

UNIVERSIDADE DE SÃO PAULO
Instituto de Ciências Matemáticas e de Computação
ISSN 0103-2569

Técnicas de Agrupamento e Algoritmos Genéticos

Murilo Coelho Naldi
André C.P.L.F. de Carvalho

Nº 267

RELATÓRIOS TÉCNICOS



São Carlos – SP
Nov./2005

SYSNO	1478799
DATA	/ /
ICMC - SBAB	

Técnicas de Agrupamento e Algoritmos Genéticos

Murilo Coelho Naldi
André C.P.L.F. de Carvalho

Universidade de São Paulo
Instituto de Ciências Matemáticas e de Computação
Departamento de Ciências de Computação e Estatística
Laboratório de Inteligência Computacional
Caixa Postal 668, 13560-970 - São Carlos, SP, Brasil
e-mail: {murilocn, andre}@icmc.usp.br

Resumo

Os algoritmos de agrupamento têm por objetivo reunir uma série de objetos ou instâncias em grupos, ou clusters, de objetos semelhantes ou relacionados. Desta forma, esses algoritmos fornecem um meio de explorar e analisar estruturas presentes nos dados. Porém, muitos desses algoritmos são determinísticos (*hill-climbing*) e tendem à encontrar soluções em mínimos locais, não necessariamente o mínimo global. Além disso, uma mesma técnica de agrupamento utilizada em um mesmo conjunto de dados pode resultar em diferentes formas de agrupar esses dados (Zeng et al. 2002). Abordagens evolutivas têm se mostrado muito eficientes para a obtenção de soluções globais e têm obtido bons resultados para problemas de agrupamento (Jain et al. 1999). Neste relatório será apresentado um estudo de diferentes métodos que combinam Algoritmos Genéticos (GA) com algoritmos de agrupamento

Palavras-Chave: Agrupamento, Algoritmos Genéticos, Clustering.

Novembro 2005
Draft impresso em 31 de outubro de 2005

Sumário

1	Introdução	1
2	Algoritmos de Agrupamento	1
3	Algoritmos Genéticos	4
4	Principais Aspectos do Agrupamento Evolutivo	7
4.1	Representação Utilizada	7
4.2	Função de Aptidão	11
4.3	Operadores Genéticos	14
4.3.1	Inicialização	14
4.3.2	Seleção	15
4.3.3	Cruzamento	16
4.3.4	Mutação	18
5	Exemplo de Agrupamento em conjunto com GAs	20
5.1	K-means em conjunto com GAs	20
5.2	Redes SOM em conjunto com GAs	29
5.3	Agrupamentos Hierárquicos	35
6	Considerações Finais	42
	Referências	46

Lista de Figuras

1	Exemplo de dendograma	2
2	Visão Geral de um Algoritmo Genético	4
3	Exemplo de Cruzamento Simples	5
4	Exemplo de Mutação	6
5	Cromossomos representando o agrupamento $\{\{A, C, F\}, \{B, D, E\}\}$ utilizando as representações: (a)Grupo-Número (b)Matriz (c)Permutação com separador (d) Permutação Gulosa	8
6	Cromossomos representando uma matriz contendo centróides	9
7	Cromossomo com número variável de subconjuntos	9
8	Exemplo de cromossomo para agrupamento hierárquico	10
9	Exemplo de cromossomo para redes SOM	10
10	Cromossomo com genes representando os parâmetros P1, P2, P3	11
11	Agrupamentos e seus representações Grupo-Número.	17
12	Intersecções não vazias e um padrão de cromossomo filho.	17
13	Cruzamento de cromossomos com número indeterminado de subconjuntos	18

Lista de Tabelas

1	Ilustração do método da Roleta	5
2	Resumo do GA utilizado por Bezdek	22
3	Resumo do GA utilizado por Murthy	22
4	Resumo do Algoritmo COWCLUS	23
5	Resumo do Algoritmo GGA	25
6	Resumo do Algoritmo KGACCLUS	26
7	Resumo do Algoritmo HGACCLUS	27
8	Resumo do Algoritmo SAGA	29
9	Resumo do Algoritmo CGA	30
10	Resumo do GA utilizado por Tanaka	31
11	Resumo do GA utilizado por Ohtsuka	33
12	Resumo do algoritmo ISOM	34
13	Resumo do algoritmo GASOM	35
14	Resumo do Algoritmo GSOMC	36
15	Resumo do Algoritmo CLUSTERING	37
16	Resumo do Algoritmo utilizado por Greene	39
17	Resumo do Algoritmo utilizado por Ma	40
18	Resumo do Algoritmo GCA	42

1 Introdução

Algoritmos de agrupamento são ferramentas valiosas na análise exploratória dos dados, mineração de dados e reconhecimento de padrões. Tais algoritmos fornecem um meio de explorar e verificar estruturas presentes nos dados, organizando-os em agrupamentos. Eles utilizam o aprendizado dirigido aos dados, não requerendo conhecimento prévio sobre as suas classes ou categorias (Mitchell 1997), ou seja aprendizado não supervisionado. Tal característica é vantajosa para o estudo de problemas com pouco conhecimento prévio sobre as possíveis classes existentes nos dados (Zeng et al. 2002).

Fazer com que uma técnica de agrupamento consiga obter partições próximas da ótima é uma tarefa difícil de ser executada empiricamente. Isso ocorre porque uma técnica de agrupamento pode resultar em diferentes formas de agrupar os dados (Zeng et al. 2002), devido à utilização de diferentes valores de seus parâmetros livres ou a escolha das partições iniciais. Por serem métodos de busca global, Algoritmos Genéticos (*Genetic Algorithms* - GAs) podem ser utilizados para otimizar os valores dos parâmetros livres utilizados pelos algoritmos de agrupamento. Estes algoritmos são métodos de busca baseados no processo evolutivo de populações dos seres vivos e seguem o princípio da seleção natural e sobrevivência dos mais aptos (Beasley et al. 1993).

Esse relatório técnico é dividido da seguinte maneira: na Seção 2 são apresentados alguns conceitos básicos a respeito de algoritmos de agrupamento. A Seção 3 descreve uma visão geral de GAs. A Seção 4 contém definições importantes e os aspectos principais de algumas técnicas que utilizam agrupamentos em conjunto com GAs. Na Seção 5 são discutidos alguns detalhes de implementações que utilizaram GAs em conjunto com algoritmos de agrupamento. As considerações finais são apresentadas na Seção 6.

2 Algoritmos de Agrupamento

Nessa seção é feita uma introdução sobre algoritmos de agrupamento e são apresentados os principais conceitos relacionados aos mesmos.

Agrupamento é a classificação não-supervisionada de padrões (observações, instâncias, exemplos, itens de dados ou vetores de características) em grupos (*subconjuntos*) (Jain et al. 1999). O objetivo dessa classificação é obter grupos que apresentem padrões semelhantes e possam refletir a forma como os dados são estruturados.

De maneira intuitiva, um agrupamento é formado a partir de um princípio indutivo. Um princípio indutivo, também conhecido como critério de agrupamento ou função objetivo, consiste em uma forma de selecionar um modelo ou estrutura para representar os agrupamentos que melhor se ajuste a um determinado conjunto de dados. A associação de um princípio indutivo à um conjunto de dados gera um problema de otimização. Na maioria das vezes, esses problemas são intratáveis ou apresentam uma complexidade muito elevada para grandes conjuntos de dados (Estivill-Castro 2002).

Os algoritmos de agrupamento se propõem a solucionar esse problema de otimização. Eles classificam o conjunto de dados de maneira não-supervisionada, ou seja, eles dividem os dados em conjuntos de classes, sem a necessidade de informações prévias sobre esses

dados, a não ser para validação. Para isso, esses algoritmos definem uma medida de proximidade e um método de busca que permitem encontrar partições dos dados de acordo com um critério de agrupamento (Jiang et al. 2003). Desta forma, o agrupamento é apropriado para explorar as relações entre os dados e avaliar sua estrutura.

A maioria dos algoritmos de agrupamento podem ser divididos em exclusivo e não exclusivo (Jain et al. 1999). Um agrupamento exclusivo é uma partição de um conjunto de objetos onde cada objeto pertence exclusivamente a um único subconjunto, ou seja, um exemplo pertence ou não pertence a um agrupamento. Um agrupamento não exclusivo pode associar um objeto a vários subconjuntos, onde cada exemplo tem um grau de pertinência a cada um dos subconjuntos. Neste trabalho serão estudados algoritmos de agrupamento exclusivos.

Além de relação de exclusividade dos objetos, os algoritmos de agrupamento também podem ser divididos em hierárquicos e particionais, de acordo com a estrutura em que os dados são particionados. A estrutura de um algoritmo hierárquico resulta em uma seqüência aninhada de partições, ou seja, é uma forma de agrupar os dados em que dois exemplos agrupados em um nível continuam fazendo parte do mesmo grupo nos níveis mais altos, formando um dendograma, ilustrado na Figura 1. Os agrupamentos hierárquicos podem ser divididos em duas abordagens: a aglomerativa e a divisiva. A abordagem aglomerativa começa com n subconjuntos com um único exemplo cada e forma a seqüência de partições agrupando os subconjuntos sucessivamente. A abordagem divisiva começa com um subconjunto com todos os exemplos e forma a seqüência dividindo os subconjuntos sucessivamente.

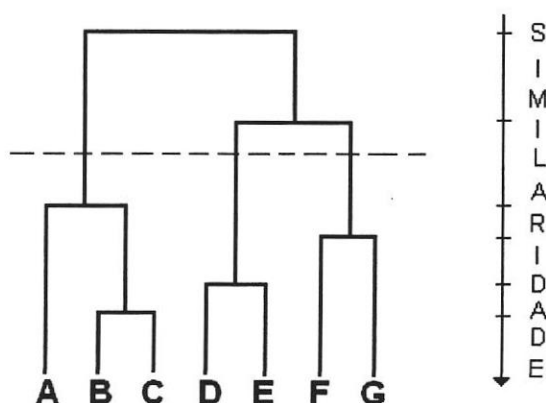


Figura 1: Exemplo de dendograma

Os algoritmos particionais resultam em uma única partição dos dados. Um algoritmo particional exclusivo tem como objetivo dividir um conjunto com N objetos $X = x_1, x_2, \dots, x_N$ em uma coleção de k subconjuntos distintos entre si $C = C_1, C_2, \dots, C_k$, de maneira que:

$$C_1 \cup C_2 \cup \dots \cup C_k = X, C_i \neq \emptyset, \text{ e } C_i \cap C_j = \emptyset \text{ para } i \neq j.$$

Como maioria dos trabalhos que relacionam técnicas de agrupamento em conjunto com GAs utiliza algoritmos particionais exclusivos (Bezdek et al. 1994; Tanaka et al. 1996; Cole 1998; Jain et al. 1999; Kivijärvi et al. 2003), portanto será dada maior ênfase nesse tipo de algoritmo no trabalho. Entre os vários algoritmos de agrupamento particionais, podem ser citados os algoritmos *k-means* (Duda et al. 2001) e o Mapa Auto-Organizável (*Self Organizing Map* - SOM) (Haykin 1999). Essas técnicas serão descritas em detalhes na Seção 5.

Muitos desses algoritmos assumem um número k de agrupamentos definido pelo usuário. Esses algoritmos utilizam algum critério pré-definido para agrupar os objetos. Desta forma, o número de maneiras que os N objetos podem ser classificados em k agrupamentos é definido pela Equação 1.

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

Portanto, encontrar a melhor partição possível não é uma tarefa fácil, mesmo conhecendo o número de agrupamentos k . Porém, esse número raramente é conhecido na prática. É comum executar um algoritmo de agrupamento várias vezes e, baseado nos resultados, escolher um valor para k que obtenha um agrupamento mais natural possível (Jain et al. 1999). Uma outra alternativa consiste em otimizar k de acordo com um critério numérico, ou seja, efetuar o agrupamento para um grupo de valores, ou cenários, de k . Considerando n cenários distintos, o número de maneiras possíveis de se agrupar N objetos é dado pela Equação 2.

$$\sum_{i=1}^n NW(N, k) \quad (2)$$

As Equações 1 e 2 mostram que encontrar uma partição ótima é um problema de complexidade *NP-completo*. Portanto, tentar encontrar uma solução que represente um ótimo global pode ser uma tarefa computacionalmente inviável. Na tentativa de reduzir o custo computacional, diversos tipos de algoritmos de agrupamentos vêm sendo desenvolvidos (Ward 1963; Duda et al. 2001; Haykin 1999). Porém, muitos desses algoritmos são determinísticos (*hill-climbing*) e tendem à encontrar soluções em mínimos locais, não necessariamente o mínimo global.

Além disso, uma mesma técnica de agrupamento utilizada em um mesmo conjunto de dados pode resultar em diferentes formas de agrupar esses dados (Zeng et al. 2002), devido à utilização de diferentes valores de seus parâmetros livres. Assim, a obtenção de um bom agrupamento pode ser visto como um processo de otimização, que procura encontrar o melhor conjunto de valores para os parâmetros. A maioria das abordagens de agrupamento é baseada em alguma heurística e apenas pode fornecer uma aproximação do resultado ótimo. O desempenho de uma técnica depende das características dos dados utilizados, da seleção do estado inicial e até mesmo da ordem de apresentação dos dados, o que faz com que não exista uma abordagem mais apropriada do que as outras (Zeng et al. 2002).

Abordagens evolutivas têm se mostrado muito eficientes para a obtenção de soluções

globais e tem obtido bons resultados para problemas de agrupamento (Jain et al. 1999). Dentre elas, podemos destacar GAs, que vêm sendo utilizados com sucesso em diversas abordagens (Cowgill et al. 1998; Jain et al. 1999; Bayram et al. 2004). Esses algoritmos serão descritos com maiores detalhes na Seção 3.

3 Algoritmos Genéticos

GAs são técnicas de busca e otimização utilizadas para gerar variações ou possíveis soluções e combinar as características daquelas que obtiverem melhor desempenho, gerando assim novas soluções com desempenho melhor a cada geração (Balakrishnan & Honavar 1995). Estes algoritmos seguem o princípio da seleção natural e sobrevivência dos mais aptos, proposto por Charles Darwin em seu livro “A origem das espécies” (Beasley et al. 1993).

O primeiro passo na construção de GAs típicos é a geração de uma população inicial de N indivíduos ou cromossomos, que representam possíveis soluções do problema a ser resolvido. Durante o processo evolutivo, essa população é avaliada e cada cromossomo recebe uma nota (aptidão), refletindo a qualidade de sua solução para o problema em questão. Em geral, os cromossomos mais aptos são selecionados e os menos aptos são descartados. Os membros selecionados podem sofrer transformações em suas características fundamentais por meio de operadores genéticos como, por exemplo, os operadores de cruzamento e mutação, gerando descendentes para a próxima geração. O processo é repetido até que um critério de parada seja obedecido. Uma visão geral do funcionamento do GA é apresentada na Figura 2.

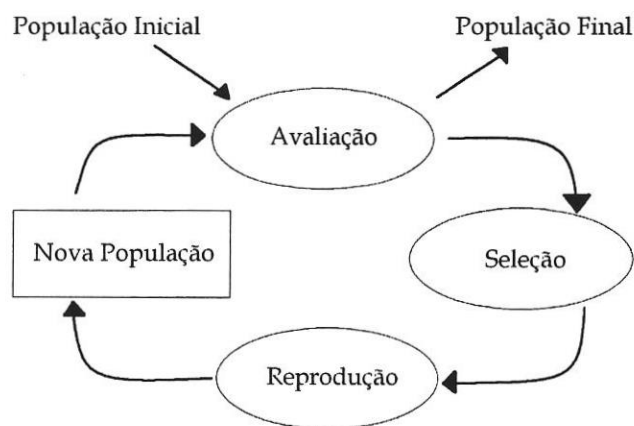


Figura 2: Visão Geral de um Algoritmo Genético

A seleção dos cromossomos que passarão para a próxima geração pode ser feita pelo seguinte procedimento: é calculada a soma das aptidões de todos os cromossomos da população; em seguida, a aptidão relativa de cada cromossomo é calculada, que será igual à aptidão desse cromossomo dividida pela soma de todas as aptidões calculadas; na sequência, um número no intervalo $[0, \text{SOMATOTAL}]$ é gerado, no qual SOMATOTAL é a soma de todas as aptidões. Por fim, a probabilidade de um dado cromossomo ser selecionado para a geração posterior é proporcional à contribuição de sua aptidão em

SOMATOTAL. Este procedimento é conhecido como Método da Roleta (Tabela 1). Este método é repetido até preencher a população intermediária com N cromossomos.

