Dissertação de Mestrado

Algoritmo BRKGA multipopulacional (MultiPop-BRKGA-PCA) para o problema de clusterização automática

Alexandre Sérgio Dantas de Lima
asdl@ic.ufal.br

Orientador:

Rian Gabriel Santos Pinheiro
Bruno Costa e Silva Nogueira

Maceió, março de 2021

Alexandre Sérgio Dantas de Lima

Algoritmo BRKGA multipopulacional (MultiPop-BRKGA-PCA) para o problema de clusterização automática

Dissertação apresentada como requisito parcial para obtenção do grau de Mestre pelo Curso de Mestrado em Informática do Instituto de Computação da Universidade Federal de Alagoas.

Orientadores:

Rian Gabriel Santos Pinheiro

Bruno Costa e Silva Nogueira

Maceió, março de 2021

Catalogação na fonte
Universidade Federal de Alagoas
Biblioteca Central
Divisão de Tratamento Técnico

Bibliotecário: Marcelino de Carvalho Freitas Neto – CRB-4 - 1767

L732a Lima, Alexandre Sérgio Dantas de.
Algoritmo BRKGA multipopulacional (MultiPop-BRKGA-PCA) para o problema de clusterização automática / Alexandre Sérgio Dantas de Lima. – 2021.
50 f. : il.

Orientador: Rian Gabriel Santos Pinheiro.
Co-orientador: Bruno Costa e Silva Nogueira.
Dissertação (mestrado em Informática) - Universidade Federal de Alagoas. Instituto de Computação. Maceió, 2021.

Bibliografia: f. 45-50.

1. Clusterização. 2. *Biased Random Key Genetic Algorithm*. 3. Algoritmos genéticos . I. Título.

CDU: 004.421

Resumo

O processo de agrupamento de dados é conhecido como clusterização. Na literatura, o processo de clusterização, ou agrupamento, tem duas variações: (i) se o número de clusters for predefinido, esse processo é conhecido como Problema de Clusterização (CP — Clustering Problem) ou Problema de k-Clusterização, e (ii) quando o número de clusters não é definido, o processo torna-se conhecido como Problema de Clusterização Automática (ACP — Automatic Clustering Problem). A importância de ter dados bem agrupados é crucial para a tomada decisões mais assertivas. A técnica de agrupamento de dados possui aplicabilidade nas mais diversas áreas do conhecimento, como: engenharia, administração, economia, biologia, física, entre outras.

Os algoritmos genéticos são baseados em processos de evolução darwinistas, selecionando as melhores soluções dentro de uma população. O BRKGA (Biased Random Key Genetic Algorithm), é apresentado como uma variação dos algoritmos genéticos, em que as soluções de um problema são representadas como vetores de chaves reais, geradas aleatoriamente, no intervalo contínuo de [0,1). O fitness de uma solução viável é determinado pelo decodificador, que mapeia este vetor em um valor real.

Este trabalho propõe um BRKGA multipopulacional para identificar o número ideal de clusters, de acordo com o índice de silhueta — uma medida de eficácia de agrupamento bastante utilizada na literatura. No algoritmo proposto, o espaço de soluções é particionado de forma que cada subpopulação o algoritmo represente soluções com um número k de cluster. Assim, a cada subpopulação, um BRKGA independente é aplicado. Experimentos computacionais foram realizados em cinquenta e cinco instâncias da literatura.

As instâncias utilizadas neste trabalho possuem dimensões que variam de 30 a 2.000 objetos e 2 a 13 atributos, com diferentes graus de dificuldade e características, por exemplo, coesão de grupo, separação, formatos e densidades.

O algoritmo proposto é comparado com métodos existentes na literatura, apresentando resultados superiores, levando ao entendimento de que o algoritmo é promissor.

Palavras-chave: Algoritmo Genético, Multipopulacional, BRKGA, Clusterização

Abstract

The process of grouping data is known as clustering. In the literature, the clustering process, or grouping, has two variations: (i) if the number of clusters is predefined, this process is known as the Clustering Problem (CP) or k -Clustering Problem , and (ii) when the number of clusters is not defined, the process becomes known as Automatic Clustering Problem (ACP). The importance of having well-grouped data is crucial to making more assertive decisions. The data grouping technique has applicability in the most diverse areas of knowledge, such as: engineering, administration, economics, biology, physics, among others.

Genetic algorithms are based on Darwinian evolution processes, selecting the best solutions within a population. The BRKGA (Bised Random Key Genetic Algorithm), is presented as a variation of the genetic algorithms, in which the solutions of a problem are represented as vectors of real keys, generated randomly, in the continuous interval of $[0.1)$. The suitability of a viable solution is determined by the decoder that maps this vector to a real value.

This work proposes a multi-population BRKGA to identify the ideal number of clusters, according to the silhouette index — a measure of clustering efficiency widely used in the literature. In the proposed algorithm, the solution space is partitioned so that each subpopulation the algorithm represents solutions with a k number of cluster. Thus, for each subpopulation, an independent BRKGA is applied. Computational experiments were carried out in fifty-five instances of the literature.

The instances used in this work have dimensions ranging from 30 to 2,000 objects and 2 to 13 attributes, with different degrees of difficulty and characteristics, for example, group cohesion, separation, shapes and densities.

The proposed algorithm is compared with existing methods in the literature, presenting superior results, leading to the understanding that the algorithm is promising.

Keyword: Genetic Algorithm, Multi-population, BRKGA, Clustering

Agradecimentos

Agradeço a Deus por me guiar e dar forças para seguir sempre. A minha família, base que sustenta e motiva.

Aos meus orientadores Rian Pinheiro e Bruno Nogueira, que sempre estiveram disponíveis para contribuir com seus conhecimentos.

Alexandre Lima

Conteúdo

Lista de Figuras

Lista de Tabelas

List of Algorithms

Capítulo 1

Introdução

A análise de agrupamentos, ou clusterização, vem sendo amplamente utilizada nas mais diversas áreas do conhecimento (Kaufman & Roussew, 1990) como, por exemplo, na Medicina, Inteligência Artificial, Biologia, Estatística, Economia, Sociologia, praticamente todas as áreas do conhecimento. O processo de clusterização é muito importante, principalmente com a ascensão da analise de dados para tomada de decisão. Uma organização consegue melhores resultados através da análise de dados, porém é necessário que esse grande volume de dados esteja bem organizado. O processo de clusterização se faz presente para atender esta necessidade.

É uma técnica de análise multivariada que agrega um conjunto de métodos que têm, por objetivo, agrupar os objetos (registros) de uma base dados em grupos, de forma que estes grupos possuam, internamente, alto grau de homogeneidade. Esta homogeneidade, por sua vez, é função das d variáveis (características ou atributos) associadas aos n objetos, bem como da métrica (distância) e da função objetivo adotada. A função objetivo, geralmente, depende da distância entre os objetos.

Em geral, ao definir a função objetivo, fica definido, consequentemente, o problema de agrupamento (clusterização) a ser abordado. Ainda neste sentido, em Hansen & Jaumard (1997) e Cruz (2010) são apresentadas várias funções objetivo adotadas em problemas de agrupamento (clusterização). Doravante, para referenciar o agrupamento de dados, a palavra clusterização será adotada com maior frequência. É importante entender que, independentemente da métrica e da função objetivo consideradas, os problemas de clusterização são, em geral, de difícil solução computacional (Hansen & Jaumard, 1997) (Cruz, 2010). Mais especificamente, à medida que o número de objetos da base de dados aumenta, mais difícil torna-se a obtenção do ótimo global para esses problemas, seja através de métodos exatos ou através de algoritmo heurísticos.

A ideia básica da clusterização que componham um mesmo cluster, ou seja um mesmo grupo, devem apresentar alta similaridade (*i.e.*, sejam elementos bem parecidos, seguem um padrão similar). Além disso, idealmente estes elementos não devem possuir similaridade

com elementos de outros clusters (grupos) (Cruz, 2010). Em outras palavras, toda clusterização é feita com objetivo de maximizar a homogeneidade dentro de cada cluster e maximizar a heterogeneidade entre clusters.

A grande vantagem do uso das técnicas de Clusterização é que, ao agrupar dados similares, pode-se descrever de forma mais eficiente e eficaz as características peculiares de cada um dos grupos identificados (Cruz, 2010). Isso fornece um maior entendimento do conjunto de dados original, além de possibilitar o desenvolvimento de esquemas de classificação para novos dados e descobrir correlações interessantes entre os atributos dos dados que não seriam facilmente visualizadas sem o emprego de tais técnicas. Alternativamente, clusterização pode ser usada como uma etapa de pré-processamento para outros algoritmos, tais como caracterização e classificação, que trabalhariam nos clusters identificados.

No enfoque do Problema de Clusterização Automática (PCA), uma vez que o número de k clusters não é previamente conhecido, algumas medidas são tomadas para tentar encontrar esse valor. Existem funções que podem ser utilizadas para determinar k, entre elas está a função Índice Silhueta, definida por Kaufman & Roussew (1990). A ideia ainda é tentar encontrar o número de k clusters e ainda reduzir o máximo possível esse número, tal que: dada uma instância com n elementos de dimensões d, separar por similaridade, formar os grupos de cada dado e tentar unir os dados que apresentem maior aproximação característica.

1.1 Justificativa

Segundo Humby (2006), dados são extremamente valiosos, em alusão ao petróleo, Humby (2006) compara o valor dos dados ao valor do petróleo e, afirma que os dados são mais valiosos, isso porque os dados são infinitos em sua essência fomentada de modo exponencial a cada segundo, já o petróleo é finito; como afirma o matemático londrino. Diante da magnitude presenciada, comum que existam novos desafios para que este bem imensurável seja compreendido e tão bem tratado quanto outros bens.

O ponto de interrogação se cria a partir da hipótese de que é necessário agrupar esses dados para que se saiba de fato o que representam. Um exemplo claro estão nos dados que as redes sociais acumulam. Segundo a revista CIO, cerca de 2,5 quintilhões de *bytes* são criados todos os dias, em geral. Diante do contexto da proeminência dos dados, métodos de clusterização automática rápido e preciso são de grande importância.

1.2 Objetivos

O objetivo deste trabalho é buscar desenvolver um novo algoritmo de clusterização automática, baseado na meta-heurística BRKGA (*Biased Random Key Genetic Algorithm* — Algoritmo Genético de Chaves Aleatórias) considerando uma estratégia multipopulacional.

Faz parte dos objetivos comparar o algoritmo proposto com algoritmos existentes, considerando as bases de dados mais utilizadas na literatura. Além do algoritmo proposto evoluir a cada população independente, foram testadas várias estratégias para selecionar a próxima população a ser evoluída.

Destaca-se que os resultados deste trabalho foram publicados nos anais da conferência SMC 2021, também incluído no IEEE Xplore, o que agrega de forma significativa todo o esforço deste trabalho.

1.3 Estrutura do trabalho

O restante deste trabalho está organizado da seguinte forma: No Capítulo 2 a Fundamentação Teórica; apresentando clusterização, as medidas de similaridade, uma breve explanação sobre as áreas em que são possíveis as aplicações envolvendo a clusterização e os métodos de otimização em um problema a partir da função objetivo e um conjunto de restrições; Ainda apresenta soluções como meta-heurísticas. O Capítulo 3 traz o levantamento bibliográfico realizado e os trabalhos em que foram base para o estudo deste trabalho. O Capítulo 4 apresenta a proposta multipopulacional BRKGA para o Problema de Clusterização Automática. No Capítulo 5 os resultados obtidos através da proposta deste trabalho e, por fim o Capítulo 6 as conclusões sobre as soluções propostas através dos resultados obtidos.

Capítulo 2

Fundamentação Teórica

Agrupamento ou Clusterização faz parte do problema clássico que busca encontrar o número ideal de agrupamentos em dados classificados da mesma categoria. Durante o desenvolvimento deste trabalho o termo utilizado será clusterização para tratar de modo genérico a formação de grupos ou agrupamentos.

2.1 Clusterização

A definição de um problema de clusterização dar-se por algumas etapas, conforme Hansen & Jaumard (1997), tais etapas são listadas abaixo:

1. **Amostragem** — Selecione um conjunto de m objetos
   X = $\{x_1, x_2, x_3 ... x_m\}$ onde estão os clusters;

2. **Dados** — observe ou tire a medida das p características de cada objeto $x_i \in X$. Isto conduz a uma matriz de dados $M_{m \times p}$;

3. **Similaridade** — calcule a partir de $M_{m \times p}$, a matriz de similaridades $D_{m \times m} = d_{k,l}$ entre os objetos de X. Estas similaridades devem satisfazer as propriedades $d_{k,l} > 0$, $d_{k,k} = 0$, $d_{k,l} = d_{l,k}$ onde $k, l = 1, 2, \ldots m$. As similaridades não precisam ser necessariamente distâncias.

4. **Restrições** — especifique o tipo de problema desejado (Subconjunto , Partição, Cobertura, etc. os tipos de problemas serão definidos posteriormente);

5. **Critério** — escolha o critério (ou possivelmente mais de um critério) para expressar a homogeneidade e/ou separação dos *clusters* no problema a ser tratado;

6. **Algoritmo** — defina um algoritmo para o problema. Codifique o algoritmo;

7. **Computação** — aplique o algoritmo escolhido na matriz $D_{m \times m} = d_{k,l}$ obtendo assim os *clusters*;

8. **Interpretação** — aplique testes formais ou informais para selecionar os melhores *clusters*. Interprete os resultados;

Generalizando a definição do Problema de Clusterização se tem a seguinte informação: Dado um conjunto de n elementos $X = \{X_1, X_2, \ldots, X_n\}$, o problema de clusterização consiste na obtenção de um conjunto de k *clusters*, $C = \{C_1, C_2, \ldots, C_k\}$, tal que os elementos contidos em um *cluster* C_i possuam uma maior similaridade entre si do que com os elementos de qualquer um dos demais *clusters* do conjunto C. O conjunto C é considerado uma clusterização com k *clusters* caso as seguintes condições sejam satisfeitas:

$$\bigcup_{i=1}^{k} C_i = X$$

$$C_i \neq \emptyset, \quad \text{para } 1 \leq i \leq k$$

$$C_i \cup C_j = \emptyset, \quad \text{para } 1 \leq ij \leq k \text{ e } i \neq j.$$

O número de *clusters* k pode variar entre 2 e $\lfloor \sqrt{n} \rfloor$.

2.1.1 Clusterização com k fixo

No caso do valor de k ser definido como parâmetro para a solução, o problema é referenciado na literatura como problema de k-clusterização.

No processo em uma k-clusterização, o quantitativo de diferentes formas de agrupamento de n elementos de um conjunto em k *clusters*, corresponde à função $N(n,k)$ exibida na equação abaixo:

$$N(n,k) = \frac{1}{k!} \sum_{i=0}^{k} (-1)^i \binom{k}{i} (k-i)^n \quad .$$

Tendo o objetivo de ilustrar o crescimento exponencial da quantidade numérica de soluções possíveis para um problema de k-clusterização, considerando a equação acima, para combinar 10 elementos em 2 *clusters*, 100 elementos em 2 *clusters*, 100 elementos em 5 *clusters* e 1000 elementos em 2 *clusters*, temos respectivamente os seguintes números de soluções possíveis: $N(10,2) = 511, N(100,2) = 6,33825 \times 10^{29}$, $N(100,5) = 6,57384 \times 10^{67}$ e $N(1000,2) = 5.3575 \times 10^{300}$.

2.1.2 Clusterização automática

Se o valor de k não é definido, o problema é classificado como "problema de clusterização automática (PCA)" e o alcance do valor de k faz parte do processo de solução do problema.

Para o problema de clusterização automática o número total de combinações sofre uns incrementos significativos, sendo definido de acordo com a equação:

$$N(n) = \sum_{k=1}^{n} \frac{1}{k!} \sum_{i=0}^{k} (-1)^i \binom{k}{i} (k-i)^n$$

Então, para um conjunto com 10 elementos, a clusterização automática tem que considerar 115.975 diferentes tipos de combinação dos elementos em um número de *clusters* que podem variar de 1 a 10.

Deve ser considerado também um outro aspecto, em relação ao problema de clusterização, que é medir o quanto um elemento é similar a outro, para a identificação dos objetos, se devem estar contidos em um mesmo cluster ou não. Este processo se faz a partir da medida de similaridade.

2.2 Medidas de similaridade

Um ponto muito importante a ser considerado durante o processo, é mensurar o quanto um dado é similar a outro, uma vez que identificado os dados ficarão contidos em um mesmo grupo, caso sejam similares entre si. A utilização de uma medida de similaridade é necessária para cada problema de clusterização a ser tratado. A distância entre os elementos é um ponto importante para sua identificação, pois trabalham com as diferenças entre os valores de cada atributo dos elementos. Logo, quanto menor a distância entre um conjunto de elementos, maior será a sua similaridade entre eles. A seguir medidas de distâncias muito utilizadas:

- **distância euclidiana:** Trata-se da distância mais comum entre dois pontos (X_j). Aquela distância medida com uma régua. Dado dois vetores X e Y, a mesma é definida como sendo considera a distância d entre dois elementos X_i e X_j no espaço p-dimensional:

  $$d(X_i, X_j) = \left[\sum_{l=1}^{p} (x_{i_l} - x_{j_l})^2 \right]^{\frac{1}{2}}$$

  Trata-se de uma distância que é invariante a rotação do sistema de coordenadas a sua reflexão em torno de um eixo, translações.

