Agrupamento visual em grandes conjuntos de dados multidimensionais

Eduardo José Tejada Gamero

Orientador: ***Profa. Dra. Rosane Minghim***

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

"VERSÃO REVISADA APÓS A DEFESA"

Data da Defesa:	19/09/2003
Visto do Orientador:	

USP – São Carlos
Setembro/2003

A meu pai e minha mãe, Eduardo e Ana María.
A meu irmão Enrique.
A meus avôs e avós José, Jesús, Consuelo e Paca.
A minha futura esposa, Paola.

Agradecimentos

A Deus.

A minha orientadora, Professora Dra. Rosane Minghim, minha total gratidão pela oportunidade de trabalhar juntos, a excelente orientação recebida e as experiências partilhadas.

À Professora Dra. Maria Cristina Ferreira de Oliveira, pelos conselhos e apoio sempre oferecidos.

Aos meus pais e meu irmão, porque sem eles não estaria aqui.

A Paola, pelo amor e compreensão.

Aos professores e alunos do Laboratório de Computação de Alto Desempenho do ICMC - USP.

Conteúdo

1 Introdução **1**
1.1 Considerações Iniciais 1
1.2 Motivação e Objetivos 3
1.3 Organização do Trabalho 6

2 Mineração de Dados: Agrupamento **8**
2.1 Considerações Iniciais 8
2.2 Agrupamento de Dados 10
2.2.1 Agrupamento Hierárquico 11
2.2.2 Agrupamento por Particionamento 13
2.3 Agrupamento em Grandes Bancos de Dados Multidimensionais 19
2.4 Considerações Finais 20

3 Agrupamento Visual de Dados **22**
3.1 Considerações Iniciais 22
3.2 Visualização de Informação e Mineração Visual de Dados 22
3.2.1 Visualização de Informação 23
3.2.2 Técnicas de Visualização de Dados Multidimensionais 23
3.2.3 Visualização na Mineração de Dados 29
3.3 Técnicas de Agrupamento Visual de Dados 31
3.3.1 HDEye 31
3.3.2 Um sistema cooperativo humano-computador para agrupamento multidimensional efetivo 36
3.4 Considerações Finais 41

4 HC-Enhanced **43**
4.1 Considerações Iniciais 43
4.2 O algoritmo *HC-Cooperative* 46
4.3 O algoritmo *HC-Enhanced* 54
4.3.1 Passo 1 - Buscas por abrangência com Omni-sequencial 55
4.3.2 Passo 3 - Encontrando e melhorando o subespaço de polarização 56
4.3.3 Passo 5 - Estimativa de densidade 60
4.4 Resultados 63

4.5 Considerações Finais 69

5 Conclusões e Trabalhos Futuros **70**

Referências Bibliográficas **71**

Lista de Figuras

1.1 Ilustração de um agrupamento projetado nos planos (a)(c) XY e (b)(d) XZ dos conjuntos P,Q e R,S. Note-se que na projeção (a) o conjunto P não forma um grupo bem definido enquanto o conjunto Q não forma um grupo bem definido na projeção (b). Os conjuntos R e S não formam bons grupos em nenhuma das projeções. As setas indicam as direções nas quais cada conjunto deveria ser projetado para ser percebido como um grupo (Aggarwal e Yu, 2000). Em cada um desses casos pode se constatar que características importantes que definem os grupos são perdidas nas projeções em que os conjuntos não são percebidos como grupos bem definidos. 5

2.1 Passos do processo de descoberta de conhecimento em bancos de dados (Fayyad et al., 1996). 9
2.2 Diagrama Voronoi no espaço bidimensional. 11
2.3 Exemplo de um dendograma construído com algoritmos de agrupamento hierárquico (Witten e Frank, 1999). 11
2.4 (a) Ponto central q e ponto de fronteira p. (b) q possui mais de $MinPts$ pontos na sua vizinhança, enquanto a vizinhança de p possui (potencialmente) menos de $MinPts$ pontos. Porém, intuitivamente, ambos os pontos devem pertencer ao mesmo grupo (Ester et al., 1996). 15
2.5 (a) p é alcançável a partir de q, porém q não é alcançável a partir de p. (b) p e q são conectados por densidade através de o (Ester et al., 1996) 15

3.1 Exemplos de *scatterplot* (a) 2D e (b) 3D (Grinstein et al., 2001). 24
3.2 *Heatmap* para o conjunto de dados Iris (Grinstein et al., 2001). 25
3.3 Exemplo de um *Heightmap* (Minghim e de Oliveira, 1997). 25
3.4 Tabela de dados explorada com *Table Lens* (Fonte: $http://www.cs.umd.edu/-class/spring2001/cmsc838b/Apps/presentations/Erica_Kolatch/$). . . . 26
3.5 (a) Ícone utilizado para mostrar uma instância (Grinstein et al., 2001). O conjunto de ícones pode definir uma textura (Keim e Ankerst, 2001). 26
3.6 Ilustração do empilhamento dimensional (Keim e Ankerst, 2001). 27
3.7 Exemplo de coordenadas paralelas (Inselberg e Dimsdale, 1987). 27
3.8 Exemplo da técnica de segmento de círculo (Ankerst, 2001). 28
3.9 Visualização utilizando RadViz para os dados Iris (Grinstein et al., 2001). . . . 29

3.10 Visualização utilizando PolyViz para os dados Iris (Grinstein et al., 2001). . . . 29
3.11 Exemplo de grupos com múltiplos centros para diferentes valores de ruído ξ. (a),(b) $\xi = 4$; (c),(d) $\xi = 2$; and (e),(f) $\xi = 1$; (Hinneburg et al., 1999) 33
3.12 Exemplos de projeções (a) gerais e (b) contrativas (Hinneburg et al., 1999) . . . 34
3.13 Exemplo de separador (Hinneburg et al., 1999) 34
3.14 Correspondência entre a covariância e a forma do grupo (Knapp, 1998) 38

4.1 Plano de corte sobre um *heightmap* representando a estimativa de densidade para especificar o nível de ruído λ sobre uma projeção do conjunto de dados Iris. 48
4.2 (a) Densidade estimada em uma das três projeções realizadas para o exemplo. (b) Triangulação de Delaunay (Edelsbrunner, 2001) da projeção. 52
4.3 Busca por abrangência com centro s_q e raio r_q. A região sombreada mostra as instâncias retornadas da poda. (a) Utilizando um foco f_1. (b) Utilizando dois focos f_1 e f_2. 56
4.4 Melhorias das projeções Fastmap usando a abordagem proposta.(a) Conjunto de dados Iris. (b) Um conjunto sintético 10-dimensional com 1000 instâncias. Em ambos os casos a cor mapeia o grupo ao qual a instância pertence. Em (b) mostramos uma ampliação do grupo mapeado em azul. Uma instância classificada como ruído (vermelho) é parte do grupo originalmente. Na projeção melhorada a instância é removida do grupo. 58
4.5 Melhoria de uma projeção do conjunto de dados de exemplo utilizado na Seção 4.2. (a) Iteração 1. (b) Iteração 7. (c) Iteração 13. (d) Iteração 19. (e) Iteração 25. (f) Iteração 31. 60
4.6 Contribuição de uma instância x_i no cálculo da densidade acumulada no ponto da malha x como função do fator de suavização h e a distância d entre x e x_i. . 61
4.7 Contribuição de uma instância x_i no cálculo da densidade acumulada no ponto de malha x como função da distância d entre x e x_i para (a) $h = 0,05$, (b) $h = 0,1$ e (c) $h = 0,3$. 61
4.8 Vizinhança de uma instância. 62
4.9 Estimativa de densidade para o conjunto de dados Iris sobre uma malha 30×30 com vizinhança (a) $v = 2$, (b) $v = 3$, (c) $v = 5$ e (d) $v = 30$. 64
4.10 Comparação da performance entre o *HC-Cooperative* e o *HC-Enhanced* para conjuntos de dados de diferentes tamanhos e 15 dimensões. 65
4.11 Triangulação de Delaunay da projeção Fastmap do conjunto de dados IDH. A altura e a cor mapeiam a classificação do país, sendo que o país melhor classificado é mapeado como azul-baixo e o pior classificado como vermelho-alto. . 66
4.12 Triangulação de Delaunay e estimativa de densidade calculada sobre as projeções (a) original e (b) melhorada de um conjunto de dados sintético com 1000 instâncias 10-dimensionais utilizado para os experimentos realizados. 67

Lista de Tabelas

3.1 Exemplo de $IDstrings$. O elemento em cada célula da tabela identifica o grupo ao qual pertence a instância em tal projeção. É altamente provável que as instâncias 1, 4, e 5 pertençam ao mesmo grupo no espaço multidimensional original. 36

4.1 Características dos algoritmos de agrupamento mais representativos (Tejada e Minghim, 2003). 45
4.2 Exemplo de $IDStrings$. O elemento em cada célula da tabela identifica o grupo ao qual pertence a instância em tal projeção. É altamente provável que as instâncias 1, 4, e 5 pertençam ao mesmo grupo no espaço multidimensional original. 48
4.3 Dados sintéticos de exemplo para ilustrar o algoritmo *HC-Enhanced*. 49
4.4 Grupo normalizado $\mathcal{O}_1$. 50
4.5 Grupo normalizado $\mathcal{O}_2$. 50
4.6 Conjunto de dados projetados. 52
4.7 $IDStrings$ resultantes após as três iterações do nosso exemplo. 53
4.8 Resultados dos algoritmos *HC-Cooperative* e *HC-Enhanced* sobre diferentes conjuntos de dados em termos de precisão do agrupamento e tempo de processamento (Tejada e Minghim, 2003). 68

Resumo

O agrupamento visual de dados é uma abordagem que integra conceitos de visualização de informação e de aprendizado de máquina, especificamente por agrupamento, em um único algoritmo. Isso permite aproveitar a capacidade dos seres humanos para tomar decisões baseados em conhecimento específico de domínio, bem como a capacidade dos computadores para registrar, armazenar, manipular, e recuperar dados.

Este trabalho apresenta a pesquisa realizada nessa linha, a qual tratou problemas específicos presentes nos métodos de agrupamento visual de dados. O principal problema em estudo foi a escalabilidade de algoritmos de agrupamento visual baseados em densidade, propondo meios para acelerar os processos de cálculo de projeções e de estimativa de densidade envolvidos neles.

Como resultado é apresentado o algoritmo *HC-Enhanced*, que possui desempenho bastante superior ao algoritmo *HC-Cooperative* no qual é baseado, além de um dispositivo de melhoria do agrupamento.

Abstract

Visual clustering is an approach that integrates concepts from information visualization and machine learning, specifically clustering, in a single algorithm. This allows combine the capability of humans beings to make decisions based on their domain knowledge, and the capability of the computers for registering, storing, manipulating, and retrieving data.

This work presents the research done on this line, which involved treating specific problems visual clustering approaches present. The main problem studied was the scalability of density-based visual clustering algorithms, proposing means for accelerating the projection and density estimation processes involved in them.

As a result it is presented the algorithm HC-Enhanced, whose performance is considerably improved compared to the algorithm HC-Cooperative on which it is based, as well as a component for clustering quality improvement.

CAPÍTULO

1

Introdução

1.1 Considerações Iniciais

A extração de conhecimento relevante a partir de bancos de dados tem se tornado uma tarefa extremamente difícil devido à enorme quantidade de dados multidimensionais disponíveis como resultado do avanço nas tecnologias de coleta e armazenamento. Os dados coletados e acumulados diariamente contêm conhecimento implícito potencialmente útil que dificilmente é descoberto através de inspeção manual.

Os algoritmos de *aprendizado de máquina* apoiam na tarefa de extração desse conhecimento, permitindo a descoberta automática de estruturas relevantes (modelos ou padrões) ocultas nos dados originais. Diferentes algoritmos de aprendizado de máquina utilizam diferentes abordagens para tornar os dados compreensíveis e fornecer um conjunto mais rico de informação agregada. Seu uso varia de aplicação a aplicação e depende da saída requerida e do conhecimento a ser descoberto. Existem algoritmos de aprendizado de máquina para agrupamento, para previsão numérica e para extrair regras de associação e de classificação (Witten e Frank, 1999).

As *regras de classificação* permitem classificar as instâncias de um banco de dados em um conjunto de classes predefinidas. Para isso, um processo de treinamento deve ser realizado, no qual um conjunto de instâncias de treinamento é utilizado para construir as estruturas de classificação (Witten e Frank, 1999). Tais estruturas são geradas a partir dos valores dos atribu-

tos (incluindo o atributo que indica a classe à qual pertence a instância), inferindo as regras que serão usadas para encontrar o valor desconhecido da classe de novas instâncias.

As *regras de associação* são uma generalização das regras de classificação. A principal diferença é que o valor de qualquer atributo pode ser previsto depois da fase de treinamento. Ao contrário das regras de classificação, geralmente o processo de treinamento não possui um atributo alvo a ser previsto. O processo de extração de regras de associação pode elucidar relações ocultas entre a ocorrência de diferentes atributos e fornecer regras que cumprem certos níveis de *confiança* e *suporte* (Witten e Frank, 1999). O *suporte* é o número de instâncias que cumprem as pré-condições da regra e a *confiança* é o número de instâncias que cumprem as pré-condições e as pós-condições. Essas métricas fornecem uma estimativa da relevância das associações descobertas.

A *previsão numérica* utiliza regressão linear (Witten e Frank, 1999) para prever valores de atributos numéricos de instâncias pertencentes a um conjunto de dados. Assim como nas regras de classificação e de associação, um conjunto de treinamento é utilizado para construir a estrutura de conhecimento para posterior análise e aplicação em novas instâncias.

Finalmente, o objetivo dos algoritmos de *agrupamento de dados* é identificar grupos significativos de objetos similares. O resultado desse processo consiste de duas partes. Primeiro, os grupos por si próprios fornecem uma idéia da distribuição das instâncias. Segundo, informação relativa a cada grupo, por exemplo seu centróide [1], pode ser usada para posterior classificação de novas instâncias.

Os processos de agrupamento baseiam-se em uma função distância para medir o grau de similaridade entre as instâncias. Portanto, definir a função distância é uma parte importante do processo de agrupamento na qual o conhecimento específico sobre o domínio tem um papel relevante (Berkhin, 2002).

Na *mineração visual de dados*, ou *mineração visual*, os algoritmos de aprendizado de máquina são assistidos por técnicas de *visualização de informação* (ver Capítulo 3) segundo três abordagens que pretendem apoiar: (1) a interpretação da estrutura dos dados originais, (2) a interpretação dos resultados da mineração de dados ou (3) a intervenção do usuário durante estágios intermediários do processo de mineração (Ankerst, 2001).

Existe uma considerável quantidade de trabalhos relacionados com a primeira dessas abordagens. Como conseqüência disso, Keim (Keim, 2002) propôs uma categorização das técnicas de mineração visual de dados pertencentes a ela, baseada no tipo de dado a ser visualizado, na técnica de visualização e nas técnicas de interação e distorção utilizadas [2].

[1] O centróide de um grupo é, geralmente, a média aritmética das instâncias que pertencem a ele.

[2] As técnicas de interação e distorção permitem aos usuários interagir diretamente com a visualização e modificar a cena.

