HEURÍSTICA APLICADA AO PROBLEMA DE AGRUPAMENTO CENTRADO CAPACITADO

Rodrigo de Carvalho
Universidade Federal de Minas Gerais
Av. Antônio Carlos, 6627 - Pampulha - Belo Horizonte - MG - CEP 31270-901
rodrigo@rclink.com.br

Rodney Rezende Saldanha
Universidade Federal de Minas Gerais
Av. Antônio Carlos, 6627 - Pampulha - Belo Horizonte - MG - CEP 31270-901
rodney@cpdee.ufmg.br

Alexandre Xavier Martins
Universidade Federal de Ouro Preto
Rua 36, 115 - Bairro Loanda - João Monlevade - MG - CEP: 35931-008
xmartins@decea.ufop.br

RESUMO

Este artigo apresenta um algoritmo heurístico multi partida, baseado no método GRASP, para o problema de agrupamento centrado capacitado. As soluções são inicializadas a partir de um método gerador de centróides virtuais eficiente com a colaboração de um método semi guloso. Para a fase de busca local é utilizado a idéia do método ILS, que possui mecanismos de fuga de ótimos locais. Para medir o desempenho do algoritmo foram utilizados três conjuntos de problemas com diferentes características, além de três instâncias propostas nesse trabalho. Os resultados mostraram que o algoritmo implementado é competitiva, quando comparado com outras metodologias propostas na literatura.
PALAVRAS CHAVE. GRASP. Agrupamento. Heurística.

ABSTRACT

This paper presents a multi-start heuristic for the capacitated centered clustering problem. The solutions are initialized from an efficient method for virtual centroid generation combined with a greedy randomized search procedure which provided good start solutions. In the analysis of the performance of the algorithms implemented were used three sets of benchmark problems with different characteristics, and three new instances. The results found show that the proposed algorithm is competitive to other methodologies in literature.
KEYWORDS. GRASP. Clustering. Heuristics.

1. Introdução

É imprescindível que os provedores de serviços estejam bem localizados em relação as suas regiões de atendimento. Uma boa localização pode influenciar diretamente na qualidade serviço e na margem de lucro das empresas, uma vez que os atendimentos podem ser mais eficientes em vários aspectos, tais como: tempo, custo de deslocamento e área de abrangência. Na literatura, esses provedores de serviços têm sido chamados de facilidades, e podem assumir diversos contextos, tais como fábricas, depósitos, antenas, entre outros.

Assim, decidir onde alocar uma ou mais facilidades em uma região geográfica é uma atividade estratégica de grande importância para muitas organizações, sejam elas públicas ou privadas, assim como as empresas de transportes precisam decidir onde alocar seus centros de apoio. Empresas especializadas em vendas precisam alocar seus vendedores em regiões estratégicas, de modo que possam atender um determinado conjunto de clientes em um intervalo de tempo. Hospitais e delegacias precisam ser distribuídos de forma que atendam uma região de maneira eficiente, suprimindo o máximo da demanda de cada unidade da região.

Em todos estes contextos, dado um conjunto de pontos distribuídos em um plano, o objetivo é agrupá-los de maneira que otimize uma métrica respeitando as restrições impostas pelo problema, particulares em cada caso. Cada grupo deve conter uma facilidade responsável no atendimento aos demais integrantes. Nesse trabalho é estudado um problema dessa natureza, conhecido como Problema de Agrupamento Centrado Capacitado (PACC).

O PACC está inserido na classe de problemas NP, mais especificamente na classe conhecida como NP-Difícil. Para tais problemas, não é conhecido nenhum método determinístico capaz de encontrar a solução ótima com complexidade de tempo polinomial independente da dimensão do problema. Assim, neste artigo, é proposta uma heurística baseada nos métodos GRASP e ILS, que tem como principais contribuições o método de construção, que é capaz de gerar boas soluções rapidamente, uma busca local eficiente que evita explorar soluções que não são promissoras além de um procedimento de perturbação eficiente.

Este trabalho está organizado como segue. Na próxima seção será feita uma descrição do problema, assim como uma breve revisão bibliográfica. Na seção 3 será apresentada a metodologia utilizada. A seção 4 apresentará os resultados experimentais. Finalmente, na seção 5 será feita a conclusão desse trabalho.

2. Problema de Agrupamento Centrado Capacitado

O PACC consiste em localizar p facilidades em um espaço euclidiano $\mathbb{R}^2$ de modo que atenda n pontos de demanda, objetivando minimizar a soma total das distâncias entre os pontos de demanda e sua facilidade. Nesse problema cada facilidade possui uma capacidade de atendimento limitada e sua localização é dada a partir do centro geométrico do grupo.

Assim, o PACC pode ser visto como um problema de agrupamento, o qual dado um conjunto de pontos, é preciso agrupa-los de tal forma que os elementos pertencentes a um mesmo agrupamento tenham a menor dissimilaridade possível, enquanto os elementos menos similares, fiquem agrupados separadamente. A medida de dissimilaridade é calculada a partir da distância euclidiana entre o centróide, obtido a partir das coordenadas dos pontos pertencentes ao mesmo agrupamento, e os pontos de demanda.

