Um *framework* para análise de
agrupamento baseado na combinação
multi-objetivo de algoritmos de
agrupamento

Katti Faceli

Um *framework* para análise de agrupamento baseado na
combinação multi-objetivo de algoritmos de
agrupamento

Katti Faceli

Orientador: *Prof. Dr. André Carlos Ponce de Leon Ferreira de Carvalho*

Tese apresentada ao Instituto de Ciências Matemáticas e de
Computação - ICMC-USP, como parte dos requisitos para
obtenção do título de Doutor em Ciências - Ciências de
Computação e Matemática Computacional.

“VERSÃO REVISADA APÓS A DEFESA”

Data da Defesa:	08/11/2006
Visto do Orientador:	

**USP – São Carlos
Janeiro/2007**

*À minha sobrinha querida, Amanda, que é sempre fonte de harmonia e carinho.
Às amigas Flávia, Solange e Renata, que, longe ou perto, estão sempre no meu caminho
me apoiando nos momentos difíceis e compartilhando dos momentos alegres.*

Agradecimentos

Agradeço a vida pela grande oportunidade de realizar este trabalho.

Agradeço à minha família, pelo amor sempre presente e também pelo apoio incondicional em todos os momentos. Em especial, aos meus pais, João e Elza, meu avô José, meu irmão Roberto, minha irmã Simoni, sempre disposta a me ajudar em tudo, e minha sobrinha Amanda.

Agradeço ao meu orientador André, por toda a dedicação durante todos esses anos. Ele representa sempre um exemplo a ser seguido, pelo excelente trabalho que realiza, tanto como pesquisador, quanto como orientador, sempre guiando seus alunos e os apoiando.

Agradeço também ao professor Marcílio que, juntamente com o professor André, contribuiu significativamente para o desenvolvimento deste trabalho. Agradeço pelas valiosas discussões e críticas construtivas e toda a orientação.

Agradeço às professoras Carolina e Solange, que também representaram exemplos a serem seguidos, contribuindo em diversos momentos com valiosas dicas. Além de toda a ajuda e inspiração profissional, agradeço-as também pela amizade.

Aos meus amigos Ana Carolina, Dimas, Edson, Eduardo, Flávia, Huei, Patrícia, Renata, Richardson e Ronaldo, por todos os bons momentos compartilhados, que dão forças para as realizações. Também agradeço ao apoio nos momentos difíceis.

Sem mencionar nomes, agradeço a todos os colegas de laboratório que em algum momento contribuíram com este trabalho, seja com discussões sobre os temas do trabalho, seja compartilhando seus algoritmos e dados, seja dando aquela mãozinha com os códigos e textos que não compilam ou máquinas que não funcionam.

Agradeço ao professor Mário, sempre disposto a ajudar nas questões relacionadas a estatística.

Agradeço também as secretárias e demais funcionários do ICMC que, com seu trabalho eficiente e dedicado, tornaram bastante agradável a interação em todos os momentos necessários.

Finalmente, agradeço à FAPESP pelo apoio financeiro ao projeto.

Resumo

Esta Tese apresenta um *framework* para análise exploratória de dados via técnicas de agrupamento. O objetivo é facilitar o trabalho dos especialistas no domínio dos dados. O ponto central do *framework* é um algoritmo de *ensemble* multi-objetivo, o algoritmo MOCLE, complementado por um método para a visualização integrada de um conjunto de partições. Pela aplicação conjunta das idéias de *ensemble* de agrupamentos e agrupamento multi-objetivo, o MOCLE efetua automaticamente importantes passos da análise de agrupamento: executa vários algoritmos conceitualmente diferentes com várias configurações de parâmetros, combina as partições resultantes desses algoritmos e seleciona as partições com os melhores compromissos de diferentes medidas de validação. MOCLE é uma abordagem robusta para lidar com diferentes tipos de estrutura que podem estar presentes em um conjunto de dados. Ele resulta em um conjunto conciso e estável de estruturas alternativas de alta qualidade, sem a necessidade de conhecimento prévio sobre os dados e nem conhecimento profundo em análise de agrupamento. Além disso, para facilitar a descoberta de estruturas mais complexas, o MOCLE permite a integração automática de conhecimento prévio de uma estrutura simples por meio das suas funções objetivo. Finalmente, o método de visualização proposto permite a observação simultânea de um conjunto de partições. Isso ajuda na análise dos resultados do MOCLE.

Abstract

This Thesis presents a framework for exploratory data analysis via clustering techniques. The goal is to facilitate the work of the experts in the data domain. The core of the framework is a multi-objective clustering ensemble algorithm, the MOCLE algorithm, complemented by a method for integrated visualization of a set of partitions. By applying together the ideas of clustering ensemble and multi-objective clustering, MOCLE automatically performs important steps of cluster analysis: run several conceptually different clustering algorithms with various parameter configuration, combine the partitions resulting from these algorithms, and select the partitions with the best trade-offs for different validation measures. MOCLE is a robust approach to deal with different types of structures that can be present in a dataset. It results in a concise and stable set of high quality alternative structures, without the need of previous knowledge about the data or deep knowledge on cluster analysis. Furthermore, in order to facilitate the discovery of more complex structures, MOCLE allows the automatic integration of previous knowledge of a simple structure via their objective functions. Finally, the visualization method proposed allows the simultaneous observation of a set of partitions. This helps in the analysis of MOCLE results.

Este documento foi preparado com o formatador de textos $\text{\LaTeX}$, com estilo elaborado por Ronaldo Cristiano Prati. O sistema de citações de referências bibliográficas utiliza o padrão *Chicago* do sistema $\text{\bibTeX}$.

Sumário

Dedicatória	i
Agradecimentos	iii
Resumo	v
Abstract	vii
Sumário	xi
Lista de Abreviaturas	xv
Notação	xvii
Lista de Figuras	xx
Lista de Tabelas	xxii
1 Introdução	1
1.1 Contextualização	1
1.2 Motivação	2
1.3 Abordagem Proposta	6
1.4 Organização do Trabalho	8
2 Algoritmos Genéticos Multi-objetivo Baseados em Pareto	9
2.1 Considerações Iniciais	9
2.2 Otimização Multi-objetivo	9
2.3 Algoritmos Evolutivos	12
2.4 Algoritmos de Interesse	16
2.5 Considerações Finais	20
3 Agrupamento de Dados	21
3.1 Considerações Iniciais	21

3.2	Definições	21
3.3	Algoritmos Usados	26
3.3.1	Algoritmos Hierárquicos - Ligação Simples e Ligação Média	28
3.3.2	k -médias	30
3.3.3	<i>Shared Nearest Neighbor</i> (SNN)	31
3.4	Validação de Agrupamentos	33
3.5	Agrupamento Semi-supervisionado	36
3.6	Considerações Finais	38
4	<i>Ensembles</i> e Agrupamento Multi-objetivo	39
4.1	Considerações Iniciais	39
4.2	<i>Ensembles</i> de Agrupamentos	40
4.2.1	Geração dos Agrupamentos Iniciais	41
4.2.2	Determinação da Função Consenso	43
4.2.3	Técnicas de Interesse	46
4.3	Agrupamento Multi-objetivo	53
4.4	Considerações Finais	56
5	Abordagem Proposta	57
5.1	Considerações Iniciais	57
5.2	Descrição do Problema	57
5.3	Metas a Serem Atingidas	63
5.4	<i>Ensemble</i> Multi-objetivo - MOCLE	64
5.5	Implementação da Proposta	67
5.6	Considerações Finais	71
6	Método de Visualização	73
6.1	Considerações Iniciais	73
6.2	Descrição do Método	74
6.3	Utilização do Método de Visualização	76
6.4	Considerações Finais	80
7	Métodos e Experimentos	81
7.1	Considerações Iniciais	81
7.2	Conjuntos de Dados	81
7.3	Experimentos	89
7.4	Metodologia de Avaliação dos Experimentos	92
7.5	Considerações Finais	95

8	Resultados	97
8.1	Considerações Iniciais	97
8.2	Qualidade Geral das Soluções	98
8.3	Análise Detalhada	114
8.3.1	Comparação do MOCLE com os Algoritmos Individuais	118
8.3.2	Comparação do MOCLE com as Outras Técnicas de Combinação	126
8.4	Aplicação da Visualização	137
8.5	Considerações Finais	141
9	Conclusão	145
9.1	Considerações Iniciais	145
9.2	Principais Resultados	146
9.3	Contribuições do Trabalho	148
9.4	Limitações	149
9.5	Trabalhos Futuros	150
	Referências	161

Lista de Abreviaturas

<i>AGs:</i>	Algoritmos Genéticos
<i>ALL:</i>	<i>Acute Lymphoblastic Leukemia</i>
<i>AML:</i>	<i>Acute Myeloid Leukemia</i>
<i>B – ALL:</i>	ALL de linhagem B
<i>CR:</i>	<i>Corrected Rand</i>
<i>CSPA:</i>	<i>Cluster-based Similarity Partitioning Algorithm</i>
<i>DBSCAN:</i>	Algoritmo <i>Density-Based Spatial Clustering of Applications with Noise</i>
<i>EAs:</i>	Algoritmos Evolutivos
<i>EM:</i>	Algoritmo <i>Expectation Maximization</i>
<i>ES:</i>	<i>Ensemble</i> de Strehl e Ghosh
<i>HBGF:</i>	<i>Hybrid Bipartite Graph Formulation</i>
<i>HGPA:</i>	<i>HiperGraph-Partitioning Algorithm</i>
<i>KM:</i>	Algoritmo <i>K-Médias</i>
<i>LM:</i>	Algoritmo hierárquico com Ligação Média
<i>LS:</i>	Algoritmo hierárquico com Ligação Simples
<i>MCLA:</i>	<i>Meta-CLustering Algorithm</i>
<i>MK:</i>	Todas as soluções do MOCK
<i>MKR:</i>	Soluções recomendadas do MOCK
<i>MOCK:</i>	<i>Multi-Objective Clustering with automatic K-determination</i>
<i>MOCLE:</i>	<i>Multi-Objective Clustering Ensemble</i>
<i>MSH:</i>	Configuração do MOCLE semi-supervisionada e com recombinação HBGF
<i>MSM:</i>	Configuração do MOCLE semi-supervisionada e com recombinação MCLA
<i>MST:</i>	<i>Minimum Spanning Tree</i>
<i>MUH:</i>	Configuração do MOCLE não supervisionada e com recombinação HBGF
<i>MUM:</i>	Configuração do MOCLE não supervisionada e com recombinação MCLA
<i>NSGA:</i>	<i>Non-dominated Sorting Genetic Algorithm</i>
<i>NSGA – II:</i>	<i>Non-dominated Sorting Genetic Algorithm II</i>
<i>OPTICS:</i>	Algoritmo <i>Ordering Points To Identify the Clustering Structure</i>
<i>PESA:</i>	<i>Pareto Envelope Selection Algorithm</i>

<i>PESA – II:</i>	<i>Pareto Envelope Selection Algorithm II</i>
<i>SNN:</i>	<i>Algoritmo Shared Nearest Neighbor</i>
<i>SOM:</i>	<i>Algoritmo Self-Organizing Maps</i>
<i>SPEA:</i>	<i>Strength Pareto Evolutionary Algorithm</i>
<i>SPEA2:</i>	<i>Strength Pareto Evolutionary Algorithm 2</i>
<i>T – ALL:</i>	ALL de linhagem T

Notação

μ_i :	Centróide do <i>cluster</i> c_i
Π :	Conjunto de partições
Π_E :	Conjunto de estruturas conhecidas
Π_I :	Conjunto de partições iniciais usadas por um algoritmo
Π_S :	Conjunto de soluções
π^{Ei} :	i -ésima estrutura conhecida
π^F :	Partição consenso
π^i :	i -ésima partição
π^{Ii} :	i -ésima partição do conjunto de partições iniciais
π^{Si} :	i -ésima partição do conjunto de soluções
$ A $:	Número de elementos do conjunto A
c_j^i :	j -ésimo <i>cluster</i> da i -ésima partição
d :	Número de dimensões (atributos) dos objetos conjunto de dados X
$d(c_i, c_j)$:	Distância entre os <i>clusters</i> c_i e c_j
$d(\mathbf{x}_i, \mathbf{x}_j)$:	Distância entre os objetos $\mathbf{x}_i$ e $\mathbf{x}_j$
k :	Número de <i>clusters</i> de uma partição qualquer
K^{Ei} :	Número de <i>clusters</i> da i -ésima estrutura conhecida
K^i :	Número de <i>clusters</i> da i -ésima partição
K^{Ii} :	Número de <i>clusters</i> da i -ésima partição do conjunto de partições iniciais
K^{max} :	Número máximo de <i>clusters</i>
K^{min} :	Número mínimo de <i>clusters</i>
K^{Si} :	Número de <i>clusters</i> da i -ésima partição do conjunto de soluções
n :	Número de objetos do conjunto de dados X
n^A :	Número de algoritmos
n^D :	Número de conjuntos de dados
n^E :	Número de estruturas conhecidas
n^I :	Número de partições no conjunto de partições iniciais
n^S :	Número de partições no conjunto de soluções
nn_{ij} :	j -ésimo vizinho mais próximo do objeto $\mathbf{x}_i$

v :	Número de vizinhos mais próximos
$\mathbf{x}_i$:	i -ésimo objeto do conjunto de dado X
X :	Conjunto de dados

Lista de Figuras

3.1	Etapas do processo de agrupamento.	23
3.2	Dendrograma	30
4.1	Exemplo do HBGF - grafo bipartido	53
5.1	Conjuntos de dados com estrutura homogênea	58
5.2	Conjunto de dados com estrutura heterogênea	59
5.3	Resultados dos algoritmos no conjunto de dados heterogêneo	60
5.4	Conjunto de dados com várias estruturas	61
5.5	MOCLE	67
5.6	Exemplo da representação de um indivíduo	68
5.7	Grafo gerado na aplicação do operador de recombinação com a técnica HBGF	70
6.1	Visualização do exemplo	76
6.2	Exemplo das informações contidas na visualização	77
7.1	Conjuntos de dados artificiais	84
8.1	Comparação do MOCLE com os algoritmos individuais - ds2c2sc13	119
8.2	Comparação do MOCLE com os algoritmos individuais - ds3c3sc6	119
8.3	Comparação do MOCLE com os algoritmos individuais - ds4c2sc8	120
8.4	Comparação do MOCLE com os algoritmos individuais - spiralsquare	120
8.5	Comparação do MOCLE com os algoritmos individuais - glass	121
8.6	Comparação do MOCLE com os algoritmos individuais - iris	121
8.7	Comparação do MOCLE com os algoritmos individuais - golub	122
8.8	Comparação do MOCLE com os algoritmos individuais - proteinas	122
8.9	Comparação do MOCLE com os algoritmos individuais - leukemia	123
8.10	Comparação do MOCLE com os algoritmos individuais - lung	123
8.11	Comparação das combinações - ds2c2sc13	127
8.12	Comparação das combinações - ds3c3sc6	128
8.13	Comparação das combinações - ds4c2sc8	129
8.14	Comparação das combinações - spiralsquare	130

8.15	Comparação das combinações - glass	131
8.16	Comparação das combinações - iris	132
8.17	Comparação das combinações - golub	133
8.18	Comparação das combinações - proteinas	134
8.19	Comparação das combinações - leukemia	135
8.20	Comparação das combinações - lung	136

Lista de Tabelas

3.1	Características dos algoritmos de agrupamento	28
4.1	Comparação das formas de combinação de agrupamentos.	45
4.2	Exemplo do MCLA - partições	50
4.3	Exemplo do MCLA - hipergrafo	50
4.4	Exemplo do MCLA - pesos	51
4.5	Exemplo do MCLA - meta-hiperarestas e vetores de associação	51
4.6	Exemplo do HBGF - partições	53
5.1	População	69
6.1	Partições do exemplo do método de visualização	74
6.2	Passos para “colorir” as partições do exemplo	75
7.1	Características dos conjuntos de dados	82
7.2	Tamanho dos <i>clusters</i> - ds2c2sc13	83
7.3	Tamanho dos <i>clusters</i> - ds3c3sc6	83
7.4	Tamanho dos <i>clusters</i> - ds4c2sc8	85
7.5	Tamanho dos <i>clusters</i> - spiralsquare	85
7.6	Tamanho dos <i>clusters</i> - glass	86
7.7	Tamanho dos <i>clusters</i> - golub	87
7.8	Tamanho dos <i>clusters</i> - proteinas	87
7.9	Tamanho dos <i>clusters</i> - leukemia	88
7.10	Tamanho dos <i>clusters</i> - lung	89
7.11	Valores dos parâmetros	90
8.1	<i>CR</i> e <i>k</i> - ds2c2sc13	99
8.2	<i>CR</i> e <i>k</i> - ds3c3sc6	99
8.3	<i>CR</i> e <i>k</i> - ds4c2sc8	100
8.4	<i>CR</i> e <i>k</i> - spiralsquare	100
8.5	<i>CR</i> e <i>k</i> - glass	101
8.6	<i>CR</i> e <i>k</i> - iris	101

8.7	<i>CR</i> e <i>k</i> - golub	102
8.8	<i>CR</i> e <i>k</i> - proteinas	103
8.9	<i>CR</i> e <i>k</i> - leukemia	103
8.10	<i>CR</i> e <i>k</i> - lung	104
8.11	Comparações entre as técnicas	105
8.12	Número de partições	113
8.13	Origem das soluções - Conjuntos artificiais	116
8.14	Origem das soluções - Conjuntos de <i>benchmarks</i> e reais	117
8.15	Visualização para o MSH considerando a estrutura E1	139
8.16	Visualização para o LM considerando a estrutura E2	142

Introdução

1.1 Contextualização

Técnicas de agrupamento são instrumentos valiosos na análise exploratória dos dados e encontram aplicações em várias áreas, tais como: biologia, medicina, engenharia, *marketing*, visão computacional e sensoriamento remoto. Uma área de aplicação recente que tem se beneficiado significativamente da análise de agrupamento é a bioinformática (Baldi and Brunak 1998; Wang et al. 2003; Narayanan 2005). Nessa área, muitos trabalhos têm sido desenvolvidos aplicando algoritmos de agrupamento para análise de dados de expressão gênica (Wang et al. 2003; Lorkowski and Cullen 2003; Zhao and Karypis 2005; Azuaje and Dopazo 2005; Narayanan 2005).

Em essência, as técnicas de agrupamento fornecem um meio de explorar e verificar estruturas presentes nos dados, organizando-os em *clusters* de objetos similares (Jain and Dubes 1988; Fred 2001). O agrupamento pode ser visto como pertencente ao paradigma de aprendizado não supervisionado, em que o aprendizado é dirigido aos dados, não requerendo conhecimento prévio sobre as suas classes ou categorias (Mitchell 1997). Tal característica é vantajosa para a bioinformática, pois, em geral, há pouca disponibilidade de conhecimento prévio sobre os dados (Zeng et al. 2002).

Na bioinformática, as investigações em genômica funcional e a análise de dados de expressão gênica têm utilizado técnicas de agrupamento para encontrar grupos de genes, amostras ou ambos. O agrupamento de genes é baseado na similaridade dos perfis moleculares de células em diferentes condições e pode ser utilizado, por exemplo, para estudar os mecanismos regulatórios dos genes ou dividir o genoma em conjuntos de genes que estão envolvidos nos mesmos processos ou em processos relacionados (Eisen et al. 1998; Spellman et al. 1998; Tamayo et al. 1999; Nikkilä et al. 2002; Hautaniemi et al. 2003).

O agrupamento das amostras, por sua vez, é feito de acordo com as similaridades nos níveis de expressão dos genes para, por exemplo, identificar condições que produzem expressões semelhantes dos genes investigados, ou determinar a “impressão digital” genômica de diferentes tipos de câncer (Porter et al. 2001; Ng et al. 2001; Ryu et al. 2002; Ma et al. 2003). Além disso, diversos trabalhos que agrupam amostras analisam a descoberta de subtipos de câncer, como os de Golub et al. (1999), Bittner et al. (2000), Alizadeh et al. (2000), Azuaje (2000), Sorlie et al. (2001) e Yeoh et al. (2002). O agrupamento simultâneo de genes e amostras pode ser utilizado para identificar quais genes são mais importantes para agrupar amostras, por exemplo (Alon et al. 1999; Getz et al. 2003).

Esses são apenas alguns poucos trabalhos dentre os muitos que empregam agrupamento para análise de expressão gênica e ilustram uma das frentes de pesquisa atuais em que a análise de agrupamento tem se mostrado de grande utilidade, embora esse tipo de análise venha contribuindo também nas pesquisas em muitas outras áreas, tanto recentes quanto mais tradicionais. A área de bioinformática tem, inclusive, motivado a proposição de uma grande quantidade de novas técnicas de agrupamento (Ben-Dor et al. 1999; Sharan and Shamir 2000; Hastie et al. 2000; Cheng and Church 2000; Herrero et al. 2001; Lazzeroni and Owen 2002; Getz et al. 2003).

É nesse contexto, de análise de dados de expressão gênica, que a abordagem proposta nesta Tese foi motivada. Em especial, a utilização das técnicas de agrupamento para a descoberta de subclasses nessa área de aplicação oferece uma motivação extra, como será discutido mais adiante. Inclusive, os quatro conjuntos de dados reais utilizados nos experimentos pertencem à área de bioinformática, sendo três deles de expressão gênica. Dentre esses três, dois ilustram bem a questão da descoberta de subclasses.

A seguir, são apresentadas as motivações que levaram à proposição deste trabalho, incluindo as dificuldades existentes na análise de agrupamento e a aplicação de técnicas de agrupamento para a descoberta de subclasses.

1.2 Motivação

A análise de agrupamento é de grande utilidade na análise exploratória de dados sobre os quais existe pouco ou nenhum conhecimento prévio disponível (Jain and Dubes 1988; Handl et al. 2005; Xu and Wunsch 2005). O objetivo do agrupamento é encontrar uma estrutura de *clusters* nos dados, em que cada *cluster* contém objetos que compartilham algumas características ou propriedades consideradas relevantes para o domínio dos dados estudados (Jain and Dubes 1988; Monti et al. 2003; Handl et al. 2005; Xu and Wunsch 2005). Entretanto, não existe uma definição precisa do que é um *cluster* e há uma grande variedade de algoritmos de agrupamento descrita na literatura, cada algoritmo com suas próprias características e peculiaridades (Barbara 2000; Estivill-Castro 2002;

Xu and Wunsch 2005).

Cada algoritmo é baseado em uma definição de *cluster* e faz uso de alguma heurística para achar o melhor agrupamento para um determinado conjunto de dados. Assim, cada algoritmo de agrupamento pode apresentar um comportamento superior aos demais para uma conformação específica dos dados no espaço de atributos. Por exemplo, um algoritmo pode ser apropriado para encontrar apenas *clusters* hiper-esféricos e outro pode encontrar *clusters* de formas arbitrárias, mas que possuam a mesma densidade. Nesse ponto surge a primeira dificuldade em análise de agrupamento: mesmo que os dados estejam estruturados idealmente segundo uma das possíveis definições de *cluster*, como selecionar o algoritmo mais apropriado, uma vez que as características dos dados não são conhecidas previamente?

Além disso, cada algoritmo de agrupamento é capaz de encontrar estruturas com diferentes níveis de refinamento (estruturas com diferentes números de *cluster* ou com *clusters* de densidades diferentes), dependendo dos valores de seus parâmetros (Jain and Dubes 1988). Por exemplo, o algoritmo *k*-médias encontra uma estrutura diferente para cada número de *clusters*, *k*, dado como parâmetro ao algoritmo. Porém, na análise exploratória de um conjunto de dados, em geral, o número de *clusters* presentes nos dados não é conhecido previamente. Esse aspecto oferece a segunda dificuldade da análise de agrupamento: dado que se conheça o algoritmo de agrupamento mais apropriado para um determinado conjunto de dados, como estabelecer quais valores de seus parâmetros fornecem as estruturas mais representativas do conjunto de dados? Em outras palavras, em que níveis de refinamento podem ser encontradas as estruturas subjacentes nos dados?

Existem inúmeras técnicas de validação de agrupamento descritas na literatura que podem auxiliar tanto na seleção do algoritmo mais apropriado, uma vez que vários algoritmos tenham sido aplicados aos dados (Halkidi et al. 2001; Handl et al. 2005; Faceli et al. 2005d), quanto na seleção da melhor partição, dada a aplicação de um algoritmo com vários valores para seus parâmetros. O caso mais típico da aplicação de validação agrupamento é na determinação do número de *clusters* mais apropriado para um conjunto de dados. Porém, a maioria das medidas de validação é tendenciosa, cada uma favorecendo um critério de agrupamento diferente (Handl et al. 2005). Nesse ponto surge outra dificuldade diretamente relacionada com a primeira: qual medida de validação empregar, se não se conhece o critério de agrupamento mais apropriado aos dados. Segundo Handl et al. (2005), para a análise exploratória deve-se considerar a aplicação de várias medidas de validação em resultados obtidos com algoritmos conceitualmente diferentes, considerando os diversos níveis de refinamento possíveis. Quanto mais medidas concordarem na qualidade de uma estrutura, maior é a confiança de que ela seja a verdadeira. A dificuldade desse tipo de análise completa é a necessidade de um conhecimento mais

profundo em análise de agrupamento, o que raramente os especialistas do domínio dos dados possuem.

Um outro ponto que deve ser destacado é que, em geral, os algoritmos de agrupamento assumem um critério homogêneo em todo o espaço de atributos (Law et al. 2004). Isso significa que todos os *clusters* encontrados por um algoritmo têm características semelhantes. Neste ponto, surge outra questão importante: como encontrar todos os *clusters* se cada região do espaço de atributos contém *clusters* de diferentes tipos (estão de acordo com um critério diferente)?

Em resumo, não existe uma técnica de agrupamento universal, capaz de revelar toda a variedade de estruturas, quer homogêneas, quer heterogêneas, que podem estar presentes em um conjunto de dados (Estivill-Castro 2002; Kleinberg 2002). Na verdade, é praticamente impossível estabelecer previamente qual é o critério de agrupamento mais apropriado para revelar uma estrutura subjacente dos dados. Além disso, um mesmo conjunto de dados pode ter mais de uma estrutura relevante, cada uma de acordo com uma definição de *cluster* (critério de agrupamento) diferente e/ou em diferentes níveis de refinamento e, a aplicação usual de análise de agrupamento para explorar o conjunto de dados (aplicar vários algoritmos e selecionar o melhor resultado usando validação) é direcionada à obtenção de uma única estrutura que melhor se ajuste aos dados. Essa busca por uma única estrutura limita a quantidade de conhecimento que pode ser extraído dos dados. A obtenção de uma série de estruturas alternativas pode oferecer diferentes interpretações dos dados, de grande utilidade para os especialistas do domínio (Handl and Knowles 2004).

Na tentativa de superar as limitações discutidas, diversas abordagens de *ensembles* de agrupamentos (Strehl and Ghosh 2002; Topchy et al. 2004; Fern and Brodley 2004; Law et al. 2004; Kuncheva et al. 2002) e a abordagem de agrupamento multi-objetivo (Handl and Knowles 2004; Handl and Knowles 2005a; Handl and Knowles 2005b; Handl and Knowles 2006a) foram propostas na literatura. Essas abordagens têm se mostrado robustas frente a diferentes conformações dos dados. Porém, da mesma maneira que a abordagem usual de análise de agrupamento, os *ensembles* são direcionado à obtenção de uma única estrutura que melhor se ajuste aos dados e necessitam de ajustes de parâmetros. Além disso, a presença de um grande número de partições iniciais de baixa qualidade influencia negativamente o resultado da combinação, uma vez que todas elas são consideradas simultaneamente para gerar a partição consenso. Já a abordagem multi-objetivo fornece como resultado um conjunto de estruturas alternativas e não requer muitos ajustes de parâmetros (Handl and Knowles 2004). Porém, quanto maior o número de alternativas, mais difícil é sua análise por parte dos especialistas do domínio (Handl and Knowles 2004).

Na proposta de Handl and Knowles (2004) um grande número de soluções alternativas é encontrado. Desse conjunto de soluções, algumas são apontadas como as melhores. Apesar disso, nos experimentos realizados para esta Tese, foi observado que essas estruturas indicadas como melhores, nem sempre correspondem às estruturas mais próximas às verdadeiras dentre todas as presentes no conjunto de soluções. Além disso, essa técnica não se mostrou muito estável, ou seja, para várias execuções do algoritmo com os mesmos dados e parâmetros, as soluções obtidas que mais se aproximam das estruturas conhecidas são diferentes.

Considerando um conjunto de soluções, mesmo que pequeno, como resultado de um agrupamento, surge uma outra dificuldade: como analisar e comparar simultaneamente todos os agrupamentos?

Todas essas dificuldades encontradas na análise exploratória de um conjunto de dados utilizando agrupamento compõem a principal motivação para a abordagem proposta nesta Tese.

Uma motivação adicional vem da aplicação da análise de agrupamento na descoberta de sub-classes (Golub et al. 1999; Alizadeh et al. 2000; Yeoh et al. 2002), conforme já mencionado. A abordagem tradicional para análise de agrupamento tem sido comumente empregada com esse objetivo. Nesse caso, um algoritmo de agrupamento tradicional, como o k -médiãs, é aplicado aos dados e, em seguida, a estrutura obtida é analisada pelo especialista que identifica que alguns dos *clusters* retornados pelo algoritmo não eram conhecidos previamente. Para essa análise, o próprio especialista faz uso de conhecimento prévio de uma classificação dos objetos. A questão que surge disso é: como automatizar a utilização desse conhecimento prévio de uma estrutura subjacente aos dados para auxiliar na descoberta de outras estruturas?

Podem ser encontradas na literatura algumas abordagens de agrupamento semi-supervisionado que consideram conhecimento prévio dos dados (Handl and Knowles 2006b; Demiriz et al. 1999). Porém, em geral, essas abordagens consideram que uma pequena parte dos objetos está rotulada e grande parte deles não está. O objetivo principal do agrupamento semi-supervisionado é a melhora de desempenho em relação às técnicas puramente supervisionadas e não supervisionadas na obtenção de uma única estrutura, parcialmente conhecida, e não em revelar novas estruturas. Assim, é importante destacar que o termo “semi-supervisionado” está sendo utilizado nesta Tese para indicar a utilização de conhecimento prévio, mas não tem relação direta com a definição de agrupamento semi-supervisionado geralmente adotada na literatura.

Com o objetivo de superar todas essas dificuldades encontradas na análise de agrupamento e também estender a análise de agrupamento para considerar conhecimento prévio para auxiliar na descoberta de novas estruturas, está sendo proposto nesta Tese o *frame-*

work descrito na Seção 1.3. Em especial, a abordagem proposta é de grande valor nas áreas de genômica funcional e análise de dados de expressão gênica, em que os experimentos para coletar os dados são caros e demorados, e o conhecimento adquirido com a análise dos dados tem potencialmente grandes compensações em temas como diagnóstico, prognóstico e tratamento de doenças. Por exemplo, nessas áreas é altamente desejável ter um conjunto de estruturas alternativas, uma vez que os dados têm um grande potencial de conter várias interpretações úteis (Handl and Knowles 2004).

Exemplos de situações em que a disponibilidade de estruturas alternativas pode ser útil são a análise das funções dos genes, uma vez que os genes podem pertencer a várias categorias funcionais, e a descoberta de subtipos de doenças. A robustez da abordagem frente a diferentes conformações dos dados também é essencial, uma vez que nessas áreas há pouco conhecimento prévio para direcionar as escolhas dos algoritmos e configurações de parâmetros, e as estruturas presentes nos dados tendem a ser complexas.

1.3 Abordagem Proposta

A abordagem proposta nesta Tese consiste de um *framework* para a análise exploratória de dados via agrupamento que facilite o trabalho dos especialistas do domínio dos dados, resolvendo de maneira integrada muitas das dificuldades comumente encontradas na análise de agrupamento.

O *framework* proposto se aplica a dois contextos diferentes, com pequenas modificações. O primeiro contexto se refere à análise de agrupamento totalmente não supervisionada. O segundo, envolve a análise de agrupamento considerando o conhecimento prévio de uma estrutura presente nos dados, ou seja, uma análise semi-supervisionada.

O ponto central do *framework* é um algoritmo de *ensemble* multi-objetivo, MOCLE (do inglês *Multi-Objective Clustering Ensemble*), que integra a saída (*output*) de diversos algoritmos de agrupamento, técnicas de validação e *ensemble* de agrupamentos em uma abordagem multi-objetivo, para encontrar um conjunto de estruturas que podem conter informações relevantes para os especialistas no domínio dos dados. Além disso, no contexto semi-supervisionado, o conhecimento prévio de uma estrutura dos dados é utilizado para auxiliar na obtenção de outras estruturas. Ainda nesse contexto, é considerado um esquema para visualização das estruturas resultantes que facilita sua análise simultânea. Nesta Tese, o termo estrutura se refere a uma partição do conjunto de dados.

O algoritmo MOCLE, como qualquer *ensemble*, pode ser dividido em dois blocos: (1) geração de um conjunto diverso de partições iniciais a serem combinadas e (2) determinação do consenso. O MOCLE difere dos *ensembles* tradicionais em dois aspectos, relacionados à obtenção do consenso.

Em primeiro lugar, o MOCLE busca por um conjunto de partições consenso, em lugar

de uma única partição. Na verdade, o conjunto de soluções que o MOCLE retorna pode conter tanto partições que resultam da combinação de outras partições, quanto partições de alta qualidade que já apareciam dentre as partições iniciais. A segunda diferença do MOCLE em relação aos demais *ensembles* é que ele combina pares de partições, iterativamente, em um processo de otimização que garante diferentes compromissos de qualidade das soluções. Com isso, o MOCLE consegue evitar a influência negativa das partições iniciais de baixa qualidade que afeta as abordagens tradicionais de *ensemble*.

Mais precisamente, o MOCLE deve ser iniciado com a geração de um conjunto de partições iniciais por meio da aplicação de vários algoritmos de agrupamento conceitualmente diferentes aos dados, também considerando várias configurações de parâmetros. Isso garante a diversidade das partições iniciais do *ensemble*. Em seguida, essas partições iniciais são utilizadas como população inicial para um algoritmo genético multi-objetivo baseado em Pareto. Esse algoritmo vai selecionar e combinar as partições iniciais por meio de duas características particulares: (1) um operador de recombinação especial, que encontra o consenso entre duas partições pais, e (2) a otimização de funções objetivo que representam diferentes medidas de qualidade de uma partição.

O operador de recombinação proposto fornece a característica de *ensemble* ao MOCLE, o que o diferencia da abordagem de agrupamento multi-objetivo pura.

Com essas características, o MOCLE faz uma seleção automática das partições mais significativas, dentre as iniciais e as combinações, sem que sejam necessários muitos ajustes de parâmetros e nem conhecimento profundo em análise de agrupamento. Com isso, ele supera as dificuldades da análise de agrupamento tradicional. Mais ainda, a integração das abordagens de *ensemble* e agrupamento multi-objetivo permite superar as dificuldades individuais de ambas as abordagens. Além disso, por meio das funções objetivo, o MOCLE permite a integração do conhecimento prévio de uma estrutura simples dos dados na busca por outras estruturas mais complexas.

Em resumo, o MOCLE constitui uma abordagem robusta para lidar com diferentes tipos de estrutura (partição) que podem estar presentes nos dados, fornecendo como resultado um conjunto conciso e estável de estruturas alternativas de elevada qualidade, sem a necessidade de conhecimento prévio dos dados e nem conhecimento profundo em análise de agrupamento.

Além do MOCLE, o *framework* proposto engloba ainda um método para a visualização simultânea de um conjunto de partições que auxilia na análise do conjunto de soluções do MOCLE. Esse método de visualização é baseado em um esquema para colorir uma partição que associa cores a cada um de seus *clusters*. O método se aplica ao contexto semi-supervisionado, necessitando do conhecimento prévio de uma estrutura dos dados. Além de parte integrante do *framework*, o método de visualização pode ser visto como uma

contribuição independente dele, facilitando a análise de qualquer conjunto de partições e complementando as informações que podem ser obtidas com índices de validação externa nas comparações entre várias técnicas de agrupamento.

O emprego do *framework* no contexto semi-supervisionado, ou seja, considerando a função objetivo apropriada e/ou o método de visualização, dá a ele o caráter automático para a utilização de conhecimento prévio para a obtenção de novas estruturas, que facilita o trabalho dos especialistas ao investigar a existência de subclasses em dados com classes conhecidas, como mencionado nas Seções 1.1 e 1.2.

1.4 Organização do Trabalho

Esta Tese está organizada da seguinte maneira. Os Capítulos 2, 3 e 4 contêm uma revisão dos temas relevantes para a compreensão, proposição e implementação da abordagem proposta, que será detalhada e analisada nos demais capítulos. Mais especificamente:

- No Capítulo 2 serão apresentados os conceitos relacionados à otimização multi-objetivo, o uso de algoritmos genéticos para esse tipo de otimização e os algoritmos genéticos multi-objetivo relacionados a este trabalho.
- No Capítulo 3 serão detalhados os conceitos básicos de agrupamento, apresentados os algoritmos e técnicas de validação que serão utilizadas e introduzido o tema de agrupamento semi-supervisionado.
- No Capítulo 4 serão apresentadas as abordagens recentes que procuram superar algumas limitações da análise de agrupamento tradicional: *ensemble* de agrupamentos e agrupamento multi-objetivo.
- No Capítulo 5, que apresenta uma das contribuições originais deste trabalho, serão detalhados os problemas que motivaram a abordagem e apresentadas as metas que se deseja atingir com o *framework* proposto. Será ainda apresentada a parte central desse *framework*, que é o algoritmo MOCLE. A outra parte original do *framework*, que é o método de visualização, será detalhada no Capítulo 6.
- O Capítulo 7 contém uma descrição dos conjuntos de dados utilizados, dos experimentos realizados e dos métodos empregados na avaliação dos resultados.
- Os resultados dos experimentos serão apresentados no Capítulo 8, mostrando que o *framework* atinge as metas estabelecidas.
- No Capítulo 9 serão apresentadas as conclusões do trabalho, com os principais resultados obtidos, as contribuições e limitações do trabalho, além da indicação de trabalhos futuros.

Algoritmos Genéticos Multi-objetivo Baseados em Pareto

2.1 *Considerações Iniciais*

Com a proposta desta Tese motivada e resumida no Capítulo 1, este capítulo revisa o primeiro tópico relevante para seu detalhamento e implementação, que são os algoritmos genéticos multi-objetivo.

Inicialmente, na Seção 2.2, serão introduzidos os conceitos gerais relacionados a otimização multi-objetivo. Em seguida, na Seção 2.3, serão apresentados os principais aspectos dos algoritmos evolutivos, incluindo os algoritmos genéticos, e discutidas as características necessárias para a utilização dos algoritmos evolutivos em problemas multi-objetivo. Nessa seção será ainda comentada a aplicação de algoritmos genéticos a problemas de agrupamento. Finalmente, na Seção 2.4, serão apresentados os algoritmos genéticos multi-objetivo de interesse para este trabalho.

2.2 *Otimização Multi-objetivo*

O problema de otimização multi-objetivo pode ser definido como: dado um vetor de variáveis de decisão $\mathbf{y} = \{y_1, y_2, \dots, y_s\}$, de dimensão s , no espaço de soluções Y , encontrar um vetor solução $\mathbf{y}^*$ que minimize um conjunto de m funções objetivo, $\mathbf{z}(\mathbf{y}^*) = \{z_1(\mathbf{y}^*), z_2(\mathbf{y}^*), \dots, z_m(\mathbf{y}^*)\}$ (Zitzler et al. 2004; Konak et al. 2006). Assim, neste trabalho, está sendo assumida a minimização dos objetivos, sem perda de generalidade, uma vez que qualquer objetivo que deva ser maximizado pode ser convertido em um problema de minimização pela sua multiplicação por -1 (Handl and Knowles 2004). $\mathbf{z}(\mathbf{y})$ mapeia Y em $\mathbb{R}^m$, com $m \geq 2$. Em geral, nos problemas reais de otimização, há vários objetivos conflitantes

que devem ser otimizados, isto é, objetivos em que melhoras em um freqüentemente causam pioras em outro (Zitzler 1999). Assim, geralmente não há uma única solução que minimize todas as funções objetivo simultaneamente. Em lugar de uma única solução ótima, como na otimização de um único objetivo, a solução para o problema de otimização multi-objetivo é dada por um conjunto de soluções com diferentes compromissos para os objetivos. Essas soluções são ótimas no sentido de que não há outras soluções no espaço de busca que sejam superiores a elas ao se considerar todos os objetivos, ou seja, não são dominadas por outras soluções (Zitzler 1999).

Uma solução $\mathbf{y}_1$ domina outra solução $\mathbf{y}_2$ ($\mathbf{y}_1 \succ \mathbf{y}_2$), se e somente se $z_i(\mathbf{y}_1) \leq z_i(\mathbf{y}_2)$ para $i = 1, \dots, m$ e $z_j(\mathbf{y}_1) < z_j(\mathbf{y}_2)$ para pelo menos uma função objetivo j . Uma solução é um ótimo de Pareto se ela não é dominada por nenhuma outra solução no espaço de soluções (Konak et al. 2006). Um ótimo de Pareto não pode ser melhorado em relação a algum objetivo sem piorar pelo menos um outro. O conjunto de todas as soluções não dominadas em Y (todos os ótimos de Pareto) é chamado de conjunto ótimo de Pareto (*Pareto optimal set*). Os valores das funções objetivo para as soluções do conjunto ótimo de Pareto compõem o fronte de Pareto ótimo (*Pareto optimal front*) (Konak et al. 2006).

O ideal para um algoritmo de otimização multi-objetivo seria identificar todas as soluções do conjunto ótimo de Pareto. Entretanto, para muitos problemas reais complexos, não é possível encontrar todas as soluções ótimas (Zitzler 1999; Konak et al. 2006). Com isso, a abordagem prática para a otimização multi-objetivo busca por uma aproximação do conjunto ótimo de Pareto, que o represente da melhor forma possível.

As principais dificuldades na resolução de um problema de otimização multi-objetivo estão no processo de otimização ou busca, em que um espaço de busca grande e complexo torna a busca difícil e impede o uso de métodos de otimização exata, e no processo de decisão, em que a seleção da solução com o compromisso mais adequado dentre as do conjunto ótimo de Pareto depende do especialista humano (Zitzler 1999). Considerando esses dois processos, a otimização multi-objetivo pode considerar a tomada de decisão antes, durante ou depois da busca (Zitzler 1999).

Na decisão antes da busca, os objetivos do problema são agregados em um único objetivo que inclui implicitamente as preferências do especialista. Nesse caso, as estratégias de otimização de um único objetivo podem ser utilizadas diretamente. Entretanto, essa abordagem requer um conhecimento profundo do domínio, o que raramente está disponível.

A tomada de decisão depois da busca evita essa desvantagem, mas faz com que a complexidade do espaço de busca seja maior, pois a busca é feita sem as considerações do especialista que ajudam a limitar esse espaço. Outro problema desse caso é a visualização e apresentação das soluções para o especialista no caso da existência de muitos objetivos.

A tomada de decisão integrada ao processo de busca une as vantagens das outras

abordagens. Nesse caso, a cada passo da otimização, várias soluções alternativas são apresentadas ao especialista. Com base nessas soluções, ele ajusta suas preferências para guiar o processo de busca.

As abordagens mais tradicionais para a otimização multi-objetivo agregam os objetivos em uma única função objetivo parametrizada, em analogia à tomada de decisão antes da busca (Zitzler 1999). Entretanto, os parâmetros dessa função são variados sistematicamente durante várias execuções, em vez de serem determinados pelo especialista. Alguns exemplos dessas abordagens são: método de ponderação (*weighting method*), método de restrição (*constraint method*) e abordagem minmax (Zitzler 1999; Coello 1999). A principal vantagem desses métodos é que o problema pode ser resolvido por algoritmos para otimização de um único objetivo já bastante estudados, incluindo algoritmos genéticos (AGs). Os principais problemas de várias dessas abordagens são:

- Sensibilidade à forma do fronte de Pareto.
- Exigência de conhecimento do problema para estabelecer a função a ser otimizada.
- Necessidade de várias execuções do algoritmo de otimização para a obtenção de uma aproximação do conjunto ótimo de Pareto.

Uma alternativa a esses métodos clássicos são os algoritmos evolutivos, principalmente as abordagens baseadas em Pareto (Coello 1999). Os algoritmos evolutivos são bastante apropriados para resolver problemas de otimização multi-objetivo pois lidam simultaneamente com um conjunto de possíveis soluções (população), que permitem encontrar ao menos uma aproximação do conjunto ótimo de Pareto em uma única execução do algoritmo (Coello 1999). Além disso, os algoritmos evolutivos são menos sensíveis à forma ou continuidade do fronte de Pareto. Mesmo assim, em muitas aplicações complexas, não é possível gerar o conjunto ótimo de Pareto completo. Com isso, é importante que, para a otimização de um problema multi-objetivo, sejam perseguidas as seguintes metas conflitantes (Zitzler 1999; Konak et al. 2006):

- A aproximação obtida deve ser tão próxima quanto possível do fronte de Pareto ótimo. Idealmente, a aproximação do conjunto de ótimo Pareto deve ser um subconjunto do conjunto de ótimo Pareto.
- As soluções na aproximação devem estar uniformemente distribuídas sobre o fronte de Pareto ótimo.
- A aproximação do fronte de Pareto deve capturar todo o espectro do fronte de Pareto ótimo, incluindo soluções nos extremos do espaço das funções objetivo. Em outras palavras, a aproximação deve ter o maior espalhamento possível.

Existe uma grande variedade de algoritmos evolutivos multi-objetivo que exploram diferentes características, como cálculo da aptidão, diversidade da população e elitismo, para atingir essas metas (Zitzler 1999; Konak et al. 2006). A seguir, os conceitos básicos relacionados aos algoritmos evolutivos serão descritos, juntamente com os aspectos necessários para a sua aplicação aos problemas multi-objetivos para que atinjam essas metas.

2.3 Algoritmos Evolutivos

Os Algoritmos Evolutivos (AEs) simulam o processo de evolução natural. De maneira bastante simplificada, evolução é o resultado da interação entre a criação de novas informações genéticas e sua avaliação e seleção (Bäck et al. 1997). Nesse processo, um indivíduo de uma população é afetado por outros indivíduos e pelo ambiente. Quanto melhor um indivíduo se sai nessas condições, maior suas chances de sobreviver por um longo período e de gerar descendentes, que herdam informações genéticas dos pais. No curso da evolução, isso faz com que as informações genéticas de indivíduos com aptidão acima da média sejam introduzidas na população. A natureza não determinística da reprodução leva a uma produção permanente de novas informações genéticas e, portanto, a criação de novos indivíduos.

Com base nesse modelo de evolução, pode ser definida uma estrutura geral para os AEs. A idéia geral é manter um conjunto de soluções candidatas que são manipuladas por operadores genéticos e passam por um processo de seleção ao longo de uma série de iterações (Bäck et al. 1997). O conjunto de soluções candidatas é chamado população e cada uma das soluções corresponde a um indivíduo. Cada iteração é chamada de geração. Assim, P_t é a população de n^P indivíduos na geração t . A seleção determina quais indivíduos vão se reproduzir, gerando descendentes para a próxima geração. Para isso, é empregada uma função que mede a qualidade de cada indivíduo, denominada aptidão, que é baseada na função objetivo, específica para cada problema. Os indivíduos com maior valor de aptidão são selecionados para reprodução. A estrutura geral de um AE é dada por (Bäck et al. 1997; Zitzler 1999; Rezende 2003):

1. Inicializa o número da geração: $t = 0$.
2. Inicializa P_t com n^P indivíduos.
3. Calcula a aptidão de cada indivíduo de P_t .
4. $t = t + 1$.
5. Seleciona P_t a partir de P_{t-1} .