A segunda abordagem utiliza técnicas de visualização de informação para apresentar os resultados finais dos algoritmos de mineração. O papel dos algoritmos de visualização é relegado ao final do processo, existindo pouco acoplamento com os algoritmos de mineração. Pode-se pensar nessa abordagem como sendo um *pipeline* com as saídas do processo de mineração constituindo as entradas do processo de visualização.

A última dessas abordagens apóia uma interação cíclica entre o usuário e o computador, permitindo ao usuário controlar parâmetros ao longo do processo e tomar decisões que geralmente são controladas algoritmicamente utilizando algum tipo de heurística (Wong, 1999). Isso é feito para aproveitar a capacidade dos seres humanos para tomar decisões baseados em conhecimento específico do domínio, bem como a capacidade de processamento dos computadores.

Recentemente, uma grande quantidade de pesquisa tem sido dirigida a técnicas pertencentes a essa última abordagem, na qual se insere este trabalho. Além de fornecer um meio efetivo de interação com o usuário, essas técnicas devem lidar com o problema do tamanho e da dimensionalidade (i.e., o número de atributos) dos conjuntos de dados disponíveis atualmente.

Este trabalho apresenta, num primeiro momento, o estado da arte dos algoritmos de agrupamento e das abordagens visuais para apoiar o processo de agrupamento e visualizar seus resultados. Será apresentada, logo depois, a pesquisa realizada para contribuir na solução dos problemas de escalabilidade das técnicas de agrupamento visual baseadas em densidade, assim como mecanismos para melhorar os resultados do agrupamento realizado com tais técnicas.

1.2 Motivação e Objetivos

O agrupamento de dados é uma área aberta e controversa pois a definição do que seria um bom agrupamento depende da aplicação e contexto para o qual o agrupamento é realizado. As variáveis e critérios usados para agrupar as instâncias mudam segundo a aplicação, sendo que um agrupamento pode ser considerado bom para uma aplicação e ruim para outra. Devido a esse fato, é sempre necessário utilizar o conhecimento do domínio do usuário para assegurar que o agrupamento esteja de acordo com as necessidades específicas da aplicação em consideração.

É nesse ponto que o *agrupamento visual de dados* entra em cena, permitindo ao usuário tomar decisões que são algoritmicamente custosas, ou mesmo impossíveis sem intervenção humana. Isso é feito através da utilização de técnicas de visualização durante o agrupamento, de forma que a intervenção humana seja permitida ao longo de todo o processo.

Por outro lado, explorar grandes volumes de dados multidimensionais é uma grande necessidade na atualidade. Alguns exemplos reais de grandes conjuntos de dados multidimensionais são: bancos de dados de bioinformática, bancos de dados de transações em comércio eletrônico

e bancos de dados de transações bancárias. As técnicas de agrupamento têm a capacidade de apoiar a descoberta de conhecimento sobre esses dados. Porém, o fato desses conjuntos de dados serem multidimensionais gera um problema conhecido como a "maldição da alta dimensionalidade", o qual refere-se à incapacidade de realizar um agrupamento significativo em conjuntos de dados que possuem muitos atributos. Para resolver esse problema, existem várias abordagens que utilizam diferentes algoritmos de projeção para reduzir a dimensionalidade dos dados de forma que seja possível aplicar algoritmos de agrupamento espacial conhecidos, bem como para permitir a validação e especificação visual dos grupos por parte do usuário (Aggarwal, 2002; Aggarwal e Yu, 2000; Aggarwal, 2001; Hinneburg e Keim, 1998; Hinneburg et al., 1999; Ribarsky et al., 1999; Xu et al., 1998).

Projeções gerais[3] e *seleção de características*[4] (Witten e Frank, 1999) poderiam ser usadas para ajudar a encontrar grupos em bancos de dados multidimensionais. Porém, nem sempre são adequadas, devido ao fato de que algumas das características importantes dos dados podem ser perdidas (Aggarwal e Yu, 2000), como ilustrado na Figura 1.1.

Porém, em algumas situações é melhor procurar grupos em espaços projetados de baixa dimensão do que tentar encontrá-los no espaço multidimensional original (Aggarwal, 2002). Adicionalmente, algumas instâncias podem se agrupar bem em um espaço projetado, enquanto que outras agrupam-se bem em outro espaço projetado como mostrado na Figura 1.1 (Aggarwal e Yu, 2000).

Ciente desses fatos, Aggarwal (Aggarwal, 2001) adotou uma abordagem interativa na qual procuram-se os grupos em diferentes projeções bidimensionais do conjunto de dados. O algoritmo começa escolhendo um conjunto de pontos aleatórios, denominados *âncoras*. A partir deles constrói grupos formados pelos k-vizinhos mais próximos a cada âncora. Uma vez formados os grupos, o algoritmo tenta identificar o subespaço bidimensional que minimize o *momento*[5] total dos pontos pertencentes a eles. Depois de que tal projeção foi encontrada, calcula uma estimativa da densidade do conjunto de pontos [6].

Uma vez calculada a estimativa de densidade, ela é mapeada em um *heigthmap* ou *colormap* (ver Seção 3.2) de modo que o usuário possa especificar os grupos interativamente estabelecendo um nível de ruído. O nível de ruído indica quão denso deve ser um conjunto de pontos

[3]Uma *projeção geral* é uma projeção sobre uma base ortonormal de dimensão menor do que a do espaço original.

[4]A *seleção de características* é um processo, geralmente manual, durante o qual escolhe-se um número reduzido de atributos representativos pertencentes às instâncias do conjunto de dados.

[5]O *momento* é uma métrica que define quão esparso é o grupo. Uma definição formal é apresentada na Seção 3.3.

[6]Informalmente, a densidade de um conjunto de dados espacial é a quantidade de pontos (instâncias) por unidade espacial. Tal densidade não é constante ao longo do domínio espacial, mas varia de acordo com a distribuição das instâncias. Uma definição formal de estimativa de densidade é apresentada na Seção 3.3.

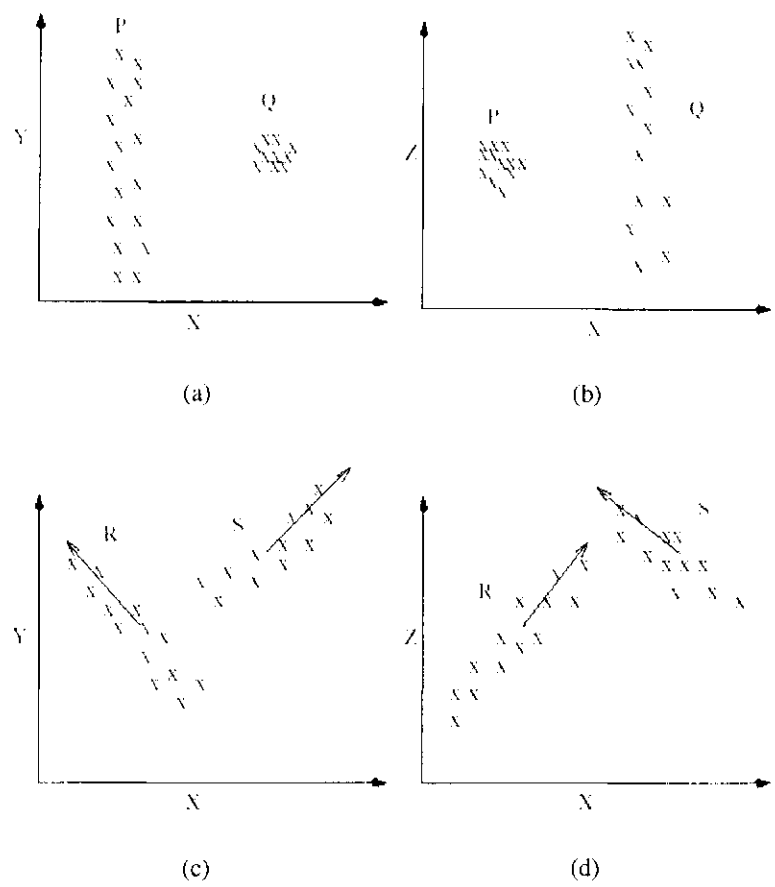


Figura 1.1: Ilustração de um agrupamento projetado nos planos (a)(c) XY e (b)(d) XZ dos conjuntos P,Q e R,S. Note-se que na projeção (a) o conjunto P não forma um grupo bem definido enquanto o conjunto Q não forma um grupo bem definido na projeção (b). Os conjuntos R e S não formam bons grupos em nenhuma das projeções. As setas indicam as direções nas quais cada conjunto deveria ser projetado para ser percebido como um grupo (Aggarwal e Yu, 2000). Em cada um desses casos pode se constatar que características importantes que definem os grupos são perdidas nas projeções em que os conjuntos não são percebidos como grupos bem definidos.

agrupados para de fato ser considerado um grupo. Dessa forma, diferentes níveis de ruído podem ser testados para assegurar um bom agrupamento. Finalmente, os grupos são classificados segundo o número de vezes que aparecem nas projeções.

Hinneburg *et al.* (Hinneburg et al., 1999) adotaram uma abordagem similar. As diferenças em relação à abordagem de Aggarwal são duas: (1) utiliza-se uma única projeção, obtida com um refinamento e cruzamento iterativo de um conjunto de projeções iniciais em um processo dirigido pelo usuário; e (2) os grupos são especificados pelo usuário utilizando *separadores* (ver Seção 3.3), ao invés de ser encontrados algoritmicamente. O processo envolve o cálculo da

estimativa de densidade para o conjunto inicial de projeções a partir das quais será encontrada a projeção final (a melhor intuitivamente).

Essas duas abordagens não são escaláveis a grandes conjuntos de dados devido ao enorme tempo de processamento gasto na determinação das projeções e no cálculo da estimativa de densidade.

Porém, são características importantes do agrupamento baseado em densidade: (1) permitir a criação de grupos de formatos arbitrários em conjuntos de dados espaciais, e (2) permitir um mapeamento direto da densidade estimada da projeção bidimensional em um *colormap* ou *heightmap* para apresentar uma visualização da distribuição das instâncias, permitindo dessa forma uma especificação dos grupos dirigida pelo usuário. Tais características, e o fato de que a densidade é preservada nas transformações de redução da dimensão (sendo igual ou maior à do espaço original), são as razões pelas quais concentramos a pesquisa realizada nas abordagens de agrupamento visual baseado em densidade.

O problema de construir um algoritmo para agrupamento cooperativo usuário-computador aplicável a grandes conjuntos de dados pode ser sumarizado em três grandes tópicos: (1) permitir a determinação de grupos em múltiplas projeções; (2) preservar as características dos pontos de dados nos subespaços projetados bidimensionais; e (3) fazer o algoritmo escalável. Esse último requerimento implica em desenvolver um mecanismo para reduzir a quantidade de tempo e memória gastos em encontrar as n projeções, em calcular a estimativa de densidade e em construir os grupos a partir das respostas do usuário armazenadas durante o processo.

Portanto, o objetivo da pesquisa apresentada neste trabalho foi propor uma estratégia visando melhorar a escalabilidade dos algoritmos de agrupamento visual baseados em densidade, assim como desenvolver mecanismos para melhorar seus resultados quando os grupos se encontram muito próximos entre si. O algoritmo resultante desse esforço, denominado *HC-Enhanced*, é uma melhoria do algoritmo *HC-Cooperative* no qual é baseado, em termos de desempenho e de qualidade dos grupos resultantes.

1.3 Organização do Trabalho

O restante deste documento é organizado da seguinte forma: o Capítulo 2 é dedicado a apresentar conceitos de mineração de dados e os algoritmos de agrupamento mais relevantes; no Capítulo 3 apresentam-se técnicas de visualização de informação para mineração visual de dados, assim como trabalhos existentes em mineração visual, com forte ênfase em agrupamento visual; no Capítulo 4, apresentamos um algoritmo de agrupamento visual baseado em densidade

aplicável a grandes conjuntos de dados baseado no algoritmo *HC-Cooperative* apresentado no Capítulo 3; e finalmente, no Capítulo 5, apresentamos as conclusões e trabalhos futuros.

CAPÍTULO

2

Mineração de Dados: Agrupamento

2.1 Considerações Iniciais

A crescente capacidade das tecnologias de coleta e armazenamento de dados gera uma enorme quantidade de informação difícil de explorar manualmente. Esses dados contêm relações e estruturas potencialmente úteis que permitiriam uma melhor compreensão do fenômeno que os originou (M.-S. Chen, 1996). Nesse contexto aparece a *mineração de dados*, a qual é definida como um processo de extração de informação implícita não trivial, previamente desconhecida e potencialmente útil, contida em bancos de dados (Piatetsky-Shapiro e Frawley, 1991).

A mineração de dados e a *descoberta de conhecimento em bancos de dados* são áreas em rápida evolução e que constituem a intersecção de várias disciplinas como estatística, inteligência artificial, bancos de dados, visualização, e computação de alto desempenho e paralela (Fayyad, 1998).

Fayyad *et al.* (Fayyad et al., 1996) definem os termos descoberta de conhecimento em bancos de dados e mineração de dados como segue.

"**Descoberta de Conhecimento em Bancos de Dados** é o processo de identificar estruturas válidas, novas, potencialmente úteis e compreensíveis, nos dados. Os dados são um conjunto de fatos (casos em um banco de dados) e o termo estrutura se refere a padrões ou modelos. Um padrão é uma expressão representando uma descrição de um subconjunto do banco de dados.

Um modelo é uma representação da fonte que gera os dados. O termo processo implica que a descoberta de conhecimento em bancos de dados é formada por passos (Figura 2.1): preparação dos dados, busca de padrões, avaliação do conhecimento e refinamento. Este processo é potencialmente repetido em múltiplas iterações."

"A **Mineração de Dados** é um passo no processo de descoberta de conhecimento que, sob limitações de eficiência computacional aceitáveis, enumera as estruturas (padrões ou modelos) dos dados."

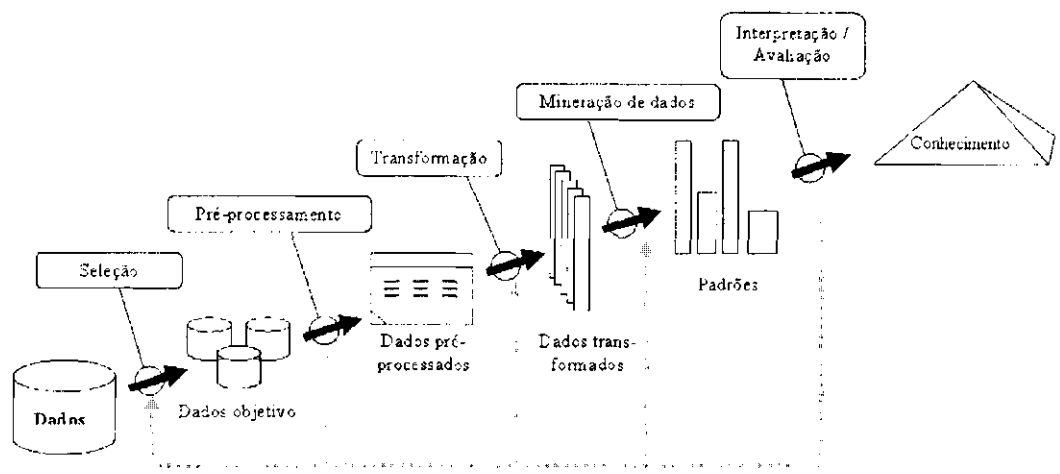


Figura 2.1: Passos do processo de descoberta de conhecimento em bancos de dados (Fayyad et al., 1996).