Para descrever o modelo matemático do PACC as seguintes definições foram adotadas: consideremos um vetor $\overrightarrow{a}$ contendo as coordenadas de um ponto i no espaço euclidiano, um vetor $\overrightarrow{q}$ com as coordenadas do centroide do agrupamento j, n_j o número de pontos no agrupamento j, d_i a demanda do ponto i, Q_j a capacidade do agrupamento j e y_{ij} uma variável que assume o valor 1 quando o ponto i está alocado ao agrupamento j, e 0 caso contrário. Além disso, considere N sendo o número total de pontos de uma instância e P o número total de grupos que devem ser criados. Assim, o PACC pode ser modeladode acordo com as Eq. (1)-(7):

$$minz = \sum_{i=1}^{n}\sum_{j=1}^{p} |\overrightarrow{a_i} - \overrightarrow{q_j}|^2 y_{ij} \quad (1)$$

$$\sum_{j=1}^{p} y_{ij} = 1, \quad \forall\, i = 1, ..., N \quad (2)$$

$$\sum_{i=1}^{n} y_{ij} = n_j \quad \forall\, j = 1, ..., P \quad (3)$$

$$\sum_{i=1}^{n} \overrightarrow{a_i} y_{ij} \leq n_j q_j \quad \forall\, j = 1, ..., P \quad (4)$$

$$\sum_{i=1}^{n} d_i y_{ij} \leq Q_j \quad \forall\, j = 1, ..., P \quad (5)$$

$$y_{ij} \in \{0, 1\} \quad n_j \in N, q_j \in \mathbb{R} \quad (6)$$

onde a função objetivo (1) minimiza o somatório do quadrado da diferença entre as coordenadas de cada ponto e o centróide do agrupamento a que este pertence. As restrições (2) determinam que um ponto deve estar associado exatamente a um cluster. As restrições (3) e (4) fornecem o número de pontos em cada cluster e as coordenadas do centróide, respectivamente. As restrições (5) determinam que a capacidade de cada agrupamento deve ser respeitada. As restrições de integridade das variáveis são descritas em (6). O CCCP é um problema *NP*-difícil e, além disso, o cálculo de dissimilaridade é dado por uma função não-linear, dificultando ainda mais sua solução por métodos exatos. Por isso, na literatura, são encontrados frequentemente métodos heurísticos com o objetivo de localizar soluções eficientes para esse problema.

Em Negreiros e Palhano (2006) é proposto um algoritmo heurístico de duas fases para resolver o PACC. A primeira fase utiliza o algoritmo Forgy (Forgy (1965)) para construir uma solução inicial. Com o objetivo de melhorar o desempenho desse algoritmo foi utilizada uma estrutura de dados baseada em árvore. A segunda fase consiste em um refinamento utilizando a metaheurística VNS (Mladenovic (1995)). Instâncias realacionadas a problemas de distribuição de força de vendas de uma indústria alimentícia e projetos de zona de coleta de lixo foram utilizadas para testar o desempenho deste algoritmo . Os resultados apontaram um ótimo desempenho, onde boas soluções foram alcançads em tempos computacionais relativamente baixos.

Em Stefanello e Muller (2009) é feito um estudo sobre o uso do revolvedor comercial CPLEX e modelos matemáticos para resolver problemas de agrupamento capacitado. O objetivo deste trabalho foi, entre outros, inferir a dificuldade de resolução dos conjuntos de instâncias comumente utilizados na literatura científica para o PACC. Para propiciar o uso do resolvedor comercial disponível, o problema foi modelado como um problema de programação inteira quadrática. Os resultados obtidos após uma hora de processamento mostraram que mesmo para pequenas instâncias não é possível obter bons resultados.

Chaves e Lorena (2009) apresentaram um procedimento para obter soluções para o PACC usando a metaheurística *Clustering Search* (CS), cuja ideia principal é identificar áreas promissoras do espaço de busca, gerando soluções e agrupando-as para que posteriormente sejam explorados com heurísticas de busca local. O CS apresentou os melhores resultados na maioria dos casos em relação a outras metaheurísticas, tais como *Simulated Annealing* e VNS(Negreiros e Palhano (2006)). Chaves e Lorena (2011) extenderam a ideia do CS inserindo um algoritmo genético com objetivo de aumentar a capacidade de exploração do espaço de busca. Os resultados mostraram um desempenho muito satisfatório .

Um algoritmo baseado nas metaheurísticas Busca Tabu e *path-relinking* foi proposto por Oria et al. (2012). Esse método foi composto por duas fases, na primeira, soluções são criadas a partir de uma heurística construtiva, e posteriormente combinadas a partir da técnica de *path-reliking*. Na segunda fase é aplicada o método de Busca Tabu para fazer uma busca local. Embora essa metodologia tenha apresentado bons resultados, o tempo computacional foi superior quando comparado a outros métodos da literatura.