- **distância "*city-block*":** corresponde à soma das diferenças entre todos os p atributos de dois elementos X_i e X_j, não sendo indicada para os casos em que existe uma correlação entre tais atributos. Dado dois vetores X_i e Y_j, esta métrica é definida como o somatória dos módulos das diferenças, e possui a seguinte fórmula:

  $$d(X_i, X_j) = \sum_{l=1}^{p} \left| x_{i_l} - x_{j_l} \right|$$

Trata-se de uma distância que depende da rotação do sistema de coordenadas, mas não depende de sua reflexão em torno de um eixo ou suas translações.

- **distância de Chebyshev:** é uma métrica definida em um espaço de vetores onde a distância entre dois vetores é a maior de suas diferenças entre suas dimensões de coordenadas. A distância de Chebyshev se assemelha muito a city block. No caso, essa métrica considera o valor máximo dos módulos das diferenças dos pontos em respectivas posições. Assim, dado dois vetores X e Y, a mesma é definida como sendo:

$$d(X,Y) = \max(|X_1 - Y_1|, |X_2 - Y_2|, ..., |X_p - Y_p|)$$

- **distância de Mahalanobis:** É baseada nas correlações entre variáveis com os quais distintos padrões podem ser identificados e analisados. É uma estatística útil para determinar a similaridade entre uma amostra desconhecida e uma conhecida. Distingue-se da distância euclidiana já que tem em conta as correlações do conjunto de dados e é invariante à escala, ou seja, não depende da escala das medições. A distância Mahalanobis é definida como:

$$d(X,Y) = [(X-Y)^T \Sigma^{-1} (X-Y)]^{\frac{1}{2}}$$

em que Σ a matriz de covariância do conjunto de dados.

A distância de Mahalanobis leva em consideração a variância de cada atributo, assim como a covariância entre eles. Transforma os dados em dados normalizados não correlacionadas e calcula a distância euclidiana para os dados transformados. É invariante à escala (não depende da escala das medições).

- **distância Quadrática:** Basicamente, a distância quadrática é uma generalização da distância de Mahalanobis. Também leva em consideração a relação entre os seguintes atributos:

$$d(X,Y) = [(X-Y)^T A (X-Y)]^{\frac{1}{2}}$$

No entanto, no lugar da matriz de covariâncias, ela utiliza uma matriz A. A deve ser simétrica positiva definida. Isso significa que A é uma matriz válida $d(X,Y) \geq 0$

De modo geral, a matriz A deverá ser calculada de acordo com o problema. Por exemplo, na distância de Mahalanobis: A é a matriz inversa da matriz de covariâncias distância Euclidiana: A é a matriz identidade.

- **coeficiente DUNN** Compreende-se a distância mínima entre pontos de diferentes *clusters* por d_{min} e a maior distância dentro do *cluster* por d_{max}. A distância entre

os *clusters* C_k e $C_{k'}$ é medida pela distância entre seus pontos mais próximos:

$$d_{kk'} = \min_{\substack{p_i \in C_k \\ p_j \in C_{k'}}} \|p_i - p_j\|$$

E a menor dessas distâncias $d_{kk'}$:

$$d_{min} = \min_{k \neq k'} d_{kk'}.$$

Para cada *cluster* C_k, compreende-se por D_k a maior distância que separa dois pontos distintos no *cluster*:

$$D_k = \max_{\substack{p_i, p_j \in C_k \\ p_i \neq p_i}} \|p_i - p_j\|.$$

Dessa forma, d_{max} é a maior dessas distâncias D_k:

$$d_{max} = \max_{1 \leq k \leq |K|} D_k.$$

Por fim, o índice de Dunn é definido como o quociente de d_{min} e d_{max}:

$$Dunn = \frac{d_{min}}{d_{max}}.$$

- Índice Silhueta

O Índice Silhueta foi proposto por Rousseeuw (1987) e é capaz de determinar a qualidade das soluções com base na proximidade entre os objetos de determinado grupo e na distância desses objetos ao seu grupo mais próximo. O índice silhueta é calculado para cada objeto, sendo possível identificar se o objeto está alocado ao grupo mais adequado. Esse índice combina as ideias de coesão e de separação. Os passos abaixo mostram uma breve explicação de como calcular:

1. O d_{ij} corresponde à distância euclidiana entre os objetos i e j, e d é a quantidade de atributos dos objetos. Para cada objeto i_x calcula-se a sua distância média $a(x_i)$ em relação a todos os demais objetos do mesmo grupo. Na equação abaixo, $|C_w|$ representa a quantidade de objetos do grupo C_w , ao qual o objeto x_i pertence.

$$d_{ij} = \sqrt{\sum_{x=1}^{d} (i_x - j_x)^2}$$

$$a(x_i) = \frac{1}{|C_w| - 1} \sum d_{ij} \qquad \forall x_j \neq x_i, x_j \in C_w$$

2. A próxima equação apresenta a distância entre o objeto x_i e os objetos do grupo C_t, em que $|C_t|$ é a quantidade de objetos do grupo C_t. Para cada objeto x_i calcula-se a sua distância média em relação a todos os objetos dos demais grupos $(b(x_i))$.

$$d(x_i, C_t) = \frac{1}{|C_t|} \sum d_{ij} \qquad \forall x_j \in C_t$$

$$b(x_i) = \min d(x_i, C_t)$$

$$C_t \neq C_w \qquad C_t \in C$$

3. O coeficiente silhueta do objeto $x_i (s(x_i))$ pode ser obtido pela equação abaixo.

$$s(x_i) = \frac{b(x_i) - a(x_i)}{\max\{b(x_i), a(x_i)\}}$$

4. O cálculo da silhueta de uma solução S é a média das silhuetas de cada objeto, conforme apresenta a equação abaixo, em que n é a quantidade de objetos da solução. Essa função deve ser maximizada.

$$Silhueta(S) = \frac{1}{n} \sum_{i=1}^{n} s(x_i)$$

Os valores positivos de silhueta indicam que o objeto está bem localizado em seu grupo, enquanto valores negativos indicam que o objeto está mais próximo de outro(s) grupo(s).

Este índice é mais apropriado para agrupamentos volumétricos, com grupos gerados de acordo com distribuições Gaussianas multidimensionais hiperesféricas ou moderadamente alongadas, porém ele não obteve bons resultados para grupos com formatos arbitrários (Rousseeuw, 1987).

Em Hruschka & Covoes (2005) é proposta uma versão simplificada do índice de silhueta. Nesta versão são efetuadas modificações nos cálculos de $a(x_i)$ e $b(x_i)$ com o objetivo de reduzir a complexidade do algoritmo de $O(n^2)$ para $O(n)$ os autores desse

trabalho, mesmo com a redução da complexidade, esse novo índice mantém a qualidade próxima ao da silhueta tradicional.

As soluções propostas para a clusterização automática têm como base o índice silhueta visto a precisão deste método proposto por Rousseeuw (1987), calculando as similaridades entre os objetos pertencentes a um grupo e a dissimilaridade entre os demais grupos.

Há problemas que não convém utilizar a distância, ou não é possível encontrar a mesma, como medida de similaridade, uma vez que os valores dos atributos não são escalares. Imagina-se, como exemplo, um problema de clusterização que contenha como atributos os seguintes campos: sexo e endereço; para tratar este problema serão necessários outros atributos que possam evidenciar algum tipo de similaridade entre os elementos.

2.3 Aplicações

Nas mais diversas áreas são encontradas as aplicações envolvendo a clusterização. As especificações de cada área pode ser distinta. Algumas das principais aplicações para a clusterização são apresentadas a seguir:

A área de pesquisa que tem por objetivo a classificação de objetos (padrões), é também conhecida como reconhecimento de padrões, em um número de categorias ou classes (clusters) (Duda et al., 1973). Por exemplo, no reconhecimento de faces, as imagens das faces são objetos e as classes são seus nomes ou identificações. Comumente as aplicações de reconhecimento de padrões têm características disponíveis nos padrões de entrada, tipicamente são milhares, não são diretamente utilizadas. Geralmente as características são extraídas dos padrões de entrada otimizados, através de procedimentos orientados por dados. A partir de um padrão, o reconhecimento, classificação consiste em uma das seguintes tarefas: *classificação não supervisionada*, pois o padrão é associado a uma classe que é aprendida com base na similaridade entre os padrões de treinamento; e *classificação supervisionada*, tendo como padrão de entrada identificado como um membro de uma classe pré-definida pelos padrões de treinamento, que são rotulados com suas classes.

Listar todos os trabalhos que utilizam técnicas de clusterização, certamente é exaustivo e não seria possível catalogar todos em um só trabalho. Segmentação de imagem, é um problema importante na ótica computacional, pode ser formulada como um problema de clusterização (Shi & Malik, 2000). Documentos podem ser clusterizados (Iwayama & Tokunaga, 1995) para gerar hierarquias de tópicos para acesso eficiente à informação ou recuperação

(Wang et al., 2007). Clusterização é utilizado também para diferentes grupos de analise clientes para estratégias de marketing (Reutterer & Dan, 2020), bem como o estudo de genoma na biologia (Baldi & Hatfield, 2011).

Os exemplos para áreas de interesse no problema de clusterização são inúmeras (Ochi et al., 2004), para entendimento dessa magnitude é possível citar algumas áreas, como: engenharia, aprendizagem de máquina (Yonamine et al., 2002), mineração de dados (Ochi et al., 2004), medicina (Hartuv & Shamir, 2000), marketing (Punj & Stewart, 1983), administração (Roses & Leis, 2002), biologia (Fontana, 2017) e gestão de negócios. Aplicações relativas a reconhecimento de padrões (Coutinho et al., 2015), processamentos de imagens (Ray & Turi, 1999), análise de dados, pesquisa de mercado, química de petróleo (Martinelli & Eidsvik, 2014), análise de sintomas de doenças (Kim et al., 2005), especificações físicas, padrão de compra (Cho et al., 2014), características de seres vivos, aspectos da personalidade de indivíduos, perfis de clientes, marketing estratégico (Liu & Ong, 2008), composição solos, funcionalidade de genes, segmentação e padrão de imagens, tecnologia da informação, estudo de dados de genoma na biologia, gestão de força de trabalho, planejamento estratégico e organizacional. Em síntese, aplicações de clusterização são estão presentes pelo menos em um dos seguintes objetivos principais:

- Compressão: como um método para a organização dos dados e resumido-o através de protótipo do *cluster*;
- Identificação da estrutura subjacente: para obter *insights*, para tomada de decisões, através dos dados, identificando padrões de grande relevância;
- Classificação Natural: Atribuir e identificar a classificação, grau de similaridade entre os objetos.

2.4 Otimização

Em um problema de otimização há uma função objetivo e um conjunto de restrições, ambos relacionados às variáveis de decisão. Os valores possíveis às variáveis de decisão são delimitados pelas restrições impostas sobre essas variáveis, formando um conjunto (contínuo ou discreto) de soluções factíveis a um problema. O problema pode ser de minimização ou de maximização da função objetivo. A resposta para o problema de otimização, ou seja, o ótimo global, será o menor (ou maior) valor possível para a função objetivo para o qual o valor atribuído às variáveis não viole nenhuma restrição.

Em alguns casos, encontra-se valores cuja alteração discreta não conduz a resultados melhores, mas que não são também o ótimo global, a essas soluções o termo conhecido como ótimo local.

2.4.1 Meta-heurísticas e Heurísticas

Meta-heurísticas são métodos de solução que coordenam procedimentos de busca locais com estratégias de mais alto nível, de modo a criar um processo capaz de escapar de mínimos locais e realizar uma busca robusta no espaço de soluções de um problema (Osman & Kelly, 1997).

Logo, a definição passou a abranger quaisquer procedimentos que empregassem estratégias para escapar de mínimos locais em espaços de busca de soluções complexas. Em especial, foram incorporados procedimentos que utilizam o conceito de vizinhança para estabelecer meios de fugir dos mínimos locais (Osman & Kelly, 1997).

Uma meta-heurística, portanto, visa produzir um resultado satisfatório para um problema, porém sem qualquer garantia de otimalidade. Meta-heurísticas são empregadas para descobrir respostas a problemas sobre os quais há poucas informações: não se sabe como é a aparência de uma solução ótima, há pouca informação heurística disponível e força-bruta é ignorada devido ao espaço de solução ser muito grande. Contudo, dada uma solução candidata ao problema, esta pode ser testada e sua otimalidade, examinada.

2.4.2 Algoritmos Genéticos

Algoritmo Genético (GA) foi um dos primeiros algoritmos estocásticos baseados em população propostos na história. Os principais operadores do GA são a seleção, crossover e mutação.

O Algoritmo Genético foi inspirado na teoria da evolução darwiniana (Holland, 1992), na qual o sobrevivência da criatura mais apta e seus genes foram simulados. Cada solução corresponde a um cromossomo e cada parâmetro representa um gene. O Algoritmo Genético avalia a aptidão de cada indivíduo na população usando uma função de aptidão (objetivo). Para melhorar soluções ruins, as melhores soluções são escolhidas aleatoriamente com um mecanismo de seleção (por exemplo, roda de roleta). Este operador tem mais probabilidade de escolher as melhores soluções, uma vez que a probabilidade é proporcional a aptidão (valor objetivo). O que reduz a escolha do ótimo local é a probabilidade de escolher soluções ruins também. Isso significa que, se boas soluções ficarem presas em uma solução local, eles podem ser extraídos com outras soluções.

O Algoritmo Genético é estocástico, então pode-se perguntar o quão confiável ele é. O que faz este algoritmo confiável e capaz de estimar o ótimo global para um determinado problema é o processo de manter as melhores soluções em cada geração e usá-las para melhorar outras soluções (Bean, 1994). Como tal, toda a população passa a ser uma geração melhor

por geração. O cruzamento entre indivíduos resulta na exploração da área entre as duas soluções pai fornecidas. Este algoritmo também se beneficia da mutação. Este operador altera aleatoriamente os genes nos cromossomos, o que mantém a diversidade dos indivíduos na população e aumenta o comportamento exploratório do algoritmo genético. Semelhante à natureza, o operador de mutação pode resultar em uma melhor solução e conduza outras soluções para o ótimo global.

População Inicial

O Algoritmo Genético começa com uma população aleatória. Essa população pode ser gerada a partir de uma distribuição aleatória gaussiana para aumentar a diversidade. Esta população inclui várias soluções, que representam cromossomos de indivíduos. Cada cromossomo possui um conjunto de variáveis, que simula os genes. O principal objetivo da etapa de inicialização é espalhar as soluções em todo o espaço de pesquisa de forma tão uniforme quanto para a diversidade da população e ter uma melhor chance de encontrar regiões promissoras. As próximas seções discutem as etapas para melhorar os cromossomos na primeira população.

Seleção

A seleção natural é a principal inspiração deste componente para o algoritmo GA. Na natureza, os indivíduos mais aptos têm uma chance maior de obter comida e acasalar. Isso faz com que seus genes contribuam mais na produção da próxima geração da mesma espécie. Inspirando-se nessa ideia simples, o algoritmo GA emprega um roda da roleta para atribuir probabilidades a indivíduos e selecioná-los para criar o próxima geração proporcional aos seus valores de aptidão (objetivo). Figura 2.1 ilustra um exemplo de roleta para seis indivíduos. Os detalhes destes indivíduos são apresentados na Tabela 2.1.

Pode ser visto que o melhor indivíduo (5) tem a maior participação do gráfico, enquanto o pior indivíduo (4) tem a menor participação. Este mecanismo simula a seleção natural do indivíduo mais apto da natureza. Uma vez que uma roda de roleta é um operador estocástico, os indivíduos menos aptos têm uma pequena probabilidade de participar da criação da próxima geração. Se uma solução ruim tem probabilidade de ser selecionada, existe a chance destes indivíduos participarem das próximas gerações. O descarte de tais soluções reduzirá a diversidade da população e deve ser evitado.