6. Aplica os operadores genéticos a P_t (os mais comuns são os operadores de recombinação e mutação).
7. Se o critério de parada não foi satisfeito, volta ao passo 3.

Existem pelo menos três categorias principais de AEs: algoritmos genéticos (AGs), programação evolutiva e estratégias de evolução (Bäck et al. 1997). Dessas categorias, foram derivadas inúmeras variações. As principais diferenças entre uma abordagem e outra estão na representação dos indivíduos, no projeto dos operadores genéticos ou nos mecanismos de seleção e reprodução (Bäck et al. 1997).

Na maioria das aplicações reais, o espaço de busca é constituído por entidades reais ou indivíduos relacionados ao problema (Bäck et al. 1997). No contexto desta Tese, por exemplo, esses indivíduos são as partições do conjunto de dados. As características ou parâmetros que definem esses indivíduos e que estão sujeitas a otimização compõem o espaço de fenótipos (Bäck et al. 1997). No exemplo, o fenótipo englobaria o número de *clusters* da partição e a distribuição dos itens de dados (referidos nas demais seções como objetos) nos *clusters*. Por outro lado, os operadores genéticos freqüentemente lidam com entidades matemáticas que representam as entidades reais. Essas representações compõem o espaço dos genótipos (Bäck et al. 1997). Com isso, faz-se necessária a utilização de uma função de mapeamento ou codificação que mapeie o fenótipo de uma entidade no seu genótipo e outra que decodifique o genótipo em fenótipo (Bäck et al. 1997).

O projeto de um AE específico para a solução de um determinado problema, em geral, pode seguir duas abordagens. A primeira delas corresponde a escolha de um dos algoritmos padrão para ser utilizado e o projeto de uma função de codificação/decodificação apropriada. Essa abordagem oferece como vantagem a utilização de representações e operadores já extensamente utilizados e com resultados teóricos demonstrados (Bäck et al. 1997). A desvantagem está nas funções de codificação/decodificação: “uma função de codificação complexa pode introduzir não linearidades e outras dificuldades matemáticas que podem retardar substancialmente o processo de busca” (Bäck et al. 1997). A segunda abordagem consiste do projeto de uma representação do indivíduo tão próxima quanto possível de seu fenótipo e a construção de operadores genéticos que trabalhem sobre essa representação (Michalewicz 1996; Bäck et al. 1997). Essa abordagem para uma representação “natural” e operadores específicos, além de evitar a necessidade das funções de codificação/decodificação, constitui uma abordagem promissora para solução de muitos problemas (Michalewicz 1996; Bäck et al. 1997).

Em agrupamento, que é o contexto de aplicação dos EAs nesta Tese, há diversos trabalhos que empregam AGs. A maioria dessas abordagens utilizam AGs para a otimização de um único objetivo. Dos trabalhos encontrados, apenas os de Handl e Knowles (Handl and

Knowles 2004; Handl and Knowles 2005a; Handl and Knowles 2005b; Handl and Knowles 2006a) utilizam AGs para a otimização de vários objetivos. Falkenauer (1998) e Cole (1998) fazem uma revisão das formas de representação e operadores comumente empregados em problemas de agrupamento. Nesta Tese não serão discutidas essas abordagens, pois se optou por usar uma representação diretamente relacionada ao conceito de partição e operadores especiais adequados. A principal razão está no centro da abordagem proposta que consiste da união da abordagem de agrupamento multi-objetivo com o *ensemble* de agrupamentos feita por um operador de recombinação especial que trabalha diretamente sobre partições. Além disso, considerou-se a facilidade de se trabalhar diretamente sobre o conceito utilizado, sem a necessidade de funções de codificação/decodificação. Mais ainda, essa representação atende a vários aspectos importantes de uma boa representação. Em primeiro lugar, ela evita o problema de redundância dos indivíduos (vários indivíduos diferentes representando uma mesma solução). Juntamente com o operador de recombinação proposto, evita-se a necessidade de correção dos cromossomos para garantir que o indivíduo seja válido. Além disso, qualquer solução é possível de ser representada, ou seja, a representação é completa.

Como já mencionado, os AEs são bastante apropriados para a otimização de múltiplos objetivos. Os principais pontos a serem considerados no projeto de algoritmos evolutivos multi-objetivo que atinjam as três metas apresentadas na Seção 2.2 são (Zitzler 1999; Konak et al. 2006):

Cálculo da função de aptidão e seleção: diferentemente da otimização de um único objetivo, em que a função objetivo e a função de aptidão freqüentemente são idênticas, nos AGs multi-objetivo, tanto o cálculo da função de aptidão, quanto a seleção, devem considerar as várias funções objetivo a serem otimizadas. As três principais alternativas gerais para calcular a função de aptidão e realizar a seleção são:

- Seleção por meio da alternância dos objetivos: a cada vez que um indivíduo é selecionado, uma função objetivo diferente é empregada. Essa tipo de abordagem é fácil de ser implementada, porém faz com que a população convirja para soluções que podem ser muito boas em relação a um objetivo, mas muito ruins em relação a outros.
- Seleção por meio de agregação: os objetivos são agregados em uma única função objetivo parametrizada, sendo os parâmetros variados sistematicamente durante a execução do algoritmo. Com isso, a seleção é feita de maneira tradicional, como nos AGs com um único objetivo. A implementação dessa abordagem também é bastante fácil, uma vez que os AGs com um objetivo podem ser empregados. A principal desvantagem é que essa abordagem tem difi-

culdades em encontrar soluções uniformemente distribuídas em um fronte de Pareto ótimo não convexo.

- Seleção por meio de *ranks* baseados em Pareto: o cálculo da função de aptidão e a seleção são feitos explicitamente utilizando o conceito de dominância de Pareto. A população é ordenada de acordo com uma regra de dominância e o valor da aptidão de um indivíduo é calculado com base no seu *rank* dentro da população, em vez de considerar diretamente os valores das funções objetivo. As técnicas baseadas em Pareto são as mais populares na otimização multi-objetivo, e é a alternativa que os algoritmos considerados nesta Tese empregam.

Diversidade da população: manter a diversidade na população é importante nos AGs multi-objetivo para gerar soluções uniformemente distribuídas sobre o fronte de Pareto ótimo (Konak et al. 2006). Dois dos métodos empregados para isso são:

- *Fitness sharing*: essa abordagem é empregada para encorajar a busca em regiões não exploradas do fronte de Pareto reduzindo artificialmente a aptidão dos indivíduos em áreas densamente povoadas. Essa abordagem é a mais frequentemente utilizada. Ela tem o objetivo de gerar e manter nichos estáveis e se baseia na idéia de que indivíduos que estão em um nicho, compartilham os recursos disponíveis. Assim, quanto mais indivíduos estão na vizinhança de um certo indivíduo, mais sua aptidão é degradada (Zitzler 1999). Uma desvantagem dessa abordagem é a necessidade de ajuste de mais um parâmetro específico para esse fim.
- *Crowding distance*: essa abordagem tem o objetivo de obter um espalhamento uniforme de soluções ao longo do melhor fronte de Pareto conhecido, sem a necessidade do parâmetro utilizado na abordagem *fitness sharing*. Nessa abordagem, os indivíduos novos substituem indivíduos similares na população.

Elitismo: nos AGs com um único objetivo, o elitismo diz respeito a manutenção do(s) indivíduo(s) com maior aptidão na população. Nos AGs multi-objetivo, todas as soluções não dominadas são consideradas como soluções de elite. A implementação do elitismo nos AGs multi-objetivo não é tão simples e direta como no caso de um único objetivo, principalmente devido ao grande número de soluções de elite (Konak et al. 2006). Existem duas estratégias básicas para implementar o elitismo nos AGs multi-objetivo, que podem inclusive ser combinadas:

- Manter as soluções elitistas na população: Uma implementação direta do elitismo nos AGs multi-objetivo seria copiar todas as soluções não dominadas na população da próxima geração e preencher o restante dos indivíduos

selecionando soluções não dominadas da população corrente. Entretanto, essa abordagem falha quando o número de soluções pais e descendentes não dominadas é maior do que o tamanho da população. Existem várias abordagens para resolver esse problema, tais como, empregar uma população de tamanho dinâmico, ou limitar o número de indivíduos não dominados que serão mantidos na população.

- Armazenar as soluções elitistas em uma população externa: as soluções não dominadas são mantidas em uma lista de soluções de elite. A maioria dos algoritmos armazena todas as soluções não dominadas encontradas ao longo do processo de busca. Nesse caso, a lista é atualizada sempre que uma solução nova é gerada, por meio da remoção de soluções da lista que são dominadas pela nova solução ou pela adição da nova solução, se ela não for dominada por nenhuma solução elitista existente. Como o número de soluções não dominadas pode ser extremamente grande, existem técnicas para controlar o tamanho da lista de soluções de elite. Além disso, é necessária uma maneira de selecionar as soluções de elite para serem reintroduzidas na população. Uma estratégia é unir as duas populações (a normal e a externa), calcular a aptidão de todos os indivíduos e selecionar o número apropriado de indivíduos para a população normal da próxima geração. Outra estratégia é reservar espaço na população da próxima geração para um determinado número de soluções de elite.

2.4 Algoritmos de Interesse

O AG multi-objetivo de maior interesse para este trabalho é o NSGA-II (*Non-dominated Sorting Genetic Algorithm*) (Deb et al. 2002), pois ele será empregado na implementação do MOCLE. Além dele, três outros AGs multi-objetivo serão mencionados neste trabalho: o SPEA (*Strength Pareto Evolutionary Algorithm*) (Zitzler and Thiele 1999), sua versão melhorada SPEA2 (Zitzler et al. 2001) e o PESA-II (*Pareto Envelope Selection Algorithm II*) (Corne et al. 2001). Como essas técnicas não serão utilizadas diretamente neste trabalho, elas serão apenas brevemente descritas.

O NSGA-II foi o algoritmo escolhido por ser um dos mais populares, juntamente com o SPEA2. Segundo Zitzler et al. (2001), em seus experimentos comparando os algoritmos SPEA, SPEA2, NSGA-II e PESA (*Pareto Envelope Selection Algorithm*) (Corne et al. 2000), os algoritmos SPEA2 e NSGA-II tiveram sempre o melhor desempenho, sendo que vantagens do SPEA2 sobre o NSGA-II apenas se destacam quando são empregados muitos objetivos, o que não é o caso deste trabalho. Nesse mesmo trabalho, os autores comentam que os algoritmos PESA e NSGA-II são mais rápidos que o SPEA2 em termos

da complexidade no pior caso. Nos trabalhos investigados para esta Tese, não foram encontradas comparações entre o algoritmo PESA-II e os algoritmos SPEA2 e NSGA-II.

O algoritmo NSGA-II é uma versão bastante melhorada do seu predecessor NSGA (Srinivas and Deb 1994), resolvendo os seguintes problemas da primeira versão: alta complexidade do procedimento para a ordenação pela não dominação, falta de elitismo e a necessidade de especificação do parâmetro para o método *fitness sharing* usado para preservar a diversidade na população. Para isso, o algoritmo NSGA-II emprega um procedimento rápido para ordenar as soluções da população com base na não dominação e emprega o conceito de *crowding distance* para manter a diversidade da população e compor um operador de comparação (*crowded comparison*).

O procedimento para a ordenação rápida pela não dominação tem os seguintes passos. Para cada indivíduo $p \in P$ é feita uma contagem do número de soluções que dominam a solução p , chamada de contagem de dominação, cd_p , e determinado o conjunto de soluções que p domina, S_p . Em seguida, os indivíduos são distribuídos em frentes de não dominação, F_i , em vários níveis i . As soluções não dominadas do primeiro frente, F_1 , têm a contagem de dominação igual a zero. Para cada solução p com $cd_p = 0$, cada $q \in S_p$ é visitado e tem sua contagem de dominação diminuída de um. Cada q para o qual a contagem de dominação foi zerada é colocado em uma lista separada, Q . Os indivíduos de Q compõem o segundo frente não dominado, F_2 . O procedimento é repetido até que todos os frentes não dominados sejam identificados (todos os indivíduos estejam associados a um nível de não dominação). Com isso, cada indivíduo terá um *rank*, que corresponde ao nível de não dominação em que ele se encontra (*rank* de não dominação, $rank(p)$).

A *crowding distance* é uma estimativa da densidade de soluções ao redor de uma solução particular. Antes de calcular essa medida, é necessário normalizar os objetivos. Dado um frente F_i , o procedimento para calcular a *crowding distance* de suas soluções é:

1. Para cada função objetivo z_j :

- Ordena as soluções de F_i em ordem crescente de z_j .
- Encontra as soluções limites p_1 , com menor valor de z_j (z_j^{min}), e p_l , com maior valor de z_j (z_j^{max}).
- Define a *crowding distance* em relação a z_j dessas soluções como sendo ∞ , ou seja, $crd(z_j, p_1) = crd(z_j, p_l) = \infty$.
- Para as demais soluções p_w , com $w = 2, \dots, l - 1$, calcula a *crowding distance* em relação a z_j pela Equação 2.1:

$$crd(z_j, p_w) = \frac{z_j(p_{w+1}) - z_j(p_{w-1})}{z_j^{max} - z_j^{min}} \quad (2.1)$$

2. A *crowding distance* de uma solução p é dada pela soma das *crowding distances* em relação aos m objetivos: $cdr(p) = \sum_{j=1}^m cdr(z_j, p)$.

O operador de comparação (*crowded comparison*), $\prec_n$, usado em vários estágios do processo de seleção para guiar o algoritmo em direção a um fronte de Pareto uniformemente distribuído, considera que cada indivíduo p na população tem dois atributos: o *rank* de não dominação ($rank(p)$) e a *crowding distance* ($crd(p)$). Com isso, o operador $\prec_n$, que compara dois indivíduos p_1 e p_2 é definido como $p_1 \prec_n p_2$ se $rank(p_1) < rank(p_2)$ ou ($rank(p_1) = rank(p_2)$ e $crd(p_1) > crd(p_2)$). Em outras palavras, entre duas soluções que possuem diferentes *ranks* de dominação, a melhor é aquela com menor *rank*. Nos casos em que elas possuem mesmo *rank* (estão no mesmo fronte), a melhor solução é aquela que está em uma região menos povoada.

Com base no procedimento para a ordenação rápida pela não dominação, na *crowding distance* e no operador de comparação descritos, o algoritmo NSGA-II funciona da seguinte maneira:

1. Gera aleatoriamente n^P indivíduos para compor a população inicial P_0 .
2. Ordena P_0 de acordo com a não dominação.
3. Determina a aptidão para cada solução p . Nessa primeira geração, a aptidão de uma solução é igual ao seu nível de não dominação, $rank(p)$. O melhor nível é 1 (a aptidão deverá ser minimizada).
4. Usa seleção por torneio binário, recombinação e mutação para criar uma população filha Q_0 , também de tamanho n^P . Para a seleção, nessa primeira geração, somente o valor da aptidão é considerado: dois indivíduos são sorteados da população P_0 e aquele que tem o menor valor de aptidão é selecionado.
5. Inicializa o número da geração: $t = 0$.
6. Forma uma população combinada $R_t = P_t \cup Q_t$, de tamanho $2n^P$.
7. Ordena R_t de acordo com a não dominação.
8. Seleciona os indivíduos para a nova população P_{t+1} da seguinte maneira:
 - Enquanto $|P_{t+1}| + |F_i| \leq n^P$ (ou seja, o número de indivíduos já adicionados a P_{t+1} juntamente com o tamanho do i -ésimo fronte não excede o número de indivíduos que a população deve ter):
 - Calcula a *crowding distance* para as soluções do fronte F_i .

- Inclui os indivíduos do fronte F_i na população P_{t+1} ($P_{t+1} = P_{t+1} \cup F_i$).
 - $i = i + 1$.
 - Ordena o fronte F_i (último fronte testado anteriormente e que não foi incluído na população P_{t+1}) de acordo com $\prec_n$, em ordem decrescente.
 - Escolhe os primeiros $n^P - |P_{t+1}|$ elementos de F_i e os inclui na população P_{t+1} .
9. Usa seleção por torneio binário, recombinação e mutação para criar a população filha Q_{t+1} de tamanho n^P . Para criar cada indivíduo da população Q_{t+1} (n^P indivíduos serão criados):
- Seleciona dois indivíduos da população P_{t+1} , utilizando torneio binário. Nesse ponto, a seleção por torneio binário é feita utilizando como critério de seleção o operador $\prec_n$, que considera o *rank* de não dominação e a *crowding distance*.
 - Aplica os operadores de recombinação e mutação.
10. $t = t + 1$.
11. Se o número de gerações dado pelo usuário não foi atingido, volta ao passo 6.

Os operadores de recombinação e mutação podem ser quaisquer operadores apropriados. No caso deste trabalho, os operadores empregados não serão os usuais. Como já mencionado, será proposto um operador especial de recombinação. Quanto à mutação, ela não será empregada. Os detalhes e justificativas dessas escolhas serão detalhados no Capítulo 5.

O algoritmo SPEA utiliza uma população externa (*archive*) para armazenar indivíduos não dominados. O tamanho dessa população externa é variável e limitado. Quando o número de soluções não dominadas excede o limite de tamanho da população externa, o número de soluções não dominadas é reduzido com a aplicação de uma técnica de agrupamento que preserva as características do fronte não dominado. A aptidão dos indivíduos nas populações externa e interna é determinado de acordo com o conceito de dominância de Pareto. A aptidão de um indivíduo da população interna é calculada com base apenas nos indivíduos da população externa, não sendo considerada a dominância entre os indivíduos da população interna. A seleção é feita por torneios binários na união de ambas as populações. A diversidade na população é garantida por meio da utilização de um método de nicho baseado em Pareto.

Assim como o SPEA, algoritmo SPEA2 também tem uma população externa. Porém, essa população tem tamanho fixo. Se o número de soluções não dominadas não é suficiente para completar a população externa, indivíduos dominados também são inseridos

na população externa. Se, por outro lado, o número de soluções não dominadas excede o tamanho da população externa, o SPEA2 tem um procedimento para limitar o número de soluções que iterativamente remove indivíduos até que a população tenha o tamanho correto. São removidos os indivíduos que estão mais próximos a outros indivíduos. No SPEA2, também é utilizado um esquema melhorado para o cálculo da aptidão de cada solução que considera quantos indivíduos essa solução domina e por quantos indivíduos ela é dominada. Além disso, esse algoritmo usa uma técnica para estimar a densidade de uma região baseada nos vizinhos mais próximos.

O algoritmo PESA-II incorpora à versão original, PESA (Corne et al. 2000), um esquema para seleção baseado em região, em que ao invés de um único indivíduo, uma região é selecionada. Isso garante ao PESA-II um melhor espalhamento das soluções ao longo do fronte de Pareto. O PESA-II mantém duas populações de soluções: uma interna, de tamanho fixo, e uma externa de tamanho variável e limitado. O propósito da população externa é tirar proveito das boas soluções. Para isso, PESA-II utiliza elitismo, mantendo um conjunto de soluções não dominadas grande e diverso. A população interna é usada para investigar novas soluções por meio dos processos padrão de recombinação e mutação. As soluções na população externa são mantidas em nichos. É mantido um registro do número de soluções que ocupam cada nicho e esse registro é usado para fazer com que as soluções cubram todo o espaço de objetivos, em lugar de se agruparem todas em uma única região. Para isso, as soluções não dominadas que entrariam em uma população externa cheia apenas o farão se elas ocupam um nicho menos cheio do que algumas outras soluções. Além disso, quando a população interna é construída a partir da população externa, os indivíduos são selecionados uniformemente dentre os nichos povoados (todos os nichos contribuem igualmente). A política de seleção baseada em nichos do PESA-II utiliza uma faixa adaptável de equalização e normalização dos valores das funções objetivo, tornando desnecessário o ajuste de parâmetros, que muitas vezes é complicado, e fazendo com que funções objetivo com variações diferentes possam ser prontamente utilizadas. Além disso, qualquer número de objetivos pode ser utilizado.

2.5 Considerações Finais

Neste capítulo foram introduzidos os conceitos de otimização multi-objetivo e de algoritmos evolutivos, dando ênfase aos algoritmos genéticos multi-objetivo. Também foram apresentados os algoritmos multi-objetivo que são de interesse para esta Tese, sendo o algoritmo NSGA-II descrito em detalhes.

No Capítulo 3 serão apresentados os conceitos importantes relacionados ao agrupamento de dados, os algoritmos de agrupamento relevantes para o trabalho desenvolvido nesta Tese e também alguns índices de validação que serão utilizados.

Agrupamento de Dados

3.1 Considerações Iniciais

No Capítulo 2 foram apresentados os conceitos relacionados à otimização multi-objetivo e o uso de algoritmos genéticos para esse fim. Foram também descritos os algoritmos genéticos multi-objetivo que serão utilizados neste trabalho.

Neste capítulo, serão descritos os conceitos básicos de agrupamento, apresentados os algoritmos e técnicas de validação que serão utilizadas e introduzido o tema de agrupamento semi-supervisionado.

3.2 Definições

Técnicas ou algoritmos de agrupamento (*clustering*) permitem a construção de importantes ferramentas para a análise exploratória de dados para os quais existe pouco ou nenhum conhecimento prévio (Jain and Dubes 1988; Handl et al. 2005; Xu and Wunsch 2005). O objetivo de uma técnica de agrupamento é encontrar uma estrutura de *clusters* (grupos) nos dados, em que os objetos pertencentes a cada *cluster* compartilham alguma característica ou propriedade relevante para o domínio do problema em estudo (Jain and Dubes 1988; Handl et al. 2005; Xu and Wunsch 2005). Embora a idéia do que constitui um *cluster* seja intuitiva, não existe uma definição formal única e precisa para esse conceito. Ao contrário, existe uma grande variedade de definições na literatura. Isso é resultado da grande diversidade de visões/objetivos dos pesquisadores de diferentes áreas que utilizam/desenvolvem técnicas de agrupamento. Algumas definições comuns para *cluster* são (Barbara 2000):

***cluster* bem separado:** um *cluster* é um conjunto de pontos tal que qualquer ponto

em um determinado *cluster* está mais próximo (ou é mais similar) a cada outro ponto nesse *cluster* do que a qualquer ponto não pertencente a ele.

***cluster* baseado em centro:** um *cluster* é um conjunto de pontos tal que qualquer ponto em um dado *cluster* está mais próximo (ou é mais similar) ao centro desse *cluster* do que ao centro de qualquer outro *cluster*. O centro de um *cluster* pode ser um centróide, como a média aritmética dos pontos do *cluster* ou um medóide (isto é, o ponto mais representativo do *cluster*).

***cluster* contínuo** (vizinho mais próximo ou agrupamento transitivo): um *cluster* é um conjunto de pontos tal que qualquer ponto em um dado *cluster* está mais próximo (ou é mais similar) a um ou mais pontos nesse *cluster* do que a qualquer ponto que não pertence a ele.

***cluster* baseado em densidade:** um *cluster* é uma região densa de pontos, separada de outras regiões de alta densidade por regiões de baixa densidade.

***cluster* baseado em similaridade:** um *cluster* é um conjunto de pontos que são similares, enquanto pontos em *clusters* diferentes não são similares.

Uma noção intuitiva do que é um *cluster* resulta em um princípio indutivo (Estivill-Castro 2002). A formulação matemática de um princípio indutivo, chamada critério de agrupamento ou função objetivo, consiste de uma forma de selecionar uma estrutura (ou modelo) para representar os *clusters* que melhor se ajuste a um determinado conjunto de dados (Estivill-Castro 2002). Em outras palavras, o critério de agrupamento é uma forma de expressar o objetivo do agrupamento. Esse critério, geralmente, é baseado na definição de *cluster* empregada e/ou em uma distribuição esperada dos dados em um domínio de aplicação específico (Jiang et al. 2004).

Um princípio indutivo associado a um conjunto de dados resulta em um problema de otimização. Em geral, esse problema de otimização é intratável, ou tem uma complexidade muito alta, para ser resolvido para conjuntos de dados grandes. Por isso, a solução do problema é aproximada por alguma heurística que busque um bom equilíbrio entre a qualidade da otimização e o esforço computacional (Estivill-Castro 2002). Na maioria das vezes, essa heurística, representada por um algoritmo, define uma medida de proximidade e um método de busca para encontrar uma partição ótima ou sub-ótima dos dados, de acordo com o critério de agrupamento adotado (Jiang et al. 2004).

O processo de agrupamento compreende diversas etapas que vão desde a preparação dos objetos, até a interpretação dos *clusters* obtidos. A Figura 3.1 resume as etapas do processo de agrupamento com as informações utilizadas e geradas em cada etapa. As

etapas e a figura apresentada são baseadas nas informações apresentadas por Jain et al. (1999) e Barbara (2000).

Figura 3.1: Etapas do processo de agrupamento.

Preparação:

Os objetos a serem agrupados podem representar um objeto físico, como uma cadeira, ou uma noção abstrata, como um estilo de escrita. Tais objetos também são comumente chamados de padrões, exemplos, amostras, instâncias ou pontos. A preparação dos dados para o agrupamento envolve vários aspectos relacionados ao seu pré-processamento e à forma de representação apropriada para sua utilização por um algoritmo de agrupamento.

O pré-processamento pode envolver, por exemplo, normalizações, conversão de tipos e redução do número de atributos por meio de seleção ou extração de características (Jain et al. 1999). Vários trabalhos discutem formas de padronização dos dados, seleção de atributos e outros aspectos relativos à preparação dos dados, como os

de Jain and Dubes (1988), Gordon (1999), He (1999), Jain et al. (1999), Barbara (2000) e Berkhin (2002).

Quanto à representação, na maioria dos casos, os objetos a serem agrupados são representados por uma matriz de objetos $X_{n \times d} = \{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_n\}$, em que $\mathbf{x}_i = \{x_{i1}, x_{i2}, \dots, x_{id}\}$, n é o número de objetos e d é o número de atributos que representam os objetos, isto é, a dimensionalidade dos objetos.

Algumas vezes, apenas a relação de proximidade entre os objetos é conhecida. Algoritmos de agrupamento podem ainda exigir uma forma de representação específica. Além da matriz de objetos, outras duas formas de representação bastante comuns são a matriz e o grafo de proximidade (Jain and Dubes 1988).

Proximidade:

Esta etapa consiste da definição de uma medida de proximidade apropriada ao domínio da aplicação. Essa medida de proximidade pode ser uma medida de similaridade ou de dissimilaridade entre dois objetos. A escolha da medida de proximidade a ser empregada com um algoritmo de agrupamento deve considerar os tipos e escalas dos atributos que definem os objetos e também as propriedades dos dados que o pesquisador deseja focalizar. Por exemplo, o pesquisador deve ter em mente se a magnitude relativa dos atributos descrevendo dois objetos é suficiente ou seu valor absoluto deve ser considerado (Gordon 1999). As medidas de proximidade, em geral, consideram que todos os atributos são igualmente importantes.

Jain and Dubes (1988) e Gordon (1999) descrevem detalhadamente as medidas de proximidade mais apropriadas para cada tipo e escala de atributo possível. Uma das medidas de proximidade mais comum para objetos cujos atributos são todos contínuos é a distância Euclideana (Equação 3.1).

$$d(\mathbf{x}_i, \mathbf{x}_j) = \sqrt{\sum_{l=1}^d (x_{il} - x_{jl})^2} \quad (3.1)$$

Agrupamento:

Esta etapa consiste da aplicação de um algoritmo de agrupamento apropriado para agrupar os dados de acordo com um objetivo específico. Existem inúmeros algoritmos que podem ser aplicados nesta etapa. Os algoritmos de agrupamento de interesse para este trabalho são apresentados na Seção 3.3.

Validação:

Esta etapa se refere à avaliação do resultado de um agrupamento e deve, de forma objetiva, determinar se os *clusters* são significativos, ou seja, se a solução é representativa para o conjunto de dados analisado. Uma estrutura de agrupamento é válida se não ocorreu por acaso ou se é “rara” em algum sentido, já que qualquer algoritmo de agrupamento encontrará *clusters*, independentemente de existir ou não similaridade nos dados (Jain and Dubes 1988). A Seção 3.4 contém uma descrição mais detalhada do processo de validação, bem como de alguns dos índices mais utilizados.

Interpretação:

Refere-se ao processo de examinar cada *cluster* com relação a seus objetos para rotulá-los, descrevendo a natureza do *cluster*. A interpretação de *clusters* é mais que apenas uma descrição. Além de ser uma forma de avaliação dos *clusters* encontrados e da hipótese inicial, de um modo confirmatório, os *clusters* podem permitir avaliações subjetivas que tenham um significado prático. Ou seja, o especialista pode ter interesse em encontrar diferenças semânticas de acordo com os objetos e valores de seus atributos em cada *cluster*.

Mais detalhes sobre cada um desses passos podem ser obtidos em (Faceli et al. 2005a). Especificamente sobre a fase de validação, uma revisão mais completa pode ser obtida em (Faceli et al. 2005d).

Como já mencionado, existe um grande número de algoritmos de agrupamento descritos na literatura (Estivill-Castro 2002; Xu and Wunsch 2005). Não existe, porém, um algoritmo de agrupamento universal, capaz de revelar toda a variedade de estruturas que podem estar presentes em um conjunto de dados. Além disso, como lembra Hartigan (1985), “diferentes agrupamentos são adequados para diferentes propósitos. Dessa forma, não é possível afirmar que um agrupamento é melhor que outro”. Isso tudo leva a dificuldades na escolha do melhor algoritmo a ser aplicado a um problema específico. Apesar de também existir uma grande diversidade de técnicas de validação capazes de auxiliar nessa escolha, em geral, cada uma apresenta uma tendência de favorecer um tipo de algoritmo, por ser baseada no mesmo conceito que o critério de agrupamento dos algoritmos desse tipo (Handl et al. 2005).

Além da dificuldade da escolha do melhor algoritmo para uma dada aplicação, muitos dos algoritmos apresentam restrições. Alguns dos problemas comuns a vários algoritmos de agrupamento são (Jain and Dubes 1988; Handl and Knowles 2005a):

- Adequação a domínios e/ou conjuntos de dados restritos.
- Restrição dos formatos da estrutura que pode ser encontrada.

- Necessidade de conhecimento prévio do número de *clusters* presentes nos dados ou o difícil ajuste de parâmetros.
- Instabilidade dos resultados obtidos. Várias execuções de um algoritmo produzem agrupamentos diferentes, associando um mesmo objeto a *clusters* diferentes.

Várias abordagens de *ensemble* de agrupamentos e agrupamento multi-objetivo têm sido investigadas para superar essas restrições. O Capítulo 4 apresenta algumas dessas abordagens.

Um último tópico relacionado à análise de agrupamento de interesse para este trabalho é o agrupamento semi-supervisionado (Handl and Knowles 2006b; Demiriz et al. 1999), que consiste na integração de conhecimento prévio sobre os dados para guiar o processo de agrupamento. Embora diferente da abordagem tradicional de agrupamento, que é não supervisionada, alguns aspectos do agrupamento semi-supervisionado serão considerados neste trabalho e encontram-se descritos na Seção 3.5.

3.3 Algoritmos Usados

Existe uma grande variedade de algoritmos de agrupamento (Estivill-Castro 2002; Xu and Wunsch 2005). Cada um desses algoritmos emprega um determinado critério de agrupamento, que impõe uma estrutura aos dados. Se, por acaso, os dados estão em conformidade com as exigências do critério empregado, então a estrutura verdadeira de *clusters* é encontrada.

Além dos numerosos algoritmos de agrupamento, também existe uma infinidade de taxonomias para classificá-los. Por exemplo, de acordo com a natureza do agrupamento obtido, os algoritmos de agrupamentos podem ser divididos em exclusivos e não exclusivos (Jain and Dubes 1988). Um agrupamento exclusivo é uma partição de um conjunto de objetos. Cada objeto pertence exclusivamente a um único subconjunto (*cluster*). Esse agrupamento é dito ser do tipo *hard* (um exemplo pertence ou não pertence a um dado *cluster*). Um algoritmo de agrupamento não exclusivo pode associar um objeto a mais de um *cluster*. Os agrupamentos *fuzzy* (cada exemplo tem um grau de pertinência à cada um dos *clusters*) são agrupamentos não exclusivos.

Os algoritmos de agrupamento exclusivos podem ser subdivididos em hierárquicos e particionais, de acordo com o tipo de estrutura imposta aos dados. A estrutura resultante de um algoritmo particional é uma única partição dos dados, enquanto que um agrupamento hierárquico resulta em uma seqüência aninhada de partições.

O foco deste trabalho está nos algoritmos exclusivos ou *hard*. E, apesar da utilização tanto de algoritmos particionais, quanto hierárquicos, o interesse dos autores está nas estruturas na forma de partições que podem ser obtidas com o agrupamento. Mais for-

malmente, dado um conjunto de objetos $X = \{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_n\}$, uma partição de X em K^i clusters é dada por $\pi^i = \{c_1^i, c_2^i, \dots, c_{K^i}^i\}$ com $K^i < n$, tal que (Xu and Wunsch 2005):

1. $c_j^i \neq \emptyset, j = 1, \dots, K^i$,
2. $\bigcup_{j=1}^{K^i} c_j^i = X$ e
3. $c_j^i \cap c_l^i = \emptyset, j, l = 1, \dots, K^i$ e $i \neq l$.

Além da taxonomia descrita anteriormente, os algoritmos de agrupamento podem ser classificados de acordo com o método adotado para definir os clusters (Jain et al. 1999; Halkidi et al. 2001). Neste caso, os algoritmos são divididos em algoritmos hierárquicos, particionais, baseados em *grid* e baseados em densidade. Muitos dos algoritmos se enquadram em mais de uma dessas categorias.

Uma outra taxonomia interessante é a de Handl et al. (2005). Esses autores classificam os algoritmos de agrupamento de acordo com o critério de agrupamento que eles implícita ou explicitamente otimizam. Esses trabalhos aplicam a mesma abordagem para classificar a maioria das medidas internas empregadas em validação de agrupamento. Eles agrupam os critérios/algoritmos de agrupamento em três categorias fundamentais, de acordo com o conceito no qual eles se baseiam (Handl and Knowles 2004; Handl et al. 2005):

- **Compactação** (*Compactness*): a compactação ou homogeneidade de um cluster é geralmente associada a uma variação intra-cluster pequena. Algoritmos que otimizam esse tipo de critério tendem a ser muito efetivos na descoberta de clusters esféricos e/ou bem separados, mas podem falhar para estruturas mais complexas. Alguns exemplos de algoritmos que buscam clusters compactos são *k*-médias (MacQueen 1967), hierárquico com ligação média (*average-link*) (Jain and Dubes 1988), *Self-Organizing Maps* (SOM) (Kohonen 1997) e algoritmos baseados em misturas, como o *Expectation Maximization* (EM) (McLachlan and Krishnan 1997).
- **Encadeamento ou ligação** (*Connectedness*): encadeamento é um conceito mais local, baseado na idéia de que objetos vizinhos devem compartilhar o mesmo cluster. Esse tipo de critério é bastante apropriado para a detecção de clusters de formas arbitrárias, mas não é robusto para os casos em que há pouca separação espacial entre os clusters. Alguns algoritmos que implementam esse conceito são algoritmos baseados em densidade como o OPTICS (*Ordering Points To Identify the Clustering Structure*) (Ankerst et al. 1999) e o DBSCAN (*Density-Based Spatial Clustering of Applications with Noise*) (Ester et al. 1996), o algoritmo hierárquico com ligação simples (*single-link*) (Jain and Dubes 1988) e o algoritmo *Shared Nearest Neighbors* (SNN) (Ertöz et al. 2002).

- **Separação espacial** (*Spatial separation*): por si só a separação espacial fornece pouca orientação durante o processo de agrupamento e pode facilmente levar a soluções triviais. Esse conceito é comumente empregado em associação com outros. Ao contrário dos dois primeiros conceitos, a separação espacial não tem sido otimizada implicitamente (Handl and Knowles 2004).

Além dos algoritmos já citados, existe uma grande variedade de algoritmos descritos na literatura. Uma descrição mais detalhada dos algoritmos citados, bem como alguns outros algoritmos, tanto tradicionais, quanto recentes, pode ser encontrada em (Faceli et al. 2005a). Neste trabalho, serão descritos, a seguir, apenas os algoritmos empregados nos experimentos, que são os algoritmos hierárquicos com Ligação Simples (LS) e Ligação Média (LM), o algoritmo k -médias (KM) e o algoritmo SNN (*Shared Nearest Neighbor*) (Ertöz et al. 2002). A Tabela 3.1 contém uma comparação de algumas características desses algoritmos. Na coluna referente à complexidade, n é o número de objetos, k é o número de *clusters* e d é a dimensionalidade (número de atributos) dos dados.

Tabela 3.1: Características dos algoritmos de agrupamento

Algoritmo	Critério	Formas dos <i>clusters</i>	Complexidade	Robusto frente a ruídos e <i>outliers</i>
KM	compactação	esféricos e compactos	$O(nkd)$	não
LM	compactação	esféricos e compactos	$O(n^2)$	sim
LS	encadeamento	formas arbitrárias e bem separados	$O(n^2)$	não
SNN	encadeamento	formas arbitrárias com tamanhos e densidades variáveis	$O(n^2)$	sim

3.3.1 Algoritmos Hierárquicos - Ligação Simples e Ligação Média

Um método de agrupamento hierárquico é um procedimento para gerar uma estrutura de partições aninhadas na forma de árvore a partir de um conjunto de objetos X . Essa estrutura pode ser definida por $\Pi = \{\pi^1, \pi^2, \dots, \pi^r\}$, com $r < n$, tal que $c_i^m \in \pi^m$, $c_j^l \in \pi^l$ e $m > l$ implica em $c_i^m \subset c_j^l$ ou $c_i^m \cap c_j^l = \emptyset, \forall i, j \neq i, m, l = 1, \dots, r$ (Xu and Wunsch 2005).

Os algoritmos hierárquicos com Ligação Simples (LS) e Ligação Média (LM) são algoritmos hierárquicos aglomerativos (Jain and Dubes 1988; Barbara 2000; Xu and Wunsch 2005), ou seja, começam com n *clusters* com um único objeto (*singleton*) e constroem a hierarquia de partições agrupando sucessivamente os *clusters* mais próximos. As medidas de proximidade entre os *clusters* são as métricas de ligação. A diferença entre o algoritmo com ligação simples e o algoritmo com ligação média está somente na métrica de

ligação que eles empregam, ou seja, na medida em que usam para determinar o quanto dois *clusters* estão próximos.

No caso do algoritmo com ligação simples, a proximidade entre os *clusters* é dada pela menor distância entre todos os possíveis pares de objetos $\mathbf{x}_i$ e $\mathbf{x}_j$, com $\mathbf{x}_i \in c_a$ e $\mathbf{x}_j \in c_b$ - Equação 3.2 (Barbara 2000).

$$d(c_a, c_b) = \min_{\mathbf{x}_i \in c_a, \mathbf{x}_j \in c_b} d(\mathbf{x}_i, \mathbf{x}_j) \quad (3.2)$$

No algoritmo com ligação média a proximidade entre dois *clusters* é dada pela média aritmética das distâncias entre todos os possíveis pares de objetos $\mathbf{x}_i$ e $\mathbf{x}_j$, com $\mathbf{x}_i \in c_a$ e $\mathbf{x}_j \in c_b$ - Equação 3.3 (Barbara 2000).

$$d(c_a, c_b) = \frac{1}{|c_a||c_b|} \sum_{\mathbf{x}_i \in c_a} \sum_{\mathbf{x}_j \in c_b} d(\mathbf{x}_i, \mathbf{x}_j) \quad (3.3)$$

Dado um conjunto de objetos X , esses algoritmos seguem o seguinte procedimento:

1. Construir uma matriz de proximidade entre *clusters*, D , a partir do conjunto de dados X . Inicialmente, cada *cluster* é composto por um dos objetos de X . A proximidade entre dois *clusters* c_a e c_b , $d(c_a, c_b)$ é calculada utilizando a Equação 3.2 ou a Equação 3.3, dependendo do algoritmo hierárquico utilizado.
2. Encontrar o par de *clusters* mais próximos, c_a e c_b , e uní-los em um único *cluster* c_{ab} .
3. Atualizar a matriz D . Para isso, eliminar as linhas e colunas correspondentes aos *clusters* c_a e c_b e acrescentar uma linha e uma coluna para o novo *cluster* c_{ab} , calculando novamente as proximidades pela Equação 3.2 ou pela Equação 3.3.
4. Se uma condição de parada não for satisfeita, voltar ao passo 2. A condição de parada pode ser, por exemplo, atingir um determinado número de *clusters*, ou ter formado um único *cluster*.

No agrupamento hierárquico, as soluções são tipicamente representadas por um dendrograma, que consiste de um tipo especial de estrutura de árvore. Um dendrograma consiste de camadas de nós, cada um representando um *cluster*. Linhas conectam nós representando *clusters* aninhados. O corte de um dendrograma na horizontal representa uma partição. A Figura 3.2 contém uma ilustração de um dendrograma. O corte no dendrograma indicado pela linha tracejada representa uma partição do conjunto de objetos $X = \{a, b, c, d, e, f, g, h, i, j, k\}$ com três *clusters*: $c_1 = \{g, a, c, i, e\}$, $c_2 = \{d, k, b, j\}$ e $c_3 = \{f, h\}$.

Figura 3.2: Dendrograma

Algoritmos de agrupamento hierárquico resultam em *clusters* de formas convexas próprias e possuem complexidade $O(n^2)$ (Xu and Wunsch 2005). Eles não requerem a especificação do número de *clusters* (Barbara 2000). Esses algoritmos não têm uma função objetivo global. São baseados em decisões locais e após um objeto ter sido associado a um *cluster*, ele não é considerado novamente, o que significa que esses algoritmos não são capazes de corrigir uma associação incorreta.

O critério otimizado pelo algoritmo LS está relacionado ao conceito de encadeamento discutido previamente (Handl and Knowles 2004). Particularmente, esse algoritmo é bom para manipular formas não elípticas, mas é bastante sensível a ruídos e *outliers* (Barbara 2000). Já o algoritmo LM otimiza um critério relacionado à compactação. Ele é menos sensível a ruídos e *outliers*, mas se limita a encontrar *clusters* de formas esféricas.

3.3.2 *k*-médias

O algoritmo *k*-médias (KM) (MacQueen 1967; Duda et al. 2001) particiona o conjunto de dados em k *clusters*, sendo k um parâmetro do algoritmo. Esse algoritmo utiliza uma técnica de realocação iterativa, que encontra um ótimo local. Existem várias versões do algoritmo, cada uma solucionando uma deficiência do algoritmo original. Por exemplo, a distância de Mahalanobis pode ser utilizada para encontrar *clusters* hiper-elipsóides, em vez de *clusters* hiper-esféricos como os encontrados na versão original. Berkhin (2002) e Jain et al. (1999) discutem brevemente algumas dessas versões.

O critério de agrupamento que o *k*-médias minimiza é o erro quadrático ou variância intra-*cluster*, var , dado pela Equação 3.4, em que μ_l é o centróide do *cluster* c_l e $d(\mathbf{x}_i, \mu_l)$ é a distância Euclideana entre $\mathbf{x}_i$ e μ_l . O centróide pode ser a média aritmética ou a mediana do *cluster*. Em outras palavras, o critério seguido pelo *k*-médias é minimizar a distância entre cada ponto e o centróide do *cluster* ao qual o ponto pertence (Halkidi et al. 2001).

$$var = \sum_{l=1}^k \sum_{\mathbf{x}_i \in c_l} d(\mathbf{x}_i, \mu_l) \quad (3.4)$$

Dado um conjunto de dados X e um número de *clusters* k , em que X deve ser particionado, o k -médias funciona da seguinte maneira (Jain et al. 1999):

1. Inicializar um conjunto de k centróides $C = \{\mu_l, l = 1, \dots, k\}$ para os *clusters*. Essa inicialização pode ser aleatória ou seguindo alguma heurística.
2. Associar cada objeto $\mathbf{x}_i \in X$ ao *cluster* com o centróide mais próximo.
3. Recalcular os centróides.
4. Se um critério de parada não foi satisfeito, voltar ao passo 2. O critério de parada pode ser, por exemplo, os centróides não serem mais alterados ou a ocorrência de um determinado número de iterações.

A complexidade do algoritmo k -médias é $O(nkd)$ (Xu and Wunsch 2005). Ele é sensível à escolha dos centróides iniciais e da sua forma de atualização. Dependendo da escolha dos centróides, o algoritmo converge para um ótimo local. O k -médias encontra *clusters* de formato globular (hiper-esféricos) de igual tamanho ou *clusters* bem separados.

3.3.3 Shared Nearest Neighbor (SNN)

O algoritmo *Shared Nearest Neighbor* (SNN) otimiza um critério relacionado ao conceito de encadeamento ou ligação, mas tem características que superam outros algoritmos que se baseiam nesse critério (Ertöz et al. 2002; Ertöz et al. 2003a; Ertöz et al. 2003b). Dentre outros tipos de *cluster*, ele detecta aqueles com densidades baixas ou médias que representam regiões relativamente uniformes cercadas por áreas não uniformes ou de densidade maior, que outros algoritmos baseados nesse tipo de critério deixam escapar (Ertöz et al. 2003b).

O SNN encontra os vizinhos mais próximos de cada objeto utilizando o número de vizinhos que dois objetos compartilham como medida de proximidade entre dois objetos. Com essa medida de proximidade, o algoritmo emprega uma abordagem baseada em densidade para encontrar pontos representativos e constrói *clusters* ao redor desses pontos. Além do número de vizinhos mais próximos a serem considerados (NN), dois outros tipos de parâmetros são empregados: (1) parâmetros relacionados ao peso das conexões do grafo dos vizinhos mais próximos compartilhados ou “similaridade SNN” (*strong, merge e label*); (2) parâmetros relacionados ao número de conexões fortes ou “densidade SNN” (*topic e noise*). Esses parâmetros são *thresholds* nos quais cada passo do algoritmo se baseia.

A seguir, são detalhados os passos desse algoritmo (Ertöz et al. 2003a; Ertöz et al. 2003b).

1. Calcular uma matriz de proximidade que contém a proximidade entre cada par de objetos em X .
2. Encontrar os NN vizinhos mais próximos de cada objeto.
3. Construir o grafo dos vizinhos mais próximos compartilhados (grafo SNN):
 - Determinar o número de vizinhos compartilhados por todo par de objetos.
 - Acrescentar uma conexão entre os objetos $\mathbf{x}_i$ e $\mathbf{x}_j$ se ambos têm o outro em sua lista de vizinhos mais próximos.
 - Tornar o peso da conexão igual ao número de vizinhos mais próximos compartilhados por $\mathbf{x}_i$ e $\mathbf{x}_j$. Esse peso é a “similaridade SNN” entre os dois objetos, $sim(\mathbf{x}_i, \mathbf{x}_j)$.
4. Calcular a “densidade SNN” (também chamada de conectividade) de cada objeto. Se o peso de uma conexão do grafo SNN (ou seja, a “similaridade SNN”) é maior do que o *threshold strong*, então essa conexão é dita forte. A “densidade SNN” de um objeto $\mathbf{x}_i$, $den(\mathbf{x}_i)$ é dada pelo número de conexões fortes que o objeto $\mathbf{x}_i$ possui.
5. Para cada objeto $\mathbf{x}_i$:
 - Se $den(\mathbf{x}_i) < noise$ então descartar o objeto $\mathbf{x}_i$.
 - Se $den(\mathbf{x}_i) > topic$ então $\mathbf{x}_i$ é similar à maioria dos seus vizinhos e é escolhido como objeto representativo de sua vizinhança.
6. Para cada par de objetos de X , $(\mathbf{x}_i, \mathbf{x}_j)$, se $sim(\mathbf{x}_i, \mathbf{x}_j) > merge$ e se pelo menos um deles é um objeto representativo, então, colocar $\mathbf{x}_i$ e $\mathbf{x}_j$ no mesmo *cluster*.
7. Associar a *clusters* os objetos que não são ruídos e nem representativos de algum *cluster*. Para isso, percorrer a lista dos vizinhos mais próximos compartilhados de todos os objetos que já estão associados a algum *cluster*, procurando por conexões com objetos que ainda não pertencem a nenhum *cluster*. Os objetos não associados a um *cluster* e cujo peso da conexão com um objeto representativo é maior do que *label*, são associados ao mesmo *cluster* do objeto representativo.