Muritiba et al. (2012) propuseram um conjunto efetivo de estratégias que se combinam à metaheurística clássica Busca Tabu. Primeiro, na fase de construção, são sorteados alguns pontos para compor os agrupamentos iniciais, posteriormente, os pontos restantes são alocados no agrupamento mais próximo que não violem, ou violem o mínimo, a capacidade dos agrupamentos. Esse procedimento é repetido 10 vezes, e a melhor solução encontrada é retornada para que o procedimento de busca local seja aplicado. Em seguida, é aplicada uma busca local utilizando 3 tipos de movimentos, denominados *transfer*, *swap* e *wave*. O movimento *swap* tem como objetivo trocar pontos de agrupamentos distintos desde que não viole a sua capacidade. Já o *tranfer* transfere os pontos de agrupamentos (um a um) respeitando a capacidade dos agrupamentos. O movimento *wave* troca um ponto de agrupamento, caso a troca resulte em uma melhora na função objetivo e a restrição de capacidade não seja respeitada, é aplicado um tipo de procedimento de correção. Esse procedimento consiste em realocar o nó mais distante do agrupamento no qual o novo ponto foi alocado em um novo agrupamento, utilizando a mesma estratégia. Tal procedimento é feito recursivamente até que as capacidades não sejam mais violadas ou o procedimento seja executado no maximo 30 vezes. Os resultados mostraram que essa metodologia foi muito eficiente, apresentando resultados superiores em aproximadamente 80% das instâncias quando comparado a outros trabalhos da literatura.

Em Oliveira et al. (2013) foi aplicado ao PACC uma variação da metaheurística CS e os resultados obtidos foram comparados a outras duas abordagens encontradas na literatura deste mesmo método propostas nos trabalhos Chaves (2009) e Chaves e Lorena (2011). A principal diferença deste trabalho é a substituição dos métodos Simulated Annealing e Algoritmo Genético pela metaheurística ILS Lourenco et al. (2003). Os resultados obtidos mostraram que as diferentes abordagens do CS obtiveram desempenho similar, com uma pequena vantagem para o método proposto.

Maravilha (2014) propôs uma variação do algoritmo Evolução Diferencial (DE, do inglês *Differential evolution*) para problemas de otimização combinatória. Vale ressaltar que originalmente o DE (Storn e Price (1995)) foi projetado para otimização com varáveis no espaço contínuo. O método proposto foi aplicado ao PACC e obteve resultados satisfatórios quando comparada a outras abordagens para solução de problemas de otimização combinatória.

3. Metodologia

Nesta seção a heurística proposta é apresentada. Serão detalhados a representação computacional de uma solução, bem como a função de avaliação, o método de construção, pertubações e busca local.

3.1. Representação de uma solução

O esquema de codificação utilizado nesse trabalho é conhecido como *group-number*, no qual um vetor de inteiros é usado para representar um agrupamento de n elementos. Neste vetor a i-ésima posição indica o número do grupo do i-ésimo elemento (Cole (1998)).

Além disso, para cada grupo é utilizado uma lista contendo a localização do centróide, capacidade utilizada, a distância aproximada do centroíde aos centroídes de outros agrupamentos, a quantidade e quais são os pontos pertencentes ao grupo e sua distância em relação ao centroíde. Todas essas informações são utilizadas durante o processo de otimização com o objetivo de tornar o algoritmo mais eficiente, evitando a busca de soluções em espaços não promissores e auxiliando

no cálculo da função objetivo. Isso será detalhado nas proximas seções.

3.2. Função de avaliação

Uma solução é avaliada a partir da soma das distâncias dos pontos ao centro geométrico dos grupos no qual estão associados. Além disso, para evitar, em algumas ocasiões, o processo de verificação de viabilidade em relação à restrição de capacidade do agrupamento, essa foi inserida na função de avaliação. Nesse caso, é aplicado uma penalidade nas regiões não factíveis, ou seja, regiões onde as restrições estão sendo violadas.

Assim, temos a função de avaliação utilizada neste trabalho representada pela Eq. (7). O lado esquerdo da adição representa a distância euclidiana, já a parte direita contabiliza, caso necessário, o quanto uma solução violou a restrição de capacidade. ϖ é um parâmetro que penaliza a função objetivo para cada unidade violada em relação a capacidade. O valor dessa penalização foi obtido após uma série de experimentos, usando todas as instâncias do problema e configurada em 1000.

Convém ressaltar que para as instâncias utilizadas neste trabalho este valor foi suficiente, mas para outras instâncias onde a distância entre pontos possa superar 1000, deve se utilizar um valor maior para que o método proposto não tenda a convergir para soluções inviáveis.

$$f(y) = \sum_{i=1}^{n}\sum_{j=1}^{p} |\overrightarrow{a_i} - \overrightarrow{q_j}|^2 y_{ij} + \varpi \sum_{j}^{p} \left[\max\left(0, \sum_{i}^{n} d_i y_{ij} - Q_j\right)\right] \tag{7}$$

3.3. Método de Construção

A construção de uma solução neste trabalho pode ser resumida em três passos: inicialmente é necessário calcular o valor médio $\overrightarrow{\mu}$ e o desvio padrão $\overrightarrow{\sigma}$ das coordenadas ($\overrightarrow{coord}$) de todos os pontos da instância a ser agrupada. Posteriormente, é aplicado um método que procura gerar centróides temporários distribuídos uniformemente no espaço usando as informações de $\overrightarrow{\mu}$ e $\overrightarrow{\sigma}$. Finalmente, os pontos são atribuídos aos centróides temporários baseado no critério adotado no método de construção *Greedy Randomized Adaptive Search Procedure* (GRASP) proposto por Feo e Resende (1995).