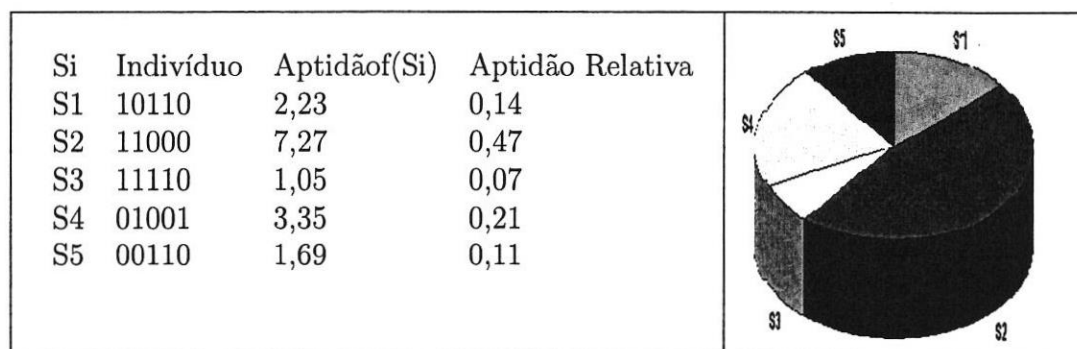


Tabela 1: Ilustração do método da Roleta

Outros métodos de seleção podem ser utilizados. Segundo Mitchell (1999), o torneio pode ser implementado da seguinte maneira: a cada seleção, dois indivíduos da população corrente são escolhidos aleatoriamente. Um número entre 0 e 1 é gerado. Se este número for menor que uma constante p_s no intervalo $[0, 1]$ (por exemplo, 0.75), o indivíduo mais apto é selecionado. Caso contrário, o menos apto é escolhido.

Os operadores de cruzamento e mutação são os principais mecanismos de busca dos GAs para explorar regiões desconhecidas do espaço de busca. Geralmente, o operador de cruzamento é geralmente aplicado a um par de cromossomos retirados da população intermediária, gerando dois cromossomos filhos. Cada um dos cromossomos pai tem seu vetor de características cortado em um ponto, ou seja, uma mesma posição definida aleatoriamente, produzindo duas cabeças e duas caudas. As caudas são trocadas, gerando dois novos cromossomos. Este tipo de cruzamento é conhecido como cruzamento simples. A Figura 3 ilustra o comportamento deste operador.

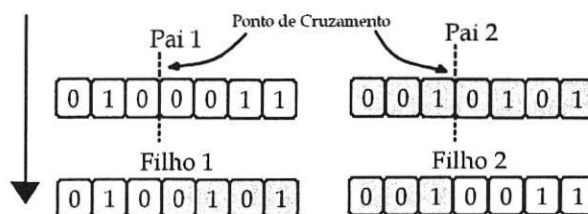


Figura 3: Exemplo de Cruzamento Simples

O operador de cruzamento é aplicado com uma dada probabilidade a cada par de cromossomos selecionados. Não ocorrendo o cruzamento, os filhos serão iguais aos pais (isto permite que algumas soluções sejam preservadas). Isto pode ser implementado gerando números pseudo-aleatórios no intervalo $[0, 1]$. Assim, o cruzamento só é aplicado se o número gerado for menor que a taxa de cruzamento.

Após a operação de cruzamento, o operador de mutação é aplicado, com dada probabilidade, em cada característica de cada filho. O operador de mutação pode alterar o valor da característica. A Figura 4 apresenta um exemplo deste operador. A mutação

melhora a diversidade dos cromossomos na população. Por outro lado, a mutação pode destruir informações regulares contidas no cromossomo. Por isto, deve ser utilizada uma taxa de mutação baixa, mas suficiente para assegurar a diversidade.

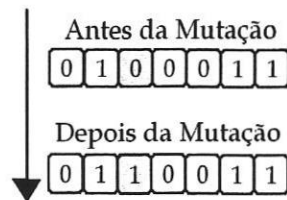


Figura 4: Exemplo de Mutação

Após a definição da primeira população, o procedimento se repete até que um critério de parada seja atingido. Quando se conhece a resposta máxima da função objetivo, este valor pode ser utilizado como critério de parada do Algoritmo Genético. Outros critérios de parada podem ser:

- Um certo número de gerações previamente definido é atingido;
- A população convergir (uma porcentagem alta pré-definida dos cromossomos possui um mesmo valor de aptidão);
- Uma grande parte dos cromossomos convergir, ou seja, apresentar os mesmos resultados;

Vale observar que o melhor cromossomo pode ser perdido de uma geração para outra devido ao corte do cruzamento ou à ocorrência de mutação. Portanto, é interessante transferir o melhor cromossomo de uma geração para outra sem alterações. Esta estratégia é denominada elitismo e é muito comum nos GAs tradicionais.

GAs merecem destaque para resolver problemas de agrupamento porque (Cowgill et al. 1998; Jain et al. 1999; Bayram et al. 2004):

1. Executam uma busca global pelas melhores soluções, enquanto a maioria dos procedimentos conhecidos de agrupamento executam uma busca local
2. Utilizam procedimentos de busca probabilísticos ao invés de regras determinísticas
3. Procuram por uma população de soluções em paralelo, permitindo evitar mínimos locais
4. Utilizam princípios evolutivos
5. A função objetivo e sua função de aptidão correspondente são suficientes para influenciar a direção da pesquisa
6. Podem obter um número de soluções potenciais para um dado problema, permitindo uma análise de diferentes soluções

Como GAs não utilizam operadores e nem funções fixas, vários métodos de adaptação de GAs foram propostos para solucionar problemas de agrupamento. Na maioria deles, as soluções candidatas ao problema são codificadas em cromossomos, que representam os indivíduos de uma determinada geração. A função objetivo de um algoritmo de agrupamento pode ser utilizada como função de aptidão do GA. Porém deve ser feita uma adaptação, pois, na maioria das vezes, a função objetivo deve ser minimizada para obter bons resultados, ou seja, as soluções que apresentam menor função objetivo recebem as maiores notas de aptidão. Em seguida, os cromossomos sofrem a ação de operadores genéticos, como seleção, cruzamento ou recombinação e mutação. A maneira com que esses operadores são utilizados varia muito nas implementações propostas, pois cada implementação pode utilizar uma representação distinta da outra. Além disso, algoritmos de agrupamento determinísticos, como os citados na Seção 2, são frequente utilizados em conjunto com os GAs. A forma com que o GA atua em conjunto com esses algoritmos varia com o algoritmo utilizado e as variáveis a serem otimizadas. Alguns exemplos de implementações serão apresentados na Seção 5.

Definir boas representações dos agrupamentos através de cromossomos, bem como funções de aptidão adequadas a resolução do problema, aplicar os operadores genéticos e encontrar parâmetros adequados ao algoritmo de agrupamento ainda são objetos de estudo. Esses aspectos serão discutidos com mais detalhes nas próximas seções.

4 Principais Aspectos do Agrupamento Evolutivo

Nesta seção serão apresentadas algumas técnicas de utilização de GAs em conjunto com algoritmos de agrupamento, sua motivação, e os métodos utilizados por essas técnicas para adaptar os algoritmos de agrupamentos aos GAs.

4.1 Representação Utilizada

Para que o algoritmo de agrupamento utilizado seja otimizado pelos GAs é necessário que as soluções para o problema possam ser efetivamente representadas. Algumas maneiras bastante utilizadas na literatura para representar o problema com um número fixo de agrupamentos consistem em (Cole 1998):

- Representar os objetos a serem agrupados como genes do cromossomo e indicar o subconjunto a que pertencem por meio do valor desses genes. Esse tipo de representação é conhecido como Grupo-Número e é ilustrada na Figura 5 (a).
- Representar o agrupamento por meio de uma matriz binária, onde as linhas representam os subconjuntos e as colunas representam os objetos. Esse tipo de representação é conhecido como matriz e pode ser vista na Figura 5 (b).
- Representar os objetos por meio do valor dos genes e o subconjunto a que ele pertence por meio da sua posição no cromossomo. Esse tipo de representação é conhecido como permutação e é apresentada nas figuras 5 (c) e (d).
- Representar apenas os centróides (Equação 12) de cada subconjunto como os genes

contidos no cromossomo. Essa representação, ilustrada na Figura 6, é conhecida como representação por centróide.

A representação Grupo-Número é uma das representações mais utilizadas para GAs em problemas de agrupamento (Raghavan & Birchard 1979; Murthy & Chowdhury 1996; Cowgill et al. 1998; Hruschka et al. 2004). Nessa representação, os cromossomos são compostos por vetores de tamanho N , com N igual ao número de padrões, onde cada gene representa um padrão e o valor contido nele indica o subconjunto que ele pertence. Essa representação pode ser visualizada na Figura 5 (a).

Bezdek et al. (1994) utiliza como cromossomo uma matriz binária $M_{k \times n}$, onde cada coluna representa um objeto a ser agrupado e cada linha um subconjunto. Se um objeto x_i pertencer ao subconjunto c_j , então o conteúdo da posição $M(i, j)$ será 1, caso contrário o mesmo será 0. Neste caso, $\sum_{j=1}^k M(i, j) = 1$ e $\sum_{i=1}^n M(i, j) \geq 1$, ou seja, cada objeto pode pertencer apenas a um subconjunto (agrupamento exclusivo) e nenhum subconjunto pode ser vazio. Bezdek et al. (1994) também utilizou essa representação para agrupamentos não exclusivos. Nesse caso a matriz era composta de números reais contendo a pertinência de cada objeto em cada subconjunto. Essa representação pode ser vista na Figura 5 (b).

Um outro tipo de representação utiliza o conteúdo dos genes para codificar os padrões e utiliza a posição dos padrões para identificar a que subconjunto pertencem. Como os diferentes indivíduos representam diferentes permutações dos padrões, esta codificação é denominada codificação por permutação. Permutação com separadores (Belew & Booker 1991) usam valores inteiros que não representam objetos para separar um subconjunto do outro dentro de um mesmo cromossomo, como mostrado na Figura 5 (c). Belew & Booker (1991) também propõem outra representação que necessita de buscas locais, também conhecida como permutação gulosa, que utiliza os k -primeiros genes como sementes para gerar os k agrupamentos, ou seja, cada padrão pertence à um agrupamento distinto e o representa. Os padrões restantes são adicionados aos agrupamentos cujas sementes apresentam a melhor similaridade com o padrão, na ordem que aparecem na permutação, durante a construção do subconjunto. A permutação gulosa pode ser mostrada na Figura 5 (d).

A Figura 5 mostra diferentes codificações para o agrupamento $\{\{A, C, F\}, \{B, D, E\}\}$. Neles, os agrupamentos são nomeados como 1 e 2 e os objetos são nomeados entre A e E .

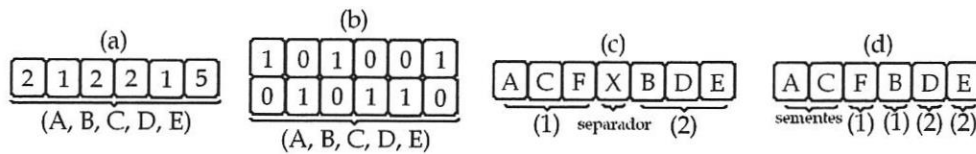


Figura 5: Cromossomos representando o agrupamento $\{\{A, C, F\}, \{B, D, E\}\}$ utilizando as representações: (a)Grupo-Número (b)Matriz (c)Permutação com separador (d) Permutação Gulosa

Uma outra forma muito utilizada para representar partições consiste compor um cromossomo com os centróides de cada subconjunto (Hall et al. 1999; Kivijärvi et al. 2003).

Neste caso, uma matrix $M_{x \times k}$ de valores representam uma partição, onde x é a dimensionalidade dos objetos da base de dados e k é o número de subconjuntos formados. Cada coluna contém o centróide de um subconjunto e cada linha representa a posição do mesmo na dimensão correspondente. Em alguns casos, os autores utilizam a mesma abordagem, porém optam por utilizar meióides à centróides (Pan et al. 2003; Ma 2005). A diferença entre centróides e meióides é o fato que os centróides são calculados como sendo o ponto central ou média de um subconjunto e o meióide é o padrão mais próximo desse centro. Esse tipo de representação pode ser visualizado na Figura 6, onde um conjunto de k centróides de x dimensões é representado.

d_1	2.5	5.8	4.9	4.87	7.48	...	7.23
d_2	6.8	9.21	4.6	3.65	6.5	...	8.5
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
d_x	8.2	1.45	3.5	8.47	6.2	...	6.58
	c_1	c_2	c_3	c_4	c_5	...	c_k

Figura 6: Cromossomos representando uma matriz contendo centróides

Quando o número de agrupamentos é fixo, um cromossomo que não seja capaz de representar todos os agrupamentos deve ser tomado como inválido ou ineficaz. Um cromossomo que represente os agrupamentos por valor do gene e não inclua todos os possíveis agrupamentos ou um cromossomo que utilize permutação com separadores e tenha separadores a mais, falta de separadores ou em posição inválida são padrões de cromossomos inválidos. Para evitar que esses cromossomos sejam gerados, é utilizada alguma heurística ou verificação da validade dos cromossomos, impedindo que esses possam influenciar negativamente no desempenho do GA.

Em alguns casos não se pode saber ao certo a quantidade correta de subconjuntos a serem formados para um determinado conjunto de padrões. É possível solucionar esse problema de duas formas: fazer com que o número de subconjuntos seja definido pela função de aptidão do GA durante a execução ou utilizar um algoritmo hierárquico. Para isso, outras representações foram propostas para esses tipos de agrupamento (Tseng & Yang 2001; Garai & Chaudhuri 2004; Greene 2003). Uma das representações consiste em utilizar uma das representações anteriores e anexar no cromossomo a quantidade de subconjuntos ou números correspondentes aos mesmos (Hruschka et al. 2004). Um exemplo deste cromossomo usando representação Grupo-Número pode ser visto na Figura 7.

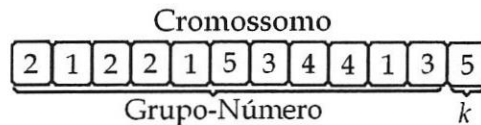


Figura 7: Cromossomo com número variável de subconjuntos

Uma outra forma de agrupar em um número indefinido de subconjuntos consiste em utilizar algoritmos hierárquicos. É possível utilizar GAs em conjunto com esses algoritmos. Uma abordagem consiste em utilizar um método que divida o conjunto de dados em m subconjuntos iniciais (Tseng & Yang 2001; Garai & Chaudhuri 2004). Esses métodos podem ser vistos com detalhes na Seção 5.3. Os cromossomos utilizados por essas técnicas representam os m subconjuntos $\{B_1, B_2, \dots, B_m\}$ gerados posteriormente pelos métodos, ou seja, representam os m subconjuntos iniciais. Cada cromossomo é constituído de um vetor binário de tamanho m . Se o valor do bit correspondente a um subconjunto for 1, esse subconjunto continuará a existir na partição. Caso o valor do bit correspondente seja 0, o subconjunto deixará de existir e os objetos contidos nele serão adicionados à outros agrupamentos cujo valor indicado pelo cromossomo seja 1. Nesta representação, os subconjuntos são adicionados de maneira aglomerativa pelo GA. A Figura 8 ilustra esse tipo de representação.

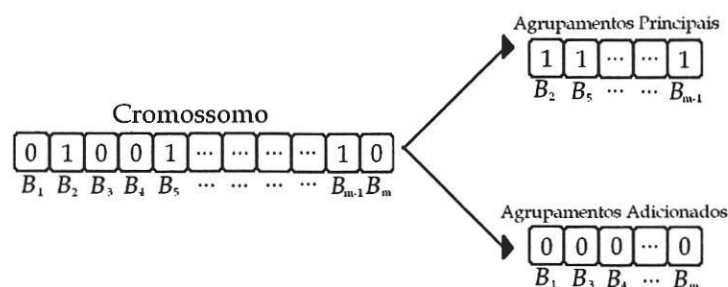


Figura 8: Exemplo de cromossomo para agrupamento hierárquico

Porém, é possível que o GA seja utilizado apenas para avaliar e selecionar conjuntos hierárquicos previamente formados, especialmente no caso de algoritmos divisivos (Greene 2003). Neste caso, cada um desses conjuntos são indivíduos da população e são representados por subconjuntos que podem ser particionados em vários níveis. Esses níveis contém partições contendo subconjuntos que são aninhados para formar novos subconjuntos, formando um dendograma (Jain et al. 1999). O processo é ilustrado pela Figura 1. A construção desses objetos pode ser vista com maior detalhamento na Seção 5.3.