Os algoritmos de aprendizado de máquina constituem um meio através do qual se realiza a mineração de dados. Diferentes algoritmos de aprendizado de máquina utilizam diferentes abordagens para tornar os dados compreensíveis e fornecer um conjunto mais rico de informação agregada. Existem algoritmos de aprendizado de máquina para extrair regras de associação e classificação, para agrupamento e para previsão numérica (Witten e Frank, 1999) conforme definidas no Capítulo 1. Dentre eles, nos concentramos em algoritmos de agrupamento, especialmente aqueles orientados a grandes bancos de dados (VLDB, *Very Large DataBases*) multidimensionais.

O agrupamento em grandes bancos de dados tem sido o foco de diferentes trabalhos de pesquisa (Aggarwal e Yu, 2000; Ganti et al., 1998, 1999; Xu et al., 1998; Zhang et al., 1996; Berkhin, 2002). Bancos de dados de transações por Internet e de dados de Bioinformática são exemplos claros de grandes bancos de dados multidimensionais, com número de atributos da ordem de centenas e número de instâncias da ordem de centenas de milhares, dos quais a aplicação de agrupamento ajudaria a extrair informação sobre os grupos naturais que esses dados formam. Uma vez gerados os grupos, é possível classificar novas instâncias e inferir suas propriedades a partir das propriedades dos representativos de cada grupo.

Este capítulo descreve o agrupamento de dados, apresentando algoritmos hierárquicos e de particionamento na Seção 2.2, assim como uma discussão do agrupamento em grandes bancos de dados multidimensionais, o qual é tratado na Seção 2.3. Na Seção 2.4 são apresentadas as conclusões e considerações finais do capítulo.

2.2 Agrupamento de Dados

Agrupamento é o processo de encontrar grupos em um conjunto de instâncias de forma que as instâncias dentro de um grupo sejam similares. Esse é um problema que tem sido estudado tanto pela comunidade de bancos de dados quanto pela da estatística. Existem diferentes problemas a serem resolvidos usando agrupamento e, portanto, diferentes abordagens para tratar dados espaciais, métricos e vetoriais. Cada uma dessas abordagens possui características particulares, e diferentes soluções têm sido propostas para resolver os problemas que elas apresentam.

É possível classificar os algoritmos de agrupamento em dois grupos principais: algoritmos de particionamento e algoritmos hierárquicos (Xu et al., 1998). Os algoritmos de particionamento realizam o agrupamento "particionando" o banco de dados em um conjunto de grupos. Usualmente os algoritmos de particionamento começam com uma partição inicial que é otimizada ao longo de um processo iterativo. Cada grupo é representado pelo seu centro de gravidade (centróide) ou pela instância mais próxima a tal centro (medóide). O centróide ou medóide do grupo é denominado *representativo*.

Os algoritmos de particionamento geralmente adotam um procedimento de dois passos em cada iteração. Primeiro, eles determinam k representativos minimizando o *momento* total dos grupos, onde k é um parâmetro geralmente especificado pelo usuário. Segundo, eles atribuem cada objeto ao grupo que possui o representativo mais próximo a ele. O segundo passo gera uma partição equivalente a um diagrama de Voronoi[1] (Figura 2.2) onde cada grupo é contido em uma das células do diagrama (Xu et al., 1998).

Os algoritmos hierárquicos (Witten e Frank, 1999) criam uma decomposição do banco de dados representada por um dendograma (Figura 2.3). Cada nível da árvore representa um agrupamento do conjunto de dados D. Tal hierarquia é obtida através de um processo *bottom-up*, realizando junção de grupos similares para formar grupos maiores, ou *top-down*, começando com um único grupo que é dividido recursivamente caso o momento total seja maior a um parâmetro definido pelo usuário; i.e., o grupo ainda seja muito esparso. Em contraste à maioria dos algoritmos de particionamento, os algoritmos hierárquicos não têm k como parâmetro de

[1]Um *diagrama de Voronoi* de um espaço $\mathcal{E}$ dado um conjunto de pontos $\mathbf{P}$, é uma partição de $\mathcal{E}$ em regiões ou células tal que a região associada a um ponto $p_i \subset \mathbf{P}$ é formada por todos o pontos do espaço $\mathcal{E}$ que se encontram mais próximos a p_i do que a qualquer outro ponto em $\mathbf{P}$.

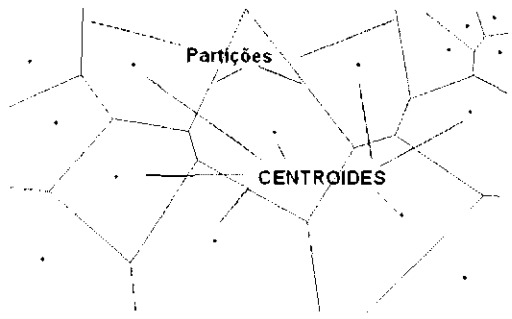


Figura 2.2: Diagrama Voronoi no espaço bidimensional.

entrada; i.e., não exigem que se defina *a priori* o número de grupos. Porém, uma condição de parada deve ser especificada indicando quando o processo de junção ou divisão deve parar.

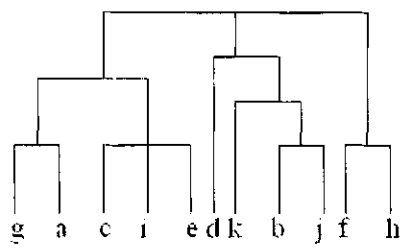


Figura 2.3: Exemplo de um dendograma construído com algoritmos de agrupamento hierárquico (Witten e Frank, 1999).

Existe uma considerável quantidade de trabalhos em agrupamento. Desse corpo de resultados, alguns trabalhos se destacam pela originalidade da abordagem ou pela classe de problemas que resolvem. Neste texto, os principais trabalhos de agrupamento segundo essa perspectiva são descritos resumidamente. Analisaremos os algoritmos apresentados principalmente sob dois pontos de vista: (1) a capacidade de realizar agrupamento em grandes bancos de dados; e (2) a capacidade de agrupar dados multidimensionais. Além disso, será discutida a capacidade dos algoritmos lidarem com *outliers*[2] e criarem grupos de formatos arbitrários. Essa análise permitirá justificar a escolha do algoritmo de Aggarwal (Aggarwal, 2001) como base da pesquisa realizada.

2.2.1 Agrupamento Hierárquico

Os algoritmos de agrupamento hierárquico podem ser divididos em duas sub-categorias: *aglomerativos* e *de divisão* (Berkhin, 2002). Os algoritmos aglomerativos são abordagens *bottom-up*, que começam com grupos formados por uma única instância (*singletons*) juntando-os recursivamente para formar grupos maiores. Já, os algoritmos de divisão são abordagens *top down* que

[2]Instâncias que não podem ser incluídas em nenhum grupo.

começam com um único grupo formado por todas as instâncias no conjunto de dados, o qual é dividido recursivamente até que uma condição de parada é atingida.

CURE é um algoritmo aglomerativo proposto por Guha *et al.* (Guha et al., 1998) que permite construir grupos de formatos arbitrários usando um número fixo de instâncias para representar cada grupo. A distância inter-grupos usada no processo aglomerativo é a distância entre os dois representativos (um de cada grupo) mais próximos. Esse algoritmo trata o efeito dos *outliers* permitindo ao usuário especificar um parâmentro α para realizar um "encolhimento" dos representativos em direção ao centróide do grupo. Como o algoritmo CURE realiza uma amostragem do conjunto de dados, sua complexidade é proporcional ao tamanho da amostra na forma $O(N^2_{amostra})$.

O fato do algoritmo utilizar amostragem gera dois problemas: (1) determinar o tamanho e o critério de escolha para a amostra; e (2) a aplicabilidade do algoritmo depende fortemente do tamanho da amostra, sendo que para grandes bancos de dados uma amostra suficientemente pequena para poder ser processada pelo algoritmo significaria uma perda de qualidade significativa no agrupamento.

Além disso, por não realizar projeções, o algoritmo não é aplicável a bancos de dados de alta dimensionalidade. Isso porque, matematicamente, nos conjuntos de dados de alta dimensionalidade as buscas pelo vizinho mais próximo se tornam instáveis: a distância ao vizinho mais próximo pode se tornar indistinguível da distância à maioria dos pontos (Beyer et al., 1999). Esse efeito torna-se severo para dimensões superiores a 15 para a maioria dos algoritmos de agrupamento (Berkhin, 2002). Portanto, a construção de grupos baseada no conceito de proximidade é duvidosa em tais situações.

Karypis *et al.* (Karypis et al., 1999) desenvolveram um algoritmo aglomerativo que utiliza modelagem dinâmica na agregação dos grupos. O processamento é feito em dois estágios. O primeiro constrói pequenos grupos que serão usados no segundo estágio para a aglomeração utilizando medidas de conectividade e proximidade inter-grupos; i.e., determinando quando dois subgrupos se encontram suficientemente próximos para serem considerados um único grupo maior. A complexidade desse algoritmo é $O(Nm + Nlog(N) + m^2log(m))$ onde m é o número de grupos iniciais construídos no primeiro estágio e N é o número de instâncias do conjunto de dados. Esse algoritmo não pode ser aplicado a grandes bancos de dados devido à sua complexidade, nem a conjuntos de alta dimensionalidade pelo fato de utilizar conceitos de proximidade no espaço multidimensional original.

BIRCH (Zhang et al., 1996) é um algoritmo que utiliza uma estrutura de tipo árvore para adicionar instâncias incrementalmente aos grupos, juntando-os caso eles satisfaçam uma condição de similaridade, dividindo-os caso o momento total do grupo ultrapasse um limite superior especificado pelo usuário, e usando informação nos nós não-folha para guiar a busca do grupo para

uma nova instância. Os grupos propriamente ditos são armazenados nos nós folha. BIRCH é um algoritmo escalável capaz de encontrar grupos em grandes bancos de dados pelo fato de trabalhar com informação reduzida sobre os grupos, denominada *características do grupo* (*cluster features* - CF). Trabalhar com informação reduzida assegura um processo ágil e de baixo custo computacional. Porém, uma desvantagem dessa abordagem é o fato de que a decisão de incluir uma nova instância em um grupo é feita localmente e, portanto, um agrupamento ótimo não pode ser assegurado. Além disso, o fato dele trabalhar com conceitos de proximidade no espaço multidimensional original faz com que o algoritmo não seja aplicável a conjuntos de dados multidimensionais.

2.2.2 Agrupamento por Particionamento

O agrupamento por particionamento usualmente realiza um refinamento incremental dos grupos, deslocando os representativos em cada iteração para minimizar uma função energia na direção da partição ótima. Devido ao fato desse problema ser NP-completo[3], os algoritmos de particionamento usam diferentes heurísticas para obter uma solução *quasi*-ótima.

Ganti *et al.* (Ganti et al., 1998, 1999) desenvolveram uma abordagem para agrupar espaços métricos arbitrários baseada apenas na distância entre as instâncias, sem necessidade de utilizar sua informação vetorial (valores dos atributos). Isto é, o algoritmo é aplicável a conjuntos de dados métricos não necessariamente espaciais. Essa abordagem permite agrupar conjuntos de dados arbitrariamente grandes através de um processo construtivo, lendo os dados do disco e adicionando iterativamente as novas instâncias aos grupos mantidos em uma estrutura de tipo árvore. A abordagem permite um rápido acesso aos grupos e requerimentos baixos de memória. Porém, ela possui uma desvantagem comum às abordagens apresentadas na seção anterior: não pode ser aplicada a dados multidimensionais pelo fato dela trabalhar com medidas de proximidade definidas no espaço original.

Aggarwal e Yu (Aggarwal e Yu, 2000) propuseram um método para encontrar grupos em bancos de dados multidimensionais. A abordagem é baseada na suposição de que diferentes grupos de instâncias agrupam-se bem em diferentes espaços projetados. Adicionalmente, alguns grupos encontrados em espaços projetados podem não ser encontrados no espaço multidimensional original. Para encontrar as projeções nas quais as instâncias formam grupos bem definidos, é necessário computar as componentes principais do espaço de dados utilizando a *análise de componentes principais* (PCA - *Principal Components Analysis*) (Jolliffe, 1986), e depois selecionar os autovetores associados aos l menores autovalores para ser a base ortonor-

[3]Um problema NP-completo é um problema que pode ser verificado (mas não resolvido) por um algoritmo determinístico em tempo polinomial. O nome NP significa "tempo polinomial não determinístico" (Cormen et al., 1990).

mal do espaço reduzido l-dimensional. O algoritmo reduz incrementalmente a dimensionalidade e o número de grupos formados, começando com um conjunto de centróides escolhidos aleatoriamente, chamados *sementes* ou *âncoras*. Essa abordagem não é escalável devido ao custo computacional do PCA.

No *agrupamento baseado em densidade*, a vizinhança de uma instância é usada para encontrar regiões densas nas quais existem grupos, utilizando conceitos de densidade, conectividade e bordo (Ester et al., 1996, 1997; Hinneburg e Keim, 1998; Sander et al., 1998). Isso permite que os algoritmos baseados em informação de densidade sejam capazes de descobrir grupos de formatos arbitrários. Tais algoritmos requerem informação espacial para computar a densidade, conectividade e informação de bordo. Para viabilizar os cálculos, algum mecanismo de indexação deve ser construído.

DBSCAN (Ester et al., 1996) e GDBSCAN (Sander et al., 1998) são algoritmos que se baseiam em uma estrutura R*-tree (Beckmann et al., 1990) para indexar os dados, reduzindo a complexidade de acesso a $O(log(N))$. Teoricamente, a complexidade total desses algoritmos é $O(Nlog(N))$. Eles serão discutidos em maior detalhe porque embutem idéias que fornecem a base para os algoritmos baseados em densidade que são o foco deste trabalho. Ambos utilizam conceitos de vizinhança e connetividade que são apresentados a seguir (Ester et al., 1996; Sander et al., 1998).

DEFINIÇÃO 1 (Eps-vizinhança de um ponto) Seja D um conjunto de pontos em um espaço multidimensional e Eps um parâmetro definido pelo usuário. A *Eps-vizinhança* de um ponto $p \in D$, denotada por $N_{Eps}(p)$, é definida por $N_{Eps}(p) = \{q \in D | dist(p,q) \leq Eps\}$.

Uma primeira abordagem para definir os grupos requer que qualquer ponto em um grupo possua um número mínimo ($MinPts$) de pontos na sua Eps-vizinhança. O problema dessa abordagem é apresentado pelos *pontos de fronteira*. Os pontos de fronteira, ao contrário dos *pontos centrais*, são aqueles que se encontram no bordo do grupo, como mostrado na Figura 2.4. Na figura observamos que o ponto p não tem mais de $MinPts$ na sua vizinhança. Porém, intuitivamente p deve pertencer ao mesmo grupo ao qual pertence q.