No passo de criação de centróides temporários foi proposto o Algoritmo 1, denominado Gerador de Centroídes Temporários (GCI). O objetivo desse algoritmo é segmentar o espaço onde estão localizados os pontos em *grids*, que posteriormente receberam seus centroídes localizados no centro de cada espaço. Os *grids* são criados considerando as seguintes informações: número de agrupamentos p; média $\overrightarrow{\mu}$ e desvio padrão $\overrightarrow{\sigma}$ das coordenadas dos pontos e; espaçador κ. O número de agrupamentos define aproximadamente a quantidade de linhas e colunas do *grid*. A média define onde o *grid* estará centrado, enquanto desvio padrão junto ao parâmetro espaçador κ define o tamanho de cada quadrante do *grid*.

Assim, as linhas 2 e 3 desse algoritmo definem o número de linhas e colunas do *grid*. As linhas 4-13 buscam identificar em qual eixo os pontos estão mais concentrados, sendo possível posicionar os centroides de forma mais eficiente, buscando aloca-los próximos aos pontos da instância. As linhas 14-28 localizam os centroídes na sua posição correta. O valor 0.5 utilizado na linha 22 localiza o centroíde no centro do *grid*.

O algoritmo CGI se mostrou eficiente onde os pontos das instâncias estão bem distribuídos no plano. Nesse cenário, os centróides temporários são localizados de forma eficiente, tornando-os bons centros de atração. Já nas instâncias onde existem uma grande concentração de pontos em áreas distintas do plano, ilhas de pontos, a alocação dos centróides é menos eficiente. A Figura 1 ilustra esses cenários, o qual dado as diferentes distribuições de pontos no plano, são mostrados os centróides criados inicialmente pelo CGI, representados aqui por triângulos. Uma má distribuição

Algoritmo 1: Algoritmo GCI

Entrada: $\overrightarrow{\mu}$,$\overrightarrow{\sigma}$,κ,p,$\overrightarrow{coord}$
Saída: s

$s = []$
$seg_x \leftarrow \lfloor\sqrt{(p)}\rfloor$
$seg_y \leftarrow \lfloor p \div seg_x\rfloor$
$\overrightarrow{ref_x} \leftarrow [(\mu_x + \sigma_x\kappa)\ (\mu_x - \sigma_x\kappa)]$
$\overrightarrow{ref_y} \leftarrow [(\mu_y + \sigma_y\kappa)\ (\mu_y - \sigma_y\kappa)]$
$ct \leftarrow (max(coord_x) - min(coord_x)) - (max(coord_y) - min(coord_y))$
se $ct > 0$ **então**
    $seg_x \leftarrow max(seg_x, seg_y)$
    $seg_y \leftarrow min(seg_x, seg_y)$
senão
    $seg_x \leftarrow min(seg_x, seg_y)$
    $seg_y \leftarrow max(seg_x, seg_y)$
fim se
$lb_x \leftarrow ref_x(2)$
$ic_x \leftarrow (ref_x(1) - ref_x(2)) \div seg_x$
$ic_y \leftarrow (ref_y(1) - ref_y(2)) \div seg_y$
$ub_x \leftarrow ic_x + lb_x$
para $ix \leftarrow 1$ *até* seg_x **faça**
    $lb_y \leftarrow ref_y(2)$
    $ub_y \leftarrow ic_y + lb_y$
    para $iy \leftarrow 1$ *até* seg_y **faça**
        $\overrightarrow{s} \leftarrow \{\overrightarrow{s}, [(lb_x + (ub_x - lb_x) \times 0.5)\ (lb_y + (ub_y - lb_y) \times 0.5)]\}$
        $lb_y \leftarrow ub_y$
        $ub_y \leftarrow ub_y + ic_y$
    fim para
    $lb_y \leftarrow ub_y$
    $ub_y \leftarrow ub_y + ic_y$
fim para

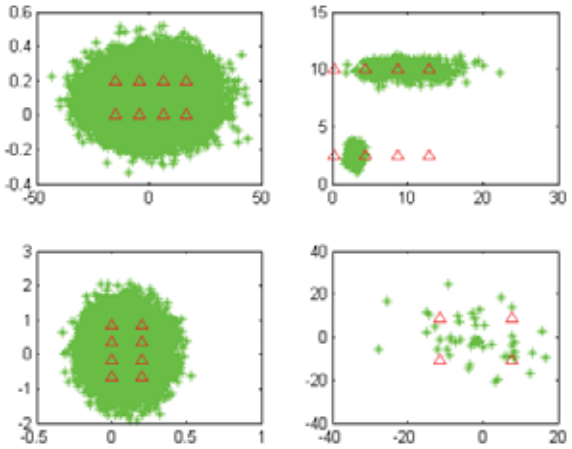
é representada quando existe um centróide e não existem pontos distribuídos nas proximidades.