A complexidade do SNN é $O(n^2)$. Esse algoritmo é bom para encontrar *clusters* com formas, tamanhos e densidades diferentes. Ele também lida bem com ruídos e *outliers*. O SNN também é apropriado para lidar com dados de alta dimensionalidade (Ertöz et al. 2002; Ertöz et al. 2003a; Ertöz et al. 2003b).

3.4 Validação de Agrupamentos

A validação de um agrupamento, em geral, é feita com base em índices estatísticos, que julgam, de uma maneira quantitativa e objetiva, o mérito das estruturas encontradas (Jain and Dubes 1988). Um índice quantifica alguma informação a respeito da qualidade de um agrupamento. A maneira pela qual um índice é aplicado para validar um agrupamento é dada pelo critério de validação. Assim, um critério de validação expressa a estratégia utilizada para validar uma estrutura de agrupamento, enquanto que um índice é uma estatística pela qual a validade é testada (Jain and Dubes 1988). Estudos sobre processos de validação de agrupamentos podem ser encontrados em (Jain and Dubes 1988; Gordon 1999; Halkidi et al. 2001; Halkidi et al. 2002a; Halkidi et al. 2002b; Handl et al. 2005).

Atualmente, existem três tipos de critérios para investigar a validade de um agrupamento (Jain and Dubes 1988):

Critérios relativos: comparam diversos agrupamentos para decidir qual deles é o melhor em algum aspecto (qual é o mais estável ou qual apresenta o melhor valor para uma dada medida, por exemplo). Podem ser utilizados para comparar diversos algoritmos de agrupamento ou para determinar o valor mais apropriado para um parâmetro de um algoritmo. Por exemplo, pode-se medir quantitativamente qual das hierarquias melhor se ajusta aos dados, se a obtida com o LM ou se a obtida com o LS. Outro exemplo é, dado um conjunto de partições com diferentes números de *clusters*, obtidas com o KM, determinar qual é o número de *clusters* mais apropriado.

Critérios internos: medem a qualidade de um agrupamento com base apenas nos dados originais (matriz de objetos ou matriz de proximidade). Por exemplo, um critério interno pode medir o grau em que uma partição obtida por um algoritmo de agrupamento é justificado pela matriz de proximidade.

Critérios externos: avaliam um agrupamento de acordo com uma estrutura pré-especificada, imposta ao conjunto de dados, e que reflete a intuição do pesquisador sobre a estrutura presente nos dados. Essa estrutura pré-especificada pode ser uma partição que se sabe previamente existir nos dados, ou um agrupamento construído por um especialista da área com base em conhecimento prévio. Por exemplo, um critério externo pode medir o grau de correspondência entre o número *clusters* obtidos com o agrupamento e os rótulos dos dados conhecidos previamente.

Existem vários índices que podem ser empregados com critérios relativos. Esses índices, em geral, podem também ser empregados em critérios internos (Jain and Dubes 1988). O que distingue a utilização de um índice em um ou outro critério é a maneira como o índice é aplicado. A forma mais comum de aplicação de um índice como um critério

relativo consiste do cálculo do seu valor para vários agrupamentos que estão sendo comparados, obtendo-se uma seqüência de valores. O melhor agrupamento é determinado pelo valor que se destaca nessa seqüência, como o valor máximo, mínimo ou o de inflexão na curva do gráfico construído com a seqüência (Jain and Dubes 1988).

Os critérios externos e internos são baseados em testes estatísticos e têm um alto custo computacional (Halkidi et al. 2001). Seu objetivo é medir o quanto o resultado obtido confirma uma hipótese pré-especificada. Neste caso, são utilizados testes de hipótese para determinar se uma estrutura obtida é apropriada para os dados. Isso é feito testando se o valor do índice utilizado é significativamente grande ou pequeno, o que requer o estabelecimento de uma população base ou de referência. O mesmo índice pode ser utilizado em um critério externo e interno, embora as distribuições de referência do índice sejam diferentes (Jain and Dubes 1988; Gordon 1999). A diferença entre esses critérios está nas informações utilizadas para o cálculo do índice. Nos critérios externos, como já mencionado, o cálculo do índice considera a utilização de uma estrutura dos dados conhecida previamente, enquanto nos critérios internos, apenas os dados são considerados.

Um grande problema em validação externa ou interna de agrupamentos é o estabelecimento da distribuição dos índices (estatísticas) sob a hipótese nula e conseqüentemente a determinação dos limiares que dizem se uma partição/hierarquia é adequada de acordo o índice. Na prática, os testes de validação são geralmente definidos utilizando ferramentas estatísticas, como análise de Monte Carlo e *bootstrapping* (Jain and Dubes 1988). Porém, essas ferramentas são bastante custosas e de difícil aplicação.

Uma revisão mais completa das abordagens de validação e dos índices mais comuns pode ser obtida em (Faceli et al. 2005d).

Neste trabalho, a avaliação dos resultados será feita utilizando um índice externo, o Rand corrigido (CR , do inglês *Corrected Rand*) (Hubert and Arabie 1985). O CR é um dos índices de validação externa mais utilizados nas avaliações e comparações de algoritmos de agrupamento. Além disso, esse índice não é sensível ao número de *clusters*, como outros índices de validação externa tradicionais (Jain and Dubes 1988).

O índice Rand determina a similaridade entre duas partições, π^a e π^b , pela concordância, positiva ou negativa, na associação de pares de objetos aos *clusters*. Ou seja, o índice penaliza as associações diferentes de pares de objetos nas duas partições. Por exemplo, se dois objetos $\mathbf{x}_1$ e $\mathbf{x}_2$ são associados ao mesmo *cluster* em π^a e a *clusters* diferentes em π^b , isso resulta em um decréscimo do valor do índice. O CR é uma normalização do índice Rand, proposta por Hubert and Arabie (1985) e é dado pela Equação 3.5. Esse índice apresenta um valor próximo de 0 (podendo inclusive ser negativo) para partições aleatórias e o valor 1 quando as partições casam perfeitamente (Jain and Dubes 1988; Gordon 1999). Para a sua utilização com informação externa, uma das partições deve ser

uma estrutura dos dados previamente conhecida, enquanto a outra partição é a que está sendo avaliada.

$$CR = \frac{\sum_{i=1}^{K^a} \sum_{j=1}^{K^b} \binom{|c_i^a \cap c_j^b|}{2} - \left[\sum_{i=1}^{K^a} \binom{|c_i^a|}{2} \sum_{j=1}^{K^b} \binom{|c_j^b|}{2} \right] / \binom{n}{2}}{\left[\sum_{i=1}^{K^a} \binom{|c_i^a|}{2} + \sum_{j=1}^{K^b} \binom{|c_j^b|}{2} \right] / 2 - \left[\sum_{i=1}^{K^a} \binom{|c_i^a|}{2} \sum_{j=1}^{K^b} \binom{|c_j^b|}{2} \right] / \binom{n}{2}} \quad (3.5)$$

Outros dois índices de interesse para este trabalho, que não dependem de informações externas, são a variância intra-*cluster* - também chamada desvio total em (Handl and Knowles 2004; Handl and Knowles 2005a) e homogeneidade média em (Shamir and Sharan 2002), e a conectividade (Handl et al. 2005).

A variância intra-*cluster* (*var*) de uma partição π é calculado como a soma total das distâncias entre os objetos e o centro dos seus *clusters*. Essa medida corresponde ao mesmo critério otimizado no *k*-médias. Ela mede a qualidade de um agrupamento em termos da compactação ou homogeneidade de seus *clusters*. Formalmente, *var* é dada pela Equação 3.6, em que μ_k é o centróide do *cluster* c_k e $d(.,.)$ é a função de distância (no *k*-médias, a distância Euclideana).

$$var(\pi) = \sum_{c_k \in \pi} \sum_{\mathbf{x}_i \in c_k} d(\mathbf{x}_i, \mu_k) \quad (3.6)$$

Quanto menor o valor do índice *var*, melhor a partição. Contudo, essa medida favorece *clusters* esféricos e não apresenta bons resultados para agrupamentos cujos *clusters* não estejam bem separados. O valor dessa medida melhora com o aumento do número de *clusters*.

A conectividade (*con*) está ligada ao conceito de encadeamento ou ligação. Esse índice reflete o grau com que objetos vizinhos são colocados no mesmo *cluster* em uma partição. Ele é dado pela Equação 3.7, em que nn_{ij} é o *j*-ésimo vizinho mais próximo do objeto $\mathbf{x}_i$, e *v* é o número de vizinhos mais próximos que contribuem para a conectividade.

$$con(\pi) = \sum_{i=1}^n \sum_{j=1}^v a(\mathbf{x}_i, nn_{ij}) \quad (3.7)$$

$$a(\mathbf{x}_i, nn_{ij}) = \begin{cases} \frac{1}{j}, & \text{se } \nexists c_k : \mathbf{x}_i, nn_{ij} \in c_k \\ 0, & \text{caso contrário} \end{cases} \quad (3.8)$$

Quanto menor o valor do índice *con*, melhor o agrupamento. Essa medida não apresenta restrições quanto a forma dos *clusters*, isto é, ela é apropriada para a avaliação

de *clusters* de formas arbitrárias. Entretanto, o índice *con*, da mesma maneira que o índice *var*, não é robusto para *clusters* sobrepostos. O valor dessa medida melhora com a diminuição do número de *clusters*.

3.5 Agrupamento Semi-supervisionado

Como já mencionado, o agrupamento semi-supervisionado utiliza conhecimento prévio sobre os dados para guiar o processo de agrupamento (Handl and Knowles 2006b; Demiriz et al. 1999). Para melhorar o desempenho dos métodos de classificação tradicionais, os algoritmos semi-supervisionados baseiam-se em aspectos tanto do aprendizado supervisionado, que considera o conhecimento prévio das classes presentes nos dados, quanto do aprendizado não supervisionado, que se baseia unicamente na distribuição subjacente dos dados (Handl and Knowles 2006b).

Algoritmos de agrupamento semi-supervisionados são indicados para dados que contêm uma pequena parte dos objetos rotulados, ou seja, dos quais se conhecem as classes a que cada um pertence, e a grande maioria dos objetos sem rótulos.

É preciso destacar que as classes são entidades relacionadas a categorias previamente definidas, do mundo real, que são utilizadas para separar os objetos de um dado problema. Já os *clusters* que compõem um agrupamento dos dados são entidades definidas pela aplicação de conceitos matemáticos/estatísticos aos dados. As classes podem ou não estar relacionadas com um ou mais conceitos matemáticos/estatísticos e não têm necessariamente uma correspondência com os *clusters*.

A abordagem semi-supervisionada assume que as informações referentes às classes e aos *clusters* dos dados são consistentes, se complementam e seu uso combinado pode melhorar a precisão da classificação dos dados (Handl and Knowles 2006b). Se essas suposições são fortemente violadas, ou seja, se as informações conhecidas das classes não estão relacionadas ao critério de agrupamento empregado, não se pode esperar melhora nos resultados com a aplicação dessa abordagem.

Existem várias maneiras pelas quais o conhecimento prévio das classes pode ser integrado no processo de agrupamento (Handl and Knowles 2006b). Exemplos são adaptações no esquema de inicialização, na medida de distância e na função objetivo (critério de agrupamento) do algoritmo de agrupamento empregado. Um exemplo de adaptação da inicialização é a utilização dos dados rotulados para gerar os *clusters* iniciais a partir dos quais o agrupamento será feito. A adaptação da função de distância tem a vantagem de poder ser integrada facilmente em quase todos os algoritmos de agrupamento. Já a otimização de um critério semi-supervisionado, segundo Handl and Knowles (2006b), em geral requer a utilização de um algoritmo de otimização de propósito geral, como os Algoritmos Genéticos, (AGs).

O interesse deste trabalho está voltado para os métodos que adaptam o critério de agrupamento para considerar a informação prévia. Nesse sentido, dois trabalhos descritos na literatura podem ser destacados: o trabalho de Demiriz et al. (1999) e o trabalho de Handl and Knowles (2006b). Demiriz et al. (1999) propõem um método para agrupar os dados que visa produzir *clusters* tão puros quanto possível em termos da distribuição de classes. Os autores utilizam um AG para minimizar uma função que é a combinação linear de uma medida de dispersão dos *clusters* (não supervisionada) e uma medida de impureza em relação às classes conhecidas (supervisionada). Com relação à medida de dispersão, Demiriz et al. (1999) realizaram testes com duas medidas: variância intra-*cluster* (Equação 3.6) e o índice Davies-Bouldin (Jain and Dubes 1988; Faceli et al. 2005d), que considera a dispersão dentro dos *clusters* e a dispersão entre os *clusters*.

Para calcular a impureza, Demiriz et al. (1999) empregam o índice Gini (Rokach and Maimon 2005), utilizado freqüentemente na literatura para determinar a impureza de uma dada divisão dos nós em árvores de decisão. A utilização desse índice favorece soluções com *clusters* puros, mesmo às custas de um aumento do erro total de classificação.

O trabalho de Handl and Knowles (2006b) é uma extensão do algoritmo MOCK (*Multi-Objective Clustering with automatic K-determination*) para considerar o conhecimento prévio. Para isso, uma terceira função objetivo que utiliza as informações a respeito das classes é adotada. Mais especificamente, os autores empregam o índice *CR* (Equação 3.5), computado apenas usando os dados rotulados. A utilização desse índice fornece uma avaliação do grau com o qual o conhecimento prévio foi preservado em uma dada solução.

Uma formulação alternativa com dois objetivos também é utilizada por Handl and Knowles (2006b). Nesse caso, o índice *CR* continua sendo empregado para incorporar o conhecimento prévio e o índice largura da silhueta (*silhouette width*) é usado como o objetivo não supervisionado. A largura da silhueta é uma medida de validação de agrupamento bastante popular e mede o quanto os objetos estão bem situados em seus *clusters* (Rousseeuw 1987; Faceli et al. 2005d).

Handl and Knowles (2006b) compararam diferentes métodos semi-supervisionados, não supervisionados e supervisionados e observaram que a abordagem multi-objetivo obteve um desempenho mais consistente em diferentes conjuntos de dados e foi mais robusta a ruído. Nesse trabalho, Handl and Knowles (2006b) empregaram o conhecimento externo para selecionar a melhor solução do fronte de Pareto.

Em resumo, as abordagens de agrupamento semi-supervisionado descritas na literatura consideram que uma pequena parte dos objetos está rotulada e grande parte deles não está. Seu objetivo principal é a melhora de desempenho em relação às técnicas puramente supervisionadas e não supervisionadas na obtenção de uma estrutura que sirva como modelo para classificação futura de novos objetos.

3.6 Considerações Finais

Neste capítulo foram descritos todos os aspectos referentes à agrupamento de interesse para este trabalho. Foram definidos os principais conceitos referentes ao assunto, com uma descrição resumida de todas as etapas do processo de agrupamento. As técnicas empregadas em cada etapa que são utilizadas neste trabalho foram descritas em detalhes. Isso inclui os algoritmos de agrupamento bem como os índices de validação e medidas de distância.

Um outro tópico, relacionado aos demais apenas por envolver a utilização de técnicas de agrupamento, também foi resumidamente apresentado: o agrupamento semi-supervisionado. Seu objetivo é estritamente diferente do agrupamento não supervisionado tradicional, mas alguns pontos relativos ao agrupamento semi-supervisionado são considerados na abordagem proposta nesta Tese.

No Capítulo 4 serão abordadas as técnicas de *ensemble* de agrupamentos e agrupamento multi-objetivo que visam superar algumas limitações da análise de agrupamento tradicional.

Ensembles e Agrupamento Multi-objetivo

4.1 *Considerações Iniciais*

Atualmente, existem dois focos principais que têm sido seguidos para o desenvolvimento de abordagens que superem as limitações e dificuldades encontradas na análise de agrupamento tradicional:

- *ensembles* de agrupamentos (Fred 2001; Fred and Jain 2002; Strehl and Ghosh 2002; Ghosh et al. 2002; Fred and Jain 2003; Topchy et al. 2003; Topchy et al. 2004; Fern and Brodley 2004; Law et al. 2004; Kuncheva et al. 2002) e
- agrupamento multi-objetivo (Handl and Knowles 2004; Handl and Knowles 2005a; Handl and Knowles 2005b; Handl and Knowles 2006a).

Segundo Handl and Knowles (2004), os *ensembles* de agrupamentos atuam por meio da combinação posterior de um conjunto de agrupamentos obtidos previamente (com um único algoritmo ou vários algoritmos complementares). A Seção 4.2 contém uma descrição dos principais conceitos relacionados aos *ensembles* de agrupamentos, uma comparação entre várias técnicas existentes e uma descrição mais detalhada das técnicas de interesse para este trabalho. Handl and Knowles (2004), assim como outros autores, concordam que os *ensembles* são mais robustos e fornecem soluções de melhor qualidade que os algoritmos individuais, mas argumentam que essas técnicas não exploram todo o potencial de se usar múltiplos objetivos, pois *clusters* que não podem ser detectados por algum dos membros do *ensemble*, provavelmente não aparecerão na solução final. Assim, eles propõem a

abordagem de agrupamento multi-objetivo, que consiste da otimização direta de vários objetivos (critérios de agrupamento) simultaneamente. Essa abordagem será detalhada na Seção 4.3.

Existem outros trabalhos que também se nomeiam de agrupamento multi-objetivo, como o de Law et al. (2004), mas que, apesar de utilizarem vários objetivos (critérios de agrupamento), esses objetivos não são otimizados diretamente. Como nos *ensembles* de agrupamentos, os autores partem de um conjunto de partições iniciais geradas com diferentes algoritmos de agrupamento. Os diferentes algoritmos é que contemplam os diferentes critérios de agrupamento. A partir dessas partições iniciais, Law et al. (2004) selecionam os *clusters* mais adequados para compor a partição final. Apesar de Law et al. (2004) designarem sua técnica como agrupamento multi-objetivo e explicitamente diferenciá-la dos *ensembles*, ela se encaixa mais nos *ensembles* heterogêneos (partições geradas por diferentes algoritmos) do que na otimização direta de diferentes critérios. Esses autores utilizam o termo *ensembles* apenas para os *ensembles* homogêneos (partições geradas com um mesmo algoritmo).

4.2 Ensembles de Agrupamentos

A combinação de estimadores independentes em comitês, ou *ensembles*, é uma técnica comumente empregada em problemas de classificação e regressão com o objetivo de melhorar a precisão de estimadores individuais, aproveitando as características intrínsecas de cada um (LeBlanc and Tibshirani 1993; Krogh and Vedelsby 1995; Merz 1998; Sharkey 1999; Gama 1999; Kuncheva 2004). Porém, a aplicação direta das técnicas de combinação de estimadores aos algoritmos de agrupamento não é possível (Topchy et al. 2003). Os algoritmos de classificação e regressão pertencem ao paradigma de aprendizado supervisionado, em que a resposta correta acompanha cada objeto utilizado na construção do modelo.

Já os algoritmos de agrupamento pertencem ao paradigma de aprendizado não supervisionado. Neste caso, os dados não possuem uma resposta correta associada, ou seja, não existe informação sobre a “resposta verdadeira” que se espera obter com a aplicação do algoritmo. Essa ausência da “resposta verdadeira” é um dos aspectos que impedem a aplicação direta das técnicas de combinação de estimadores (Topchy et al. 2003). Assim, os *ensembles* de agrupamentos requerem técnicas novas ou as adaptações das técnicas usadas em combinação de estimadores, ou seja, as técnicas utilizadas para combinação em aprendizado supervisionado servem apenas como uma base, a partir da qual se pode extrair idéias e direções para o desenvolvimento de técnicas para combinação de algoritmos de agrupamento.

Uma definição simplificada do problema de combinação de algoritmos de agrupamento

é (Topchy et al. 2003): dado um conjunto de n^I partições, $\Pi = \{\pi^1, \pi^2, \dots, \pi^{n^I}\}$, de um conjunto de dados X resultantes de várias aplicações de um ou mais algoritmos de agrupamento, encontrar uma partição final π^F (partição consenso) de melhor qualidade do que as partições iniciais (partições base). Essa “melhor qualidade” depende do objetivo que se deseja atingir com a combinação, que pode estar relacionado à robustez (Strehl and Ghosh 2002; Topchy et al. 2004; Fern and Brodley 2004), novidade (Topchy et al. 2003; Law et al. 2004), estabilidade e estimação da confiança (Monti et al. 2003; Topchy et al. 2004), computação distribuída, paralelização e escalabilidade (Strehl and Ghosh 2002; Topchy et al. 2004), reuso de conhecimento (Strehl and Ghosh 2002; Ghosh et al. 2002; Topchy et al. 2004), consistência (Fred 2001; Fred and Jain 2002; Fred and Jain 2003), e desempenho e custo (Topchy et al. 2003). Mais especificamente, neste trabalho, os objetivos para a utilização dos *ensembles* são:

- **Robustez:** desenvolver uma forma de agrupamento mais robusta, isto é, que tenha um desempenho médio melhor para diversos domínios e conjuntos de dados, de preferência sem a necessidade de muitos ajustes manuais.
- **Novidade:** encontrar uma partição final que não pode ser obtida com nenhum algoritmo, individualmente.
- **Estabilidade:** obter soluções de agrupamento com menor sensibilidade a ruídos, *outliers*, variações de amostragem ou à variabilidade dos algoritmos.

Segundo Fred and Jain (2003), a partição obtida pela combinação das várias partições iniciais deve ser consistente ou concordar de alguma forma com essas partições, não ser sensível a pequenas variações nas partições individuais e ser consistente com informações externas sobre a estrutura dos dados, se essas informações estiverem disponíveis.

As principais tarefas, ou desafios, inerentes ao problema de combinação de múltiplos agrupamentos são: geração dos agrupamentos a serem combinados e determinação de uma função consenso para combinar os agrupamentos (Topchy et al. 2004; Kuncheva et al. 2002; Hadjitodorov et al. 2006). Assim como existem diferentes objetivos para combinar várias partições, também existem várias maneiras para se resolver essas questões. Nas Seções 4.2.1 e 4.2.2 são resumidas as abordagens mais comuns descritas na literatura para gerar a diversidade necessária nas partições iniciais e para encontrar a partição consenso, sendo ressaltadas as abordagens de interesse para este trabalho. Na Seção 4.2.3 são detalhadas as duas técnicas de *ensemble* que serão utilizadas.

4.2.1 Geração dos Agrupamentos Iniciais

As partições a serem combinadas devem ser diferentes, de forma a acrescentar informações úteis para a partição final. Por isso, ao se buscar uma partição consenso, é preciso

ter muito claro o objetivo do agrupamento final e conhecer bem os algoritmos a serem combinados, a fim de garantir que os algoritmos escolhidos possam contribuir para atingir esse objetivo. Assim, de acordo com o objetivo da combinação, deve-se escolher algoritmos de agrupamento, ou formas de aplicação de um algoritmo, que forneçam a diversidade necessária. As maneiras usadas atualmente para se obter essa diversidade são (Kuncheva 2004; Kuncheva et al. 2002; Hadjitodorov et al. 2006):

- Utilizar diferentes algoritmos convencionais de agrupamento para gerar as partições (Qian and Suen 2000; Kellam et al. 2001; Strehl and Ghosh 2002; Zeng et al. 2002; Weingessel et al. 2003).
- Executar várias vezes um mesmo algoritmo convencional, com diferentes inicializações (Fred 2001; Fred and Jain 2002; Fred and Jain 2003; Frossyniotis et al. 2002; Weingessel et al. 2003; Topchy et al. 2004).
- Empregar algoritmos mais simples do que os convencionais, chamados de algoritmos de agrupamento fracos (Topchy et al. 2003).
- Avaliar diferentes conjuntos de dados, que podem ser referentes aos mesmos objetos, cada um com um subconjunto dos atributos originais (Strehl and Ghosh 2002), ou com projeções do conjunto original em um espaço de dimensão menor (Fern and Brodley 2004), ou ainda referentes a subconjuntos de objetos (reamostragem), com todas as características originais (Strehl and Ghosh 2002; Monti et al. 2003; Fern and Brodley 2004).

Nos casos em que as partições base são geradas por um mesmo algoritmo, o *ensemble* é dito homogêneo. Um *ensemble* heterogêneo combina partições geradas com algoritmos diferentes. Neste trabalho, os *ensembles* formados serão heterogêneos. Além disso, a diversidade das partições iniciais será garantida pelo emprego das duas primeiras abordagens. No primeiro caso, devem ser considerados algoritmos com características bastante distintas, para que as partições geradas possam ser bastante diversificadas. No segundo caso, dependendo das características de cada algoritmo, deve ser estabelecido um conjunto de condições iniciais com as quais o algoritmo deve ser executado para gerar as partições. Por exemplo, o algoritmo k -médias depende da inicialização dos centróides e tem como parâmetro o valor de k . Assim, para esse algoritmo, pode-se gerar partições com vários valores de k , ou partições com um mesmo valor de k , mas geradas com diferentes inicializações dos centróides. Para o algoritmo SNN, pode-se gerar partições diferentes variando cada um de seus parâmetros.

4.2.2 Determinação da Função Consenso

A utilização de uma função consenso é a forma empregada para encontrar uma partição consenso (partição final gerada pela combinação) a partir de partições iniciais (base). Ela constitui a essência da combinação, pois diz como as partições são combinadas.

Vários aspectos tornam a definição de uma função consenso difícil (Topchy et al. 2003). A ausência de rótulos nos objetos a serem agrupados faz com que não haja uma correspondência explícita entre os *clusters* das diversas partições. Isso é agravado quando as partições possuem diferentes números de *clusters*, resultando em um problema computacional intratável (problema de correspondência de rótulos¹). De fato, o problema de combinação de agrupamentos é equivalente ao problema de encontrar uma partição mediana em relação às partições dadas, que é um problema comprovadamente NP-completo (Topchy et al. 2003). Assim, como o critério de agrupamento dos algoritmos convencionais, as funções consenso são heurísticas propostas para a resolução do problema formal de obtenção de uma partição consenso. Topchy et al. (2004) resume as funções consenso em:

Funções baseadas em co-associação: a similaridade entre dois objetos pode ser estimada pelo número de *clusters* compartilhados por eles em todas as partições base. As partições são representadas por uma matriz em que essa similaridade é utilizada para representar a força de co-associação entre os objetos. A partição consenso é obtida pela aplicação de um algoritmo de agrupamento qualquer, que seja baseado em similaridade, a essa matriz de co-associação (Kellam et al. 2001; Strehl and Ghosh 2002; Fred and Jain 2002; Fred and Jain 2003; Monti et al. 2003).

Alguns dos problemas dessa abordagem são: falta de uma metodologia para a definição do algoritmo de agrupamento a ser utilizado para a combinação e a baixa confiabilidade da estimativa dos valores de co-associação quando um pequeno número de partições é utilizado.

Funções baseadas em grafo/hipergrafo: neste caso, as partições base são representadas por um grafo ou por um hipergrafo, e a partição consenso é encontrada empregando uma técnica de particionamento de grafos ou hipergrafos (Strehl and Ghosh 2002; Fern and Brodley 2004). Esse tipo de função consenso é o mais apropriado para este trabalho, conforme será discutido na Seção 4.2.3, em que serão apresentadas detalhadamente as duas técnicas de *ensemble* a serem utilizadas.

Funções baseadas em informação mútua: a função consenso é formulada em termos da informação mútua entre os rótulos na partição consenso e os rótulos nas partições

¹Rótulo dado pelo algoritmo de agrupamento para identificar um *cluster*.

iniciais. Strehl and Ghosh (2002) definem a informação mútua normalizada média entre uma partição qualquer e um conjunto de partições iniciais. A partição consenso é dada pelo máximo dessa função, considerando o número de *clusters* desejado. Porém, a otimização dessa função é um problema difícil e esses autores utilizam heurísticas baseadas em co-associação e hipergrafo para resolver o problema.

Fred and Jain (2003) também definem formalmente uma função consenso baseada em informação mútua, mas resolvem o problema com uma heurística baseada em co-associação.

Topchy et al. (2003) definem uma função consenso baseada na informação mútua generalizada, que é equivalente à variância intra-*cluster* num espaço de rótulos dos *clusters* especialmente transformado. A função é então otimizada com o algoritmo *k*-médias.

Funções baseadas em votação: um mecanismo de votação é utilizado para atribuir os objetos aos *clusters* da partição consenso, dado que o problema de correspondência dos rótulos seja solucionado para todas as partições base. Entretanto, esse problema de correspondência dos rótulos é de difícil solução, sendo as vezes intratável. Porém, pode-se obter uma aproximação heurística de uma rotulação consistente. Todas as partições podem ser re-rotuladas com base em sua melhor concordância com uma partição referência, que pode ser uma das partições base ou um novo agrupamento do conjunto de dados. Esse procedimento é utilizado por Fred (2001) e Weingessel et al. (2003).

Frossyniotis et al. (2002) propõem a construção das partições juntamente com um processo de renumeração dos *clusters* seguido de votação. A partir disso, são estabelecidas relações de vizinhança entre os *clusters*. Essas informações são utilizadas para fundir os *clusters* mais próximos, resultando na partição final.

Além desses tipos de função consenso, Topchy et al. (2004) definem uma função consenso baseada em uma solução para o problema de probabilidade máxima para um modelo misto finito do conjunto de partições iniciais. Esse conjunto de partições é modelado como uma mistura de distribuições multinomiais no espaço dos rótulos dos *clusters*. O problema de probabilidade máxima pode ser resolvido com o algoritmo EM.

A Tabela 4.1 contém um resumo das principais características das abordagens citadas, utilizadas para *ensemble* de agrupamentos. Nessa tabela estão resumidos a forma de representação das partições base, a função consenso, o objetivo da combinação, os algoritmos empregados e a maneira como foram utilizados para gerar diversidade para as partições base.

Tabela 4.1: Comparação das formas de combinação de agrupamentos.

Artigo	Representação das partições	Função consenso	Objetivo da combinação	Algoritmos combinados	Diversidade
(Kellam et al. 2001)	Matriz de concordância	Os <i>clusters</i> finais são aqueles que possuem os mesmos objetos em todas as partições	<i>Clusters</i> robustos (<i>clusters</i> em que os objetos aparecem junto em todas as partições)	Hierárquico, <i>k</i> -médias, SOM e algoritmos genéticos, com o coeficiente de correlação de Pearson	Vários algoritmos
(Fred 2001)	Matriz de co-associação	Votação	Consistência	<i>k</i> -médias	Mesmo algoritmo com diferentes inicializações
(Fred and Jain 2002)	Matriz de co-associação	ligação simples com um novo critério para determinar a partição final	Consistência	<i>k</i> -médias	Mesmo algoritmo com diferentes inicializações
(Strehl and Ghosh 2002)	Hipergrafo	Particionamento de grafo de similaridade, particionamento de corte mínimo e meta <i>clusters</i>	Reuso de Conhecimento, Computação distribuída, Robustez	Particionamento de grafo e <i>k</i> -médias, com várias medidas de similaridade, SOM e particionamento de hipergrafo	Vários algoritmos e mesmo algoritmo com dados diferentes
(Frossymiotis et al. 2002)	Tabela de votação e tabela de relação de vizinhança	Votação	Robustez e estabilidade	<i>k</i> -médias e <i>greedy</i> -EM	Mesmo algoritmo com diferentes inicializações
(Monti et al. 2003)	Matriz consenso	LM determinando <i>k</i> com base na estabilidade dos agrupamentos	Estabilidade	LM e SOM	Mesmo algoritmo com dados diferentes
(Fred and Jain 2003)	Matriz de co-associação, usando votação	ligação simples (pode ser qualquer função baseada em similaridade)	Consistência, estabilidade e robustez	<i>k</i> -médias	Mesmo algoritmo com diferentes inicializações
(Weingessel et al. 2003)	Conjunto de matrizes de pertinência das partições iniciais	Votação/fusão	Robustez	<i>k</i> -médias, <i>hard competitive learning</i> e aprendizado competitivo <i>fuzzy</i> não supervisionado	Vários algoritmos e mesmo algoritmo com diferentes inicializações
(Topchy et al. 2003)	Novo conjunto de características dos padrões	Baseada no <i>k</i> -médias aplicado no novo espaço de características	Desempenho e custo	Algoritmos fracos que usam projeções ou divisões aleatórias dos dados	Mesmo algoritmo com diferentes inicializações
(Topchy et al. 2004)	Novo conjunto de características dos objetos	Probabilidade máxima encontrada com o método EM	Robustez, estabilidade, escalabilidade e reuso do conhecimento	<i>k</i> -médias	Mesmo algoritmo com diferentes inicializações
(Fern and Brodley 2004)	Grafo	Particionamento de grafo	Robustez	<i>k</i> -médias	Mesmo algoritmo com dados diferentes
(Law et al. 2004)	Conjunto com todos os <i>clusters</i>	<i>Clusters</i> mais estáveis	Novidade e robustez	<i>k</i> -médias, EM, hierárquico com ligação simples e <i>spectral clustering</i>	Vários algoritmos

4.2.3 Técnicas de Interesse

Topchy et al. (2005) sugerem que as abordagens para construção de *ensembles* de agrupamentos baseadas em grafos podem ser mais robustas que as técnicas baseadas em matriz de co-associação. Segundo Kuncheva et al. (2002), uma das técnicas de *ensembles* de agrupamentos mais populares é o algoritmo MCLA (*Meta-CLustering Algorithm*), proposta por Strehl and Ghosh (2002). Por isso, esse algoritmo foi escolhido para ser investigado neste trabalho. Foi escolhida também uma outra abordagem para *ensemble* baseada em particionamento de grafos, o algoritmo HBGF (*Hybrid Bipartite Graph Formulation*), proposto por Fern and Brodley (2004). Nas duas abordagens, não são necessários os atributos originais dos objetos, apenas os rótulos dos *clusters* de cada objeto nas partições a serem combinadas. Essas duas abordagens são detalhadas a seguir.

Ensemble de Strehl e Ghosh

Strehl and Ghosh (2002) definem formalmente a combinação de algoritmos de agrupamento como um problema de otimização de uma função consenso baseada na informação mútua compartilhada entre as soluções individuais. Entretanto, como a otimização dessa função é um problema combinatorial difícil, eles propõem três algoritmos de combinação baseados em heurísticas: CSPA (*Cluster-based Similarity Partitioning Algorithm*), HGPA (*HiperGraph-Partitioning Algorithm*) e MCLA (*Meta-CLustering Algorithm*). Como esses algoritmos têm um custo computacional baixo, Strehl and Ghosh (2002) aplicam os três algoritmos, cada um gerando uma partição consenso, e utilizam uma função supra-consenso baseada na informação mútua compartilhada para escolher qual delas será a partição consenso final, ou seja, a partição consenso final será aquela, dentre as três, que tem a melhor informação mútua compartilhada.

A definição formal do problema como a otimização de uma função consenso baseada na informação mútua compartilhada entre as soluções individuais é feita da seguinte maneira. Seja a informação mútua normalizada (*NMI*) estimada entre duas partições π^a e π^b , dada pela Equação 4.1, em que $|\cdot|$ indica o número de objetos de um conjunto, K^a e K^b são os números de *clusters* das partições π^a e π^b , respectivamente, c_h^a é o h -ésimo *cluster* de π^a , c_l^b é o l -ésimo *cluster* de π^b e n é o número de objetos do conjunto de dados (Strehl and Ghosh 2002).

$$\phi^{(NMI)}(\pi^a, \pi^b) = \frac{\sum_{h=1}^{K^a} \sum_{l=1}^{K^b} |c_h^a \cap c_l^b| \log\left(\frac{n |c_h^a \cap c_l^b|}{|c_h^a| |c_l^b|}\right)}{\sqrt{\left(\sum_{h=1}^{K^a} |c_h^a| \log\left(\frac{|c_h^a|}{n}\right)\right) \left(\sum_{l=1}^{K^b} |c_l^b| \log\left(\frac{|c_l^b|}{n}\right)\right)}} \quad (4.1)$$

Com base nessa medida de informação mútua entre duas partições, Strehl and Ghosh

(2002) definem uma medida de informação mútua entre uma partição π^i e um conjunto Π de r partições, como a informação mútua normalizada média (*ANMI*), dada pela Equação 4.2.

$$\phi^{(ANMI)}(\pi^i, \Pi) = \frac{1}{r} \sum_{q=1}^r \phi^{(NMI)}(\pi^i, \pi^q) \quad (4.2)$$

$\phi^{(ANMI)}$ é a função objetivo, e a partição consenso $\pi^{F(k-opt)}$ é aquela com informação mútua normalizada média ($\phi^{(ANMI)}$) máxima, em relação às partições individuais em Π , dado que o número de *clusters* desejado para a partição consenso é k . $\pi^{F(k-opt)}$ é dada pela Equação 4.3, em que π^i corresponde a todas as possíveis partições com k *clusters*.

$$\pi^F = \arg \max_{\pi^i} \sum_{q=1}^r \phi^{(NMI)}(\pi^i, \pi^q) \quad (4.3)$$

Como já mencionado, a $\phi^{(ANMI)}$ não é otimizada. Os três algoritmos baseados em heurísticas são executados e a $\phi^{(ANMI)}$ é usada para selecionar a melhor partição dentre as geradas com eles.

Os três algoritmos baseados em heurísticas propostos para encontrar uma partição consenso partem de uma representação inicial das partições na forma de um hipergrafo. No CSPA, o problema não depende dessa representação, mas ela é utilizada para calcular facilmente uma matriz de similaridade que servirá como entrada para um algoritmo de agrupamento. No HGPA o hipergrafo é empregado diretamente. Já no MCLA, ele é utilizado para a construção de um meta-grafo a ser particionado e posterior determinação da partição consenso, como descrito mais adiante.

O hipergrafo utilizado é representado por uma matriz de adjacências. Cada coluna da matriz é uma hiperaresta, que representa um *cluster*. Essa matriz é construída pela concatenação das matrizes binárias de pertinência de cada partição. As linhas da matriz de pertinência de uma partição correspondem aos objetos e as colunas correspondem aos seus *clusters*. Cada célula da matriz contém o valor 1 se o objeto pertence ao *cluster* e 0 caso contrário.

O algoritmo CSPA utiliza a heurística mais simples, porém possui complexidade quadrática no número de objetos. Esse algoritmo está classificado entre as funções baseadas em co-associação. Como já mencionado, esse algoritmo se baseia na construção de uma nova matriz de similaridade a partir das partições originais. As entradas dessa matriz denotam a fração das partições nas quais dois objetos pertencem ao mesmo *cluster*. A matriz de similaridade gerada é então utilizada para re-agrupar os objetos com um algoritmo de agrupamento qualquer, que seja baseado em similaridade. Strehl and Ghosh

(2002) utilizam o algoritmo METIS²(Karypis and Kumar 1999) para particionar o grafo de similaridade induzido.

No algoritmo HGPA, a combinação é tratada como um problema de particionamento de um hipergrafo definido apropriadamente, no qual as hiperarestas representam *clusters*. Esse particionamento é feito cortando um número mínimo de hiperarestas. Para isso é utilizado o pacote de particionamento de hipergrafos HMETIS³.

O algoritmo MCLA trata a combinação como um problema de correspondência dos *clusters* das partições iniciais. Ele se baseia no agrupamento desses *clusters*. Um meta-grafo em que cada vértice corresponde a um *cluster* é construído. Em seguida, o meta-grafo é particionado de maneira que os *clusters* que permaneceram em um mesmo grupo (meta-*cluster*) são correspondentes. Os objetos são então atribuídos aos meta-*clusters* com os quais eles estão mais fortemente associados.

Em experimentos controlados realizados por Strehl and Ghosh (2002) para comparar os três algoritmos, foi observado que o MCLA apresentou melhor resultado na presença de uma quantidade média para alta de ruído, o que ocorre com frequência em problemas reais. Em relação à complexidade, o algoritmo MCLA também é o mais vantajoso. Em outros experimentos, Strehl and Ghosh (2002) observaram que cada algoritmo tem um desempenho melhor para uma situação diferente. O algoritmo MCLA é melhor quando há menos diversidade nas partições iniciais, o que está de acordo com a suposição inicial do MCLA de que existe uma correspondência entre os *clusters* das partições a serem combinadas. Assim, o MCLA será um dos algoritmos empregados neste trabalho. Os principais passos do algoritmo MCLA são:

- Construir um meta-grafo: cada *cluster* das partições iniciais (as hiperarestas do hipergrafo descrito) é considerado um vértice de um outro grafo regular não direcionado, o meta-grafo. O peso da aresta entre os vértices correspondentes aos *clusters* c_j^i e c_t^s é dado pela razão da intersecção pela união dos objetos pertencentes aos dois *clusters*, conforme a Equação 4.4.

$$w(c_j^i, c_t^s) = \frac{|c_j^i \cap c_t^s|}{|c_j^i \cup c_t^s|} \quad (4.4)$$

- Agrupar as hiperarestas (*clusters*): o agrupamento dos *clusters* é feito pelo particionamento do meta-grafo em k meta-*clusters* balanceados, sendo k definido pelo usuário. Para isso, o pacote de particionamento de grafos METIS⁴(Karypis and Kumar 1999) é utilizado. Essa fase permite encontrar os *clusters* das partições

²<http://glaros.dtc.umn.edu/gkhome/metis/metis/overview>

³<http://glaros.dtc.umn.edu/gkhome/metis/hmetis/overview>

⁴<http://glaros.dtc.umn.edu/gkhome/metis/metis/overview>

iniciais que são correspondentes. Cada meta-*cluster* resultante do particionamento representa um grupo de *clusters* correspondentes.

- Unir os *clusters* de cada meta-*cluster*: para cada meta-*cluster*, transforma as hiperarestas em uma única meta-hiperaresta. Cada objeto pode pertencer a mais de um meta-*cluster*. Assim, para cada meta-hiperaresta, é calculado um vetor de associação descrevendo o nível de associação de cada objeto com o meta-*cluster*. O vetor de associação é obtido calculando-se a média dos vetores que representam as hiperarestas de um meta-*cluster*. Em outras palavras, o nível de associação de um objeto a um meta-*cluster* é dado pela média do número de *clusters* desse meta-*cluster*, que contêm o objeto.
- Determinar o meta-*cluster* final de cada objeto: cada objeto é associado ao meta-*cluster* para o qual ele possui o maior valor de associação. Desempates são decididos aleatoriamente. A partição dos objetos indicada pelos meta-*clusters* é a partição final π^F resultante do *ensemble*. Deve-se notar que não é garantido que todo meta-*cluster* tenha pelo menos um objeto. Assim, a partição π^F tem no máximo (e não exatamente) k *clusters*.

Quanto à diversidade das partições iniciais, dependendo do cenário de aplicação considerado, foi definida uma alternativa diferente para gerar as partições iniciais. No primeiro cenário as partições originais foram formadas com a aplicação de um único algoritmo de agrupamento a diferentes subconjuntos de atributos dos dados. No segundo cenário, as partições foram obtidas também com um único algoritmo, porém aplicado a diferentes subconjuntos de objetos, considerando sempre todos os atributos. Finalmente, no terceiro cenário, as partições iniciais foram geradas a partir da execução de diferentes algoritmos, empregando diferentes medidas de proximidade, ao mesmo conjunto de dados.

Para ilustrar o funcionamento do algoritmo MCLA, considere os agrupamentos mostrados na Tabela 4.2 (Strehl and Ghosh 2002). O hipergrafo que representa esses agrupamentos pode ser observado na matriz de adjacências da Tabela 4.3. Nessa tabela, cada objeto $\mathbf{x}_i$ corresponde a um vértice e cada hiperaresta h_j representa um dos *clusters* de um dos agrupamentos. Com essas informações é construído o meta-grafo em que cada hiperaresta é um vértice e cujos pesos das arestas entre os vértices podem ser observados na Tabela 4.4. O particionamento desse grafo em 3 partes resulta nos meta-*clusters* $c_1^M = \{h_3, h_4, h_9\}$, $c_2^M = \{h_2, h_6, h_8, h_{10}\}$ e $c_3^M = \{h_1, h_5, h_7, h_{11}\}$. Assim, os *clusters* representados por h_3 , h_4 e h_9 , por exemplo, são correspondentes.

Em seguida, os *clusters* de cada meta-*cluster* são unidos, formando as meta-hiperarestas h_i^M representadas na Tabela 4.5. Nessa tabela também estão representados os respectivos vetores de associação $a(h_i^M)$. Com base nessas informações, o meta-*cluster* final de cada

objeto é determinado. Assim, $\mathbf{x}_1$, por exemplo, vai pertencer ao meta-*cluster* c_3^M , pois é o meta-*cluster* com o qual $\mathbf{x}_1$ tem o maior valor de associação (0,75). A partição consenso π^F fica composta pelos *clusters* $c_1^F = \{\mathbf{x}_6, \mathbf{x}_7\}$, $c_2^F = \{\mathbf{x}_4, \mathbf{x}_5\}$ e $c_3^F = \{\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3\}$.