GAs também podem ser utilizadas para encontrar valores para alguns parâmetros livres nas redes SOM como a taxa de aprendizado, a topologia da rede (número de neurônios), os pesos da rede e a função de vizinhança (incluindo seus parâmetros). Esses parâmetros são representados por genes no cromossomo, podendo ser compostos por números reais ou binários. Um exemplo de cromossomo que pode ser usado é ilustrado na Figura 9

Pesos			Taxa de Aprendizagem	Número de Neurônios	Função de Vizinhança
2.5	5.8	4.9	0.15	19.7	4.21
7.23	3.32	36.0			

Figura 9: Exemplo de cromossomo para redes SOM

Ao selecionar uma representação para o problema, é necessário levar em consideração o tipo de dado a ser agrupado, o conhecimento prévio do número de subconjuntos a serem formados e a complexidade de algumas funções necessárias como algoritmos de busca (no caso da permutação) e do cálculo da vizinhança entre os padrões (no caso da representação gulosa e por centróides). Para os algoritmos que não utilizam um número fixo de subconjuntos, é necessário que o número de subconjuntos representados pelo cromossomo seja o mesmo que o indicado, não apresentando nenhum subconjunto inválido ou vazio. No caso da utilização de algoritmos hierárquicos, é necessário que haja um pré-particionamento dos dados em subconjuntos para serem aglomerados pelo GA ou ao menos avaliados.

Todas as representações citadas nessa seção apresentam algum nível de redundância, ou seja, é possível que múltiplos cromossomos apresentem o mesmo resultado. Essa redundância cresce exponencialmente a medida que cresce o número de padrões a serem representados. Os parâmetros utilizados pelos algoritmos de agrupamento também podem ser otimizados pelos GAs. Uma maneira de se fazer isso é adicionando novos genes dentro do cromossomo que codifiquem esses parâmetros, como mostra a Figura 10. Esses genes deverão ser tratados pelo GA durante a execução.

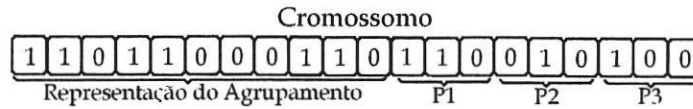


Figura 10: Cromossomo com genes representando os parâmetros P1, P2, P3

4.2 Função de Aptidão

A cada geração, os GAs aplicam operadores de seleção nos indivíduos para selecionar os melhores e criar uma nova população. Durante a seleção, é desejável distinguir quais cromossomos codificam os melhores resultados. Para isso é necessário escolher uma função de aptidão adequada ao tipo de técnica de agrupamento utilizado.

Muitas vezes a função de aptidão do GA é baseada na função objetivo do algoritmo de agrupamento. Geralmente, a meta do algoritmo de agrupamento é minimizar o resultado da função objetivo. Logo, é comum atribuir valores de aptidão maiores aos indivíduos que apresentarem os menores resultados da função objetivo. Uma medida muito comum utilizada pelo algoritmo *k-means* é a soma da distância Euclidiana quadrática entre os objetos contidos no subconjuntos e seus centróides (Hall et al. 1999; Maulik & Bandyopadhyay 2000). Ela obtém um significado de variância total entre subconjuntos e é apropriada para conjuntos de dados que possuem subconjuntos compactos ou isolados. Seu valor pode ser obtido por meio da Equação 3.

$$S_{ij} = \left(\sum_{n=1}^k (x_{in} - x_{jn})^2 \right)^{1/2} \quad (3)$$

Quanto maior a soma da distância Euclidiana, menos compacto é o agrupamento e portanto menor é aptidão do mesmo. Muitas funções de aptidão utilizam a distância Euclidiana como base. Uma delas é o erro médio quadrático (*mean squared error* - MSE)

dos subconjuntos obtidos (Raghavan & Birchard 1979; Kivijärvi et al. 2003). Dada uma partição $P = (p_1, p_2, \dots, p_N)$ e um conjunto de centróides $C = (c_1, c_2, \dots, c_k)$ é possível calcular o MSE por meio da Equação 4.

$$e(P, C) = \frac{1}{Nk} \sum_{i=1}^N d(x_i, c_{pi})^2, \quad (4)$$

onde N é o número de objetos a serem agrupados, K é o total de agrupamentos, e $d(x_i, c_{pi})$ é a distância Euclidiana entre o objeto x_i e o centróide do subconjunto que o contém c_{pi} . Neste caso, a aptidão de cada indivíduo w_i pode ser calculada por $\frac{1}{1+e(w_i)}$.

Cowgill et al. (1998) preferiu utilizar GA para minimizar a taxa do critério de variância (*Variance-Ratio Criterion* - VRC) tendo em vista o isolamento externo do subconjunto e homogeneidade interna. Se utilizada em um agrupamento com número indeterminado de subconjuntos, a função VRC ajudará a determinar o número correto de agrupamentos no conjunto de dados resultando em valores maiores para agrupamentos que se aproximam da forma natural dos dados. Essa função é dada pela Equação 5:

$$VRC = \frac{\text{trace}B/(k-1)}{\text{trace}W/(n-k)}, \quad (5)$$

onde n é o número total de objetos e k o número de agrupamentos, e *trace* é a soma quadrática das matrizes de covariância.

A função objetivo utilizada por Hruschka et al. (2004) para representações com número variável de subconjuntos é baseada em silhueta (Kaufman & Rousseeuw 1990). Para calcular a silhueta de um objeto i pertencente a um agrupamento A, é necessário calcular a dissimilaridade média do objeto i para todos os outros objetos contidos no agrupamento A, dada por $a(i)$. Em seguida é calculada a dissimilaridade do objeto i para cada agrupamento diferente de A e o a menor dissimilaridade obtida é dada por $b(i)$. Portanto, a silhueta $s(i)$ é calculada pela Equação 6.

$$s(i) = \frac{b(i) - a(i)}{\max\{a(i), b(i)\}} \quad (6)$$

A aptidão de cada indivíduo é dada pela média do cálculo da silhueta dos objetos agrupados. Quando maior a média, maior será a aptidão do indivíduo.

Na utilização de um número fixo de subconjuntos, a função de aptidão pode ser utilizada para evitar indivíduos com partições degeneradas, ou seja, partições com subconjuntos vazios. Para minimizar a chance do GA ficar preso em um desses indivíduos, Hall et al. (1999) utiliza uma penalização para esses cromossomos. Se uma partição apresentar b agrupamentos vazios será avaliada pela função objetivo calculada pela Equação 7.

$$\overline{\text{aptidao}}(V) = \text{aptidao}(V) + b \times \text{aptidao}(V) \quad (7)$$

Como a função de aptidão é inversamente proporcional a função objetivo do agrupamento, esses indivíduos são penalizados pela proporção de agrupamentos vazios que apresentam, reduzindo a chance de serem escolhidos.

Tseng & Yang (2001) utilizam como função de aptidão o cálculo das distâncias entre os padrões de um mesmo agrupamento e a distância entre o agrupamento e os demais agrupamentos. O cálculo dessa função de aptidão do cromossomo R é ilustrado na Equação 8, na qual C_i representa um agrupamento, D_{inter} a distância entre o agrupamento C_i e os outros agrupamentos, D_{intra} a distância interna de C_i e w uma constante que serve para aumentar a influência de D_{inter} ou D_{intra} , dependendo do seu valor. As distâncias inter e intra agrupamentos são medidas utilizando as equações 9 e 10, respectivamente.

$$Fitness(R) = \sum D_{inter(C_i)} * w - D_{intra(C_i)} \quad (8)$$

Distância *inter*-agrupamento:

$$D_2 = \left(\frac{\sum_{i=1}^{n_{k1}} \sum_{j=n_{k1}+1}^{n_{k1}+n_{k2}} (x_i - x_j)^2}{n_{k1}n_{k2}} \right)^{1/2} \quad (9)$$

Distância *intra*-agrupamento:

$$D_3 = \left(\frac{\sum_{i=1}^{n_{k1}+n_{k2}} \sum_{j=n_{k1}+1}^{n_{k1}+n_{k2}} (x_i - x_j)^2}{(n_{k1} + n_{k2})(n_{k1} + n_{k2} - 1)} \right)^{1/2} \quad (10)$$

Para representação hierárquica utilizada por Greene (2003), a aptidão dos indivíduos é calculada de duas maneiras distintas. Como o algoritmo hierárquico utilizado faz com que a maneira que os conjuntos são construídos varie durante a execução do GA, o subconjunto representado pelos cromossomos pode ser particionado ou não. Se o conjunto for formado pela combinação de subconjuntos menores de iterações anteriores, ou seja particionado, a função de aptidão é calculada pela soma das médias de aptidão e a distância média entre os centróides desses subconjuntos. Caso ele não seja particionado, a aptidão é calculada pela soma dos elementos contidos no agrupamento e a diferença entre dois atributos calculados durante a construção do subconjunto: o raio atual e o raio máximo permitido para o subconjunto.

No caso das redes SOM, um método comum para se calcular a aptidão é consiste na diferença entre a quantidade de instâncias de cada classe recebidas por cada neurônio da rede. Desta forma, quanto maior a diferença dos padrões de classes distintas em um neurônio, maior é a aptidão do mesmo e conseqüentemente da rede (Tanaka et al. 1996). Outras formas de validação de agrupamentos podem ser utilizadas para redes SOM, como a soma dos quadrados das distâncias entre os padrões e o representante do subconjunto mais próximo (Ma 2005).

Em resumo, existem vários métodos para avaliar diferentes tipos de agrupamento pois existem medidas distintas de validação de agrupamentos, como pode ser visto na Seção 5. Essas técnicas tem como objetivo avaliar o agrupamento resultante e não as técnicas utilizadas para gera-los (pelo menos não diretamente). Portanto, é possível comparar o desempenho de técnicas distintas de agrupamento por meio uma mesma medida de validação.

4.3 Operadores Genéticos

Os operadores genéticos são os responsáveis pela transferência do código genético de uma geração para outra. São eles que determinam como os cromossomos devem ser iniciados e como eles devem ser alterados durante as gerações. Os principais operadores genéticos consistem na inicialização dos cromossomos, seleção dos cromossomos, cruzamento e mutação. A seguir, é apresentada uma discussão sobre esses tópicos.

4.3.1 Inicialização

Ao gerar uma população inicial é interessante que os indivíduos possam abranger o maior espaço de busca possível. A inicialização também deve levar em conta o tipo de representação utilizada, de maneira que os cromossomos iniciais sejam válidos. Um método de se inicializar uma geração é gerar aleatoriamente os valores assumidos pelos cromossomos. Outros tipos de inicialização podem usar de heurística para obter uma população inicial que facilite a convergência do GA para uma solução ótima e reduza o tempo de execução. A seguir, são feitas algumas considerações sobre a inicialização de populações para algumas das representações citadas (Seção 4.1).

- Grupo-Número

A maneira mais comum de se inicializar um cromossomo quando se utiliza o valor do gene para indicar a que agrupamento o padrão pertence é gerar esses valores aleatoriamente. Porém, essa abordagem pode gerar cromossomos que não representem todos os agrupamentos. A população deve ser verificada e os cromossomos inválidos substituídos.

- Matriz Binária

Uma das formas de inicializar uma matriz de valores consiste em preencher todas as colunas com zeros e todas as posições e depois escolher aleatoriamente uma linha por coluna para receber o valor 1, indicando que aquele objeto pertence ao subconjunto correspondente a linha. No caso de agrupamentos não exclusivos, valores aleatórios podem ser utilizados para preencher as colunas, desde que seja respeitada

$$\sum_{j=1}^k M(i, j) = 1 \text{ e } \sum_{i=1}^n M(i, j) \geq 1.$$

- Permutação com separador

Belew & Booker (1991) geram uma população inicial utilizando representação permutação com separador, selecionando aleatoriamente os padrões em cada agrupamento de maneira que nenhum fique vazio. Em seguida a ordem desses padrões no agrupamento é permutada e o cromossomo é criado colocando os padrões e separadores.

- Permutação Gulosa

Para essa representação, Belew & Booker (1991) utilizam uma permutação aleatória para gerar a população inicial, evitando objetos repetidos.

- Utilizando Centróides

Cromossomos que representam centróides podem ser inicializados de duas formas: a posição inicial dos centróides pode ser escolhida aleatoriamente (Hall et al. 1999) ou obtida pelos subconjuntos resultantes da utilização de algum algoritmo de agrupamento (Ward 1963).

- Número variável de subconjuntos

É feita da mesma forma que as outras representações. Porém o número de subconjuntos representados em cada cromossomo é importante para o cálculo da aptidão ou mesmo para possibilitar o reconhecimento desses subconjuntos. Muitas vezes é necessário calcular esse número, mesmo o número de subconjuntos não seja definido previamente. Se esse número estiver contido no final do cromossomo, talvez seja necessário conferir.

- Representação Hierárquica

A maioria dos algoritmos hierárquicos necessitam de algum tipo de processamento antes que os indivíduos possam ser construídos. No caso dos algoritmos hierárquicos aglomerativos é preciso que haja a divisão dos objetos da base de dados em subconjuntos iniciais, para que possam ser aglomerados posteriormente (Tseng & Yang 2001; Garai & Chaudhuri 2004). Esse processo pode ser visto em detalhes na Seção 5.3. Em seguida, os agrupamentos formados irão compor os cromossomos da população inicial. Esses cromossomos são representados por vetores binários cujos valores são inicializados aleatoriamente.

Uma das formas de inicialização dos agrupamentos hierárquicos divisivos é adicionar os objetos ao subconjunto principal, causando seu particionamento. Ao inicializar a população, Greene (2003) organiza os padrões da base de dados de maneira aleatória e seleciona o primeiro padrão que é adicionado ao primeiro subconjunto. Os outros padrões são adicionados no subconjunto na sequência em que foram organizados, podendo causar seu particionamento (Seção 5.3). O processo é repetido para cada indivíduo, permitindo uma variedade grande de agrupamentos.

- Redes SOM

Os parâmetros e pesos utilizados pela rede SOM recebem valores iniciais aleatórios, dentro de um intervalo previamente definido.

4.3.2 Seleção

Os cromossomos são selecionados de acordo com sua aptidão relativa. Sendo assim, a seleção está diretamente ligada a função de aptidão, independente da representação utilizada. O método da roleta ou qualquer outro método de seleção pode ser utilizado. Utilizar elitismo possibilita preservar as melhores soluções para as gerações futuras.

Belew & Booker (1991) e Bezdek et al. (1994) utilizam métodos de seleção por ranking. Belew & Booker (1991) utilizam um método de seleção que escolhe apenas dois indivíduos a cada iteração que geram um filho que substitui o pior membro da geração. A escolha dos

indivíduos é feita com probabilidade linear entre b/P para o melhor membro e $(2-b)/P$, na qual P é o tamanho da população e b o *bias*.

Bezdek et al. (1994) organizam os indivíduos da geração em ordem de aptidão e selecionam os R melhores indivíduos. Deste conjunto, são selecionadas as duplas que irão se reproduzir. Depois da reprodução, os cromossomos filhos são inseridos na população e os indivíduos com os piores resultados são retirados.

Greene (2003) utiliza o método da roleta padrão para selecionar os indivíduos que serão utilizados como pais. Garai & Chaudhuri (2004) selecionam os pais aleatoriamente na população.

4.3.3 Cruzamento

O operador de cruzamento ou recombinação deve ser escolhido com cuidado para cada tipo de representação utilizada. Cuidados devem ser tomados para que durante o cruzamento não sejam gerados cromossomos inválidos. Para isso, heurísticas tem sido desenvolvidas para possibilitar operadores mais sensíveis ao contexto (Belew & Booker 1991; Greene 2003). A seguir, são feitas algumas considerações sobre tipos de cruzamento para cada uma das representações citadas na Seção 4.1.