Crossover

Depois de selecionar os indivíduos usando um operador de seleção, eles devem ser empregados para criar a nova geração. Na natureza, os cromossomos nos genes de um macho e de uma fêmea são combinadas para produzir um novo cromossomo. Isso é simulado combinando duas soluções (soluções pai) selecionadas pela roleta para produzir duas novas

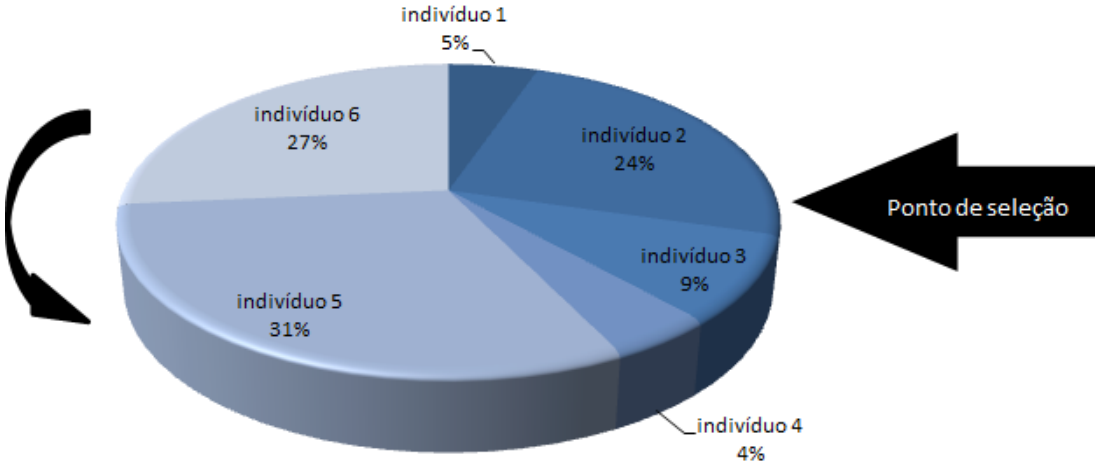


Figura 2.1: Mecanismo da roleta no Algoritmo Genético. O melhor indivíduo (5) tem a maior participação da roleta, enquanto o pior indivíduo (4) tem a menor participação.

Tabela 2.1: Detalhes dos indivíduos apresentados na Figura 2.1. O indivíduo mais apto é o Indivíduo 5.

Indivíduo	Valor Fitness	% Total
1	12	5
2	55	24
3	20	8
4	10	4
5	60	33
6	70	26
Total	227	100

soluções (crianças) no algoritmo genético. Existem diferentes técnicas para o operador de crossover na literatura, dos quais dois; ponto único e ponto duplo (Srinivas & Patnaik, 1994), são mostrados na Figura 2.1

Na Figura 2.2, duas técnicas populares de *crossover* em no algoritmo genético: ponto único e ponto duplo. No cruzamento de um único ponto, os cromossomos de duas soluções pai são trocados antes e depois de um único ponto. No cruzamento de ponto duplo, existem dois pontos de cruzamento e os cromossomos entre os pontos são trocados apenas.

No cruzamento de um único ponto, os cromossomos de duas soluções pai são trocado antes e depois de um único ponto. No cruzamento de dois pontos, no entanto, há são dois pontos cruzados e os cromossomos entre os pontos são trocados apena, como mostra a Figura 2.3.

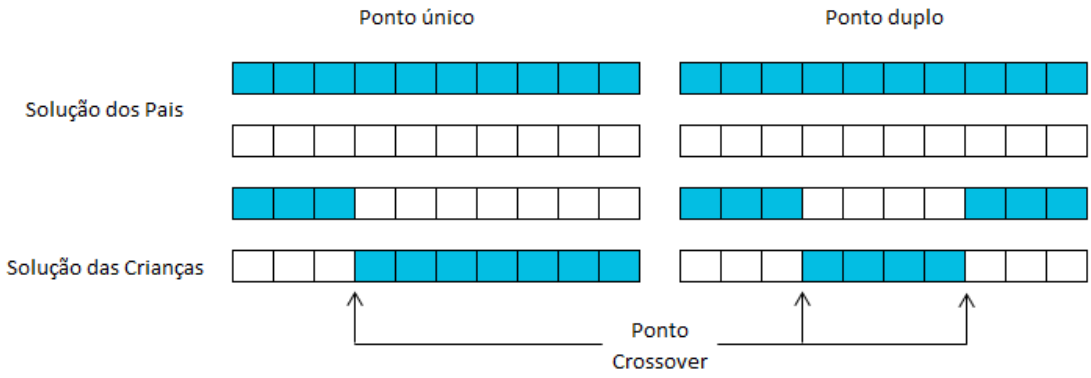


Figura 2.2: O melhor indivíduo (5) tem a maior parcela de da roleta, enquanto o pior indivíduo (4) tem a menor participação.

Mutação

O último operador evolutivo, no qual um ou vários genes são alterados após ter criado novas soluções. A taxa de mutação é definida como baixa em no algoritmo genético porque as alta taxas de mutação convertem o algoritmo em uma pesquisa aleatória primitiva. O operador de mutação mantém a diversidade da população, introduzindo outro nível de aleatoriedade. No fato, este operador impede que as soluções se tornem semelhantes e aumentam a probabilidade de evitar soluções locais no algoritmo genético. Um exemplo conceitual deste operador.

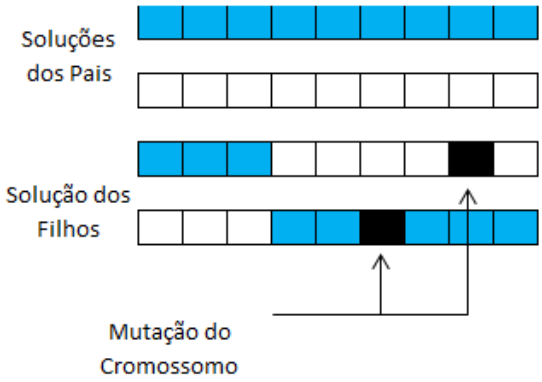


Figura 2.3: Operador de mutação altera um ou vários genes nas soluções geradas depois da fase de crossover.

Pode-se ver nesta Figura 2.3 que pequenas mudanças em alguns dos genes selecionados aleatoriamente ocorrem após a fase de crossover (cruzamento).

O algoritmo genético começa com uma população aleatória de indivíduos. Até o fim do critério final, este algoritmo melhora a população. A melhor solução entre todas as po-

pulações é retornada como a melhor aproximação do ótimo global para um determinado problema. A taxa de seleção, crossover (cruzamento), e a mutação pode ser alterada ou configurada para fixar números durante a otimização (Ahn & Ramakrishna, 2003).

2.4.3 RKGA

O RKGA foi proposto pela primeira vez por Bean (1994). Bean (1994) descreve uma nova classe de algoritmos genéticos para problemas e otimização combinatória, através de vetores de permutação, essas soluções podem ser representadas. Esse algoritmos, chamados de algoritmos genéticos de chaves aleatórias (ou RKGA, do inglês random-key genetic algorithms), representam uma solução do problema de otimização com um vetor de chaves aleatórias. Uma chave aleatória é um número real, gerado aleatoriamente, no intervalo contínuo $[0, 1)$. Um decodificador é um procedimento que mapeia um vetor de chaves aleatórias numa solução do problema de otimização e calcula o custo desta solução. O decodificador de Bean (1994) simplesmente ordena os elementos do vetor de chaves e com isso gera uma permutação que corresponde aos índices dos elementos ordenados (Resende, 2011).

Usando esta representação, soluções inviáveis poderiam ser evitadas ao realizar operações de evolução. A estrutura do RKGA é descrita como segue a representação:

1. inicialização: N indivíduos da população inicial são gerados aleatoriamente;
2. Calcule a aptidão de cada indivíduo na população;
3. Escolha os pais aleatoriamente e, usando a seleção do torneio, execute o cruzamento e operações de mutação nesses pais e, em seguida, atualize;
4. Se o número de soluções decodificadas atingir o valor máximo, a pesquisa será encerrada; caso contrário, vá para a Etapa 2.

Um algoritmo genético de chaves aleatórias (RKGA), evolui uma população, com n vetores de chaves, a partir do princípio darwinista, compreendendo que os indivíduos mais fortes de uma determinada população têm maiores probabilidades de encontrar um indivíduo melhor e assim manter a sua espécie com seu material genético. Como descrito, o algoritmo inicia a população de vetores de chaves aleatórias, logo produzindo uma série de populações. Os vetores da população são particionados em um conjunto pequeno, isso na k-ésima geração, esse pequeno conjunto de $P_e < p/2$ vetores que correspondem às melhores soluções (este conjunto é chamado de elite), o restante da população forma um outro conjunto (chamado de não elite), visto na Figura 2.4. Para evoluir a população, uma nova geração de indivíduos deve ser produzida. Todos os indivíduos de elite da população da geração k são copiados sem modificação para a população da geração $k+1$, visto na Figura 2.5. A mutação, tanto nos algoritmos genéticos quanto na biologia, é a chave para a evolução da população. RKGAs implementam a mutação introduzindo mutantes na população.

Um mutante é simplesmente um vetor de chaves aleatórias geradas da mesma forma que um elemento da população inicial. A cada geração, um pequeno número (P_m) de mutantes é introduzido na população, Figura 2.5. Com os indivíduos P_e elite e os mutantes P_m contabilizados na população $k+1$, $p-P_e-P_m$ indivíduos adicionais precisam ser produzidos para completar os p indivíduos que constituem a nova população. Isso é feito através da produção de progênie p - P_e - P_m através do processo de cruzamento, como mostra a Figura 2.6.

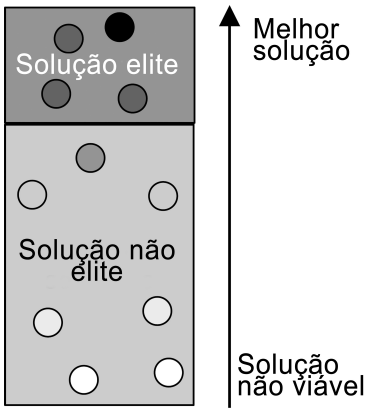


Figura 2.4: A população de soluções p é particionada em um conjunto menor de soluções Pe elite (mais adequadas) e um conjunto maior de soluções $p-Pe$ não elite (menos adequadas)- (Gonçalves et al., 2014).

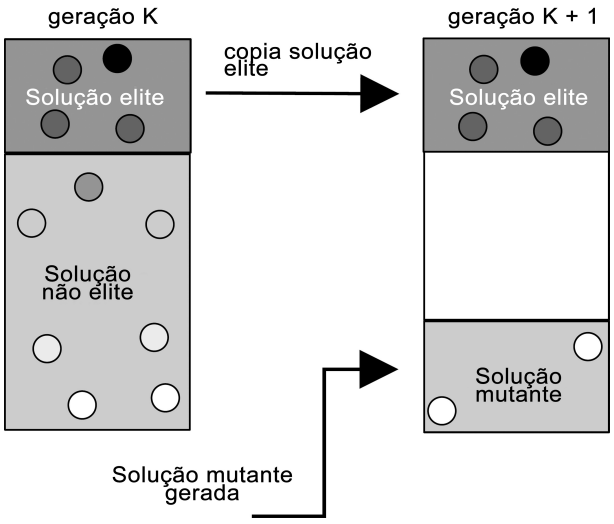


Figura 2.5: Todas as soluções Pe elite da população k são copiadas inalteradas para a população $k+1$ e as soluções mutantes Pm são geradas na população $k+1$ como vetores de chave aleatória (Gonçalves et al., 2014).

Bean (1994) seleciona dois pais aleatoriamente de toda a população para implementar o acasalamento em um RKGA e permite que um pai seja selecionado mais de uma vez em uma determinada geração.

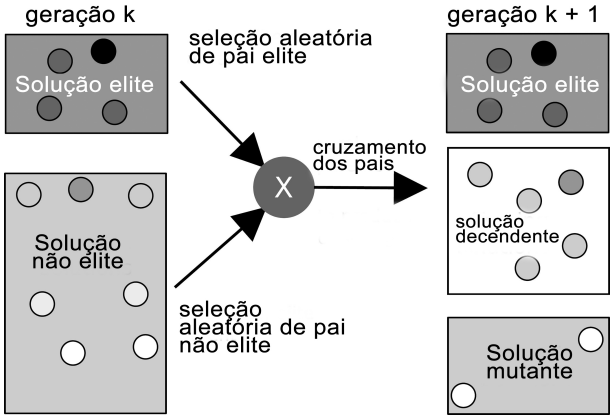


Figura 2.6: Para completar a população $k+1, p-Pe-Pm$ descendentes são criados combinando um pai selecionado aleatoriamente do conjunto de elite da população k com um pai selecionado aleatoriamente do conjunto não elite da população k. Os pais podem ser selecionados para acasalamento mais de uma vez por geração (Gonçalves et al., 2014).

2.4.4 BRKGA

O algoritmo BRKGA (Biased Random-Key Genetic Algorithm) é uma variante do RKGA e tem sido alvo de estudos na literatura nos últimos anos. Denominada por BRKGA em Gonçalves & Resende (2011), assim como o RKGA, esta heurística obtém uma solução viável pra o problema através da decodificação de uma solução codificada.

Diferentemente do RKGA, o algoritmo BRKGA utiliza um mecanismo tendencioso para seleção dos indivíduos para a operação de *crossover* (Gonçalves & Resende, 2011). Este tipo de seleção consiste em selecionar um indivíduo de boa qualidade e outro de qualidade pior para combinação e obtenção de uma nova solução. A seguir a apresentação do pseudo-código do BRKGA (Algoritmo 1). A primeira linha inicializa com uma variável $\int^*$ que armazena o valor de fitness da melhor solução com infinito (∞). Das linhas 2 a 23 são executadas até que a condição de parada da linha 2 seja satisfeita. A condição de parada pode ser a quantidade de vezes que será executada a geração da população. Na 3ª linha, é criada uma população inicial que é construída por meio da geração de chaves aleatórias. As linhas 4–20 são executadas até que o critério de reinicialização seja cumprido. O critério de reinicialização verifica se foi atingida uma população que contém indivíduos ótimos e pode ser, por exemplo, um número máximo de gerações que serão criadas. Na 5ª linha é realizada a partição dos indivíduos elite e não-elite, na linha 6 são copiados para a próxima população ($P^+ \leftarrow P_e$). Na 7ª linha são gerados os mutantes e adicionados para a próxima população ($P^+ \leftarrow P^+ \cup P_m$). A Figura 2.7 trás uma representação de geração de população utilizando o algoritmo genético.

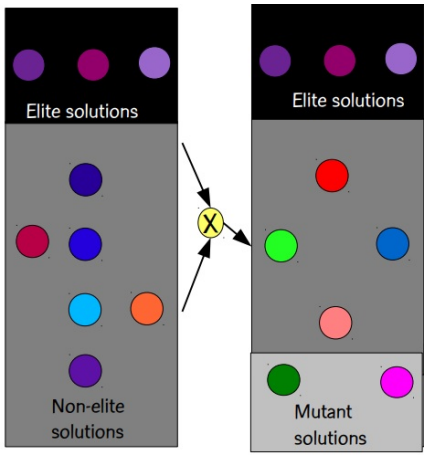


Figura 2.7: Combinação de elite e não elite.

Na Figura 2.7 um exemplo de geração de nova população através de seleção de elites e não elites. (Resende, 2014).

Das linhas 8 a 13 são gerados os indivíduos filhos. Essas linhas são executadas até preencher o restante da população $P - P_e - P_m$. Na 9ª linha é escolhido o pai A aleatoriamente entre os indivíduos elite e na linha 10 o pai B escolhido aleatoriamente entre os indivíduos não-elite. Na 11ª linha é gerado o indivíduo filho dos pais A e B e na linha 12 ele é adicionado para a próxima população $P^+ \leftarrow P^+ \cup \{c\}$.

Na 14ª linha a população atual é atualizada com a proxima população ($P \leftarrow P^+$). Na linha 15 a população P é percorrida para encontrar o melhor individuo baseado na função *fitness* e na 16ª linha, uma variável recebe esse indivíduo. Entre as linhas 17 e 20 é realizada uma validação se o valor do *fitness* encontrado é o maior, caso seja verdadeiro a variável que armazena a melhor solução é atualizada e o valor do melhor *fitness* também. E por fim, na linha 23 é retornado a solução X^*.

A procedure para Crossover (Algoritmo 2) recebe como parâmetros o pai A elite e o pai B não-elite.

Algorithm 1: BRKGA

```
1  BRKGA(p, p_e, p_m, ...)
2      f* ← ∞
3      X* ← ∅
4      while o critério de parada não seja satisfeito do
5          P ← ∅
6          for i ← 1 → |P| do
7              S_i ← ∅
8              for j ← 1 → n do
9                  S_i ← {Random(0, 1)}
10             end
11             P ← P ∪ {S_i}
12         end
13         Crie uma população P com vetores de n chaves aleatórias
14         while o critério de restart não seja satisfeito do
15             for i ← 1 → |P − P_e − P_m| do
16                 Escolha um pai a elite aleatoriamente
17                 Escolha um pai não elite b aleatoriamente
18                 c ← Crossover(a, b)
19                 Adicione o filho c à próxima geração: P+ ← P+ ∪ (c)
20             end
21             Atualize a população: P ← P+
22             Encontre a melhor solução P
23             X+ ← melhor solução de P
24             if f(X+) < f* then
25                 X* ← X+
26                 f* ← f(X+)
27             end
28         end
29     end
30     return X*
```

Algorithm 2: Operador *crossover*

Input : Indivíduo elite A,
Indivíduo não elite B,
probabilidade de herdar o gene de um pai elite p_a
Output: C

```
2  Crossover(A, B, p_a)
3      for j ← 1 → n do
4          r ← Random (0,1)
5          if r < p_a then
6              C[j] ← A[j]
7          end
8          else
9              C[j] ← B[j]
10         end
11     end
12     return C
```

Nas linhas de 1 a 8 são executadas n vezes, onde n é a quantidade de genes de um indivíduo. Na 2ª linha é utilizado um método de escolha aleatória enviesado para garantir que a escolha do pai elite seja maior. As linhas 4 e 8 é realizada uma validação, se for cara o filho recebe uma chave do pai A elite, caso não o filho recebe a chave do pai B não-elite. Por fim, na linha 10 o resultado traz o valor do filho C.