Tabela 4.2: Exemplo do MCLA - partições

Partição	Clusters
π^1	$c_1^1 = \{\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3\}$, $c_2^1 = \{\mathbf{x}_4, \mathbf{x}_5\}$, $c_3^1 = \{\mathbf{x}_6, \mathbf{x}_7\}$
π^2	$c_1^2 = \{\mathbf{x}_6, \mathbf{x}_7\}$, $c_2^2 = \{\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3\}$, $c_3^2 = \{\mathbf{x}_4, \mathbf{x}_5\}$
π^3	$c_1^3 = \{\mathbf{x}_1, \mathbf{x}_2\}$, $c_2^3 = \{\mathbf{x}_3, \mathbf{x}_4\}$, $c_3^3 = \{\mathbf{x}_5, \mathbf{x}_6, \mathbf{x}_7\}$
π^4	$c_1^4 = \{\mathbf{x}_1, \mathbf{x}_4\}$, $c_2^4 = \{\mathbf{x}_2, \mathbf{x}_5\}$, objetos $\mathbf{x}_3$, $\mathbf{x}_6$ e $\mathbf{x}_7$ não agrupados

Tabela 4.3: Exemplo do MCLA - hipergrafo

Vértices	h_1	h_2	h_3	h_4	h_5	h_6	h_7	h_8	h_9	h_{10}	h_{11}
	(c_1^1)	(c_2^1)	(c_3^1)	(c_1^2)	(c_2^2)	(c_3^2)	(c_1^3)	(c_2^3)	(c_3^3)	(c_1^4)	(c_2^4)
$\mathbf{x}_1$	1	0	0	0	1	0	1	0	0	1	0
$\mathbf{x}_2$	1	0	0	0	1	0	1	0	0	0	1
$\mathbf{x}_3$	1	0	0	0	1	0	0	1	0	0	0
$\mathbf{x}_4$	0	1	0	0	0	1	0	1	0	1	0
$\mathbf{x}_5$	0	1	0	0	0	1	0	0	1	0	1
$\mathbf{x}_6$	0	0	1	1	0	0	0	0	1	0	0
$\mathbf{x}_7$	0	0	1	1	0	0	0	0	1	0	0

Tabela 4.4: Exemplo do MCLA - pesos

Vértices	h_1	h_2	h_3	h_4	h_5	h_6	h_7	h_8	h_9	h_{10}	h_{11}
h_1	-	-	-	0	0,50	0	0,40	0,20	0	0,20	0,20
h_2	-	-	-	0	0	0,50	0	0,25	0,20	0,25	0,25
h_3	-	-	-	0,50	0	0,00	0	0	0,40	0	0
h_4	0	0	0,50	-	-	-	0	0	0,40	0	0
h_5	0,50	0	0	-	-	-	0,40	0,20	0	0,20	0,20
h_6	0	0,50	0	-	-	-	0	0,25	0,20	0,25	0,25
h_7	0,40	0	0	0	0,40	0	-	-	-	0,25	0,25
h_8	0,20	0,25	0	0	0,20	0,25	-	-	-	0,25	0
h_9	0	0,20	0,40	0,40	0	0,20	-	-	-	0	0,20
h_{10}	0,20	0,25	0	0	0,20	0,25	0,25	0,25	0	-	-
h_{11}	0,20	0,25	0	0	0,20	0,25	0,25	0	0,20	-	-

Tabela 4.5: Exemplo do MCLA - meta-hiperarestas e vetores de associação

Vértices	$c_1^M = \{h_3, h_4, h_9\}$		$c_2^M = \{h_2, h_6, h_8, h_{10}\}$		$c_3^M = \{h_1, h_5, h_7, h_{11}\}$	
	h_1^M	$a(h_1^M)$	h_2^M	$a(h_2^M)$	h_3^M	$a(h_3^M)$
$\mathbf{x}_1$	0	0	1	0,25	1	0,75
$\mathbf{x}_2$	0	0	0	0	1	1
$\mathbf{x}_3$	0	0	1	0,25	1	0,5
$\mathbf{x}_4$	0	0	1	1	0	0
$\mathbf{x}_5$	1	0,33	1	0,5	1	0,25
$\mathbf{x}_6$	1	1	0	0	0	0
$\mathbf{x}_7$	1	1	0	0	0	0

Ensemble de Fern e Brodley

A abordagem de Fern and Brodley (2004) utiliza particionamento de um grafo bipartido para encontrar uma partição consenso, dado um conjunto de partições base. Nessa abordagem, os autores constroem um grafo bipartido a partir do conjunto de partições a serem combinadas, modelando tanto objetos quanto *clusters* simultaneamente como vértices do grafo e, posteriormente, particionando o grafo com uma técnica tradicional de

particionamento de grafos. O algoritmo de Fern and Brodley (2004), chamado HBGA (*Hybrid Bipartite Graph Formulation*), pode ser detalhado da seguinte maneira:

- Construir o grafo: a partir de um conjunto de partições base, um grafo $G = (V, W)$ é construído da seguinte maneira:
 - $V = V^C \cup V^O$, em que os vértices em V^C representam os *clusters* do conjunto de partições base Π , e os vértices em V^O representam os objetos do conjunto de dados X .
 - O peso entre dois vértices i e j é definido da seguinte maneira. Se $i, j \in V^C$ ou $i, j \in V^O$ (ambos representam *clusters* ou ambos representam vértices), então $w(i, j) = 0$, caso contrário, se o objeto $\mathbf{x}_i$ pertence ao *cluster* c_j , $w(i, j) = 1$, senão, $w(i, j) = 0$.
- Particionar o grafo: qualquer técnica de particionamento de grafos tradicional pode ser empregada nesse passo. Fern and Brodley (2004) utilizam duas técnicas distintas: *Spectral Graph Partitioning* (Ng et al. 2002) e METIS (Karypis and Kumar 1999). O grafo deve ser particionado em k *clusters*, em que k é escolhido pelo usuário.
- A partição consenso π^F é dada pela divisão dos objetos resultante do particionamento do grafo.

Para gerar as partições base, Fern and Brodley (2004) consideraram duas abordagens. A primeira consiste em executar um algoritmo em diferentes subconjuntos dos dados (reamostragem). A segunda consiste da aplicação de um único algoritmo aos dados representados em projeções do espaço de atributos de menor dimensão que o original.

Nos experimentos apresentados em (Fern and Brodley 2004), os autores observaram que sua abordagem, HGBF, apresentou um desempenho equivalente ou superior àqueles obtidos pelas abordagens de Strehl and Ghosh (2002).

Para ilustrar o funcionamento do algoritmo HBGF, considere os agrupamentos mostrados na Tabela 4.6 (Fern and Brodley 2004). Primeiramente é construído o grafo bipartido mostrado na Figura 4.1. Os vértices de V^C , representados por um losango, correspondem aos *clusters* de π^1 e π^2 e os vértices de V^O , representados por um círculo, correspondem aos objetos. Todas as arestas representadas têm peso 1 e ligam um objeto a um *cluster*, indicando que o objeto pertence àquele *cluster*. A linha tracejada mostra uma partição desse grafo em duas partes, obtida com algum algoritmo de particionamento de grafos tradicional. A divisão dos objetos resultante desse particionamento é a partição consenso π^F , composta pelos *clusters* $c_1^F = \{\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3, \mathbf{x}_4, \mathbf{x}_5\}$ e $c_2^F = \{\mathbf{x}_6, \mathbf{x}_7, \mathbf{x}_8, \mathbf{x}_9\}$.

Tabela 4.6: Exemplo do HBGF - partições

Partição	Clusters
π^1	$c_1^1 = \{\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3, \mathbf{x}_4\}$, $c_2^1 = \{\mathbf{x}_5, \mathbf{x}_6, \mathbf{x}_7, \mathbf{x}_8, \mathbf{x}_9\}$
π^2	$c_1^2 = \{\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3, \mathbf{x}_4, \mathbf{x}_5, \mathbf{x}_6\}$, $c_2^2 = \{\mathbf{x}_7, \mathbf{x}_8, \mathbf{x}_9\}$

Figura 4.1: Exemplo do HBGF - grafo bipartido

4.3 Agrupamento Multi-objetivo

A idéia básica do agrupamento multi-objetivo é otimizar simultaneamente dois ou mais critérios de agrupamento que sejam complementares (Handl and Knowles 2004; Handl and Knowles 2005a; Handl and Knowles 2005b; Handl and Knowles 2006a). Com o objetivo de demonstrar sua abordagem, Handl e Knowles apresentam um algoritmo evolucionário multi-objetivo, chamado MOCK (*Multi-Objective Clustering with automatic K-determination*) (Handl and Knowles 2004; Handl and Knowles 2005a; Handl and Knowles 2005b; Handl and Knowles 2006a).

O algoritmo MOCK é capaz de otimizar simultaneamente dois objetivos complementares e identificar automaticamente as melhores partições do fronte de Pareto, determinando automaticamente o número de *clusters*. Com base na forma do fronte de Pareto, MOCK retorna não apenas um conjunto de partições com diferentes compromissos em um intervalo de números de *clusters*, mas também uma indicação de quais dessas partições são as melhores. MOCK tenta encontrar o fronte de Pareto mais completo possível para, posteriormente reduzir esse conjunto a uma única solução, por meio de um *attainment score*.

Em (Handl and Knowles 2004; Handl and Knowles 2005a), MOCK é comparado com três algoritmos de agrupamento convencionais (k -médias e algoritmos hierárquicos com ligação simples e média) e também com o *ensemble* de agrupamentos proposto por Strehl and Ghosh (2002), utilizando os três algoritmos e a função supra-consenso citados anteriormente. Os autores mostram que a sua abordagem é mais robusta em relação à variedade de estruturas encontradas em conjuntos de dados diferentes e que é capaz de encontrar certas estruturas que outros métodos não conseguem.

O MOCK é baseado no algoritmo evolucionário multi-objetivo PESA-II (Corne et al. 2001), descrito no Capítulo 2.

A representação dos indivíduos usada no MOCK é a representação de adjacência baseada em locus (*locus-based adjacency representation*) (Park and Song 1998). Nessa representação, cada indivíduo g consiste de n (número de objetos do conjunto de dados) genes, $g_1, g_2, \dots, g_n$. Cada gene g_i pode assumir um valor j no intervalo $[1, n]$, significando que existe uma ligação entre os objetos $\mathbf{x}_i$ e $\mathbf{x}_j$, ou seja, os objetos $\mathbf{x}_i$ e $\mathbf{x}_j$ estão no mesmo *cluster*. A decodificação dessa representação requer a identificação de todos os sub-grafos. Todos os objetos pertencentes ao mesmo sub-grafo são associados ao mesmo *cluster*.

Como operador de recombinação, é utilizado cruzamento uniforme. Para mutação, os autores originalmente empregavam um operador especializado que reduz significativamente o tamanho do espaço de busca (Handl and Knowles 2004), a mutação dos vizinhos mais próximos. Nessa mutação, cada objeto apenas pode ter sua ligação alterada para um dos seus v vizinhos mais próximos. Consequentemente, $g_i \in \{nn_{i1}, \dots, nn_{iv}\}$, em que nn_{il} é o l -ésimo vizinho mais próximo do objeto $\mathbf{x}_i$. Nesse caso, todos os genes têm uma probabilidade de mutação igual ($\frac{1}{n}$). Posteriormente, os autores propuseram uma modificação na mutação que altera a probabilidade de mutação de ligações individuais, $i \rightarrow j$, para $p_m = \frac{1}{n} + (\frac{l}{n})^2$, em que $j = nn_{il}$ (Handl and Knowles 2005b).

O procedimento para inicialização da população originalmente usa árvore geradora mínima (MST, do inglês *Minimum Spanning Tree*) para gerar os indivíduos (Handl and Knowles 2004). Primeiramente, é gerada uma MST completa usando o algoritmo de Prim (Wilson and Watkins 1990). O i -ésimo indivíduo da população inicial é inicializado pela MST com as $(i - 1)$ -ésimas ligações mais longas removidas. Porém, esse procedimento tende a gerar soluções boas na região do fronte de Pareto em que a conectividade é baixa e, quando os *clusters* não são bem separados, gerar soluções muito parecidas (Handl and Knowles 2005b). Para melhorar o espalhamento das soluções iniciais, Handl and Knowles (2005b) propõem uma nova inicialização baseada em uma mistura de soluções geradas com o algoritmo k -médias e com MST.

Para as soluções baseadas na MST, inicialmente é construída uma MST e são identificadas todas as suas b ligações interessantes. Uma ligação $i \rightarrow j$ é considerada interessante

se e somente se $i = nn_{jl} \wedge j = nn_{ik}$, com $l > v$ e $k > v$, em que v , o número de vizinhos mais próximos, é dado pelo usuário. O grau de interessabilidade é $gi = \min(l, k)$. As b ligações interessantes são ordenadas pelo seu grau de interessabilidade. O conjunto de partições baseadas na MST é construído da seguinte maneira: para cada $g \in [0, \min(b, 0, 5n^I)]$, em que n^I é o tamanho da população inicial, é gerado um agrupamento π^g removendo as g primeiras ligações interessantes. As ligações perdidas são substituídas por uma ligação com um vizinho j escolhido aleatoriamente, com $j = nn_{il} \wedge l \leq v$. Para as soluções baseadas no k -médias, o algoritmo é executado, gerando partições com números de *clusters* $k \in [2, n^I - (\min(b, 0, 5n^I) + 1)]$. As partições obtidas dessa maneira são convertidas para a representação apropriada.

Como funções objetivo, MOCK emprega dois objetivos complementares, a variância intra-*cluster* (*var*) (Equação 3.6) e a conectividade (*con*) (Equação 3.7), ambos definidos na Seção 3.4. Essas medidas foram escolhidas por representar dois aspectos fundamentalmente diferentes de qualidade de um agrupamento. Esses objetivos contrabalançam suas tendências de aumentar ou diminuir com o número de *clusters*. Isso é importante para explorar bem o espaço de soluções, evitando a convergência para soluções triviais (n *clusters* com um único objeto, no caso da variância intra-*cluster*, e apenas um *cluster* com n elementos, no caso da conectividade) (Handl and Knowles 2004).

A aplicação desse algoritmo gera um conjunto de soluções não dominadas com diferentes compromissos dos dois objetivos e com diferentes números de *clusters*. Para encontrar a melhor solução, Handl e Knowles geram o fronte de Pareto mais completo possível e, posteriormente, fazendo uso de várias considerações específicas do domínio, reduzem o conjunto de soluções à uma única solução (Handl and Knowles 2004; Handl and Knowles 2005a).

O procedimento para selecionar a melhor solução é baseado na intuição de que a estrutura dos dados está refletida na forma do fronte de Pareto (Handl and Knowles 2004; Handl and Knowles 2005a). Das tendências observadas nos objetivos empregados, é possível afirmar que incrementando o número de *clusters* k , obtém-se uma melhora em *var*, δV , ao custo de uma degradação em *con*, δC . Para um número de *clusters* k menor do que o verdadeiro, espera-se que a razão $R = \delta V / \delta C$ seja grande, pois a separação de dois *clusters* causa uma grande diminuição em *var*, com pouco ou nenhum aumento em *con*. Para os números de *clusters* maiores do que o verdadeiro, essa razão se torna menor, pois a diminuição em *var* é menor, mas ao preço de um aumento maior da *con* (um *cluster* verdadeiro está sendo dividido). Pela tendência das medidas, as soluções no fronte de Pareto estão aproximadamente ordenadas por k .

Assim, a mudança evidente em R que ocorre no número correto de *clusters* pode ser observada como um ponto de inflexão no gráfico do fronte de Pareto. Para determinar

corretamente esse ponto, os autores utilizam distribuições aleatórias de dados como referência (Handl and Knowles 2004; Handl and Knowles 2005a; Handl and Knowles 2005b). Esses dados são agrupados com o MOCK, gerando um conjunto de fronte de referência. O fronte solução é normalizado e, para cada ponto nesse fronte é calculado um *attainment score*, dado pela sua distância até as *attainment surfaces* dos fronte de referência. Em seguida, é traçado um gráfico dos *attainment scores* em função de k . A solução correspondente ao máximo dessa curva é selecionada como a melhor solução.

Além de uma melhor solução, também é possível identificar outros possíveis máximos locais, que podem revelar estruturas em outros níveis. Além disso, o *attainment score* também serve como uma estimativa da qualidade de cada uma das soluções individuais.

4.4 Considerações Finais

Neste capítulo foram introduzidas as técnicas de *ensemble* de agrupamentos e agrupamento multi-objetivo que visam superar algumas limitações da análise de agrupamento tradicionais. Entretanto, essas técnicas também têm suas próprias limitações, que serão discutidas na Seção 5.2 do Capítulo 5, juntamente com as dificuldades gerais da análise de agrupamento. A partir da discussão dessas dificuldades, serão detalhadas as metas a serem atingidas nesta Tese e apresentado um *framework* para análise de agrupamento que visa superar as dificuldades, atingindo as metas estabelecidas. Tal *framework* integra as idéias dos *ensembles* e do agrupamento multi-objetivo, fazendo com que as vantagens de uma abordagem auxiliem na superação das limitações da outra.

Neste capítulo também foram discutidas em maiores detalhes duas técnicas de *ensemble* de agrupamentos que serão utilizadas na implementação do *framework* proposto, como será visto na Seção 5.5.

No Capítulo 5, a abordagem proposta é detalhada, sendo discutidas suas principais diferenças e vantagens em relação às técnicas descritas na literatura e apresentadas neste capítulo.

Abordagem Proposta

5.1 *Considerações Iniciais*

Nos Capítulos 2, 3 e 4 foi feita uma revisão da literatura existente sobre os temas relevantes para este trabalho. Especificamente, nos Capítulos 3 e 4, foram detalhados os aspectos importantes da análise de agrupamento, fundamentais para o desenvolvimento do *framework* proposto nesta Tese, que são: as diferentes definições de *cluster* e conseqüente variedade de tipos de critérios de agrupamento, as medidas de proximidade, os algoritmos de agrupamento de interesse para o trabalho, a validação de agrupamento, com destaque para as medidas a serem usadas, o agrupamento multi-objetivo e as técnicas de *ensembles* de agrupamentos.

Este capítulo contém uma descrição detalhada do *framework* proposto, bem como das técnicas e medidas específicas, utilizadas para ilustrar a proposta. Inicialmente, na Seção 5.2 são revistos os principais problemas da análise de agrupamento que motivaram este trabalho. Com base nesses problemas, na Seção 5.3 são ressaltadas as metas a serem atingidas. Em seguida, na Seção 5.4, é detalhada a proposta propriamente dita, destacando como cada um de seus aspectos contribui para atingir as metas destacadas. Na Seção 5.5 são apresentadas as alternativas implementadas para ilustrar o *framework* proposto.

5.2 *Descrição do Problema*

Conforme apresentados na Seção 1.2, os principais problemas da análise de agrupamento estão relacionados a:

- escolha do algoritmo,

- dados com estrutura heterogênea,
- existência de mais de uma estrutura e
- necessidade de conhecimento prévio.

Escolha do algoritmo: a escolha do algoritmo de agrupamento mais apropriado para um conjunto de dados é uma tarefa complexa, uma vez que cada algoritmo favorece a descoberta de um tipo de *cluster* diferente e, em geral, não se conhece previamente que tipo de *cluster* melhor representa os dados. Como apresentado na Seção 3.3, algoritmos baseados no conceito de compactação, como o KM e o LM, encontram *clusters* de formas esféricas. Já os baseados no conceito de encadeamento, como o LS e o SNN, não têm limitações quanto à forma, porém falham para os casos em que há pouca separação espacial entre os *clusters*.

Para ilustrar essa questão, considere os conjuntos de dados da Figura 5.1. O conjunto de dados 5.1(a) possui dois *clusters* globulares bastante próximos um do outro. Esses *clusters* são facilmente identificados pelos algoritmos KM e LM, mas não são distinguidos pelos algoritmos LS e SNN. Já a estrutura do conjunto 5.1(b), com dois *clusters* no formato de anel, é identificada pelos algoritmos LS e SNN, mas não por KM e LM.

Figura 5.1: Conjuntos de dados com estrutura homogênea

Dados com *clusters* de tipos heterogêneos: os algoritmos de agrupamento, em geral, assumem um critério homogêneo em todo o espaço de atributos (Law et al. 2004). Entretanto, os dados podem ter conformações em que cada região do espaço apresenta um tipo de *cluster* diferente. Um exemplo simples é o conjunto de dados da Figura 5.2. Nesse caso, estão presentes dois *clusters* de formato globular bastante próximos e um *cluster* em formato de anel.

Figura 5.2: Conjunto de dados com estrutura heterogênea

Na Figura 5.3 podem ser observados os resultados da aplicação dos algoritmos KM, LM, LS e SNN a esse conjunto de dados. Nenhum dos algoritmos foi capaz de encontrar os três *clusters* desse conjunto de dados, mas, em geral, cada um identificou bem os *clusters* que estão de acordo com seu critério, com exceção do LM. O algoritmo KM encontrou os dois *clusters* globulares (Figura 5.3(a)), mas não conseguiu obter o *cluster* em anel. No caso do LM, a presença do anel atrapalhou a obtenção dos dois *clusters* globulares (Figura 5.3(b)). Os algoritmos LS e SNN separaram bem o *cluster* em forma de anel, mas uniram os dois *clusters* globulares em um único grupo (Figuras 5.3(c) e 5.3(d)).

Existência de mais de uma estrutura: um conjunto de dados pode ter várias estruturas, tanto em diferentes níveis de refinamento, quanto em relação a diferentes critérios. Porém, a análise de agrupamento tradicional, bem como as técnicas de *ensemble*, buscam por uma única estrutura: a que melhor se ajusta aos dados. Por exemplo, considere o conjunto de dados da Figura 5.4. A estrutura que mais se destaca, e a mais facilmente obtida com os algoritmos de agrupamento e técnicas de validação, é a estrutura E1, com dois *clusters*. Os dois *clusters* nesse nível podem facilmente ser identificados por qualquer dos algoritmos mencionados, pois possuem formato aproximadamente esférico e estão muito bem separados um do outro. Porém, existem outros dois níveis de refinamento que apresentam estruturas claramente reconhecíveis: um com uma estrutura de cinco *clusters* (estrutura E2) e outro com uma estrutura de 13 *clusters* (estrutura E3). Pode-se observar que os *clusters* nesses níveis são bastante heterogêneos. Na estrutura E2, há um *cluster* em formato de sorriso, um *cluster* alongado e três *clusters* aproximadamente globulares. Já na estrutura E3, o mesmo *cluster* em forma de sorriso aparece, mas o *cluster* alongado da estrutura E2 pode ser visto como três *clusters* esféricos. Além disso, cada um dos *clusters* globulares de E2 pode ser visto como três *clusters* alon-

Figura 5.3: Resultados dos algoritmos no conjunto de dados heterôgeneo

gados em E3. A aplicação da análise de agrupamento tradicional a esses dados, ou seja, a utilização de vários algoritmos e vários parâmetros e posterior seleção da melhor estrutura, levaria à identificação da estrutura E1, fazendo com que as demais estruturas não fossem identificadas.

Figura 5.4: Conjunto de dados com várias estruturas

Necessidade de conhecimento prévio: como mencionado no Capítulo 1, a análise exploratória de um conjunto de dados utilizando agrupamento requer a aplicação de diversos algoritmos, com variadas configurações para seus parâmetros, e posterior utilização de uma ou mais medidas de validação para selecionar a partição de melhor qualidade. Porém, a escolha das medidas de validação também é uma tarefa difícil, pois, em geral, cada medida tende a favorecer um determinado algoritmo. A realização de todo esse processo requer um conhecimento bastante especializado em análise de agrupamento. Suposições sobre a distribuição dos dados, em geral, também precisam ser feitas. Tais conhecimentos são necessários na escolha do(s) algoritmo(s), no ajuste de seus parâmetros e na(s) escolha(s) da(s) medida(s) de validação. A necessidade de conhecimento sobre agrupamento limita a profundidade da análise realizada pelos especialistas do domínio dos dados, que raramente possuem um conhecimento profundo em análise de agrupamento.

Na tentativa de superar alguns dos problemas anteriores, surgiram as abordagens de *ensembles* de agrupamentos e agrupamento multi-objetivo. Os *ensembles* de agrupamentos (em especial os heterogêneos) ajudam na robustez quanto a diferentes propriedades nos dados, evitando a necessidade de escolha de um algoritmo que seja mais apropriado e permitindo a obtenção de estruturas que sejam heterogêneas no espaço de atributos. Entretanto, os resultados dos *ensembles* continuam sendo uma única partição. Além disso, muitas vezes ainda é preciso ajustar seus parâmetros (como o número de *clusters*) e aplicar

técnicas de validação para escolher a melhor partição. Um outro problema dos *ensembles* é que eles consideram simultaneamente todas as partições iniciais para construir a partição consenso. Se muitas dessas partições tiverem uma qualidade ruim, o resultado final da combinação pode ser prejudicado, apesar do *ensemble* conter, inicialmente, algumas partições de ótima qualidade.

A abordagem de agrupamento multi-objetivo de Handl and Knowles (2004) também fornece soluções robustas em relação a diferentes propriedades dos dados. Além disso, encontra várias estruturas e não exige muito conhecimento prévio e ajuste de parâmetros. As suas principais restrições são: número de soluções encontradas muito elevado; as sugestões de melhores partições nem sempre correspondem às estruturas mais próximas às verdadeiras, dentre todas as que ele é capaz de encontrar; e grande variabilidade das soluções mais próximas às conhecidas para execução do algoritmo com uma mesma configuração dos parâmetros.

Todas essas questões se referem ao paradigma não supervisionado, ou seja, à análise dos dados sem considerar conhecimento prévio sobre os dados, embora conhecimento prévio seja bastante útil para orientar as escolhas que devem ser feitas. Contudo, há casos em que existe um conhecimento prévio parcial das estruturas presentes nos dados. Exemplos típicos podem ser encontrados em aplicações nas áreas de genômica funcional e análise de dados de expressão gênica (Golub et al. 1999; Alizadeh et al. 2000; Yeoh et al. 2002). Nessas áreas, a revelação de um conjunto de soluções alternativas é bastante útil, uma vez que os dados têm um alto potencial para conter várias interpretações significativas. Por exemplo, a análise de várias soluções alternativas em conjunto com o conhecimento prévio de uma delas pode auxiliar na análise das funções dos genes, uma vez que eles podem pertencer a mais de uma categoria funcional, ou ainda auxiliar na descoberta de subtipos de doenças, como o câncer (Golub et al. 1999; Alizadeh et al. 2000; Yeoh et al. 2002).

Dois trabalhos clássicos da literatura, que tratam da descoberta de subtipos de câncer utilizando análise de agrupamento em dados de expressão gênica, são os trabalhos de Golub et al. (1999) e Yeoh et al. (2002). Eles aplicam algoritmos tradicionais de agrupamento e mostram que subtipos da doença são revelados. Mais especificamente, Golub et al. (1999) aplicaram o algoritmo SOM (do inglês, *Self-Organizing Maps*) a dados de expressão gênica de pacientes com leucemia. Com isso, eles identificaram a divisão dos casos em dois tipos de leucemia: leucemia mielóide aguda (AML, do inglês *Acute Myeloid Leukemia*) e leucemia linfoblástica aguda, (ALL, do inglês *Acute Lymphoblastic Leukemia*). Posteriormente obtiveram a distinção de dois subtipos da ALL: linhagem T (T-ALL) e linhagem B (B-ALL).

No trabalho de Yeoh et al. (2002), um algoritmo de agrupamento hierárquico foi aplicado a amostras de seis subtipos de leucemia aguda linfoblástica pediátrica (*pediatric*

acute lymphoblastic leukemia) conhecidas, juntamente com amostras não classificadas, ou melhor, colocadas em um sétimo grupo de amostras heterogêneas que não pertencem aos outros subtipos. O agrupamento distinguiu apropriadamente os seis subtipos conhecidos e permitiu a identificação de um sétimo subtipo, até então desconhecido.

Nas duas aplicações anteriores, existe o conhecimento prévio de uma estrutura conhecida (a distinção AML/ALL em (Golub et al. 1999) e o conhecimento dos subtipos em (Yeoh et al. 2002)), mas não há um procedimento automático que utilize esse conhecimento para auxiliar o especialista a identificar os subtipos.

Na literatura, existem abordagens que consideram conhecimento prévio no processo de agrupamento. São as técnicas de agrupamento semi-supervisionado (Handl and Knowles 2006b; Demiriz et al. 1999), discutidas no Capítulo 3. Essas abordagens consideram o conhecimento das classes de uma pequena parte dos objetos, e não de uma estrutura que contemple todos eles. Além disso, o objetivo dessas propostas é melhorar o desempenho da classificação dos objetos e não auxiliar na descoberta de outras estruturas, como as que revelam os subtipos de doenças.

Todas essas questões referentes às dificuldades na análise de agrupamento, bem como a aplicação do agrupamento para a descoberta de subclasses, motivaram a abordagem proposta nesta Tese e auxiliaram na delimitação das metas que se deseja atingir. Na Seção 5.3 essas metas são destacadas de maneira clara e sucinta. Posteriormente, na Seção 5.4, a abordagem é detalhada, identificando como cada aspecto da proposta contribui para atingir as metas.

5.3 Metas a Serem Atingidas

A abordagem proposta neste trabalho, que consiste da combinação de técnicas de *ensembles* com técnicas multi-objetivos, é um *framework* para análise exploratória de dados por meio de agrupamento. O objetivo mais geral da proposta é ter uma ferramenta que facilite o trabalho dos especialistas do domínio na análise de seus dados, permitindo uma análise mais rápida e direta, sem a exigência de conhecimento técnico profundo sobre análise de agrupamento. Mais detalhadamente, as metas do *framework* proposto são as seguintes.

Seleção automática de modelo: não necessitar de conhecimento técnico profundo tanto sobre análise de agrupamento quanto a respeito dos dados para poder:

- escolher um critério de agrupamento ou algoritmo para aplicar aos dados,
- ajustar os parâmetros do(s) algoritmo(s) escolhido(s), e
- escolher dentre várias partições quais delas melhor se ajustam aos dados.

Diversidade: ser capaz de encontrar o maior número possível de estruturas subjacentes nos dados, de acordo com diferentes definições de *cluster* e em diferentes níveis de refinamento.

Concisão e qualidade: ter um conjunto de soluções conciso e de alta qualidade.

Estabilidade: ter um resultado estável quando as mesmas configurações são usadas.

Informações complementares: fornecer informações complementares que auxiliem os especialistas na análise das partições.

Semi-supervisão: permitir que o conhecimento de uma das estruturas subjacentes nos dados auxilie na obtenção das demais (ver Seção 5.5).

Visualização: fornecer uma visualização das soluções que facilite sua avaliação pelos especialistas.

As seis primeiras metas são atingidas pelo *ensemble* multi-objetivo proposto, detalhado na Seção 5.4. A sétima meta é alcançada com o método de visualização proposto no Capítulo 6. Esse método de visualização também contribui com a meta da semi-supervisão.

5.4 Ensemble Multi-objetivo - MOCLE

A essência do *framework* proposto é o algoritmo de *ensemble* multi-objetivo, MOCLE (do inglês, *Multi-Objective Clustering Ensemble*). A idéia geral do MOCLE é, dado um conjunto de objetos, $X = \{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_n\}$, utilizar um algoritmo genético multi-objetivo baseado em Pareto para encontrar um conjunto conciso e estável de partições alternativas de alta qualidade.

Como qualquer *ensemble*, o MOCLE é composto de dois pontos principais: (1) geração de um conjunto diverso de partições iniciais a serem combinadas e (2) determinação do consenso. O MOCLE difere dos *ensembles* tradicionais em dois aspectos principais, relacionados à obtenção do consenso. Primeiramente, o MOCLE busca por um conjunto de partições consenso, ao invés de uma única partição. Na verdade, o conjunto de soluções que o MOCLE retorna pode conter tanto partições que resultam da combinação de outras partições, quanto partições de alta qualidade que já apareciam dentre as partições iniciais. Além disso, o MOCLE combina pares de partições, iterativamente, em um processo de otimização que garante diferentes compromissos de qualidade das soluções. Mais detalhadamente, o MOCLE deve ter as seguintes particularidades para atingir as metas descritas na Seção 5.3:

- O algoritmo genético deve ser baseado em Pareto, para gerar como solução um conjunto de n^S partições, $\Pi_S = \{\pi^{S1}, \pi^{S2}, \dots, \pi^{Sn^S}\}$, com diferentes compromissos dos objetivos otimizados. Essa característica contribui com a segunda meta, que se refere à obtenção das várias estruturas alternativas presentes nos dados.
- Deve ser construída uma população inicial especial, $\Pi_I = \{\pi^{I1}, \pi^{I2}, \dots, \pi^{In^I}\}$, em que n^I é o número de partições iniciais e $\pi^{Ii} = \{c_1^{Ii}, c_2^{Ii}, \dots, c_{K^{Ii}}^{Ii}\}$ é uma partição de X em K^{Ii} clusters, tal que $\cup_{j=1}^{K^{Ii}} c_j^{Ii} = X$. Π_I deve ser construída utilizando diversos algoritmos de agrupamento que otimizam diferentes critérios. Isso permite a obtenção de vários tipos de cluster nas partições finais. Diversas configurações dos parâmetros dos algoritmos também devem ser empregadas de modo a gerar partições com diferentes níveis de refinamento (por exemplo, partições com diferentes números de clusters ou partições com clusters de várias densidades). Em resumo, a população inicial deve conter partições com diferentes tipos de clusters e em diferentes níveis de refinamento, visando fornecer subsídios para que o maior número possível de estruturas subjacentes do conjunto de dados seja descoberto. Isso contribui para a robustez da abordagem diante de diferentes propriedades dos dados, também relacionada à meta de diversidade. A população inicial construída dessa maneira garante a diversidade necessária ao conjunto de partições iniciais.
- Um operador de recombinação especial também deve ser empregado. Esse operador deve resultar em uma partição que seja o consenso entre duas partições pais, ou seja, ele é responsável pela parte de combinação (*ensemble*) da abordagem, tornando-a diferente da abordagem de agrupamento multi-objetivo de Handl and Knowles (2004). Além disso, esse operador permite a estabilidade e a concisão do conjunto de soluções, pois restringe a busca às soluções iniciais e suas combinações. Um operador desse tipo só faz sentido se o conjunto de partições iniciais já contiver partições boas em relação aos objetivos otimizados, o que é garantido com o procedimento para gerar a população inicial.

Com esse operador, as partições são combinadas duas a duas, iterativamente, no processo de evolução. As partições consenso geradas em cada aplicação do operador também são consideradas nas próximas combinações. Isso tudo dá à combinação realizada com esse operador, em relação às técnicas tradicionais de *ensembles*, a vantagem de evitar que as partições iniciais de baixa qualidade tenham um peso negativo na qualidade do resultado final. Em resumo, as partições iniciais de baixa qualidade são eliminadas gradativamente, enquanto as melhores soluções iniciais e as boas combinações vão sendo mantidas, contribuindo para novas combinações. Nesse sentido, esse operador também contribui para a qualidade das soluções.

- As funções objetivo a serem otimizadas devem corresponder a medidas de qualidade de uma partição. Elas devem avaliar essa qualidade de diferentes maneiras, que se complementem. Podem, por exemplo, favorecer cada um dos diferentes critérios empregados para gerar a população inicial. De fato, a utilização de tais funções serve para selecionar as partições de melhor qualidade, sob diferentes aspectos, para compor o conjunto de soluções. A otimização de funções diferentes também contribui para a robustez da abordagem frente a diferentes características dos dados, o que está relacionado à meta de diversidade. Os valores das medidas de qualidade também servem como informações complementares que podem ser empregadas pelo especialista na análise das soluções.

Entre outras coisas, elas permitem ao especialista escolher as partições que tenham as características que mais lhes interessem, dentre todas as presentes no conjunto de soluções, para fazer uma investigação mais detalhada. Por exemplo, o especialista pode preferir analisar as partições com *clusters* mais compactos, ou então as partições que não misturem as classes conhecidas (contexto semi-supervisionado). Essas informações também podem ser utilizadas para estabelecer as melhores soluções do conjunto, com base no formato do fronte de Pareto obtido, como feito por Handl and Knowles (2004). Esse tipo de informação complementar é gerado pela abordagem proposta. Entretanto, neste trabalho, não será dada ênfase à análise da sua utilização, por demandar a contribuição de um especialista, abrindo um novo leque de experimentos a serem realizados.

Um operador de mutação poderia ser empregado para um ajuste fino das partições consenso. Esse operador possibilitaria a obtenção de partições diferentes das consenso. Como o foco principal deste trabalho está na combinação das partições, não será utilizado o operador de mutação. Com isso, o espaço de busca fica limitado às combinações das partições iniciais. Assim, o algoritmo genético é empregado para selecionar as melhores combinações, e não para explorar todo o espaço das possíveis partições.

A Figura 5.5 contém uma representação do funcionamento geral do MOCLE. Nessa figura, n^A é o número de algoritmos diferentes utilizados. Resumidamente, vários algoritmos de agrupamento, com várias configurações de seus parâmetros, geram a população inicial do algoritmo genético. Essa constitui a fase de agrupamento propriamente dita. Essas partições iniciais são então selecionadas (validação) e combinadas (*ensemble*) no processo de otimização do algoritmo genético, resultando em um conjunto de partições conciso e de alta qualidade.

A utilização integrada de todos esses aspectos como um *framework* para a análise de agrupamento é o que reduz a necessidade de conhecimento prévio, tanto sobre os dados quanto em análise de agrupamento, auxiliando na seleção automática de modelo.

Figura 5.5: MOCLE

A semi-supervisão é atingida de duas maneiras, que podem ser empregadas em conjunto ou separadamente. A primeira delas é a utilização de um objetivo extra que considere o conhecimento prévio de uma das estruturas dos dados, que será discutido mais adiante. A segunda maneira consiste da aplicação do método de visualização que será discutido no Capítulo 6, também considerando a estrutura conhecida.

Para implementar essas idéias, várias alternativas podem ser empregadas para cada um dos itens. A seguir são detalhadas as alternativas utilizadas neste trabalho, bem como alguns outros aspectos práticos do MOCLE.

5.5 Implementação da Proposta

O primeiro aspecto importante na definição de um algoritmo genético é a forma de representação de um indivíduo. Como já foi mencionado, cada indivíduo corresponde a uma partição do conjunto de dados. Para representar essa partição, optou-se por usar uma estrutura de dados que representa diretamente o conceito de partição. Um indivíduo é representado como um vetor de conjuntos. Cada conjunto, por sua vez, representa um *cluster* e contém os rótulos dos objetos desse *cluster*. Por exemplo, considere um conjunto de seis objetos $X = \{\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3, \mathbf{x}_4, \mathbf{x}_5, \mathbf{x}_6\}$. Considere também uma partição

desse conjunto em três *clusters* $\pi^i = \{c_1^i, c_2^i, c_3^i\}$, em que $c_1^i = \{\mathbf{x}_1\}$, $c_2^i = \{\mathbf{x}_3, \mathbf{x}_4, \mathbf{x}_5\}$ e $c_3^i = \{\mathbf{x}_2, \mathbf{x}_6\}$. A Figura 5.6 ilustra a representação dessa partição.

Essa representação evita o problema de redundância dos indivíduos (vários indivíduos diferentes representando uma mesma solução). Na representação utilizada, cada partição é representada unicamente por um indivíduo, pois a representação dos indivíduos se baseia no conceito de partição. Juntamente com o operador de recombinação proposto, é evitada também a necessidade de correção dos cromossomos para garantir que o indivíduo seja válido. Isso ocorre porque o operador de recombinação gera uma partição válida como resultado. Além disso, qualquer solução é possível de ser representada, ou seja, a representação é completa.

c_1^i	c_2^i	c_3^i
$\{\mathbf{x}_1\}$	$\{\mathbf{x}_3, \mathbf{x}_4, \mathbf{x}_5\}$	$\{\mathbf{x}_2, \mathbf{x}_6\}$

Figura 5.6: Exemplo da representação de um indivíduo

Para o processo de otimização, foram realizados experimentos com dois algoritmos diferentes: SPEA (Zitzler and Thiele 1999) e NSGA-II (Deb et al. 2002), descritos na Seção 2.4. Esses algoritmos foram escolhidos por sua popularidade. Para o SPEA existe uma versão melhorada, o SPEA2 (Zitzler et al. 2001). Segundo Zitzler et al. (2001), em seus experimentos, os algoritmos SPEA2 e NSGA-II tiveram sempre o melhor desempenho, sendo que vantagens do SPEA2 sobre o NSGA-II apenas se destacam quando são empregados muitos objetivos, o que não é o caso do MOCLE. Na realidade, o algoritmo principal escolhido para o MOCLE foi o NSGA-II, mas foram também realizados alguns testes com o SPEA, por ele estar disponível na biblioteca utilizada para o NSGA-II e ser fácil a sua integração no MOCLE.

Para o operador de recombinação especial, que encontra o consenso entre duas partições pais, foram utilizadas duas abordagens de *ensembles* de agrupamentos descritas na literatura: MCLA (Strehl and Ghosh 2002) e HGBF (Fern and Brodley 2004), descritas no Capítulo 4. Essas abordagens são baseadas em particionamento de grafos, que as torna mais robustas em relação a diferentes propriedades dos dados se comparadas às técnicas baseadas em matriz de co-associação (Topchy et al. 2005). Além disso, podem ser utilizadas para encontrar o consenso entre apenas duas partições.

Para a aplicação da recombinação, são selecionados dois indivíduos, π^1 e π^2 , utilizando seleção por torneio binário. Com as duas partições selecionadas, é montado o grafo apropriado, dependendo da técnica que estiver sendo considerada, MCLA ou HGBF. Em seguida, o número de *clusters* que a partição filha, π^F , deverá conter, K^F , é escolhido

5.5 Implementação da Proposta

aleatoriamente no intervalo de variação do número de *clusters* dos pais ($K^F \in [K^1, K^2]$). O grafo é então particionado em K^F partes. Neste trabalho, o particionamento foi feito com o METIS¹ (*Multilevel Graph Partitioning System*) (Karypis and Kumar 1999), que é uma técnica frequentemente empregada no contexto de *ensembles* baseados em particionamento de grafo, incluindo as duas abordagens utilizadas aqui. Com o resultado do grafo particionado, a partição consenso π^F é construída segundo a indicação da técnica empregada (MCLA ou HBGF), conforme apresentado no Capítulo 4.

Por exemplo, considere a população de cinco indivíduos mostrada na Tabela 5.1. Suponha que, por torneio binário, os indivíduos π^1 e π^2 tenham sido selecionados para a aplicação do operador de recombinação. Como ambas as partições possuem dois *clusters*, o número de *clusters* da partição filha, π^F , também deverá ser dois ($K^F = 2$). Se a técnica HBGF estiver sendo utilizada para a recombinação, será construído o grafo da Figura 5.7. O particionamento desse grafo em duas partes também é ilustrado na Figura 5.7 pela linha tracejada. A partição filha π^F é obtida pela divisão dos objetos resultante desse particionamento, ou seja, π^F será formada pelos *clusters* $c_1^F = \{\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3, \mathbf{x}_4, \mathbf{x}_5\}$ e $c_2^F = \{\mathbf{x}_6, \mathbf{x}_7, \mathbf{x}_8, \mathbf{x}_9\}$. Esse indivíduo será considerado na construção da população da próxima geração e, caso venha a fazer parte dela, poderá ser selecionado para ser combinado com outro indivíduo, pela aplicação do operador de recombinação na próxima geração.

O operador de recombinação será aplicado tantas vezes quantas forem necessárias em cada geração, dependendo do AG considerado, sendo que, a cada aplicação, um par de indivíduos é selecionado da população, para ser combinado. Os indivíduos gerados com a recombinação que integrarem a população da próxima geração também poderão, por sua vez, ser selecionados e combinados.

Tabela 5.1: População

Partição	<i>Clusters</i>
π^1	$c_1^1 = \{\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3, \mathbf{x}_4\}$, $c_2^1 = \{\mathbf{x}_5, \mathbf{x}_6, \mathbf{x}_7, \mathbf{x}_8, \mathbf{x}_9\}$
π^2	$c_1^2 = \{\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3, \mathbf{x}_4, \mathbf{x}_5, \mathbf{x}_6\}$, $c_2^2 = \{\mathbf{x}_7, \mathbf{x}_8, \mathbf{x}_9\}$
π^3	$c_1^3 = \{\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3, \mathbf{x}_9\}$, $c_2^3 = \{\mathbf{x}_4, \mathbf{x}_5, \mathbf{x}_8\}$, $c_3^3 = \{\mathbf{x}_6, \mathbf{x}_7\}$
π^4	$c_1^4 = \{\mathbf{x}_6, \mathbf{x}_7, \mathbf{x}_8, \mathbf{x}_9\}$, $c_2^4 = \{\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3\}$, $c_3^4 = \{\mathbf{x}_4, \mathbf{x}_5\}$
π^5	$c_1^5 = \{\mathbf{x}_1, \mathbf{x}_2\}$, $c_2^5 = \{\mathbf{x}_3, \mathbf{x}_4, \mathbf{x}_8\}$, $c_3^5 = \{\mathbf{x}_5, \mathbf{x}_6, \mathbf{x}_7, \mathbf{x}_9\}$

¹<http://glaros.dtc.umn.edu/gkhome/metis/metis/overview>

Figura 5.7: Grafo gerado na aplicação do operador de recombinação com a técnica HBGF

Quanto às funções objetivo utilizadas, foram considerados dois contextos: não supervisionado e semi-supervisionado. No contexto não supervisionado, a abordagem é usada para uma análise de agrupamento pura, sem considerar conhecimento prévio dos dados. Para esse caso, como função objetivo, foram empregadas duas medidas de validação de agrupamento que medem a qualidade das partições com base exclusivamente nos dados: a variância intra-*cluster* (Equação 3.6), que também corresponde ao critério otimizado no *k*-médiãs, e a conectividade (Equação 3.7), apresentadas na Seção 3.4. Esses são os mesmos objetivos otimizados no algoritmo MOCK (Handl and Knowles 2004). As mesmas motivações que levaram os autores do MOCK a escolherem essas medidas, bem como os bons resultados obtidos por eles, justificam seu emprego neste trabalho.

Como já destacado, essas medidas representam dois aspectos fundamentalmente diferentes de qualidade de um agrupamento e contrabalançam suas tendências de aumentar ou diminuir com o número de *clusters*. Essa característica é importante para uma boa exploração do espaço de soluções, evitando a convergência para soluções triviais: *n clusters* com um único objeto, no caso da variância intra-*cluster*, e apenas um *cluster* com *n* elementos, no caso da conectividade (Handl and Knowles 2004).

Na versão semi-supervisionada, da mesma maneira que no agrupamento semi-supervisionado, em que as informações das classes e *clusters* devem ser consistentes e se complementar de maneira que a sua combinação possa fornecer resultados melhores, neste trabalho é necessário que a estrutura conhecida esteja relacionada com as demais para que sua utilização possa contribuir com a obtenção das outras.

Assim, para a versão semi-supervisionada, foram consideradas três funções objetivo a serem otimizadas simultaneamente: a variância intra-*cluster*, a conectividade e uma terceira função que considera a estrutura conhecida. O objetivo da utilização dessa terceira

função é fazer com que as partições resultantes tenham os *clusters* mais puros possível em relação à estrutura conhecida. Isso favorece a obtenção de partições que sejam refinamentos dessa estrutura, ou seja, partições que contenham *clusters* que subdividam os grupos da estrutura conhecida, misturando o mínimo possível os objetos dos *clusters* conhecidos.

Nos trabalhos sobre agrupamento semi-supervisionado discutidos no Capítulo 3, foram apresentadas duas medidas para considerar o conhecimento prévio: o *CR* (Handl and Knowles 2006b) e o índice Gini (Demiriz et al. 1999). O *CR* não é apropriado para o propósito deste trabalho, uma vez que penaliza subdivisões dos *clusters*. Demiriz et al. (1999) usam o índice Gini para encontrar *clusters* puros, mas também indicam o índice ganho de informação como uma outra medida apropriada para esse fim. Essas duas medidas constituem critérios para divisão dos nós nas árvores de decisão (Breiman 1996; Raileanu and Stoffel 2004; Rokach and Maimon 2005) e não penalizam subdivisões dos *cluster*. Segundo Raileanu and Stoffel (2004), a maioria dos estudos empíricos apontam que não há como distinguir qual delas é melhor como critério de divisão. Mostram ainda que essas funções discordam em apenas 2% dos casos. Segundo Breiman (1996), o índice Gini prefere divisões que colocam a classe maior em um único nó puro e todas as outras em um outro nó. Já o ganho de informação favorece a geração de nós de tamanho balanceado. Essa distinção aparece principalmente para um número grande de classes. No caso desse trabalho, a utilização do índice Gini favoreceria partições que tivessem um *cluster* puro grande (em relação aos *clusters* conhecidos) e misturassem os outros *clusters* conhecidos em um único grupo, em detrimento de partições que subdividissem mais esse *cluster* grande, mas que também separassem melhor os outros *clusters* conhecidos. Assim, para este trabalho, optou-se pela utilização do ganho de informação como terceira função objetivo.