- Grupo-Número

Raghavan & Birchard (1979) utilizam o operador de cruzamento padrão com um ponto (Seção 3), o que pode permitir que o operador gere resultados inferiores. Por exemplo, supondo um conjunto composto pelos objetos $\{A,B,C,D,E,F\}$ cuja partição ótima seja $\{A,B,C\}$ e $\{D,E,F\}$ para dois agrupamentos. A representação agrupamento-número para esses agrupamentos poderia ser 111000 e 000111. Aplicando o cruzamento, o resultado poderia ser 111111 e 000000, um resultado muito inferior ao anterior.

Problemas como esse motivaram os pesquisadores a definir melhores representações e operadores de cruzamento. Belew & Booker (1991) compara três tipos de cruzamento para esse tipo de representação: cruzamento com um simples ponto (Seção 3), uniforme e baseado em margens. Também é aplicado rejeição e reenumeração ao operador de cruzamento. Rejeição é aplicado aos cromossomos que são considerados inválidos e reenumeração consiste em reenumerar os pais canonicamente antes do cruzamento, permitindo uma maior sensibilidade ao contexto.

Para o cruzamento uniforme com um único ponto, todos os cromossomos filhos são gerados e apenas um cromossomo válido é selecionado. O Cruzamento uniforme é repetido até que o cromossomo filho contenha todos os agrupamentos ou um número máximo de iterações seja alcançado. O cruzamento baseado em margens constrói os cromossomos filhos por meio da combinação das margens dos cromossomos pais (dois padrões são considerados no mesmo agrupamento se estão conectados pela mesma margem). Por exemplo, considere os agrupamentos e suas respectivas representações ilustrados na Figura 11:

Os cromossomos filhos são inicializados com os conjuntos de intersecções não vazias desses agrupamentos até que o número correto de agrupamentos seja gerado. As

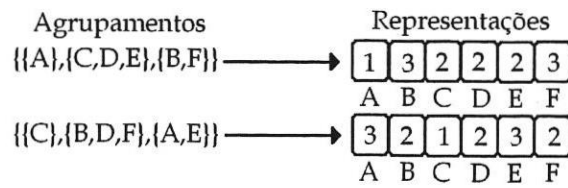


Figura 11: Agrupamentos e suas representações Grupo-Número.

intersecções e um padrão de filho para os agrupamentos da Figura 11 podem ser visualizados na Figura 12, na qual o cromossomo filho recebe $\{C, D\}$ do cromossomo 1, $\{A, E\}$ do cromossomo 2 e $\{B, F\}$ de ambos cromossomos pais.

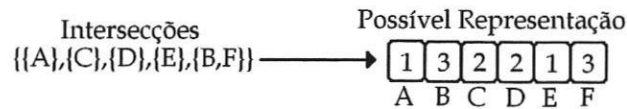


Figura 12: Intersecções não vazias e um padrão de cromossomo filho.

- Matriz Binária

Bezdek et al. (1994) efetua uma troca de colunas, trocando o padrão de subconjunto. A coluna e o número de colunas são selecionados aleatoriamente.

- Permutação com separadores e gulosa

As técnicas mais comuns de cruzamento para permutação com separadores consistem em cruzar repetidamente os cromossomos pais até que os filhos contenham todos os k agrupamentos, ou seja, sejam válidos (Belew & Booker 1991).

- Utilizando Centróides

Para esse tipo de representação podem ser aplicados os tipos comuns de cruzamento, como o simples ou o duplo. Kivijärvi et al. (2003) utilizou um conjunto com seis métodos distintos de cruzamento, de maneira que o método a ser utilizado é escolhido durante a execução do GA. Os detalhes dessa implementação podem ser vistos na Seção 5.1.

- Número variável de subconjuntos

Hruschka et al. (2004) utiliza uma troca de subconjuntos para esse tipo de representação. Primeiro são selecionados c subconjuntos do cromossomo pai A e os objetos contidos nesses subconjuntos são copiados para o cromossomo pai B. Os objetos que restaram dos subconjuntos de B alterados pelo cruzamento são colocados nos subconjuntos mais próximos. O processo é repetido invertendo-se a ordem dos cromossomos pais para gerar um segundo filho. Um exemplo é ilustrado na Figura 13, onde os subconjuntos 2 e 3 foram selecionados do cromossomo A e copiados para o cromossomo B. Os genes que foram afetados pela mudança ganham valor 0 no filho C e são posteriormente adicionados aos subconjuntos mais próximos.

- Representação Hierárquica

	Cromossomos Pais																		
P1-	1	1	2	3	2	4	5	1	2	5	4	3	2	5	3	3	4	2	4
P2-	1	2	1	2	3	3	2	1	2	4	4	2	3	2	2	1	3	2	1
P1-	1	1	2	3	2	4	5	1	2	5	4	3	2	5	3	3	4	2	4
P2-	1	2	2	3	2	3	2	1	2	4	4	3	2	2	3	3	3	2	1
	Cromossomo Filho																		
F1-	0	0	2	3	2	0	0	0	2	4	4	3	2	0	3	3	0	2	0

Figura 13: Cruzamento de cromossomos com número indeterminado de subconjuntos

Entre as técnicas de cruzamento hierárquico, Greene (2003) utilizam um cruzamento guloso direcionado pela aptidão dos subconjuntos que formam o cromossomo. Seja j o número de subconjuntos do primeiro cromossomo, k o número de conjuntos do segundo cromossomo e n a média entre j e k . São selecionados os n subconjuntos que apresentem a aptidão mais alta entre os dois cromossomos. Esses são os subconjuntos que formarão os cromossomos filhos. Por serem conjuntos matemáticos, os subconjuntos selecionados podem conter o mesmo padrão duas vezes ou não conter um padrão específico, invalidando o cromossomo. Uma verificação é feita para evitar cromossomos inválidos, sorteando o objeto comum entre os subconjuntos ou incluindo o objeto que falta.

Tseng & Yang (2001) selecionam um par de cromossomos R e Q e depois geram dois números aleatórios p e q no intervalo $[1, m]$, onde m é o tamanho dos cromossomos. Os cromossomos são divididos nesses pontos e o segmento entre eles é trocado entre os dois cromossomos. Garai & Chaudhuri (2004) propuseram o algoritmo HCMA para combinar e formar novos subconjuntos durante as gerações. Esse algoritmo seleciona dois indivíduos da população e efetua cruzamento de um ponto entre eles, gerando dois cromossomos filhos.

- Redes SOM A maioria das implementações utilizando Redes SOM efetua cruzamento simples em um ou dois pontos. Tanaka et al. (1996) seleciona dois cromossomos pais e troca os pesos contidos nos genes alelos entre os dois. Esse operador é aplicado com probabilidade p para cada dupla de genes alelos.

4.3.4 Mutação

O operador de mutação é responsável pelo aumento da diversidade da população. Quando GAs são utilizados para otimizar técnicas de subconjunto, o operador de mutação geralmente tem o efeito de mover um objeto de um subconjunto para outro. A seguir, são feitas algumas considerações sobre tipos de mutação que podem ser aplicados para cada tipo de representação citada (Seção 4.1).

- Grupo-Número

Para esse tipo de representação, a mutação consiste em alterar o valor do subconjunto no cromossomo. Belew & Booker (1991) alteram cada gene do cromossomo

com o valor $P_{mut} = \frac{1}{n}$, no qual n é o número de padrões contidos no cluster. Isso é feito apenas se o padrão não for o único padrão de um determinado subconjunto, pois não haveria o subconjunto caso a alteração fosse efetuada.

- **Matriz Binária** Nesse tipo de representação basta trocar um padrão para um subconjunto novo, ou seja, a posição do bit 1 é modificada na coluna. É necessária uma verificação para evitar subconjuntos vazios depois de efetuada a mutação.

- **Permutação com separadores e gulosa**

Para permutação, Belew & Booker (1991) selecionam dois padrões aleatoriamente, extraí esses padrões do cromossomo e depois efetua a troca de posições entre eles. É necessário verificar os separadores nos cromossomos, para permutação com separador, evitando a geração de cromossomos inválidos.

- **Utilizando Centróides**

Uma das formas mais simples de executar mutação nesse tipo de representação consiste em alterar os valores dos centróides para novos valores aleatórios (Hall et al. 1999; Maulik & Bandyopadhyay 2000). Uma outra forma de executar mutação no cromossomo consiste em substituir um centróide por um objeto selecionado aleatoriamente da base de dados (Kivijärvi et al. 2003). Porém essas técnicas podem sobrepor o centróide de um subconjunto distinto, fazendo com que uma verificação talvez seja necessária dependendo do algoritmo utilizado.

- **Número variável de subconjuntos** Hruschka et al. (2004) utiliza dois operadores de mutação distintos: (a) elimina aleatoriamente um subconjunto colocando seus objetos nos subconjuntos mais próximos, (b) divide um subconjunto selecionado aleatoriamente em dois novos subconjuntos, um com os objetos mais próximos do centróide original e outro tendo como centróide o objeto mais afastado do centróide original.

- **Representação Hierárquica**

A mutação de um subconjunto utilizada por Greene (2003) consiste em retirar um subconjunto do conjunto de subconjuntos e adicioná-lo posteriormente. Isso pode alterar a estrutura dos subconjuntos anteriores, pois o subconjunto retirado poderia fazer parte de um subconjunto maior. Além disso, ele utiliza um método estocástico e gradual que possibilita uma maior sensibilidade ao contexto. Estocástico porque ao tentar fazer a mutação, existe 50% de chance de que ela ocorra e gradual porque a medida que a aptidão de um cromossomo aumenta, menor será a quantidade de tentativas de mutação no cromossomo.

Tseng & Yang (2001) utilizam mutação *bit-a-bit*, aplicando a cada bit do cromossomo. Garai & Chaudhuri (2004) também utilizam mutação *bit-a-bit*, com probabilidade $\eta = \eta_0 * t$, sendo η_0 a probabilidade de mutação inicial e t o número de gerações que o algoritmo executou.

- **Redes SOM** A mutação dos parâmetros da rede SOM geralmente é feita alterando o valor dos dados contidos nos genes. Isso pode ser feito *bit-a-bit* ou gerando um novo número válido para o parâmetro a ser alterado.

5 Exemplo de Agrupamento em conjunto com GAs

Nessa seção são apresentados alguns detalhes de várias implementações que combinam algoritmos de agrupamento com GAs. Nela são apresentados 3 tipos de algoritmos que podem ser utilizados com GAs: *k-means*, Mapa Auto-Organizável (*Self Organizing Map* - SOM) e algoritmos de agrupamento hierárquico. Cada um desses algoritmos possuem diferentes tipos de parâmetros e topologias que podem ser

5.1 *K-means* em conjunto com GAs

A técnica *K-means* particiona o conjunto de dados em k subconjuntos com base em uma medida de similaridade. O algoritmo começa inicializando um grupo de k centróides de maneira que cada centróide possa formar um subconjunto. Cada padrão pertencente ao conjunto de dados é representado por um ponto em um espaço μ -dimensional (onde μ é o tamanho do vetor de entrada). A cada interação, um vetor média é computado para cada subconjunto e os pontos são realocados ao subconjunto com o vetor média mais próximo, de acordo com a medida de similaridade utilizada. Esse processo é repetido até que os subconjuntos converjam (o vetor média para cada subconjunto não mude) ou o número máximo de iterações seja atingido. O critério de agrupamento do *k-means* consiste em minimizar a distância entre cada ponto e o centróide do subconjunto ao qual ele pertence, sendo descrito pela Equação 11.

$$E = \sum_{k=1}^K \sum_{x_i \in C_k} D(x_i, c_k) \quad (11)$$

onde c_k é o centróide de um subconjunto C_k e $D(x_i, c_k)$ é a distância entre um objeto x_i da base de dados e o centróide c_k . O centróide pode ser a média ou a mediana de um grupo de pontos, dado pela Equação 12.

$$c_k = \frac{\sum_{i=1}^{n_k} x_{ik}}{n_k} \quad (12)$$

onde n_k é o número de objetos pertencentes ao subconjunto C_k e x_{ik} representa um objeto do subconjunto C_k .

O resultado dessa função objetivo é minimizado por subconjuntos de formato globular de mesmo tamanho ou subconjuntos bem separados (Halkidi et al. 2001). A complexidade do algoritmo é $O(n)$, geralmente sendo executado em um número pequeno de iterações e $k \ll n$ (Barbara 2000). Também é levado em consideração que a dimensionalidade dos dados é muito menor que a quantidade de dados. O algoritmo é sensível à escolha inicial dos centróides e da sua forma de atualização. Dependendo da escolha dos centróides, o algoritmo pode convergir para um ótimo local. Além disso, é restrito a dados em espaços Euclidianos e os subconjuntos encontrados são desbalanceados.

Na tentativa de minimizar esses problemas, GAs são utilizados em conjunto com *k-means*. Várias implementações foram sugeridas com esse objetivo. Um dos trabalhos

que utilizam GAs para melhorar a qualidade dos agrupamentos gerados por *k-means* foi Bezdek et al. (1994). Os autores utilizaram a representação do tipo matriz (Seção 4.1). Como *k-means* é sensível à escolha inicial dos centróides, eles utilizaram dois tipos de inicialização:

- Os dados são divididos na mesma proporção entre os subconjuntos.
- As características dos centróides da população inicial são escolhidas de um padrão aleatório a ser agrupado no respectivo subconjunto. Isso é feito para todas as características de cada centróide. Os subconjuntos iniciais são formados a partir desses centróides.

As partições geradas por cada indivíduo são avaliados pela Equação 13

$$J = 0.5 \times \sum_{i=1}^k n_i s_i, \quad (13)$$

onde n_i é o número de elementos do agrupamento i e

$$s_i = \frac{1}{n_i^2} \times i = \sum_{i=1}^{n_i} \sum_{j=1}^{n_i} \|x_i - x_j\|^2 \quad (14)$$

com x_i e x_j sendo a posição dos elementos contidos no agrupamento.

Um banco de dados Íris com 150 padrões e 3 classes foi utilizado para classificação e comparação dos resultados, sendo que cada classe continha 50 padrões. A avaliação da técnica foi feita contra *k-means* exclusivo, utilizando três medidas de distância distintas: euclidiana, mahalanobis e diagonal (Bezdek et al. 1994). Como o número de grupos era fixo, o critério utilizado para comparação consistiu em verificar a quantidade de padrões agrupados corretamente. Na maioria dos experimentos, a técnica que utilizou GA apresentou um erro menor.

Para determinar o efeito do cruzamento e mutação em seu algoritmo, Bezdek et al. (1994) mantiveram a mesma população inicial e utilizaram diferentes sementes para esses operadores. Além disso, a frequência de mutação foi variada. Não houve efeito positivo na taxa de erro e pouco efeito no aumento da aptidão. Isso levou os autores a afirmar que o erro obtido pelo método utilizado é mais sensível a população inicial do que aos operadores utilizados. As principais características do algoritmo podem ser vistas na Tabela 2.

Uma implementação simples de GAs otimizando *k-means* foi feita por Murthy & Chowdhury (1996). Os autores utilizam GAs para evitar que o algoritmo fique preso em ótimos locais. Para isso eles geram três bases de dados artificiais e utilizam uma base de óleo não refinado (Johnson & Wichern 1982) para serem agrupadas pelos algoritmos. Essa implementação difere da implementação de Bezdek et al. (1994) no tipo de representação utilizado, sendo esta a Grupo-Número. O principal resultado obtido pelos autores foi verificar que o método que utilizou GA conseguiu encontrar partições ótimas em todos os experimentos executados, pesquisando um número muito menor de partições

Algoritmo de Agrupamento	<i>k-means</i> exclusivo
Tipo de Representação	Matriz binária representando um padrão nos agrupamentos
População Inicial	Parcialmente aleatória.
Função de Aptidão	Baseada na densidade do agrupamento.
Seleção	Seleção por torneio
Cruzamento	Troca de colunas (troca o padrão de agrupamento), com a coluna e o número de colunas selecionados aleatoriamente.
Mutação	Há a troca de um padrão para um agrupamento novo, ou seja, a posição do bit 1 é modificada na coluna.
Elistismo	Os melhores indivíduos são adicionados na nova população, juntamente os indivíduos resultantes do cruzamento, para complementar o número de indivíduos faltantes.
Comparação dos Resultados	Avaliação da técnica foi feita contra <i>k-means</i> exclusivo
Detalhes	Estudou o efeito dos operadores genéticos sobre o erro.

