução média dos algoritmos, indicando uma perda de eficiência na busca dos valores mínimos em instâncias com maior número de cidades.

5. Considerações Finais

O problema do caixeiro alugador com coleta de prêmios aqui analisado possui uma maior dificuldade na fase de busca local, pois possui três regiões onde as buscas podem atuar: na rota da solução, nos pontos de troca dos veículos, e no recolhimento de bônus. Cada busca aplicada pode ser efetiva para um dos aspectos acima, porém sem garantias de melhoria dos demais, sendo um dos aspectos mais sensíveis do problema, pois eles são conflitantes entre si.

As abordagens GRASP demandaram pouco tempo de processamento entre quando comparada às abordagens utilizadas em [MENEZES et al, 2014] e [GOLDBARG et al, 2016]. O Algoritmo Grasp2PRol não obteve resultados qualitativos satisfatórios, indicando que a abordagem 2-Potencial com escolha através de roleta não foi bem adaptada a esta variante do problema. Para a abordagem GRASP-VNS, analisamos quatro variações onde apenas a primeira versão não contou com uma fase de pós-otimização. Nas demais, analisamos o operador de *Path-Relinking*. A segunda versão do algoritmo mostrou que o operador foi eficiente na melhoria da solução ótima encontrada pelo algoritmo, a um custo de tempo de processamento relativamente baixo. Foi proposta uma nova abordagem de montagem da Lista de Candidatos denominada 2-Potencial, que não teve resultados promissores. Nessa proposta, o sorteio da LCR também foi feito através de roleta, cujos resultados foram inferiores a todas as outras abordagens. O ponto positivo dessa metaheurística foi relativo ao baixo tempo de processamento, quando comparado às abordagens analisadas em outros estudos para este problema. De uma forma geral, os algoritmos analisados tiveram uma boa adaptação para as instâncias menores, tanto Euclidianas, quanto não Euclidianas. Ao aumentar o tamanho da instância, os algoritmos vão perdendo eficiência na busca pelo valor ótimo, e tendem a demandar maior tempo de processamento. A perda de eficiência é mais notória nas instâncias não Euclidianas.

Referências

BODIN, L.D. Twenty years of routing and scheduling.Operations Research, v. 38, n. 4, pp. 571-579, 1990.

CHAOVALITWONGSE, W.; KIM, D.; PARDALOS, P. Grasp with a new local search scheme for vehicle routing problems with time windows. Journal of Combinatorial Optimization, Kluwer Academic Publishers, v. 7, n. 2, p. 179_207, 2003. ISSN 1382-6905.

CHAVES, A. A. et al. Metaheurísticas híbridas para resolução do problema do caixeiro viajante com coleta de prêmios. Produção, scielo, v. 17, p. 263 - 272, 08 2007. ISSN 0103-6513

DA SILVA, A.R.V;OCHI, L.S. An efficient hybrid algorithm for the Traveling Car Renter Problem. Expert Systems With Applications,v. 64, pp. 132–140, 2016.

FEO, T. A.; RESENDE, M. G. A probabilistic heuristic for a computationally dificult set covering problem. Operations Research Letters, v. 8, n. 2, p. 67 - 71, 1989. ISSN 0167-6377.

FEO, T. A.; RESENDE, M. G. Greedy randomized adaptive search procedures. Journal of Global Optimization, v. 6, p. 109-133, 1995.

GLOVER, F. Tabu search and adaptive memory programing: Advances, applications and challenges. In: Interfaces in Computer Science and Operations Research. [S.l.]: Kluwer, 1996. p. 1-75.

GOLDBARG, M. C.; ASCONAVIETA, P. H.; GOLDBARG, E. F. G. Memetic algorithm for the traveling car renter problem: an experimental investigation. Memetic Computing, v. 4, n. 2, p. 89-108, 2012.

GOLDBARG, M. C.; ASCONAVIETA, P. H.; GOLDBARG, E. F. G. Algoritmos evolucionários na solução do problema do caixeiro alugador. In: LOPES, H. S.;TAKAHASHI, R. H. C. (Org.). Computação Evolucionária em Problemas de Engenharia. Curitiba: Omnipax, 2011, p. 301-330.

GOLDBARG, M. C.; GOLDBARG, E. F. G., MENEZES, M.S., PACCA LUNA, Henrique P. Quota traveling car renter problem: Model and evolutionary algorithm. In: Information Sciences. Volumes 367–368, 1 Novembro 2016, Pags. 232-245, ISSN 0020-0255.