2.4.5 Algoritmos Genéticos Multipopulacionais

O algoritmo genético aplicado ao problema de otimização em uma abordagem unipopulacional torna mais intensa a busca em um espaço de soluções limitadas, com isso existe uma diminuição das possibilidades de se encontrar soluções de uma qualidade melhor e, até mesmo de soluções possíveis. A abordagem multipopulacional considera a postura exploratória na busca de soluções (Toledo et al., 2013).

Nos algoritmos genéticos multipopulacionais, várias populações evoluem em paralelo, possibilitando compartilhar os bons resultados entre as múltiplas populações, através de migrações de indivíduos, criando uma sinergia e tornando assim as abordagens multipopulacionais mais eficientes. Comumente, abordagem multipopulacional utiliza os mesmos valores para os parâmetros em todas as subpopulações. Com o objetivo dominar a migração dos indivíduos, outros parâmetros podem ser definidos, como: (i) um valor representando a taxa permitida de entrada dos indivíduos; (ii) tipo de comunicação que realiza a conexão entre as subpopulações; (iii) intervalo de migração que afeta a frequência de migração.

Segundo Aguirre et al. (2000), a migração tem que incluir estratégias para a seleção de migrantes e suas inclusão na sua nova subpopulação. A dimensão das subpopulações, sua comunicação; grau de conexão, a taxa de migração e a frequência, são fatores importantes relacionados com o desempenho de algoritmos genéticos. A Figura 2.8 ilustra o processo de formação das subpopulações.

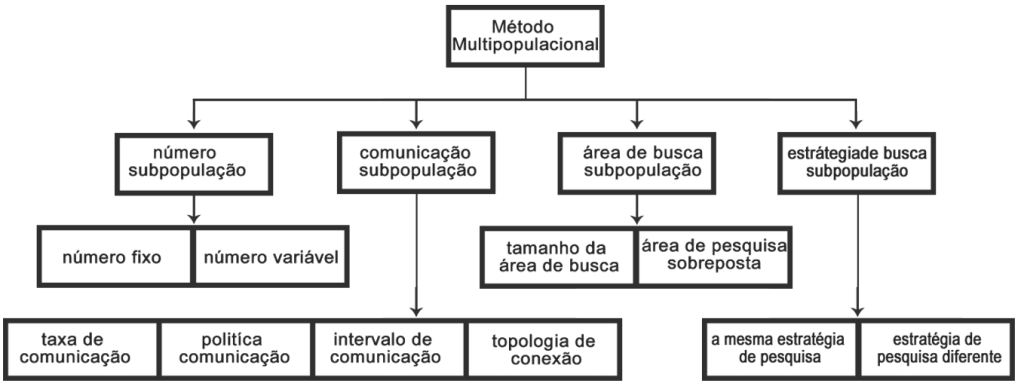


Figura 2.8: Diagrama de questões básicas e importantes de métodos multipopulacionais.

Capítulo 3

Revisão da literatura

Clusterização ou agrupamento é um termo bastante discutido e estudado pela literatura. Desde a década de 60, onde se tem os primeiros trabalhos a cerca do tema. Contudo, em sua maioria os estudos o Problema de Clusterização (PC) em que há uma definição prévia do valor de k, ou seja, o número de clusters são definidos (Ochi et al., 2004). De acordo com Tan et al. (2013), é provável que um dos problemas de clusterização mais conhecido seja o de encontrar e determinar o número ideal de clusters. Várias técnicas não supervisionadas podem ser usadas para o processo de avaliação de soluções. Uma dessas técnicas consiste analisar o valor da Soma dos Erros Quadráticos (Mirkin, 1999) das soluções obtidas em função do número de grupos. A ideia central é obter naturalmente a quantidade de grupos, buscando por grupos em que há uma inflexão no valor do SEQ (Soma de Erros Quadráticos). Segundo Semaan et al. (2012) essa abordagem pode falhar em algumas situações, quais sejam: quando existem grupos entrelaçados, superpostos ou até mesmo aninhados. Na Equação (c_i, x) indica a distância (Euclidiana) entre o objeto x e o centróide a ele mais próximo (c_i).

$$SEQ = \sum_{i=1}^{k} \sum_{x \in C_i} dist(c_i, x)^2$$

Vale destacar um método bastante conhecido na literatura, para o Problema de clusterização (PC), que é o K-means (MacQueen et al., 1967). O método k-means, utiliza o conceito de centróide para representar os clusters. Está abordagem de SEQ pode ser realizada neste tipo de problema.

Outra abordagem concernente à determinação do número ideal de grupos consiste na avaliação da função silhueta proposta por Rousseeuw (1987) e utilizada em diversos trabalhos, dentre os quais: Wang et al. (2007) e Tseng & Yang (2001) mais especificamente, aplica-se um algoritmo de agrupamento para alguns valores de k no intervalo $[2, n]$, escolhendo-se como o k ideal aquele associado ao maior valor da função silhueta (ver equação 3.1). Na Figura 3.1 ilustração SEQ versus número de grupos e a Figura 3.1 mostra os pontos com os

índices silhueta .

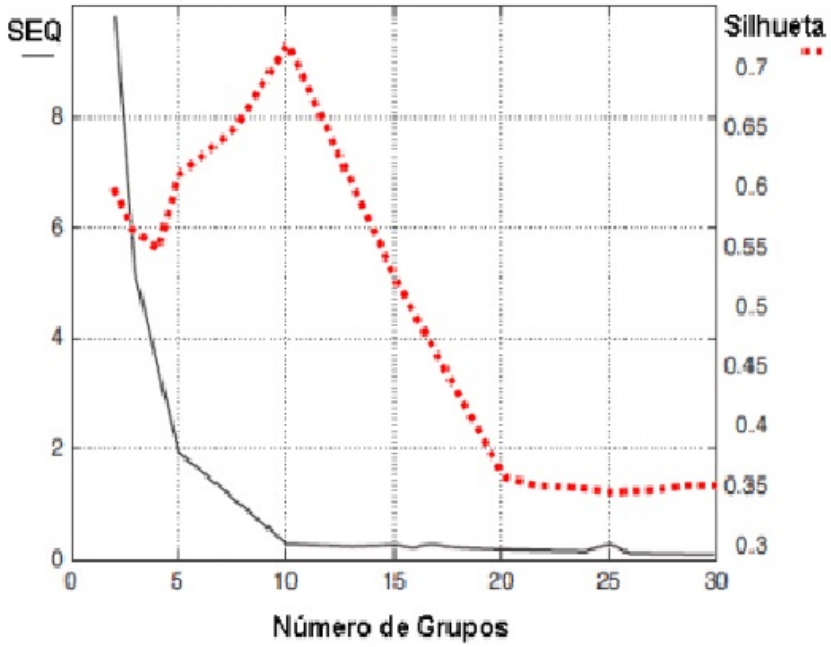


Figura 3.1: SEQ versus Número de Grupos e Silhueta versus Número de Grupos (adaptação de (Tan et al., 2013))

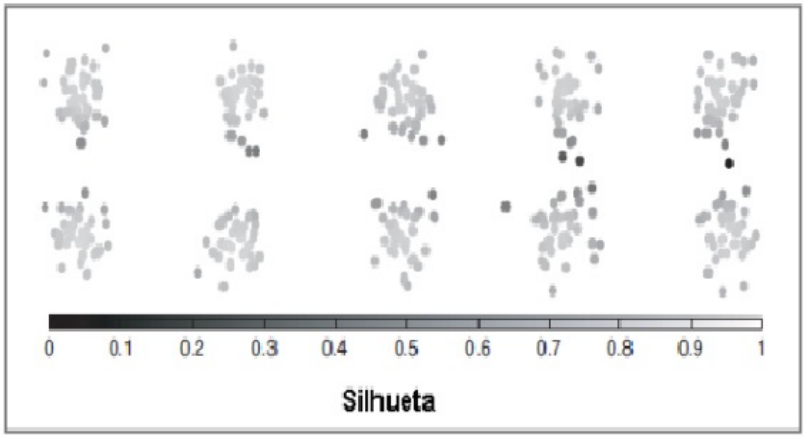


Figura 3.2: Instância associada ao gráfico da Figura 3.1 (Tan et al., 2013)

O Bisecting k-Means, proposto por Steinbach et al. (2000), corresponde a uma versão hierárquica do k-Means, em que a cada iteração, um grupo é selecionado e dividido em dois novos grupos. Dessa forma, novamente são obtidas soluções para todos os valores de k pertencentes a um intervalo de k pré-estabelecido. O critério de seleção do grupo a ser dividido pode ser, por exemplo, o grupo com maior diâmetro (distância entre dois objetos em um mesmo grupo) ou o grupo com o menor valor de função silhueta.

Já o problema de Clusterização Automática (PCA) não apresenta um número tão expressivo de estudos quanto os estudos a cerca do Problema de Clusterização (PC). O X-means (Pelleg et al., 2000) faz uma adaptação do método k-means (MacQueen et al., 1967) para o Problema de Clusterização Automática (PCA). Um algoritmo recursivo onde executa divisões binárias do espaço até que se chegue no melhor valor de k, nos limites fornecidos. Para decidir qual o valor de k será retornado, o X-means utiliza o índice BIC (Bayesian Information Criteion) (Neath & Cavanaugh, 2012).

Como exemplos de trabalhos relacionados a clusterização automática, os métodos baseados em modelo apresentam um padrão de referência para cada cluster. Eles tentam otimizar a curva entre os objetos dados e algum modelo matemático. Um algoritmo baseado em modelo pode descobrir clusters construindo uma função de densidade que reflete a distribuição espacial dos pontos de dados. Ele também conduz a um modo de determinar automaticamente o número de clusters baseado na estatística padrão, identificando ruídos no relatório e assim produzindo métodos de Clusterização robustos. Ao contrário dos métodos de Clusterização convencionais, que primariamente identificam grupos de objetos, os métodos de Clusterização Baseados em Modelos, também chamados de Métodos de Clusterização Conceitual, realizam uma etapa adicional para encontrar descrições características para cada grupo, onde cada grupo representa um conceito ou classe. Sendo assim, a qualidade de Clusterização não é unicamente uma função dos objetos individuais (Han et al., 2001).

Na literatura, ainda sobre o problema de clusterização automática, diversos trabalhos propõem algoritmos baseados em meta-heurísticas com objetivo de alcançar um número ideal de clusters. Alguns trabalhos foram base para o desenvolvimento deste trabalho, os que se destacam são os trabalhos: Tseng & Yang (2001) Wang et al. (2007) Cruz (2010)Semaan et al. (2012).

Em Tseng & Yang (2001) é apresentado um algoritmo genético denominado CLUSTERING, que utiliza também o índice silhueta para determinar os índices e assim criar os clusters ideiais. Esse algoritmo constrói um grafo conexo, indetifica os seus componentes e atua na clusterização desses componentes. O seu objetivo é maximizar o índice da silhueta (*i.e.,* quanto mais próximo de 1,0 o índice silhueta, melhor seu posicionamento no cluster).

O algoritmo CLUES (Clusterisng based on Local Shirinking), segue a mesma abordagem de problema de clusterização automática, onde permite a aplicação da índice silhueta ou índice CH (índice de Calinski-Harabasz) para determinar o número de clusters ideal. Um algoritmo iterativo que, com a utilização do encolhimento (Shirinking procedure) baseado nos k-vizinhos mais próximos, realiza a união dos objetos mais homogêneos. Logo após a aplicação do procedimento de encolhimento, o algoritmo CLUES constrói soluções, avaliando-as mediante o valor do índice silhueta ou do índice CH. Em Wang et al. (2007) é mencionado que os resultados obtidos com a utilização do índice silhueta e do índice CH foram comparadados. A partir dessa comparação, observou-se que mediante a aplicação do índice silhueta

foram produzidas soluções de melhor qualidade no que concerne ao número de clusters definidos e à formação de soluções denominadas perfeitas em tal trabalho.

O trabalho de Cruz (2010) traz uma proposta de algoritmos heurísticos mais aprimorado no que tange a técnica de construção, de busca local e de perturbação. Especificamente, esses algoritmos foram baseados nas meta-heurísticas Algoritmos Genéticos (Bean, 1994), Busca Local Iterada (Lourenço et al., 2003) (Iterated Local Search) e GRASP (Cano et al., 2002) (Greedy Randomized Adaptive Search Procedure). O ponto individual desses algoritmos está na integração de métodos para a construção de grupos parciais, definição de Memória Adaptativa (Ahmadi & Osman, 2005) e Buscas Locais (Ishibuchi & Murata, 1996) que utilizam o algoritmo k-means para a união de grupos parciais.

Ainda no trabalho de Cruz (2010), foram propostos também métodos híbridos. Estes métodos utilizam algumas das soluções produzidas pelos algoritmos heurísticos, ou seja, soluções associadas com alguns valores de k e que sejam consideradas promissoras no que refere-se ao número ideal de clusters, porém não obrigatoriamente a melhor solução para tal número. Considerando estes valores específicos de k, são aplicadas duas formulações de programação inteira, quais sejam: para o problema de agrupamento com diâmetro mínimo e dos k-Medoids (Cruz, 2010). Nos experimentos apresentados no trabalho foram realizadas comparações com o algoritmo da literatura CLUES (Wang et al., 2007).

O trabalho de Cruz (2010) propõe um método de classificação baseado em densidade que tem por objetivo a identificação do número ideal de clusters. Ou seja, identificar de forma não supervisionada padrões semelhantes e que possam refletir na forma como os dados são estruturados. O método proposto consiste na aplicação de um algoritmo de agrupamento clássico baseado em densidade DBSCAN (*Density Based Spatial Clustering of Applications with Noise*) (Ester et al., 1996). Esse algoritmo necessita de dois parâmetros, sejam eles: a distância e a densidade (quantidade de objetos no raio de alcance de um objeto, incluindo o próprio objeto). Para a calibração desses parâmetros foi utilizada a técnica proposta na literatura denominada DistK. Esta abordagem consiste em, para um valor inteiro k^* fornecido como o parâmetro de entrada, analisar o comportamento das distâncias entre cada objeto e o seu vizinho de índice $k*$ mais próximo, ou seja, o seu k*-ésimo* vizinho mais próximo. O objetivo desse procedimento é identificar os valores de distâncias que resultariam em soluções de qualidade, obtidas mediante a execução do algoritmo DBSCAN.

O DBSCAN foi adaptado para que todos os objetos que compõem uma instância sejam considerados. Essa modificação decorre do fato de o algoritmo DBSCAN tradicional classificar os objetos em Interiores; objetos que pertencem ao interior de um grupo baseado em densidade. Deve possuir uma quantidade de objetos em seu raio *raioDBSCAN* igual ou superior ao parâmetro *qtdeObjetos* - 1, limítrofes; não é um objeto central, mas é alcançável por ao menos um objeto central, ou seja, está dentro do raio de vizinhança de algum objeto central, ruídos; demais objetos que não são centrais e nem estão na vizinhança de um objeto central, e os objetos classificados como ruídos serem ignorados pelo algoritmo em sua

versão original.

Foram obtidos diferentes valores para cada um de seus parâmetros, como o algoritmo DBSCAN é determinístico, com o objetivo de encontrar soluções diversificadas no que diz respeito ao número de grupos e à distribuição dos objetos nesses grupos. Conforme Naldi et al. (2011), os índices de validação relativos têm sido utilizados e investigados extensivamente, tendo estes apresentado resultados satisfatórios em diversos cenários.

Semaan et al. (2012) apresenta MRDBSCAN, uma proposta baseada no BDSCAN (Ester et al., 1996), considerando diferentes parâmetros. Os parâmetros são obtidos através da tecnica DistK, que se basea na distância dos k-vizinhos mais próximos de cada objeto. O índice silhueta determina a qualidade das soluções obtidas, quanto mais próximo de um o valor do índice silhueta, melhor será a solução. A proposta de Semaan et al. (2012) aplica quatro regras considerando um conjunto de valores de k^* para a analise de DistK. São as regras: **Mediana**, que considera o valor da mediana obtida a partir de V. **Maior** considera o maior valor de V. **Pico10** divide o vetor em dez partes com a mesma quantidade de distâncias e verifica a maior diferença. **Pico20** assim como o **pico10**, é verificada a maior diferença, agora em vinte partes, distribuidas com a mesma quantidade de distâncias.