Essa medida é baseada no conceito de entropia da teoria da informação. Seja a estrutura conhecida $\pi^E = \{c_1^E, c_2^E, \dots, c_{K^E}^E\}$, com K^E *clusters*, o ganho de informação (*ganho*) de uma partição $\pi = \{c_1, c_2, \dots, c_k\}$, com k *clusters*, é dado pela Equação 5.1.

$$ganho(\pi) = - \sum_{c_i^E \in \pi^E} \frac{|c_i^E|}{n} \log_2\left(\frac{|c_i^E|}{n}\right) + \sum_{c_j \in \pi} \frac{|c_j|}{n} \sum_{c_i^E \in \pi^E} \frac{|c_i^E \cap c_j|}{|c_j|} \log_2\left(\frac{|c_i^E \cap c_j|}{|c_j|}\right) \quad (5.1)$$

5.6 Considerações Finais

Neste capítulo foram detalhados os problemas que motivaram este trabalho, destacadas as metas a serem atingidas e apresentado o *framework* proposto. Foi também discutido como cada aspecto da proposta contribui para atingir cada uma das metas.

O MOCLE combina as idéias de agrupamento multi-objetivo e *ensemble* de agrupa-

mentos para superar as dificuldades da análise de agrupamentos tradicional. Além de manter a característica de robustez em relação a diferentes tipos de *clusters*, que ambas as abordagens garantem, a sua integração ajuda a superar as limitações individuais de cada uma delas. O aspecto multi-objetivo permite que o *ensemble* gere um conjunto de partições consenso, evitando também a necessidade de determinação prévia do número de *clusters*, e garante a qualidade das soluções, mesmo com um número elevado de partições iniciais ruins. Por outro lado, o aspecto *ensemble* auxilia na concisão e estabilidade do conjunto de soluções, uma vez que limita o espaço de busca às partições iniciais e suas combinações.

O MOCLE permite ainda a integração de conhecimento prévio, por meio das funções objetivo, para auxiliar na obtenção de outras estruturas presentes nos dados. De maneira sucinta, as metas são atingidas com a contribuição das seguintes combinações de aspectos do MOCLE:

- Seleção automática de modelo: população inicial e funções objetivo.
- Diversidade: algoritmo genético baseado em Pareto, população inicial e funções objetivo.
- Concisão e qualidade: população inicial, operador de recombinação e funções objetivo.
- Estabilidade: população inicial e operador de recombinação.
- Informações complementares: funções objetivo.
- Semi-supervisão: funções objetivo, se usada uma função que considere conhecimento prévio.

A meta da visualização é atingida com o método de visualização de um conjunto de partições, que será apresentado no Capítulo 6. Esse método também contribui para a meta da semi-supervisão. Ele pode ser empregado com diferentes finalidades. A primeira delas é contribuir na utilização do conhecimento prévio de uma das estruturas dos dados para auxiliar na identificação de outras estruturas. A segunda finalidade é no auxílio à análise dos resultados de um agrupamento com critérios externos de validação, complementando as informações de índices como o *CR*, que comparam uma partição obtida com agrupamento com uma estrutura previamente conhecida.

Método de Visualização

6.1 Considerações Iniciais

No Capítulo 5 foram apresentadas as metas a serem atingidas neste trabalho e a parte central do *framework* proposto, que é o *ensemble* multi-objetivo de agrupamentos, MOCLE.

Neste capítulo, será apresentado um método para a visualização de um conjunto de partições. Uma descrição detalhada do método de visualização será feita na Seção 6.2. Esse método de visualização pode ser visto como parte integrante do *framework* proposto no Capítulo 5 ou como uma contribuição independente dele. No primeiro caso, o método contribui com duas das metas destacadas no Capítulo 5: visualização e semi-supervisão. Apesar desse método somente poder ser aplicado quando alguma estrutura dos dados é conhecida, ele pode ser usado tanto com as partições resultantes da configuração semi-supervisionada do MOCLE, quanto com as da não supervisionada. Enquanto contribuição independente do *framework*, o método pode ser aplicado a qualquer conjunto de partições, gerado com qualquer técnica de agrupamento ou combinação (*ensembles* ou MOCK, por exemplo). Mais ainda, pode ser aplicado a um conjunto composto de partições obtidas com diferentes técnicas. Nesse sentido, o método auxilia na avaliação da qualidade das partições, permitindo uma comparação mais detalhada de seu conteúdo, de maneira a complementar as análises feitas com índices de validação externa como o *CR*. As formas de utilização do método serão discutidas na Seção 6.3. É importante mencionar que não foram encontrados trabalhos semelhantes na literatura para a visualização de um conjunto de partições.

6.2 Descrição do Método

O método de visualização proposto é adequado para a visualização de um conjunto de n^P partições quaisquer, $\Pi = \{\pi^1, \pi^2, \dots, \pi^{n^P}\}$, dado o conhecimento prévio de uma estrutura presente nos dados, π^E . Para evitar confusão na explicação do método que se segue, a palavra “classe” será utilizada para se referir aos grupos da estrutura conhecida, sendo o termo “*cluster*” reservado para os grupos do conjunto de partições a serem visualizadas. A utilização dessa nomenclatura se restringirá a este capítulo.

Esse método se baseia em um esquema para colorir uma partição, que associa cores a cada um de seus *cluster* e, conseqüentemente, aos objetos pertencentes a esses *clusters*.

Para ilustrar a aplicação do método, será considerado um conjunto de dados com 10 objetos, $X = \{\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3, \mathbf{x}_4, \mathbf{x}_5, \mathbf{x}_6, \mathbf{x}_7, \mathbf{x}_8, \mathbf{x}_9, \mathbf{x}_{10}\}$, a estrutura conhecida, π^E , e o conjunto de partições a serem visualizadas, $\Pi = \{\pi^1, \pi^2\}$, dados na Tabela 6.1.

Tabela 6.1: Partições do exemplo do método de visualização

Partição	<i>Clusters</i>
π^E	$c_1^E = \{\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3, \mathbf{x}_4, \mathbf{x}_5, \mathbf{x}_6\}$, $c_2^E = \{\mathbf{x}_7, \mathbf{x}_8, \mathbf{x}_9, \mathbf{x}_{10}\}$
π^1	$c_1^1 = \{\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3\}$, $c_2^1 = \{\mathbf{x}_4, \mathbf{x}_5, \mathbf{x}_7\}$, $c_3^1 = \{\mathbf{x}_6\}$, $c_4^1 = \{\mathbf{x}_8, \mathbf{x}_9, \mathbf{x}_{10}\}$
π^2	$c_1^2 = \{\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_7, \mathbf{x}_8, \mathbf{x}_9\}$, $c_2^2 = \{\mathbf{x}_3, \mathbf{x}_4, \mathbf{x}_5\}$, $c_3^2 = \{\mathbf{x}_6\}$, $c_4^2 = \{\mathbf{x}_{10}\}$

O primeiro passo no procedimento para gerar a visualização das partições é a associação de uma cor para cada classe de π^E . Para manter a consistência, as mesmas associações de cores às classes deve ser consideradas ao se colorir cada uma das partições a serem visualizadas. Por exemplo, a cor **verde** pode ser associada à classe c_1^E e a cor **laranja**, à classe c_2^E .

Para se colorir uma partição, deve-se contar o número de objetos de cada classe, que está presente em cada *cluster* da partição. Por exemplo, o *cluster* c_1^2 tem 2 objetos da classe c_1^E ($\mathbf{x}_1$ e $\mathbf{x}_2$) e 3 objetos da classe c_2^E ($\mathbf{x}_7$, $\mathbf{x}_8$ e $\mathbf{x}_9$). Com base nessa informação, determina-se a classe majoritária de cada *cluster* (isto é, a classe com o maior número de objetos no *cluster*). No exemplo, a classe majoritária de c_1^2 é c_2^E . Em seguida, cada *cluster* deve ser “colorido” com a cor da sua classe majoritária. A Tabela 6.2 mostra os resultados da aplicação desses passos a ambas as partições do exemplo, π^1 e π^2 .

Tabela 6.2: Passos para “colorir” as partições do exemplo

Partição	Cluster	Número de objetos		Classe majoritária	Cor	Intensidade
		c_1^E	c_2^E			
π^1	c_1^1	3	0	c_1^E	verde	alta
	c_2^1	2	1	c_1^E	verde	média
	c_3^1	1	0	c_1^E	verde	baixa
	c_4^1	0	3	c_2^E	laranja	alta
π^2	c_1^2	2	3	c_2^E	laranja	alta
	c_2^2	3	0	c_1^E	verde	alta
	c_3^2	1	0	c_1^E	verde	média
	c_4^2	0	1	c_2^E	laranja	média

Para distinguir entre os *clusters* de uma partição que possuem a mesma classe majoritária, é definido um esquema de gradação de cores que associa um valor de intensidade a cada *cluster*. Dados todos os *clusters* de uma partição com a mesma classe majoritária, o maior valor de intensidade deve ser associado ao *cluster* com o maior número de objetos dessa classe. A menor intensidade deve ser associada ao *cluster* com o menor número de objetos da classe majoritária. As outras intensidades devem ser distribuídas aos *clusters* de acordo com o número de objetos da classe majoritária que eles contêm. Por exemplo, a partição π^1 tem três *clusters* com classe majoritária c_1^E (c_1^1 , c_2^1 e c_3^1), associados à cor **verde**. Portanto, são necessárias três intensidades diferentes que aqui serão representadas por alta, média e baixa. O *cluster* com mais objetos da classe majoritária é c_1^1 (3 objetos). Esse *cluster* terá a intensidade alta. O segundo *cluster* com mais objetos da classe majoritária é c_2^1 (2 objetos) e terá a intensidade média. O *cluster* c_3^1 , com 1 objeto, terá a intensidade baixa. Todas as partições consideradas, incluindo a estrutura conhecida, devem ser “coloridas” dessa maneira, utilizando o mesmo valor para a intensidade mais elevada. No exemplo, considerando todas as partições, são necessários no máximo três níveis de intensidade para distinguir os *clusters* com a mesma classe majoritária (alta, média e baixa).

Uma vez que todas as partições tenham sido “coloridas”, os objetos devem ser organizados como linhas de uma tabela, usando a estrutura conhecida como referência. Cada coluna dessa tabela representa uma das partições. Cada célula (i, j) da tabela deve ser preenchida com um número representando o *cluster* ao qual o i -ésimo objeto pertence na j -ésima partição (um rótulo do *cluster*) e a cor associada a esse *cluster*. Como convenção, a primeira coluna dessa tabela representa os identificadores dos objetos no conjunto de

dados. A segunda coluna representa a estrutura conhecida. Cada uma das demais colunas representa uma das partições a serem visualizadas. As colunas podem ser ordenadas pelo número de *clusters* das partições que elas representam (do menor para o maior), caso sejam representadas partições com diferentes números de *clusters*. A Figura 6.1 mostra o resultado final da visualização do exemplo apresentado.

Objeto	π^E	π^1	π^2
$\mathbf{x}_1$	1	1	1
$\mathbf{x}_2$	1	1	1
$\mathbf{x}_3$	1	1	2
$\mathbf{x}_4$	1	2	2
$\mathbf{x}_5$	1	2	2
$\mathbf{x}_6$	1	3	3
$\mathbf{x}_7$	2	2	1
$\mathbf{x}_8$	2	4	1
$\mathbf{x}_9$	2	4	1
$\mathbf{x}_{10}$	2	4	4

Maior *cluster* com classe majoritária c_1^E

Segundo maior *cluster* com classe majoritária c_1^E

Terceiro maior *cluster* com classe majoritária c_1^E

Maior *cluster* com classe majoritária c_2^E

Segundo maior *cluster* com classe majoritária c_2^E

Figura 6.1: Visualização do exemplo

6.3 Utilização do Método de Visualização

O objetivo principal do método de visualização é auxiliar na análise de qualquer conjunto de partições. Diferentes tipos de informação podem ser obtidas, dependendo de como são essas partições. O método pode ser aplicado a partições geradas com um mesmo algoritmo e vários valores de seus parâmetros, como o número de *clusters*, a partições obtidas com diferentes algoritmos, auxiliando na sua comparação, ou a um conjunto de partições resultantes de uma única técnica, como as obtidas com o MOCK e o MOCLE. Algumas das informações dependem também do tipo de algoritmo utilizado. Por exemplo, para os algoritmos hierárquicos, o processo de separação das classes pode ser claramente identificado.

Considere a visualização de um conjunto de 8 partições, mostrada na Figura 6.2. Essas partições possuem de 2 a 9 *clusters*. As colunas estão ordenadas pelo número de *clusters* das partições (π^1 tem dois *clusters*, π^2 tem três e assim por diante). A estrutura conhecida, π^E , possui 6 classes: c_1^E , representada em **vermelho**, c_2^E em **violeta**, c_3^E em **marrom**, c_4^E em **azul**, c_5^E em **laranja** e c_6^E em **verde**.

Objeto	π^E	π^1	π^2	π^3	π^4	π^5	π^6	π^7	π^8
$\mathbf{x}_1$	2	2	2	2	3	3	4	5	6
$\mathbf{x}_2$	2	2	2	2	3	3	4	5	6
$\mathbf{x}_3$	2	2	2	2	3	3	4	5	6
$\mathbf{x}_4$	2	2	2	2	3	3	4	5	5
$\mathbf{x}_5$	2	2	2	2	3	3	4	5	5
$\mathbf{x}_6$	0	2	3	4	5	6	7	8	9
$\mathbf{x}_7$	0	2	3	4	5	6	7	8	9
$\mathbf{x}_8$	0	2	3	4	5	6	7	8	9
$\mathbf{x}_9$	0	1	1	1	1	1	1	1	1
$\mathbf{x}_{10}$	1	1	1	1	2	2	2	2	2
$\mathbf{x}_{11}$	1	1	1	1	2	2	2	2	2
$\mathbf{x}_{12}$	1	1	1	1	2	2	2	2	2
$\mathbf{x}_{13}$	3	2	2	2	3	3	3	4	4
$\mathbf{x}_{14}$	3	2	2	2	3	3	3	3	3
$\mathbf{x}_{15}$	3	2	2	2	3	3	3	3	3
$\mathbf{x}_{16}$	3	2	2	2	3	3	3	4	4
$\mathbf{x}_{17}$	3	2	2	2	3	3	3	4	4
$\mathbf{x}_{18}$	4	2	2	2	3	3	4	5	5
$\mathbf{x}_{19}$	4	2	3	3	4	4	5	6	7
$\mathbf{x}_{20}$	4	2	3	3	4	4	5	6	7
$\mathbf{x}_{21}$	5	2	3	3	4	5	6	7	8
$\mathbf{x}_{22}$	5	2	3	3	4	5	6	7	8
$\mathbf{x}_{23}$	5	2	3	3	4	5	6	7	8

Maior *cluster* de c_1^E

2º maior *cluster* de c_1^E

Maior *cluster* de c_2^E

2º maior *cluster* de c_2^E

Maior *cluster* de c_3^E

Maior *cluster* de c_4^E

2º maior *cluster* de c_4^E

Maior *cluster* de c_5^E

Maior *cluster* de c_6^E

Figura 6.2: Exemplo das informações contidas na visualização

Algumas das informações que podem ser obtidas pelo especialista a partir aplicação do método de visualização, independentemente de como as partições foram geradas, são:

Subdivisões de uma classe: a subdivisão de uma dada classe em mais de um *cluster* pode dar suporte a descoberta de subclasses nos dados. Esse é o aspecto da visualização que contribui para a meta da semi-supervisão apresentada no Capítulo 5. O conhecimento prévio fica relacionado às cores de cada classe e as subdivisões das classes podem ser observadas com a presença de diferentes intensidades nas cores. Por exemplo, nas partições π^7 e π^8 da Figura 6.2, a classe c_4^E , em **azul**, está dividida em dois *clusters*, um com os objetos $\mathbf{x}_{13}$, $\mathbf{x}_{16}$ e $\mathbf{x}_{17}$, representados em **azul** escuro, e outra com os objetos $\mathbf{x}_{14}$ e $\mathbf{x}_{15}$, representados em *azul* claro.

Similaridades/diferenças gerais entre as partições: essas similaridades e diferenças podem ser observadas, tanto considerando a partição toda, quanto olhando no nível das classes, pela observação dos padrões de cores na visualização. As partições podem ser comparadas com a estrutura conhecida e também entre si. As partições

77

são similares se as mesmas cores são associadas aos mesmos objetos, mesmo se as cores não são aquelas das classes dos objetos. Também pode ser identificado se o comportamento das partições é similar para algumas classes e difere em outras. Uma homogeneidade das cores de uma classe entre as partições significa um comportamento similar, e a sua mistura indica diferença.

Se as partições são provenientes de algoritmos diferentes, a visualização pode ajudar a distinguir qual algoritmo é mais apropriado para identificar cada uma das classes. Nas partições dos algoritmos apropriados para separar uma classe, os objetos dessa classe, e apenas eles, vão estar representados com a cor correta, enquanto que nas partições dos demais algoritmos eles vão estar representados com a mesma cor de objetos de outras classes, quer essa cor seja a de sua classe, quer essa cor seja a da classe dos outros objetos.

Associações incorretas e objetos problemáticos: essas associações incorretas são objetos que são associados ao mesmo *cluster* incorreto na maioria das partições. Os objetos problemáticos são objetos que são agrupados incorretamente nas partições com números de *cluster* próximos do verdadeiro, mas que podem ser isolados em *clusters* pequenos, ou de um único elemento, nas partições com um número de *clusters* mais elevado. Em ambos os casos, esses objetos podem ser ruído, objetos que tenham sido originalmente classificados erroneamente ou objetos que realmente apresentam um comportamento atípico em relação aos demais objetos da mesma classe. Uma vez identificados, os objetos que são ruído podem ser descartados e os que foram erroneamente rotulados, corrigidos. Por outro lado, os objetos com comportamento inesperado devem ser mantidos, uma vez que eles podem conter informações importantes para o especialista do domínio que está analisando os dados.

Na visualização, uma associação incorreta pode ser vista pela representação do objeto com uma cor diferente dos outros objetos de sua classe. Por exemplo, na Figura 6.2, o objeto $\mathbf{x}_{18}$ pertence originalmente à classe c_5^E , representada em **laranja**. Entretanto, nas partições π^1 a π^5 , esse objeto está representado em **azul**, e nas partições π^6 a π^8 , em **vermelho**. Isso é, esse objeto foi sempre agrupado com objetos de outras classes.

Os objetos problemáticos podem ser identificados da seguinte forma. Nas partições com número de *clusters* pequeno, são representados com uma cor diferente da cor de sua classe. Nas partições com k a partir de um determinado valor, passam a ser sempre representados corretamente, na cor da sua classe, mas com uma intensidade menor do que a do *cluster* principal dessa classe. Por exemplo, o objeto $\mathbf{x}_9$, na Figura 6.2, pertence à classe c_2^E . Ele foi agrupado com objetos da classe c_3^E nas

partições com k variando de 2 a 4 (π^1 , π^2 e π^3), sendo representado em marrom. Para as partições com valores de k maiores, esse objeto foi isolado em um *cluster* em que ele é o único elemento, sendo representado sozinho em **violeta** claro. Com isso, nessas partições com número de *clusters* mais elevado, as classes c_2^E e c_3^E estão bem separadas, ou seja, todos os objetos de c_2^E , e apenas eles, estão representados em **violeta** e todos os objetos da classe c_3^E , e apenas eles, estão representados em marrom.

Processo de separação das classes: o processo de separação das classes, com o aumento do número de *clusters*, pode ser observado na visualização quando as partições apresentadas possuem diferentes números de *clusters* e foram geradas com um único algoritmo. O início da separação de uma classe pode ser observado na visualização quando a cor da classe começa a aparecer ou a cor começa a representar a maioria dos objetos da classe. Deve ficar claro que esse processo de separação não representa o processo de separação que o algoritmo realiza, mas apenas o comportamento do algoritmo para diferentes números de *clusters*. Analisando esse processo, é possível identificar:

- Classes que são rapidamente distinguidas das demais. Tais classes começam a aparecer separadas das demais para as partições com valor baixo de k (geralmente $k = 2$). Para ser considerada bem separada, uma classe deve ter a maioria de seus objetos representados com a cor correta, independentemente da intensidade, e ter poucos ou nenhum objeto de outras classes representados com essa cor. Na Figura 6.2, a classe c_3^E , representada em **marrom**, aparece separada das demais classes em todas as partições, com apenas um objeto de outra classe agrupado com seus objetos em algumas partições.
- Classes que são muito similares entre si. Tais classes aparecem como um único *cluster* em todas ou quase todas as partições. Nos casos em que elas aparecem separadas, a separação apenas ocorre quando todas as outras classes já foram separadas e, em alguns casos, apenas nas partições com número de *clusters* bastante elevado. A dificuldade de um algoritmo em separar duas classes pode indicar similaridades não esperadas entre seus objetos. Na visualização, tais classes aparecem separadas apenas para partições com números de *clusters* maiores que o verdadeiro. Por exemplo, na Figura 6.2, as classes c_1^E e c_4^E aparecem como um único *cluster* até a partição π^5 , com $k = 6$, sendo representadas em **azul**. Nas partições com $k > 6$, a classe c_4^E permanece representada em **azul**, sua cor correta, e a classe c_1^E passa a ser representada em **vermelho**. Pode haver casos em que as classes nunca são separadas, sendo representadas

com a mesma cor em todas as partições. Não há exemplos desses casos na Figura 6.2.

6.4 Considerações Finais

Neste capítulo foi descrito um método de visualização de um conjunto de partições. Dependendo de algumas características desse conjunto, diferentes informações podem ser obtidas. Esse método depende do conhecimento prévio de uma estrutura existente nos dados. Essa visualização contribui com as metas de visualização e semi-supervisão do *framework* proposto, quando aplicada às partições resultantes do MOCLE. As informações extraídas da visualização que colaboram diretamente com essas metas são: as subdivisões de uma classe, permitindo a utilização do conhecimento das classes na investigação da existência de outras estruturas que sejam refinamentos da estrutura conhecida, e as similaridades/diferenças gerais entre as partições, auxiliando o especialista na análise das partições obtidas. O método pode também ser visto como uma contribuição independente, servindo para a análise de qualquer conjunto de partições. O método de visualização proposto permite ainda a identificação de associações incorretas e objetos problemáticos além da observação do processo de separação das classes com o aumento do número de *clusters*.

No Capítulo 7 serão descritos os dados e métodos empregados na realização dos experimentos com o *framework* proposto. Nesse capítulo, serão discutidas as configurações de parâmetros consideradas nos vários algoritmos utilizados, justificadas as técnicas escolhidas e apresentada a metodologia empregada para avaliar os resultados.

Métodos e Experimentos

7.1 Considerações Iniciais

Nos Capítulos 5 e 6 foram apresentadas as técnicas desenvolvidas neste trabalho, sendo o algoritmo de *ensemble* multi-objetivo, MOCLE, descrito no Capítulo 5 e o método de visualização no Capítulo 6.

Neste capítulo serão detalhados os experimentos realizados para ilustrar o emprego das técnicas propostas e mostrar como elas atingem as metas estabelecidas. Mais especificamente, na Seção 7.2 serão apresentados os conjuntos de dados empregados nos experimentos; na Seção 7.3 serão detalhados os experimentos propriamente ditos, mostrando os algoritmos e parâmetros utilizados em cada caso; e o procedimento adotado para a avaliação dos resultados será apresentado na Seção 7.4.

7.2 Conjuntos de Dados

Para os experimentos realizados, foram considerados dez conjuntos de dados. Cada um deles possui um conjunto de n^E estruturas conhecidas $\Pi_E = \{\pi^{E1}, \pi^{E2}, \dots, \pi^{En^E}\}$. Quatro desses conjuntos são artificiais e foram criados especialmente para conter várias estruturas heterogêneas e em diferentes níveis de refinamento. Os conjuntos artificiais são: **ds2c2sc13**, **ds3c3sc6**, **ds4c2sc8** e **spiralsquare**. Dois dos conjuntos (**glass** e **iris**) são *benchmarks* comumente usados para testar problemas em aprendizado de máquina e encontram-se disponíveis no repositório de dados da UCI (Newman et al. 1998). Esses dados representam aplicações reais, porém já foram intensamente manipulados e são amplamente empregados na literatura. Os resultados obtidos com esses conjuntos de dados são facilmente reproduzidos e comparados com resultados de trabalhos semelhantes. Os outros quatro conjuntos (**golub**, **proteinas**, **leukemia** e **lung**) são dados reais que re-

presentam problemas de bioinformática, sendo três deles de dados de expressão gênica. A maioria desses conjuntos contém pelo menos duas estruturas distintas. Nos conjuntos *benchmarks* e reais, essas estruturas correspondem a diferentes classificações conhecidas dos dados. Assim, é assumido que as divisões dos dados nas classes estão em concordância com algum dos critérios de agrupamentos empregados, sendo as classes conhecidas referidas como *clusters*. Algumas das classificações podem não ter relação com um critério de agrupamento, resultando em um baixo desempenho de todas as técnicas de agrupamento utilizadas. Apenas dois dos conjuntos apresentam apenas uma estrutura conhecida: o conjunto *iris*, que será empregado por ser um dos conjuntos de dados mais freqüentemente utilizados nos testes de algoritmos de agrupamento, e o *lung*, que, apesar de não ter disponível uma segunda estrutura, sabe-se que um de seus *clusters* apresenta subdivisões (Bhattacharjee et al. 2001).

A Tabela 7.1 contém um resumo das características dos conjuntos de dados. Nessa tabela, n é o número de objetos presentes no conjunto de dados, d é a dimensionalidade dos dados (número de atributos), n^E é o número de estruturas distintas que são conhecidas e K^{Ej} é o número de *clusters* da j -ésima estrutura. Pode-se observar que esses dados possuem características bastante variadas. Foram utilizados dados contendo de um a quatro estruturas diferentes, contendo de dois a 27 *clusters* e um número de objetos variando entre 72 e 2000. Quanto à dimensionalidade, os conjuntos artificiais contêm apenas dois atributos, os *benchmarks* apresentam uma dimensionalidade intermediária (entre quatro e nove) e os conjuntos reais apresentam uma dimensionalidade bastante elevada (entre 125 e 3571). A Figura 7.1 mostra os conjuntos de dados artificiais.

Tabela 7.1: Características dos conjuntos de dados

Tipo	Conjunto de dados	n	d	n^E	K^{E1}	K^{E2}	K^{E3}	K^{E4}
Artificiais	ds2c2sc13	588	2	3	2	5	13	-
	ds3c3sc6	905	2	2	3	6	-	-
	ds4c2sc8	485	2	2	2	8	-	-
	spiralsquare	2000	2	2	2	6	-	-
<i>Benchmarks</i>	glass	214	9	3	2	5	6	-
	iris	150	4	1	3	-	-	-
Reais	golub	72	3571	4	2	3	4	2
	proteinas	698	125	2	4	27	-	-
	leukemia	327	271	2	3	7	-	-
	lung	197	1000	1	4	-	-	-

O conjunto de dados **ds2c2sc13** foi especialmente projetado para conter três estruturas diferentes: E1, E2 e E3. A Figura 7.1(a) mostra esse conjunto de dados e suas estruturas. Como pode ser observado, E1 é a estrutura mais geral e contém dois *clusters*, E2 é um refinamento de E1 e contém cinco *clusters*, e E3 é um refinamento de E2, com 13 *clusters*. Também pode ser notado que os *clusters* nesse conjunto de dados têm tamanhos e formas variados. Os $n = 588$ objetos desse conjunto de dados estão divididos nos *clusters* conforme pode ser observado na Tabela 7.2. Como já mencionado na Seção 5.2, E1 é a estrutura que mais se destaca, sendo obtida facilmente por qualquer técnica de agrupamento. Nessa estrutura, os *clusters* têm um formato aproximadamente esférico e estão muito bem separados uns dos outros. Já nas estruturas E2 e E3, os *clusters* são bastante heterogêneos. Em E2, há um *cluster* em formato de sorriso, um *cluster* alongado e três *clusters* aproximadamente globulares. Já na estrutura E3, o mesmo *cluster* em forma de sorriso aparece, mas o *cluster* alongado da estrutura E2 pode ser visto como três *clusters* esféricos. Além disso, cada um dos *clusters* globulares de E2 pode ser visto como três *clusters* alongados em E3.

Tabela 7.2: Tamanho dos *clusters* - **ds2c2sc13**

Estrutura	Número de objetos no <i>cluster</i>												
	1	2	3	4	5	6	7	8	9	10	11	12	13
E3	108	24	27	73	32	49	33	40	28	24	32	71	47
E2	108	124			114			92			150		
E1	232				356								

O conjunto de dados **ds3c3sc6** (Figura 7.1(b)) contém duas estruturas: E1, com 3 *clusters* e E2, com 6 *clusters*, um refinamento de E1. Nesse conjunto de dados, os *clusters* também têm formatos e tamanhos variados. Entretanto, ao contrário do que ocorre em **ds2c2sc13**, os *clusters* em ambas as estruturas não estão bem separados. Os $n = 905$ objetos desse conjunto estão divididos conforme Tabela 7.3.

Tabela 7.3: Tamanho dos *clusters* - **ds3c3sc6**

Estrutura	Número de objetos no <i>cluster</i>					
	1	2	3	4	5	6
E2	79	197	287	56	135	151
E1	276		287	342		

O conjunto de dados **ds4c2sc8** (Figura 7.1(c)) também contém duas estruturas: E1,

Figura 7.1: Conjuntos de dados artificiais

com 2 *clusters* e seu refinamento, E2, com 8 *clusters*. Nesse conjunto de dados, os *clusters* também têm formatos e tamanhos variados, e não estão bem separados. Os $n = 485$ objetos desse conjunto estão divididos conforme mostrado na Tabela 7.4.

Tabela 7.4: Tamanho dos *clusters* - **ds4c2sc8**

Estrutura	Número de objetos no <i>cluster</i>							
	1	2	3	4	5	6	7	8
E2	40	82	52	53	69	70	62	57
E1	174			311				

O conjunto de dados **spiralsquare** foi construído a partir de dois conjuntos de dados descritos em (Handl and Knowles 2004) e disponibilizados juntamente com o código fonte do MOCK. Esse conjunto de dados contém duas estruturas: E1 com 2 *clusters* e E2 com 6 *clusters* (também um refinamento de E1), como pode ser observado na Figura 7.1(d). A estrutura mais facilmente distinguida é a de E1, que apresenta dois *clusters* esféricos e bem separados um do outro. Em E2, um dos *clusters* de E1 é subdividido em dois *clusters* em espiral e o outro é subdividido em outros quatro *clusters* globulares bastante próximos um do outro. Os $n = 2000$ objetos desse conjunto de dados estão distribuídos conforme mostrado na Tabela 7.5.

Tabela 7.5: Tamanho dos *clusters* - **spiralsquare**

Estrutura	Número de objetos no <i>cluster</i>					
	1	2	3	4	5	6
E2	500	500	250	250	250	250
E1	1000		1000			

O conjunto de dados **glass** foi obtido do repositório de dados da UCI (Newman et al. 1998) e se refere à classificação de vidros. A estrutura mais refinada, denominada E3, considera os *clusters*: BWFP - vidro processado de janela de edifício (*building windows float processed*), BWNFP - vidro não processado de janela de edifício (*building windows non float processed*), VWFP - vidro processado de janela de carro (*vehicle windows float processed*), C - vidro de recipientes (*containers*), T - vidro de louças (*tableware*) e H - vidro de farol (*headlamps*). Além disso, foram consideradas duas outras estruturas: E1, com 2 *clusters*, que separa os objetos em vidro de janela (BWFP, BWNFP e VWFP) e vidro que não é de janela (C, T e H), e E2, com 5 *clusters*, que distingue entre vidro

de edifício (BWFP e BWNFP), vidro de carro (VWFP), recipiente (C), louça (T) e farol (H). Os $n = 214$ objetos estão divididos nos *clusters* conforme ilustrado na Tabela 7.6. Os atributos correspondem ao índice de refração e à porcentagem de diversos óxidos no vidro (de Na, Fe, K, entre outros). Como os atributos correspondem a medidas em unidades diferentes, todos os atributos foram normalizados para o intervalo $[0, 1]$.

Tabela 7.6: Tamanho dos *clusters* - **glass**

Estrutura	Número de objetos no <i>cluster</i>					
	BWFP	BWNFP	VWFP	C	T	H
E3	70	76	17	13	9	29
E2	146		17	13	9	29
E1	163			51		

O conjunto de dados **iris** também foi obtido do repositório de dados da UCI (Newman et al. 1998). Esse conjunto contém três *clusters* correspondentes a três espécies da planta *iris*: *Iris Setosa*, *Iris Versicolor*, *Iris Virginica*. Cada *cluster* possui 50 objetos. Os atributos correspondem ao comprimento e largura das pétalas e sépalas, em centímetros. Esses dados contêm apenas uma estrutura.

O conjunto de dados **golub** consiste de dados de expressão gênica de amostras de pacientes com leucemia aguda (Golub et al. 1999). Para esses dados foram consideradas quatro estruturas conhecidas distintas. As duas estruturas principais se referem aos tipos e subtipos de leucemia aguda. A estrutura E1 representa a divisão das amostras em dois *clusters*, um com as amostras de leucemia aguda linfoblástica, ou ALL (do inglês, *Acute Lymphoblastic Leukemia*) e outro com as amostras de leucemia mielóide aguda, ou AML (do inglês, *Acute Myeloid Leukemia*). A estrutura E2, com 3 *clusters*, é um refinamento de E1, em que o *cluster* com as amostras de ALL é subdividido em dois: um com as amostras de células de linhagem T (T-ALL) e outro com as amostras de células de linhagem B (B-ALL). As duas outras estruturas consideradas correspondem a outros tipos de informações sobre as amostras. E3 indica a instituição de onde foram obtidas as células analisadas: DFCI (do inglês, *Dana-Farber Cancer Institute*), CALGB (do inglês, *Cancer and Leukemia Group B*), SJCRH (do inglês, *St. Jude Children's Research Hospital*) e CCG (do inglês, *Children's Cancer Group*). E4 indica se as amostras eram provenientes de medula óssea, BM (do inglês, *Bone Marrow*), ou de sangue periférico, PB (do inglês, *Peripheral Blood*). Os $n = 72$ objetos estão divididos nos *clusters* conforme mostrado na Tabela 7.7. Para essa base foi aplicado o seguinte pré-processamento, como em (Dudoit et al. 2000): todos os valores menores do que 100 foram convertidos para 100

e todos os valores maiores do que 16000 foram convertidos para 16000. Dos 7129 genes originais, foram selecionados apenas os genes que possuíam um valor de expressão tal que $\text{minimo}/\text{maximo} > 5$ e $(\text{maximo} - \text{minimo}) > 500$, o que resulta em um total de 3571 genes. Em seguida, foi aplicado o logaritmo na base 10.

Tabela 7.7: Tamanho dos *clusters* - **golub**

Estrutura	Número de objetos no <i>cluster</i>			
	B-ALL	T-ALL	AML	
E2	38	9	25	
E1	47		25	
Estrutura	Número de objetos no <i>cluster</i>			
	DFCI	CALGB	SJCRH	CCG
E3	44	15	8	5
Estrutura	Número de objetos no <i>cluster</i>			
	BM		PB	
E4	62		10	

O conjunto de dados **proteínas** se refere a dobras de proteínas (Tan et al. 2003). Para as análises, foram consideradas duas estruturas: E1, que representa a divisão dos objetos nas 4 classes de dobras principais, e E2, que representa a divisão em 27 tipos de dobras, sendo um refinamento de E1. Os $n = 698$ objetos estão divididos nos *clusters* conforme ilustrado na Tabela 7.8. Os atributos correspondem a características extraídas a partir das seqüências das proteínas. Como todos os atributos são dados em porcentagem, nenhuma normalização foi realizada.

Tabela 7.8: Tamanho dos *clusters* - **proteínas**

Estrutura	Número de objetos no <i>cluster</i>														
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
E2	19	16	32	15	18	16	74	21	29	13	16	32	12	13	16
E1	116						226								

Estrutura	Número de objetos no <i>cluster</i>											
	16	17	18	19	20	21	22	23	24	25	26	27
E2	77	23	24	40	22	17	24	18	15	15	40	41
E1	260									96		

O conjunto de dados **leukemia**, referido na literatura como St. Jude *leukemia* (Yeoh et al. 2002), contém dados de expressão gênica obtidos com *microarrays*. Os dados se referem a subtipos de leucemia aguda linfoblástica pediátrica (*pediatric acute lymphoblastic leukemia*). Os $n = 327$ objetos estão divididos em 6 grupos diagnosticáveis (BCR-ABL, E2A-PBX1, Hyperdiploid>50, MLL, T-ALL e TEL-AML1) e um outro grupo que contém amostras que não se enquadram em nenhum dos grupos anteriores (OTHERS). Como este trabalho envolve a análise de várias estruturas alternativas em um mesmo conjunto de dados, considerou-se também uma estrutura que divide esses dados em três grupos mais gerais: as leucemias linhagem B ou B-ALL, que englobam as amostras de BCR, E2A, TEL, MLL e Hyperdiploid>50, as leucemias linhagem T ou T-ALL e as amostras rotuladas como OTHERS. Assim, a divisão dos objetos em três *clusters* é considerada a estrutura mais geral, E1, e a estrutura com 7 *clusters* a estrutura mais refinada, E2. A distribuição dos objetos nos *clusters* pode ser observada na Tabela 7.9. O conjunto original contém 12558 atributos (genes). Uma versão pré-processada do conjunto de dados, com todos os passos de pré-processamento descritos em (Yeoh et al. 2002), foi obtida em <http://sdmc.lit.org.sg/GEDatasets/>. Neste trabalho foram utilizados como atributos apenas os genes que melhor definem cada grupo (40 genes por grupo), identificados por Yeoh et al. (Yeoh et al. 2002) com a medida qui-quadrado. Os valores dos atributos foram transformados para o intervalo $[0, 1]$.

Tabela 7.9: Tamanho dos *clusters* - **leukemia**

Estrutura	Número de objetos no <i>cluster</i>						
	BCR	E2A	Hyperdiploid>50	MLL	TEL	T-ALL	OTHERS
E2	15	27	64	20	79	43	79
E1	205					43	79

O conjunto de dados **lung**, originalmente empregado em (Bhattacharjee et al. 2001), contém apenas uma estrutura (E1) com 4 *clusters*. Ela consiste de amostras relacionadas a câncer de pulmão, englobando amostras de tecido normal (NL) e 3 tipos de câncer de pulmão: adenocarcinomas (AD), carcinomas de células escamosas (*squamous cell carcinomas*) (SQ) e 20 carcinóides (COID). Os $n = 197$ objetos estão distribuídos nos *clusters* conforme ilustrado na Tabela 7.10. O *cluster* com as amostras AD é bastante heterogêneo e sabe-se da existência de sub-grupos nesse *cluster*. Para esse conjunto de dados, foi empregada a mesma versão utilizada por Monti et al. (2003), gentilmente cedida por esses autores.

Tabela 7.10: Tamanho dos *clusters* - lung

Estrutura	Número de objetos no <i>cluster</i>			
	AD	SQ	COID	NL
E1	139	21	20	17

7.3 Experimentos

Conforme antecipado no Capítulo 5, foram feitos testes preliminares com dois algoritmos genéticos multi-objetivo baseados em Pareto, o SPEA e o NSGA-II. Nesses testes, ambos os algoritmos mostraram desempenhos similares. Diante disso, como o NSGA-II é uma técnica mais recente, comparável inclusive à versão melhorada do SPEA, o SPEA2, o algoritmo NSGA-II foi escolhido para os experimentos mostrados neste trabalho. Os resultados obtidos com o SPEA encontram-se descritos em um artigo que foi aceito na *6th International Conference on Hybrid Intelligent Systems (HIS'06)* (Faceli et al. 2006).

Para analisar as capacidades do MOCLE, foram realizados experimentos com quatro configurações diferentes:

MUH: contexto não supervisionado, com a variância (*var*) e a conectividade (*con*) como objetivos, e operador de recombinação usando a técnica HBGF (veja Seção 4.2.3).

MUM: contexto não supervisionado, com a variância (*var*) e a conectividade (*con*) como objetivos, e operador de recombinação usando a técnica MCLA (veja Seção 4.2.3).

MSH: contexto semi-supervisionado, com a variância (*var*), a conectividade (*con*) e o ganho de informação (*ganho*) como objetivos, e operador de recombinação usando a técnica HBGF.

MSM: contexto semi-supervisionado, com a variância (*var*), a conectividade (*con*) e o ganho de informação (*ganho*) como objetivos, e operador de recombinação usando a técnica MCLA.

A Tabela 7.11 contém os valores dos parâmetros empregados em todas as técnicas, que dependem do conjunto de dados utilizado. K^{min} e K^{max} são os números de *clusters* mínimo e máximo usados nos experimentos, v é o número de vizinhos mais próximos empregado no cálculo da conectividade para o MOCK e o MOCLE, NN é o número de vizinhos mais próximos considerado no algoritmo SNN e n^I é o tamanho da população inicial do MOCLE. As justificativas para a escolha dos valores para esses parâmetros,

Tabela 7.11: Valores dos parâmetros

Conjunto de dados	K^{min}	K^{max}	v	NN	n^I
ds2c2sc13	2	26	30	12, 29, 59, 176, 235	114
ds3c3sc6	3	12	46	18, 45, 91, 272, 362	42
ds4c2sc8	2	16	25	10, 24, 49, 146, 194	66
spiralsquare	2	12	100	40, 100, 200, 600, 800	73
glass	2	12	11	4, 11, 21, 64, 86	48
iris	3	6	8	3, 8, 15, 45, 60	20
golub	2	8	4	1, 4, 7, 22, 29	46
proteinas	4	54	35	14, 35, 70, 209, 279	170
leukemia	3	14	17	7, 16, 33, 98, 131	38
lung	4	8	10	4, 10, 20, 59, 79	22

juntamente com todas as configurações e procedimentos adotados nos experimentos, serão apresentados a seguir.

A população inicial, $\Pi_I = \{\pi^{I1}, \pi^{I2}, \dots, \pi^{In^I}\}$, com n^I indivíduos foi construída utilizando quatro algoritmos de agrupamento: KM e LM, que são baseados na compactação dos *clusters*, e LS e SNN, que são baseados no encadeamento dos objetos nos *clusters*. Com isso, a população inicial conterá diferentes tipos de *clusters*. Para os diferentes níveis de refinamento, decidiu-se que Π_I deveria conter partições com número de *clusters* $k \in [K^{min}, K^{max}]$, em que $K^{min} = \min_{\pi^{Ej} \in \Pi_E} K^{Ej}$ (menor número de *clusters* dentre os números de *clusters* das estruturas conhecidas), e $K^{max} = 2 \max_{\pi^{Ej} \in \Pi_E} K^{Ej}$. K^{min} e K^{max} para cada conjunto de dados podem ser observados na Tabela 7.11. Além dos diferentes tipos de *clusters* que esses algoritmos encontram, os algoritmos KM, LM e LS foram escolhidos por serem tradicionais e largamente empregados em diversas aplicações, inclusive na análise de dados de expressão gênica (Jain and Dubes 1988; Jiang et al. 2004; Xu and Wunsch 2005). O algoritmo SNN é uma técnica mais recente, mais foi selecionado por sua tolerância à alta dimensionalidade e por lidar bem com ruídos e *outliers* (Ertöz et al. 2002).

Para gerar a população inicial, os algoritmos foram utilizados da seguinte maneira:

KM: Como esse algoritmo depende da inicialização dos centróides, para minimizar o problema com os ótimos locais, o algoritmo foi executado 30 vezes para cada valor de k , escolhendo-se aleatoriamente os centróides iniciais. A partição com a menor variância intra-*cluster* obtida para cada valor de k foi selecionada para compor a partição inicial. Assim, uma partição para cada valor de k foi usada em Π_I .

LM e LS: Com esses algoritmos, as partições foram obtidas gerando as árvores e as cortando de maneira a produzir uma partição para cada valor de k .

SNN: O algoritmo SNN foi executado com diferentes configurações de seus parâmetros:

- *NN*: 2%, 5%, 10%, 20%, 30% e 40% de n ,
- *topic* e *merge*: 0, 0,2, 0,4, 0,6, 0,8 e 1,
- *strong*: 0,25 (*default*) e
- *noise* e *label*: 0 (todos os pontos devem ser rotulados e nenhum deve ser excluído como ruído).

Experimentos preliminares mostraram que variar os parâmetros *strong*, *noise* e *label* não produz resultados muito diferentes. Dentre as partições produzidas com a execução do SNN com todas essas combinações de valores de seus parâmetros, apenas as que continham um número de *clusters* k no intervalo de interesse foram selecionadas para compor a população inicial.

Com isso, a população inicial possui um número diferente de indivíduos para cada conjunto de dados. O tamanho da população inicial para cada caso pode ser observado na Tabela 7.11. O software Cluster 3.0 (Hoon et al. 2004), obtido em <http://bonsai.ims.u-tokyo.ac.jp/~mdehoon/software/cluster/software.htm>, foi utilizado para executar os algoritmos KM, LM e LS. Para o SNN, foi empregada a implementação enviada pelos seus autores, mediante solicitação.

Para a execução do MOCLE, o número de gerações foi fixado em 50, para todas as configurações e todos os conjuntos de dados. Experimentos preliminares mostraram que aumentar o número de gerações não altera o fronte de Pareto obtido como solução. Para o parâmetro v (número de vizinhos mais próximos), usado no cálculo do objetivo conectividade, foi utilizado um valor diferente para cada conjunto de dados: 5% de n . Esses valores podem ser observados na Tabela 7.11. A mesma população inicial foi considerada em todas as configurações.

Os resultados obtidos com o MOCLE foram comparados com os algoritmos de agrupamento utilizados para gerar a população inicial e com o agrupamento multi-objetivo de Handl and Knowles (2005b) - MOCK, e com o *ensemble* de agrupamentos de Strehl and Ghosh (2002), utilizando a função supra-consenso para selecionar o melhor consenso dentre os gerados com as funções CSPA, HGPA e MCLA. Nos casos do MOCK e do ES, foram utilizadas as implementações dos respectivos autores. A implementação do MOCK foi obtida em <http://dbkgroup.org/handl/mock/> e a implementação do *ensemble* de Strehl foi obtida em <http://www.lans.ece.utexas.edu/~strehl/soft.html>.

O MOCK, assim como o MOCLE, gera um fronte de Pareto, com soluções que correspondem a diferentes compromissos entre os objetivos otimizados (*var* e *con*). Essas soluções apresentam números de *clusters* variados. Para executar esse algoritmo, foram utilizados os mesmos valores dos parâmetros correspondentes do MOCLE: 5% de n para o parâmetro v e $K^{max} = 2 \max_{\pi^{Ej} \in \Pi_E} K^{Ej}$ para o parâmetro que define o número máximo de *clusters* (ver valores na Tabela 7.11). Quanto ao número de frentes de controle, foram usados cinco, como em (Handl and Knowles 2004).

O *ensemble* de Strehl e Ghosh, daqui por diante abreviado por ES, gera uma única partição com k *clusters* como solução, em que k é um parâmetro do algoritmo, que deve ser fornecido pelo usuário. Assim, para gerar um conjunto de soluções para ser comparado com o MOCLE, o algoritmo foi executado várias vezes, com os mesmos valores de k usados para gerar a população inicial do MOCLE ($k \in [K^{min}, K^{max}]$). As mesmas partições iniciais usadas no MOCLE foram também utilizadas no ES.

Apesar de os algoritmos individuais e o ES não gerarem um conjunto de soluções, com os experimentos realizados, foi obtido um conjunto de n^S soluções, $\Pi_S = \{\pi^{S1}, \pi^{S2}, \dots, \pi^{Sn^S}\}$, para cada técnica. Esses conjuntos são necessários para se poder comparar essas técnicas com o MOCLE, que gera um conjunto de soluções. Para os algoritmos KM, LM e LS, o conjunto de soluções contém uma partição para cada valor de $k \in [K^{min}, K^{max}]$. Para o algoritmo SNN, o conjunto de soluções contém partições nesse intervalo, mas o número de partições para cada k pode variar. Inclusive, pode não haver partições para alguns valores de k . No caso do *ensemble* ES, é gerada uma partição para cada valor de k , mas essa solução contém no máximo k *clusters*. Para o MOCK e o MOCLE, Π_S constitui um fronte de Pareto em que as soluções correspondem a diferentes compromissos entre os objetivos otimizados. Como a otimização da variância favorece o surgimento de partições com número elevado de *clusters* e a conectividade, partições com poucos *clusters*, uma medida contrabalança a tendência da outra, fazendo com que o fronte de Pareto gerado possua soluções com variados números de *clusters*.