Tabela 2: Resumo do GA utilizado por Bezdek

que todas as possíveis. Além disso, os autores conseguiram otimizar os resultados previamente obtidos por *k-means* por meio de GA, fazendo com que a busca pelo agrupamento com menor objetivo não ficasse presa em um mínimo local. As principais características do algoritmo podem ser vistas na Tabela 3.

Algoritmo de Agrupamento	<i>k-means</i>
Tipo de Representação	Grupo-Número
População Inicial	Aleatória
Função de Aptidão	Soma dos quadrados da distância Euclidiana entre os membros do agrupamento.
Seleção	Roleta
Cruzamento	Cruzamento simples
Mutação	Há a troca de um padrão para um agrupamento novo
Elistismo	O melhor indivíduo é adicionados a nova população somente se sua aptidão for maior que a menor aptidão dos indivíduos da população
Comparação dos Resultados	Avaliação da técnica foi feita contra <i>k-means</i> exclusivo
Detalhes	Encontrou agrupamentos ótimos sem a necessidade de procurar todas a possíveis partições

Tabela 3: Resumo do GA utilizado por Murthy

Um dos importantes estudos sobre a utilização de GAs para agrupamento foi feito por Cowgill et al. (1998). Os autores afirmam que a maioria das técnicas de agrupamento tende a não procurar soluções ótimas tendo em vista o isolamento externo do subconjunto e homogeneidade interna e que eles não vão obrigatoriamente encontrar a melhor separação e coesão dos subconjuntos possíveis. Por isso, ao invés de utilizar GA para minimizar a soma dos quadrados da distância Euclidiana entre os padrões, ele utiliza GA para minimizar a taxa do critério de variância (*Variance-Ratio Criterion* - VRC), descrita

na Equação 5. Essa medida também é utilizada para determinar o número correto de subconjuntos contidos no conjunto de dados, uma vez que ela resulta em valores maiores para agrupamentos que se aproximam da partição natural dos dados.

Alternativamente ao método de Ward (1963) (hierárquico), COWCLUS procura pelos dados testando uma população de agrupamentos, mantendo e cruzando os melhores membros. Uma das vantagens desse algoritmo é que sua busca não é determinada pelo cálculo de um valor único, mas utiliza um conjunto de soluções, trabalhando em torno de um ótimo local.

Um dos diferenciais do trabalho realizado por Cowgill et al. (1998) consiste no fato que nenhum algoritmo de agrupamento determinístico (*hill-climbing*) é utilizado durante a execução do GA, para reduzir o custo computacional do algoritmo. *K-means* é utilizado apenas para melhorar os indivíduos mais aptos da última geração, se esses forem passíveis de melhora. Os autores chamam essa estratégia de “*local improvement*”.

Trabalho foi comparado com o Método de Ward (1963), o algoritmo *k-means* com os centróides iniciais escolhidos pelo algoritmo de Ward (1963) (*k-means* - W) e *k-means* com os centróides iniciados aleatoriamente (*k-means* - R). A maioria dos métodos obteve boa performance na classificação de bases de dados constantes e não constantes, exceto pelo *k-means* (R). Quando o método de validação utilizado foi o VRC, COWCLUS obteve 97,53% dos melhores resultados.

As principais características do algoritmo COWCLUS podem ser visualizadas na Tabela 4.

Algoritmo de Agrupamento	GA com VRC como função de aptidão
Tipo de Representação	Grupo-Número
População Inicial	Aleatória
Função de Aptidão	Variance Ratio Criterion (VRC)
Seleção	Seleção com probabilidade $p_r = \frac{(b+1-r)}{b(b+1)/2}$
Cruzamento	Cruzamento simples
Mutação	Substituição por um número aleatório de um agrupamento válido
Elistismo	O melhor indivíduo é adicionados a nova população
Comparação dos Resultados	Trabalho foi comparado com o Método de Ward, <i>k-means</i> (W), <i>k-means</i> (R)
Detalhes	Uma variação utiliza algum algoritmo determinístico (<i>Hill-climbing</i>) como <i>k-means</i> para otimizar os melhores indivíduos da última geração

Tabela 4: Resumo do Algoritmo COWCLUS

A ação dos operadores genéticos aplicados durante a execução do GA pode gerar partições degeneradas, ou seja, partições que contenham subconjuntos vazios. Além disso, utilizar GAs em conjunto com técnicas de agrupamento gera a necessidade de bons valores para os que vários parâmetros livres como taxas de cruzamento, mutação, critério de parada, tamanho da população e o número indivíduos a ser selecionados. Na tentativa de solucionar esses problemas, Hall et al. (1999) dão continuidade ao trabalho

feito por Bezdek et al. (1994) utilizando GAs para melhorar o desempenho de *k-means* nas versões exclusivo e não exclusivo. A população utilizada também é composta por indivíduos representados por matrizes, porém são representados apenas os centróides de cada subconjunto. As funções objetivos utilizadas por ele são semelhantes as utilizadas por Bezdek et al. (1994). Para *k-means* exclusivo, o resultado da função é dado pela Equação 15.

$$R(V) = \sum_{k=1}^n \min\{D_{1k}, D_{2k}, \dots, D_{ck}\} \quad (15)$$

onde $D_{ik}(v_i, x_k)$ é a medida da distância Euclidiana de x_k para o centróide i e n é o número de padrões a serem agrupados. Para evitar partições degeneradas, os autores utilizam o termo de penalização dado pela Equação 7, na Seção 4.2.

Como para *k-means* não exclusivo, a função de aptidão é dado pela Equação 16.

$$R_m(V) = \sum_{k=1}^n \left(\sum_{i=1}^c D_{ik}^{1/(1-m)} \right)^{1-m} \quad (16)$$

com c sendo o número de agrupamentos, m é o grau de pertinência ao agrupamento e $D_{ik}(v_i, x_k)$ é a medida da distância que pode ser Euclidiana ou Diagonal (Bezdek et al. 1994).

Antes de passarem pelo processo de seleção por torneio, os indivíduos são transformados em sequências *gray* binárias. Logo após a seleção, os operadores de cruzamento e mutação são aplicados com probabilidade p_c e p_m respectivamente, sendo que o cruzamento é feito em dois pontos e a mutação é feita bit-a-bit, invertendo se o valor do bit.

Os autores utilizam algumas heurísticas para tentar encontrar bons valores para os parâmetros livres. Hall et al. (1999) utiliza valores fixos para o tamanho da população (dentro do intervalo de 30 - 100 indivíduos) e o número de gerações utilizado como critério de parada. Porém, os outros parâmetros têm seus valores modificados durante a execução. Um número alto de indivíduos selecionados para torneio pode levar a uma convergência rápida para um mínimo local. Por isso, o número de indivíduos selecionados é inicialmente baixo (2 indivíduos), porém nas últimas 50 gerações essa quantidade é aumentada (10 indivíduos). A probabilidade de cruzamento e mutação varia de acordo com a aptidão de cada indivíduo. Tomando f_{max} como a maior aptidão da população, $\bar{f}$ a aptidão média entre os indivíduos da população, f a aptidão do cromossomo filho prestes a sofrer mutação e f' a maior aptidão entre os dois cromossomos filhos prestes a sofrer cruzamento, as probabilidades de cruzamento e mutação são dadas pelas Equações 17, 18, 19 e 20.

$$p_c = k_1(f_{max} - f')/(f_{max} - \bar{f}), f' \geq \bar{f}, \quad (17)$$

$$p_c = k_3, f' < \bar{f} \quad (18)$$

$$p_m = k_2(f_{max} - f)/(f_{max} - \bar{f}), f \geq \bar{f}, \quad (19)$$

$$p_m = k_4, f < \bar{f} \quad (20)$$

com k_1, k_2, k_3 e k_4 constantes e ≤ 1.0 .

Para testar o algoritmo, Hall et al. (1999) realizou experimentos com seis bases de dados distintas incluindo a base Iris, artificiais e imagens de ressonância magnética e Lenna. Os autores afirmam que as partições resultantes dos agrupamentos guiados por GAs são idênticos aos obtidos por *k-means* exclusivo e não exclusivo quando os últimos recebem a melhor inicialização possível. Durante o experimento, os autores utilizaram *k-means* exclusivo sobre as partições obtidas pelo algoritmo guiado por GA antes de convergir e os resultados obtidos foram bons inclusive para GAs com a população pequena.

Os resultados obtidos com GA com parâmetros adaptativos não mostraram um aumento significativo na aptidão dos indivíduos e o tempo de execução foi 2.5 vezes maior. A utilização de valores reais nos cromossomos mostrou uma redução de tempo de execução em relação a utilização de valores binários. Porém o número de gerações necessárias para a conversão da população foi maior, o que fez com que o algoritmo não apresentasse uma economia significativa de tempo. A tabela 5 apresenta as principais características do algoritmo GGA utilizado por Hall et al. (1999).

Algoritmo de Agrupamento	baseado em <i>k-means</i> exclusivo e não exclusivo
Tipo de Representação	Matriz contendo os centróides dos agrupamentos
População Inicial	Os valores iniciais são escolhidos aleatoriamente
Função de Aptidão	Média das somas das distâncias Euclidiana e Diagonal
Seleção	Torneio
Cruzamento	Duplo
Mutação	bit-a-bit
Elistismo	Os dois membros mais aptos vão para a próxima geração
Comparação dos Resultados	<i>k-means</i> exclusivo e não exclusivo
Detalhes	Utilizou funções para encontrar automaticamente bons valores para os parâmetros livres

Tabela 5: Resumo do Algoritmo GGA

Vários trabalho semelhantes aos anteriores foram realizados, dentre eles os algoritmos GKMCA (Wu et al. 2003) e KGACLU (Maulik & Bandyopadhyay 2000). Algumas diferenças em relação aos operadores de cruzamento e seleção foram implementadas, porém características como a utilização de matriz de centróides, utilização de distância Euclidiana como medida de dissimilaridade, método de cruzamento e mutação continuam as mesmas.

O algoritmo KGACLU é utilizado como comparação por alguns trabalhos, dentre eles Pan et al. (2003). Ele foi utilizado para agrupar 7 bases de dados, sendo que 4 continham dados artificiais divididos de 2 a 9 grupos e três reais com tipos de dado já utilizados anteriormente, contendo dados de *Vowel*, *Iris* e óleo não-refinado (Johnson & Wichern 1982). É importante ressaltar que os resultados obtidos pelo algoritmo utilizando GAs (KGACLU) são semelhantes aos resultados obtidos por *k-means* com a melhor inicialização. Os detalhes do algoritmo utilizado podem ser vistos na Tabela 6.

Algoritmo de Agrupamento	baseado em <i>k-means</i> exclusivo
Tipo de Representação	Matriz contendo os centróides dos agrupamentos
População Inicial	Os valores iniciais são retirados de uma padrão da base de dados aleatoriamente
Função de Aptidão	Média da Soma da Distância Euclidiana
Seleção	Roleta
Cruzamento	Simples
Mutação	Altera o valor do cromossomo
Elistismo	Os dois membros mais aptos vão para a próxima geração
Comparação dos Resultados	Com <i>k-means</i> exclusivo
Detalhes	

Tabela 6: Resumo do Algoritmo KGACCLUS

Pan et al. (2003) afirma que um dos problemas dos algoritmos de agrupamento que utilizam GA é a convergência prematura para uma solução não ótima. Por isso, o algoritmo HGACCLUS (Hybrid GA based Clustering) utiliza de um método de esfriamento durante a fase de seleção que evita que poucos indivíduos das populações iniciais dominem rapidamente a população. Esse método vai sendo alterado conforme o algoritmo é executado, permitindo uma maior discriminação dos indivíduos nas últimas gerações, onde a diferença de aptidão é mínima. Esse método é calculado utilizando a Equação 22 e é utilizado durante a seleção de indivíduos, calculado pela Equação 21.

$$p(c_k) = \frac{\exp(f(c_k)/T)}{\sum_{k=1}^p \exp(f(ch)/T)}, \quad (21)$$

onde $f(c_k)$ é o cálculo da VRC, dada pela Equação 5, para o cromossomo k , p é o tamanho da população e

$$T(g) = \frac{G-g}{G}T_0, g = 0, 1, \dots, G-1, \quad (22)$$

onde G é o total de gerações, g é o número da geração atual e T_0 é a temperatura inicial.

Assim como o algoritmo COWCLUS (Cowgill et al. 1998), HGACCLUS utiliza VRC (Equação 5) como função objetivo. Porém, a representação utilizada é a matriz de centróides dos subconjuntos, como no algoritmo KGACCLUS (Maulik & Bandyopadhyay 2000).

Os resultados dos experimentos obtidos foram comparados com *k-means*, PAM, KGACCLUS e COWCLUS. Os valores obtidos por meio da utilização dos índices de validação VRC e silhueta (Equação 6) foram utilizados como critério de avaliação. Apesar dos algoritmos COWCLUS e KGACCLUS terem obtido bons resultados em relação à *k-means* e PAM, eles convergiram prematuramente em comparação ao algoritmo HGACCLUS, que obteve valores de VRC maiores. A principais características desse algoritmo podem ser

vistas na Tabela 7.

Algoritmo de Agrupamento	Algoritmo que utiliza VRC
Tipo de Representação	Matriz contendo os centróides dos agrupamentos
População Inicial	Os valores iniciais são retirados de uma padrão da base de dados aleatoriamente
Função de Aptidão	Variance Ratio Criterion (VRC)
Seleção	Stochastic Universal Sampling
Cruzamento	Cruzamento uniforme (cada gene é escolhido aleatoriamente de um dos pais)
Mutação	Substituição por um número aleatório de um agrupamento válido
Elistismo	O membro com maior aptidão é transferido
Comparação dos Resultados	Com <i>k-means</i> , PAM, KGACCLUS, COWCLUS
Detalhes	Os critérios de VRC e silhueta são utilizados

Tabela 7: Resumo do Algoritmo HGACCLUS

Kivijärvi et al. (2003) utiliza *k-means* para executar sintonia fina no algoritmo de agrupamento evolutivo utilizado, o SAGA. Entretanto, a sintonia fina é executada pelo algoritmo para cada indivíduo antes dele ser adicionado a uma população, ao contrário do algoritmo COWCLUS (Cowgill et al. 1998) onde *k-means* é executado apenas em alguns indivíduos da última geração. Isso fez com que o algoritmo pudesse convergir mais rápido, porém aumentou muito seu custo computacional.

Além disso, os valores dos parâmetros livres do algoritmo variam durante o tempo de execução por meio do GA, permitindo a utilização de valores mais apropriados. A representação utilizada é composta pelo conjunto de centróides dos subconjuntos adicionados dos valores dos parâmetros livres. Os indivíduos recebem uma nota de aptidão proporcional a MSE de suas partições, dada pela Equação 4, e são selecionados pelo método da roleta, podendo passar pelos operadores de cruzamento e mutação. Kivijärvi et al. (2003) utilizou seis métodos distintos de cruzamento, sendo eles:

1. **Cruzamento aleatório em vários pontos:** metade dos agrupamentos são selecionados aleatoriamente de cada pai para formar o cromossomo filho.
2. **Cruzamento baseado na distância do centróide:** os agrupamentos mais próximos do centróide do conjunto de objetos são retirados de um cromossomo pai, e os mais afastados são retirados do outro pai.
3. **Cruzamento com as maiores partições:** os centróides dos agrupamentos com maior número de objetos são selecionados.
4. **Cruzamento de pares em vários pontos:** os agrupamentos dos pais são comparados de entre si de maneira que cada agrupamento de um cromossomo pai tenha o seu respectivo no outro.
5. **Cruzamento de pares em um ponto:** a primeira metade de agrupamentos é retirada de um pai, a outra metade do outro.

6. Cruzamento de pares com o vizinho mais próximo: os pais são mesclados em uma única solução e os agrupamentos mais próximos são unidos 2 à 2.

O cruzamento com as maiores partições, o de pares em um ponto e o de pares com o vizinho mais próximo são determinísticos de maneira que, para uma dupla fixa w_a e w_b de cromossomos pais, a solução será sempre a mesma. Portanto, para que não haja perda de variação genética, deve-se aplicar operadores como mutação e ruído. O operador de mutação consiste em substituir cada um dos centróides do cromossomo com probabilidade ψ por um objeto selecionado aleatoriamente da base de dados. O operador de ruído consiste em adicionar um vetor com valores aleatórios entre $[-v, v]$ em cada centróide, para um valor da variação v que seja reduzido em tempo de execução. Depois da aplicação dos operadores genéticos, *k-means* é utilizado para fazer a sintonia fina das novas soluções.