Ainda é apresentado por Semaan et al. (2020) duas abordagens o AECBL1 e *híbrida* ILS-DBSCAN, que utilizam a *Estatística de Hopkins–EH* (Banerjee & Dave, 2004); com um critério interno em que nenhuma informação a priori é necessária para a realização das análises. Quando uma instância era identificada com Tendência à formação de Agrupamentos (Heurística para EH), o ILS-DBSCAN era selecionado para a resolução do problema. Caso contrário, o AECBL1 deveria ser executado. Em Hosseini et al. (2010) a regra de decisão adotada não rejeita a hipótese nula quando o valor obtido para *EH* for igual ou menor do que 0,5. Um estudo realizado por Assunçao & Reis (2000) apontou que tal teste apresenta maior probabilidade de rejeitar a hipótese nula, quando a hipótese alternativa é verdadeira, do que diversos testes bem conhecidos de aleatoriedade espacial. Nos experimentos realizados a solução proposta apresenta bom desempenho, utilizando a variação entre as duas abordagens.

Capítulo 4

Método proposto: Multipopulacional BRKGA para o PCA

Este trabalho tem por objetivo tratar do Problema de Clusterização Automática utilizando o índice de silhueta como função objetivo. Nas seções a seguir, será apresentado o algoritmo multipopulacional proposto para a resolução deste problema, o MultPop-BRKGA-PCA. Inicialmente, na Seção 4.1, é apresentada uma visão geral do algoritmo. Em seguida, na Seção 4.2, é apresentado um algoritmo BRKGA que irá evoluir cada subpopulação de forma independente. Por fim, na Seção 4.3, serão apresentadas diferentes estratégias para selecionar a subpopulação que irá evoluir a cada iteração.

Os trabalhos apresentados pela literatura sobre os métodos multipopulacionais confirmam que usar a multipopulação é um dos métodos mais eficazes para manter a diversidade populacional (Trojanowski & Wierzchoń, 2003). Em algoritmos de otimização inspirados na natureza, a diversidade é indicada como a diferença entre as soluções candidatas e o progresso reside fundamentalmente na existência de variações populacionais.

Para tornar os métodos de várias populações mais eficientes, vários métodos básicos e questões importantes do projeto do algoritmo são discutidas, as quais são mostradas na Figura 2.8. Esses problemas incluem o número de subpopulações, a comunicação entre subpopulações, a área de pesquisa de subpopulações e a estratégia de pesquisa de subpopulações.

4.1 Algoritmo Multipopulacional (MultPop-BRKGA-PCA)

Este trabalho propõe um novo método baseado em uma meta-heurística bem conhecida, o BRKGA (Bised Random Key Genetic Algorithm) acoplado com o conceito de multipopulação, a fim de maximizar a homogeneidade dos *clusters* formados. Multipopulação é uma estratégia de otimização eficaz que é frequentemente usada em algoritmos evolutivos (AEs) para melhorar o desempenho de otimização (Andrade et al., 2021). Nos últimos anos, o conceito de multipopulação é frequentemente usado para melhorar a eficiência das meta-heurísticas populacionais.

Nesta estratégia, a população original é dividida em várias subpopulações. Então, para cada subpopulação, um determinado procedimento de evolução é executado. O objetivo da multipopulação é manter a diversidade populacional e aumentar o processo de exploração, a fim de evitar convergência prematura para ótimos locais. Os algoritmos de multipopulação consistem em três etapas principais (Ma & Song, 2013): (i) divisão da população para formar subpopulações (ii) pesquisa em cada subpopulação e (iii) comunicação da subpopulação.

O MultPop-BRKGA-PCA usa várias populações BRKGAs independentes simultaneamente, de forma que cada população busca a melhor clusterização para um valor de k (número de grupos). A gestão multipopulacional é feita por uma estratégia de seleção que leva em consideração a qualidade de cada subpopulação. Nesta abordagem, a cada iteração, uma subpopulação é escolhida para evoluir. Cada evolução de uma subpopulação corresponde a uma iteração do BRKGA. Assim, cada população evolui em velocidades diferentes de acordo com a qualidade de suas soluções.

Inicialmente, o MultPop-BRKGA-PCA divide a única população em várias subpopulações que evoluirão de forma independentemente. Uma abordagem semelhante foi apresentada por El Dor et al. (2012), em que os autores apresentaram um algoritmo PSO multipopulacional em um espaço de busca particionado para otimização contínua. O algoritmo proposto começa com um conjunto de subpopulações disjuntas $\mathscr{P} = \{P_1, \ldots, P_\ell\}$, uma para cada valor de k, ou seja, cada subpopulação representa um conjunto de soluções para um número particular de *clusters* (Problema de k-clusterização usando o índice de silhueta). Como no PCA, o valor de k varia entre 2 e $\lfloor\sqrt{n}\rfloor$ (ver Seção 2.1), então a subpopulação P_1 irá conter apenas soluções com 2 *clusters*, a subpopulação P_2 apenas soluções com 3 *clusters*, e assim por diante, até a subpopulação P_ℓ que irá conter apenas soluções com $\lfloor\sqrt{n}\rfloor$ *clusters*. A ideia é fazer com que cada iteração do MultPop-BRKGA-PCA corresponda a uma iteração de um BRKGA para o problema de com k fixo.

Dado um conjunto de subpopulações a cada iteração, uma subpopulação proeminente é selecionada com base em informações históricas e outras informações relacionadas. Então, a subpopulação selecionada evolui sob uma iteração de BRKGA. Essa abordagem leva a diferentes números de iterações para cada subpopulação, ou seja, subpopulações com melhor qualidade evoluirão por mais iterações.

O Algoritmo 3 apresenta um *framework* geral do MultPop-BRKGA-PCA. Em resumo, o método possui duas fases principais: (1) `selectPop`, que seleciona uma subpopulação em $\mathscr{P}$ para evoluir (mais detalhes na Seção 4.2) e (2) `BRKGAevolve`, que realiza uma iteração de BRKGA para o problema de k-clusterização (ver Seção 4.3).

Algorithm 3: Pseudo-código do MultPop-BRKGA-PCA.

```
1 Gerar um conjunto de subpopulações iniciais 𝒫 = {P_1, ..., P_ℓ}
2 while critério de parada não satisfeito do
3 |   i ← selectPop(𝒫) // Seção 4.2
4 |   BRKGAevolve(P_i)// Seção 4.3
5 end
6 return bestIndividual(𝒫)
```

Na literatura, os métodos multipopulacionais comumente trocam indivíduos entra as subpopulações. Note que no MultPop-BRKGA-PCA, não existe comunicação entre as subpopulação já que a solução de uma população $P_i \in \mathscr{P}$ não é viável em relação ao problema abordado pelas soluções de uma população $P_j \neq P_i$, já que cada população resolve um problema de k-clusterização com valores de k distintos.

4.2 Seleção da população

Nas subseções seguintes serão apresentadas as estratégias propostas para a seleção da próxima subpopulação a ser evoluída.

4.2.1 Roleta

O modelo de seleção da roleta baseou-se na seleção da roda da roleta; também conhecido como seleção de aptidão proporcional, para selecionar soluções potencialmente úteis para recombinação em um AG clássico (Lipowski & Lipowska, 2012). Em particular, no caso do MultPop-BRKGA-PCA, a probabilidade de uma subpopulação ser escolhida para evoluir é proporcional à aptidão de seu melhor indivíduo. A probabilidade p_i de selecionar a subpopulação P_i é dada por:

$$p_i = \frac{\texttt{best}(P_i)}{\sum_{j=1}^{\ell} \texttt{best}(P_j)}. \tag{4.1}$$

em que $\texttt{best}(P_i)$ é o índice de silhueta do melhor indivíduo na subpopulação P_i.

4.2.2 r-random

A seleção r-random consiste em selecionar k subpopulações de $\mathscr{P}$ ao acaso e, escolher a subpopulação contendo o indivíduo com melhor aptidão dentre todas as subpopulações

selecionados aleatoriamente. Basicamente, os passos seguidos pela solução r-random são:

1. Selecione k subpopulações aleatoriamente;
2. Escolha a subpopulação com o melhor indivíduo.

4.2.3 dynamic r-random

De forma similar ao r-random, o dynamic r-random realiza os seguintes passo:

1. Selecione as k melhores populações;
2. Escolha uma dessas populações aleatoriamente e a evolua;
3. Faça $k \leftarrow k+1 \mod \ell$.

4.2.4 Torneio

A seleção do Torneio consiste nas seguintes etapas:

1. Dividir as subpopulações em pares;
2. Realizar a disputa entre cada par,
3. Evoluir a subpopulação vencedora; e
4. Repetir todas as etapas considerando apenas as subpopulações vencedoras até isso resultar em uma subpopulação com melhor aptidão. Finalmente, a subpopulação final é evoluída.

Porém, ao contrário das estratégias anteriores, a seleção do torneio realiza várias iterações de evolução durante o processo.

4.2.5 Simulate annealing

Esta seleção é baseada na meta-heurística *simulated annealing* proposta por Kirkpatrick et al. (1983) que se fundamenta em uma analogia com a termodinâmica dita recozimento ou *annealing*, utilizado em metalurgia para obtenção de estados de baixa energia em um sólido.

Em cada iteração, o método decide se troca de população de acordo com a diferença entre os valores da função-objetivo — a qual o método faz analogia à energia do material. De acordo com a temperatura e a diferença de energia (melhores índices de silhueta de cada população), o algoritmo pode trocar uma população P_i por uma P_j com pior índice. A variável T (temperatura) regula a aleatoriedade dessa troca para uma pior população. Quanto maior

for T, maior a componente aleatória que será incluída na próxima solução escolhida. À medida que o algoritmo progride, o valor de T é decrementado, assim, o algoritmo diminui a probabilidade de selecionar populações com baixo índice de silhueta.

O Algoritmo 4 ilustra esse processo de seleção. Inicialmente a temperatura T é inicializada com um valor alto e a medida que ocorrem as iterações do algoritmo, a temperatura vai decaindo de acordo com o parâmetro α. O algoritmo utiliza uma população de referência P_{ref} e uma população candidata P_i, escolhida aleatoriamente a cada iteração, que poderá se tornar a população referência na próxima iteração. A cada iteração, o algoritmo evolui P_{ref} e P_i (linhas 9–10) e calcula a energia de ambas populações. Para o *simulatted annelling*, uma menor energia representa uma menor solução, dessa forma, foi definido que a `energia`(P_i) corresponde ao índice de silhueta da melhor solução em P_i multiplicado por -1. Por fim, na linha 18, se P_i possuir uma menor energia que P_{ref}, então a população referência P_{ref} é atualizada para a população P_i; caso contrário, P_i pode ser aceita como população referência de acordo com uma função de probabilidade conhecida como fator de Boltzmann que é dada por $(e^{(-\Delta/T)})$, em que Δ é a diferença de energia entre P_i e P_{ref}. A aceitação desse tipo de solução é mais provável a altas temperaturas (iterações iniciais) e bastante improvável a

temperaturas reduzidas (iterações finais).

Algorithm 4: Seleção por simulated annealing.

```
1  Procedure SimmulatedAnnealing
2      P ← Geração das populações iniciais
3      T ← Temperatura Inicial
4      α ← Taxa de decaimento
5      P_best ← MelhorPopulação(P)
6      P_ref ← P_best
7      while até estar satisfeito: do
8          P_i ← RandomPopulation(P \ {P_ref} )
9          Evolve(P_ref)
10         Evolve(P_i)
11         if P_ref for melhor que P_best then
12             P_best ← P_ref
13         end
14         if P_i for melhor que P_best then
15             P_best ← P_i
16         end
17         Δ ← energia(P_i) - energia(P_ref)
18         if Δ < 0 OU random(1,0) < e^(-Δ/T) then
19             P_ref ← P_i
20         end
21         T ← Atualiza a temperatura em função de α
22     end
23     return bestIndividual(P)
```

4.2.6 Pontas

Após testes empíricos, observou-se que em boa parte das instâncias as melhores soluções possuíam um valor de k perto dos extremos do intervalo $[2, \sqrt{n}]$. Com base nesta observação, a seleção pelas pontas inicia o processo com uma população que represente um valor extremo de k. O algoritmo trabalha com uma janela com 3 populações (P_{i-1}, P_i e P_{i+1}) que vai se deslocando para lado ou para outro com igual probabilidade. O Algoritmo 5 apresen-

tação essa estratégia de seleção.

Algorithm 5: Pontas

```
1  Procedure Pontas
2      inicialize 𝒫                                  ▷ Criação das populações iniciais
3      i ← 1 ou ℓ                                    ▷ Escolha aleatoriamente o valor 1 ou ℓ
4      notImprovement ← 0
5      while até estar satisfeito do
6          improvement ← False
7          for j ∈ {0,1,2}                           ▷ Para cada população na janela
8           do
9              bestValue ← Fitness(P_{i-1+j})
10             Evolve(P_{i-1+j})
11             if Fitness(P_{i-1+j}) for melhor que bestValue then
12                 improvement ← True
13             end
14         end
15         if improvement == False then
16             notImprovement ← notImprovement + 1
17         end
18         if não melhorar Y vezes then
19             Mude o valor de i
20         end
21     end
22     return bestIndividual(𝒫)
```

4.2.7 Roleta pela média

O modelo de seleção da roleta pela média difere da seleção da roleta por usar a qualidade média das soluções no lugar do melhor. Dessa forma, a probabilidade p_i de selecionar a subpopulação P_i é dada por:

$$p_i = \frac{\mathtt{q}(P_i)}{\sum_{j=1}^{\ell} \mathtt{q}(P_j)}. \tag{4.2}$$

em que $\mathtt{q}(P_i) = \mathtt{media}(P_i)/b^*$, $\mathtt{media}(P_i)$ é a média dos índices de silhueta dos indivíduos na subpopulação P_i e b^* é a melhor índice encontrado.

4.3 BRKGA para o problema de clusterização com k fixo

Nesta seção será descrito o BRKGA que é utilizado na linha 4 do algoritmo 3. Conforme explicado na Seção 2.4.4, a implementação do BRKGA para um problema de otimização requer, basicamente, a especificação do procedimento de decodificador para o problema particular.

O Algoritmo 6 descreve o decodificador para o problema de k-clusterização. Cada solução do problema de k-clusterização está associada a um conjunto de n chaves aleatórias r_i, para $i = 1, \ldots, n$. Cada chave aleatória é um número real no intervalo $[0, 1)$ e corresponde a um objeto do conjunto X. Para decodificar as chaves aleatórias como soluções viáveis, o MultPop-BRKGA-PCA ordena os valores de forma crescente, assim, as chaves aleatórias ordenadas correspondem à sequência dos objetos a serem inseridos nos *clusters* atuais. Nas linhas 4–8, os k objetos com a menor chave aleatória são usados como representantes dos *clusters*, ou seja, k *clusters* são criados, cada um com um representante. Os demais objetos também são atribuídos aos *clusters* de acordo com a sequência (linhas 10–14). Para isso, é aplicado um procedimento guloso para obter o melhor cluster C_j para qual o objeto x_i deverá ser atribuído.

Algorithm 6: Decoder do BRKGA para o problema de k-clusterização.

Input : Chaves-aleatórias $R = \{r_1, \ldots, r_n\}$, número de *clusters* k, conjunto de objetos X.

Output: Uma k-clusterização $\pi = \{C_1, \ldots, C_k\}$.

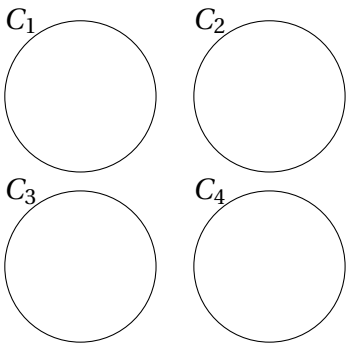
1 **Procedure** *Decoder*(R, k, X)
2 $X' \leftarrow X$
3 Gerar um conjunto de subpopulações iniciais $\mathscr{P} = \{P_1, \ldots, P_\ell\}$
4 **for** $j \leftarrow 1 \rightarrow k$ **do**
5 $C_j \leftarrow \emptyset$ // Initialize C_j
6 $i \leftarrow \arg\min \{ r_i \mid i \in [1, |X'|] \}$
7 $X' \leftarrow X' \setminus \{x_i\}$
8 $C_j \leftarrow C_j \cup \{x_i\}$
9 **end**
10 **for** $i' \leftarrow k+1 \rightarrow n$ **do**
11 $i \leftarrow \arg\min \{ r_i \mid i \in [1, |X'|] \}$
12 $X' \leftarrow X' \setminus \{x_i\}$
13 $j \leftarrow \arg\min \{ s(\{C_j \cup \{i\}\}) \mid j \in [1, k] \}$
14 $C_j \leftarrow C_j \cup \{x_i\}$
15 **end**
16 **return** $\pi = \{C_1, \ldots, C_k\}$

A Figura 4.1 mostra o exemplo da representação e decodificação proposta no algoritmo MultPop-BRKGA-PCA, considerando uma instância com $k = 4$ e $n = 10$. Após ordenar os valores do vetor R (Figura 4.1a), os objetos são alocados de acordo com o índice de cada gene.