Para solucionar esse problema, o conceito de grupo é suportado pelas seguintes definições.

DEFINIÇÃO 2 (Diretamente alcançável por densidade) Um ponto p é *diretamente alcançável por densidade* a partir de um ponto q dados Eps e $MinPts$, se

1. $p \in N_{Eps}(q)$, e
2. $|N_{Eps}(q)| \geq MinPts$.

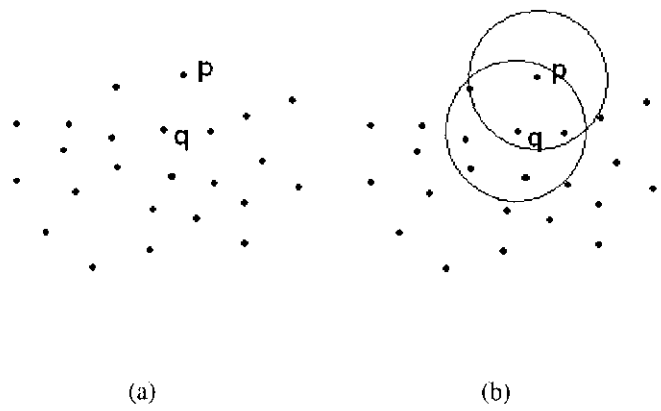


Figura 2.4: (a) Ponto central q e ponto de fronteira p. (b) q possui mais de $MinPts$ pontos na sua vizinhança, enquanto a vizinhança de p possui (potencialmente) menos de $MinPts$ pontos. Porém, intuitivamente, ambos os pontos devem pertencer ao mesmo grupo (Ester et al., 1996).

DEFINIÇÃO 3 (Alcançável por densidade) Um ponto p é *alcançável por densidade* a partir de um ponto q, dados Eps e $MinPts$, se existe uma cadeia de pontos $p_1,...,p_n$, $p_1 = q$, $p_n = p$, tais que p_{i+1} é diretamente alcançável por densidade a partir de p_i, $\forall i = 1, ..., n-1$.

DEFINIÇÃO 4 (Conectado por densidade) Um ponto p é *conectado por densidade* a um ponto q, dados Eps e $MinPts$, se existe um ponto o a partir do qual tanto p quanto q são alcançáveis por densidade.

Esses dois últimos conceitos são ilustrados na Figura 2.5.

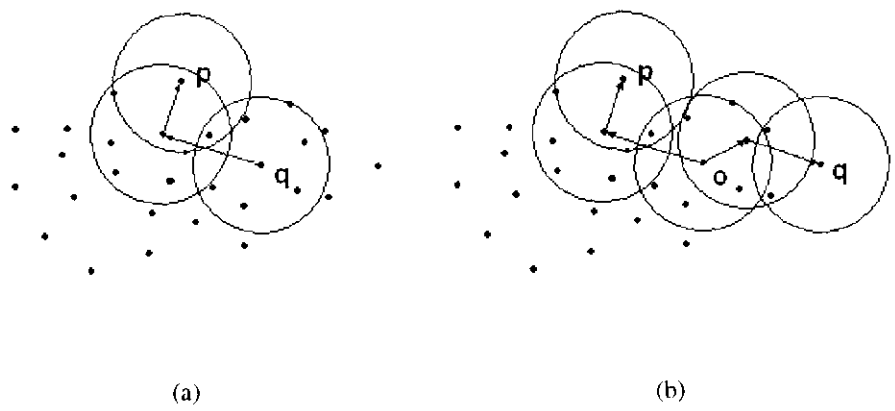


Figura 2.5: (a) p é alcançável a partir de q, porém q não é alcançável a partir de p. (b) p e q são conectados por densidade através de o (Ester et al., 1996)

A partir dessas definições podemos apresentar a definição de grupo adotada pelos algoritmos DBSCAN e GDBSCAN.

DEFINIÇÃO 5 (Grupo) Seja D um conjunto de pontos. Um *grupo* C, dados Eps e $MinPts$, é um subconjunto não vazio de D que satisfaz as seguintes condições:

1. $\forall p, q$: se $p \in C$ e q é diretamente alcançável a partir de p, dados Eps e $MinPts$, então $q \in C$. (Maximalidade).

2. $\forall p, q \in C$: p é conectado por densidade a q, dados Eps e $MinPts$. (Conectividade).

Diretamente dessa definição pode-se derivar a definição de ruído.

DEFINIÇÃO 6 (Ruído) Sejam C_1,...,C_k os grupos presentes no conjunto de pontos D, dados Eps_i e $MinPts_i$, i=1,...,k. Definimos *ruído* como o subconjunto de pontos de D que não pertencem a nenhum grupo C_i, i.e. $ruido = \{p \in D | \forall i : p \notin C_i\}$

Note que qualquer grupo C_i, dados Eps_i e $MinPts_i$, contém pelo menos $MinPts_i$ pontos.

Utilizando essas definições podemos apresentar o algoritmo do DBSCAN (Ester et al., 1996).

Algoritmo 1

```
DBSCAN(ConjuntoDePontos, Eps, MinPts)
inicio //IdGrupo inicial de ConjuntoDePontos: NÃOCLASSIFICADO
    IdGrupo := SeguinteId(RUIDO); //Gera um novo ID a partir do ID atual
    para i de 1 a ConjuntoDePontos.tamanho fazer
    inicio  Ponto := ConjuntoDePontos.devolver(i); //Devolve o ponto i
        se Ponto.ClId = NÃOCLASSIFICADO então
        inicio
          se ExpandirGrupo(ConjuntoDePontos, Ponto, IdGrupo, Eps, MinPts) então
          //se foi gerado um novo grupo contendo os pontos
          //alcançaveis por densidade a partir de Ponto
          inicio
             IdGrupo := ProximoId(IdGrupo);//gera um novo ID
          fim
        fim
    fim
fim;
```

Um dos principais passos nesse algoritmo é *ExpandirGrupo*, no qual adiciona-se ao grupo $IdGrupo$ os pontos conectados por densidade a $Ponto$. O pseudocódigo desse procedimento é apresentado a seguir.

Algoritmo 2

```
ExpandirGrupo(ConjuntoDePontos, Ponto, IdCluster, Eps, MinPts): booleano
inicio
    //devolve os pontos contidos na Eps vizinhança de Ponto
    sementes := ConjuntoDePontos.BuscaAbrangencia(Ponto, Eps);
    se sementes.tamanho<MinPts então //não existem pontos centrais
    inicio
        //muda o ID do grupo ao qual Ponto pertence, para RUIDO
        ConjuntoDePontos.MudarIdGrupo(Ponto, RUIDO);
        retorna falso;
    fim
    else    //todos os pontos em "sementes" são alcançaveis
            //por densidade a partir de Ponto
    inicio
        //muda o ID do grupo que contem o Ponto, para o ID atual ClId
        ConjuntoDePontos.MudarIdGrupo(Ponto, ClId);
        sementes.eliminar(Ponto);
        enquanto sementes <> Vazio fazer
        inicio
            //devolve o primeriro elemento de sementes
            Patual := sementes.primeiro();
            //devolve os pontos na Eps vizinhança de Patual
            resultado := ConjuntoDePontos.BuscaAbrangencia(Patual, Eps);
                se resultado.tamanho >= MinPts então
                inicio
                para i de 1 a resultado.tamanho fazer
                inicio
                    Presultado := resultado.devolver(i);
                    se Presultado.IdGrupo em {NÃOCLASSIFICADO, RUIDO} então
                    inicio
                        se Presultado.IdGrupo = NÃOCLASSIFICADO então
                        inicio
                            sementes.adicionar(Presultado);
                        fim
                        ConjuntoDePontos.mudarIdGrupo(Presultado, IdGrupo);
                    fim //NÃOCLASSIFICADO OU RUIDO
                fim
```

```
            fim //resultado.tamanho >= MinPts
            sementes.eliminar(Patual);
        fim //sementes <> Vazio
        retorna verdadeiro;
    fim
fim;
```

O algoritmo GDBSCAN (Sander et al., 1998) é uma generalização do DBSCAN (Ester et al., 1996) em dois aspectos: (1) na definição de vizinhança; e (2) na medida da cardinalidade de uma vizinhança. Essas generalizações permitem agrupar dados que possuem atributos não espaciais.

DEFINIÇÃO 7 (Vizinhança de um ponto) Seja $NPred$ uma relação binária sobre o conjunto de pontos D que é reflexiva e simétrica, i.e., $\forall p, q \in D : NPred(p,p)\ e\ NPred(p,q) \Rightarrow NPred(q,p)$. Então a $NPred$ vizinhança V_{NPred} de um objeto $o \in D$ é definida como $V_{NPred}(o) = \{o' \in D / NPred(o,o')\}$.

DEFINIÇÃO 8 ($MinPeso$ de um conjunto de pontos) Seja $wCard$ uma função do conjunto potência de um conjunto de dados D nos números reais não negativos, $wCard : 2^D \rightarrow \mathbb{R}^+$ e $MinCard$ um número real positivo. Então o predicado $MinPeso$ para um conjunto de pontos S é definido verdadeiro se e somente se $wCard(S) \geq MinCard$.

Todas as demais definições são iguais às do DBSCAN, substituindo as medidas de distância e de cardinalidade pelas definições acima apresentadas e moldando-as à aplicação específica. O algoritmo do GDBSCAN é idêntico ao apresentado anteriormente para o DBSCAN. Esses algoritmos não podem ser aplicados a dados de alta dimensionalidade devido à sua dependência de conceitos de proximidade definidos no espaço original.

Vários dos algoritmos apresentados neste capítulo buscam tratar dois dos problemas mais importantes do agrupamento de dados: o tamanho e a dimensionalidade dos bancos de dados. Porém, nenhum deles representa uma solução para ambos os problemas ao mesmo tempo. Além disso, a maioria deles não gera grupos de formatos arbitrários, nem tratam os *outliers*. Existe a necessidade de um algoritmo que inclua todas essas características de forma que possa ser aplicado a conjuntos de dados reais. Esses tópicos foram o foco do trabalho de pesquisa realizado, e serão discutidos na próxima seção.

2.3 Agrupamento em Grandes Bancos de Dados Multidimensionais

Ultimamente, considerável atenção tem sido dada ao problema de realizar agrupamento em conjuntos de dados arbitrariamente grandes (Aggarwal e Yu, 2000; Ganti et al., 1998, 1999; Xu et al., 1998; Zhang et al., 1996) e diferentes abordagens foram propostas, baseadas em mineração incremental, compressão de dados e amostragem confiável. Tais abordagens foram apresentadas neste capítulo na discussão das várias técnicas de agrupamento.

O DIGNET (Thomopoulos et al., 1995; C.-D.Wann e Thomopoulos, 1997) é um exemplo de algoritmos de aprendizado incremental não supervisionado. O algoritmo é incremental porque trabalha com um único ponto de dado por vez. O DIGNET utiliza a representação de grupos k-médias (por centróides) sem otimização iterativa. O centróide mais próximo ao ponto a ser inserido na iteração atual é deslocado em direção a tal ponto. Esse agrupamento *on-line* executa uma única passagem pelo banco de dados, mas depende fortemente da ordem dos dados, e pode resultar em grupos de baixa qualidade. Porém, o algoritmo trata os *outliers*, os grupos podem ser criados ou eliminados dinamicamente e novas instâncias podem ser adicionadas sem necessidade de treinar novamente o algoritmo com todo o conjunto de treinamento. Isso o torna uma opção muito boa para bancos de dados dinâmicos.

As técnicas de compressão de dados calculam certos resumos dos dados (estatisticamente suficientes) (DuMouchel et al., 1999). Os resumos obtidos são usados ao invés dos dados originais, para posterior classificação. Um exemplo de algoritmo que utiliza essa técnica é o BIRCH (Zhang et al., 1996, 1997), descrito na Seção 2.2. Porém, compressão de dados acarreta perda de informação.

Ganti et al. (Ganti et al., 1999) propuseram um algoritmo, também mencionado na Seção 2.2, que utiliza uma compressão de dados semelhante à utilizada pelo BIRCH para agrupamento em grandes bancos de dados espaciais. Cada nó folha da árvore que descreve um nível de agrupamento, é caracterizado por: (1) o número de pontos; (2) o medóide (chamado clustróide) que assegure um erro[4] mínimo; e (3) o raio do grupo, o qual é igual à raiz quadrada da média do erro por ponto. O algoritmo utiliza um agrupamento incremental que permite realizar uma única passagem pelo banco de dados. A utilização do algoritmo *Fastmap* (Faloutsos e Lin, 1995) permite a redução da dimensionalidade de forma que o cálculo das distâncias seja mais leve computacionalmente. Porém, projetar o conjunto de dados em um único subespaço acarreta perda de informação e é muito provável que não sejam encontrados todos os grupos.

[4] A distância quadrática entre o medóide e todos os outros pontos que pertencem ao nó folha.

Nos bancos de dados atuais, as instâncias possuem centenas, ou inclusive milhares, de atributos. A descoberta de conhecimento em tais espaços multidimensionais apresenta uma enorme dificuldade. Nas árvores de decisão, por exemplo, os atributos irrelevantes poderiam ser escolhidos para a divisão dos nós. No agrupamento, mais de 31 dimensões apresentam um problema duplo (Berkhin, 2002; Beyer et al., 1999). Primeiro, sob qualquer definição de similaridade, a presença de atributos irrelevantes elimina qualquer possibilidade de uma tendência de agrupamento. A quantidade de atributos irrelevantes cresce na medida que cresce a dimensionalidade do conjunto de dados. O segundo problema é a "maldição da dimensionalidade", denominação para a falta de separação dos dados em espaços multidimensionais.

Uma análise exploratória básica dos dados (por exemplo, seleção de atributos) que preceda ao agrupamento é a melhor forma de abordar o primeiro problema (Berkhin, 2002). Uma das técnicas para lidar com situações em que o número de atributos selecionados ainda é muito grande, é o agrupamento de subespaços: construir grupos em subespaços apropriados do espaço original. Esta técnica é utilizada por Aggarwal (Aggarwal, 2001) no trabalho que foi tomado como base para a pesquisa realizada, e que é apresentado no Capítulo 3.

Aggarwal e Yu (Aggarwal e Yu, 2000) apresentaram uma abordagem que permite agrupar de forma eficaz conjuntos de dados multidimensionais ao projetar os dados em subespaços bidimensionais, acrescentando a probabilidade de encontrar todos os grupos utilizando múltiplas projeções. Porém, a complexidade dos cálculos envolvidos não permite seu uso em grandes conjuntos de dados.

O algoritmo de Aggarwal (Aggarwal, 2001) permite agrupar conjuntos de dados multidimensionais, gerar grupos de formatos arbitrários e lidar com o efeito dos *outliers*. Porém, ele não é escalável a grandes conjuntos de dados devido à complexidade dos cálculos envolvidos no processo. Torná-lo aplicável a conjuntos de dados maiores foi o foco da pesquisa desenvolvida, assim como criar mecanismos para melhorar o agrupamento resultante quando os grupos se encontram muito próximos entre si.

2.4 Considerações Finais

Este capítulo descreveu estratégias de agrupamento de dados, apresentando uma revisão geral das técnicas de agrupamento, bem como uma discussão dos problemas apresentados pelos conjuntos de dados atuais devido a sua multidimensionalidade e grande tamanho. O problema de fazer um algoritmo aplicável a bancos de dados arbitrariamente grandes é chamado de problema da escalabilidade.