Alguns parâmetros livres como o método de cruzamento utilizado γ , a probabilidade de mutação ψ e a variação de ruído v devem ser estimados para cada conjunto de dados, de maneira que esses permitam um melhor desempenho do algoritmo. Kivijärvi et al. (2003) utiliza a adaptação dos valores desses dados por meio da execução do próprio GA, adicionando os parâmetros citados no final do cromossomo. Esses parâmetros são obtidos dos cromossomos pais durante a geração de um novo cromossomo, de maneira que haja o cruzamento e mutação dos próprios parâmetros livres dos pais para se obter qual será utilizado durante a aplicação dos operadores genéticos no restante do cromossomo, ou seja, no vetor de centróides.

Alguns testes foram feitos com o algoritmo SAGA e os resultados foram comparados com os algoritmos *k-means*, *Splitting method with local repartitioning* (Fränti et al. 1997), método de Ward (Ward 1963), *Stochastic Relaxation* (Zeger & Gersho 1989), Busca Local Aleatória (Fränti & Kivijärvi 2000) e GA (Fränti et al. 1997). A função objetivo utilizada foi o MSE (Seção 4.2) das partições criadas por cada algoritmo. As bases utilizadas apresentavam dados obtidos em sua maioria de imagens, formando até 256 agrupamentos (um para cada cor utilizada). O MSE do algoritmo SAGA foi menor do que dos algoritmos comparados. A pequena diferença obtida entre o algoritmo SAGA e o algoritmo GA indica que a adaptação dos parâmetros livres gera um benefício mínimo na redução do MSE. Além disso, uma redução do número de gerações do algoritmo SAGA foi causada pela utilização de *k-means* para fazer a sintonia fina dos novos cromossomos. As principais características do algoritmo SAGA podem ser visualizadas na Tabela 8.

Hruschka et al. (2004) propôs algumas melhorias para utilizar GAs como ferramenta para agrupamento. No algoritmo proposto, o número de subconjuntos não é fixo, ou seja varia durante a execução. A representação utilizada é a Grupo-Número, porém o número de subconjuntos é adicionado no final do cromossomo, o que faz com que não seja necessário percorrer o cromossomo em busca desta informação. A população inicial é gerada de forma que o número de subconjuntos seja selecionado aleatoriamente de um intervalo pré-definido. Sem seguida, os objetos da base de dados são adicionados aleatoriamente aos subconjuntos e os seus cromossomos validados.

A função objetivo utilizada é baseada em silhueta (Kaufman & Rousseeuw 1990), como descrita na Seção 4.2, e a seleção é feita pelo método da roleta.

Algoritmo de Agrupamento	GA em conjunto com <i>k-means</i>
Tipo de Representação	Conjunto de centróides adicionados de valores para os parâmetros livres
População Inicial	k objetos são selecionados da base de dados aleatoriamente para formar os cromossomos
Função de Aptidão	MSE baseada na distância Euclidiana
Seleção	Roleta
Cruzamento	Utiliza 6 métodos de Cruzamento distintos
Mutação	Substitui um centróide por um objeto selecionado aleatoriamente da base de dados
Elistismo	O membro com maior aptidão é transferido
Comparação dos Resultados	A comparação com <i>k-means</i> , SLR, Ward's, SR, Randomised local search e GA simples
Detalhes	Utiliza <i>k-means</i> para sintonia fina e o parâmetros livres sofrem adaptação durante a execução. Tem alto custo computacional

Tabela 8: Resumo do Algoritmo SAGA

Nessa implementação o cruzamento é feito tendo como base dos subconjuntos dos cromossomos pais, como descrito na Seção 4.3.3, e utiliza dois operadores de mutação, descritos na Seção 4.3.4.

O indivíduo com maior aptidão da geração anterior é copiado para a próxima geração. O processo é repetido até que a população convirja. Alguns aprimoramentos foram feitos no algoritmo desde a primeira versão, sendo eles:

1. A utilização do algoritmo *k-means*, que é aplicado em cada indivíduo no início de cada geração.
2. A aplicação de uma medida de similaridade baseada em centróides, o que faz com que não haja mais a necessidade de armazenar as matrizes de dissimilaridade.
3. A utilização de uma função objetivo mais sensível ao contexto, dada por $s(i) = \frac{b(i)}{a(i)+\epsilon}$, onde ϵ é o termo utilizado para evitar problemas quando $a(i)$ for 0.

A principais características do algoritmo CGA utilizado por Hruschka et al. (2004) podem ser vistas na Tabela 9.

5.2 Redes SOM em conjunto com GAs

Uma rede SOM (Haykin 1999) é uma rede neural não supervisionada baseada em aprendizado competitivo. Os neurônios de saída da rede competem entre si para serem ativados. Nesse tipo de rede, os neurônios são organizados em um reticulado de uma ou duas dimensões. Cada neurônio no reticulado está conectado a todas as entradas da rede.

Esta rede geralmente utiliza uma única camada computacional. Para cada padrão de entrada apresentado à rede, os neurônios computam seus valores de ativação, ativando

Algoritmo de Agrupamento	Algoritmo que utiliza Silhueta
Tipo de Representação	Grupo-Número, com o número de agrupamentos
População Inicial	Aleatoriamente
Função de Aptidão	Baseado em Silhueta
Seleção	Roleta
Cruzamento	Feito por troca de agrupamentos entre os pais
Mutação	Utiliza dois operadores: o primeiro elimina aleatoriamente um agrupamento e o segundo divide um agrupamento selecionado aleatoriamente em dois novos agrupamentos
Elistismo	O membro com maior aptidão é transferido
Comparação dos Resultados	A comparação de resultados foi feita entre as variações do algoritmo
Detalhes	Foram implementados alguns aprimoramentos, incluindo a utilização do algoritmo <i>k-means</i>

Tabela 9: Resumo do Algoritmo CGA

uma região diferente do reticulado. O neurônio com maior valor de ativação é o vencedor da competição. Em seguida, é determinada a localização espacial de uma vizinhança topológica de neurônios excitados, centrada no neurônio vencedor. O próximo passo consiste de uma adaptação dos pesos. Os ajustes dos pesos são tais que a resposta do neurônio vencedor à aplicação subsequente de um padrão de entrada similar é melhorada. Assim, os vetores de entrada direcionam o movimento dos vetores de peso, durante a execução do algoritmo, promovendo uma organização topológica dos neurônios da rede. Ainda durante o treinamento, uma região de vizinhança dos neurônios é gradativamente reduzida. Esta região define que neurônios, além do neurônio vencedor, terão seus pesos ajustados.

O objetivo da rede SOM é encontrar um conjunto de vetores de referência e associar cada ponto do conjunto de dados ao vetor referência mais próximo. O algoritmo depende da inicialização destes vetores. O resultado consiste de um conjunto de vetores de referência que definem implicitamente os subconjuntos. Uma deficiência da rede SOM é que ela não detecta automaticamente a borda dos subconjuntos.

As redes SOM recebem como entradas atributos numéricos e geram subconjuntos hiperesféricos. Elas podem utilizar como medida de similaridade a distância Euclidiana, produto interno e, no caso de dados de expressão gênica, a medida de correlação de Pearson. Os parâmetros que podem ser otimizados nas redes SOM são a taxa de aprendizado, sua topologia (número de neurônios) e a função de vizinhança (incluindo seus parâmetros).

Tanaka et al. (1996) utilizaram SOM supervisionado em conjunto com GAs para tentar extrair informações internas contida em uma base de dados com possíveis pacientes de diabetes. Neste trabalho, GAs são utilizados para a sintonia dos coeficientes dos pesos da rede. Eles modificam o algoritmo SOM canônico de duas maneiras:

1. Informação categórica é adicionada ao vetor de treinamento através de um parâmetro fixo p . Esse parâmetro é utilizado no aprendizado supervisionado do SOM e seu valor está contido no cromossomo.

2. Elementos do vetor de treino também utilizam pesos, e esses pesos são determinados pelos GAs. A avaliação do vetor de pesos é feita pela habilidade de classificação do SOM gerado pelo cromossomo.

O método utilizado por Tanaka et al. (1996) tem como objetivo fazer discriminações na malha do SOM. Os pesos são decididos por GAs, onde cada indivíduo corresponde a uma rede SOM. Essa rede é treinada ciclicamente um número fixo T de vezes na construção de cada população. Em seguida, cada rede é avaliada pela quantidade de objetos de cada classe recebidas por cada neurônio da rede. Quanto maior a diferença entre o número de objetos, maior a aptidão do indivíduo. A Tabela 10 apresenta um resumo das características do algoritmo.

Algoritmo de Agrupamento	SOM utilizando distância Euclidiana
Tipo de Representação	Cada gene representa um peso da rede, sendo que para cada gene são utilizados 3 bits.
População Inicial	Aleatória.
Função de Aptidão	Feita pela quantidade de instâncias que cada neurônio da rede recebeu de cada classe. Quanto maior a diferença dos padrões maior é a aptidão
Seleção	Torneio
Cruzamento	São selecionados dois cromossomos pais e depois os pesos contidos nos genes são trocados entre si com probabilidade p , para cada dupla de genes alelos.
Mutação	Inverte o valor de um bit com probabilidade c (bitwise).
Elitismo	O melhor candidato de cada geração é copiado.
Comparação dos Resultados	Comparação com MLP utilizando Backpropagation
Detalhes	Nenhum

Tabela 10: Resumo do GA utilizado por Tanaka

GAs não precisam ser utilizados para melhorar a topologia das redes SOM ou mesmo encontrar bons valores para os parâmetros livres das redes necessariamente. Ohtsuka et al. (2002) utilizaram redes SOM em conjunto com GAs para detectar confusão em amostras de sangue. As redes SOM selecionam alguns entre 15 elementos de uma tabela para detectar confusão em amostras de sangue, a tabela CBC (*Complete Blood Count*). Segundo o autor, é possível que a utilização de todos os elementos da tabela CBC não ajudem a detectar confusão. Em outras palavras, o mapa treinado com os dados de aprendizado nem sempre resulta na melhor detecção. Por isso GAs são utilizados para selecionar os elementos a serem utilizados na detecção de confusão nas amostras de sangue.

Os cromossomos utilizados são formados por 15 bits que representam a utilização ou não dos elementos da tabela CBC. Em seguida são gerados SOMs correspondentes à cada um dos indivíduos e a atualização dos pesos dessas redes é instruída pelos elementos contidos no cromossomo. A aptidão de cada indivíduo é dada pela Equação 23:

$$Fit = \frac{1}{\sum_{i=1}^P \sum_{j=1}^Q H_{ij}}, \quad (23)$$

com P e Q sendo o número de linhas e colunas respectivamente, e

$H_{ij} = \begin{cases} ND_{ij}^* : ND_{ij} \geq ND_{ij}^* \\ ND_{ij} : ND_{ij} < ND_{ij}^* \end{cases}$, sendo ND_{ij}^* o número de padrões de entrada com confusão que ativaram o neurônio e ND_{ij} o número de padrões normais. Se a maior aptidão dos indivíduos de uma geração for maior que o critério de parada para o aprendizado, o indivíduo é selecionado e utilizado para treinar um mapa maior (versão final do SOM).

Para avaliar os resultados obtidos, foram utilizadas três métricas: sensibilidade, especificidade e eficiência, que são dadas pelas Equações 24,25,26 respectivamente.

$$Sensibilidade = \frac{N_{TP}}{N_{TP} + N_{FN}} \times 100 \quad (24)$$

$$Especificidade = \frac{N_{TN}}{N_{FP} + N_{TN}} \times 100 \quad (25)$$

$$Eficiencia = \frac{N_{TP} + N_{TN}}{N_{TP} + N_{FN} + N_{FP} + N_{TN}} \times 100 \quad (26)$$

onde N_{TP} e N_{TF} são os números de verdadeiros e falsos positivos, N_{TN} e N_{FN} são os números de verdadeiros e falsos negativos.

Os resultados mostraram que as redes que tiveram os elementos de detecção selecionados por GAs obtiveram maior eficiência e sensibilidade, especialmente quando os dados de entrada não foram utilizados para treinamento. Porém houve redução na especificidade.

Um resumo do algoritmo pode ser visualizado na Tabela 11:

Jin et al. (2003) afirmam que existem três caminhos principais para se otimizar uma rede SOM:

1. Modificando a estrutura da rede, inserindo ou retirando neurônios da camada de saída.
2. Amenizando o critério de competição, através de um novo termo que inibe os neurônios que vencem com muita frequência ou basear a competição no produto interno.
3. Aprimorando a regra de aprendizado da rede.

Nesse trabalho, os autores integram todos esse mecanismos através de uma nova regra de aprendizado, formando o integrated SOM (ISOM). Um ISOM eficiente deve ter uma

Algoritmo de Agrupamento	SOM utilizando distância Euclidiana
Tipo de Representação	Cromossomos binário contendo 15 genes, representando um dos elementos da tabela CBC.
População Inicial	Aleatória.
Função de Aptidão	A função de aptidão baseada na quantidade de instâncias que cada neurônio da rede recebeu de cada classe. Quanto maior a diferença dos padrões maior é a aptidão
Seleção	Roleta
Cruzamento	Simples
Mutação	Bit a bit (<i>bitwise</i>)
Elistismo	Sim. O cromossomo com menor aptidão é descartado, o cromossomo com maior aptidão é levado para a próxima geração.
Análise dos Resultados	Compara os resultados com SOM utilizando os 15 elementos da tabela CBC.
Detalhes	Com a utilização do GA, autor altera o cálculo da distância Euclidean para receber o vetor de características, fazendo que o calculo seja feito apenas entre as características selecionadas pelo vetor.

Tabela 11: Resumo do GA utilizado por Ohtsuka

boa coordenação entre otimização local dos SOMs tradicionais, a otimização global dos coeficientes expandidos, e o valor de uma constante elástica utilizada para atualizar os pesos da rede. Além disso, uma implementação sob medida do coeficiente expandido e da seleção de parâmetros deve ser determinada.

Depois da inicialização, um algoritmo genético é utilizado para evoluir bons esquemas de aprendizado, representados por indivíduos. O algoritmo concreto de uma rede neural SOM é obtido por meio de um esquema de aprendizado. As redes neurais são, em seguida, usadas para solucionar um número de pequenos problemas para estimar o valor da aptidão do esquema correspondente. O valor de aptidão de cada indivíduos é dado pela função de aptidão da Equação 27.

$$Aptidao = 3 - media(Qualidade) - variancia(Qualidade) \quad (27)$$

sendo que

$$Qualidade = (tamanho - tamanhootimo)/tamanhootimo \quad (28)$$

Os melhores Q indivíduos distintos são armazenados. O processo de evolução interage até que o critério de parada é satisfeito e os esquemas de aprendizado armazenados são analisados em um conjunto de problemas de larga escala. No final, o esquema que obtiver a melhor aptidão nos problemas de validação é retornado como o ISOM evoluído.

Os autores utilizam o algoritmo para solucionar o problema do caixeiro viajante. Os

resultados obtidos são comparados com outras três implementações: o SOM expandido, redes elásticas convexas, e o algoritmo FLEXMAP e conseguiu melhorias na redução do espaço percorrido pelo caixeiro com relação a essas implementações.

Um resumo do algoritmo utilizado por ser visto na Tabela 12.

Algoritmo de Agrupamento	SOM
Tipo de Representação	Inclui o tipo de fórmula e os parâmetros utilizado para calcular o coeficiente expandido da rede. Além disso, ela também inclui alguns parâmetros do ISOM.
População Inicial	Aleatória
Função de Aptidão	Baseado na média e variância da qualidade
Seleção	Roleta
Cruzamento	Utiliza dois tipos de cruzamento: o primeiro é o cruzamento simples e o segundo gera um valor aleatório perto do valor contido entre cada par de alelos e utiliza esse valor no filho.
Mutação	Modifica o valor antigo para um novo dentro do domínio
Elistismo	Os 30 melhores indivíduos são poupados.
Comparação dos Resultados	Compara com o SOM expandido, redes elásticas convexas, e o algoritmo FLEXMAP
Detalhes	Taxa de cruzamento foi de 0.99, de mutação foi de 0.01, a população tinha 100 indivíduos e o número máximo de gerações foi de 6000.