7.4 Metodologia de Avaliação dos Experimentos

Para a análise dos experimentos, inicialmente foi realizada uma análise da qualidade geral das soluções, seguida de uma análise mais detalhada. Nos dois casos, todas as configurações do MOCLE (MUH, MUM, MSH e MSM) foram comparadas com os algoritmos individuais (KM, LM, LS e SNN) e com o algoritmo de agrupamento multi-objetivo MOCK e o *ensemble* ES.

Como os algoritmos MOCLE e MOCK não são determinísticos, foram realizadas 30 execuções, com as mesmas configurações iniciais. O ES depende da ordem de apresentação das partições iniciais. Assim, para o ES, também foram realizadas 30 execuções, cada uma

apresentando as partições em uma ordem aleatória. Para o KM, na análise geral também foram consideradas todas as 30 execuções realizadas. Na análise detalhada, foi empregado o conjunto de partições do KM que foi considerado na população inicial do MOCLE.

A qualidade geral das soluções foi avaliada com o índice CR , descrito na Seção 3.4 (Equação 3.5), pois, como já mencionado, esse é um dos índices de validação externa mais utilizados, não sendo sensível ao número de *clusters*, como outros índices.

A primeira questão a ser avaliada é a habilidade dos algoritmos em gerar soluções de alta qualidade. Para isso, foi considerado o conhecimento do conjunto de estruturas $\Pi_E = \{\pi^{E1}, \pi^{E2}, \dots, \pi^{En^E}\}$ sabidamente existentes em cada conjunto de dados. Esse conhecimento externo é usado para selecionar a melhor solução de cada conjunto em relação a cada estrutura conhecida. Deve-se considerar que tal conhecimento não foi empregado pelos algoritmos na determinação das soluções, com exceção das configurações semi-supervisionadas do MOCLE, que utilizaram apenas parte desse conhecimento. Para a obtenção dos valores do CR a serem comparados foram seguidos os seguintes procedimentos.

Para os algoritmos LM, LS e SNN, que são determinísticos, foi calculado o CR entre cada partição π^{Si} do conjunto de soluções, $\Pi_S = \{\pi^{S1}, \pi^{S2}, \dots, \pi^{Sn^S}\}$ e cada estrutura π^{Ej} do conjunto de estruturas conhecidas Π_E . Para cada estrutura conhecida, π^{Ej} , foi selecionada a melhor partição do conjunto de soluções em relação a essa estrutura, ou seja, a partição π^{Si} que apresenta o maior CR quando comparada com π^{Ej} .

Para as demais técnicas (KM, MOCK, ES e MOCLE), o mesmo foi feito para cada execução. Assim, para cada estrutura conhecida, π^{Ej} , foi selecionada uma partição π^{Si} mais próxima de π^{Ej} para cada uma das 30 execuções. Em seguida, para cada π^{Ej} , foram calculados a média e o desvio padrão do CR dessas 30 partições π^{Si} selecionadas. Esse procedimento é o mesmo seguido em (Handl and Knowles 2006a). A única diferença é que neste trabalho estão sendo consideradas várias estruturas para cada conjunto de dados e em (Handl and Knowles 2006a), uma única estrutura.

Além do conjunto de soluções completo, o MOCK retorna um número reduzido (1 a 4) de soluções recomendadas como as melhores do fronte de Pareto. Para esse conjunto de recomendações, também foram selecionadas as partições que mais se aproximam de cada uma das estruturas conhecidas em cada execução e calculados a média e desvio padrão para as 30 execuções. A recomendação do MOCK será denominada MKR, enquanto que o MOCK considerando todas as soluções será denominado apenas por MK.

Com isso, as referências futuras ao CR dizem respeito tanto ao CR calculado para uma partição, quanto a média do CR das partições de várias execuções, dependendo se as técnicas são as determinísticas ou não determinísticas.

Para verificar a significância estatística das diferenças de desempenho (CR) entre os

algoritmos foi empregado o teste Friedman e o pós teste de Nemenyi, apropriado para comparar vários algoritmos aplicados em múltiplos conjuntos de dados (Hollander and Wolfe 1999; Demšar 2006). Segundo Demšar (2006), esse teste é mais apropriado para analisar algoritmos de aprendizado de máquina do que o teste ANOVA, pois não se pode garantir que as suposições necessárias para o ANOVA não sejam violadas.

Para aplicar esse teste, inicialmente são ordenados os algoritmos para cada conjunto de dados, separadamente. O melhor algoritmo recebe o *rank* 1, o segundo melhor, *rank* 2, e assim por diante. Para os casos de empate, a média dos *ranks* é associada a todos os algoritmos que têm desempenho igual.

Dados n^A algoritmos e n^D conjuntos de dados, r_i^j é o *rank* do j -ésimo algoritmo no i -ésimo conjunto de dados. No teste Friedman, a hipótese nula é a de que todos os algoritmos são equivalentes e, portanto, a média dos *ranks*, $R_j = \frac{1}{n^D} \sum_{i=1}^{n^D} r_i^j$ deve ser igual. A estatística Friedman é dada pela Equação 7.1. Como χ_F^2 é indesejavelmente conservadora, Iman and Davenport (1980) derivaram uma estatística melhor, dada pela Equação 7.2, que é distribuída de acordo com a distribuição F com $(n^A - 1)$ e $(n^A - 1)(n^D - 1)$ graus de liberdade.

$$\chi_F^2 = \frac{12n^D}{n^A(n^A + 1)} \left[\sum_{j=1}^{n^A} R_j^2 - \frac{n^A(n^A + 1)^2}{4} \right] \quad (7.1)$$

$$F_F = \frac{(n^D - 1)\chi_F^2}{n^D(n^A - 1) - \chi_F^2} \quad (7.2)$$

Se a hipótese nula for rejeitada, será aplicado o pós teste de Nemenyi, apropriado para comparar cada algoritmo com todos os outros. Segundo esse teste, dois algoritmos são significativamente diferentes se a média dos *ranks*, R_j , difere de pelo menos uma diferença crítica, CD , dada pela Equação 7.3, em que os valores críticos, q_α , são baseados na *Studentized range statistic*, dividido por $\sqrt{2}$ (Demšar 2006).

$$CD = q_\alpha \sqrt{\frac{n^A(n^A + 1)}{6n^D}} \quad (7.3)$$

Neste trabalho, serão considerados como algoritmos a serem comparados, as quatro configurações do MOCLE (MUH, MUM, MSH e MSM), o *ensemble* ES, o MOCK, considerando todas as soluções (MK) e somente as recomendações (MKR), e os algoritmos individuais (KM, LM, LS e SNN), perfazendo um total de 11 técnicas ($n^A = 11$). Como conjuntos de dados, foram considerados os 10 conjuntos descritos na Seção 7.2 e cada uma de suas estruturas conhecidas. Assim, o conjunto de dados **glass**, por exemplo, será considerado como três casos diferentes, uma vez que, para cada estrutura conhecida, será obtida uma solução diferente. Considerando os 10 conjuntos de dados e suas respectivas

estruturas, serão analisadas um total de $n^D = 22$ estruturas.

Ainda na análise geral, além da habilidade das técnicas em encontrar as estruturas conhecidas, foram avaliadas a concisão e a estabilidade das técnicas que geram um conjunto de soluções. Para a concisão, será analisado o número médio de soluções obtidos nas 30 execuções. Para a estabilidade, serão observados os desvios padrão do CR calculados como já descrito.

A análise detalhada ilustra outros aspectos importantes do MOCLE que não podem ser observados na análise geral. Para isso, foi selecionada uma execução de cada técnica em que foram realizadas 30 execuções. A execução selecionada foi aquela que possuía a partição com maior CR em relação a estrutura mais refinada. Assim, para cada técnica, foi analisado um conjunto de soluções, Π_S , por meio de gráficos empregando o CR e os valores de var e con de cada partição π^{Si} , e também o método de visualização proposto no Capítulo 6. Além disso, para as soluções do MOCLE, será avaliado se cada solução já fazia parte da população inicial, ou se foi gerada durante o processo de evolução, por meio da combinação de outras partições. Para a análise com o método de visualização, foi selecionado apenas um conjunto de dados, pois sua aplicação a todos os conjuntos geraria um grande volume de tabelas e apenas um conjunto já serve para ilustrar bem a utilização do método.

7.5 Considerações Finais

Neste capítulo foi apresentada a metodologia empregada para a realização dos experimentos. Em resumo, foram empregados 10 conjuntos de dados com características bastante distintas. Quatro configurações diferentes do MOCLE foram investigadas, duas no contexto não supervisionado e duas no semi-supervisionado. As partições iniciais para o MOCLE foram geradas com quatro algoritmos de agrupamento distintos e com diferentes números de *clusters*. Além dos algoritmos individuais, o MOCLE também foi comparado com o *ensemble* de agrupamentos ES e o algoritmo de agrupamento multi-objetivo MOCK, cujos princípios foram combinados para a proposição do MOCLE. As avaliações dos resultados serão feitas com um índice de validação bastante tradicional, o CR , sendo também aplicado um teste estatístico para avaliar a significância estatística das diferenças de desempenho entre as técnicas.

No Capítulo 8 serão apresentados os resultados obtidos com os experimentos. Será ainda discutido como cada meta foi atingida.

Resultados

8.1 Considerações Iniciais

No Capítulo 7 foram apresentados os dados utilizados, as configurações de parâmetros e a metodologia seguida para a realização e avaliação dos experimentos.

Neste capítulo serão apresentados e discutidos os resultados obtidos com esses experimentos. Inicialmente, na Seção 8.2, será feita uma análise da qualidade geral das soluções obtidas com o MOCLE, em comparação com as outras técnicas. O primeiro aspecto dessa análise é a comparação do desempenho das técnicas na obtenção das diversas estruturas conhecidas de cada um dos conjuntos de dados, por meio do índice de validação CR . Para auxiliar nessa análise, foi empregado um teste estatístico, como descrito na Seção 7.4. A estabilidade e a concisão dos conjuntos de soluções obtidos com as técnicas de combinação também serão comparados. A estabilidade se refere a obtenção das várias estruturas, e não propriamente de todo o conjunto de soluções, será avaliada utilizando o desvio padrão do CR nas várias execuções. A concisão será analisada pelo número médio de soluções presentes nos conjuntos de soluções das 30 execuções.

Em seguida, na Seção 8.3, será dada uma visão mais detalhada dos conjuntos de soluções, por meio de gráficos que mostram a qualidade de todas as partições de cada conjunto em relação a uma das estruturas conhecidas. Para isso, será escolhido um conjunto de soluções de cada técnica. Dois tipos de gráfico serão utilizados: CR versus k , que permite observar as técnicas que mais se aproximaram da obtenção da estrutura considerada, e variância versus conectividade, que mostra o desempenho das técnicas em achar os melhores compromissos entre essas medidas, que fazem parte dos critérios de agrupamento investigados.

Finalmente, na Seção 8.4, será ilustrada a aplicação do método de visualização, tanto

para visualizar as partições e comparar seus conteúdos, quanto para auxiliar na investigação da existência de subclasses.

8.2 Qualidade Geral das Soluções

O primeiro objetivo desta seção é mostrar a capacidade do MOCLE em encontrar partições próximas das estruturas conhecidas, independente das definições de *cluster* nas quais essas estruturas se baseiam e também do nível de refinamento em que ocorrem. Para isso, como já mencionado no Capítulo 7, as partições que mais se assemelham a cada uma das estruturas foram selecionadas e foram feitos os cálculos para o *CR*, conforme previamente descrito.

As tabelas 8.1 até 8.10 contêm os valores do *CR* e *k* para os conjuntos de dados *ds2c2sc13*, *ds3c3sc6*, *ds4c2sc8*, *spiralsquare*, *glass*, *iris*, *golub*, *proteinas*, *leukemia* e *lung*, respectivamente. Conforme já relatado no Capítulo 7, Seção 7.4, para os algoritmos LM, LS e SNN, cada valor corresponde à melhor solução em relação à cada estrutura conhecida. Para as demais técnicas, os valores correspondem à média (M) e desvio padrão (DP) do *CR* em 30 execuções. Em cada estrutura, o melhor valor médio do *CR* está destacado. Na análise que se segue, o termo “algoritmos individuais” diz respeito aos algoritmos KM, LM, LS e SNN e o termo “técnicas de combinação” se refere aos *ensembles* ES, MUH, MUM, MSH e MSM, e ao algoritmo multi-objetivo MK, uma vez que ele também combina vários critérios de agrupamento, embora de maneira distinta dos *ensembles*. O conjunto de recomendações do MOCK, MKR, será considerado separadamente das técnicas de combinação, pois apresentou um desempenho bastante distinto.

A partir desses valores do *CR*, foi construída a Tabela 8.11, que é um resumo da comparação das diversas técnicas. Nessa tabela, são apresentadas as técnicas que obtiveram a melhor partição (partição com melhor *CR*) nas diversas comparações:

- “Estr.”: estrutura conhecida dos dados a que as comparações da respectiva linha se referem.
- “Indiv.”: comparação entre os algoritmos individuais KM, LM, LS e SNN.
- “Melhor ES x MK”: comparação entre as técnicas de combinação que estão sendo comparadas com o MOCLE (MK e ES).
- “Melhor MUH x MUM”: comparação das duas configurações não supervisionadas do MOCLE.
- “Melhor MSH x MSM”: comparação das duas configurações semi-supervisionadas do MOCLE.

Tabela 8.1: CR e k - ds2c2sc13

Técnica	E1 ($k = 2$)				E2 ($k = 5$)				E3 ($k = 13$)			
	CR		k		CR		k		CR		k	
	M	DP	M	DP	M	DP	M	DP	M	DP	M	DP
LM	1	-	2	-	1	-	5	-	0,617	-	19	-
LS	1	-	2	-	1	-	5	-	0,872	-	11	-
SNN	1	-	2	-	1	-	5	-	1	-	13	-
KM	1	0	2	0	0,789	0,04	5	0,9	0,651	0,008	20	4,8
MKR	0,638	0,143	4	2,2	0,735	0,074	5	3,8	0,652	0,094	15	3,8
MK	1	0	2	0	1	0	5	0	0,708	0,024	15	1,8
ES	0,691	0,058	3	0,4	0,992	0,022	5	0,2	0,786	0,042	14	1,1
MUH	1	0	2	0	1	0	5	0	0,777	0	12	0
MUM	1	0	2	0	1	0	5	0	0,777	0	12	0
MSH	1	0	2	0	1	0	5	0	0,777	0	12	0
MSM	1	0	2	0	1	0	5	0	0,777	0	12	0

Tabela 8.2: CR e k - ds3c3sc6

Técnica	E1 ($k = 3$)				E2 ($k = 6$)			
	CR		k		CR		k	
	M	DP	M	DP	M	DP	M	DP
LM	0,789	-	4	-	0,4901	-	4	-
LS	0,525	-	12	-	0,2496	-	11	-
SNN	0,927	-	3	-	0,6181	-	3	-
KM	0,900	0	3	0	0,6014	0,030	4	1,4
MKR	0,577	0,157	2	0,5	0,3536	0,096	2	0,5
MK	0,861	0,072	4	0,7	0,6482	0,059	6	1,1
ES	0,890	0,023	3	0,2	0,5969	0,044	5	1,8
MUH	0,941	0,010	3	0	0,6737	0,017	5	1,0
MUM	0,980	0,014	3	0	0,6740	0,013	3	0,3
MSH	0,962	0,013	3	0	0,6969	0,048	5	1,1
MSM	0,967	0,015	3	0,3	0,6644	0,014	3	0,4

Tabela 8.3: CR e k - **ds4c2sc8**

Técnica	E1 ($k = 2$)				E2 ($k = 8$)			
	CR		k		CR		k	
	M	DP	M	DP	M	DP	M	DP
LM	0,2807	-	8	-	0,831	-	11	-
LS	0,0974	-	16	-	0,019	-	16	-
SNN	0,1606	-	16	-	0,586	-	16	-
KM	0,7943	0,050	9	0	0,816	0,0192	9	0,7
MKR	0,1883	0,024	8	1	0,860	0,0394	9	1,0
MK	0,2359	0,010	7	0,4	0,885	0,0229	8	1,0
ES	0,2958	0,078	5	2,2	0,846	0,0576	9	1,0
MUH	0,3486	0,002	3	0	0,856	0,0258	9	1,3
MUM	0,2876	0,047	6	1,9	0,853	0,0259	10	1,2
MSH	0,3485	0,003	3	0	0,861	0,0278	9	1,2
MSM	0,3180	0,042	5	1,8	0,844	0,0192	10	1,1

Tabela 8.4: CR e k - **spiralsquare**

Técnica	E1 ($k = 2$)				E2 ($k = 6$)			
	CR		k		CR		k	
	M	DP	M	DP	M	DP	M	DP
LM	1	-	2	-	0,66080	-	5	-
LS	1	-	2	-	0,55795	-	12	-
SNN	1	-	2	-	0,99428	-	6	-
KM	1	0	2	0	0,66326	0,008	5	0,3
MKR	1	0	2	0	0,64756	0,056	5	0,6
MK	1	0	2	0	0,66719	0	5	0
ES	1	0	2	0	0,63812	0,062	6	1,0
MUH	1	0	2	0	0,66650	0	5	0
MUM	1	0	2	0	0,66702	0	5	0
MSH	1	0	2	0	0,66650	0	5	0
MSM	1	0	2	0	0,66695	0	5	0

Tabela 8.5: CR e k - glass

Técnica	E1 ($k = 2$)				E2 ($k = 5$)				E3 ($k = 6$)			
	CR		k		CR		k		CR		k	
	M	DP	M	DP	M	DP	M	DP	M	DP	M	DP
LM	0,672	-	9	-	0,5599	-	12	-	0,264	-	12	-
LS	0,171	-	12	-	0,1270	-	12	-	0,040	-	12	-
SNN	0,712	-	6	-	0,5152	-	6	-	0,250	-	6	-
KM	0,632	0,018	3	0,8	0,4752	0,036	3	0,8	0,235	0,015	5	2,8
MKR	0,446	0,104	3	2,1	0,3470	0,074	3	2,1	0,151	0,023	4	2,9
MK	0,556	0,012	3	0,9	0,4405	0,010	4	0,7	0,204	0,013	7	5,0
ES	0,634	0,048	4	1,4	0,4764	0,039	4	1,7	0,224	0,016	4	2,1
MUH	0,672	0	9	0	0,5599	0	12	0	0,264	0	12	0
MUM	0,672	0	9	0	0,5599	0	12	0	0,264	0	12	0
MSH	0,754	0,015	2	0,7	0,5622	0,006	10	3,3	0,274	0,010	4	2,8
MSM	0,714	0,006	6	1,0	0,5618	0,007	11	2,3	0,266	0,009	11	2,3

Tabela 8.6: CR e k - iris

Técnica	E1 ($k = 3$)			
	CR		k	
	M	DP	M	DP
LM	0,586	-	6	-
LS	0,564	-	3	-
SNN	0,823	-	5	-
KM	0,723	0,0071	3	0
MKR	0,580	0,0358	2	0,610
MK	0,816	0,0884	3	0,484
ES	0,759	0,0178	3	0,531
MUH	0,765	0,0184	3	0,305
MUM	0,779	0,0288	3	0,479
MSH	0,823	0	5	0
MSM	0,825	0,0114	5	0,365

Tabela 8.7: CR e k - golub

Técnica	E1 ($k = 2$)				E2 ($k = 3$)			
	CR		k		CR		k	
	M	DP	M	DP	M	DP	M	DP
LM	0,876	-	6	-	0,798	-	8	-
LS	0,078	-	8	-	0,003	-	8	-
SNN	0,855	-	3	-	0,855	-	5	-
KM	0,507	0,211	3	0,9	0,502	0,127	3	1,2
MKR	0,012	0,138	3	2,3	0,262	0,141	3	2,3
MK	0,684	0,059	2	0,6	0,795	0,059	4	0,6
ES	0,743	0,175	4	1,5	0,637	0,129	4	1,8
MUH	0,876	0	6	0	0,855	0	5	0
MUM	0,876	0	6	0	0,855	0	5	0
MSH	0,900	0,023	2	1,0	0,877	0,041	4	0,9
MSM	0,942	0,037	2	0,5	0,857	0,009	5	0,2
Técnica	E3 ($k = 4$)				E4 ($k = 2$)			
	CR		k		CR		k	
	M	DP	M	DP	M	DP	M	DP
LM	0,693	-	7	-	0,057	-	4	-
LS	0,108	-	8	-	0,315	-	8	-
SNN	0,677	-	3	-	0,112	-	8	-
KM	0,402	0,147	3	1,0	0,024	0,015	5	2,3
MKR	-0,015	0,154	3	2,3	-0,070	0,043	3	2,3
MK	0,622	0,055	4	1,8	0,057	0,008	2	0,3
ES	0,589	0,124	4	1,4	0,110	0,074	5	1,8
MUH	0,693	0	7	0	0,315	0	8	0
MUM	0,693	0	7	0	0,315	0	8	0
MSH	0,699	0,015	6	1,9	0,315	0	8	0
MSM	0,723	0,015	2	0,5	0,315	0	8	0

Tabela 8.8: CR e k - proteínas

Técnica	E1 ($k = 4$)				E2 ($k = 27$)			
	CR		k		CR		k	
	M	DP	M	DP	M	DP	M	DP
LM	0,313	-	38	-	0,062	-	38	-
LS	0,066	-	52	-	0,007	-	54	-
SNN	0,300	-	40	-	0,055	-	40	-
KM	0,304	0,023	7	1,0	0,146	0,005	32	9,7
MKR	0,228	0,044	10	4,5	0,044	0,013	10	4,4
MK	0,277	0,010	14	3,5	0,071	0,009	19	4,1
ES	0,315	0,020	11	3,6	0,133	0,006	16	4,5
MUH	0,313	0	38	0	0,143	0	30	0
MUM	0,313	0	38	0	0,143	0	30	0
MSH	0,515	0,077	14	12,7	0,174	0,025	27	12,9
MSM	0,313	0	38	0	0,143	0	30	0

Tabela 8.9: CR e k - leukemia

Técnica	E1 ($k = 3$)				E2 ($k = 7$)			
	CR		k		CR		k	
	M	DP	M	DP	M	DP	M	DP
LM	0,325	-	6	-	0,543	-	12	-
LS	0,020	-	13	-	0,005	-	9	-
SNN	0,044	-	7	-	0,004	-	7	-
KM	0,677	0,062	9	0	0,748	0,021	8	2,1
MKR	0,243	0,034	6	2,5	0,622	0,133	6	2,4
MK	0,413	0	2	0	0,782	0,014	9	1,7
ES	0,315	0,046	7	2,2	0,659	0,053	7	2,2
MUH	0,276	0	8	0	0,770	0	8	0
MUM	0,279	0,003	8	0,2	0,774	0,005	8	0,2
MSH	0,335	0,034	5	1,2	0,775	0,009	7	1,2
MSM	0,410	0,041	4	1,1	0,772	0,002	8	0

Tabela 8.10: CR e k - lung

Técnica	E1 ($k = 4$)			
	CR		k	
	M	DP	M	DP
LM	0,524	-	6	-
LS	0,117	-	8	-
SNN	0,645	-	4	-
KM	0,350	0,043	5	0,9
MKR	0,431	0,052	5	2,1
MK	0,760	0,023	5	0,6
ES	0,438	0,071	5	1,0
MUH	0,645	0	4	0
MUM	0,767	0,079	4	0,3
MSH	0,655	0,019	5	1,6
MSM	0,743	0,089	4	0

- “Melhor MOCLE”: comparação das quatro configurações do MOCLE.
- “Melhor combinação”: comparação das combinações usando o MOCLE com as combinações feitas com as outras técnicas.
- “Melhor de todas”: se refere à(s) técnica(s) de melhor desempenho para aquela estrutura, incluindo as combinações e os algoritmos individuais. Nessa última coluna, estão destacados os casos em que pelo menos uma das configurações do MOCLE teve o melhor desempenho (ou empatou com outras técnicas no primeiro lugar).

Antes de analisar cada uma dessas comparações em detalhes, foi aplicado um teste estatístico para avaliar a significância estatística das diferenças de desempenho observadas. Primeiramente, foi aplicado o teste de Friedman (Hollander and Wolfe 1999; Demšar 2006) para comparar todas as técnicas. Com isso, identificou-se que as técnicas podem ser consideradas estatisticamente diferentes a um nível de significância de 0,05. Aplicando o pós-teste de Nemenyi (Hollander and Wolfe 1999; Demšar 2006), pôde-se observar que a configuração MSH do MOCLE teve um desempenho significativamente superior a todos os algoritmos individuais, ao ES e ao MKR. O MKR, por sua vez, da mesma maneira que o LS, teve um desempenho significativamente pior do que todas as configurações do MOCLE (MUH, MUM, MSH, MSM). De fato, se observados os valores de CR para o MKR em todos os conjuntos de dados e estruturas, pode ser observado que o MKR teve

Tabela 8.11: Comparações entre as técnicas

Conjunto de dados	Estr.	Indiv.	Melhor ES x MK	Melhor MUH x MUM	Melhor MSH x MSM	Melhor MOCLE	Melhor combinação	Melhor de todas
ds2c2sc13	E1	KM, LM, LS, SNN	MK	MUH, MUM	MSH, MSM	MUH, MUM, MSH, MSM	MK, MUH, MUM, MSH, MSM	KM, LM, LS, SNN, MK, MUH, MUM, MSH, MSM
	E2	LM, LS, SNN	MK	MUH, MUM	MSH, MSM	MUH, MUM, MSH, MSM	MK, MUH, MUM, MSH, MSM	LM, LS, SNN, MK, MUH, MUM, MSH, MSM
	E3	SNN	ES	MUH, MUM	MSH, MSM	MUH, MUM, MSH, MSM	ES	SNN
ds3c3sc6	E1	SNN	ES	MUM	MSM	MUM	MUM	MUM
	E2	SNN	MK	MUM	MSH	MSH	MSH	MSH
ds4c2sc8	E1	KM	ES	MUH	MSH	MUH	MUH	KM
	E2	LM	MK	MUH	MSH	MSH	MK	MK
spiralsquare	E1	KM, LM, LS, SNN	MK, ES	MUH, MUM	MSH, MSM	MUH, MUM, MSH, MSM	MK, ES, MUH, MUM, MSH, MSM	KM, LM, LS, SNN, MK, ES, MUH, MUM, MSH, MSM
	E2	SNN	MK	MUM	MSM	MUM	MK	SNN
glass	E1	SNN	ES	MUH, MUM	MSH	MSH	MSH	MSH
	E2	LM	ES	MUH, MUM	MSH	MSH	MSH	MSH
	E3	LM	ES	MUH, MUM	MSH	MSH	MSH	MSH
iris	E1	SNN	MK	MUM	MSM	MSM	MSM	MSM
	E1	LM	ES	MUH, MUM	MSM	MSM	MSM	MSM
	E2	SNN	MK	MUH, MUM	MSH	MSH	MSH	MSH
golub	E3	LM	MK	MUH, MUM	MSM	MSM	MSM	MSM
	E4	LS	ES	MUH, MUM	MSH, MSM	MUH, MUM, MSH, MSM	MUH, MUM, MSH, MSM	LS, MUH, MUM, MSH, MSM
proteinas	E1	LM	ES	MUH, MUM	MSH	MSH	MSH	MSH
	E2	KM	ES	MUH, MUM	MSH	MSH	MSH	MSH
leukemia	E1	KM	MK	MUM	MSM	MSM	MK	KM
	E2	KM	MK	MUM	MSH	MSH	MK	MK
lung	E1	SNN	MK	MUM	MSM	MUM	MUM	MUM

o pior desempenho em relação a todas as combinações em 19 dos 22 casos. Por isso, o MKR não será considerado nas descrições seguintes.

Apesar da falta de evidência estatística da diferença de desempenho entre as técnicas individuais e a maioria das técnicas de combinação, deve ser observado que o desempenho de cada técnica varia bastante dependendo do conjunto de dados e da estrutura considerada. As maiores variações ocorreram entre os algoritmos individuais. Observa-se na Tabela 8.11 que, para cada caso, um algoritmo individual diferente teve o melhor desempenho. Mais do que isso, há uma grande diferença nos valores do CR entre eles. Como pode ser visto nas tabelas que contêm o CR , todos os algoritmos individuais apresentam pelo menos um caso cuja partição tem $CR = 1$ e um outro caso com partição de $CR < 0,3$. Isso ocorre porque, como já discutido, cada conjunto de dados/estrutura está em conformidade com um critério de agrupamento, e portanto algoritmo, diferente.

Considerando esse comportamento dos algoritmos individuais e a baixa qualidade geral do MKR, o teste Friedman foi aplicado novamente, agora para comparar somente as técnicas de combinação. Nesse caso, também pôde ser identificada uma diferença de desempenho das técnicas a um nível de significância de 0,05. Com o pós-teste de Nemenyi pôde-se identificar que a configuração MSH apresenta um desempenho significativamente superior em relação ao ES (já identificada previamente) e também em relação ao MK. Também foram identificadas diferenças significativas no desempenho de duas outras configurações do MOCLE em relação ao ES: o MUM e o MSM.

Com esses testes, não foram encontradas evidências estatísticas para afirmar que o desempenho das demais técnicas foi diferente. Isto é, não é possível afirmar que as configurações MUH, MUM e MSM do MOCLE e o MK apresentam um desempenho diferente. Também não é possível afirmar que o MK e o ES apresentam desempenhos diferentes. Porém, deve ser lembrado que esse teste estatístico considera apenas o *rank* das técnicas nos vários conjuntos de dados e estruturas, não considerando a diferença no valor do CR para determinar a significância das diferenças de desempenho. Pelas tabelas do CR , observa-se que, para vários conjuntos de dados, há uma grande diferença nos valores do CR obtidos com o MOCLE em comparação com os obtidos com o MK e o ES. Os exemplos mais evidente disso podem ser observados nos casos da estrutura E1 do conjunto de dados **ds3c3sc6** (Tabela 8.2), das estruturas E1 e E2 do conjunto **glass** (Tabela 8.5) e das estruturas E1, E2 e E3 do conjunto **golub** (Tabela 8.7). Assim, é importante observar o que ocorreu em cada conjunto de dados/estrutura separadamente. Posteriormente, serão discutidas algumas particularidades observadas para cada conjunto de dados. A seguir, serão feitas observações mais detalhadas a respeito das comparações apresentadas na Tabela 8.11, também considerando os valores obtidos para o CR .

Algumas observações interessantes podem ser feitas a partir da Tabela 8.11 A primeira

delas é que houve diversos casos em que mais de uma técnica obteve o mesmo desempenho. Em cada comparação considerada (cada coluna), pode ser vista a alternância entre as técnicas ou predominância de uma técnica como melhor:

Algoritmos individuais: em cada caso, um algoritmo diferente foi considerado o melhor. Desconsiderando os empates, o algoritmo que encontrou a melhor partição mais vezes foi o SNN (8 vezes). O LM encontrou a melhor partição 6 vezes, o KM 4 vezes e o LS, 1 vez. Esse comportamento era o esperado, pois, como discutido na Seção 5.2, cada algoritmo é mais apropriado para uma determinada conformação dos dados.

MK x ES: apenas em um caso essas técnicas tiveram o mesmo desempenho. Nos demais casos, ambas as técnicas aparecem como melhor praticamente o mesmo número de vezes. Dois fatores influem nessa alternância de qualidade entre essas técnicas. Em primeiro lugar, o MK considera os critérios de compactação e encadeamento, tanto ao gerar a população inicial, quanto ao selecionar as melhores soluções. Partições como algumas das geradas com o SNN não podem ser encontradas somente com esses critérios. Como o *ensemble* considera todas as partições iniciais com o mesmo peso, essas partições boas do SNN influenciam positivamente seu resultado, fazendo-o sobressair ao MK em algumas situações. Por outro lado, o ES não consegue manter o melhor desempenho em todas as situações devido a presença de grande número de partições iniciais de baixa qualidade em outros casos.

MUH x MUM: na maioria dos casos, ambas as configurações não supervisionadas do MOCLE encontraram uma solução de mesma qualidade. Nos demais casos, a configuração MUM se destacou, aparecendo como melhor em 7 casos, contra 2 do MUH.

MSH x MSM: para as configurações semi-supervisionadas, ocorreu o contrário. Na maioria dos casos, apenas uma das configurações se destacou. A configuração MSH aparece 10 vezes em destaque contra 6 vezes da MSM.

MOCLE: excetuando os poucos casos de empate, as configurações semi-supervisionadas predominaram como melhores (13 casos). Apenas em 4 casos, as configurações não supervisionadas foram superiores. Esse comportamento mostra o benefício do uso de conhecimento prévio. Essa questão será discutida em mais detalhes no decorrer do capítulo.

Combinações: dentre todas as combinações, também predominaram as feitas com o MOCLE. O ES se destacou em apenas um caso e o MK em apenas 5, contra 13 destaques do MOCLE.

Todos: o MOCLE continuou em evidência quando as partições individuais foram incluídas nas comparações (12 casos). Porém, nessa comparação, o ES não se destaca em nenhum caso e o MK cai para apenas 2 destaques. Os algoritmos individuais aparecem como o melhor resultado 4 vezes. Nesses casos em que um algoritmo individual se destacou, aparentemente a integração dos dois critérios nas combinações prejudicou a obtenção da melhor estrutura que poderia ser obtida (aquela que foi encontrada pelo individual). Isso ocorreu para a estrutura E1 dos conjuntos `ds4c2sc8` e `leukemia`, em que o melhor resultado foi obtido com o KM, e para a estrutura E3 de `ds2c2sc13` e E2 de `spiralsquare`, em que o SNN apresentou melhor desempenho. Mais estudos seriam necessários para caracterizar o contexto em que a combinação dos dois critérios prejudica a qualidade das partições.

Apesar da utilização dos dois operadores de recombinação terem apresentado bons resultados, o operador que utiliza a técnica HBGF se saiu melhor para o caso semi-supervisionado, ao passo que o operador baseado na técnica MCLA se adequou mais ao contexto não supervisionado. Uma investigação do motivo dessa diferença se faz necessária, porém não será feita nesta Tese.

Considerando as comparações entre as combinações, em todas as vezes que o MK ou o ES apareceram como a melhor técnica, todas, ou ao menos a maioria, das configurações do MOCLE apareceram nas colocações seguintes, e com valores de CR bastante próximos ao do melhor colocado. Ou seja, para os casos em que o MK teve o melhor desempenho, o MOCLE teve um desempenho mais próximo do MK do que o ES, e, para os casos em que o ES foi melhor, o MOCLE também se aproximou mais dele do que o MK. Isso, juntamente com a predominância do MOCLE como melhor técnica, indica que o MOCLE é menos sensível às diferenças de características dos dados do que o MK e o ES, que ora tem o melhor desempenho, ora tem o pior. Ou seja, o MOCLE é mais robusto em relação a diferentes propriedades dos dados do que o MK e o ES, o que contribui para que a meta de diversidade seja atingida.

Essa meta da diversidade foi atingida na medida em que o MOCLE obteve o melhor desempenho na maioria dos conjuntos de dados e também nas várias estruturas que cada um deles possui. Cada conjunto e também cada estrutura tem propriedades distintas, e que cada estrutura está em um nível de refinamento diferente. Além de ter tido o melhor desempenho, o CR do MOCLE, na maioria dos casos, foi de médio a alto (entre 0,65 e 1). Em resumo, com as tabelas apresentadas e as análises feitas, foi mostrado que o MOCLE é capaz de encontrar as partições mais próximas à maioria das estruturas conhecidas, atingindo a meta da diversidade. Isso também mostra que o MOCLE atinge o aspecto “qualidade” da meta concisão e qualidade.

A seguir, serão discutidas algumas particularidades observadas em cada conjunto de

dados. A primeira observação a ser feita é que, para as estruturas mais simples, a maioria das técnicas conseguiu encontrar a solução verdadeira. São os casos das estruturas E1 e E2 do conjunto **ds2c2sc13** e da estrutura E1 do conjunto **spiralsquare**.

Para o conjunto de dados **ds2c2sc13**, todas as configurações tiveram o mesmo desempenho na obtenção das estruturas E1 e E2, com exceção do ES. Por outro lado, o ES foi a técnica de combinação que encontrou a partição mais próxima à estrutura E3. Deve-se observar que, para essa estrutura, dois algoritmos individuais tiveram um desempenho superior a todas as técnicas de combinação: LS e SNN, sendo que o último recuperou a estrutura correta. Ainda assim, é possível notar que o MOCLE encontrou uma mesma partição em todas as configurações e em todas as execuções, e que essas partições têm uma boa qualidade.

No conjunto de dados **ds3c3sc6**, as melhores partições foram obtidas com o MOCLE. Para a estrutura E1, a configuração MUM encontrou uma partição muito próxima da estrutura verdadeira, com uma variação pequena nas várias execuções. Deve ser observado que a informação prévia empregada nas configurações semi-supervisionadas não favoreceu o aparecimento de soluções melhores para essa estrutura. Entretanto, com os resultados para a estrutura E2, observa-se que o conhecimento da estrutura E1, empregado nas configurações semi-supervisionadas, permitiu a obtenção de uma partição de melhor qualidade, quando a recombinação pelo HBGF foi usada (MSH). Além disso, as partições obtidas com MOCLE foram sempre melhores do que as partições obtidas com os algoritmos individuais. Isso significa que novas partições de alta qualidade foram geradas por meio dos operadores de recombinação empregados. Todas as configurações do MOCLE também foram melhores que o MK e o ES.

Para a estrutura E1 do conjunto de dados **ds4c2sc8**, a única técnica que apresentou um bom resultado foi o algoritmo individual KM. Apesar das várias configurações do MOCLE terem superado as demais combinações, o desempenho de todas elas foi bastante baixo. Já para a estrutura E2, todas as combinações tiveram um desempenho bom (CR acima de 0,84) e bastante semelhante, apesar do MK ter apresentado um CR um pouco mais elevado.

No conjunto de dados **spiralsquare**, todas as técnicas encontraram perfeitamente a estrutura E1. Já para a estrutura E2, todas as técnicas obtiveram um desempenho mediano (CR em torno de 0,6) e muito parecido. Observando o conteúdo dessas partições (dados não mostrados), detectou-se que todas as técnicas de combinação falharam em recuperar os dois *clusters* em espiral. O único algoritmo que conseguiu encontrar uma partição bastante próxima à estrutura E2 foi o SNN.

Para o conjunto de dados **glass**, todas as configurações do MOCLE foram superiores ao MK e ao ES em todas as três estruturas. Além disso, também para as três estruturas, as

duas configurações semi-supervisionadas foram superiores às não supervisionadas, sendo a MSH sempre a melhor dentre elas. Na estrutura E3, porém, o desempenho de todas as técnicas foi bastante baixo e estável, indicando que essa estrutura não é consistente com os critérios de agrupamento investigados.

No conjunto de dados *iris*, todas as configurações do MOCLE se saíram melhor que o ES, mas apenas as semi-supervisionadas superaram o MK. Deve-se observar que o algoritmo SNN também obteve uma partição melhor que o MK, mas sem considerar o conhecimento da estrutura como no caso de MSH e MSM. As melhores configurações do MOCLE tiveram um desempenho bom, com um CR próximo de 0,82.

Para o conjunto de dados *golub*, todas as configurações do MOCLE também apresentaram um desempenho superior às demais combinações e superaram ou igualaram o desempenho dos algoritmos individuais, em todas as quatro estruturas conhecidas. Para E1, o MOCLE teve desempenho sempre superior a 0,9, contra um desempenho abaixo de 0,75 das demais combinações. Para E2, o CR do MOCLE passou de 0,85, contra menos de 0,8 nas demais combinações. Para E3, a configuração MSM chegou a 0,723, enquanto o MK ficou em 0,622. Isso tudo ainda é acrescido da maior estabilidade do MOCLE em todos os casos, chegando o desvio padrão a 0 nos casos semi-supervisionados. Deve-se notar que, para a estrutura E4, todas as técnicas tiveram um desempenho ruim. Em E1, E2 e E3, as configurações semi-supervisionadas também superaram as configurações não supervisionadas.

No conjunto de dados *proteinas*, todas as técnicas tiveram um desempenho baixo. Apenas a MSH obteve um desempenho mediano para a estrutura E1. Apesar disso, todas as configurações do MOCLE foram melhores que o ES em E2 e melhor que o MK em ambas as estruturas.

Para o conjunto de dados *leukemia*, a maioria das técnicas teve um desempenho ruim para a estrutura E1. A única exceção foi o algoritmo KM. Para essa estrutura, o MK teve o melhor desempenho entre as combinações. As configurações semi-supervisionadas do MOCLE superaram as não supervisionadas. Para a estrutura E2, o MK teve o melhor desempenho dentre todas as técnicas, porém todas as configurações do MOCLE tiveram um desempenho bastante próximo ao MK.

No conjunto *lung*, o melhor desempenho foi obtido com o MUM, seguido de perto pelo MK. As configurações que empregaram a recombinação via HBGF tiveram um desempenho inferior às que usaram a recombinação com a MCLA.

O desempenho mais baixo do ES para várias estruturas em que alguma das técnicas individuais encontrou uma boa partição ilustra o problema de se considerar todas as partições iniciais ao mesmo tempo na obtenção do consenso. Um grande número de partições iniciais de qualidade baixa influencia negativamente o resultado final. Esse

desempenho mais baixo do ES pôde ser observado para na estrutura E1 de `ds2c2sc13`, E1 e E2 de `ds3c3sc6`, E2 de `spiralsquare`, E1 de `iris`, E1, E2, E3, E4 de `golub`, E1 e E2 de `leukemia` e E1 de `lung`. Nesse sentido, a combinação iterativa do MOCLE foi vantajosa.

Essas análises das particularidades do que ocorreu em cada conjunto de dados confirmam a capacidade do MOCLE em encontrar partições de alta qualidade e em conformidade com diferentes definições de *cluster* e em diferentes níveis de refinamento. Também confirmam a utilidade da semi-supervisão na obtenção de novas estruturas.

Um outro aspecto que pode ser observado nas Tabelas 8.1-8.10 é que, na grande maioria dos casos, os desvios padrão das configurações do MOCLE foram menores do que os das demais combinações. Em vários casos, o desvio padrão foi de zero: todas as configurações em todas as estruturas dos conjuntos de dados `ds2c2sc13` e `spiralsquare`, as configurações não supervisionadas em todas as estruturas para os conjuntos de dados `glass` e `golub`, MSH de `iris`, MUH, MUM e MSM em ambas as estruturas do conjunto `proteinas` e a configuração MUH em todas as estruturas de `leukemia` e `lung`. Isso mostra que o MOCLE é mais estável do que o MK e o ES em relação à presença das partições mais próximas às conhecidas no conjunto de soluções. Ou seja, dada uma configuração de parâmetros, soluções boas, muito parecidas, são obtidas. Portanto, a meta da estabilidade foi atingida.

Observando o que ocorreu com o número de *clusters*, k , dessas partições, vê-se que pelo menos uma configuração do MOCLE obteve partições com número de *clusters* mais próximo do real ou, ao menos, igual ao obtido por outras técnicas, na maioria dos conjuntos de dados/estruturas (E1, E2 e E3 de `ds2c2sc13`, E1 de `ds3c3sc6`, E1 de `ds4c2sc8`, E1 de `spiralsquare`, E1 de `glass`, E1 e E2 de `golub`, E2 de `proteinas` e E1 de `lung`). Mesmo nos casos em que o MOCLE apresentou um valor médio de k mais distante do real, em comparação com as outras combinações, o CR no MOCLE foi maior do que os das combinações com melhor k (E2 de `ds3c3sc6`, E2 de `spiralsquare`, E2 e E3 de `glass`, E1 de `iris`, E3 e E4 de `golub`, E1 de `proteinas` e E2 de `leukemia`). Em apenas 2 casos, nem o CR nem o k no MOCLE foram melhores.

Comparando as configurações não supervisionadas e semi-supervisionadas do MOCLE, pôde-se verificar que, principalmente para os conjuntos de dados reais e de *benchmarks*, a utilização de conhecimento prévio contribuiu para a melhoria da qualidade das soluções. Deve-se notar que o bom desempenho das configurações semi-supervisionadas tem uma utilidade maior quando auxilia na recuperação das estruturas que não foram consideradas como conhecimento prévio. A melhora da qualidade das partições em relação à própria estrutura conhecida não representa uma grande contribuição, afinal, já se sabe da existência dessa estrutura.

Em todos os conjuntos de dados, a estrutura utilizada como conhecimento prévio para o contexto semi-supervisionado foi a estrutura E1. Em relação a essa estrutura, as configurações semi-supervisionadas obtiveram desempenho superior nos seguintes casos: configuração MSH em **ds3c3sc6**, **proteinas** e **lung**, MSM em **ds4c2sc8** e ambas as configurações em **glass**, **iris**, **golub** e **leukemia**.

Em relação às demais estruturas, que é o caso mais relevante, a semi-supervisão contribuiu para a melhora das soluções nos seguintes casos: MSH em **ds3c3sc6**, **ds4c2sc8**, **proteinas** e **leukemia**, e as configurações E2 e E3 dos conjuntos de dados **glass** e **golub**. Além disso, para alguns casos, todas as configurações do MOCLE obtiveram soluções com a mesma qualidade: todas as estruturas de **ds2c2sc13**, E1 de **spiralsquare**, E4 de **golub** e as duas estruturas de **proteinas**, exceto para a configuração MSH que foi superior. Nos conjuntos de dados **ds2c2sc13** e **spiralsquare**, não era esperado melhora com a semi-supervisão, pois a estrutura conhecida é muito clara e todas as técnicas a encontram facilmente. Todos os possíveis refinamentos dessa estrutura vão resultar no mesmo valor para o ganho de informação, que é a medida usada para incorporar o conhecimento prévio no MOCLE semi-supervisionado. Essa qualidade superior das partições obtidas com as configurações semi-supervisionadas mostra que o MOCLE atingiu a meta da semi-supervisão.

Um outro aspecto importante no qual o MOCLE se destaca diz respeito a concisão do conjunto de soluções gerado. A Tabela 8.12 apresenta, para cada conjunto de dados, o número de estruturas que ele possui (n^E), o número de partições iniciais utilizadas para o MOCLE (n^I) e, para as técnicas determinísticas, o número de soluções. Para as demais técnicas, o número médio de soluções obtidas nas 30 execuções. Nesse último caso, o desvio padrão é dado entre parênteses.

Para o ES, foram geradas, manualmente, uma solução para cada valor de k , uma vez que ele combina as soluções iniciais em uma única partição consenso. Com isso, o mesmo número de soluções obtido com os algoritmos KM, LM e LS foi obtido com o ES. Apesar desse número de soluções ser baixo, não faz sentido comparar essas técnicas com o MOCLE e o MOCK em termos de concisão, pois elas, por si mesmas, não geram um conjunto de soluções. Esses valores apenas são mostrados a título de documentar a quantidade de partições empregadas com cada um dos algoritmos.

Pela Tabela 8.12, pode-se observar que o número de soluções geradas pelo MK foi sempre maior do que o número de soluções geradas pelo MOCLE. A única exceção ocorreu com o conjunto de dados **proteinas**, em que o MSH encontrou mais soluções do que o MK. A relação entre o número de soluções geradas por ambas as técnicas varia de acordo com o conjunto de dados. O MK apresentou no mínimo 1,29 e no máximo 10,29 vezes mais soluções que o MOCLE.