Além da escalabilidade, uma característica que a maioria das abordagens de agrupamento tenta incluir nos algoritmos é um requerimento mínimo do conhecimento de domínio para encontrar os grupos. Uma vez que não é possível definir o que é um bom agrupamento para uma aplicação específica sem a intervenção direta do usuário, ou sem o conhecimento de domínio ganho da experiência do usuário, não é possível conceber um algoritmo geral que gere bons agrupamentos para todas as aplicações. Portanto, acreditamos que necessariamente deve-se admitir conhecimento de domínio nos algoritmos de agrupamento introduzido através de técnicas de visualização e de interação.

O próximo capítulo é dedicado aos trabalhos prévios relatados na literatura relacionados a essa idéia. Tais abordagens, coletivamente denominadas *agrupamento visual*, permitem uma interação direta durante o agrupamento para otimizar o processo.

CAPÍTULO

3

Agrupamento Visual de Dados

3.1 Considerações Iniciais

A *mineração visual de dados* consiste no processo de encontrar estruturas (padrões ou modelos) ocultas dentro dos dados originais, apoiando os algoritmos de mineração com técnicas de visualização de informação para permitir a interação do usuário, tanto nas etapas de pré-processamento e de apresentação dos resultados, quanto no processamento propriamente dito. Dentre essas técnicas se encontra o *agrupamento visual de dados* que, pelas razões mencionadas no capítulo anterior, pode se beneficiar da interferência do usuário de várias formas.

Este capítulo é dedicado à apresentação de conceitos relacionados a agrupamento visual. Técnicas de *visualização de informação*, assim como sua aplicação na mineração de dados, são abordadas na Seção 3.2. Na Seção 3.3 são apresentadas técnicas de agrupamento visual encontradas na literatura, incluindo as duas baseadas em densidade que fundamentam o presente trabalho. Finalmente, na Seção 3.4 são apresentadas as considerações finais do capítulo.

3.2 Visualização de Informação e Mineração Visual de Dados

Tradicionalmente, a visualização tem assistido os processos de mineração de dados em etapas de pré-processamento e de apresentação de resultados (Kreuseler e Schumann, 2002). Uma

estratégia de mineração visual de dados mais interessante baseia-se em juntar os processos analíticos e de visualização em uma única ferramenta de mineração de dados (Wong, 1999), permitindo ao usuário tomar decisões e controlar aspectos nos quais, tradicionalmente, heurísticas são empregadas nos algoritmos de mineração. Integrar métodos visuais e não visuais para apoiar as tarefas envolvidas na mineração visual de dados requer técnicas sofisticadas de mineração, visualização e interação (Kreuseler e Schumann, 2002).

Nesta seção apresentaremos conceitos de visualização de informação, assim como técnicas de visualização de dados multidimensionais e o papel da visualização na mineração de dados. Uma coletânea sobre técnicas de visualização de informação e sua utilidade foi publicada por Oliveira e Levkowitz (Oliveira e Levkowitz, 2003).

3.2.1 Visualização de Informação

Segundo Card *et al.* (Card et al., 1999) a *visualização de informação* é o uso de computadores para possibilitar a interação de uma pessoa com representações visuais de dados abstratos, com o objetivo de ampliar a seu conhecimento. A interação possui um papel importante nas técnicas de visualização de informação.

A visualização é usada no processo de exploração de dados, mas ainda não em toda a extensão possível. Nos últimos anos tem sido usada para mostrar os resultados de cálculos estatísticos ou de algoritmos de mineração de dados. O termo "visualização de dados multidimensionais" é usualmente, confundido com o termo "visualização multidimensional". Porém, no primeiro o termo "multidimensional" refere-se aos dados enquanto no segundo refere-se à visualização (Grinstein et al., 2001). Na seguinte subseção são apresentadas técnicas de visualização de informação para dados multidimensionais.

3.2.2 Técnicas de Visualização de Dados Multidimensionais

Existem diferentes classificações para as técnicas de visualização de dados (Keim, 2002; Grinstein et al., 2001), mas, segundo Grinstein *et al.* (Grinstein et al., 2001), entre as técnicas que podem ser classificadas como visualizações de dados multidimensionais encontramos: *scatterplots* 2D e 3D, Matrizes de *scatterplots*, *Heatmaps*, *Heightmaps*, *Table lens*, *Survey plots*, Iconográficas, Empilhamento Dimensional, Coordenadas Paralelas, Grafos Lineares, Grafos Multi-lineares, Técnicas Orientadas a Pixel, Segmentos de Círculo, Cartas Polares, RadViz, PolyVis, Análise de Componentes e Curvas Principais, *Grand Tour*, *Projection Pursuit* e Mapas Auto-organizáveis de Kohonen. Muitas delas são similares ou estão relacionadas (Grinstein

et al., 2001). Nesta subseção são apresentadas técnicas que podem vir a ser consideradas para apoio visual no desenvolvimento deste trabalho.

***Scatterplots* 2D e 3D**

Um *scatterplot* é uma projeção 2D ou 3D na tela do computador de marcadores gráficos (por exemplo, pontos) representando os dados, sendo um dos métodos mais utilizados atualmente. Várias transformações e operações podem ser aplicadas a esta visualização (por exemplo, seleção de intervalos) e diferentes atributos podem ser exibidos, além dos dois ou três mapeados nos eixos existentes, através de mapeamento para cor, tamanho, orientação, etc., dos marcadores gráficos. Um exemplo dessa técnica é mostrada na Figura 3.1 para o conjunto de dados Iris.

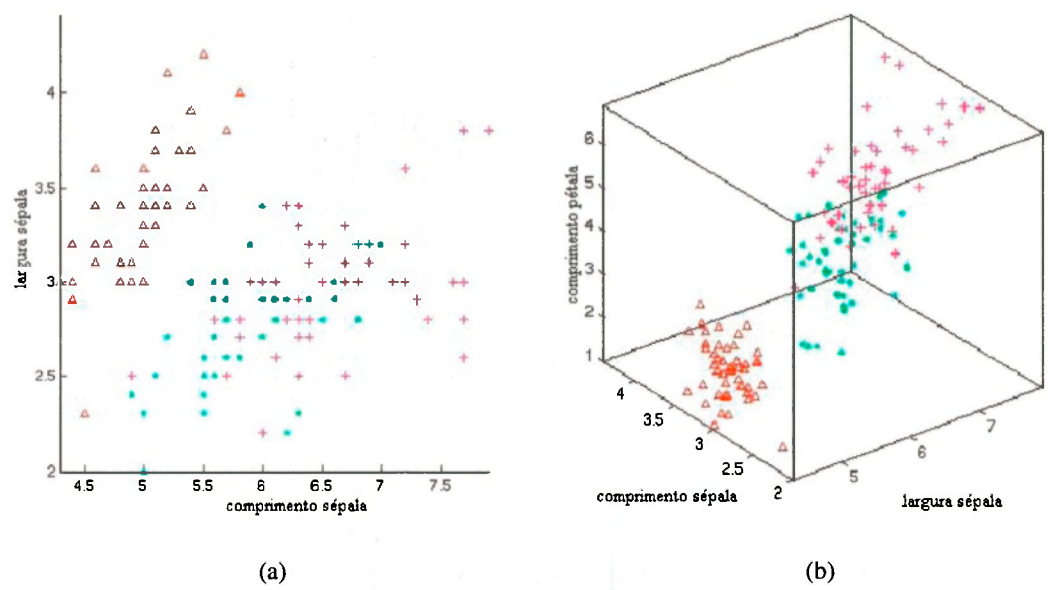


Figura 3.1: Exemplos de *scatterplot* (a) 2D e (b) 3D (Grinstein et al., 2001).

HeatMaps

Um *Heatmap* é uma matriz de células coloridas com base nos valores de algum atributo dos dados. O método é uma generalização do *scatterplot* 2D onde os pontos são células da malha e cada célula é colorida (Grinstein et al., 2001). A Figura 3.2 mostra um exemplo dessa técnica aplicada ao bem conhecido conjunto de dados Iris (Witten e Frank, 1999).

HeightMaps

Os *heightmaps* mapeiam um dos atributos do conjunto de dados na altura de uma malha que representa dois outros atributos no plano bidimensional. O deslocamento no terceiro eixo é proporcional ao valor do terceiro atributo. Outro atributo pode ser mapeada na cor, como apresentado na Figura 3.3. Nesse caso, quando a altura dos vértices da malha é constante, obtemos um *colormap*.

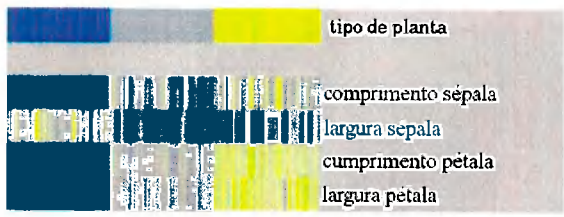


Figura 3.2: *Heatmap* para o conjunto de dados Iris (Grinstein et al., 2001).

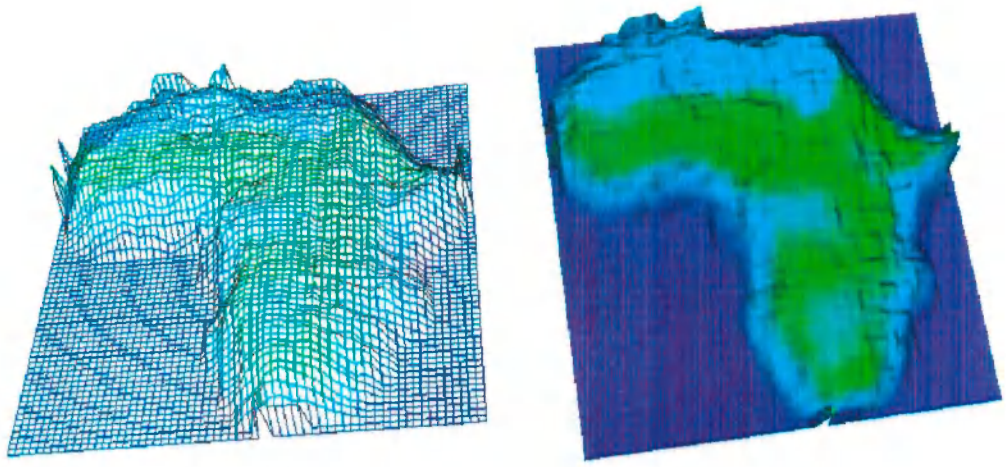


Figura 3.3: Exemplo de um *Heightmap* (Minghim e de Oliveira, 1997).

Table lens

Table lens é uma técnica de visualização que associa os valores das instâncias de uma tabela de dados ao comprimento de uma linha horizontal associada a cada entrada na tabela. Também pode-se mostrar um outro atributo utilizando a cor da linha, como apresentado na Figura 3.4. O usuário pode interagir com a representação para realizar operações de ordenação, seleção, etc., sobre os atributos e as instâncias.

Iconográficas

As técnicas iconográficas utilizam *glyphs*, ou ícones, para representar as instâncias e os seus valores. Essas técnicas mapeiam um valor a cada característica do ícone. Alguns ícones podem ser apresentados de forma separada para cada instância, ou juntos, formando uma textura sobre a tela (Keim e Ankerst, 2001). Um exemplo dessa abordagem é mostrado na Figura 3.5.

Empilhamento dimensional

O empilhamento dimensional é uma projeção pontual 2D ou 3D dos dados em que as dimensões são embutidas em outras dimensões (Grinstein et al., 2001). O empilhamento divide o plano 2D em um conjunto de retângulos. Dois dos atributos (os mais relevantes, em geral) são representados pelos eixos X e Y mais externos. Para adicionar dois atributos à representação,

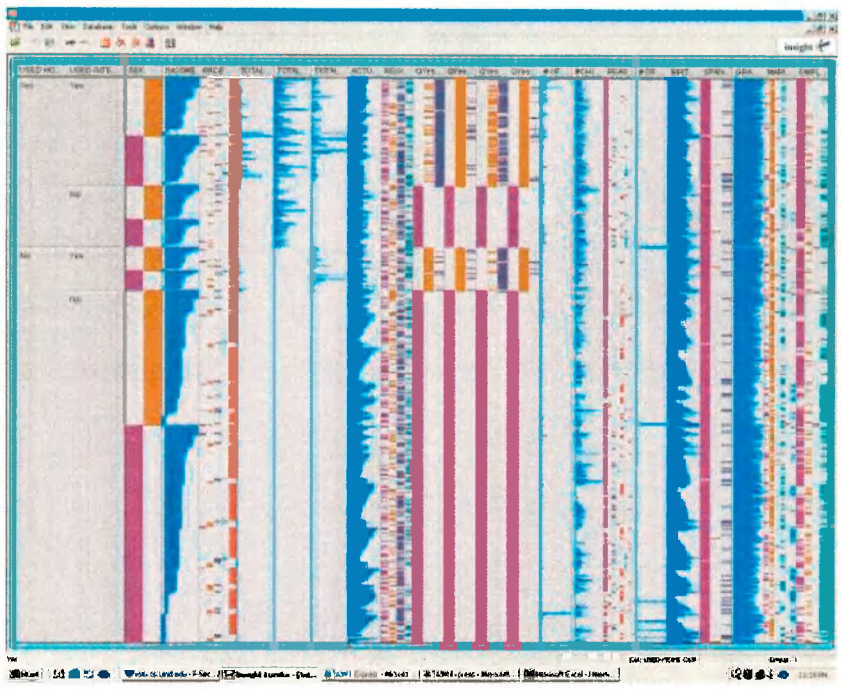


Figura 3.4: Tabela de dados explorada com *Table Lens* (Fonte: *http : //www.cs.umd.edu/-class/spring2001/cmsc838b/Apps/presentations/Erica_Kolatch/*).

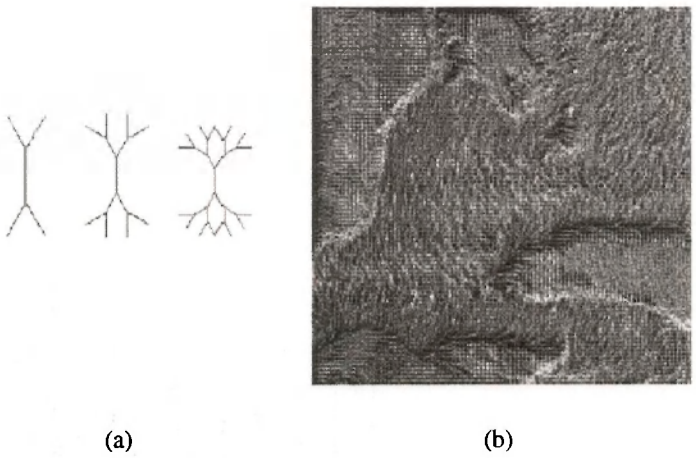


Figura 3.5: (a) Ícone utilizado para mostrar uma instância (Grinstein et al., 2001). O conjunto de ícones pode definir uma textura (Keim e Ankerst, 2001).

divide-se o plano em retângulos cujos eixos X e Y internos representam as novas dimensões. Dessa forma pode-se criar uma hierarquia, ou árvore de dimensões na representação, como mostrado na Figura 3.6.