Tabela 12: Resumo do algoritmo ISOM

Outra maneira de utilizar GAs em conjunto com redes SOM é permitir que o GA faça uma seleção dos objetos contidos na base de dados, enquanto as redes SOM modelam as partições. É o caso de problemas complexos como os QSAR/QSPR que consistem em modelar os relacionamentos das propriedades/atividades de descritores nos sistemas biológicos. Bayram et al. (2004) utilizam SOMs para agrupar os objetos da base de dados de acordo com os descritores e a propriedade alvo, gerando um modelo QSAR/QSPR, enquanto o GA seleciona os descritores a serem incluídos no modelo.

Para representar esse modelo, Bayram et al. (2004) utilizam um cromossomo binário, onde o bit 1 indica que o descritor correspondente está inserido no modelo e 0 o contrário. Além disso, um vetor binário contendo a classe a qual cada descritor pertence é anexo ao vetor de descritores para ser utilizado no treinamento supervisionado do SOM.

As características mais importantes do algoritmo utilizado por Bayram et al. (2004) podem ser visualizadas na Tabela 13.

As redes SOM também podem trabalhar com GAs que utilizam o cromossomo para representar os subconjuntos diretamente, uma técnica bastante utilizada com o algoritmo *k-means*. Ma (2005) faz com que alguns padrões sejam representantes de um agrupamento. Um a um, os padrões são adicionados aos agrupamentos com a menor distância Euclidiana entre ele e o seu representante. O cromossomo é composto de um vetor de inteiros de tamanho k , onde k é o número de subconjuntos representados. Cada inteiro contém o valor

Algoritmo de Agrupamento	SOM
Tipo de Representação	Binária, representando os descritores e a classe a que eles pertencem
População Inicial	Aleatória
Função de Aptidão	Percentual da classificação correta dos dados de treino
Seleção	Método da Roleta
Cruzamento	Os cromossomos pais são divididos em partes nas posições de alelos escolhidos aleatoriamente, e depois as partes são trocadas
Mutação	Inverte os bits de alelos selecionados aleatoriamente
Elistismo	5% melhores indivíduos vão para a próxima geração
Comparação dos Resultados	Obteve melhores resultados do que SOM simples e PLS. Apesar disso, o autor afirma que GASOM acabou tendo <i>overfitting</i> aos dados de treino, obtendo pouca generalização.
Detalhes	Utiliza treinamento supervisionado para o SOM

Tabela 13: Resumo do algoritmo GASOM

do padrão que será usado como representante daquele subconjunto (similar a *meioides*). O tamanho da população é o número de padrões dividido pelo número de agrupamentos. Isso tem como objetivo permitir a utilização da maioria dos padrões como representativos.

A aptidão dos indivíduos é obtida por meio da soma dos quadrados das distâncias entre todos os padrões e o representante de agrupamento mais próximo. Esse cálculo é feito utilizando a Equação 29.

$$f(p_i) = \frac{1}{\text{sqrt}(\sum_{n=1}^N ||x(n) - y(x(n))||^2)}, \quad (29)$$

onde p_i é o cromossomo, $x(n)$ são os dados de entrada e $y(x(n))$ denota a inserção do padrão $x(n)$ no agrupamento de representante mais próximo.

Ma (2005) utiliza redes SOM para a detecção de intrusão de sistemas através de arquivos de log do UNIX. Ele teve dificuldades em agrupar esses dados, devido aos diferentes tipos de dados e escalas, que foram divididos em 4 categorias: boolean, string, count e rate. Os resultados obtidos mostram que os resultados finais para esse tipo de representação dependem muito do conjunto inicial de soluções, assim como o algoritmo *k-means*.

Um resumo do algoritmo GSOMC implementado por Ma (2005) pode ser visto na Tabela 14.

5.3 Agrupamentos Hierárquicos

Algoritmos hierárquicos resultam em uma sequência aninhada de partições, ou seja, dois exemplos agrupados em um nível continuam fazendo parte do mesmo grupo nos

Algoritmo de Agrupamento	SOM utilizando distância Euclidiana
Tipo de Representação	Cada alelo representa o meioíde de um agrupamento
População Inicial	Aleatório. Porém, dois indivíduos devem ser diferentes
Função de Aptidão	Soma dos quadrados das distâncias entre todos os padrões e o representante de agrupamento mais próximo.
Seleção	Não especificada
Cruzamento	Simples
Mutação	Substitui um representante por outro que não tenha sido utilizado na população inicial, para permitir a utilização de novos representantes.
Elistismo	Não especificado
Comparação dos Resultados	Não especificada
Detalhes	Pára depois que a população convergir ou o algoritmo executar um determinado número de gerações.

Tabela 14: Resumo do Algoritmo GSOMC

níveis mais altos, formando um dendograma (Figura 1). Assim como foi dito na Seção 2, agrupamentos hierárquicos podem ser divididos em duas abordagens: a aglomerativa e a divisiva. A abordagem aglomerativa começa com n subconjuntos com um único exemplo cada e forma a seqüência de partições agrupando os subconjuntos sucessivamente. A abordagem divisiva começa com um subconjunto com todos os exemplos e forma a seqüência dividindo os subconjuntos sucessivamente.

Existem várias formas de utilizar GAs em conjunto com algoritmo hierárquicos. Uma abordagem consiste em utilizar um método que divida o conjunto de dados em m subconjuntos iniciais e depois agrupá-los de maneira aglomerativa. Apesar do algoritmo utilizado por Tseng & Yang (2001) não ser hierárquico por natureza (não forma dendograma), os autores formam os subconjuntos de maneira aglomerativa. Primeiramente, é calculada a distância do vizinho mais próximo para cada par de objetos $d(o_i) = \min_{j \neq i} \|o_j - o_i\|$ e em seguida a média dessas distâncias d_{av} . Os objetos são vistos como nós de um grafo e os elementos matriz de adjacência A são calculados conforme a Equação 30.

$$A(i, j) = \begin{cases} 1 & \text{se } \|o_i - o_j\| \leq d_{av}, \\ 0 & \text{se } \|o_i - o_j\| > d_{av} \end{cases} \quad (30)$$

Através do grafo formado é possível encontrar os componentes conectados e formar m subconjuntos $\{B_1, B_2, \dots, B_m\}$ contendo esses componentes. Esses subconjuntos são representados por cromossomos compostos por vetores binários de tamanho m . Os valores contidos nesses vetores são determinados aleatoriamente no início da primeira geração. Se o valor do bit correspondente a um subconjunto for 1, esse subconjunto continuará a existir na partição. Caso o valor do bit correspondente for 0, o subconjunto deixará de existir e os objetos contidos nele serão adicionados à outros agrupamentos cujo valor indicado pelo cromossomo seja 1. A Figura 8 da Seção 4.1 ilustra esse tipo de representação.

Em seguida, esses cromossomos recebem uma nota de aptidão baseadas nas distâncias *inter* e *intra* agrupamento, como descrito na Seção 4.2. Um par de cromossomos é

selecionado, a partir da sua aptidão, e são divididos em dois pontos aleatórios e o segmento entre os pontos é trocado entre os cromossomos com probabilidade de cruzamento p_c . A mutação é feita bit-a-bit com probabilidade p_m .

Tseng & Yang (2001) utilizaram o algoritmo CLUSTERING em 3 bases de dados, sendo 2 delas artificiais e 1 real. Em seguida, os resultados obtidos foram comparados com os algoritmos *k-means*, *single-link* (Dubes & Jain 1976) e *complete-link* (Dubes & Jain 1976). O algoritmo obteve bons resultados ao agrupar bases de dados com subconjuntos de diferentes densidades, ao contrário dos métodos *single-link* e *complete-link*. A utilização do algoritmo CLUSTERING resultou em subconjuntos cuja média da distância do centro é menor, quando o número de subconjuntos obtidos pelos algoritmos foi o mesmo. O algoritmo *k-means* não obteve bons resultados na maioria dos experimentos, mesmo quando o número de agrupamentos era previamente conhecido.

Um resumo com as principais características do algoritmo CLUSTERING pode ser visto na Tabela 15.

Algoritmo de Agrupamento	Agrupamento por GA utilizando aglomeração
Tipo de Representação	Binária, representando os agrupamentos iniciais
População Inicial	Aleatória
Função de Aptidão	Baseadas nas distâncias <i>inter</i> e <i>intra</i> agrupamento
Seleção	Não especificada
Cruzamento	Duplo. Dois pontos aleatórios e o segmento entre os pontos é trocado
Mutação	bit-a-bit
Elistismo	O melhor indivíduo é copiado
Comparação dos Resultados	<i>k-means</i> , <i>single-link</i> e <i>complete-link</i>
Detalhes	Utiliza método aglomerativo para formar os subconjuntos

Tabela 15: Resumo do Algoritmo CLUSTERING

Tseng & Yang (2001) utilizaram um método aglomerativo para formar os subconjuntos durante a execução do agrupamento. Entretanto, no algoritmo utilizado não é possível visualizar os vários níveis de partições, apenas a partição final. Greene (2003) utiliza um algoritmo hierárquico divisivo que, ao contrário do algoritmo CLUSTERING, permite visualizar as partições formadas anteriormente. Cada indivíduo é composto de um único subconjunto contendo todos os dados da base. Um subconjunto pode ser formado de outros subconjuntos, o que o caracteriza como um subconjunto particionado. Um subconjunto particionado pode ser formando de outros subconjuntos particionados, formando assim uma hierarquia.

A implementação dos subconjuntos utiliza orientação a objetos, ou seja, cada subconjunto é uma instância da classe subconjunto. Os subconjuntos contêm a lista de características dos objetos que nele estão contidos. Além disso, cada subconjunto possui os atributos “raio atual”, que consiste na distância máxima de qualquer elemento ao centróide do subconjunto, e “raio máximo”, que consiste no raio atual máximo que um subconjunto pode ter. O raio máximo que um subconjunto filho pode ter é sempre uma

fração do raio do subconjunto pai, porém sempre maior que uma constante previamente determinada. O tamanho do raio máximo dos subconjuntos filhos influencia na quantidade de subconjuntos que um subconjunto pai pode possuir, como pode ser visto mais a diante.

Ao inicializar a população, Greene (2003) organiza os padrões da base de dados de maneira aleatória e seleciona o primeiro padrão que é adicionado ao primeiro subconjunto. Os outros padrões são adicionados no subconjunto na seqüência em que foram organizados, podendo causar o particionamento do subconjunto caso seu raio atual seja maior que o raio máximo dos subconjuntos filhos a serem construídos. Se o subconjunto que recebe o novo padrão é particionado, o padrão é inserido no subconjunto filho com centróide mais próximo (sorteio no caso de empate). Porém, se o raio máximo dos subconjuntos filhos for maior ou igual ao raio atual do subconjunto pai, a estrutura é descartada pois os subconjuntos filhos podem sobrepor o subconjunto pai. A adição de padrões pode ser vista por meio do seguinte pseudo-código:

```
// Função de adiciona um novo padrão no subconjunto de dados

Funcao AdicionarPadrao(novo_padrao){
  Adiciona(novo_padrao) //adiciona o novo padrão ao subconjunto
  CalculaCentroide() //calcula o novo centróide
  CalculaRaioAtual() //calcula o novo valor do raio atual
  CalculaAptidao() //calcula a aptidão do subconjunto

  Se(for_particionado) //se o subconjunto for particionado

  Se(raio_atual <= raio_maximo_filho){
    DescartaEstrutura() // a subestrutura é descartada
    Senão //continua particionado
    AdicionaPadraoFilho() //adiciona o padrão em um filho

  Senão // o subconjunto não é particionado

  Se(raio_atual > raio_maximo_filho)
  ParticionaSubconjunto() //particiona o subconjunto em dois

} //Fim da função
```

O processo é repetido até que todos os padrões sejam adicionados ao subconjunto principal. Essa função também permite adicionar novos padrões a hierarquia, mesmo depois de a partição final ser encontrada.

GAs são utilizados para selecionar uma população de subconjuntos. Como pode ser visto no pseudo código acima, a aptidão dos subconjuntos é calculada durante o processo de adição de padrões. A aptidão dos indivíduos é calculada de duas maneiras distintas. Se o conjunto for particionado, a função de aptidão é calculada pela soma das médias de aptidão dos subconjuntos filhos e a distância média entre os centróides desses subconjuntos.

Caso ele não seja particionado, a aptidão é calculada pela soma dos elementos contidos no agrupamento e a diferença entre o raio atual e o raio máximo. Os dois indivíduos com as maiores taxas de aptidão são copiados para a próxima geração. Os outros indivíduos são selecionados pelo método da roleta e serão aplicados operadores de cruzamento e mutação. Para isso aptidão dos indivíduos é linearmente transformada em valores reais no intervalo [1,4]. Tomando a aptidão transformada do indivíduo m_i como f_i , a probabilidade do indivíduo de posição k ser selecionado é dada por $f_k/(\sum_i f_i)$. Os operadores de cruzamento e mutação aplicados são descritos na Seção 4.3.

Greene (2003) aplica o algoritmo em dados com características nominais, ou seja, características cujos valores pertencem a um conjunto discreto simples, sem nenhuma estrutura adicional (cor,sexo,etc). Os experimentos foram realizados com bases de dados do repositório da *UC-Irvine* para aprendizado de máquina. Os objetos contidos nas bases foram agrupados corretamente. O autor ainda não comparou os resultados com outros algoritmos utilizados na base porque afirma não ter detalhes suficientes para fazê-la. As principais características do algoritmo podem ser vistas na Tabela 16.

Algoritmo de Agrupamento	Agrupamento Hierárquico Divisivo
Tipo de Representação	Objetos representam os subconjuntos
População Inicial	Aleatória
Função de Aptidão	Baseadas na aptidão dos subconjuntos filhos
Seleção	Roleta
Cruzamento	Guloso, direcionado pela aptidão dos subconjuntos que formam o indivíduo
Mutação	Consiste em retirar um subconjunto e adicioná-lo posteriormente. Baseada na aptidão dos indivíduos
Elistismo	O dois melhores indivíduos são copiados
Comparação dos Resultados	Nenhuma comparação foi feita
Detalhes	Representa vários níveis de partições e é incremental

Tabela 16: Resumo do Algoritmo utilizado por Greene

Uma outra implementação consiste em utilizar o algoritmo hierárquico aglomerativo de Spellman (Eisen et al. 1998) em conjunto com GAs. Ma & Chan (2003) utiliza essas técnicas para agrupar genes coexpressos através de dados de expressão gênica. Essas técnicas são utilizadas para tratar o ruído nos dados, obtidos de origem biológica (durante a análise biológica das seqüências) e técnica (durante a extração dos dados). As técnicas de agrupamento ajudam a identificar as estruturas reais dos dados, enquanto o GA efetua uma pesquisa global pelas melhores partições no espaço de busca.

Assim como nos algoritmos hierárquicos apresentados anteriormente, é necessário gerar partições iniciais que possam ser representadas pelos indivíduos da primeira geração. Nessa implementação o primeiro conjunto de partições é gerado com as partições resultantes da utilização do algoritmo de Spellman no conjunto de dados. Cada cromossomo é composto de duas partes: a primeira parte é uma representação grupo-número dos objetos e a segunda parte é uma seqüência contendo os números dos grupos. A similaridade entre os objetos é medida pela correlação de Pearson durante a construção dos subconjuntos. A função de aptidão utilizada é dividida em dois passos: o primeiro consiste em descobrir

associação de padrões no conjunto de treino que sejam estatisticamente significantes e o segundo consiste em classificar todos os objetos contidos no conjunto de teste baseado nas associações descobertas. A aptidão de cada indivíduo é utilizada para a seleção pela Roleta.

O cruzamento consiste no seguinte processo: um subconjunto é selecionado aleatoriamente do primeiro pai e substituído pelo subconjunto com o maior número de objetos em comum no segundo pai. Os objetos do subconjunto do primeiro pai que estiverem em outros subconjuntos no segundo, são apagados desses subconjuntos. Os objetos que sobraram do subconjunto substituído são re-allocados para novos subconjuntos, utilizando o algoritmo de Spellman. Os autores utilizam dois tipos de mutação, com chances iguais. No primeiro são selecionados 30% dos objetos para formar um conjunto de teste e os 70% restantes são utilizados para treino. Os objetos no conjunto de teste são re-agrupados em novos conjuntos utilizando o algoritmo hierárquico de Spellman. No segundo são selecionados 30% dos objetos de cada subconjunto e são alocados em novos subconjuntos aleatoriamente.