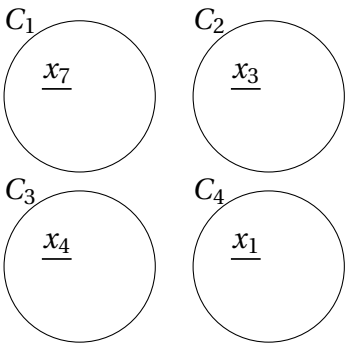
Inicialmente, são construídos k clusters vazios, como mostra a Figura 4.1b. Neste exemplo, os k objetos com menor chaves-aleatórias (x_7, x_3, x_4 e x_1) são alocados como representantes de *clusters* (ver Figura 4.1c). Em seguida, os demais objetos são alocados em cada um dos k *clusters* de forma gulosa, de acordo com índice de silhueta aplicado na clusteriação parcial considerando a entrada do objeto, *e.g.*, o objeto x_2 possui a menor chaves-aleatórias dentre os objetos não representantes, logo ele será o primeiro a ser alocado. Para decidir o *cluster* de x_2, o algoritmo avalia o índice de silhueta dos seguintes *clusters* parciais: $\{x_7, x_2\}$, $\{x_3, x_2\}$, $\{x_4, x_2\}$ e $\{x_1, x_2\}$. A Figura 4.1d mostra que, neste exemplo, x_2 foi alocado no cluster C_2 de acordo com o coeficiente silhueta. Por fim, a Figura 4.1e mostra a alocação final.

Objetos	x_0	x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	x_9
r_i	0,7	0,4	0,5	0,2	0,3	0,8	0,9	0,1	0,9	0,6

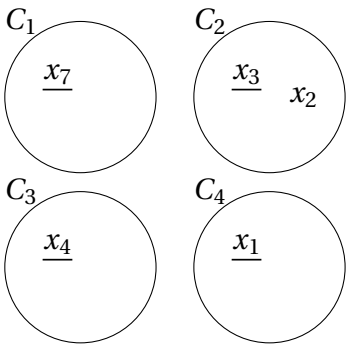
(a) Chaves-aleatórias.



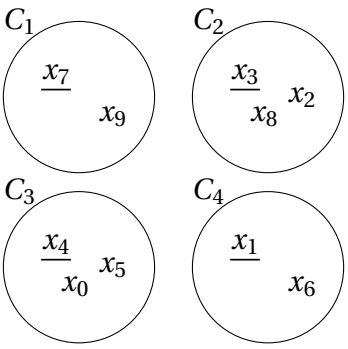
(b) Criação dos k *clusters*.



(c) Escolha dos representantes.



(d) Alocação gulosa de acordo com a ordem das chaves-aleatórias.



(e) Solução final.

Figura 4.1: Exemplo de decodificação para o problema da k-clusterização.

Capítulo 5

Resultados computacionais

Os experimentos foram conduzidos em uma máquina com as seguintes configurações: processador Intel Core i7 com 3,7 GHz, 16 GB de RAM, com sistema operacional Ubuntu 18.04.02 LTS. Os algoritmos foram codificados em C++ e compilados com g++ 7.5.0 e *flag* '-O3'. Foi utilizado o *framework* BRKGA implementado em C++ e desenvolvido por Resende (2011). Neste *framework* apenas a função do decodificador deve ser implementada.

5.1 Instâncias

As instâncias selecionadas têm tamanhos que variam de 30 a 2.000 objetos e 2 a 13 atributos, com diferentes graus de dificuldade e características, por exemplo, coesão de grupo, separação, formatos e densidades. Foram utilizadas 55 instâncias amplamente utilizadas por muitos autores para avaliar seus métodos. Estas instâncias estão disponíveis no trabalho de Semaan et al. (2014). Algumas são bastantes conhecidas na literatura como *Maronna* (Maronna & Jacovkis, 1974), *200DATA* (Fisher, 1936), *Vowel* (Hastie et al., 2009), *Iris* (Fisher, 1936), *Ruspini* Bezdek (1974) e *Broken Ring* (Wang et al., 2007). Outras instâncias foram utilizadas como `200p4c`, este tipo de instância segue um padrão de pontos e *clusters*, como por exemplo: o termo `200p` representa o número de pontos, `4c` indica possíveis 4 *clusters* e caso possua o termo `1` no final, indica-se que a instância não é comportada.

Um breve exemplo pode ser compreendido pelas Figuras 5.1 e 5.2, representando dois tipos de instâncias: comportadas e não comportadas. A literatura considera que instâncias bem comportadas possuem *clusters* bem definidos, ou seja, alguns *clusters* podem ser facilmente identificados visualmente através de um *plot* dos pontos. O que não acontece nas instâncias não comportadas. Os detalhes de cada instâncias são exibidos na Tabela 5.1. Nesta tabela, as colunas 'Melhor' e 'k' representam o melhor índice de silhueta encontrado na literatura e a quantidade de *clusters* para esta solução, respectivamente.

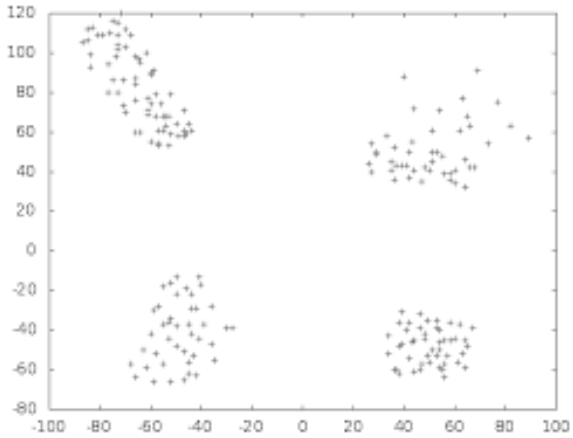


Figura 5.1: Instância comportada 200p4c.

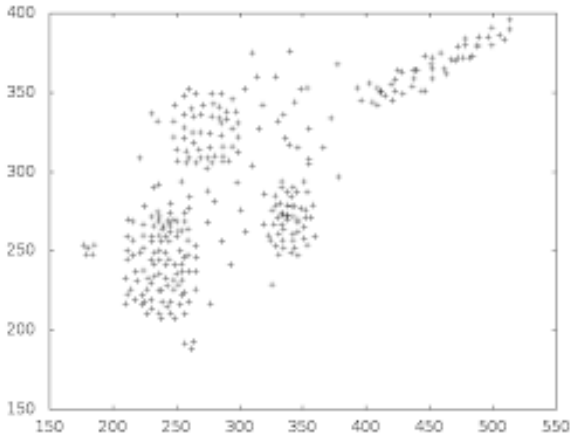


Figura 5.2: Instância não comportada 300p4c1.

5.2 Calibração dos parâmetros

Para encontrar uma configuração adequada dos parâmetros do MultPop-BRKGA-PCA foi utilizado o pacote `irace` (López-Ibáñez et al., 2016). Seu principal objetivo é configurar automaticamente os parâmetros de algoritmos de otimização, através de um método chamado *iterated racing*. O `irace` configurou os seguintes melhores parâmetros: tamanho da população ($|P|$), taxa da população elite (P_e), taxa da população mutantes (P_m), rhoe (ρ), temperatura (T) e alpha (α). Para cada versão do algoritmo proposto, os resultados são apresentadas a seguir.

- Roleta: $|P| = 262$, $P_e = 0,13$ $P_m = 0,35$ e $\rho = 0,66$;
- r-random: $|P| = 189$, $P_e = 0,22$, $P_m = 0,26$, $\rho = 0,65$;
- dynamic r-random: $|P| = 274$, $P_e = 0,46$, $P_m = 0,10$, $\rho = 0,44$;
- Torneio: $|P| = 244$, $P_e = 0,27$, $P_m = 0,12$, $\rho = 0,72$;

Tabela 5.1: Detalhes das instâncias consideradas.

Instância	Melhor	k	Instância	Melhor	k	Instância	Melhor	k
1000p14c	0,831	11	300p2c1	0,776	5	200DATA	0,823	3
1000p27c1	0,563	14	300p2c1	0,776	5	800p23c	0,787	22
1000p5c1	0,643	7	300p3c1	0,677	4	200p12c1	0,575	9
1000p6c	0,736	6	300p3c	0,766	3	800p4c1	0,714	4
100p10c	0,834	10	300p4c1	0,592	4	200p2c1	0,764	3
100p2c1	0,743	5	300p6c1	0,664	7	900p12c	0,841	10
100p3c1	0,597	3	400p17c1	0,552	15	200p3c1	0,680	3
100p3c	0,786	3	400p3c	0,799	3	900p5c	0,716	7
100p7c1	0,551	7	400p4c1	0,620	4	200p4c1	0,754	4
100p7c	0,834	7	500p3c	0,825	3	Brokenring	0,500	5
1100p6c1	0,685	6	500p4c1	0,660	3	200p4c	0,773	4
1300p17c	0,823	15	500p6c1	0,669	6	iris	0,687	3
1500p20c	0,823	13	600p15c	0,781	16	200p7c1	0,576	9
1500p6c1	0,660	6	600p3c1	0,721	3	Maronna	0,575	4
1500p6c	0,694	5	700p15c1	0,680	17	300p13c1	0,594	9
1800p22c	0,804	18	700p4c	0,797	4	Ruspini	0,738	4
2000p11c	0,713	8	800p10c1	0,507	8	Vowel2	0,448	9
2000p9c1	0,623	7	800p18c1	0,694	19			

- Simulate Annealing: $|P| = 234$, $P_e = 0,14$, $P_m = 0,27$, $\rho = 0,68$, $T = 12$, $\alpha = 0,9$;
- Pontas: $|P| = 191$, $P_e = 0,32$, $P_m = 0,18$, $\rho = 0,65$;
- Roleta pela média: $|P| = 264$, $P_e = 0,26$, $P_m = 0,11$, $\rho = 0,65$.

5.3 Análise dos resultados

A Tabela 5.2 mostra as comparações entre as bases selecionadas. Para cada algoritmo da literatura ou proposto, a tabela apresenta: nome da instância ('instância'), melhor valor de índice de silhueta ('melhor'), número de *clusters* encontrados ('k'), o gap para entre os valores encontrados e o melhor resultado da literatura ('gap') e o tempo para encontrar o melhor valor do índice silhueta ('ttb'). Destaca-se que gap é a diferença entre o valor real e o valor previsto de algo, neste caso, o valor real trata-se do valor de índice silhueta encontrado e o valor previsto é o índice silhueta da literatura. Ainda sobre gap, quanto mais próximo de 0% significa que o valor real está mais próximo do valor previsto. Para cada instância, foram realizadas 10 execuções com 30 segundos de tempo limite para cada execução. Com exceção da instância `1500p20c, 1500p6c1, 1500p6c, 2000p11c, 2000p9c1`, que por possuir uma dimensão superior às demais, foi executada com tempo limite superior. Os resultados da literatura foram obtidos diretamente dos artigos em que os algoritmos foram propostos. Colunas ausentes indicam que os autores não disponibilizaram a informação. Os algoritmos

AECBL e AECBL1 foram executados 5x, o MRDBSCAN e CLUES foram executados 10x.

Como mostra a Tabela 5.2, a estratégia de seleção roleta obteve os melhores resultados dentre as estratégias de seleção propostas. A roleta alcançou um gap melhor que a literatura em 19 das 55 instâncias, totalizando um percentual de 34.5%. Na comparação total, o gap médio foi de 0,12%, que demonstra que o método é promissor para o problema de agrupamento automático.

A comparação da literatura com a estratégia r-random apresenta bons resultados em comparação com os literatura, em 19 casos o algoritmo obteve melhor desempenho do que os valores encontrados na literatura. Mesmo tendo um bom desempenho, visto na tabela acima, o algoritmo r-random apresentou um gap de 2,45 %, ou seja, não possui uma robustez desejável.

Na estratégia Torneio, os valores encontrados e analisados apresentam um gap de 0.91% e um ganho superior a literatura em 20 instâncias. Já a estratégia dynamic r-random apresenta ganho em 19 instâncias e um gap de 0.50%. O que corrobora o fato de ser uma proposta viável. Para a estratégia S.A, os resultados apresentam um gap de 0.14% e um melhore resultado em 20 instâncias. A proposta utilizando o algoritmo chamado de Pontas traz um gap de 1.28% e um melhor resultado em 19 instâncias. Por fim, a solução Roleta pela média apresenta um gap de 0,25% e um melhor resultado em 20 instâncias.

Para todas as estratégias apresentadas, a comparação com a literatura mostra que o algoritmo MultPop-BRKGA-PCA pode obter bons resultados tento em qualidade de solução quanto em tempo computacional. Todas as estratégias apresentam similaridade em seus ganhos, se comparado a literatura. No final da Tabela 5.2 é apresentada a média dos gaps, em que se pode observar as qualidades das soluções. E a média dos tempos para as melhores soluções (coluna 'ttb' — *time to best*). Observa-se que o tempo médio para se encontrar o melhor valor é inferior a 30 segundos, exceto nas estratégias Pontas e Torneio.

Apesar de obter um resultado médio superior, foi observado que em algumas instâncias, há um baixo desempenho do MultPop-BRKGA-PCA. Este baixo desempenho ocorre, principalmente em instâncias com número elevado de objetos.

Observa-se que em 12 instâncias, novas soluções (desconhecidas pela literatura) foram encontradas. Estas instâncias são: `100p3c1`, `100p5c1`, `100p7c1`, `200p12c1`, `200p7c1`, `300p13c1`, `300p3c1`, `300p3c1`, `300p4c1`, `400p17c1`, `400p4c1`, `500p6c1` e `800p10c1`. Para estas instâncias, o MultPop-BRKGA-PCA encontrou novas soluções com o gap variando de -9.78% a -34.54%. O que indica que o algoritmo proposto consegue melhorar a solução encontrada pela literatura em até 34%.

Como cada trabalho utiliza um conjunto diferente de instâncias, a comparação entre os gaps não é o ideal. Dessa forma, a Tabela 5.3 apresenta um comparação entre cada um dos 4 trabalhos da literatura vs. cada versão do MultPop-BRKGA-PCA utilizando o mesmo conjunto de instâncias do trabalho da literatura. Na tabela 5.3, as colunas têm os seguintes significados: a coluna *Algoritmo* apresenta os nomes dos algoritmos da literatura utilizada para