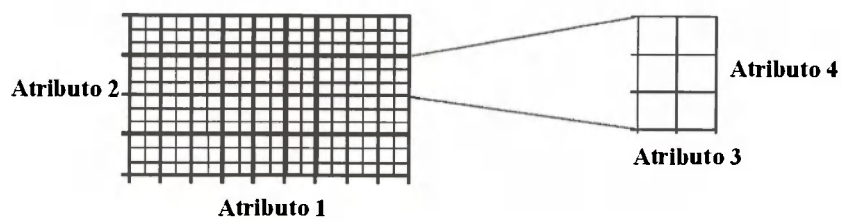


Figura 3.6: Ilustração do empilhamento dimensional (Keim e Ankerst, 2001).

Coordenadas paralelas

A técnica de coordenadas paralelas, desenvolvida por Inselberg (Inselberg, 1985; Inselberg e Dimsdale, 1987) muda o conceito da representação dos eixos de coordenadas perpendiculares, alinhando-os paralelos entre si sobre um plano, como apresentado na Figura 3.7.

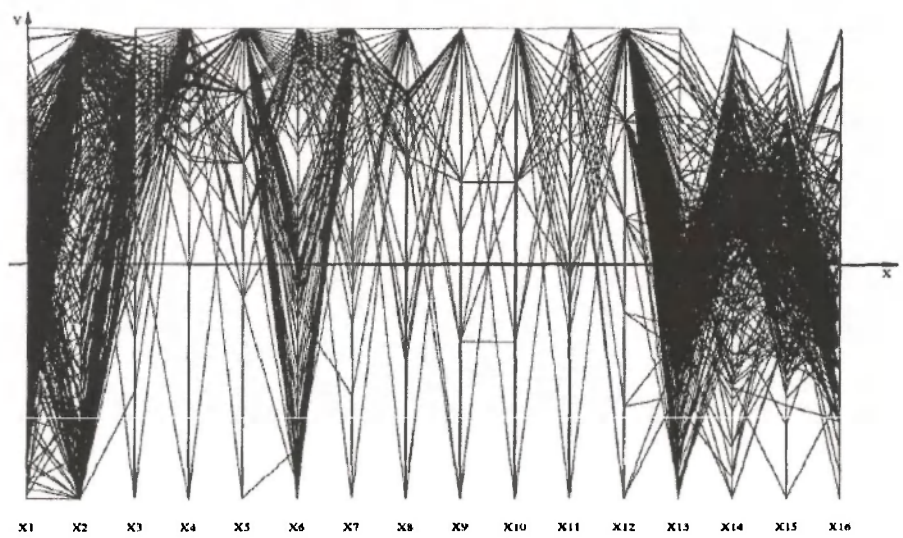


Figura 3.7: Exemplo de coordenadas paralelas (Inselberg e Dimsdale, 1987).

Esta representação permite a visualização de todos os atributos em um único plano, permitindo observar tendências nos valores dos atributos das instâncias localmente. A capacidade dessa técnica para mostrar as relações entre os atributos é afetada pela ordem em que eles são atribuídos aos eixos paralelos. Diferentes extensões para permitir uma melhor interpretação, como interação e seleção de intervalos, foram propostas na literatura (Keim, 2002).

Técnicas orientadas a pixel

As técnicas orientadas ao pixel são uma generalização dos *heatmaps* no sentido que os pixels são coloridos seguindo um padrão espiral, circular, linear, etc., sendo que a interpretação da cor do pixel em determinada posição da janela depende do padrão utilizado. A Figura 3.8 mostra um exemplo de uma abordagem utilizando um padrão circular, denominada Técnica de Seg-

mentos de Círculo (Ankerst, 2001). Essa técnica mapeia cada atributo a uma seção do círculo, apresentando os valores dos atributos de cada instância na mesma distância radial relativa nos diferentes segmentos. Os valores são mapeados na cor do segmento e seção correspondentes.

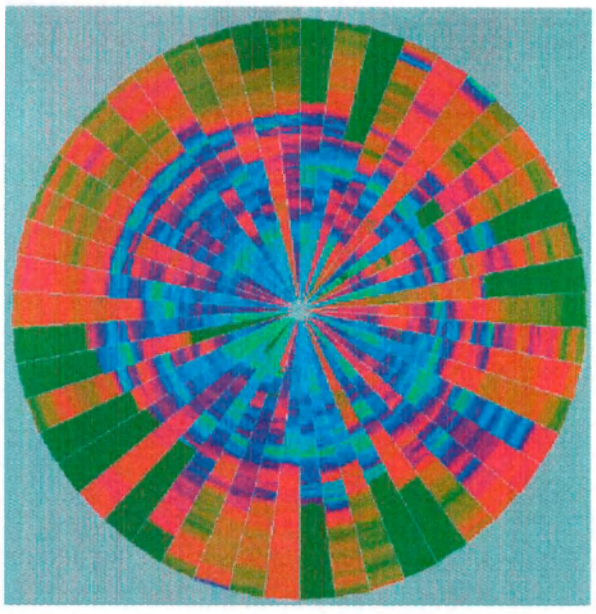


Figura 3.8: Exemplo da técnica de segmento de círculo (Ankerst, 2001).

RadViz

RadViz é uma técnica de visualização de informação que associa cada atributo a um eixo radial em uma circunferência como apresentado na Figura 3.9, onde se ilustra um exemplo dessa técnica aplicada ao conjunto de dados Iris. Cada instância é representada por um ícone. As relações entre os atributos das instâncias do banco de dados são representadas pela posição do ícone, a qual é o ponto no plano onde a soma das "forças" (valores) dos atributos da instância é zero. A força de um atributo é um vetor na direção do eixo radial associado ao atributo, onde a magnitude do vetor é dada pelo valor do atributo.

PolyViz

A técnica PolyViz estende o método RadViz ao substituir os ícones por linhas traçadas a partir da posição da instância até os pontos em que a circunferência se intersecta com os eixos radiais associado a cada atributo. Dessa forma, é possível observar a distribuição das instâncias com respeito a cada atributo através das posições das linhas associadas a ele, como pode ser visto na Figura 3.10.

As técnicas apresentadas buscam tratar dados multidimensionais. Elas são geralmente utilizadas na etapa de pré-processamento, de forma que apoiem a análise dos dados dos quais se

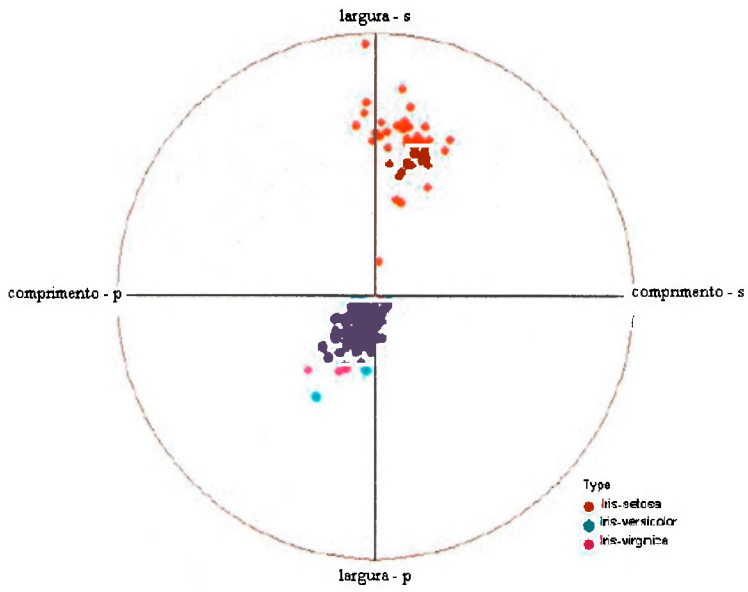


Figura 3.9: Visualização utilizando RadViz para os dados Iris (Grinstein et al., 2001).

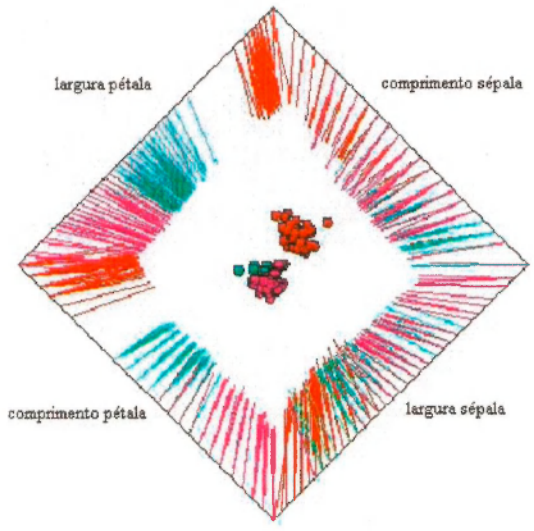


Figura 3.10: Visualização utilizando PolyViz para os dados Iris (Grinstein et al., 2001).

extrairá o conhecimento. Elas podem ser usadas em um primeiro estágio dos algoritmos de agrupamento visual para determinar os parâmetros de entrada para o processo de agrupamento.

3.2.3 Visualização na Mineração de Dados

Embora as técnicas de visualização tenham melhorado significativamente na última década, essa tecnologia ainda é utilizada unicamente no final do processo de descoberta de conheci-

mento (Collier et al., 2002). A maioria das ferramentas de mineração de dados não aproveita os benefícios das técnicas de visualização. Segundo Collier *et al.* (Collier et al., 2002), duas coisas devem acontecer para uma melhor utilização da visualização de informação:

- Uma métrica deve ser desenvolvida para avaliar de forma quantitativa e comparativa a eficácia de técnicas de visualização.
- A visualização de informação deve ser incluída desde a fase de pré-processamento até a fase de apresentação dos resultados, passando pelo processo de mineração propriamente dito.

O primeiro requisito está sendo abordado por pesquisadores que tentam desenvolver uma métrica que permita comparar as diversas técnicas de visualização de informação (ver (Grinstein et al., 2001)) e avaliar sua usabilidade (ver (Freitas et al., 2002)). O segundo será um processo longo que dependerá muito da confiança que as pessoas venham a ganhar em relação à visualização como ferramenta para a interpretação de dados.

Os trabalhos em visualização de informação na mineração de dados vêm buscando a formalização de uma área de pesquisa própria concentrada nesse tema. Kreuseler e Schumann (Kreuseler e Schumann, 2002) apresentam várias referências a trabalhos sobre mineração visual de dados. Essas abordagens são classificadas em três categorias (Ankerst, 2001). A primeira inclui aquelas abordagens que usam visualização em tarefas de pré-processamento. A segunda categoria é formada por aquelas abordagens que utilizam técnicas de visualização para apresentar os resultados dos processos de mineração de dados. Finalmente, o terceiro grupo inclui abordagens que integram tanto as técnicas de aprendizado de máquina quanto as técnicas de visualização em um único processo de mineração, explorando a sinergia entre a capacidade humana e a do computador. Esse grupo inclui as abordagens de agrupamento visual baseado em densidade que serão apresentadas na Seção 3.3.

Existe uma considerável quantidade de pesquisa em visualização de resultados de algoritmos de aprendizado de máquina (segunda abordagem) (Becker, 1997). MineSet (Becker, 1997) é um exemplo claro de uma ferramenta comercial que permite ao usuário visualizar os resultados de algoritmos de construção de árvores de decisão e de extração de regras. Essa ferramenta foi projetada para permitir uma fácil interação com o usuário, o que requer um processamento rápido para possibilitar uma retroalimentação contínua entre o usuário e o computador.

O agrupamento, sendo uma das técnicas de aprendizado de máquina utilizada na mineração de dados, não foi uma exceção a esse movimento de inclusão de técnicas de visualização no processo de mineração. Na Seção 3.3 serão apresentados os trabalhos em agrupamento visual de dados baseados em densidade que constituem a base da pesquisa apresentada neste documento.

3.3 Técnicas de Agrupamento Visual de Dados

Trabalhos diretamente relacionados com agrupamento visual de dados foram densenvolvidos por Ribarsky *et al.* (Ribarsky et al., 1999; Ribarsky, 1999), Hinneburg *et al.* (Hinneburg et al., 1999) e Aggarwal (Aggarwal, 2001, 2002).

Ribarsky *et al.* (Ribarsky et al., 1999; Ribarsky, 1999) introduziram o termo *Discovery Visualization* para descrever um processo de agrupamento de dados centrado no usuário, com capacidade de resposta adaptada para maximizar suas capacidades. Eles desenvolveram um algoritmo de agrupamento visual com as restrições de espaço e tempo em mente desde a etapa de projeto, para permitir uma animação da variação da estrutura do agrupamento ao longo do tempo. Eles usam amostragem, *binsort* e agrupamento rápido para assegurar um tempo de resposta de 0.1 segundo ou menos. Além disso, provêm um meio para exibir a hierarquia do agrupamento.

Aggarwal (Aggarwal, 2002) propôs uma abordagem para buscas de vizinho mais próximo em bancos de dados multidimensionais, a partir da perspectiva do usuário, projetando um sistema que encoraja a cooperação efetiva entre o humano e o computador. Para isso, é necessário encontrar projeções centradas no ponto de busca (*query point*), denominadas *projeções discriminativas*, nas quais é possível distinguir os grupos que contêm o ponto de busca do resto dos dados. Aggarwal usou PCA (Jolliffe, 1986) para encontrar tais projeções. Dessa forma o usuário separa visualmente o grupo ao qual pertence a instância, do resto do espaço de dados, reduzindo a quantidade de operações necessárias para encontrar o vizinho mais próximo.

As principais estratégias para agrupamento visual de dados multidimensionais baseado em densidade foram desenvolvidas por Hinneburg *et al.* (Hinneburg et al., 1999) e Aggarwal (Aggarwal, 2001). Nos concentraremos nessas soluções particulares devido à sua capacidade de identificar grupos de formatos arbitrários em bancos de dados multidimensionais. Além disso, elas permitem um mapeamento visual direto da estimativa de densidade das projeções bidimensionais do conjunto de dados. Isso pode ser feito através de um *colormap* ou *heightmap*, de modo a apresentar visualmente a distribuição das instâncias e permitir uma especificação dos grupos dirigida pelo usuário.

3.3.1 HDEye

A abordagem proposta por Hinneburg *et al.*(Hinneburg et al., 1999), denominada HDEye, apresenta o *colormap* da estimativa de densidade de diferentes projeções contractivas do conjunto de dados, bidimensionais ou unidimensionais, para que o usuário escolha aquela na qual os dados formam grupos bem diferenciados. Uma vez escolhida a projeção, utiliza-se um *separador* (um

polígono fechado para o caso bidimensional, ou um *glyph* para o caso unidimensional) para definir os grupos sobre o *colormap* (construído sobre uma grade bidimensional) que mapeia a distribuição de densidade dos dados no espaço bidimensional projetado. Para isso, é necessário estimar a distribuição de densidade. Isso é feito utilizando um *núcleo estimador* (Silverman, 1986). A estimativa de densidade $\hat{f}_D$ é definida como segue.