Vale ressaltar que as instâncias da literatura, que são comumente utilizadas, não possuem a característica de ilhas de pontos. Assim, para verificar o desempenho do método proposto, nessa situação, foram propostas 3 novas instâncias.

A etapa de construção da metaheurística *GRASP* foi utilizada para atribuir os pontos da instância para um dos centróides temporários criados pelo Algoritmo 1. Esse método pode ser considerado iterativo, guloso, randômico e adaptativo. É considerado iterativo pelo fato de construir uma solução ponto a ponto, já a característica adaptativa é devido a escolha da próxima atribuição de um ponto a um centróide ser influenciada pelas escolhas anteriores. Essa escolha considera, a princípio, todas as possíveis atribuições, contudo, o número de possibilidades pode em muitos casos ser extremamente elevado, por isso é considerada apenas uma lista restrita de candidatos (*LRC*). Os elementos contidos na *LRC* devem ter sua função de avaliação no seguinte intervalo $[fp_{min}, f_{min} + \ell(f_{max} - f_{min})]$.

Onde f_{max} e f_{min} são, respectivamente, o elemento de maior e o elemento de menor incremento na função de avaliação da solução e ℓ um parâmetro de entrada que pode variar entre 0

Figura 1: Diferentes resultados do método GCI



e 1. Quanto mais próximo de 1 for o valor de ℓ, mais aleatórias são as soluções geradas devido à diversidade de valores que irão compor a LRC. Um elemento da LRC é escolhido aleatoriamente, o que da a característica de randômica para o método. Esta escolha aleatória permite que este procedimento possa ser usado várias vezes para gerar diferentes soluções.

Assim, um a um, os pontos da instância a ser otimizada são escolhidos e atribuídos a um dos centróides temporários. Após essa etapa, os centróides temporários são substituídos pelos centróides baseados nas coordenadas dos pontos de cada agrupamento.

Este método foi capaz de gerar boas soluções em um curto espaço de tempo, sendo necessário para a maior instância utilizada nesse trabalho apenas 1 segundo de processamento. Salienta-se que quando comparado com outras duas metodologias, proposto por Muritiba et al. (2012) e um método aleatório, apresentou os melhores resultados.

3.4. Busca Local

Seja s uma solução para o PACC, é necessário definir uma estrutura de vizinhança, ou seja, uma função V, na qual podemos chegar a s' por meio de uma modificação, geralmente chamada de movimento.

Dessa forma, para exploração do espaço de soluções, foram utilizados apenas dois tipos de movimentos nesse trabalho. O primeiro, denominado troca, consiste na troca de dois pontos alocados em diferentes agrupamentos. Já o segundo, denominado realocação, busca alocar um ponto em um novo agrupamento.

A busca local tem como objetivo encontrar uma solução de melhor qualidade que possa ser alcançada a partir de uma solução s e uma estrutura de vizinhança V. As soluções que são geradas por V geralmente são exploradas de duas formas, conhecidas como: primeiro de Melhora (do inglês *First Improving*) e Melhor Aprimorante (do inglês *Best Improving)*. Na primeira são verificadas as soluções até que se encontre uma solução que melhore a solução corrente. A segunda, Melhor Aprimorante, todas as soluções geradas por V são verificadas, retornando a melhor entre elas. Neste trabalho foi utilizada a estratégia Primeiro de Melhora, pois apresentou os melhores resultados em testes preliminares.

Com o objetivo de tornar mais eficiente a busca local, apenas parte da vizinhança é explorada. Um movimento é executado apenas entre agrupamentos onde os centróides estão próximos, evitando trocas desnecessárias, onde a distância entre os pontos são grandes. Essa proximidade é calculada no método de construção, levando em consideração a distância entre os centróides tem-

porários e armazenada em uma lista contendo as informações sobre os agrupamentos. Assim, os movimentos são aplicados apenas entre os r primeiros agrupamentos vizinhos.

Além disso, para ambos os movimentos é verificado se os agrupamentos possuem viabilidade em relação a capacidade disponivél antes de executá-los . Tal procedimento é feito devido ao cálculo da função objetivo possuir um custo computacional alto e também pelo fato de soluções que violam a restrição de capacidade possuírem qualidade ruim.

3.5. Pertubações

Para permitir que a heurística proposta crie uma mecanismo de fuga de ótimos locais, foram propostas neste trabalho 3 estrégias de perturbação. A primeira, denominada P_1, consiste em desalocar aleatóriamente um número pré determinado de pontos e alocá-los novamente aos agrupamentos mais próximos que possuem capacidade de atendimento. Caso não seja possível a alocação devido a falta de capacidade do grupo, é considerada somente a distância como critério de alocação. Ao desalocar os pontos dos agrupamentos, os centróides devem ser corrigidos, porém na fase de alocação os centróides só serão atualizados depois que todos os pontos forem alocados. O número pré determinado de pontos é definido proporcionalmente ao tamanho da instância, neste trabalho foi utilizado 20% dos pontos, valor obtido a partir de testes empíricos.