A cada nova geração, apenas os dois indivíduos menos aptos são substituídos e os cromossomos filhos que sejam duplicatas de cromossomos que já existam na população são descartados. Logo, os cromossomos pais e filhos podem coexistir na mesma geração.

Os detalhes do algoritmo utilizado estão descritos na Tabela 17.

Algoritmo de Agrupamento	Agrupamento Hierárquico Spellman
Tipo de Representação	Grupo-número contendo os números dos grupos
População Inicial	Gerada pelas partições formadas pelo algoritmo de Spellman
Função de Aptidão	Consiste em descobrir associação de padrões estatisticamente significantes e classificar todos os objetos contidos no conjunto de teste baseado nessas associações
Seleção	Roleta
Cruzamento	Um subconjunto é selecionado aleatoriamente de um dos pais e substituído pelo subconjunto com mais objetos em comum do outro
Mutação	Consiste em retirar 30% dos objetos dos subconjuntos e adicioná-los posteriormente
Elistismo	Apenas os dois piores indivíduos são substituídos
Comparação dos Resultados	Apesar de vários testes terem sido aplicados, nenhuma comparação foi feita
Detalhes	Mesmo utilizando algoritmo hierárquico, os indivíduos representam apenas um nível de partição

Tabela 17: Resumo do Algoritmo utilizado por Ma

Garai & Chaudhuri (2004) apresenta um algoritmo de agrupamento guiado por GA (GCA) que, como o algoritmo de Tseng & Yang (2001), consiste em dividir a base de dados em subconjuntos iniciais e depois aglomera-los. Porém o algoritmo utilizado por

Tseng & Yang (2001) falha se o agrupamentos contém subconjuntos parcialmente ou completamente confinados em outro subconjunto. Para solucionar esse problema, Garai & Chaudhuri (2004) utilizam um algoritmo de verificação de adjacência nos subconjuntos (*Adjacent Cluster Checking Algorithm* - ACCA)

A primeira parte do algoritmo consiste em dividir os objetos do conjunto de dados em k subconjuntos iniciais utilizando o CDA (*Cluster Decomposition Algorithm*), que é constituído dos seguintes passos:

1. Dado um conjunto de dados χ com n padrões. Para cada padrão $x_i, \forall i \in \{1, 2, \dots, n\}$, é calculada a distância do vizinho mais próximo utilizando a Equação 32, de maneira que $j \in \{1, 2, \dots, n\}$ e $\|x_i - x_j\|$ é a distância euclidiana entre os dois padrões (Equação 3).

$$d_{min}(x_i) = \min_{i \neq j} \|x_i - x_j\| \quad (31)$$

2. É calculada a distância mínima média d_{med} através da Equação 32:

$$d_{med} = \frac{1}{n} \sum_{i=1}^n d_{min}(x_i) \quad (32)$$

3. Considere x_i como o centro de uma região circular de raio r , dado pela Equação 33, que contém um grupo de padrões do conjunto de dados. São gerados os subconjuntos $\{B_1, B_2, \dots, B_k\}$, contendo os padrões dessa região circular, com $k \leq n$ e $B_i \cap B_j = \emptyset, \forall i \neq j$ e $i, j \in \{1, 2, \dots, k\}$. A constante real u tem seu valor escolhido previamente, geralmente no intervalo $[2, 4]$.

$$r = u * d_{med} \quad (33)$$

4. Faça $p = 1$.
5. Retire B_p do conjunto de dados χ de maneira que $\chi = |\chi - B_p|$.
6. Termina o algoritmo se $|\chi| = \emptyset$. Caso contrário, $p = p + 1$ volte ao passo 5.

Em seguida os subconjuntos iniciais $\{B_1, B_2, \dots, B_k\}$ são representados por vetores binários, como descrito na Seção 4.1. Os subconjuntos representados por 0 serão adicionados pelo algoritmo HCMA (*Hierarchical Cluster Merging Algorithm*) aos subconjuntos mais próximos representados por 1. O centróide do subconjunto formado é calculado pela soma dos centróides dos subconjuntos que o compõem. O algoritmo ACCA é utilizado antes do HCMA para evitar sejam gerados subconjuntos confinados em outro subconjunto ou evitar ruídos nos subconjuntos. Esse algoritmo utiliza os limiares T_b e T_d , que representam o número mínimo de objetos em comum aos subconjuntos a serem aglomerados e a diferença de densidade máxima entre os subconjuntos, respectivamente. O algoritmo ACCA consiste nos seguintes passos:

1. Define o raio r , dado pela Equação 33.
2. Seleciona dois subconjuntos iniciais B^0 e B^1 que sejam representados por 0 e 1 respectivamente e satisfaçam as condições para serem aglomerados (sejam os mais próximos).
3. Conte o número de pontos em comum entre B^0 e B^1 que estejam dentro da área formada pelos respectivos raios r . Deixe que seja representado por N_b e a densidade de objetos dos subconjuntos B^0 e B^1 seja D^1 e D^0 respectivamente.
4. Se $N_b \geq T_b$ e $abs(D^0 - D^1) \leq T_d$ então B^0 e B^1 são adjacentes um ao outro e são aglomerados.

A aptidão é calculada utilizando as distâncias *inter* e *intra* agrupamento, descritas na Seção 4.2. Em seguida, os cromossomos são selecionados pelo método da roleta e são aplicados os operadores genéticos de cruzamento simples e mutação bit-a-bit. O algoritmo executa até que um número k de subconjuntos é encontrado ou depois de G_{max} gerações. O algoritmo foi aplicado em vários tipos de dados artificiais, contendo subconjuntos confinados e com ruídos, e também foi aplicado na base de dados Íris e obteve bons resultados. Um resumo com as principais características do algoritmo pode ser visualizado na Tabela 18.

Algoritmo de Agrupamento	Agrupamento por GA utilizando aglomeração
Tipo de Representação	Binária, representando os agrupamentos iniciais
População Inicial	Aleatória
Função de Aptidão	Baseadas na distâncias <i>inter</i> e <i>intra</i> agrupamento
Seleção	Roleta
Cruzamento	Simplex
Mutação	bit-a-bit
Elistismo	Não especificado
Comparação dos Resultados	CURE, DBScan e Chameleon
Detalhes	Checa a adjacência dos subconjuntos antes de aglomerar

Tabela 18: Resumo do Algoritmo GCA

6 Considerações Finais

Este relatório apresentou uma visão geral sobre agrupamento de dados utilizando GAs, enfocando as diversas etapas do processo representação e aplicação dos operadores genéticos para técnicas de agrupamento e descrevendo alguns dos algoritmos existentes, fornecendo os elementos necessários para a pesquisa em combinação de agrupamentos com GAs proposta neste trabalho.

Vários métodos de utilização de GAs em conjunto com algoritmos de agrupamentos foram descritos durante o relatório. Muitas das implementações utilizam GAs tanto para a

escolha de bons subconjuntos iniciais, quando para encontrar bons valores para parâmetros livres.

Um fato relevante a ser citado é que o número de campos e os valores usados para os parâmetros de um agrupamento genético variam consideravelmente com o tipo de algoritmo de agrupamento utilizado, e há pouca documentação justificando a utilização de um valor específico para eles. Dentre esses parâmetros, podem ser citados os parâmetros utilizados pelo próprio algoritmo de agrupamento, taxas de cruzamento, mutação, além de outros.

Os parâmetros utilizados pelos agrupamentos podem ser otimizados pelo GA, assim como os centróides iniciais. Para isso, eles devem ser representados no cromossomo, variando conforme a representação utilizada. Algumas vezes, devido ao tamanho do cromossomo, a probabilidade de alteração desses parâmetros é baixa. Operadores genéticos podem ser utilizados para alterar apenas o segmento que contém os valores dos padrões.

Referências

- Balakrishnan, K., & V. Honavar (1995). Evolutionary Design of Neural Architectures: A Preliminary Taxonomy and Guide to Literature. Technical report, Department of Computer Science, Iowa State University, Ames, Iowa.
- Barbara, D. (2000). An introduction to cluster analysis for data mining. http://www-users.cs.umn.edu/~han/dmclass/cluster_survey_10_02_00.pdf [Acessado em 03/04/2005].
- Bayram, E., P. S. II, R. Harris, Y.-D. Xiao, A. J. Clauset, & J. D. Schmitt (2004). Genetic algorithms and self-organizing maps: a powerful combination for modeling complex qsar and qspr problems. In *Journal of Computer-Aided Molecular Design*, Volume 18, pp. 483 – 493. Springer Science+Business Media B.V., Formerly Kluwer Academic Publishers B.V.
- Beasley, D., D. R. Bull, & R. R. Martin (1993). An overview of genetic algorithms: Part 2, research topics. *University Computing* 15(4), 170–181.
- Belew, R. K., & L. B. Booker (Eds.) (1991). *Solving Partitioning Problems with Genetic Algorithms*. Morgan Kaufmann.
- Bezdek, J. C., S. Boggavaparu, L. O. Hall, & A. Bensaid (1994). Genetic algorithm guided clustering. *Proceedings of the First IEEE Conference on Evolutionary Computation, IEEE World Congress on Computational Intelligence, 27-29 June 1*, 34–49.
- Cole, R. M. (1998). *Clustering with Genetic Algorithms*. Ph. D. thesis, Department of Computer Science, University of Western Australia.
- Cowgill, M. C., R. J. Harvey, & L. T. Watson (1998). A genetic algorithm approach to cluster analysis. Technical report, Virginia Polytechnic Institute & State University, Blacksburg, VA, USA.
- Dubes, R., & A. K. Jain (1976). Clustering techniques: the user's dilemma. *Pattern Recognition* 8, 247 – 260.
- Duda, R., P. Hart, & D. Stork (2001). *Pattern Classification*. John Wiley & Sons.
- Eisen, M. B., P. Spellman, P. Brown, & D. Botstein (1998). Cluster analysis and display of genome-wide expression patterns. In *Proc. Natl. Acad. Sci. USA*, Volume 95, pp. 14863–14868.
- Estivill-Castro, V. (2002). Why so many clustering algorithms - a position paper. *SIGKDD Explorations* 4(1), 65–75.
- Fränti, P., T. Kaukoranta, & O. Nevalainen (1997). On the splitting method for vq codebook generation. *Optical Engineering* 36, 3043–3051.
- Fränti, P., & J. Kivijärvi (2000). Randomised local search algorithm for the clustering problem. *Pattern Analysis and Applications* 3, 358–369.
- Fränti, P., J. Kivijärvi, T. Kaukoranta, & O. Nevalainen (1997). Genetic algorithms for large scale clustering problems. *The Computer Journal* 40, 547–554.

- Garai, G., & B. Chaudhuri (2004). A novel genetic algorithm for automatic clustering. *Pattern Recognition Letters* 25, 173–187.
- Greene, W. A. (2003). Unsupervised hierarchical clustering via a genetic algorithm. In *Proceedings of the 2003 Congress on Evolutionary Computation*, pp. 998–1005. IEEE Press.
- Halkidi, M., Y. Batistakis, & M. Vazirgiannis (2001). On clustering validation techniques. *Intelligent Information Systems Journal* 17(2-3), 107–145.
- Hall, L., B. Ozyurt, & J. Bezdek (1999). Clustering with a genetically optimized approach. In *Transactions on Evolutionary Computation*, Volume 3, Department of Computer Science and Engineering, University of South Florida, Department of Computer Science and Engineering, University of West Florida, pp. 103–112. IEEE.
- Haykin, S. (1999). *Neural Networks: A Comprehensive Foundation*. Prentice Hall.
- Hruschka, E. R., R. J. G. B. Campello, & L. N. de Castro (2004). Improving the efficiency of a clustering genetic algorithm. In *Advances in Artificial Intelligence - IBERAMIA 2004: 9th Ibero-American Conference on AI, Puebla, Mexico, November 22-25. Proceedings*, Volume 3315, pp. 861. Springer-Verlag GmbH, Lecture Notes in Computer Science.
- Jain, A. K., M. N. Murty, & P. J. Flynn (1999). Data clustering: a review. *ACM Computing Surveys* 31(3), 264–323.
- Jiang, D., C. Tang, & A. Zhang (2003). Cluster analysis for gene expression data: A survey. *IEEE Transactions on Knowledge and Data Engineering*.
- Jin, H.-D., K.-S. Leung, M.-L. Wong, & Z.-B. Xu (2003). An efficient self-organizing map designed by genetic algorithms for the traveling salesman problem. In *Systems, Man and Cybernetics, Part B, IEEE Transactions on*, Volume 33, pp. 877 – 888. IEEE.
- Johnson, R. A., & D. W. Wichern (1982). *Applied multivariate statistical analysis*. Prentice Hall, Englewood Cliffs, NJ.
- Kaufman, L., & P. Rousseeuw (1990). Finding groups in data: An introduction to cluster analysis. *Wiley Series in Probability and Mathematical Statistics*.
- Kivijärvi, J., P. Fränti, & O. Nevalainen (2003). Self-adaptive genetic algorithm for clustering. In *Journal of Heuristics*, Volume 9, pp. 113 – 129. Springer Science + Business Media B.V., Formerly Kluwer Academic Publishers B.V.
- Ma, P., & K. Chan (2003). Discovering clusters in gene expression data using evolutionary approach. *Proceedings on 15th IEEE International Conference on Tools with Artificial Intelligence* 3-5, 459 – 466.
- Ma, Z. (2005). A genetic som clustering algorithm for intrusion detection. In *Advances in Neural Networks - ISNN 2005: Second International Symposium on Neural Networks, Chongqing, China, May 30*, Volume 3498, Department of Computer Science and Engineering, Chongqing University, Chongqing 400044, China, pp. 421. Lecture Notes in Computer Science, Springer-Verlag GmbH.
- Maulik, U., & S. Bandyopadhyay (2000). Genetic algorithm-based clustering technique. *Pattern Recognition* 33, 1455 – 1465.

- Mitchell, M. (1999). *An introduction to Genetic Algorithms*. MIT Press.
- Mitchell, T. (1997). *Machine Learning*. McGraw Hill.
- Murthy, C. A., & N. Chowdhury (1996). In search of optimal clusters using genetic algorithms. *Pattern Recogn. Lett.* 17(8), 825 – 832.
- Ohtsuka, A., N. Kamiura, T. Isokawa, & N. Matsui (2002). On detection of confused blood samples using self organizing maps and genetic algorithm. In *Neural Information Processing, 2002. ICONIP '02. Proceedings of the 9th International Conference on*, Volume 5, Department of Computer Science and Illinois Genetic Algorithms Laboratory, pp. 2233 – 2238.
- Pan, H., J. Zhu, & D. Han (2003). Genetic algorithms applied to multi-class clustering for gene expression data. In *Genomics, Proteomics and Bioinformatics*, Volume 1, pp. 279–287.
- Raghavan, V. V., & K. Birchand (1979). A clustering strategy based on formalism of reproductive process in a natural system. In *Proceedings of the Second International Conference on Information Storage and Retrieval*, pp. 10–22.
- Tanaka, M., Y. Furukawa, & T. Tanino (1996). Weight tuning and pattern classification by self organizing map using genetic algorithm. In *Proceedings of IEEE International Conference on Evolutionary Computation*, pp. 602 – 605. May, 20–22.
- Tseng, L., & . Yang, S.B. (2001). A genetic approach to the automatic clustering problem. *Pattern Recognition* 34, 415–424.
- Ward, J. H. (1963). Hierarchical grouping to optimize an objective function. *Journal of the American Statistical Association* 58, 236–244.
- Wu, F.-X., W. J. Zhang, & A. J. Kusalik (2003). A genetic k-means clustering algorithm applied to gene expression data. In *Advances in Artificial Intelligence: 16th Conference of the Canadian Society for Computational Studies of Intelligence, AI 2003, Halifax, Canada, June 11-13. Proceedings*, Volume 2671, pp. 520 – 526. Springer-Verlag GmbH.
- Zeger, K., & A. Gersho (1989). Stochastic relaxation algorithm for improved vector quantiser design. *Electronics Letters* 25, 896–898.
- Zeng, Y., J. Tang, J. Garcia-Frias, & G. Gao (2002). An adaptive meta-clustering approach: Combining the information from different clustering results. In *IEEE Computer Society Bioinformatics Conference (CSB'02)*, Stanford, California, pp. 276.