Tabela 8.12: Número de partições

	ds2c2sc13	ds3c3sc6	ds4c2sc8	spiralsquare	glass	iris	golub	proteinas	leukemia	lung
n^E	3	2	2	2	3	1	4	2	2	1
n^I	114	42	66	73	48	20	46	170	38	22
LM	25	10	15	11	11	4	7	51	12	5
LS	25	10	15	11	11	4	7	51	12	5
SNN	39	12	21	40	15	8	25	17	2	7
KM	25	10	15	11	11	4	7	51	12	5
MKR	3 (1,0)	1 (0,3)	3 (1,0)	2 (0,5)	3 (0,8)	1 (0,3)	1 (0,4)	2 (1,0)	6 (0,7)	2 (0,5)
MK	81 (9,0)	105 (11,2)	84 (9,0)	98 (13,7)	83 (11,8)	69 (12,2)	79 (10,6)	96 (9,2)	53 (7,9)	87 (12,5)
ES	25 (0,0)	10 (0,0)	15 (0,0)	11 (0,0)	11 (0,0)	4 (0,0)	7 (0,0)	51 (0,0)	12 (0,0)	5 (0,0)
MUH	28 (0,6)	34 (2,0)	23 (1,3)	19 (0,7)	19 (1,2)	7 (1,1)	14 (0,4)	68 (0,6)	13 (0,5)	10 (0,0)
MUM	30 (1,3)	54 (5,7)	38 (4,9)	23 (3,3)	18 (1,4)	7 (1,5)	14 (0,3)	68 (0,2)	23 (3,5)	17 (3,7)
MSH	29 (0,7)	81 (10,2)	23 (1,4)	19 (0,9)	47 (8,9)	15 (3,1)	19 (1,6)	121 (16,2)	39 (16,9)	27 (7,4)
MSM	30 (0,2)	74 (13,2)	39 (5,1)	23 (3,2)	36 (7,2)	12 (1,7)	20 (1,6)	74 (6,0)	21 (2,6)	18 (6,1)

O número de soluções recomendadas pelo MOCK como as melhores do fronte de Pareto, MKR, é, em geral, próximo do número de estruturas conhecidas de cada conjunto de dados. Porém, como já foi discutido anteriormente, essas soluções correspondem aos piores resultados em termos de semelhança com as estruturas conhecidas.

Em geral, o MOCLE reduziu bastante o número de partições do conjunto inicial, n^I . Em apenas 5 casos isso não ocorreu. Houve casos em que o MOCLE apresentou uma redução de aproximadamente 1/4 no número de soluções iniciais.

Comparando as várias configurações do MOCLE, pode-se observar que as configurações semi-supervisionadas obtiveram, na maioria dos casos, um número mais elevado (e também mais variado) de partições. Isso se deve a presença do terceiro objetivo, ganho de informação.

Com isso, vê-se que, mesmo nos piores casos, o MOCLE se mostrou bem mais conciso do que o MK. Isso, juntamente com o lado da “qualidade” das soluções já discutido, completa a meta da concisão e qualidade.

8.3 Análise Detalhada

Para a análise detalhada, foi escolhido um conjunto de soluções Π_S para cada técnica. Para cada algoritmo individual, o conjunto de soluções escolhido corresponde às soluções por ele obtidas, que foram utilizadas para compor a população inicial do MOCLE. Para cada técnica de combinação, foi selecionado um conjunto de soluções gerado em uma das 30 execuções. Mais especificamente, dentre os 30 conjuntos gerados, foi selecionado aquele que possuía a partição com maior CR em relação à uma das estruturas mais refinadas do conjunto de dados. Para os conjuntos que possuem uma única estrutura, ela foi a considerada. Para os que possuem duas, a estrutura considerada foi sempre E2. Para os dados com mais de duas estruturas, foram feitas algumas considerações especiais para a escolha da estrutura. Para o conjunto **ds2c2sc13**, foi considerada a estrutura E3, pois é a mais refinada e as técnicas continuam tendo um desempenho razoável em recuperá-la. Para o conjunto **glass**, a estrutura mais refinada é E3, porém todas as técnicas têm um desempenho muito baixo em recuperá-la. Assim, a estrutura E2 foi considerada nas análises. Para o conjunto de dados **golub**, sabe-se que a estrutura E2 é um refinamento de E1, mas as demais estruturas não tem relação com as duas primeiras. Com isso, a estrutura considerada para esse conjunto foi E2. Para o caso do MKR, foram usadas as soluções recomendadas na execução do MK selecionada.

Inicialmente, será investigado o conteúdo dos conjuntos de soluções gerados pelo MOCLE, por meio da análise da origem de suas partições. As partições podem ter se originado de um dos algoritmos iniciais, ou ser uma solução nova, gerada durante o processo de evolução. No primeiro caso, a partição já aparecia na população inicial e foi preservada na

população durante todo o processo de evolução. O segundo caso representa as partições geradas por meio da combinação iterativa das partições realizada pelo MOCLE.

Em seguida, as soluções das diversas técnicas serão comparadas por meio de dois tipos de gráficos. O primeiro mostra o valor de CR por k . Esse tipo de gráfico permite observar as diferenças de comportamento das técnicas em relação à recuperação da estrutura considerada.

O segundo tipo de gráfico mostra a variância e a conectividade das soluções. Para o MK e as configurações não supervisionadas do MOCLE, os pontos no gráfico representam o fronte de Pareto obtido. Nesse tipo de gráfico são representados os valores normalizados de maneira similar a utilizada em (Handl and Knowles 2006a). Inicialmente foram encontrados os valores mínimo e máximo de ambas as medidas em relação a todos os conjuntos de soluções considerados. Em seguida, seus valores foram transformados para o intervalo $[0, 1]$. Posteriormente, foi extraída a raiz quadrada de cada valor. Segundo Handl and Knowles (2006a), devido ao comportamento não linear da variância e da conectividade em relação a k (a variância diminui muito rapidamente para os primeiros valores de k , com mudanças bem menos marcadas para os valores maiores de k , ocorrendo o contrário para a conectividade), os intervalos dos valores dos objetivos são amostrados de forma desigual, o que aparece no gráfico do fronte de Pareto como uma densidade elevada de pontos nas extremidades e muito poucas soluções no centro. A aplicação da raiz quadrada reduz esse efeito (Handl and Knowles 2006a).

As tabelas 8.13 e 8.14 mostram, para cada conjunto de dados, o número de partições do conjunto de soluções originalmente geradas com cada algoritmo inicial, em cada configuração do MOCLE, o número de soluções novas, geradas pela combinação durante o processo de evolução, o total de soluções provenientes dos algoritmos individuais, o total de soluções considerando também as novas e o número de partições que apareceram idênticas em todas as quatro configurações do MOCLE. As colunas LM, LS, SNN e KM representam as partições obtidas unicamente com o respectivo algoritmo. Os casos em que mais de um algoritmo inicial identificou as mesmas soluções são contados nas colunas identificadas por mais de um algoritmo.

A permanência de partições iniciais no conjunto de soluções, conforme observado nas Tabelas 8.13 e 8.14, mostra que o MOCLE realmente faz uma seleção automática de modelos. Como os algoritmos individuais geram partições de alta qualidade em relação ao critério de agrupamento que eles otimizam e também em vários níveis de refinamento, partições que faziam parte da população inicial se mantiveram no conjunto de soluções, na medida em que satisfaziam diferentes compromissos dos objetivos utilizados. Nesse processo de seleção, estão embutidos a escolha do(s) critério(s) de agrupamento e dos valores dos parâmetros que mais se adequam aos vários compromisso entre a conectividade,

Tabela 8.13: Origem das soluções - Conjuntos artificiais

Conjunto de dados	Configuração	LM	LS	SNN	KM	LM, SNN	SNN, KM	LM, LS, SNN	LM, LS, SNN, KM	Novas	Total iniciais	Total	Idênticas
ds2c2sc13	MUH	20	3	1	0	1	0	1	2	2	28	30	28
	MUM	20	3	1	0	1	0	1	2	2	28	30	
	MSH	20	3	1	0	1	0	1	2	1	28	29	
	MSM	20	3	1	0	1	0	1	2	2	28	30	
ds3c3sc6	MUH	7	1	4	8	0	0	0	0	14	20	34	19
	MUM	6	2	7	8	0	0	0	0	40	23	63	
	MSH	7	4	7	9	0	0	0	0	48	27	75	
	MSM	6	4	8	9	0	0	0	0	64	27	91	
ds4c2sc8	MUH	8	1	6	4	0	0	0	0	5	19	24	13
	MUM	6	0	2	4	0	0	0	0	20	12	32	
	MSH	8	1	6	4	0	0	0	0	3	19	22	
	MSM	6	0	3	4	0	0	0	0	22	13	35	
spiralsquare	MUH	8	4	1	3	0	1	0	1	0	18	18	14
	MUM	5	4	0	3	0	1	0	1	9	14	23	
	MSH	8	4	1	3	0	1	0	1	1	18	19	
	MSM	6	4	0	3	0	1	0	1	12	15	27	

Tabela 8.14: Origem das soluções - Conjuntos de *benchmarks* e reais

Conjunto de dados	Configuração	LM	LS	SNN	KM	LM, LS	LM, KM	LS, SNN	Novas	Total iniciais	Total	Idênticas
glass	MUH	9	3	0	2	2	0	0	5	16	21	16
	MUM	9	3	0	3	2	0	0	2	17	19	
	MSH	9	3	3	5	2	0	0	29	22	51	
	MSM	9	5	3	5	2	0	0	5	24	29	
iris	MUH	2	0	5	1	0	0	0	2	8	10	6
	MUM	3	0	5	1	0	0	0	1	9	10	
	MSH	1	0	6	1	0	0	0	5	8	13	
	MSM	1	0	6	2	0	0	0	5	9	14	
golub	MUH	4	4	4	1	0	0	1	1	14	15	14
	MUM	4	4	4	1	0	0	1	1	14	15	
	MSH	4	4	6	1	0	0	1	1	16	17	
	MSM	4	4	7	1	0	0	1	5	17	22	
proteinas	MUH	49	8	1	9	0	1	0	0	68	68	66
	MUM	49	8	1	9	0	1	0	0	68	68	
	MSH	49	8	1	13	0	1	0	63	72	135	
	MSM	49	6	1	13	0	1	0	2	70	72	
leukemia	MUH	1	0	0	12	0	0	0	1	13	14	9
	MUM	1	0	0	9	0	0	0	16	10	26	
	MSH	2	0	0	11	0	0	0	27	13	40	
	MSM	1	0	0	10	0	0	0	12	11	23	
lung	MUH	5	1	1	3	0	0	0	0	10	10	10
	MUM	5	1	1	4	0	0	0	6	11	17	
	MSH	5	1	1	3	0	0	0	25	10	35	
	MSM	5	1	1	4	0	0	0	13	11	24	

a variância e, quando é o caso, o ganho de informação.

O número de soluções que apareceram idênticas nas quatro configurações do MOCLE foi bastante próximo ao número de partições provenientes das partições iniciais. Como poucas partições novas foram geradas nas configurações não supervisionadas, pode-se concluir que, em geral, as partições iniciais de alta qualidade são mantidas no conjunto de soluções, independentemente do operador de recombinação utilizado e da consideração ou não de conhecimento prévio (contexto semi-supervisionado ou não supervisionado). As partições iniciais mantidas nos conjuntos de soluções do MOCLE poderão ser melhor observadas nos gráficos que serão apresentados na Seção 8.3.1.

As soluções novas apareceram em maior número nas configurações semi-supervisionadas. Aparentemente, as soluções geradas com a combinação não tem uma variância ou conectividade melhor do que as iniciais, mas possuem *clusters* mais puros em relação à estrutura conhecida, ou seja, tem um ganho de informação melhor.

Pode-se observar que, para cada conjunto de dados, um algoritmo individual diferente contribuiu com maior número de soluções. Isso era o esperado, pois cada conjunto de dados pode estar mais em conformidade com um critério de agrupamento diferente.

A seguir, são apresentados os gráficos que permitem analisar essas características em mais detalhes. Como a recombinação com a técnica HBGF teve um melhor desempenho no contexto semi-supervisionado e a recombinação com a técnica MCLA para o contexto não supervisionado, como visto na Seção 8.2, decidiu-se mostrar apenas as configurações MUM e MSH nos gráficos a seguir. Na Seção 8.3.1, o MOCLE é comparado com os algoritmos individuais e na Seção 8.3.2 com as demais técnicas de combinação (ES e MK - incluindo suas recomendações).

8.3.1 Comparação do MOCLE com os Algoritmos Individuais

Os gráficos das Figuras 8.1-8.10 comparam as duas configurações do MOCLE mencionadas e os algoritmos individuais, em termos da variância e conectividade de seus conjuntos de soluções. Cada gráfico corresponde a um conjunto de dados.

Nesses gráficos podem ser observadas as diferenças de qualidade das soluções obtidas pelos algoritmos individuais, em relação às duas medidas. O principal aspecto que esses gráficos mostram é que o MOCLE de fato seleciona as melhores soluções dentre as geradas com os algoritmos individuais. Mais precisamente, é possível observar de que algoritmo inicial cada uma das soluções do MOCLE se originou, quando elas são provenientes de um deles, na medida em que os símbolos que representam cada técnica se sobrepõem. Por exemplo, um $\times$ (em vermelho), que representa o MUM, sobreposto a um $\square$ (SNN, em preto), indica que aquela solução do MUM foi obtida originalmente pelo SNN e selecionada para permanecer no conjunto de soluções do MUM, durante o processo de evolução.

As soluções novas são também distinguidas facilmente, uma vez que os símbolos do

Figura 8.1: Comparação do MOCLE com os algoritmos individuais - ds2c2sc13

Figura 8.2: Comparação do MOCLE com os algoritmos individuais - ds3c3sc6

Figura 8.3: Comparação do MOCLE com os algoritmos individuais - ds4c2sc8

Figura 8.4: Comparação do MOCLE com os algoritmos individuais - spiralsquare

Figura 8.5: Comparação do MOCLE com os algoritmos individuais - glass

Figura 8.6: Comparação do MOCLE com os algoritmos individuais - iris

Figura 8.7: Comparação do MOCLE com os algoritmos individuais - golub

Figura 8.8: Comparação do MOCLE com os algoritmos individuais - proteínas

Figura 8.9: Comparação do MOCLE com os algoritmos individuais - leukemia

Figura 8.10: Comparação do MOCLE com os algoritmos individuais - lung

MOCLE (MUM e MSH) não ficam sobrepostos aos símbolos dos algoritmos individuais.

Há alguns casos em que podem ser vistas várias soluções novas do MSH (triângulos azuis) que não representam melhoras na variância ou conectividade. Essas partições possuem uma melhor qualidade em relação ao terceiro objetivo, o ganho de informação, que não está representado nos gráficos. Isso pode ser identificado claramente nas Figuras 8.2, 8.5, 8.8, 8.9 e 8.10 como os vários triângulos azuis em uma determinada região do gráfico.

Para o conjunto de dados **ds2c2sc13**, as soluções do MOCLE, nas duas configurações, estão bem distribuídas ao longo dos valores dos dois objetivos. Além disso, a grande maioria das soluções do MUM e MSH são iguais. Observa-se ainda que, do grande número de partições dos algoritmos iniciais, apenas aquelas com os melhores compromissos entre a conectividade e a variância permaneceram nos conjuntos de soluções de ambas as configurações do MOCLE.

No caso do conjunto **ds3c3sc6**, nota-se a presença de diversas soluções novas tanto para o MUM quanto para o MSH. Entretanto, em cada caso, as soluções estão situadas em uma região diferente. No caso do MUM, elas estão em uma região com valores intermediários de ambos os objetivos. Já para o MSH, elas estão em uma região de conectividade mais elevada, representando os casos em que a vantagem para elas permanecerem no conjunto de soluções está no objetivo ganho de informação. Há também um conjunto de algumas poucas soluções de variância próximas de 1 e com conectividade baixa, que foram provenientes dos algoritmos SNN e LS.

Para o conjunto **ds4c2sc8**, não há soluções com conectividade muito elevada. Também há várias soluções novas na região de valores intermediários dos dois objetivos. As poucas soluções novas do MSH estão na mesma região das soluções novas do MUM, porém com valores dos objetivos um pouco acima das primeiras.

As soluções obtidas para o conjunto de dados **spiralsquare** estão todas concentradas na região com conectividade abaixo de 0,5. A maioria das soluções do MOCLE aparecem em ambas as configurações. Também há poucas soluções novas.

Para o conjunto **glass**, as soluções também estão concentradas na região de conectividade mais baixa. Neste caso, porém, podem ser observadas diversas soluções do MSH que são novas e que não foram identificadas por MUM. Essas soluções apresentam uma variância mais elevada do que outras identificadas com os mesmos valores de conectividade pelas duas configurações.

No caso do conjunto **iris**, todas as soluções possuem valores medianos de conectividade e de variância. Nota-se também a presença de poucas soluções novas no caso do MSH.

Para o conjunto de dados **golub**, todas as soluções apresentam uma variância bastante elevada, estando distribuídas nos valores intermediários da conectividade. Poucas soluções

novas podem ser identificadas.

No gráfico do conjunto de dados **proteinas**, é possível identificar quatro regiões distintas em que se concentram as soluções. A primeira tem uma conectividade baixa e variância bastante elevada, sendo as soluções provenientes, em sua maioria, do LS. A segunda região apresenta variâncias intermediárias e conectividades de baixas a intermediárias, com soluções provenientes do algoritmo LM. A terceira região corresponde a soluções de variância baixa e conectividade elevada, contendo soluções geradas com o KM. Nessas três regiões são observadas soluções das duas configurações, MUM e MSH. Na quarta região são observadas somente soluções novas do MSH, com variâncias e conectividades mais elevadas (são as soluções com melhor ganho de informação).

Para o conjunto de dados **leukemia**, observa-se que as soluções estão bem distribuídas pelos valores dos objetivos, mas não chegam a atingir os valores mais elevados de ambos. Podem ser observadas soluções novas de ambas as configurações do MOCLE, porém elas são bem distintas. Enquanto as do MUM representam diferentes compromissos da variância e da conectividade, as do MSH estão relacionadas a um maior ganho de informação, visto o seu posicionamento no gráfico.

As soluções do conjunto **lung** possuem uma variância de média a alta. Tanto MUM quanto MSH apresentam soluções novas, mas, como nos conjuntos **ds3c3sc6** e **leukemia**, para cada configuração, essas soluções estão em uma região diferente. No MUM, elas têm conectividades e variâncias intermediárias, enquanto no MSH, têm conectividades mais elevadas.

Esse tipo de gráfico, que mostra o valor dos objetivos, constitui também um aliado do especialista em suas análises, pois permite a visualização do comportamento das partições em relação aos objetivos. Com isso, o especialista pode optar por detalhar suas análises escolhendo as partições com as características que mais lhes interesse: partições mais compactas (variância mais baixa), partições cujos *clusters* possuem objetos bem conectados (conectividade baixa), ou partições com *clusters* mais puros em relação a uma estrutura conhecida (nos gráficos apresentados, foram as partições do MSH que se destacaram das demais por estarem fora da região do fronte de Pareto do MUM). Os valores dos objetivos servem como informações complementares para o especialista, e os gráficos o ajudam a visualizar o que ocorre com esses valores nas várias partições que ele precisa analisar. É nesse aspecto, da disponibilidade desses valores dos objetivos, que o MOCLE atinge a meta das informações complementares.

Embora as informações complementares estejam disponíveis e possam ser visualizadas por meio dos gráficos, o potencial de sua utilização ainda requer investigações adicionais, que não serão feitas neste trabalho. No MOCK, essas informações são empregadas para extrair um conjunto de soluções menor, recomendadas como as melhores do fronte de

Pareto obtido. Contudo, nos experimentos realizados neste trabalho, observou-se que essas recomendações não correspondem às partições de melhor qualidade que o MOCK consegue gerar no que diz respeito a recuperar as estruturas existentes nos dados.

8.3.2 Comparação do MOCLE com as Outras Técnicas de Combinação

Nos gráficos das Figuras 8.11-8.20 são comparados os conjuntos de soluções das configurações MUM e MSH do MOCLE com o conjunto de soluções do ES, do MK e seu conjunto MKR correspondente. O primeiro gráfico de cada figura mostra o *CR* das soluções com a variação de k . O segundo gráfico das figuras mostra a variância *versus* a conectividade.

O primeiro aspecto que chama a atenção nos gráficos da variância *versus* conectividade é que MOCLE e MK apresentam soluções com os melhores compromissos em diferentes regiões do espaço de objetivos. Existem inclusive casos em que os dois se assemelham. Entretanto, na maioria dos casos, ora um, ora outro, se sobressai. Um exemplo pode ser observado no gráfico da Figura 8.11(b). Na região de baixa conectividade e alta variância, as mesmas soluções foram encontradas nas duas configurações do MOCLE e no MK. Nas regiões de valores intermediários de ambos os objetivos, o MK apresentou os melhores compromissos. Na região de conectividade elevada e baixa variância, as soluções do MOCLE se destacaram.

Considerando os gráficos de todos os conjuntos de dados, pode-se verificar que, de forma geral, o MK se destaca “mais vezes” como sendo o melhor fronte de Pareto, ou seja, na maioria das vezes, as soluções do MK apresentaram um melhor compromisso dos objetivos. Isso se deve a exploração mais minuciosa do espaço de objetivos realizada pelo MK por meio de seus operadores de recombinação e mutação. Se, por um lado, isso faz com que, em vários casos, o MK apresente um fronte de Pareto melhor, ou com uma extensão maior de soluções com bons compromissos, por outro lado, gera um número muito elevado de soluções, como observado na Seção 8.2.

Além disso, as soluções com os melhores compromissos nem sempre correspondem às soluções mais próximas das estruturas conhecidas. Pelos resultados obtidos neste trabalho, vê-se que não é necessária uma exploração muito minuciosa do espaço de objetivos para se chegar a soluções de qualidade elevada em relação às estruturas presentes nos dados. Isso ocorre porque, em geral, os dados não têm uma conformação ideal em relação a um ou outro objetivo. Isso fica bastante evidente nos gráficos do conjunto de dados **glass**, na Figura 8.15. Nesse caso, pelo gráfico 8.15(b) é bastante evidente a melhor qualidade do fronte de Pareto do MK. Porém, ao se observar o gráfico 8.15(a), verifica-se que, na grande maioria dos casos, o MUM e o MSH obtiveram os melhores desempenhos ao recuperar a estrutura E2.

Apesar do número muito elevado de soluções do MK, é possível observar nos gráficos que muitas dessas soluções são bastante parecidas, tanto em termos dos valores para os

(a) $CR \times k$

(b) Variância x Conectividade

Figura 8.11: Comparação das combinações - ds2c2sc13

(a) $CR \times k$

(b) Variância x Conectividade

Figura 8.12: Comparação das combinações - ds3c3sc6

(a) $CR \times k$

(b) Variância x Conectividade

Figura 8.13: Comparação das combinações - ds4c2sc8

(a) $CR \times k$

(b) Variância x Conectividade

Figura 8.14: Comparação das combinações - spiralsquare

(a) $CR \times k$

(b) Variância x Conectividade

Figura 8.15: Comparação das combinações - glass

(a) $CR \times k$

(b) Variância x Conectividade

Figura 8.16: Comparação das combinações - iris

(a) $CR \times k$

(b) Variância x Conectividade

Figura 8.17: Comparação das combinações - golub

(a) $CR \times k$

(b) Variância x Conectividade

Figura 8.18: Comparação das combinações - proteínas

(a) $CR \times k$

(b) Variância x Conectividade

Figura 8.19: Comparação das combinações - leukemia

(a) $CR \times k$

(b) Variância x Conectividade

Figura 8.20: Comparação das combinações - lung

objetivos, quanto em termos do CR .

Quanto ao ES, pode-se observar que, em termos dos objetivos, suas soluções sempre apresentam os piores valores. Contudo, nos gráficos do CR podem ser identificados vários casos em que as soluções do ES apresentam bom desempenho. Exemplos claros disso podem ser vistos nos gráficos 8.11(a), 8.13(a), 8.14(a), 8.15(a) e 8.17(a).

Uma outra observação interessante a ser feita é que, em muitos casos, o MOCLE se aproximou mais das estruturas conhecidas no caso das partições com k mais próximo do verdadeiro. Exemplos claros podem ser observados nos gráficos das Figuras 8.11(a), 8.12(a), 8.15(a), 8.17(a) e 8.18(a).

Em relação a contribuição da semi-supervisão na qualidade das soluções obtidas, há vários casos a serem observados. O mais evidente é o conjunto de dados `ds3c3sc6`, representado nos gráficos da Figura 8.12. No gráfico 8.12(a), pode-se ver claramente a melhora de qualidade das soluções obtidas com o MSH em relação ao MUM para $k = 5$ e $k = 6$ (lembrando que $K^{E2} = 6$). Para $k = 5$, os CR s das soluções de MUM estão, na maioria das vezes, entre 0,4 e 0,6, enquanto os de MSH variam de 0,6 a mais de 0,8. Para $k = 6$, MUM tem somente soluções com $k \leq 6$, ao passo que MSH chega a ter uma solução com $CR > 0,7$. Outros gráficos em que podem ser observadas soluções melhores de MSH em relação a MUM são 8.13(a), 8.15(a), 8.16(a) ($k = 5$), 8.17(a) ($k = 3$), 8.18(a) e 8.19(a). Dentre esses, o caso do conjunto de dados `golub` (8.17(a)) merece uma atenção especial, uma vez que o MSH conseguiu encontrar uma partição que corresponde perfeitamente à estrutura E2 ($CR = 1$).

8.4 Aplicação da Visualização

Para ilustrar a aplicação do método de visualização, foram selecionados dois dos conjuntos de soluções gerados para o conjunto de dados `golub`. Esse conjunto de dados foi escolhido por várias razões. Primeiro, pelos bons resultados alcançados com a combinação, sendo ele um conjunto de dados real para o qual se conhece mais de uma estrutura, ao menos uma delas um refinamento de outra. Esses dados também representam um caso típico de aplicação real que pode se beneficiar da semi-supervisão e do método de visualização para a descoberta de subclasses. Em terceiro lugar, esse conjunto de dados possui poucos objetos, o que facilita a sua discussão neste trabalho.

Antes de iniciar as análises, é preciso mencionar que, devido a dificuldade na perfeita distinção de intensidades muito próximas de algumas das cores, para as visualizações mostradas aqui foram consideradas apenas quatro intensidades diferentes. Para os casos em que um *cluster* conhecido foi subdividido em mais do que quatro subgrupos, as três intensidades mais fortes foram associadas aos *clusters* com mais objetos, e a quarta intensidade foi associada a todos os outros *clusters* menores. Esses *clusters* menores podem

ainda ser diferenciados, pois o rótulo de *cluster* de cada um deles também é representado na visualização.

O primeiro aspecto importante a ser mostrado a respeito da visualização é como o método auxilia na descoberta de subclasses. Como ilustração, foi selecionado o conjunto de soluções gerado com o MSH para o conjunto de dados `golub`. O método de visualização foi aplicado a esse conjunto de soluções utilizando o conhecimento prévio da estrutura E1. Além de ser usada na visualização, essa estrutura conhecida também foi utilizada para gerar o conjunto de soluções que será visualizado (configuração MSH). Lembrando, essa estrutura divide os objetos em dois tipos de leucemia: ALL e AML. Porém, sabe-se também que o tipo ALL pode ser subdividido em dois sub-grupos: B-ALL e T-ALL (estrutura E2), como discutido no Capítulo 7.

A Tabela 8.15 é a visualização das soluções do MSH para o conjunto `golub`, considerando a estrutura E1. Nessa tabela, pode-se observar que várias das partições separam os objetos ALL dos AML: a maioria dos objetos ALL estão representados em **verde** e a maioria dos objetos AML estão representados em **laranja** nas partições π^3 , π^4 , π^5 , π^7 , π^{11} , π^{12} , π^{14} , π^{16} e π^{17} . Ou seja, a estrutura E1 é facilmente identificada em várias partições.

Mais importante do que identificar E1, é identificar E2, sem considerar o conhecimento de sua existência. Conforme dito no Capítulo 6, a identificação de subdivisões de um *cluster* conhecido (chamado de classe no Capítulo 6) é percebida pela existência de mais de uma intensidade na sua cor. Na Tabela 8.15, pode-se observar a divisão dos objetos ALL em dois *clusters* distintos, em algumas das partições (π^2 , π^3 , π^7 , π^{11} e π^{17}). Nesses casos, os objetos ALL estão representados em duas intensidades principais de **verde**. Observando os rótulos dos objetos, verifica-se que a grande maioria dos objetos no **verde** mais escuro é do subtipo B-ALL, enquanto que os objetos representados com a segunda intensidade de **verde**, um pouco mais claro, são do tipo T-ALL. Vale mencionar que os rótulos dos objetos foram compostos dessa maneira de modo a facilitar a avaliação dos resultados da visualização neste trabalho, mas as informações contidas neles não foram consideradas ao se gerar a visualização. Portanto, a subdivisão do tipo de leucemia ALL nos subtipos B-ALL e T-ALL pôde ser obtida com o MOCLE, e claramente identificada com o método de visualização, com o auxílio apenas da divisão dos objetos nos tipos conhecidos, ALL e AML.

Pode-se notar ainda nessa visualização que alguns objetos do tipo ALL foram isolados dos demais desse tipo em diversas partições, e que a partição π^3 corresponde exatamente à estrutura E2.

Para ilustrar outros aspectos do método de visualização, foi utilizado o conjunto de soluções gerado com o algoritmo inicial LM. Como esse algoritmo é hierárquico, o processo

8.4 Aplicação da Visualização

Tabela 8.15: Visualização para o MSH considerando a estrutura E1

Objeto	π^1	π^2	π^3	π^4	π^5	π^6	π^7	π^8	π^9	π^{10}	π^{11}	π^{12}	π^{13}	π^{14}	π^{15}	π^{16}	π^{17}	
ALL-Bcell-BM-DFCI-1	1	1	1	2	1	1	1	1	1	1	1	1	1	1	1	1	7	1
ALL-Bcell-BM-DFCI-12	1	1	1	2	1	1	1	1	1	1	1	1	1	1	1	1	3	1
ALL-Bcell-BM-DFCI-13	1	1	1	2	1	1	1	1	1	1	1	1	1	1	1	1	1	1
ALL-Bcell-BM-DFCI-15	1	1	1	2	1	1	1	1	1	1	1	1	1	1	1	1	1	1
ALL-Bcell-BM-DFCI-16	1	1	1	2	1	1	1	1	1	1	1	1	1	1	1	1	7	1
ALL-Bcell-BM-DFCI-17	1	1	1	2	1	1	2	2	2	1	2	2	2	2	2	2	1	3
ALL-Bcell-BM-DFCI-18	1	1	1	2	1	1	1	1	1	1	1	1	1	1	1	1	7	1
ALL-Bcell-BM-DFCI-19	1	1	1	2	1	1	1	1	1	1	1	1	1	1	1	1	3	1
ALL-Bcell-BM-DFCI-20	1	1	1	2	1	1	1	1	3	1	3	1	3	3	3	3	1	4
ALL-Bcell-BM-DFCI-21	1	2	2	2	2	2	3	3	4	2	4	3	4	4	4	4	8	5
ALL-Bcell-BM-DFCI-22	1	1	1	2	1	1	1	1	1	1	1	1	1	1	1	1	4	1
ALL-Bcell-BM-DFCI-24	1	1	1	2	1	1	1	1	1	1	1	1	1	1	1	1	1	1
ALL-Bcell-BM-DFCI-25	1	1	1	2	1	1	1	1	1	1	1	1	1	1	1	1	3	1
ALL-Bcell-BM-DFCI-26	1	1	1	2	1	1	1	1	1	1	1	1	1	1	1	1	7	1
ALL-Bcell-BM-DFCI-27	1	1	1	2	1	1	1	1	1	1	1	1	1	1	1	1	4	1
ALL-Bcell-BM-DFCI-39	1	1	1	2	1	1	1	1	1	1	1	1	1	1	1	1	7	1
ALL-Bcell-BM-DFCI-4	1	1	1	2	1	1	1	1	1	1	1	1	1	1	1	1	4	1
ALL-Bcell-BM-DFCI-40	1	1	1	2	1	1	1	1	1	1	1	1	1	1	1	1	4	1
ALL-Bcell-BM-DFCI-41	1	1	1	2	1	1	1	1	1	1	4	1	1	1	1	1	2	1
ALL-Bcell-BM-DFCI-42	1	1	1	2	1	1	1	1	1	1	1	1	1	1	1	1	3	1
ALL-Bcell-BM-DFCI-43	1	1	1	2	1	1	1	1	1	1	1	1	1	1	1	1	3	1
ALL-Bcell-BM-DFCI-44	1	1	1	2	1	1	1	1	1	1	1	1	1	1	1	1	3	1
ALL-Bcell-BM-DFCI-45	1	1	1	2	1	1	1	1	1	1	1	1	1	1	1	1	3	1
ALL-Bcell-BM-DFCI-46	1	1	1	2	1	1	1	1	1	1	1	1	1	1	1	1	3	1
ALL-Bcell-BM-DFCI-47	1	1	1	2	1	1	1	1	1	1	1	1	1	1	1	1	2	1
ALL-Bcell-BM-DFCI-48	1	1	1	2	1	1	1	1	1	1	1	1	1	1	1	1	1	1
ALL-Bcell-BM-DFCI-49	1	1	1	2	1	1	1	1	1	1	1	1	1	1	1	1	7	1
ALL-Bcell-BM-DFCI-5	1	1	1	2	1	1	1	1	1	1	1	1	1	1	1	1	1	1
ALL-Bcell-BM-DFCI-7	1	1	1	2	1	1	1	1	1	1	1	1	1	1	1	1	4	1
ALL-Bcell-BM-DFCI-8	1	1	1	2	1	1	1	1	1	1	1	1	1	1	1	1	4	1
ALL-Bcell-BM-StJude-55	1	1	1	2	1	1	1	1	1	1	1	1	1	1	1	1	2	1
ALL-Bcell-BM-StJude-56	1	1	1	2	1	1	1	1	1	1	1	1	1	1	1	1	7	1
ALL-Bcell-BM-StJude-59	1	1	1	2	1	1	1	1	1	1	1	1	1	1	1	1	3	1
ALL-Bcell-PB-DFCI-68	1	1	1	2	1	1	1	1	1	1	1	1	1	1	1	1	1	1
ALL-Bcell-PB-DFCI-69	1	1	1	2	1	1	1	1	1	1	1	1	1	1	1	1	1	1
ALL-Bcell-PB-DFCI-70	1	1	1	2	1	1	1	1	1	1	1	1	1	1	1	1	3	1
ALL-Bcell-PB-DFCI-71	1	1	1	2	1	1	1	1	1	1	1	1	1	1	1	1	7	1
ALL-Bcell-PB-DFCI-72	1	1	1	2	1	1	1	1	1	1	1	1	1	1	1	1	7	1
ALL-Tcell-BM-DFCI-10	1	1	3	3	1	1	1	4	1	1	1	5	1	1	1	1	8	2
ALL-Tcell-BM-DFCI-11	1	1	3	3	1	1	1	4	1	1	1	5	1	1	1	1	8	2
ALL-Tcell-BM-DFCI-14	1	1	3	3	1	1	1	4	1	1	1	5	1	1	1	1	8	2
ALL-Tcell-BM-DFCI-2	1	1	3	3	1	1	1	4	1	1	1	5	1	1	1	1	8	2
ALL-Tcell-BM-DFCI-23	1	1	3	3	1	1	1	4	1	1	1	5	1	1	1	1	4	2
ALL-Tcell-BM-DFCI-3	1	1	3	3	1	1	1	4	1	1	1	5	1	1	1	1	8	2
ALL-Tcell-BM-DFCI-6	1	1	3	3	1	1	1	4	1	1	1	5	1	1	1	1	4	2
ALL-Tcell-BM-DFCI-9	1	1	3	3	1	1	1	4	1	1	1	5	1	1	1	1	8	2
ALL-Tcell-PB-DFCI-67	1	1	1	3	1	1	1	1	1	1	1	1	1	1	1	1	2	2
AML-BM-CALGB-28	2	1	1	1	3	3	1	5	1	1	1	6	5	1	5	1	5	6
AML-BM-CALGB-29	2	1	1	1	3	3	1	5	1	1	1	1	5	1	6	1	2	7
AML-BM-CALGB-30	2	1	1	1	3	3	1	5	1	1	1	6	5	1	5	1	6	6
AML-BM-CALGB-31	2	1	1	1	3	3	1	5	1	1	1	6	5	1	5	1	6	6
AML-BM-CALGB-32	2	1	1	1	3	3	1	5	1	1	1	6	5	1	5	1	6	6
AML-BM-CALGB-33	2	1	1	1	3	3	1	5	1	3	1	6	5	1	5	1	5	6
AML-BM-CALGB-34	2	1	1	1	3	3	1	5	1	1	1	6	5	1	5	1	6	6
AML-BM-CALGB-35	2	1	1	1	3	3	1	5	1	1	1	6	5	1	5	1	6	6
AML-BM-CALGB-36	2	1	1	1	3	3	1	5	1	4	1	6	5	1	5	1	5	6
AML-BM-CALGB-37	2	1	1	1	3	3	1	5	1	1	1	6	5	1	5	1	5	6
AML-BM-CALGB-38	2	1	1	1	3	3	1	5	1	1	1	6	5	1	5	1	5	6
AML-BM-CALGB-50	2	1	1	1	3	3	1	5	1	1	1	6	5	1	5	1	6	6
AML-BM-CALGB-51	2	1	1	1	3	3	1	5	1	1	1	6	5	1	5	1	6	6
AML-BM-CALGB-53	2	1	1	1	3	3	1	5	1	1	1	6	5	1	5	1	5	6
AML-BM-CCG-65	2	1	1	1	3	3	1	5	1	1	1	6	5	1	5	1	6	6
AML-BM-CCG-66	2	1	1	1	1	1	1	1	1	1	1	1	5	1	6	1	2	7
AML-BM-StJude-54	2	1	1	1	1	1	1	5	1	1	1	6	5	1	6	1	2	7
AML-BM-StJude-57	2	1	1	1	1	1	1	5	1	1	1	6	5	1	6	1	2	7
AML-BM-StJude-58	2	1	1	1	1	3	1	5	1	1	1	6	5	6	5	6	6	6
AML-BM-StJude-60	2	1	1	1	1	1	1	5	1	1	1	6	5	1	6	1	2	7
AML-BM-StJude-61	2	1	1	1	3	3	1	5	1	1	1	6	5	1	5	1	6	6
AML-PB-CALGB-52	2	1	1	1	3	3	4	5	1	1	5	6	5	5	6	5	6	7
AML-PB-CCG-62	2	1	1	1	3	3	1	5	1	1	1	6	5	1	5	7	6	6
AML-PB-CCG-63	2	1	1	1	3	3	1	5	1	1	1	6	5	1	5	7	6	6
AML-PB-CCG-64	2	1	1	1	3	3	1	5	5	5	6	6	6	7	7	8	5	8

de separação dos *clusters* conhecidos é facilmente observado, com o aumento de k . A visualização apresentada na Tabela 8.16, diferente da anterior, foi gerada considerando a estrutura conhecida E2, para facilitar a identificação dos outros aspectos que não estão relacionados à descoberta de subclasses. É importante mencionar que essas análises poderiam ser feitas com a visualização considerando a estrutura E1, mas optou-se por mostrar a análise com E2 para tornar mais “visíveis” os três *clusters* considerados e ilustrar a diferença entre usar uma ou outra estrutura conhecida.

Nos trabalhos encontrados na literatura, que avaliam a qualidade das partições obtidas com índices externos de validação, é considerado que os dados possuem uma única estrutura verdadeira. Nesses casos, a visualização pode ser aplicada a essa estrutura conhecida, para auxiliar nas avaliações da qualidade das partições e comparar seus *clusters* em mais detalhes.

Na Tabela 8.16, a primeira observação a ser feita é que as partições π^1 , π^2 , π^3 e π^4 são bastante parecidas, porém nenhuma delas apresenta *clusters* relevantes.

Além disso, da mesma maneira que na Tabela 8.15, os mesmos objetos, agora agrupados como B-ALL, foram isolados dos demais. É importante notar que, para esse algoritmo, cada partição corresponde a um corte na hierarquia de *clusters* gerada, representando valores sucessivamente maiores de k . Assim, π^1 possui 2 *clusters* e π^7 , 8 *clusters*. A partir disso, nota-se que esses objetos foram isolados antes que qualquer um dos *clusters* pudesse ser identificado. Como discutido no Capítulo 6, esses objetos, com padrão de comportamento distinto dos demais objetos pertencentes ao mesmo *cluster* conhecido, podem ser ruídos que devem ser descartados, podem ter sido originalmente classificados de maneira errada, devendo ser corrigidos, ou então podem ser objetos que de fato têm um comportamento atípico e merecem ser investigados pelos especialistas.

A separação dos *clusters* só começou a ocorrer de fato a partir da partição π^5 , que possui 5 *clusters*. A partir dessa partição, duas cores passam a ser identificadas na visualização, cada uma correspondendo a um dos *clusters* de E1. As partições π^5 e π^6 são bastante parecidas, porém π^6 contém uma subdivisão dos objetos AML em dois *clusters*

principais. Somente com a consideração de um *cluster* a mais, em π^7 , é que o algoritmo LM consegue a separação dos três *clusters* de E2, ou seja, a estrutura conhecida de três *clusters* só foi recuperada ao se considerar a divisão dos objetos em 8 grupos.

Com essas duas visualizações, foi mostrada a aplicação do método para a análise de um conjunto de dados real, mostrando os vários aspectos em que ele contribui na análise de agrupamento. O primeiro deles é auxiliar o especialista a inspecionar e comparar em detalhes o conteúdo das diversas partições obtidas com o MOCLE, ou qualquer outra técnica de agrupamento. Com isso, mostrou-se que o *framework* proposto consegue atingir a meta da visualização. O outro aspecto importante mostrado foi como o método de visualização pode ser útil na utilização de conhecimento prévio para auxiliar na identificação de estruturas novas. Com isso, vê-se que o método de visualização também contribui com a meta da semi-supervisão.

8.5 Considerações Finais

Neste capítulo foram apresentados os resultados obtidos nos experimentos realizados nesta Tese, mostrando que as metas descritas no Capítulo 5 foram atingidas com o *framework* proposto. Mais precisamente, cada análise feita nas seções anteriores mostrou que o *framework* atinge as metas estabelecidas da seguinte maneira:

Seleção automática de modelo: a análise da origem das soluções, juntamente com os gráficos da variância pela conectividade comparando o MOCLE com os algoritmos individuais mostrou que as partições com os melhores compromissos dos objetivos, dentre todas as usadas inicialmente, se mantiveram no conjunto de soluções do MOCLE, independentemente da configuração considerada. Ou seja, dentre uma grande variedade de partições geradas com diferentes algoritmos e com vários valores dos parâmetros desses algoritmos, apenas as de melhor qualidade em relação às medidas empregadas foram selecionadas como soluções do MOCLE. Um aspecto importante a destacar é que o MOCLE não requer um ajuste fino de parâmetros. Inicialmente foram feitas algumas convenções e os experimentos feitos com todos os conjuntos de dados consideraram as mesmas convenções para alguns parâmetros e os mesmos valores para outros. As convenções se referem aos números mínimo e máximo de *clusters* e ao número de vizinhos mais próximos considerados nos algoritmos, que foram atrelados ao tamanho do conjunto de dados ou ao número de *clusters* da estrutura conhecida. Assim, o especialista não precisa se preocupar com o ajuste fino de parâmetros para obter um bom desempenho. Apenas precisa escolher um número mínimo e máximo de *clusters*, o que não é uma escolha muito crítica para o desempenho da técnica.

Diversidade: o primeiro ponto necessário para ilustrar como o *framework* atinge essa

Tabela 8.16: Visualização para o LM considerando a estrutura E2

Objeto	π^{E1}	π^1	π^2	π^3	π^4	π^5	π^6	π^7
ALL-Bcell-BM-DFCI-1	1	1	1	1	1	1	1	1
ALL-Bcell-BM-DFCI-12	1	1	1	1	1	1	1	1
ALL-Bcell-BM-DFCI-13	1	1	1	1	1	1	1	1
ALL-Bcell-BM-DFCI-15	1	1	1	1	1	1	1	1
ALL-Bcell-BM-DFCI-16	1	1	1	1	1	1	1	1
ALL-Bcell-BM-DFCI-17	1	2	2	2	2	2	2	3
ALL-Bcell-BM-DFCI-18	1	1	1	1	1	1	1	1
ALL-Bcell-BM-DFCI-19	1	1	1	1	1	1	1	1
ALL-Bcell-BM-DFCI-20	1	2	2	2	3	3	3	4
ALL-Bcell-BM-DFCI-21	1	1	3	3	4	4	4	5
ALL-Bcell-BM-DFCI-22	1	1	1	1	1	1	1	1
ALL-Bcell-BM-DFCI-24	1	1	1	1	1	1	1	1
ALL-Bcell-BM-DFCI-25	1	1	1	1	1	1	1	1
ALL-Bcell-BM-DFCI-26	1	1	1	1	1	1	1	1
ALL-Bcell-BM-DFCI-27	1	1	1	1	1	1	1	1
ALL-Bcell-BM-DFCI-39	1	1	1	1	1	1	1	1
ALL-Bcell-BM-DFCI-4	1	1	1	1	1	1	1	1
ALL-Bcell-BM-DFCI-40	1	1	1	1	1	1	1	1
ALL-Bcell-BM-DFCI-41	1	1	1	1	1	1	1	1
ALL-Bcell-BM-DFCI-42	1	1	1	1	1	1	1	1
ALL-Bcell-BM-DFCI-43	1	1	1	1	1	1	1	1
ALL-Bcell-BM-DFCI-44	1	1	1	1	1	1	1	1
ALL-Bcell-BM-DFCI-45	1	1	1	1	1	1	1	1
ALL-Bcell-BM-DFCI-46	1	1	1	1	1	1	1	1
ALL-Bcell-BM-DFCI-47	1	1	1	1	1	1	1	1
ALL-Bcell-BM-DFCI-48	1	1	1	1	1	1	1	1
ALL-Bcell-BM-DFCI-49	1	1	1	1	1	1	1	1
ALL-Bcell-BM-DFCI-5	1	1	1	1	1	1	1	1
ALL-Bcell-BM-DFCI-7	1	1	1	1	1	1	1	1
ALL-Bcell-BM-DFCI-8	1	1	1	1	1	1	1	1
ALL-Bcell-BM-StJude-55	1	1	1	1	1	1	1	1
ALL-Bcell-BM-StJude-56	1	1	1	1	1	1	1	1
ALL-Bcell-BM-StJude-59	1	1	1	1	1	1	1	1
ALL-Bcell-PB-DFCI-68	1	1	1	1	1	1	1	1
ALL-Bcell-PB-DFCI-69	1	1	1	1	1	1	1	1
ALL-Bcell-PB-DFCI-70	1	1	1	1	1	1	1	1
ALL-Bcell-PB-DFCI-71	1	1	1	1	1	1	1	1
ALL-Bcell-PB-DFCI-72	1	1	1	1	1	1	1	1
ALL-Tcell-BM-DFCI-10	2	1	1	1	1	1	1	2
ALL-Tcell-BM-DFCI-11	2	1	1	1	1	1	1	2
ALL-Tcell-BM-DFCI-14	2	1	1	1	1	1	1	2
ALL-Tcell-BM-DFCI-2	2	1	1	1	1	1	1	2
ALL-Tcell-BM-DFCI-23	2	1	1	1	1	1	1	2
ALL-Tcell-BM-DFCI-3	2	1	1	1	1	1	1	2
ALL-Tcell-BM-DFCI-6	2	1	1	1	1	1	1	2
ALL-Tcell-BM-DFCI-9	2	1	1	1	1	1	1	2
ALL-Tcell-PB-DFCI-67	2	1	1	1	1	1	1	2
AML-BM-CALGB-28	3	1	1	1	1	5	5	6
AML-BM-CALGB-29	3	1	1	1	1	5	6	7
AML-BM-CALGB-30	3	1	1	1	1	5	5	6
AML-BM-CALGB-31	3	1	1	1	1	5	5	6
AML-BM-CALGB-32	3	1	1	1	1	5	5	6
AML-BM-CALGB-33	3	1	1	1	1	5	5	6
AML-BM-CALGB-34	3	1	1	1	1	5	5	6
AML-BM-CALGB-35	3	1	1	1	1	5	5	6
AML-BM-CALGB-36	3	1	1	1	1	5	5	6
AML-BM-CALGB-37	3	1	1	1	1	5	5	6
AML-BM-CALGB-38	3	1	1	1	1	5	5	6
AML-BM-CALGB-50	3	1	1	1	1	5	5	6
AML-BM-CALGB-51	3	1	1	1	1	5	5	6
AML-BM-CALGB-53	3	1	1	1	1	5	5	6
AML-BM-CCG-65	3	1	1	1	1	5	5	6
AML-BM-CCG-66	3	1	1	1	1	5	6	7
AML-BM-StJude-54	3	1	1	1	1	5	6	7
AML-BM-StJude-57	3	1	1	1	1	5	6	7
AML-BM-StJude-58	3	1	1	1	1	5	5	6
AML-BM-StJude-60	3	1	1	1	1	5	6	7
AML-BM-StJude-61	3	1	1	1	1	5	5	6
AML-PB-CALGB-52	3	1	1	1	1	5	6	7
AML-PB-CCG-62	3	1	1	1	1	5	5	6
AML-PB-CCG-63	3	1	1	1	1	5	5	6
AML-PB-CCG-64	3	1	1	4	5	6	7	8

Maior <i>cluster</i> de B-ALL
Segundo maior <i>cluster</i> de B-ALL
Terceiro maior <i>cluster</i> de B-ALL
Demais <i>clusters</i> de B-ALL
Maior <i>cluster</i> de T-ALL
Maior <i>cluster</i> de AML
Segundo maior <i>cluster</i> de AML
Terceiro maior <i>cluster</i> de AML

meta foi a escolha de diversos conjuntos de dados que sabidamente possuíam mais de uma estrutura. Além disso, foram escolhidos conjuntos de dados e estruturas que estavam em conformidade com diferentes definições de *cluster* e especialmente, as estruturas que estavam em diferentes níveis de refinamento. A partir disso, foi mostrado que os conjuntos de soluções gerados pelo MOCLE contêm partições que se aproximam das várias estruturas conhecidas. Essa proximidade foi medida pelo índice de validação externa CR . Nas diversas formas em que o MOCLE foi comparado com as demais técnicas de combinação, foi mostrado que o MOCLE tem um desempenho superior em recuperar as estruturas conhecidas, sendo o maior destaque dado a configuração MSH.