DEFINIÇÃO 9 (Função núcleo de estimativa de densidade) Seja D um conjunto de n pontos d-dimensionais e h um fator de suavização. Podemos definir uma *função de estimativa de densidade* baseada no núcleo K como

$$\hat{f}_D = \frac{1}{nh^d} \sum_{i=1}^{n} K_h(x - x_i) \tag{3.1}$$

para qualquer $x \in D$.

Portanto, cada ponto discreto x_i do conjunto de dados é trocado por uma função contínua $K_h(\cdot)$ com máximo em x_i e cuja variância é determinada pelo fator de suavização h. Um exemplo de tal distribuição seria o núcleo Gaussiano com fator de suavização (ou largura de banda) h.

$$K_h(x - x_i) = (1/\sqrt{2\pi \cdot h}) \cdot e^{-(x-x_i)^2/(2h^2)} \tag{3.2}$$

O erro na estimativa de densidade é determinado pela largura de banda h. Uma aproximação conhecida (Silverman, 1986) para determinar a largura de banda é dada pela fórmula $h = 1.06 \cdot \sigma \cdot n^{-1/5}$ para um conjunto de dados com n pontos e desvio padrão σ. Exemplos de estimativa de densidade para um conjunto de dados, e respectivas projeções sobre planos representando diferentes níveis de ruído são mostrados na Figura 3.11.

DEFINIÇÃO 10 (Grupo de Densidade com Centro Único) Um *grupo de densidade com centro único* para um máximo $x*$ da função de densidade $\overline{f}(x)$ é o subconjunto $C \subseteq D$, com $x \in C$ sendo atraído por densidade por $x*$ e $\overline{f}(x*) \geq \xi$, onde um ponto x é atraído por densidade por $x*$ se existe um caminho $P \subset F^d$ a partir de x até $x*$ acima do nível de ruído ξ. F^d é o conjunto de células da grade d dimensional sobre a qual foi estimada a densidade.

DEFINIÇÃO 11 (Grupo de Densidade com Múltiplos Centros) Um *grupo de densidade com múltiplos centros* para um conjunto de máximos X é o subconjunto $C \subseteq D$, onde $\forall x \in C$ $\exists x* \in X$ com $\overline{f}(x*) \geq \xi$, x é atraído por densidade por $x*$ e $\forall x_1*, x_2* \in X$ existe um caminho $P \subset F^d$ a partir de x_1* até x_2* acima do nível de ruído ξ.

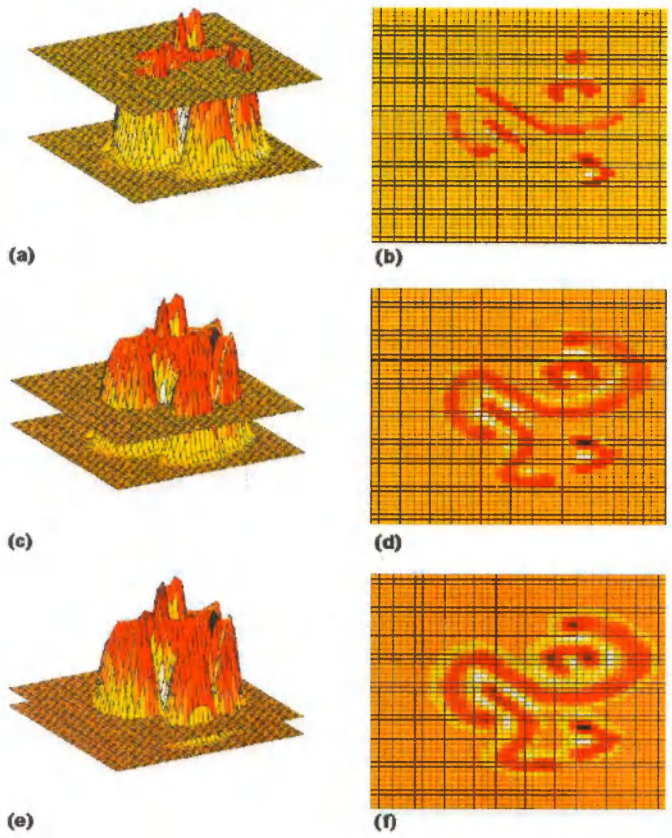


Figura 3.11: Exemplo de grupos com múltiplos centros para diferentes valores de ruído ξ. (a),(b) $\xi = 4$; (c),(d) $\xi = 2$; and (e),(f) $\xi = 1$; (Hinneburg et al., 1999)

Para assegurar que a densidade no subespaço bidimensional projetado seja maior ou igual à densidade no espaço multidimensional original, é necessário trabalhar com projeções contrativas.

DEFINIÇÃO 12 (Projeção Contrativa) Uma *projeção contrativa* para um espaço de dados d-dimensional S e uma métrica apropriada $\|\cdot\|$ é uma transformação linear P definida sobre todos os pontos $x \in S$

$$P(x) = Ax \text{ com } \|A\| = \max_{y \in S} \left(\frac{\|Ay\|}{\|y\|} \right) \leq 1 \tag{3.3}$$

onde A é uma matriz de projeção.

A Figura 3.12 ilustra a diferença entre projeções gerais e contrativas.

Como mencionado, o usuário especifica o grupo utilizando um separador sobre o *colormap* criado a partir da estimativa de densidade do subespaço projetado, onde um separador pode ser definido como segue.

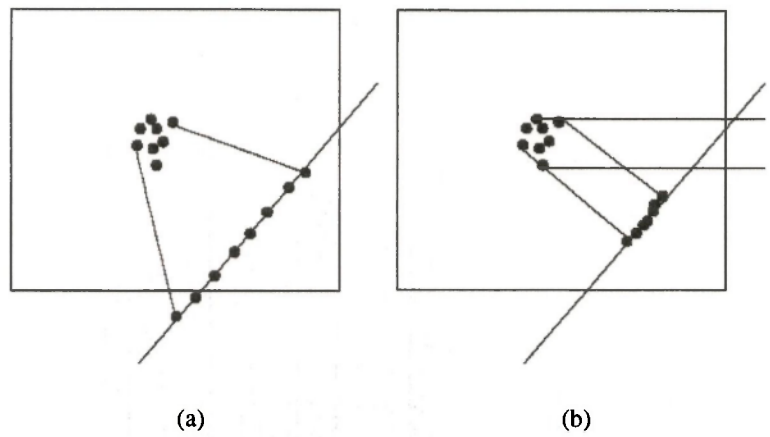


Figura 3.12: Exemplos de projeções (a) gerais e (b) contrativas (Hinneburg et al., 1999)

DEFINIÇÃO 13 (Separador) Um *separador* é um objeto geométrico que divide o espaço $\mathbb{R}^d$ em dois semi-espaços h_0, h_1. A função de decisão $H(x)$ determina os semi-espaços, onde um ponto $x \in \mathbb{R}^d$ pertence a

$$H(x) = \begin{cases} 1 & x \in h_1 \\ 0 & \text{c. c.} \end{cases} \tag{3.4}$$

Exemplos de separadores são hiperplanos P definidos por um valor de divisão x_s

$$\mathbb{R}^d \rightarrow \mathbb{R}^1, H(x) = \begin{cases} 1 & P(x) \geq x_s \\ 0 & \text{c. c.} \end{cases} \tag{3.5}$$

e cilindros multidimensionais P definidos por polígonos fechados (Figura 3.13).

$$\mathbb{R}^d \rightarrow \mathbb{R}^2, H(x) = \begin{cases} 1 & P(x) \text{ dentro do polígono } p \\ 0 & \text{c. c.} \end{cases} \tag{3.6}$$

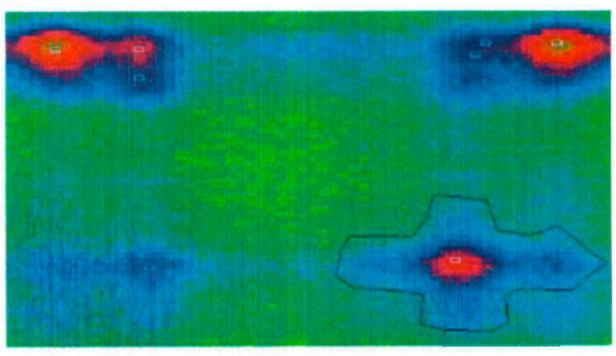


Figura 3.13: Exemplo de separador (Hinneburg et al., 1999)

Na medida que o usuário especifica os separadores, uma grade multidimensional é criada, a qual divide o espaço em várias regiões que separam os grupos do resto de espaço. Uma grade multidimensional é definida como segue.

DEFINIÇÃO 14 (Grade Multidimensional) Uma *grade multidimensional* G, no espaço de dados S, é definida como o conjunto de separadores $H = \{H_1, ...H_k\}$. Uma função de codificação $c^G : S \rightarrow \mathbb{N}$ sobre essa grade multidimensional pode ser expressa como

$$x \in S, c(x) = \sum_{i=1}^{k} 2^i \cdot H_i(x) \tag{3.7}$$

A partir dessas definições é possível apresentar o seguinte pseudo-código para o algoritmo HDEye.

Algoritmo 3
HDEye(ConjuntoDeDados D)
inicio
InicializarHierarquia(root ← D)
  enquanto *um no v (contendo o conjunto de dados D_v)*
  possa ser dividido ***fazer***
    $P = \{P_1, ..., P_k\}$=*EncontrarVisualmenteProjeçõesUteis(D)*
    $H = \{H_1, ...H_r\}$=*ConstruirSeparadores(P)*
    //D_v: o subconjunto de D que contem os pontos do no v.
    G=ConstruirGradeMultidimensional(H, D_v)
    //utiliza a função de codificação
    //para incluir cada instância no respectivo semiespaço.
    AdicionarInstanciasNaGrade(G,v)
    //adiciona os descendentes (novos grupos) do no atual v
    //na hierarquia de grupos.
    AdicionarDescendentesGrupoHierarquia(v).
  fim
fim;

HDEye foi o primeiro trabalho em agrupamento visual baseado em densidade. Uma abordagem mais poderosa foi desenvolvida por Aggarwal (Aggarwal, 2001), a qual permite uma busca dos grupos em múltiplas projeções e fornece uma métrica para medir a qualidade de um grupo. Esse trabalho é descrito na próxima seção.

3.3.2 Um sistema cooperativo humano-computador para agrupamento multidimensional efetivo

Um fato importante mostrado na literatura (Aggarwal e Yu, 2000; Aggarwal, 2001, 2002) é que diferentes grupos de instâncias agrupam-se bem em diferentes subespaços projetados. Aggarwal (Aggarwal, 2001) propôs uma abordagem para encontrar grupos em múltiplas projeções do conjunto de dados, usando *ID strings* para armazenar as respostas sucessivas dos usuários sobre os grupos e os espaços projetados nos quais eles parecem ser consistentes.

Por exemplo, na tabela 4.3, têm-se os grupos aos quais as instâncias 1 a 5 pertencem em cinco diferentes projeções. Pode-se observar que as instâncias 1, 4 e 5 pertencem ao mesmo grupo em quatro das cinco projeções. Portanto, é altamente provável que elas pertençam ao mesmo grupo no espaço multidimensional original.

	Projeção				
Instância	I	II	III	IV	V
1	1	3	2	1	2
2	2	1	1	3	3
3	3	1	3	2	1
4	1	3	2	1	2
5	1	3	2	2	2

Tabela 3.1: Exemplo de $IDstrings$. O elemento em cada célula da tabela identifica o grupo ao qual pertence a instância em tal projeção. É altamente provável que as instâncias 1, 4, e 5 pertençam ao mesmo grupo no espaço multidimensional original.

As *ID strings* permitem calcular a *cobertura*[1] de cada grupo e fornecem a métrica da "qualidade" de um grupo. O algoritmo de agrupamento apresentado por Aggarwal (Aggarwal, 2001) consiste de quatro estágios: (1) determinação dos subespaços de polarização, (2) interação com o usuário para a criação dos grupos, (3) atualização das cadeias de identificação (*ID strings*) e (4) criação dos grupos finais.

A fase de *determinação de subespaços de polarização* é centralizada em encontrar conjuntos de projeções nas quais os dados agrupam-se bem e os grupos podem ser percebidos visualmente pelo usuário. Em cada iteração um pequeno número k de pontos do banco de dados é amostrado. Os pontos são denotados por $y_1 ... y_k$, e são os pontos de polarização ao redor dos quais os grupos são buscados. Devido ao fato de que $y_1 ... y_k$ são amostrados aleatoriamente, muitos deles podem pertencer a um grupo bem definido em uma projeção cuidadosamente escolhida denominada *subespaço de polarização*. O processo de encontrar os subespaços de polarização começa no espaço d-dimensional original $\mathcal{U}$. O subespaço atual $\mathcal{E}_c$ de dimensão l_c é reduzido em cada iteração até um espaço bidimensional. Em cada iteração, o algoritmo encontra o conjunto de

[1] A cobertura de um grupo é o número de pontos que pertencem a ele.

pontos de dados $\mathcal{M}_i$ mais próximos à âncora de polarização y_i, baseado na métrica de distância do subespaço $\mathcal{E}_c$. O número de pontos de dados em cada conjunto $\mathcal{M}_i$ é determinado por um valor s definido pelo usuário. O subespaço no qual as instâncias agrupam-se bem ao redor do centróide z_i do conjunto $\mathcal{M}_i$ é aquele no qual o momento de todos os grupos se minimiza. Isso é equivalente a minimizar $\sum_{i=1}^{k} \mathcal{R}(\mathcal{M}_i, \mathcal{E}, z_i)$ sobre todos os subespaços $\mathcal{E}$, onde $\mathcal{R}$ é o momento do grupo $\mathcal{M}_i$ com centróide z_i no espaço $\mathcal{E}$, definido como segue.

DEFINIÇÃO 15 (Momento de um grupo) O *momento* de um grupo $\mathcal{C} = \{\overline{x}_1, \overline{x}_2, ..., \overline{x}_t\}$ no subespaço $\mathcal{E}$ com respeito ao ponto $\overline{y}$ é denotado por $R(\mathcal{C}, \mathcal{E}, \overline{y})$ e é dado por:

$$R(\mathcal{C}, \mathcal{E}, \overline{y}) = \sum_{i=1}^{t} \{Pdist(\overline{x}_i, \overline{y}, \mathcal{E})\}^2/t \tag{3.8}$$

onde $Pdist$ é a distância projetada entre dois pontos definida a seguir.

DEFINIÇÃO 16 (Distância projetada) A *distância projetada* entre dois pontos y_1 e y_2 no subespaço $\mathcal{E}$ é denotada por $Pdist(\overline{y}_1, \overline{y}_2, \mathcal{E})$ e é igual à distância Euclideana entre os pontos $\overline{y}_1$ e $\overline{y}_2$ no espaço l-dimensional representado por $\mathcal{E}$.