GUTIN, G.; PUNNEN, A.The traveling salesman problem and its variations.Kluwer Academic Publishers, 2002.

LABADIE, N.; MELECHOVSKý, J.; CALVO, R. W. Hybridized evolutionary local search algorithm for the team orienteering problem with time windows. Journal of Heuristics, Springer US, v. 17, n. 6, p. 729-753, 2011. ISSN 1381-1231.

MARINAKIS, Y.; MIGDALAS, A.; PARDALOS, P. Expanding neighborhood grasp for the traveling salesman problem. Computational Optimization and Applications, Kluwer Academic Publishers, v. 32, n. 3, p. 231-257, 2005. ISSN 0926-6003.

MENEZES, Matheus da Silva, GOLDBARG, Marco César, GOLDBARG, Elizabeth F. G. A Memetic Algorithm for the Prize-collecting Traveling Car Renter Problem. IEE Congress on Evolutionary Computation (IEEE CEC 2014) Beijing, China.

MLADENOVIC, N. A variable neighborhood algorithm: a new metaheuristics for combinatorial optimization. In: Abstracts of Papers Presented at Optimization Days. Montreal. 1995

SOUFFRIAU, W. et al. A greedy randomised adaptive search procedure for the team orienteering problem. In: EU/MEeting 2008 on metaheuristics for logistics and vehicle routing, Troyes, France, 23-24 October 2008.

VANSTEENWEGEN, P.; SOUFFRIAU, W.; OUDHEUSDEN, D. V. The orienteering problem: A survey. European Journal of Operational Research, v. 209, n. 1, p. 1-10, 2011.