A segunda estratégia, denominada P_2, difere da primeira pelo fato da desalocação/alocação ser feita totalmente de forma aleatória. Esse método é utilizado somente quando o algoritmo fica estagnado em um ótimo local durante algumas iterações, pois permite a metaheurística pesquisar áreas do espaço de busca mais distantes. O número de pontos utilizados nessa perturbação é o mesmo de P_1.

Finalmente, a terceira estratégia, denominada aqui de P_3, desaloca 50% dos pontos mais afastados de cada centróide. Posteriormente, esses pontos são inseridos nos agrupamentos mais próximos. Os centróides são corrigidos somente após o processo de desalocação/alocação de todos os pontos.

A cada iteração do algoritmo proposto, apenas uma das pertubações é aplicada, escolhida de acordo com o número de iterações sem melhora (ism) da heurística. São utilizados dois limiares, l_1 e l_2, assim, até que o limiar l_1 seja alcançado, a estratégia P_1 é utilizada. Ao se atingir l_1 a estratégia P_2 é utilizada até que l_2 seja alcançada, quando é utilizada a estratégia P_3. Após a utilização de P_3, o número de iterações sem melhora é zerado. Os valores de l_1 e l_2 foram configurados, respectivamente em 10 e 25. O algoritmo 2 ilustra o procedimento de pertubação.

Algoritmo 2: Pertubação

Entrada: s, ism
Saída: $s_{perturbada}$
se $ism < l_1$ **então**
  $s_{perturbada} \leftarrow P_1(s)$
senão
  se $ism < l_2$ **então**
    $s_{perturbada} \leftarrow P_2(s)$
  senão
    $s_{perturbada} \leftarrow P_3(s)\ ism \leftarrow 0$
  fim se
fim se

3.6. Heurística

Neste trabalho é proposto uma heurística construtiva multi-partida baseadas nos conceitos das metaheurísticas GRASP e ILS. A principal ideia é gerar boas soluções iniciais, aplicar o método de busca local e posteriormente, ao se cair em um ótimo local, aplicar perturbações moderadas para permitir uma melhor expliração no espaço de busca.

Algoritmo 3: Heurística

```
   Entrada: tempo
   Saída: s
1  ism ← 0
2  enquanto critério de parada não satisfeito faça
3  |  s ← criar solução inicial
4  |  se s* = {} então
5  |  |  s* ← s
6  |  fim se
7  |  enquanto critério de estagnação não alcançado faça
8  |  |  s' ← busca local(s)
9  |  |  s* ← criterio de aceitação(s',s*,ism)
10 |  |  s ← pertubação(s,ism)
11 |  fim enqto
12 fim enqto
```

O Algoritmo 3 ilustra os passos do método proposto. Na linha 1 é inicializado o número de iterações sem melhora (ism) com o valor 0. O método de construção proposto neste trabalho é utilizado na linha 3. A melhor solução encontrada pela heurística é armazena em $s*$, assim, na primeira iteração quando ainda não existe valores a serem comparados, a melhor solução passa a ser a solução inicial (linhas 4-6). O processo de busca é representado pelas linhas 7-11, onde os métodos de busca local e pertubação proposto neste trabalho são utilizados. O critério de aceitação consiste em escolher entre duas soluções aquela que apresenta melhor valor de função de avaliação. A cada chamada desse método onde $s*$ seja melhor que s' a variável ism é incrementada em uma unidade (linha 9). O critério de estagnação adotado neste trabalho (linha 7) consite em ism chegar ao valor de l_2 duas vezes seguindas. O processo é repetido até que o critério de parada do algoritmo seja satisfeito, nesse trabalho foi utilizado o tempo de execução.

4. Experimentos Computacionais e Análise Computacional

Essa seção tem como objetivo apresentar os resultados e discutir o desempenho do método proposto. Primeiro será feito uma breve explanação sobre as instâncias utilizadas. Posteriormente será detalhado o planejamento experimental que visa avaliar o desempenho do método em relação a algumas propostas da literatura. Por fim, será feita uma breve análise dos resultados.

4.1. Instâncias

Para fazer a avaliação do método proposto foram utilizadas 25 instâncias comumente utilizadas na literatura (Muritiba et al. (2012)), variando entre 100 a 13221 pontos e 3 instâncias criadas neste trabalho, variando de 100 a 1000 pontos. Embora o problema permita capacidades diferentes entre os agrupamentos, todas as instâncias utilizadas neste trabalho utilizam essa capacidade de forma homogênea, ou seja, todos os grupos possuem a mesma capacidade. As instâncias propostas, r_0, r_1 e r_2, diferem-se das propostas na literatura, pois possuem concentração de pontos em formas

de ilhas, e a relação capacidade *x* demanda é mais justa. O objetivo é verificar o desempenho dos algoritmos em situações diferentes das encontradas nas instâncias atuais.