Tabela 5.2: Resultados Finais

LITERATURA			AECBL				CLUES				MRDBSCAN			ILS-DBSCAN		ROLETA				*r*-RANDOM				TORNEIO				DYNAMIC *r*-RANDOM				S.A				PONTAS				ROL. MÉDIA			
instância	melhor	k	Silh	k	gap(%)	ttb	Silh	k	gap(%)		Silh	k	gap(%)	Silh	gap(%)	Silh	k	gap(%)	ttb	Silh	k	gap(%)	ttb	Silh	k	gap	ttb	Silh	k	gap(%)	ttb	Silh	k	gap(%)	ttb	Silh	k	gap	ttb	Silh	k	gap(%)	ttb
1000p14c	0,831	11	0,831	14	0	51,6	0,767	11	7,7		0,808	15	2,7	0,831	0	0,721	17	11,46	10	0,719	17	13,45	9,44	0,718	17	13,54	80,16	0,721	19	13,21	19,67	0,721	18	13,21	15,67	0,721	21	13,21	30	0,721	24	13,21	10,17
1000p27c1	0,563	14	0,516	24	8,4	67,4	0,563	14	0		-0,293	3	152	0,523	7,12	0,552	5	1,83	10,6	0,551	5	2,1	84,22	0,552	5	2,02	81	0,551	5	2,2	7,33	0,552	5	1,94	15,5	0,551	5	2,2	30	0,552	5	1,89	10,67
1000p5c1	0,643	7	0,643	5	0	38	0,558	7	13,2		0,164	2	74,5	0,639	0,62	0,65	7	-1,51	14,33	0,652	7	-1,43	31,67	0,651	7	-1,22	80	0,65	7	-1,04	19	0,652	7	-1,37	15,5	0,646	7	-0,39	30	0,65	7	-1,04	14,17
1000p6c	0,736	6	0,736	6	0	38,3	0,736	6	0		0,736	6	0	0,736	0	0,695	9	5,23	14,33	0,696	9	5,33	32	0,697	9	5,3	80	0,694	9	5,63	12	0,694	9	5,63	9,33	0,694	9	5,69	30	0,695	9	5,58	13,5
100p10c	0,834	10	0,834	10	0	2,7	0,834	10	0		0,691	8	17,1	0,834	0	0,748	10	10,22	0,83	0,748	10	10,22	1	0,73	10	12,43	< 1	0,748	10	10,22	0,16	0,748	10	10,22	0,33	0,716	10	14,13	15	0,748	10	10,22	0,83
100p2c1	0,743	5	-	-	-		-	-	-		-	-	-	0,743	0	0,791	5	-6,54	< 1	0,791	5	-6,55	0,33	0,791	5	-6,55	< 1	0,791	5	-6,55	< 1	0,791	5	-6,55	< 1	0,791	5	-6,55	15	0,791	5	-6,55	< 1
100p3c1	0,597	3	0,58	3	2,8	1,8	0,597	3	0		0,104	5	82,6	0,579	2,95	0,731	6	-22,44	< 1	0,731	6	-22,45	9,16	0,731	6	-22,45	< 1	0,731	6	-22,45	< 1	0,731	6	-22,45	< 1	0,731	6	-22,45	16	0,731	6	-22,45	< 1
100p3c	0,786	3	0,786	3	0	1,6	0,786	3	0		0,786	3	0	0,786	0	0,679	8	13,54	1,66	0,679	8	13,55	8,5	0,675	7	14,04	< 1	0,679	10	13,55	0,33	0,679	8	13,55	3,33	0,67	8	14,73	15	0,679	8	13,55	1,67
100p5c1	0,703	6	0,696	7	1,1	3,1	0,703	6	0		0,424	2	39,7	0,7	0,48	0,787	10	-11,86	< 1	0,787	10	-11,86	1,17	0,77	10	-9,44	< 1	0,787	10	-11,86	0,67	0,787	10	-11,86	0,17	0,763	10	-8,44	19	0,787	10	-11,86	< 1
100p7c1	0,551	7	0,491	27	10,9	4,1	0,551	7	0		-0,012	2	102,2	-	-	0,699	10	-26,75	0,83	0,699	10	-26,75	4	0,68	10	-23,33	< 1	0,699	10	-26,75	0,67	0,699	10	-26,75	0,83	0,699	10	-26,75	10	0,699	10	-26,75	1
100p7c	0,834	7	0,834	7	0	3,7	0,834	7	0		0,834	7	0	0,834	0	0,748	10	10,25	0,83	0,748	10	10,24	1	0,73	10	12,45	< 1	0,748	10	10,24	0,17	0,748	10	10,24	0,33	0,716	10	14,15	16	0,748	10	10,24	0,83
1100p6c1	0,685	6	-	-	-		0,685	6	0		-	-	-	0,671	2	0,677	10	1,12	13,5	0,673	10	1,66	32	0,673	10	1,66	104	0,673	10	1,76	8,5	0,673	10	1,76	< 1	0,673	10	1,76	30	0,677	10	1,19	17,83
1300p17c	0,823	15	-	-	-		0,738	15	10,3		-	-	-	0,823	0	0,696	22	13,05	33	0,598	22	27,29	31,67	0,701	22	14,79	159	0,696	22	15,41	28	0,696	24	15,41	< 1	0,696	24	15,41	35	0,696	25	15,41	< 1
1500p20c	0,823	13	-	-	-		0,687	13	16,5		-	-	-	0,644	21,77	0,677	20	14,83	46,83	0,527	20	35,95	30,83	0,693	20	15,77	223	0,677	22	17,72	48,33	0,677	18	17,72	< 1	0,677	24	17,72	49	0,677	26	17,72	< 1
1500p6c1	0,66	6	-	-	-		0,66	6	0		-	-	-	-	-	0,653	12	0,32	46,5	0,572	12	13,29	31,67	0,655	12	0,74	223	0,654	12	0,87	49,17	0,654	11	0,85	16	0,653	12	0,98	49	0,653	12	0,98	< 1
1500p6c	0,694	5	-	-	-		0,685	5	1,4		-	-	-	-	-	0,653	12	5,26	46,5	0,572	12	17,59	31,67	0,655	12	5,66	223	0,654	12	5,78	49,17	0,654	11	5,76	16	0,653	12	5,89	49	0,653	12	5,89	< 1
1800p22c	0,804	18	-	-	-		0,643	18	20		-	-	-	0,802	0,2	0,668	26	16,65	24,66	0,534	26	33,61	35,33	0,663	26	17,47	356	0,668	27	16,91	78,5	0,668	22	16,91	12,67	0,668	28	16,91	77	0,668	36	16,91	25
2000p11c	0,713	8	-	-	-		0,606	8	15		-	-	-	0,713	0	0,632	11	11,01	96	0,514	11	27,83	35	0,633	11	11,27	461	0,632	11	11,3	51,67	0,632	11	11,3	49	0,632	12	11,3	100	0,632	12	11,3	< 1
2000p9c1	0,623	7	-	-	-		0,535	7	14,1		-	-	-	0,624	-0,16	0,632	11	-1,83	96	0,514	11	17,42	35	0,633	11	-1,54	462	0,632	11	-1,5	51,67	0,632	11	-1,5	49	0,632	12	-1,5	100	0,632	12	-1,5	< 1
200DATA	0,823	3	0,823	3	0	2,8	0,462	3	43,9		0,823	3	0	-	-	0,823	3	0	20	0,823	3	-0,01	1	0,823	3	-0,01	1	0,823	3	-0,01	< 1	0,823	3	-0,01	< 1	0,823	3	-0,01	30	0,823	3	-0,01	< 1
200p12c1	0,575	9	0,57	8	0,9	8,5	0,562	9	2,3		0,403	3	29,9	0,577	-0,3	0,758	14	-31,76	9,83	0,758	14	-31,76	20,5	0,726	12	-26,23	1	0,758	14	-31,75	2,5	0,758	13	-31,76	11,17	0,717	14	-24,68	30	0,758	14	-31,76	14,33
200p2c1	0,764	3	0,764	2	0	3,1	0,591	3	22,7		0,624	2	18,3	0,764	0	0,673	6	12	2,66	0,673	6	11,99	7	0,673	6	12	1	0,673	6	11,99	1,5	0,673	6	11,99	4	0,673	6	11,99	30	0,673	6	11,99	3
200p3c1	0,68	3	0,68	3	0	3,9	0,674	3	0,8		0,648	2	4,7	-	-	0,733	7	-7,86	3,5	0,733	7	-7,86	7,17	0,733	7	-7,78	1	0,733	7	-7,86	0,83	0,733	7	-7,86	3,5	0,733	7	-7,86	30	0,733	7	-7,86	4,33
200p4c1	0,754	4	0,754	4	0	3,4	0,754	4	0		0,622	3	17,6	0,745	1,25	0,582	12	23,78	14,5	0,58	12	23,06	23,33	0,57	9	24,46	1	0,581	14	23	1,67	0,582	7	22,89	21	0,57	9	24,42	30	0,582	14	22,89	13,5
200p4c	0,773	4	0,773	4	0	3,6	0,773	4	0		0,773	4	0	0,773	0	0,629	12	18,98	11,5	0,629	12	18,56	22,83	0,619	9	19,85	1	0,629	12	18,56	1,5	0,629	8	18,56	10,5	0,625	12	19,15	30	0,629	13	18,56	10
200p7c1	0,576	9	0,576	8	0	8,8	0,555	9	3,6		0,392	3	31,9	0,579	-0,54	0,765	13	-32,78	6,83	0,765	13	-32,79	15,67	0,729	13	-26,51	1	0,765	13	-32,79	1,33	0,765	12	-32,79	4,67	0,753	14	-30,68	30	0,765	14	-32,79	7
300p10c1	0		-	-	-		-	-	-		-	-	0	0,609	-	0,667	12	0	26	0,665	12	0	17,67	0,648	12	0	3	0,667	12	0	1	0,667	4	0	26,33	0,648	4	0	30	0,667	16	0	26
300p13c1	0,594	9	-	-	-		-	-	-		-	-	0	0,566	4,78	0,74	13	-23,69	16,66	0,739	13	-24,3	23,33	0,706	12	-18,82	2	0,723	13	-21,66	2,17	0,74	13	-24,54	15	0,705	15	-18,58	30	0,74	15	-24,54	16,83
300p2c1	0,776	5	0,776	2	0	4,7	0,49	5	36,9		0,62	4	20,1	0,776	0	0,607	6	21,84	16,16	0,607	6	21,84	14	0,607	6	21,86	3	0,607	6	21,84	1,83	0,607	6	21,84	17,83	0,607	6	21,84	30	0,607	6	21,84	14,33
300p3c1	0,677	4	0,677	3	0	4,4	0,592	3	12,5		0,639	2	5,6	0,676	0,12	0,743	7	-9,77	7,66	0,743	7	-9,75	13,83	0,743	7	-9,75	3	0,743	7	-9,75	1,5	0,743	7	-9,75	7	0,743	7	-9,73	30	0,743	7	-9,75	7
300p3c	0,766	3	0,766	3	0	9,2	0,552	3	28		0,766	3	0	0,766	0	0,644	6	15,97	12,5	0,644	6	15,98	8,17	0,644	6	15,98	3	0,644	6	15,98	1	0,644	6	15,98	15,83	0,644	6	15,98	30	0,644	6	15,98	11
300p4c1	0,592	4	0,592	2	0	4,6	0,592	4	0		0,269	3	54,6	0,607	-2,46	0,797	3	-34,59	9,66	0,797	3	-34,58	9,5	0,797	3	-34,58	3	0,797	3	-34,58	1,5	0,797	3	-34,58	9,33	0,797	3	-34,58	30	0,797	3	-34,58	12,2
300p4c2	-		-	-	-		-	-	-		-	-	-	-	-	0,623	8	0	15,5	0,621	8	0	16,17	0,618	8	0	3	0,622	8	0	17,5	0,622	8	0	30	0,623	9	0	14				
300p6c1	0,664	7	0,661	8	0,4	11,9	0,579	7	12,7		0,548	2	17,4	0,662	0,24	0,657	10	1,33	22,16	0,653	10	1,58	16	0,644	10	2,89	3	0,657	10	0,99	2,5	0,656	10	1,11	21	0,634	16	4,4	30	0,657	13	0,99	22,33
400p17c1	0,552	15	0,514	2	7	20,2	0,552	15	0		0,183	14	66,9	0,513	7,13	0,652	3	-15,19	20,33	0,616	3	-11,54	16,17	0,616	3	-11,54	8	0,655	3	-18,58	0,67	0,648	3	-17,25	24,33	0,616	18	-11,54	30	0,652	20	-18	20,5
400p3c	0,799	3	0,799	3	0	9,6	0,799	3	0		0,799	4	0	-	-	0,68	4	14,87	4,33	0,68	4	14,87	13,17	0,68	4	14,87	8	0,68	4	14,87	0,67	0,68	4	14,87	3,83	0,68	4	14,87	30	0,68	4	14,87	4,17
400p4c1	0,62	4	0,607	4	2,2	12,8	0,62	4	0		0,379	2	38,9	0,602	2,97	0,794	8	-27,92	6,67	0,794	8	-27,91	12,33	0,794	8	-27,91	8	0,789	8	-27,2	1,33	0,794	8	-27,91	8	0,789	8	-27,2	30	0,794	8	-27,91	6,33
500p3c	0,825	3	0,825	3	0	10,1	0,823	3	0,2		0,825	3	0	0,825	0	0,776	5	5,88	5,5	0,776	5	5,88	14,83	0,776	5	5,88	14	0,776	5	5,88	1	0,776	5	5,88	6,83	0,776	5	5,88	30	0,776	5	5,88	7,17
500p4c1	0,66	3	0,66	3	0	9,9	0,515	3	21,9		0,305	2	53,8	0,661	-0,2	0,7	6	-6,08	11	0,7	6	-6,07	16	0,7	6	-6,07	14	0,7	6	-6,07	1	0,7	6	-6,07	12,67	0,7	6	-6,07	30	0,7	7	-6,08	14,17
500p6c1	0,669	6	0,631	6	5,6	16,7	0,669	6	0		0,494	12	26,1	0,63	5,76	0,764	9	-14,23	19	0,764	9	-14,23	18,67	0,764	9	-14,23	14	0,758	9	-13,46	2,5	0,764	9	-14,23	17,17	0,755	9	-12,91	30	0,764	9	-14,23	18,83
600p15c	0,781	16	0,781	15	0	37,1	0,753	16	3,6		0,781	15	0	0,781	0	0,708	16	11,83	21	0,683	16	12,54	24,17	0,692	16	11,46	22	0,674	19	13,76	4,33	0,698	15	10,71	14,83	0,674	22	13,76	30	0,69	23	11,62	22,67
600p3c1	0,721	3	0,721	3	0	9,6	0,703	3	2,5		0,686	2	4,8	0,721	0	0,775	5	-7,49	6,67	0,775	5	-7,48	23	0,775	5	-7,48	22	0,775	5	-7,48	2,5	0,775	5	-7,48	10,33	0,775	5	-7,48	30	0,775	5	-7,48	9,33
700p15c1	0,68	17	0,68	15	0	22,9	0,66	17	3		0,122	2	82,1	0,68	0	0,673	21	-1,03	19,83	0,682	21	-0,19	30,33	0,694	21	-1,94	33	0,672	21	1,2	5,83	0,689	22	-1,21	18,17	0,672	26	1,2	30	0,684	26	-0,48	25,17
700p4c	0,797	4	0,797	4	0	12,9	0,797	4	0		0,797	4	0	0,797	0	0,663	6	16,85	16,16	0,662	6	16,92	30,17	0,662	6	16,94	33	0,662	6	16,99	3,33	0,663	6	16,85	12,17	0,662	6	16,99	30	0,663	6	16,85	15,67
800p10c1	0,507	8	0,478	2	5,7	29,7	0,507	8	0		0,079	2	84,4	0,468	7,71	0,616	18	-23,25	20	0,62	18	-22,3	30	0,616	18	-21,54	46	0,613	18	-20,95	12,67	0,624	18	-23,05	25,33	0,602	20	-18,69	30	0,615	21	-21,29	21,67
800p18c1	0,694	19	0,691	19	0,4	42,2	0,694	19	0		0,265	24	61,8	0,692	0,3	0,664	23	2	26	0,641	23	7,62	30	0,655	23	5,67	46	0,654	23	5,74	12,5	0,668	18	3,77	18,17	0,648	28	6,69	30	0,664	28	4,37	25,5
800p23c	0,787	22	0,787	23	0	47,7	0,739	22	6,1		0,787	23	0	0,787	0	0,662	22	13,93	22,5	0,644	22	18,25	30	0,664	22	15,63	47	0,641	23	18,61	13,67	0,676	21	14,14	17	0,632	26	19,77	30	0,662	28	15,9	23
800p4c1	0,714	4	0,705	4	1,3	15,9	0,714	4	0		0,508	2	28,9	0,702	1,72	0,656	10	7,43	19,83	0,656	10	8,23	30,17	0,656	10	8,19	46	0,658	10	7,91	8,5	0,657	9	8,05	14,83	0,655	11	8,29	30	0,656	11	8,17	20
900p12c	0,841	10	0,841	12	0	63,1	0,798	10	5,1		0,841	12	0	0,841	0	0,721	17	13,27	6,33	0,7	17	16,77	31,33	0,724	17	13,86	63	0,717	17	14,73	11,83	0,729	18	13,32	20,17	0,717	21	14,73	30	0,721	24	14,3	6,33
900p5c	0,716	7	0,716	5	0	24,4	0,613	7	14,4		0,716	5	0	0,716	0	0,666	5	6,93	20,16	0,666	5	7,04	0,33	0,666	5	6,96	63	0,666	5	7,01	14	0,666	5	6,96	15	0,665	5	7,07	30	0,666	5	6,96	16,67
Broken_ring	0,5	5	-	-	-		-	-	-		-	-	-	-	-	0,502	5	-0,72	13,83	0,503	5	-0,63	30	0,503	5	-0,66	46	0,502	5	-0,54	5,5	0,503	5	-0,63	14,67	0,502	5	-0,54	30	0,503	5	-0,66	13,17
iris	0,687	3	0,686	3	0,1	1,3	0,563	3	18		0,687	2	0	0,687	0	0,686	2	0	4,83	0,686	2	0,02	16	0,686	2	0,02	< 1	0,686	2	0,02	0,83	0,686	2	0,02	7,67	0,686	2	0,02	30	0,686	2	0,02	3,83
Maronna	0,575	4	0,575	4	0	4,2	0,575	4	0		0,562	2	2,2	0,575	0	0,575	4	0	3,67	0,575	4	0	5	0,575	4	0	1	0,575	4	0	0,17	0,575	4	0	8	0,575	4	0	30	0,575	4	0	8,33
ruspini	0,738	4	0,738	4	0	0,6	0,738	4	0		0,738	4	0	0,738	0	0,738	4	0	< 1	0,738	4	-0,01	0,67	0,738	4	-0,01	< 1	0,738	4	-0,01	< 1	0,738	4	-0,01	< 1	0,738	4	-0,01	9	0,738	4	-0,01	< 1
Vowel2	0,448	9	0,425	3	5,2	26,8	0,448	9	0		0,417	2	7	0,451	-0,6	0, 421	3	6,02	20,16	0,421	3	6,11	17,83	0,421	3	6,14	15	0,42	3	6,27	2	0,421	3	6,02	24,67	0,422	3	5,91	30	0,421	3	6	20,67
					1.24%	16,64s			7.30%				26.85%		1.37%			0.12%	18.41s			2.47%	19.67s			0.91%	67.64s			0.5%	10.83s			0.14%	14.21s			1.28%	32.25s			0.25%	13.06s

comparação. A coluna *gap* contém o gap de cada solução relatado no trabalho proposto. As demais colunas contêm o gap da referida estratégia considerando apenas as instâncias utilizadas no trabalho em que o algoritmo foi proposto. Um valor em negativo (em negrito) indica que a média das soluções encontradas está melhor que a melhor média da literatura.