TABELA I. RESULTADOS PARA AS INSTÂNCIAS EUCLIDIANAS

Problemas			Exato	GRASP1				GraspPR				Grasp2PSorteio				Grasp2PRoleta			
Instancia	**nCid**	**nCar**	**GLPK**	**Min**	**Media**	**#vezes**	**T (s)**	**Min**	**Media**	**#vezes**	**T (s)**	**Min**	**Media**	**#vezes**	**T (s)**	**Min**	**Media**	**#vezes**	**T (s)**
Mauritania10e	10	2	**422**	**422**	422	30	1	**422**	422	30	1	**422**	422	30	1	**422**	422	30	1
Colombia11e	11	2	**326**	**326**	326	30	1	**326**	326	30	1	**326**	326	30	1	**326**	326	30	1
Angola12e	12	2	**490**	**490**	490	30	1	**490**	490	30	1	**490**	490	30	1	**490**	490	30	2
Peru13e	13	2	**556**	**556**	556	25	1	**556**	556	25	1	**556**	559	7	2	**556**	560	16	2
Libia14e	14	2	**521**	**521**	529	10	2	**521**	521	16	2	**533**	533	26	2	**533**	544	5	2
BrazilRJ14e	14	2	**230**	**230**	233	1	2	**230**	232	15	2	**230**	230	28	2	**230**	230	26	2
Congo15e	15	2	**513**	**513**	513	28	2	**513**	514	24	2	**513**	513	30	2	**513**	520	4	2
Argentina16e	16	2	**719**	**719**	729	6	2	**719**	727	9	2	**719**	727	3	2	**719**	729	2	3
EUA17e	17	2	**602**	610	639	4	3	610	642	1	3	**602**	623	3	3	**602**	620	2	3
Bolivia10e	10	3	**384**	**384**	384	30	1	**384**	384	30	1	**384**	384	30	1	**384**	386	22	2
AfricaSul11e	11	3	**402**	**402**	402	30	1	**402**	402	30	1	**402**	402	30	2	**402**	402	30	2
Niger12e	12	3	**564**	**564**	564	30	1	**564**	564	30	2	**564**	564	30	2	**564**	567	24	2
Mongolia13e	13	3	**543**	544	545	1	2	545	545	15	2	**543**	545	1	2	**545**	545	28	2
Indonesia14e	14	3	**504**	**504**	517	15	2	**504**	515	10	2	**504**	519	14	3	**504**	532	8	2
Argelia15e	15	3	**487**	**487**	492	13	2	**487**	493	12	2	**487**	489	21	3	**487**	501	1	3
India16e	16	3	**705**	**705**	724	5	2	**705**	725	5	2	**705**	705	28	3	**705**	721	6	3
China17e	17	3	**735***	741	762	3	2	**736**	762	1	3	**736**	744	6	3	**736**	766	1	3
Etiopia10e	10	4	**283**	**283**	283	30	1	**283**	283	30	1	**283**	383	30	1	**283**	283	30	3
Mali11e	11	4	**428**	**428**	428	30	1	**428**	428	30	2	**428**	428	30	2	**428**	428	30	2
Chade12e	12	4	**655**	**655**	676	1	2	**655**	673	4	2	**655**	673	3	2	**675**	682	1	2
Ira13e	13	4	**532**	**532**	541	12	2	**532**	540	17	2	**532**	534	20	3	**532**	583	1	3
Mexico14e	14	4	**492**	**492**	492	30	2	**492**	492	29	2	**492**	492	30	3	**492**	492	30	3
Sudao15e	15	4	**422**	**422**	422	30	2	**422**	422	30	2	**422**	422	30	3	**422**	422	30	3
Australia16e	16	4	**682**	**711**	777	1	3	**711**	775	1	3	730	762	1	4	727	788	1	4
Canada17e	17	4	**783**	810	878	1	3	817	879	1	3	**783**	836	1	4	837	924	1	5
Arabia14e	14	5	**482**	**482**	482	28	2	**482**	482	30	2	**482**	482	30	3	**482**	490	15	4
Cazaquistao15e	15	5	**574**	589	611	2	2	**585**	607	1	3	**585**	600	2	4	589	640	1	3
Brasil16e	16	5	**619**	**637**	647	3	2	**637**	648	3	3	**637**	652	1	4	638	658	9	4
Russia17e	17	5	**760***	790	821	1	3	781	818	1	3	**775**	814	1	4	799	821	1	5
BrasilAM26e	25	3	371*	372	392	1	6	349	366	1	7	**342**	373	1	9	349	375	1	9
BrasilMG30e	26	3	412*	431	445	1	8	391	423	1	9	**382**	423	1	14	394	439	1	13
BrasilCO40e	40	5	-	636	677	1	22	599	632	1	26	**597**	648	1	36	609	640	2	32
att48eA	48	3	25234*	26535	27903	1	30	**23268**	26569	1	30	26663	28347	1	42	30457	32545	1	37
att48eb	48	4	26555*	27719	29658	1	35	**26522**	28678	1	36	27125	29705	1	49	29101	32801	1	44
berlin52eA	52	3	**5802***	6514	6803	1	37	6154	6694	1	37	**6063**	6807	1	51	7381	7888	1	46
berlin52eB	52	4	6871*	6130	6537	1	39	**5667**	6240	1	39	6390	6652	1	55	7031	7294	1	49
st70eA	70	3	**1494***	1804	1902	1	87	**1801**	1867	1	89	1817	1906	1	105	2028	2093	1	103
st70eB	70	4	1708*	1689	1809	1	116	**1689**	1752	1	118	1707	1792	1	133	1894	1991	1	132
eil76eA	76	3	**1563***	1898	1938	1	107	1853	1910	1	109	**1833**	1919	1	148	1920	1976	1	129
eil76eB	76	4	**1216***	1657	1734	1	124	**1588**	1692	1	126	1607	1687	1	171	1712	1821	1	145