Concisão e qualidade: a qualidade das soluções foi mostrada pelas análises empregando o índice CR para comparar as partições obtidas com as estruturas conhecidas. A concisão foi mostrada pela comparação do número médio de soluções obtidas com o MOCLE nas diversas execuções em comparação com o número de partições iniciais que ele utiliza e também com o número médio de soluções obtidas nas diversas execuções com o algoritmo MK. É preciso reforçar que, apesar de o MK retornar um conjunto de recomendações com um número de soluções bastante próximo do número de estruturas conhecidas, a qualidade dessas soluções foi das piores dentre todas as técnicas avaliadas.

Estabilidade: na análise da qualidade das partições em relação às estruturas conhecidas, o desvio padrão do CR nas 30 execuções teve um valor bem mais baixo para o MOCLE na maioria dos casos, em comparação com as demais técnicas de combinação. Além disso, mostrou-se que em muitos dos casos, a mesma solução parecida com uma estrutura conhecida foi obtida nas 30 execuções (desvio padrão 0), e em alguns casos, em todas as configurações do MOCLE.

Informações complementares: como um resultado complementar da seleção das partições feita pelo MOCLE, cada partição carrega os respectivos valores das funções

objetivo otimizadas. Cada uma dessas funções dá uma visão diferente da qualidade de cada partição. As características de duas dessas funções foram mostradas nas comparações do MOCLE com as demais técnicas feitas com os gráficos desses objetivos (variância e conectividade). Os valores dessas funções objetivo podem ajudar o especialista na interpretação do conteúdo das partições, ou na seleção de alguma em especial que tenha as qualidades de maior interesse. Essas informações complementares estão disponíveis no MOCLE. Porém a investigação do seu potencial de uso pelos especialistas nos dados ainda requer investigação.

Semi-supervisão: em todas as seções foram apresentados aspectos que mostram como o MOCLE atinge essa meta. Nas Seções 8.2 e 8.3 foi mostrado que a qualidade das partições obtidas com as configurações semi-supervisionadas superava a qualidade das obtidas com as configurações não supervisionadas. Na Seção 8.4, foi ilustrado como a visualização permite a consideração do conhecimento prévio de uma estrutura na identificação de outras estruturas.

Visualização: na Seção 8.4 foram apresentados dois exemplos do método de visualização, que permitem, de maneira sucinta, apresentar o potencial do método. Esses exemplos correspondem a dois conjuntos de soluções diferentes obtidos para o conjunto de dados `golub`, um conjunto de dados real, que ilustra claramente os potenciais benefícios do método de visualização.

Assim, mostrou-se que o *framework* proposto atinge as metas especificadas no Capítulo 5, sendo uma ferramenta bastante útil aos especialistas no domínio dos dados, sem requerer conhecimento profundo em análise de agrupamento.

Não foram feitas medidas dos tempos de execução das técnicas, pois os experimentos foram realizados em diversas máquinas com configurações bastante diferentes, muitas vezes compartilhando recursos com outros usuários. Entretanto, pôde-se observar que uma execução do MOCLE variou de poucos minutos a aproximadamente 6 horas, dependendo das características do conjunto de dados (número de atributos, objetos, *clusters*), tempos não muito diferentes dos obtidos com o MOCK. Neste caso, uma execução corresponde ao processo de evolução do AG para chegar ao conjunto de soluções, dada uma população inicial gerada previamente.

No Capítulo 9 serão apresentadas as conclusões do trabalho, ressaltando os principais resultados obtidos, as contribuições e limitações do trabalho e indicando os trabalhos futuros que podem ser desenvolvidos a partir desta Tese.

Conclusão

9.1 Considerações Iniciais

Nesta Tese foi proposto um *framework* para a análise de exploratória de dados por meio de técnicas de agrupamento. O objetivo central desse *framework* é facilitar o trabalho dos especialistas do domínio dos dados. Isso é feito pela integração de diversos aspectos da análise de agrupamento de maneira a minimizar as principais dificuldades encontradas.

O *framework* proposto engloba um algoritmo de *ensemble* multi-objetivo, MOCLE, e um método para a visualização integrada de um conjunto de partições. MOCLE, na verdade, é mais do que um algoritmo de agrupamento tradicional, como o k -médias, ou um *ensemble*, pois ele integra no próprio algoritmo diversos aspectos da análise de agrupamento.

Mais especificamente, por meio da integração das idéias de agrupamento multi-objetivo e *ensembles* de agrupamentos, MOCLE realiza automaticamente as várias etapas importantes da análise de agrupamento: aplica vários algoritmos conceitualmente diferentes, utilizando várias configurações de seus parâmetros, combina as partições resultantes desses algoritmos e seleciona aquelas com os melhores compromissos de diferentes aspectos de qualidade das partições, dados por diferentes medidas de validação.

Com essas características, MOCLE constitui uma abordagem robusta para lidar com diferentes tipos de estrutura que podem estar presentes nos dados, fornecendo como resultado um conjunto conciso e estável de estruturas alternativas de alta qualidade, sem a necessidade de conhecimento prévio dos dados e nem conhecimento profundo em análise de agrupamento.

Além disso, MOCLE permite a integração automática de conhecimento prévio, por meio das funções objetivo, para auxiliar na obtenção de outras estruturas presentes nos

dados. Assim, ele pode ser utilizado tanto em um contexto não supervisionado, em que nenhum conhecimento prévio sobre os dados está disponível, quanto em um contexto semi-supervisionado, que considera um conhecimento parcial sobre os dados.

A integração do agrupamento multi-objetivo com *ensembles* garante as vantagens de ambas as técnicas e ajuda a superar as limitações individuais de cada uma delas. O aspecto multi-objetivo permite que o *ensemble* gere um conjunto de partições consenso, evitando também a necessidade de determinação prévia do número de *clusters*, e garante a qualidade das soluções, mesmo com um número elevado de partições iniciais ruins. Por outro lado, o aspecto *ensemble* auxilia na concisão e estabilidade do conjunto de soluções, uma vez que limita o espaço de busca às partições iniciais e suas combinações.

O método de visualização permite a observação simultânea de um conjunto de partições, auxiliando na análise do resultado gerado pelo MOCLE. Esse método associa cores aos *clusters* das partições considerando o conhecimento prévio de uma estrutura dos dados. Portanto, o método de visualização se aplica somente ao contexto semi-supervisionado. Além de parte integrante do *framework*, o método de visualização pode ser visto como uma contribuição independente dele, facilitando a análise de qualquer conjunto de partições e complementando as informações que podem ser obtidas com índices de validação externa nas comparações entre várias técnicas de agrupamento.

Conforme já mencionado, as metas estabelecidas para esta Tese são atingidas por meio dos seguintes aspectos do *framework*:

- Seleção automática de modelo: população inicial e funções objetivo.
- Diversidade: AG baseado em Pareto, população inicial e funções objetivo.
- Concisão e qualidade: população inicial, operador de recombinação e funções objetivo.
- Estabilidade: população inicial e operador de recombinação.
- Informações complementares: funções objetivo.
- Semi-supervisão: funções objetivo, se usada uma função que considere conhecimento prévio, e método de visualização.
- Visualização: método de visualização.

9.2 Principais Resultados

Os experimentos e análises descritos nesta Tese mostraram que o *framework* proposto tem as seguintes características:

- Estabilidade: para um determinado conjunto de dados e configuração de parâmetros, as mesmas melhores partições (ou, pelo menos, soluções muito similares) com relação a cada estrutura conhecida sempre aparecem no conjunto de soluções.
- Concisão: o número de partições no conjunto de soluções é menor do que o obtido com MOCK, podendo ser mais facilmente analisado pelos especialistas do domínio dos dados.
- Robustez: a abordagem proposta resulta em soluções de alta qualidade para uma grande variedade de conformações dos dados. A qualidade das partições pode ser observada tanto nos valores do CR quanto no número de *clusters* das partições do conjunto de soluções.
- Diversidade: a abordagem foi capaz de encontrar, em uma única execução, várias partições próximas às diversas estruturas presentes em um mesmo conjunto de dados, em diferentes níveis de refinamento e com diferentes conformações.
- A aplicação do *framework* não requer um ajuste muito refinado dos parâmetros dos algoritmos para diferentes conjuntos de dados. Os valores dos parâmetros são definidos empregando o tamanho do conjunto de dados ou o número de *clusters* de uma estrutura conhecida para determinar um intervalo de variação de k . Mesmo que não seja conhecida uma estrutura dos dados, a escolha do intervalo de variação de k não é uma escolha crítica para o desempenho da técnica. Assim, os ajustes necessários, incluindo a geração das partições iniciais, podem ser realizados facilmente pelo usuário, sem a necessidade de muito conhecimento sobre o algoritmo, análise de agrupamento ou ainda sobre os dados.
- Permite a utilização automática de conhecimento prévio de uma estrutura simples do conjunto de dados para auxiliar na obtenção de outras estruturas mais complexas.
- Permite visualizar simultaneamente todas as partições resultantes do MOCLE.
- Juntamente com as partições, fornece informações sobre a sua qualidade sob diversos aspectos, dadas pelos valores das diversas medidas de validação otimizadas.

Baseado nisso, pode-se argumentar que MOCLE, juntamente com o método de visualização, consegue atingir todas as metas estabelecidas. Isso mostra que o *framework* consegue superar, ou ao menos minimizar, as dificuldades apontadas a respeito das abordagens existentes para análise de agrupamento.

Além disso, o método de visualização permite a comparação de qualquer conjunto de partições, auxiliando as investigações comumente realizadas em análise de agrupamento

que comparam várias partições, servindo como um complemento às informações obtidas com índices de validação externa, como o *CR*.

9.3 Contribuições do Trabalho

As duas contribuições centrais desta Tese foram o algoritmo de *ensemble* multi-objetivo proposto, MOCLE, e o método de visualização desenvolvido, que compõem um *framework* útil aos especialistas que fazem uso de agrupamento para análise exploratória de seus dados.

A principal inovação do algoritmo MOCLE está na integração das abordagens de agrupamento multi-objetivo e *ensemble* de agrupamentos. Para isso, um AG multi-objetivo foi empregado para otimizar duas ou três medidas de qualidade de uma partição, utilizando um operador especial de recombinação que realiza a combinação de duas partições. Também é uma novidade do *ensemble* proposto a combinação das partições aos pares, num processo iterativo.

Outra novidade está na integração de conhecimento prévio a respeito de uma estrutura simples dos dados para auxiliar na descoberta de outras estruturas. Nesse sentido, tanto o MOCLE quanto o método de visualização constituem uma contribuição importante. Principalmente, pode ser considerada uma contribuição, a utilização automática de conhecimento prévio na investigação da existência de subclasses nos dados com classes conhecidas, como as feitas com dados de doenças, como o câncer, para descobrir seus subtipos.

Também é uma novidade do trabalho a avaliação da habilidade das técnicas de agrupamento para a obtenção de mais de uma estrutura existente nos dados em uma única análise. Os trabalhos existentes na literatura, investigados até o momento, mencionam que os dados podem conter estruturas em diferentes níveis de refinamento e/ou com diferentes conformações, mas nenhum deles apresenta uma análise do desempenho de técnicas de agrupamento, quer as tradicionais, quer os *ensembles* ou o agrupamento multi-objetivo, na obtenção de todas as estruturas. Os trabalhos investigados avaliam as técnicas apenas em relação a uma única estrutura. Os algoritmos convencionais e os *ensembles* não têm a habilidade de encontrar várias estruturas em uma única execução. O MOCLE apresenta essa habilidade. Apesar disso, nos artigos que o descrevem, nenhuma investigação a esse respeito foi feita. Nesta Tese foram propositalmente escolhidos conjuntos de dados com várias estruturas com diferentes conformações e em diferentes níveis de refinamento para avaliar a obtenção de mais de uma estrutura.

O método para a visualização simultânea de um conjunto de partições é uma outra novidade. Na literatura existem vários métodos para visualizar partições, mas, nos trabalhos estudados, os métodos se aplicam a visualização de uma única partição de cada

vez, apesar de poderem ser geradas visualizações para várias partições que podem ser posteriormente comparadas. A visualização integrada das partições, como proposta nesta Tese, facilita sua comparação.

Como fruto dessas pesquisas, até o presente momento, foi apresentado e publicado um artigo sobre o método de visualização no *Brazilian Symposium on Bioinformatics*, BSB 2005 (Faceli et al. 2005c) e aceito outro artigo para apresentação e publicação na *6th International Conference on Hybrid Intelligent Systems*, HIS'06, em dezembro, sobre os resultados preliminares obtidos com o MOCLE. Além disso, juntamente com esta Tese, está sendo submetido um artigo para a revista *IEEE Transactions on Evolutionary Computation*, apresentando os resultados obtidos com a configuração não supervisionada do MOCLE. Em seguida, serão concluídos dois outros artigos, um envolvendo a totalidade das análises feitas com o MOCLE, e outro com novas análises feitas com o método de visualização, ambos a serem submetidos a periódicos internacionais.

Além das novidades da abordagem proposta, a partir das investigações feitas, foram gerados três relatórios técnicos com revisões sobre algoritmos de agrupamento (Faceli et al. 2005a), técnicas de validação de agrupamento (Faceli et al. 2005d) e análise de dados de expressão gênica (Faceli et al. 2005b).

Nas investigações preliminares sobre dados de expressão gênica que contribuíram para a motivação da abordagem proposta, também foram feitos experimentos no contexto de seleção de genes para classificação de dados sobre câncer. Essas investigações também geraram diversas publicações: um resumo apresentado na *1st International Conference on Bioinformatics and Computational Biology*, ICoBiCoBi'2003 (Faceli et al. 2003b), a posterior publicação da versão completa desse trabalho na revista *Genetics and Molecular Biology* (Faceli et al. 2004), e um trabalho publicado nos anais do IV Encontro Nacional de Inteligência Artificial, ENIA'2003 (Faceli et al. 2003a), classificado entre os 15 melhores artigos e convidado para a publicação de uma versão completa na revista *Scientia* (Faceli et al. 2003c).

9.4 Limitações

As principais limitações do trabalho apresentado são:

- Foram feitos testes com apenas duas medidas de validação internas e uma externa.
- Foram utilizados três algoritmos tradicionais de agrupamento e apenas um mais recente para gerar a população inicial. Seria interessante investigar outros algoritmos mais recentes, que, por si só, já melhoram algumas das limitações das abordagens tradicionais.
- Os experimentos realizados indicam haver pouca influência do algoritmo genético

empregado. Entretanto, estudos comparativos mais detalhados envolvendo diversos AGs são necessários para investigar isso.

- Não foram feitas investigações sobre o impacto do uso das informações complementares resultantes do MOCLE.
- O método de visualização requer o conhecimento prévio de uma estrutura dos dados.

9.5 Trabalhos Futuros

Alguns dos possíveis trabalhos futuros encontram-se na solução das limitações da proposta, outros se referem à ampliação das capacidades do *framework*.

A primeira possibilidade de trabalho futuro é investigar a utilização de outros algoritmos de agrupamento para gerar a população inicial, juntamente com a otimização de outras funções objetivo. A primeira opção é incluir algoritmos que encontrem *clusters* em sub-espacos do espaco de atributos original e adaptar as funções objetivo para este caso.

Outro trabalho consiste da elaboração de uma forma adequada para mostrar as informações complementares, dadas pelos valores dos objetivos, ao especialista, que complementem as informações dos gráficos. A partir disso, investigar mais sistematicamente as maneiras como essas informações podem ser úteis aos especialistas.

Outra possibilidade interessante é investigar a utilização de AGs que limitem o número de soluções obtidas, como o SPEA2 que emprega uma população externa de tamanho limitado. Com isso seria possível reduzir ainda mais o número de soluções obtidas.

Provavelmente o trabalho mais imediato a ser feito seja adequar o método de visualização para não necessitar do conhecimento prévio. Uma possibilidade é usar como referência, no lugar da estrutura conhecida, uma partição que seja o consenso entre as partições visualizadas.

Ainda para melhorar o método de visualização, uma tarefa a ser feita é investigar uma maneira para ordenar os objetos na visualização.

Finalmente, para facilitar ainda mais a interpretação do conjunto de soluções, um trabalho interessante está em investigar uma maneira para integrar as informações complementares obtidas com o MOCLE ao método de visualização.

Referências

- Alizadeh, A., M. Eisen, R. Davis, C. Ma, I. Lossos, A. Rosenwald, J. Boldrick, H. Sabet, T. Tran, X. Yu, J. Powell, L. Yang, G. Marti, T. Moore, J. J. Hudson, L. Lu, D. Lewis, R. Tibshirani, G. Sherlock, W. Chan, T. Greiner, D. Weisenburger, J. Armitage, R. Warnke, R. Levy, W. Wilson, M. Grever, J. Byrd, D. Botstein, P. Brown, and L. Staudt (2000). Distinct types of diffuse large B-cell lymphoma identified by gene expression profiling. *Nature* 403(6769), 503–511.
- Alon, U., N. Barkai, D. A. Notterman, K. Gish, S. Ybarra, D. Mack, and A. Levine (1999). Broad patterns of gene expression revealed by clustering analysis of tumor and normal colon tissues probed by oligonucleotide arrays. In *Proc. Natl. Acad. Sci. USA*, Volume 96, pp. 6745–6750.
- Ankerst, M., M. M. Breunig, H.-P. Kriegel, and P. Sander (1999). Optics: Ordering points to identify the clustering structure. In *Proceedings of ACM SIGMOD International Conference on Management of Data (SIGMOD'1999)*, pp. 49–60.
- Azuaje, F. (2000). Gene expression patterns and cancer classification: A self adaptive and incremental neural approach. Technical Report TCD-CS-2000-26, Trinity College Dublin, Computer Science Department.
- Azuaje, F. and J. Dopazo (Eds.) (2005). *Data Analysis and Visualization in Genomics and Proteomics*. John Wiley & Sons.
- Bäck, T., U. Hammel, and H.-P. Schwefel (1997). Evolutionary computation: comments on the history and current state. *IEEE Transactions on Evolutionary Computation* 1(1), 3–17.
- Baldi, P. and S. Brunak (1998). *Bioinformatics: The Machine Learning Approach*. Adaptive Computation and Machine Learning. MIT Press.
- 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 [Acesso em 12/11/2003].
- Ben-Dor, A., R. Shamir, and Z. Yakhini (1999). Clustering gene expression patterns. *Journal of Computational Biology* 6(3/4), 281–297.

- Berkhin, P. (2002). Survey of clustering data mining techniques. Technical report, Accrue Software, San Jose, CA. http://www.accrue.com/products/rp_cluster_review.pdf [Acessado em 05/02/2004].
- Bhattacharjee, A., W. G. Richards, J. Staunton, C. Li, S. Monti, P. Vasa, C. Ladd, J. Beheshti, R. Bueno, M. Gillette, M. Loda, G. Weber, E. J. Mark, E. S. Lander, W. Wong, B. E. Johnson, T. R. Golub, D. J. Sugarbaker, , and M. Meyerson (2001). Classification of human lung carcinomas by mRNA expression profiling reveals distinct adenocarcinoma sub-classes. In *Proc. Natl. Acad. Sci. USA*, Volume 98, pp. 13790–13795.
- Bittner, M., P. Meltzer, Y. Chen, Y. Jiang, E. Seftor, M. Hendrix, M. Radmacher, R. Simon, Z. Yakhini, A. Ben-Dork, N. Sampas, E. Dougherty, E. Wang, F. Marincola, C. Gooden, J. Lueders, A. Glatfelter, P. Pollock, J. Carpten, E. Gillanders, D. Leja, K. Dietrich, C. Beaudry, M. Berens, D. Alberts, V. Sondak, N. Hayward, and J. Trent (2000). Molecular classification of cutaneous malignant melanoma by gene expression profiling. *Nature* 406, 536–540.
- Breiman, L. (1996). Technical note: some properties of splitting criteria. *Machine Learning* 24(1), 41–47.
- Cheng, Y. and G. Church (2000). Biclustering of expression data. In *Proceedings of the Eighth International Conference on Intelligent Systems for Molecular Biology (ISMB'2000)*, pp. 93–103.
- Coello, C. A. C. (1999). A comprehensive survey of evolutionary-based multiobjective optimization techniques. *Knowledge and Information Systems* 1(3), 129–156.
- Cole, R. M. (1998). Clustering with genetic algorithms. Master's thesis, University of Western Australia, Australia.
- Corne, D. W., N. R. Jerram, J. D. Knowles, and M. J. Oates (2001). PESA-II: Region-based selection in evolutionary multiobjective optimization. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-2001)*, San Francisco, California, USA, pp. 283–290. Morgan Kaufmann.
- Corne, D. W., J. D. Knowles, and M. J. Oates (2000). The pareto envelope-based selection algorithm for multiobjective optimization. In M. Schoenauer, G. R. Kalyanmoy Deb, X. Yao, E. Lutton, J. J. Merelo, and H.-P. Schwefel (Eds.), *Proceedings of the Parallel Problem Solving from Nature Conference (PPSN'2000)*, Volume 1917 of *Lecture Notes in Computer Science*, Paris, France, pp. 869–878. Springer Verlag.
- Deb, K., A. Pratap, S. Agarwal, and T. Meyrivan (2002). A fast and elitist multi-objective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Com-*

- putation* 6(2), 182–197.
- Demiriz, A., K. P. Bennett, and M. J. Embrechts (1999). Semi-supervised clustering using genetic algorithms. In *Artificial Neural Networks in Engineering (ANNIE-1999)*, pp. 809–814.
- Demšar, J. (2006). Statistical comparisons of classifiers over multiple data sets. *Journal of Machine Learning Research* 7, 1–30.
- Duda, R., P. Hart, and D. Stork (2001). *Pattern Classification*. John Wiley & Sons.
- Dudoit, S., J. Fridlyand, and T. P. Speed (2000). Comparison of discrimination methods for the classification of tumors using gene expression data. Technical Report 576, Department of Statistics, UC Berkeley.
- Eisen, M. B., P. Spellman, P. Brown, and D. Botstein (1998). Cluster analysis and display of genome-wide expression patterns. In *Proc. Natl. Acad. Sci. USA*, Volume 95, pp. 14863–14868.
- Ertöz, L., M. Steinbach, and V. Kumar (2002). A new shared nearest neighbor clustering algorithm and its applications. In *Proceedings of the Workshop on Clustering High Dimensional Data and its Applications, 2nd SIAM International Conference on Data Mining (SDM'2002)*, pp. 105–115.
- Ertöz, L., M. Steinbach, and V. Kumar (2003a). *Clustering and Information Retrieval*, Chapter Finding Topics in Collections of Documents: A Shared Nearest Neighbor Approach, pp. 83–103. Network Theory and Applications. Kluwer Academic Publishers.
- Ertöz, L., M. Steinbach, and V. Kumar (2003b). Finding clusters of different sizes, shapes, and densities in noisy, high dimensional data. In *SIAM International Conference on Data Mining (SDM'2003)*. http://www.siam.org/meetings/sdm03/proceedings/sdm03_05.pdf [Acessado em 16/10/2006].
- Ester, M., H.-P. Kriegel, J. Sander, and X. Xu (1996). A density-based algorithm for discovering clusters in large spatial databases with noise. In *Proceedings of 2nd Int. Conf. on Knowledge Discovery and Data Mining (KDD'1996)*, pp. 226–231.
- Estivill-Castro, V. (2002). Why so many clustering algorithms - a position paper. *SIGKDD Explorations* 4(1), 65–75.
- Faceli, K., A. Carvalho, and W. A. S. Jr (2003a). Gene selection for tumor cell classification. In *Anais do IV Encontro Nacional de Inteligência Artificial (ENIA'2003), XXIII Congresso da Sociedade Brasileira de Computação (SBC'2003)*, Campinas, SP, pp. 309–316.

- Faceli, K., A. Carvalho, and W. S. Jr (2003b). Evaluation of gene selection metrics for tumor cell classification. Resumo apresentado na 1st International Conference on Bioinformatics and Computational Biology (ICoBiCoBi), Ribeirão Preto, SP.
- Faceli, K., A. Carvalho, and W. S. Jr (2003c). Gene selection for tumor cell classification. *SCIENTIA* 14(2), 309–324.
- Faceli, K., A. Carvalho, and W. S. Jr (2004). Evaluation of gene selection metrics for tumor cell classification. *Genetics and Molecular Biology* 27(4), 651–657.
- Faceli, K., A. Carvalho, and M. Souto (2005a). Algoritmos de agrupamentos de dados. Relatório Técnico 249, Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo. <http://www.icmc.usp.br/~biblio/index.php?destino=relatoriostecnicos> [Acessado em 28/01/2005].
- Faceli, K., A. Carvalho, and M. Souto (2005b). Análise de dados de expressão gênica. Relatório Técnico 250, Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo. <http://www.icmc.usp.br/~biblio/index.php?destino=relatoriostecnicos> [Acessado em 28/01/2005].
- Faceli, K., A. Carvalho, and M. Souto (2005c). Evaluation of the contents of partitions obtained with clustering gene expression data. In J. C. Setubal and S. Verjovski-Almeida (Eds.), *Proceedings of the Brazilian Symposium on Bioinformatics (BSB'2005)*, Volume 3594 of *Lecture Notes in Computer Science*, São Leopoldo, Brasil, pp. 65–76. Springer Verlag.
- Faceli, K., A. Carvalho, and M. Souto (2005d). Validação de algoritmos de agrupamento. Relatório Técnico 254, Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo. <http://www.icmc.usp.br/~biblio/index.php?destino=relatoriostecnicos> [Acessado em 04/06/2005].
- Faceli, K., A. Carvalho, and M. Souto (2006). Multi-objective clustering ensemble. In *Proceedings of the 6th International Conference on Hybrid Intelligent Systems (HIS'2006)*, Auckland, New Zealand. IEEE Computer Society Press. To appear.
- Falkenauer, E. (1998). *Genetic algorithms and grouping problems*. John Wiley & Sons.
- Fern, X. Z. and C. E. Brodley (2004). Solving cluster ensemble problems by bipartite graph partitioning. In *Proceedings of the Twenty-first International Conference on Machine Learning (ICML'2004)*, New York, NY, USA, pp. 36. ACM Press.
- Fred, A. and A. Jain (2002). Evidence accumulation clustering based on the k-means algorithm. In *Proceedings of Structural and Syntactic Pattern Recognition (SSPR'2002)*, Windsor, Canada, pp. 442–451.

- Fred, A. and A. K. Jain (2003). Robust data clustering. In *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition, (CVPR'2003)*, Volume II, Madison - Wisconsin, USA.
- Fred, A. L. N. (2001). Finding consistent clusters in data partitions. In J. Kittler and F. Roli (Eds.), *Second International Workshop on Multiple Classifier Systems (MCS'2001)*, Volume 2096 of *Lecture Notes in Computer Science*, Cambridge, UK, pp. 309–318.
- Frossyniotis, D., M. Pertselakis, and A. Stafylopatis (2002). A multi-clustering fusion algorithm. In I. P. Vlahavas and C. D. Spyropoulos (Eds.), *Methods and Applications of Artificial Intelligence, Proceedings of the 2nd Hellenic Conference on AI (SETN'2002)*, Volume 2308 of *Lecture Notes in Computer Science*, Thessaloniki, Greece, pp. 225–236. Springer Verlag.
- Gama, J. M. P. (1999). *Combining Classification Algorithms*. Ph. D. thesis, Departamento de Ciência de Computadores, Faculdade de Ciências da Universidade do Porto. <http://www.ncc.up.pt/~jgama/tese.ps.gz> [Acessado em 19/07/2002].
- Getz, G., H. Gal, I. Kela, D. A. Notterman, and E. Domany (2003). Coupled two-way clustering analysis of breast cancer and colon cancer gene expression data. *Bioinformatics* 19, 1079–1089.
- Ghosh, J., A. Strehl, and S. Merugu (2002). A consensus framework for integrating distributed clusterings under limited knowledge sharing. In *Proceedings of NSF Workshop on Next Generation Data Mining*, pp. 99–108.
- Golub, T., P. T. D.K. Slonim and, C. Huard, M. Gaasenbeek, J. Mesirov, H. Coller, M. Loh, J. Downing, M. Caligiuri, C. Bloomfield, and E. Lander (1999). Molecular classification of cancer: Class discovery and class prediction by gene expression monitoring. *Science* 286(5439), 531–537.
- Gordon, A. (1999). *Classification*. Chapman & Hall/CRC.
- Hadjitodorov, S. T., L. I. Kuncheva, and L. P. Todorova (2006). Moderate diversity for better cluster ensembles. *Information Fusion* 7(3), 264–275.
- Halkidi, M., Y. Batistakis, and M. Vazirgiannis (2001). On clustering validation techniques. *Intelligent Information Systems Journal* 17(2-3), 107–145.
- Halkidi, M., Y. Batistakis, and M. Vazirgiannis (2002a). Cluster validity methods: Part I. *SIGMOD Record* 31(2), 40–45.
- Halkidi, M., Y. Batistakis, and M. Vazirgiannis (2002b). Cluster validity methods: Part II. *SIGMOD Record* 31(3), 19–27.

- Handl, J. and J. Knowles (2004). Multiobjective clustering with automatic determination of the number of clusters. Technical Report TR-COMPSYSBIO-2004-02, UMIST, Manchester.
- Handl, J. and J. Knowles (2005a). Exploiting the trade-off - the benefits of multiple objectives in data clustering. In C. A. Coello et al. (Eds.), *Proceedings of the Third International Conference on Evolutionary Multi-Criterion Optimization (EMO'2005)*, Volume 3410 of *Lecture Notes in Computer Science*, Guanajuato, Mexico, pp. 547–560. Springer-Verlag.
- Handl, J. and J. Knowles (2005b). Improvements to the scalability of multiobjective clustering. In *IEEE Congress on Evolutionary Computation*, pp. 438–445. IEEE Computer Society Press.
- Handl, J. and J. Knowles (2006a). An evolutionary approach to multiobjective clustering. *IEEE Transactions on Evolutionary Computation*. To appear.
- Handl, J. and J. Knowles (2006b). On semi-supervised clustering via multiobjective optimization. In *Proceedings of the 8th annual conference on Genetic and evolutionary computation (GECCO'2006)*, New York, NY, USA, pp. 1465–1472. ACM Press.
- Handl, J., J. Knowles, and D. Kell (2005). Computational cluster validation in post-genomic data analysis. *Bioinformatics* 21(15), 3201–3212.
- Hartigan, J. A. (1985). Statistical theory in clustering. *Journal of Classification* 2(1), 63–76.
- Hastie, T., R. Tibshirani, M. B. Eisen, A. Alizadeh, R. Levy, L. Staudt, W. Chan, D. Botstein, and P. Brown (2000). Gene shaving as a method for identifying distinct sets of genes with similar expression patterns. *Genome Biology* 1(2), research0003.1–0003.21.
- Hautaniemi, S., O. Yli-Harja, J. Astola, P. Kauraniemi, A. Kallioniemi, M. Wolf, J. Ruiz, S. Mousses, and O.-P. Kallioniemi (2003). Analysis and visualization of gene expression microarray data in human cancer using self-organizing maps. *Machine Learning* 52(1-2), 45–66.
- He, Q. (1999). A review of clustering algorithms as applied in IR. Technical Report UIUCLIS-1999/6+IRG, Information Retrieval Group, University of Illinois.
- Herrero, J., A. Valencia, and J. Dopazo (2001). A hierarchical unsupervised growing neural network for clustering gene expression patterns. *Bioinformatics* 17(2), 126–136.
- Hollander, M. and D. A. Wolfe (1999). *Nonparametric Statistical Methods*. Wiley-Interscience.

- Hoon, M. J. L., S. Imoto, J. Nolan, and S. Miyano (2004). Open source clustering software. *Bioinformatics* 20(9), 1453–1454.
- Hubert, L. J. and P. Arabie (1985). Comparing partitions. *Journal of Classification* 2, 193–218.
- Iman, R. L. and J. M. Davenport (1980). Approximations of the critical region of the Friedman statistic. *Comm. Stat. Theor. Meth.* A9(6), 571–595.
- Jain, A. and R. Dubes (1988). *Algorithms for Clustering Data*. Prentice Hall.
- Jain, A., M. Murty, and P. Flynn (1999). Data clustering: A review. *ACM Computing Surveys* 31(3), 264–323.
- Jiang, D., C. Tang, and A. Zhang (2004). Cluster analysis for gene expression data: A survey. *IEEE Transactions on Knowledge and Data Engineering* 16(11), 1370–1386.
- Karypis, G. and V. Kumar (1999). A fast and high quality multilevel scheme for partitioning irregular graphs. *SIAM Journal on Scientific Computing* 20(1), 359–392.
- Kellam, P., X. Liu, N. J. Martin, C. Orengo, S. Swift, and A. Tucker (2001). Comparing, contrasting and combining clusters in viral gene expression data. In *Proceedings of 6th Workshop on Intelligent Data Analysis in Medicine and Pharmacology*, pp. 56–62.
- Kleinberg, J. (2002). An impossibility theorem for clustering. *Advances in Neural Information Processing Systems* 15, 446–453.
- Kohonen, T. (1997). *Self-organizing Maps*. Springer, Berlin.
- Konak, A., D. Coit, and A. Smith (2006). Multi-objective optimization using genetic algorithms: A tutorial. *Reliability Engineering & System Safety* 91(9), 992–1007.
- Krogh, A. and J. Vedelsby (1995). Neural network ensembles, cross validation, and active learning. In *Advances in Neural Information Processing Systems*, Volume 7, pp. 231–238.
- Kuncheva, L. I. (2004). *Combining Pattern Classifiers*. John Wiley & Sons.
- Kuncheva, L. I., S. T. Hadjitodorov, and L. P. Todorova (2002). Experimental comparison of cluster ensemble methods. In *Proceedings of FUSION 2006*, pp. 105–115.
- Law, M., A. Topchy, and A. K. Jain (2004). Multiobjective data clustering. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Volume 2, pp. 424–430.
- Lazzeroni, L. and A. Owen (2002). Plaid models for gene expression data. *Statistica Sinica* 12(1), 61–86.

- LeBlanc, M. and R. Tibshirani (1993). Combining estimates in regression and classification. Technical Report 9318, Department of Statistics, University of Toronto.
- Lorkowski, S. and P. Cullen (Eds.) (2003). *Analysing Gene Expression: A Handbook of Methods Possibilities and Pitfalls*. Wiley-VCH.
- Ma, X.-J., R. Salunga, J. T. Tuggle, J. Gaudet, E. Enright, P. McQuary, T. Payette, M. Pistone, K. Stecker, B. M. Zhang, Y.-X. Zhou, H. Varnholt, B. Smith, M. Gadd, E. Chatfield, J. Kessler, T. M. Baer, M. G. Erlander, and D. C. Sgroi (2003). Gene expression profiles of human breast cancer progression. In *Proc. Natl. Acad. Sci. USA*, Volume 100, pp. 5974–5979.
- MacQueen, J. B. (1967). Some methods for classification and analysis of multivariate observations. In *Proceedings of the 5th Berkeley Symposium on Mathematical Statistics and Probability*, Volume 1: Statistics, pp. 281–297.
- McLachlan, G. and T. Krishnan (1997). *The EM algorithm and extensions*. Wiley Series in Probability and Statistics. New York, NY: John Wiley & Sons.
- Merz, C. J. (1998). *Classification and Regression by Combining Models*. Ph. D. thesis, University of California, Irvine. <http://www.ics.uci.edu/~cmerz/thesis.ps> [Acessado em 19/07/2002].
- Michalewicz, Z. (1996). *Genetic Algorithms + Data Structures = Evolution Programs*. Springer-Verlag.
- Mitchell, T. (1997). *Machine Learning*. McGraw Hill.
- Monti, S., P. Tamayo, J. Mesirov, and T. Golub (2003). Consensus clustering: A resampling-based method for class discovery and visualization of gene expression microarray data. *Machine Learning* 52(1-2), 91–118.
- Narayanan, E. K. A. (2005). *Intelligent Bioinformatics: The Application of Artificial Intelligence Techniques to Bioinformatics Problems*. John Wiley & Sons.
- Newman, D., S. Hettich, C. Blake, and C. Merz (1998). UCI repository of machine learning databases. <http://www.ics.uci.edu/~mllearn/MLRepository.html> [Acessado em 06/07/2006]. University of California, Irvine, Dept. of Information and Computer Sciences.
- Ng, A. Y., M. I. Jordan, and Y. Weiss (2002). On spectral clustering: Analysis and an algorithm. In *Advances in Neural Information Processing Systems*, Volume 14.
- Ng, R. T., J. Sander, and M. C. Sleumer (2001). Hierarchical cluster analysis of SAGE data for cancer profiling. In *Workshop on Data Mining in Bioinformatics (BIOKDD'2001)*, pp. 65–72.

- Nikkilä, J., P. Törönen, S. Kaski, J. Venna, E. Castrén, and G. Wong (2002). Analysis and visualization of gene expression data using self-organizing maps. *Neural Networks, Special issue on New Developments on Self-Organizing Maps* 15(8-9), 953–966.
- Park, Y.-J. and M.-S. Song (1998). A genetic algorithm for clustering problems. In *Proceedings of the Third Annual Conference on Genetic Programming*, San Francisco, CA, USA, pp. 568–575. Morgan Kaufmann Publisher.
- Porter, D. A., I. E. Krop, S. Nasser, D. Sgroi, C. M. Kaelin, J. R. Marks, G. Riggins, and K. Polyak (2001). A SAGE (Serial Analysis of Gene Expression) view of breast tumor progression. *Cancer Research* 61, 5697–5702.
- Qian, Y. and C. Suen (2000). Clustering combination method. In *Proceedings of the International Conference on Pattern Recognition (ICPR'2000)*, Volume II, pp. 736–739.
- Raileanu, L. E. and K. Stoffel (2004). Theoretical comparison between the Gini index and information gain criteria. *Annals of Mathematics and Artificial Intelligence* 1(41), 77–93.
- Rezende, S. O. (2003). *Sistemas Inteligentes - Fundamentos e Aplicações*. Manole.
- Rokach, L. and O. Maimon (2005). Top-down induction of decision trees classifiers - a survey. *IEEE Transactions on Systems, Man and Cybernetics, Part C* 35(4), 476–487.
- Rousseeuw, P. J. (1987). Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. *Journal of Computational and Applied Mathematics* 20, 53–65.
- Ryu, B., J. Jones, N. J. Blades, G. Parmigiani, M. A. Hollingsworth, R. H. Hruban, and S. E. Kern (2002). Relationships and differentially expressed genes among pancreatic cancers examined by large-scale serial analysis of gene expression. *Cancer Research* 62, 819–826.
- Shamir, R. and R. Sharan (2002). *Current Topics in Computational Biology*, Chapter Algorithmic Approaches to Clustering Gene Expression Data, pp. 269–299. MIT Press.
- Sharan, R. and R. Shamir (2000). CLICK: A clustering algorithm with applications to gene expression analysis. In *Proceedings of the 8th International Conference on Intelligent Systems for Molecular Biology (ISMB'2000)*, pp. 307–316.
- Sharkey, A. J. C. (1999). *Combining Artificial Neural Nets: Ensemble and Modular Multi-Net Systems (Perspectives in Neural Computing)*. Springer Verlag.

- Sorlie, T., C. M. Perou, R. Tibshirani, T. Aas, S. Geisler, H. Johnsen, T. Hastie, M. B. Eisen, M. van de Rijn, S. S. Jeffrey, T. Thorsen, H. Quist, J. C. Matese, P. O. Brown, D. Botstein, P. E. Lonning, and A. L. Borresen-Dale (2001). Gene expression patterns of breast carcinomas distinguish tumor subclasses with clinical implications. In *Proc. Natl. Acad. Sci. USA*, Volume 98, pp. 10869–10874.
- Spellman, P., G. Sherlock, M. Zhang, V. Iyer, K. Anders, M. Eisen, P. Brown, D. Botstein, and B. Futcher (1998). Comprehensive identification of cell cycle-regulated genes of the yeast *saccharomyces cerevisiae* by microarray hybridization. *Molecular Biology of the Cell* 9, 3273–3297.
- Srinivas, N. and K. Deb (1994). Multiobjective optimization using nondominated sorting in genetic algorithms. *Evolutionary Computation* 2(3), 221–248.
- Strehl, A. and J. Ghosh (2002). Cluster ensembles - a knowledge reuse framework for combining multiple partitions. *Journal on Machine Learning Research (JMLR)* 3, 583–617.
- Tamayo, P., D. Slonim, J. Mesirov, Q. Zhu, S. Kitareewan, E. Dmitrovsky, E. S. Lander, and T. R. Golub (1999). Interpreting patterns of gene expression with self-organizing maps: Methods and application to hematopoietic differentiation. In *Proc. Natl. Acad. Sci. USA*, Volume 96, pp. 2907–2912.
- Tan, A. C., D. Gilbert, and Y. Deville (2003). Multi-class protein fold classification using a new ensemble machine learning approach. *Genome Informatics* 14, 206–217.
- Topchy, A., A. Jain, and W. Punch (2003). Combining multiple weak clusterings. In *Proceedings of the IEEE International Conference on Data Mining (ICDM'2003)*, Melbourne, Florida, USA, pp. 331–338.
- Topchy, A., A. Jain, and W. Punch (2004). A mixture model for clustering ensembles. In *Proceedings of the SIAM International Conference on Data Mining (SDM'2004)*, Lake Buena Vista, Florida, USA, pp. 331–338.
- Topchy, A., A. K. Jain, and W. Punch (2005). Clustering ensembles: models of consensus and weak partitions. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 27(12), 1866–1881.
- Wang, J. T. L., M. J. Zaki, H. T. T. Toivonen, and D. E. Shasha (Eds.) (2003). *Data Mining in Bioinformatics*. Advanced Information and Knowledge Processing. Springer.
- Weingessel, A., E. Dimitriadou, and K. Hornik (2003). An ensemble method for clustering. In *Distributed Statistical Computing (DSC'2003)*, Wien, Austria. [http:](http://)

-
- [//www.ci.tuwien.ac.at/Conferences/DSC-2003/](http://www.ci.tuwien.ac.at/Conferences/DSC-2003/) [Acessado em 19/01/2004].
- Wilson, R. J. and J. J. Watkins (1990). *Graphs: An Introductory Approach - A First Course in Discrete Mathematics*. John Wiley & Sons.
- Xu, R. and D. Wunsch (2005). Survey of clustering algorithms. *IEEE Transactions on Neural Networks* 16(3), 645–678.
- Yeoh, E.-J., M. E. Ross, S. A. Shurtleff, W. K. Williams, D. Patel, R. Mahfouz, F. G. Behm, S. C. Raimondi, M. V. Relling, A. Patel, C. Cheng, D. Campana, D. Wilkins, X. Zhou, J. Li, H. Liu, C.-H. Pui, W. E. Evans, C. Naeve, L. Wong, and J. R. Downing (2002). Classification, subtype discovery, and prediction of outcome in pediatric acute lymphoblastic leukemia by gene expression profiling. *Cancer Cell* 1(2), 133–143.
- Zeng, Y., J. Tang, J. Garcia-Frias, and G. Gao (2002). An adaptive meta-clustering approach: Combining the information from different clustering results. In *IEEE Computer Society Bioinformatics Conference (CSB'2002)*, Stanford, California, pp. 276–287.
- Zhao, Y. and G. Karypis (2005). Data clustering in life sciences. *Molecular Biotechnology* 31(1), 55–80.
- Zitzler, E. (1999). *Evolutionary Algorithms for Multiobjective Optimization: Methods and Applications*. Ph. D. thesis, Swiss Federal Institute of Technology (ETH), Zurich, Switzerland.
- Zitzler, E., M. Laumanns, and S. Bleuler (2004). A tutorial on evolutionary multi-objective optimization. In K. S. Xavier Gandibleux, Marc Sevaux and V. T'kindt (Eds.), *Metaheuristics for Multiobjective Optimisation*, Volume 535 of *Lecture Notes in Economics and Mathematical Systems*, Berlin, pp. 3–37. Springer Verlag.
- Zitzler, E., M. Laumanns, and L. Thiele (2001). SPEA2: Improving the performance of the strength pareto evolutionary algorithm. Technical Report 103, Computer Engineering and Communication Networks Lab (TIK), Swiss Federal Institute of Technology (ETH) Zurich.
- Zitzler, E. and L. Thiele (1999). Multiobjective evolutionary algorithms: A comparative case study and the strength pareto approach. *IEEE Transactions on Evolutionary Computation* 3(4), 257–271.