4.2. Planejamento Experimental

A avaliação do desempenho entre os algoritmo estudados foi realizada por meio de um teste estátistico, que visa identificar se existe diferença estatística significante entre os algoritmos. Essa análise foi feita utilizando dados obtidos por cada algoritmo, contendo o valor da função de avaliação após 10 execuções independentes. O critério de parada utilizado para obter esse valores foram os tempos de execução reportados em Muritiba et al. (2012). Para as instâncias propostas neste trabalho foram dados 50 segundos para cada execução. Os testes foram executados em uma mesma máquina, de maneira aleatorizada, tanto em relação a instância quanto ao algoritmo. Além disso, todos os algoritmos utilizaram a mesma semente geradora de números aleatórios em cada replicação de cada instância. Esses procedimentos garantem condições experimentais homogêneas e indepêndencia dos erros, evitando a introdução de efeitos espúrios e tendenciosidades nas amostras (Montgomery e Runger (2003)).

O experimento foi projetado como um teste comparativo simples entre os algoritmos estudados. Para eliminar os efeitos das diferentes instâncias no desempenho dos métodos, cada uma delas foi tratada como bloco. O modelo estátistico que descreve os dados deste experimento é dada pela equação 8.

$$x_{ijk} = \mu_i + \beta_j + \varepsilon_{ijk} \tag{8}$$

Onde x_{ijk} é o valor da métrica calculada para o i-ésimo algoritmo, na j-ésima instância da k-ésima replicação. μ_i é a média real da métrica para o i-ésimo algoritmo. β_j é o efeito no valor da métrica devido a j-ésima instância. ε_{ijk} é a componente de erro aleatório. Ainda é possível decompor μ_i em $\mu + \tau_i$, que representam, respectivamente, a média geral e o efeito do algoritmo.

Para evitar suposições de que os dados apresentam uma distribuição normal, é utilizado o teste de Friedman (Montgomery e Runger (2003)). Esse visa testar as hipóteses nulas de ausência de diferença no desempenho dos algoritmos contra a hipótese alternativa de que pelo menos um algoritmo apresenta diferença em relação a um outro. Deste modo, o experimento foi definido da seguinte maneira:

- Hipóteses: H_0 (hipótese nula), os algoritmos não apresentam diferença significativa quando aplicados ao problema estudado, ou seja $\tau_1 = \tau_2 = \tau_3$. H_1 (hipótese alternativa), os algoritmos apresentem diferença significativa quando aplicados ao problema estudado;
- Fator: Algoritmos;
- Níveis: *Cluster Search+ILS* (Oliveira et al. (2013)), DE (Maravilha (2014)) e Busca tabu (Muritiba et al. (2012));
- Replicações: 10;
- Região de rejeição H_0: Consiste em todos os valores da estatística do teste utilizado, tais que sua probabilidade de ocorrência, sob H_0, não supere o valor do nível de significância adotado nos testes.

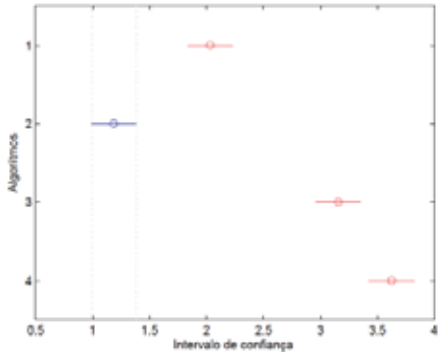
4.3. Resultados Computacionais

Ao aplicar o teste estatístico, foi indicada a existência de diferença significativa no desempenho dos métodos heurísticos, uma vez que H_0 foi rejeitado. Para tentar estimar onde existe a diferença a um nível de significância estudado, foi utilizada uma função fornecida no MATLAB,

denominada *multcompare*, que a partir de informações como média de rank e desvio padrão, tenta estimar onde existe a diferença.

Esse procedimento retorna um gráfico que pode ser interpretado da seguinte forma: duas médias são significativamente diferentes se os intervalos são disjuntos, e não são significativamente diferentes, se os seus intervalos se sobrepõem. Quanto mais a esquerda a média e desvio padrão, melhor o algoritmo. A Figura 2 mostra a média dos ranks e o intervalo de confiança obtidos, onde o eixo das coordenadas determinam os algoritmos, no caso 1,2,3 e 4 que são respectivamente a heurítica proposta neste trabalho, Busca Tabu, DE e CS+ILS. O eixo das abscissas representa o valor dos ranks.

Figura 2: Grafico de comparações múltiplas



E possível perceber que o método Busca Tabu é significantemente diferente dos demais, além disso, lenvando-se em consideracão que temos um problema de maximização, e que no processo de atribução de ranks o maior valor é dado para quem tem o maior valor da métrica, é possível chegar a conclusão que esse algorimo em média é superior aos demais, bem como, o algoritmo proposto neste trabalho é superior as outras duas abordagens.

Embora, quando considerado todo o experimento o método de Busca Tabu tenha apresentado os melhores resultados, para as instâncias r_0, r_1 e r_2 o método proposto foi superior. Existem duas diferenças significantes dessa instância em relação as outras, primeiro que a relação capacidade x demanda é mais justa, e os pontos estão divididos em formas de ilhas. Para essas instâncias foi percebido que as estratégias de pertubação, que não consideram a capacidade dos agrupamentos para realocação dos pontos, contribuiram significantemente nos resultados obtidos. As perturbações permitem que o algoritmo explore espaços de soluções inviáveis, mas que favorecem a convergência para melhores soluções quando utilizados os movimentos da busca local.