TABELA II. RESULTADOS PARA AS INSTÂNCIAS NÃO EUCLIDIANAS

Problemas			Exato	GRASP1				GraspPR				Grasp2PSorteio				Grasp2PRoleta			
Instancia	**nCid**	**nCar**	**GLPK**	**Min**	**Media**	**#vezes**	**T (s)**	**Min**	**Media**	**#vezes**	**T (s)**	**Min**	**Media**	**#vezes**	**T (s)**	**Min**	**Media**	**#vezes**	**T (s)**
Mauritania10n	10	2	**306**	**306**	306	30	1	**306**	306	30	1	**306**	306	30	1	**306**	306	30	1
Colombia11n	11	2	**461**	**461**	461	30	1	**461**	461	30	1	**461**	461	30	1	**461**	461	30	1
Angola12n	12	2	**409**	**409**	409	30	1	**409**	409	30	1	**409**	409	30	2	**409**	409	30	2
Peru13n	13	2	**502**	**502**	502	30	1	**502**	502	30	1	**502**	502	25	2	**502**	503	18	2
Libia14n	14	2	**504**	**504**	504	10	2	**504**	504	7	2	**504**	504	30	2	**504**	504	13	2
BrazilRJ14n	14	2	**101**	**101**	101	30	2	**101**	101	30	2	**101**	101	24	2	**101**	101	5	2
Congo15n	15	2	**573**	**573**	573	21	2	**573**	573	25	2	**573**	573	29	2	**573**	573	30	2
Argentina16n	16	2	647*	**642**	645	4	2	643	645	3	2	648	658	1	3	646	666	1	3
EUA17n	17	2	**579**	589	602	1	2	589	600	2	2	583	590	1	3	**580**	588	3	3
Bolivia10n	10	3	**448**	**448**	448	30	1	**448**	448	30	1	**448**	448	30	1	**448**	448	30	2
AfricaSul11n	11	3	**537**	**537**	537	30	1	**537**	537	30	1	**537**	537	30	2	**537**	537	30	2
Niger12n	12	3	**607**	608	614	1	1	609	614	1	1	**607**	608	11	2	**607**	613	6	2
Mongolia13n	13	3	**551**	**551**	551	28	1	**551**	551	27	1	**551**	551	30	2	**551**	551	26	3
Indonesia14n	14	3	**522**	**522**	522	30	1	**522**	522	28	2	**522**	522	30	2	**522**	522	30	2
Argelia15n	15	3	619*	**616**	620	2	2	617	621	3	2	617	620	2	3	620	628	1	3
India16n	16	3	734*	**723**	729	2	2	**723**	728	5	3	**723**	726	5	3	**723**	731	1	3
China17n	17	3	651*	**646**	659	1	3	**646**	659	2	3	651	661	2	4	656	665	1	4
Etiopia10n	10	4	**403**	**403**	403	30	1	**403**	403	30	1	**403**	403	30	2	**403**	403	30	2
Mali11n	11	4	**494**	**494**	494	22	2	**494**	494	17	2	**494**	494	26	2	**494**	494	22	2
Chade12n	12	4	654*	**649**	650	12	2	**649**	650	20	2	**649**	650	18	3	**649**	655	4	3
Ira13n	13	4	697*	**693**	693	24	2	**693**	693	20	2	**693**	693	12	3	**693**	694	3	3
Mexico14n	14	4	620*	**610**	610	27	2	**610**	610	28	2	**610**	611	21	3	**610**	611	24	3
Sudao15n	15	4	793*	**769**	773	9	2	**769**	774	4	3	**769**	774	3	4	771	778	2	4
Australia16n	16	4	551*	**525**	537	1	3	**525**	534	3	3	**525**	537	2	4	534	545	1	4
Canada17n	17	4	827*	827	898	1	3	834	886	1	3	**825**	859	2	4	834	931	1	4
Arabia14n	14	5	701*	**688**	705	1	3	**688**	698	1	3	**688**	694	14	4	**688**	712	5	4
Cazaquistao15n	15	5	843*	833	859	1	3	**830**	859	1	3	831	855	1	4	834	870	1	4
Brasil16n	16	5	769*	**742**	742	13	3	**742**	742	16	3	**742**	742	25	4	**742**	742	24	4
Russia17n	17	5	820*	781	790	3	3	779	792	1	4	**778**	789	1	5	780	803	1	5
BrasilAM26n	25	3	**107***	116	121	1	6	110	65	2	7	**109**	112	3	9	110	117	1	9
BrasilMG30n	26	3	179*	**171**	186	1	9	173	180	4	10	**171**	179	2	14	**171**	179	1	13
BrasilCO40n	40	5	--	444	469	1	25	416	436	2	28	**407**	439	1	35	452	482	1	35
att48nA	48	3	**616***	671	751	1	32	684	723	2	33	**648**	736	1	38	738	799	1	37
att48nB	48	4	**484***	655	712	1	45	**618**	673	1	42	659	699	1	49	729	775	1	50
berlin52nA	52	3	**812***	953	1026	1	38	938	995	1	39	**923**	983	1	46	912	1077	2	45
berlin52nB	52	4	**520***	638	716	1	49	**646**	691	1	49	651	695	1	58	679	772	1	59
st70nA	70	3	**764***	**993**	1203	1	81	1055	1162	1	92	1104	1193	1	108	1199	1305	1	109
st70nB	70	4	**725***	**870**	992	1	95	890	957	1	96	876	967	1	145	979	1053	1	139
eil76nA	76	3	**965***	1225	1337	1	106	**1194**	1261	1	122	1252	1348	1	140	1361	1483	1	138
eil76nB	76	4	**746***	1034	1118	1	110	**917**	1001	1	120	938	1045	1	150	1091	1189	1	146