Tabela 5.3: Tabela de Resumo

Literatura		MultPop-BRKGA-PCA						
Algoritmo	gap	Roleta	r-rand.	Torneio	dyn. r-ran.	Pontas	S.A	R média
AECBL	1,24%	**-0,69%**	**-0,64%**	**-0,48%**	**-0,74%**	0,00%	0,81%	**-0,09%**
CLUES	7,30%	0,63%	0,67%	0,81%	0,46%	0,97%	1,94%	0,83%
MRDBSCAN	26,85%	**-0,69%**	**-0,64%**	**-0,48%**	**-0,74%**	0,00%	0,81%	**-0,09%**
ILS-DBSCAN	1,37%	0,33%	0,37%	0,60%	0,23%	0,57%	1,64%	0,35%

A Tabela 5.3 mostra que em todos os caso, com exceção do S.A vs. AECBL1, o algoritmo proposto obteve melhores soluções que os algoritmos da literatura — utilizando o mesmo conjunto de instâncias. Em especial, a estratégia dynamic r-random obteve o melhor desempenho com: um gap de -0,74% contra um gap de 1,24% do algoritmo AECBL; um gap de 0,46% conta um gap de 7,30% do algoritmo CLUES; um gap de -0,74% contra um gap de 26,85% do algoritmo MRDBSCAN e um gap de 0,23% contra um gap de 1,37% do algoritmo AECBL1. Estes resultados mostram a eficiência e robustez do MultPop-BRKGA-PCA para o PCA.

Capítulo 6

Considerações finais

Neste trabalho, foram apresentadas soluções para o problema de Clusterização Automática com o índice de silhueta como função objetivo. O método proposto é um meta-heurística multipopulacional, denominada MultPop-BRKGA-PCA, que utiliza o BRKGA para resolver subproblemas de k-clusterização utilizando o índice de silhueta como função objetivo. Foram propostas 7 estratégias de seleção de população para o MultPop-BRKGA-PCA, e uma função de decodificação (BRKGA) para o problema de k-clusterização. As soluções propostas foram validadas em 55 instâncias bem conhecidas da na literatura (Semaan et al., 2020), e comparadas com abordagens como MRDBSCAN (Semaan et al., 2012), AECBL1 (Semaan et al., 2019) , AECBL (Cruz & Ochi, 2011) e CLUES (Wang et al., 2007).

A análise dos experimentos mostra que as soluções propostas obtiveram resultados melhores em um tempo computacional menor que as soluções encontradas na literatura. Em especial, a estratégia dynamic r-random obteve o melhor desempenho com bastante superioridade às demais abordagens da literatura. Estes resultados corroboram a eficiência e robustez do MultPop-BRKGA-PCA para o problema de Clusterização Automática.

Em comparação dos métodos propostos e os algoritmos da literatura, os algoritmos propostos sobressaem, principalmente nas instâncias com menos pontos (objetos), o que leva o entendimento que as soluções propostas apresentam eficiência em tempo curto e para menores unidades de pontos. Isto é importante visto que é possível particionar

Ressalta que em 12 instâncias, novas soluções (desconhecidas pela literatura) foram encontradas. Para estas instâncias, o MultPop-BRKGA-PCA encontrou novas soluções com o gap variando de -9.78% a -34.54%. O que indica que o algoritmo proposto consegue melhorar a solução encontrada pela literatura em até 34%.

Apesar de obter um resultado médio superior, foi observado que em algumas instâncias, há um baixo desempenho do MultPop-BRKGA-PCA. Este baixo desempenho ocorre, principalmente em instâncias com número elevado de objetos.

6.1 Trabalhos futuros

Com relação aos trabalhos futuros, propõem-se:

- Melhoria no processo de clusterização para instancias de número elevado de objetos;
- Novas meta-heurísticas multipopulacionais para o PCA;
- Métodos híbridos com programação matemática e/ou aprendizado de máquina para a solução do PCA;
- Aplicação do algoritmo proposto para outros tipos de problemas de clusterização automática — com outras funções objetivo;
- Novas estratégias de gerenciamento de população.

Referências bibliográficas

Aguirre, H. E., Tanaka, K., Sugimura, T. & Oshita, S. (2000), Improved distributed genetic algorithm with cooperative-competitive genetic operators, *in* 'Smc 2000 conference proceedings. 2000 ieee international conference on systems, man and cybernetics.'cybernetics evolving to systems, humans, organizations, and their complex interactions'(cat. no. 0', Vol. 5, IEEE, pp. 3816–3822.

Ahmadi, S. & Osman, I. H. (2005), 'Greedy random adaptive memory programming search for the capacitated clustering problem', *European Journal of Operational Research* **162**(1), 30–44.

Ahn, C. W. & Ramakrishna, R. S. (2003), 'Elitism-based compact genetic algorithms', *IEEE Transactions on Evolutionary Computation* **7**(4), 367–385.

Andrade, C. E., Toso, R. F., Gonçalves, J. F. & Resende, M. G. (2021), 'The multi-parent biased random-key genetic algorithm with implicit path-relinking and its real-world applications', *European Journal of Operational Research* **289**(1), 17–30.

Assunçao, R. M. & Reis, I. A. (2000), 'Testing spatial randomness: a comparison between t^2 methods and modifications of the angle test', *Brazilian Journal of Probability and Statistics* pp. 71–86.

Baldi, P. & Hatfield, G. W. (2011), *DNA microarrays and gene expression: from experiments to data analysis and modeling*, Cambridge university press.

Banerjee, A. & Dave, R. N. (2004), Validating clusters using the hopkins statistic, *in* '2004 IEEE International conference on fuzzy systems (IEEE Cat. No. 04CH37542)', Vol. 1, IEEE, pp. 149–153.

Bean, J. C. (1994), 'Genetic algorithms and random keys for sequencing and optimization', *ORSA journal on computing* **6**(2), 154–160.

Bezdek, J. C. (1974), 'Numerical taxonomy with fuzzy sets', *Journal of mathematical biology* **1**(1), 57–71.

Cano, J. R., Cordón, O., Herrera, F. & Sánchez, L. (2002), A grasp algorithm for clustering, *in* 'Ibero-American Conference on Artificial Intelligence', Springer, pp. 214–223.

Cho, Y. S., Moon, S. C. & Ryu, K. H. (2014), 'Clustering analysis by customer feature based on som for predicting purchase pattern in recommendation system', *Journal of the Korea Society of Computer and Information* **19**(2), 193–200.

Coutinho, P. H. S., Thiago, P., Cruz, F. d. C., Simas Filho, E. F., Albuquerque, M. C., Silva, I. C. & Farias, C. T. (2015), Reconhecimento de padroes de falhas tipo fenda em chapa de aluminio utilizando clusterizacao fuzzy, *in* 'Anais do 12 Congresso Brasileiro de Inteligencia Computacional', pp. 1–6.

Cruz, M. D. (2010), 'O problema de clusterização automática'.

Cruz, M. D. & Ochi, L. S. (2011), O problema de clusterização automática: Um novo método utilizando ils, *in* 'Anais do X Congresso Brasileiro de Inteligência Computacional (X CBIC), Fortaleza-CE'.

Duda, R. O., Hart, P. E. et al. (1973), *Pattern classification and scene analysis*, Vol. 3, Wiley New York.

El Dor, A., Clerc, M. & Siarry, P. (2012), 'A multi-swarm pso using charged particles in a partitioned search space for continuous optimization', *Computational Optimization and Applications* **53**(1), 271–295.

Ester, M., Kriegel, H.-P., Sander, J., Xu, X. et al. (1996), A density-based algorithm for discovering clusters in large spatial databases with noise., *in* 'Kdd', Vol. 96, pp. 226–231.

Fisher, R. A. (1936), 'The use of multiple measurements in taxonomic problems', *Annals of eugenics* **7**(2), 179–188.

Fontana, E. A. (2017), 'Algoritmos de clusterização aplicados na análise genômica da bactéria escherichia coli'.

Gonçalves, J. F. & Resende, M. G. (2011), 'Biased random-key genetic algorithms for combinatorial optimization', *Journal of Heuristics* **17**(5), 487–525.

Gonçalves, J. F., Resende, M. G. & Toso, R. F. (2014), 'An experimental comparison of biased and unbiased random-key genetic algorithms', *Pesquisa Operacional* **34**(2), 143–164.

Han, J., Kamber, M. & Tung, A. K. (2001), 'Spatial clustering methods in data mining', *Geographic data mining and knowledge discovery* pp. 188–217.

Hansen, P. & Jaumard, B. (1997), 'Cluster analysis and mathematical programming', *Mathematical programming* **79**(1-3), 191–215.

Hartuv, E. & Shamir, R. (2000), 'A clustering algorithm based on graph connectivity', *Information processing letters* **76**(4-6), 175–181.

Hastie, T., Tibshirani, R. & Friedman, J. (2009), *The elements of statistical learning: data mining, inference, and prediction*, Springer Science & Business Media.

Holland, J. H. (1992), 'Genetic algorithms', *Scientific american* **267**(1), 66–73.

Hosseini, S. M. S., Maleki, A. & Gholamian, M. R. (2010), 'Cluster analysis using data mining approach to develop crm methodology to assess the customer loyalty', *Expert Systems with Applications* **37**(7), 5259–5264.

Hruschka, E. R. & Covoes, T. F. (2005), Feature selection for cluster analysis: an approach based on the simplified silhouette criterion, *in* 'International Conference on Computational Intelligence for Modelling, Control and Automation and International Conference on Intelligent Agents, Web Technologies and Internet Commerce (CIMCA-IAWTIC'06)', Vol. 1, IEEE, pp. 32–38.

Humby, C. (2006), 'Data is the new oil', *Proc. ANA Sr. Marketer's Summit. Evanston, IL, USA*.

Ishibuchi, H. & Murata, T. (1996), Multi-objective genetic local search algorithm, *in* 'Proceedings of IEEE international conference on evolutionary computation', IEEE, pp. 119–124.

Iwayama, M. & Tokunaga, T. (1995), Cluster-based text categorization: a comparison of category search strategies, *in* 'Proceedings of the 18th annual international ACM SIGIR conference on Research and development in information retrieval', pp. 273–280.

Kaufman, L. & Roussew, P. (1990), 'Finding groups in data-an introduction to cluster analysis. a wiley-science publication john wiley & sons'.

Kim, H.-J., McGuire, D. B., Tulman, L. & Barsevick, A. M. (2005), 'Symptom clusters: concept analysis and clinical implications for cancer nursing', *Cancer nursing* **28**(4), 270–282.

Kirkpatrick, S., Gelatt, C. D. & Vecchi, M. P. (1983), 'Optimization by simulated annealing', *Science* **220**(4598), 671–680.

Lipowski, A. & Lipowska, D. (2012), 'Roulette-wheel selection via stochastic acceptance', *Physica A: Statistical Mechanics and its Applications* **391**(6), 2193–2196.

Liu, H.-H. & Ong, C.-S. (2008), 'Variable selection in clustering for marketing segmentation using genetic algorithms', *Expert Systems with Applications* **34**(1), 502–510.

Lourenço, H. R., Martin, O. C. & Stützle, T. (2003), Iterated local search, *in* 'Handbook of metaheuristics', Springer, pp. 320–353.

López-Ibáñez, M., Dubois-Lacoste, J., Pérez Cáceres, L., Birattari, M. & Stützle, T. (2016), 'The irace package: Iterated racing for automatic algorithm configuration', *Operations Research Perspectives* **3**, 43–58.

Ma, J. & Song, W. (2013), 'Automatic clustering method of abnormal crowd flow pattern detection', *Procedia engineering* **62**, 509–518.

MacQueen, J. et al. (1967), Some methods for classification and analysis of multivariate observations, *in* 'Proceedings of the fifth Berkeley symposium on mathematical statistics and probability', Vol. 1, Oakland, CA, USA, pp. 281–297.

Maronna, R. & Jacovkis, P. M. (1974), 'Multivariate clustering procedures with variable metrics', *Biometrics* pp. 499–505.

Martinelli, G. & Eidsvik, J. (2014), 'Dynamic exploration designs for graphical models using clustering with applications to petroleum exploration', *Knowledge-Based Systems* **58**, 113–126.

Mirkin, B. (1999), 'Concept learning and feature selection based on square-error clustering', *Machine Learning* **35**(1), 25–39.

Naldi, A., Remy, E., Thieffry, D. & Chaouiya, C. (2011), 'Dynamically consistent reduction of logical regulatory graphs', *Theoretical Computer Science* **412**(21), 2207–2218.

Neath, A. A. & Cavanaugh, J. E. (2012), 'The bayesian information criterion: background, derivation, and applications', *Wiley Interdisciplinary Reviews: Computational Statistics* **4**(2), 199–203.

Ochi, L. S., Dias, C. R. & Soares, S. S. F. (2004), 'Clusterização em mineração de dados', *Instituto de Computação-Universidade Federal Fluminense-Niterói* p. 26.

Osman, I. H. & Kelly, J. P. (1997), 'Meta-heuristics theory and applications', *Journal of the Operational Research Society* **48**(6), 657–657.

Pelleg, D., Moore, A. W. et al. (2000), X-means: Extending k-means with efficient estimation of the number of clusters., *in* 'Icml', Vol. 1, pp. 727–734.

Punj, G. & Stewart, D. W. (1983), 'Cluster analysis in marketing research: Review and suggestions for application', *Journal of marketing research* **20**(2), 134–148.

Ray, S. & Turi, R. H. (1999), Determination of number of clusters in k-means clustering and application in colour image segmentation, *in* 'Proceedings of the 4th international conference on advances in pattern recognition and digital techniques', Citeseer, pp. 137–143.

Resende, M. G. (2011), 'Introdução aos algoritmos genéticos de chaves aleatórias viciadas', *Simpósio Brasileiro de Pesquisa Operacional* pp. 3680–3691.

Resende, M. G. (2014), 'Packing with biased random-key genetic algorithms'.

Reutterer, T. & Dan, D. (2020), 'Cluster analysis in marketing research'.

Roses, C. F. & Leis, R. P. (2002), 'Um estudo das condições sócio-econômicas de municípios gaúchos através da análise de cluster', *Revista Administração On Line*.

Rousseeuw, P. J. (1987), 'Silhouettes: a graphical aid to the interpretation and validation of cluster analysis', *Journal of computational and applied mathematics* **20**, 53–65.

Semaan, G. S., Cruz, M. D., Brito, G. d. M. & Ochi, L. S. (2012), 'Proposta de um método de classificação baseado em densidade para a determinação do número ideal de grupos em problemas de clusterização', *Journal of the Brazilian Computational Intelligence Society* **10**(4), 242–262.

Semaan, G. S., Fadel, A. C. & de Moura Brito, J. A. (2020), 'Heuristics applied to minimization of the maximum-diameter clustering problem', *IEEE Latin America Transactions*.

Semaan, G. S., Fadel, A. C., de Moura Brito, J. A. & Ochi, L. S. (2019), 'Uma eficiente heurística híbrida com estatística de hopkins para o problema de agrupamento automático', *IEEE Latin America Transactions* **17**(1), 7–17.

Semaan, G. S., Vasconcelos, R. B., Brito, J. A. d. M. & Ochi, L. S. (2014), 'Proposta de um método baseado em densidade e grade para o problema de agrupamento automático', *XVII Simpósio de Pesquisa Operacional e Logística da Marinha. Ed. Edgard Blücher*.

Shi, J. & Malik, J. (2000), 'Normalized cuts and image segmentation', *IEEE Transactions on pattern analysis and machine intelligence* **22**(8), 888–905.

Srinivas, M. & Patnaik, L. M. (1994), 'Genetic algorithms: A survey', *computer* **27**(6), 17–26.

Steinbach, M., Karypis, G. & Kumar, V. (2000), 'A comparison of document clustering techniques'.

Tan, P.-N., Steinbach, M. & Kumar, V. (2013), 'Data mining cluster analysis: basic concepts and algorithms', *Introduction to data mining* pp. 487–533.

Toledo, C. F. M., De Oliveira, R. R. R. & França, P. M. (2013), 'A hybrid multi-population genetic algorithm applied to solve the multi-level capacitated lot sizing problem with backlogging', *Computers & Operations Research* **40**(4), 910–919.

Trojanowski, K. & Wierzchoń, S. T. (2003), Studying properties of multipopulation heuristic approach to non-stationary optimisation tasks, *in* 'Intelligent Information Processing and Web Mining', Springer, pp. 23–32.

Tseng, L. Y. & Yang, S. B. (2001), 'A genetic approach to the automatic clustering problem', *Pattern recognition* **34**(2), 415–424.

Wang, X., Qiu, W. & Zamar, R. H. (2007), 'Clues: A non-parametric clustering method based on local shrinking', *Computational Statistics & Data Analysis* **52**(1), 286–298.

Yonamine, F. S., Specia, L., Carvalho, V. & Nicoletti, M. (2002), 'Aprendizado não supervisionado em domínios fuzzy–algoritmo fuzzy c-means', *São Carlos: UFSCAR*.