Além disso, considerando as instâncias variando entre 25 à 402 pontos, os resultados alcançados por ambos os métodos não apresentam diferença significativa. O Busca Tabu se destacou nas instâncias variando entre 3038 à 13000 pontos, apresentando na maioria dos experimentos resultados superiores em relação a qualidade. Um estudo mais aprofundado deverá ser feito para que possa ser explicada a diferença de comportamento do método proposto em relação a essas instâncias. Vale ressaltar que os outros algoritmos apresentaram resultados significantemente piores para todo o conjunto de instâncias. Levando em consideração o melhor valor de função objetivo, o algoritmo proposto ficou, em média, apenas 8% acima dessas soluções.

5. Conclusão

Este trabalho apresentou uma heurística baseado nos métodos GRASP e ILS para o problema de agrupamento centrado capacitado. Os resultados obtidos por esse algoritmo foram comparados com outras três hurísticas da literatura. Para que as conclusões obtidas possuíssem maior significância, foi utilizada uma análise estatísticas nos resultados obtidos. Desta forma, analisou-se o desempenho médio dos algoritmos em relação ao valor de função objetivo obtido em diferentes

instâncias. A análise estatística mostrou a existência de diferença significativa entre os algoritmos, sendo o algoritmo proposto em Muritiba et al. (2012) o que obteve os melhores resultados.

O algoritmo proposto se destacou em relação a qualidade da solução inicial gerada, além disso, mostrou-se mais eficiente nas instâncias propostas nesse trabalho. Em instâncias de médio porte, obteve o mesmo desempenho do melhor algoritmo estudado. Quando comparado com outras abordagens de heurísticas da literatura, o método proposto se mostrou consideravelmente mais eficiente.

Como trabalhos futuros, pretende-se entender o que causou o mau desempenho do algoritmo para algumas instâncias. Além disso, anexar o método de criação de soluções iniciais a outras heurísticas com o objetivo de analisar a diferença no desempenho destes métodos.

Referências

Chaves, A. A. (2009). Uma metaheristica híbrida com busca por agrupamentos aplicada a problemas de otimização combinatoria,. Master's thesis, Instituto Nacional de Pesquisa Espaciais.

Chaves, A. A. e Lorena, L. A. N. (2009). Clustering search algorithm for the capacitated centred clustering problem. Technical report, LAC, INPE.

Chaves, A. A. e Lorena, L. A. N. (2011). Hybrid evolunionary algorithm for the capacitated centered clustering problem. *Expert Systems with Applications*, 5:5013–5018.

Cole, R. M. (1998). Clustering with genetic algorithms. *Dissertação de mestrado, Departamento de Ciencia da Computação, Universidade de Western Australia*.

Feo, T. A. e Resende, M. G. C. (1995). Greedy randomized adaptative search procedures. *J. of Global Optimization*, 6:109–133.

Forgy, E. W. (1965). Cluster analysis of multivariate data: efficiency versus interpretability of classifications. *Biometrics*, 3:768.

Lourenco, H. R.; Martin, O. e Stutzle, T. (2003). *Iterated Local Search*. Handbook of Metaheuristics.

Maravilha, A. L. (2014). Algoritmo baseado em evolução diferencial para solução de problemas de otimização combinatória. Master's thesis, Universidade Federal de Minas Gerais, Programa de Pós Graduação em Engenharia Elétrica.

Mladenovic, N. (1995). A variable neighborhood algorithm - a new metaheuristics for optimization combinatorial. *Abstracts of Papers at Optimization Days*.

Montgomery, D. C. e Runger, G. C. (2003). *Applied Statistics and Probability for Engineers*. J. W. Sons.

Muritiba, A. E. F.; Negreiros, M.; Oriá, H. L. G. e Souza, M. F. (2012). A tabu search algorithm for the capacitated centred clustering problem. *Congresso Latino-Iberoamericano de Investigación Operativa*, 1.

Negreiros, M. e Palhano, A. (2006). The capacitated centred clustering problem. *Computers and Operations Research*, 33:1639–1663.

Oliveira, A. C. M.; Chaves, A. A. e Lorena, L. A. N. (2013). Clustering search. *Pesquisa Operacional*, 33:105–121.

Oria, H. L. G.; Negreiros, M. e Muritiba, A. E. F. (2012). Um algoritmo path reliking com busca tabu para o problema de agrupamento centrada capacitado em centro geométrico. *Claio SBPO, Simpósio de Pesquisa Operacional, Rio de Janeiro*.

Stefanello, F. e Muller, F. (2009). Um estudo sobre o problema de agrupamento capacitado. Technical report, Universidade Federal de Santa Maria.

Storn, R. e Price, K. (1995). Diferential evolution - a simple e efficient adaptative schema global optimzation over continuos spaces. *Technical Report TR-95-012, ICSI*.