Para reduzir o momento global $\mathcal{M}_i$ de todos os grupos é necessário transformá-los de forma que seus centróides z_i virem $\overline{0}$. Isso é feito subtraindo z_i de cada ponto em $\mathcal{M}_i$. Sendo $\mathcal{O}_i$ o conjunto de pontos resultantes desta transformação, a tarefa é reduzida a encontrar o espaço onde $\sum_{i=1}^{k} \mathcal{R}(\mathcal{O}_i(z_i), \mathcal{E}, \overline{0})$ é minimizado. Sabendo que $\sum_{i=1}^{k} \mathcal{R}(\mathcal{O}_i(z_i), \mathcal{E}, \overline{0}) = \mathcal{R}(\bigcup_{i=1}^{k} \mathcal{O}_i(z_i), \mathcal{E}, \overline{0})$, tudo o que precisa ser feito é encontrar os l_c autovetores associados aos l_c menores autovalores da matriz de covariância do conjunto $\mathcal{N}_p = \bigcup_{i=1}^{k} \mathcal{O}_i(z_i)$. Em outras palavras, deve-se encontrar as direções das menores componentes, usando a Análise de Componentes Principais (PCA - *Principal Components Analysis*) (Jolliffe, 1986), do conjunto de pontos em $\mathcal{N}_p$.

A matriz de covariância Δ de um conjunto de dados fornece uma medida da distância que é invariante às transformações lineares dos dados (Knapp, 1998). A covariância de duas características mede sua tendência de variarem juntas, i.e., co-variar. Enquanto a variância é a média do quadrado do desvio padrão do atributo com respeito à sua média, a covariância é a média dos produtos dos desvios dos valores dos atributos em relação às suas médias.

Para ser mais preciso, considere os atributos C_i e C_j. Seja $\{x(1,i), x(2,i), ..., x(n,i)\}$ um conjunto de n amostras de C_i e $\{x(1,j), x(2,j), ..., x(n,j)\}$ o conjunto correspondente de n amostras de C_j. (Isto é, $x(k,i)$ e $x(k,j)$ são atributos do mesmo padrão, P_k). Similarmente,

seja $m(i)$ a média de C_i, e $m(j)$ a média de C_j. Então, a covariância de C_i e C_j é definida por

$$\begin{aligned} c(i,j) &= \{[x(1,i) - m(i)][x(1,j) - m(j)] + ... + \\ &\quad +[x(n,i) - m(i)][x(n,j) - m(j)]\}/n \end{aligned} \tag{3.9}$$

A covariância possui várias propriedades importantes

- Se C_i e C_j tendem a incrementar juntos, então $c(i,j) > 0$;
- Se C_i tende a decrementar quando C_j incrementa, então $c(i,j) < 0$;
- Se C_i e C_j são independentes, então $c(i,j) = 0$;
- $|c(i,j)| \leq s(i)s(j)$, onde $s(i)$ é o desvio padrão de C_i;
- $c(i,i) = s(i)^2 = v(i)$, onde $v(i)$ é a variância de C_i.

Portanto, a covariância $c(i,j)$ é um número entre $-s(i)s(j)$ e $+s(i)s(j)$ que mede a dependência entre C_i e C_j, com $c(i,j) = 0$ se não existe dependência. A correspondência entre a covariância e a forma do grupo é ilustrada na Figura 3.14

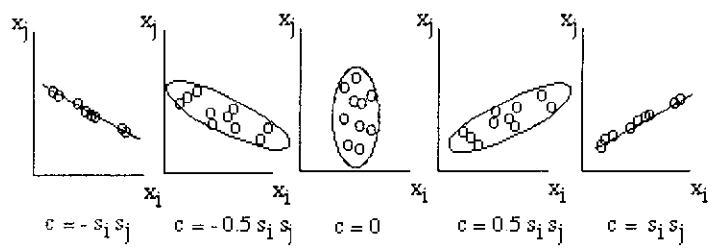


Figura 3.14: Correspondência entre a covariância e a forma do grupo (Knapp, 1998)

Os pontos em $\mathcal{N}_p$ possuem dimensão $|\mathcal{E}_c|$. Portanto, a matriz de covariância é uma matriz $\mathcal{E}_c * \mathcal{E}_c$ cujo elemento (i,j) denota a covariância entre as dimensões i e j. Esta matriz de covariância é semidefinida positiva e pode ser expressa como $\Delta = P \cdot D \cdot P^T$, onde D é uma matriz diagonal contendo os autovalores e as colunas de P contém os autovetores que formam um sistema axial ortonormal. Tais autovetores são as componentes principais e representam as direções nos dados ao longo das quais as covariâncias de segunda ordem dos pontos em $\mathcal{N}_p$ são zero. Foi mostrado por Jolliffe (Jolliffe, 1986) que os autovetores correspondentes aos menores autovalores definem o subespaço com o menor momento em relação ao centróide. Portanto, o subespaço $\mathcal{E}_c$ corresponde aos autovetores relacionados aos menores autovalores; i.e., espera-se que os pontos de dados em $\mathcal{M}_i$ formem grupos na projeção $\mathcal{E}_c$.

O processo de projeção é repetido, reduzindo a dimensão pela metade em cada iteração para refinar gradualmente o subespaço projetado, de forma que os espaços esparsos sejam eliminados.

Na fase de *interação do usuário para a criação dos grupos*, os grupos são especificados através da variação do nível de ruído ξ sobre o campo de densidade calculado usando o núcleo de estimativa de densidade previamente definido. Portanto, os grupos são regiões conexas que são atraídas pelo conjunto de máximos X definido por Hinneburg *et al.* (Hinneburg et al., 1999) e descritos na subseção anterior. Dessa forma, o usuário pode enxergar múltiplas visões separadas para diferentes valores de ruído ξ de forma a convergir interativamente para o valor intuitivo mais apropriado para o qual os pontos se agrupem bem e o ruído seja eliminado.

Uma vez que o usuário definiu o nível de ruído e, portanto os grupos foram identificados, as *ID strings* são atualizadas na terceira fase do algoritmo (*atualização das ID strings*). Nessa fase, o conjunto $\mathcal{K}$ de respostas do usuário em uma dada projeção é usado para atualizar o conjunto $\mathcal{I}$ de *ID strings*. O primeiro passo é determinar os contornos de um grupo ao encontrar todas as células elementares da malha que pertencem ao contorno. Assume-se que uma célula da malha (de quatro vértices) se encontra dentro do contorno de um grupo se pelo menos três dos seus vértices possuem um valor de densidade superior ao nível de ruído ξ especificado pelo usuário. Uma vez determinadas as células elementares, é necessário consolidá-las em componentes principais utilizando algum algoritmo de busca em grafos, assumindo que cada célula da malha é um nó e que os nós estão conectados por uma aresta em caso compartilharem um lado comum. Para cada um dos grupos, o passo seguinte é determinar os pontos de dados que se encontram nas células correspondentes. Eles constituem os grupos separados pelo usuário naquela projeção particular.

Depois que todos os grupos foram encontrados, um número de identificação é atribuído a cada um deles. Se o ponto de dados se encontra dentro do grupo, então sua identidade é igual à do grupo. De outra forma, ela é "*". Então o número de identificação é concatenado à *ID string* correspondente. Portanto, no final do processo, o tamanho da *ID string* é igual ao número total de separações aceitáveis do grupo especificadas pelo usuário.

Na fase de *criação dos grupos finais*, as *ID strings* são utilizadas para determinar os grupos que aparecem pelo menos uma quantidade de vezes igual à cobertura c especificada pelo usuário. Isto é feito buscando nas *ID strings* sub-padrões que ocorram pelo menos c vezes. Esses sub=padrões definem os grupos formados pelas instâncias associadas a tais *ID strings*. Isso leva à definição de suporte de um padrão e, portanto, do grupo que representa.

DEFINIÇÃO 17 (Suporte) O *suporte* s_i de um padrão S é a porcentagem de *ID strings* em $\mathcal{I}$ para as quais a cadeia S é um sub-padrão, onde a cadeia S é um sub-padrão de uma *ID*

string I_r se e somente se para cada posição $i \in 1, ..., l$ em S que não seja "*", a i-ésima posição de I_r possui o mesmo valor.

Cada sub-padrão que possui um suporte maior que o parâmetro c especificado pelo usuário define um grupo. Esses sub-padrões são denominados *padrões dos grupos*. As $IDstrings$ permitem definir uma métrica de "quão bom" é um grupo.

DEFINIÇÃO 18 (Razão de Interesse) A *razão de interesse* $IR(S)$ de um padrão S de um grupo C é a razão entre o suporte de S e o produto dos seus suportes fracionários. Isto é

$$IR(S) = \theta/(\beta_1 \cdot \beta_2 \cdot \beta_3 ... \beta_l) \tag{3.10}$$

onde β_i é o suporte fracionário do número de pontos correspondente a m_i em $S = m_1 ... m_l$ (i.e., a fração de $IDstrings$ para as quais sua posição i é igual a m_i) e θ o suporte de S.

Uma razão de interesse maior que 1 é indicativa de um grupo que revela uma afinidade significativamente maior entre os diferentes pontos de dados com base no comportamento do usuário. Os algoritmos para essa abordagem são mostrados a seguir[2].

Algoritmo 4

IPClus(Dados: $\mathcal{D}$, Min Suporte: s, Num. de Pontos de Polarização: k)

inicio

//O vetor de Id's $\mathcal{I} = \{I_1, ..., I_N\}$ tem uma cadeia para cada instância

//e inicializado com NULO

enquanto não** (condição de parada) **fazer

inicio

Amostrar aleatoriamente k pontos $y_1, ..., y_k$ de $\mathcal{D}$;

$\mathcal{E}$=CalcularProjeçõesdePolarização($\mathcal{D}, y_1 ... y_k, s$);

$\mathcal{K}$=SeparaçãoInterativadosGrupos($\mathcal{D}, \mathcal{E}, y_1 ... y_k, s$);

$\mathcal{I}$=AtualizarIdStrings($\mathcal{I}, \mathcal{K}$);

***fim**;*

($\mathcal{C}_1^F ... \mathcal{C}_K^F, Q_1 ... Q_k$)=CriaçãoFinaldosGrupos($\mathcal{I}, s$);

($IR_1 ... IR_K, \mathcal{C}_1^F ... \mathcal{C}_K^F, Q_1 ... Q_k$);

***fim**;*

Algoritmo 5

CalcularProjeçõesdePolarização(Dados: $\mathcal{D}$, Âncoras de polarização: $y_1 ... y_k$, Suporte: s)

inicio

[2]Esse algoritmo será denominado *HC-Cooperative* no Capítulo 4.

$l_c = d/2; \mathcal{E}_c = \mathcal{U};$
enquanto $l_c > 2$ ***fazer***
inicio
para *cada ponto* $x \in \mathcal{D}$ *e cada âncora de polarização* y_i
calcular $PDist(y_i, x, \mathcal{E}_c)$
Seja $\mathcal{M}_i$ *o conjunto de* $N \cdot s$ *pontos*
mais proximos da âncora de polarização y_i*;*
$\mathcal{N} = \mathcal{M}_1 \cup ... \cup \mathcal{M}_k$*;*
Seja z_i *o centroide de* $\mathcal{M}_i$
para $i = 1$ ***ate*** k *calcular* $\mathcal{O}_i(z_i)$ *subtraindo* z_i *de cada ponto*
em $\mathcal{M}_i$*;*
Seja c_{ij} *a covariancia de* (i, j) *para* $\bigcup_{q=1}^{k} \mathcal{O}_q(z_q)$*;*
Seja Δ *a matriz* $d \times d$ $[c_i j]$ *para* $[i, j] \in [d \times d]$*;*
Determinar os autovalores $\mathcal{V} = \{\overline{e}_1 ... \overline{e}_d\}$ *da matriz* Δ*;*
$\mathcal{E}_c$*=Conjunto de* l_c *vetores de* $\mathcal{V}$ *com os menores autovalores*
$l_c = \max\{2, [l_c/2]\}$
fim
retorna$(\mathcal{E}_c)$*;*
fim;

As abordagens apresentadas aqui são as mais relevantes em agrupamento visual baseado em densidade existentes na literatura. O principal problema associado a essas abordagens é a sua baixa escalabilidade. Conforme mencionado anteriormente, centralizamos a pesquisa realizada nesse problema, usando o trabalho de Aggarwal (Aggarwal, 2001) como base.

3.4 Considerações Finais

A escalabilidade dos algoritmos de agrupamento visual baseados em densidade é comprometida principalmente pelo custo dos cálculos das projeções e da estimativa de densidade. É possível usar a abordagem de Hinneburg (Hinneburg et al., 1999) em grandes bancos de dados uma vez que ela atribui ao usuário a escolha da melhor projeção. Ainda assim, o usuário não tem segurança de que uma boa projeção será identificada. Além disso, essa abordagem utiliza uma única projeção para o cálculo do agrupamento, o que reduz a probabilidade de encontrar grupos que possam se revelar em outras projeções.

A abordagem proposta por Aggarwal (Aggarwal, 2001) soluciona esse último problema através do uso de múltiplas projeções, mas não é aplicável a grandes bancos de dados devido

à quantidade de cálculos requeridos para estimar a densidade sobre cada uma das projeções bidimensionais. Além disso, o algoritmo de Aggarwal apresenta problemas de agrupamento quando os grupos se encontram muito próximos entre si.

O presente trabalho realizou extensões ao trabalho de Aggarwal (Aggarwal, 2001) quanto ao desempenho e à qualidade dos grupos. Esses aprimoramentos serão apresentados no Capítulo 4. Em tal capítulo destacamos as vantagens da abordagem original e as razões pelas quais foi escolhida como base para o presente projeto de pesquisa, assim como os ganhos obtidos dos aprimoramentos efetuados nela.

CAPÍTULO

4

HC-Enhanced

4.1 Considerações Iniciais

Conforme mencionado anteriormente, realizar agrupamento em grandes conjuntos de dados multidimensionais apresenta dois problemas principais além da qualidade do agrupamento: escalabilidade e capacidade de lidar com alta dimensionalidade. Escalabilidade usualmente implica na obtenção de um algoritmo de complexidade linear ($O(n)$) ou melhor ($O(\log n)$, etc.). Para sistemas altamente interativos, tais como as técnicas de agrupamento visual (Aggarwal, 2001; Hinneburg et al., 1999; Ribarsky et al., 1999), a escalabilidade é um fator crítico. Algoritmos com complexidade linear e constante alta ($O(kn)$, k constante) não são apropriados para tais sistemas: a interatividade é um fator chave que deve ser incluído em todas as técnicas de mineração visual (Wong, 1999), portanto o tempo de resposta deve ser tão baixo quanto possível.

Tratar a multidimensionalidade é também uma tarefa difícil. As pesquisas efetuadas para lidar com esse problema são baseadas em três abordagens: agrupamento em subespaços, co-agrupamento (Berkhin, 2002) e seleção de características (Witten e Frank, 1999). Essas abordagens estão concentradas em resolver o problema conhecido como "maldição da alta dimensionalidade", que é a incapacidade de gerar estruturas significativas (padrões ou modelos) a partir de dados com alta dimensionalidade. Para os algoritmos de agrupamento na maioria dos casos alta dimensionalidade significa mais de 15 dimensões (Berkhin, 2002; Beyer et al., 1999).

O agrupamento em subespaços refere-se a abordagens que aplicam redução de dimensionalidade antes de agrupar os dados. Diferentes abordagens para redução da dimensionalidade têm sido desenvolvidas, tais como *Principal Components Analysis* (PCA) (Jolliffe, 1986), Fastmap (Faloutsos e Lin, 1995), *Singular Value Decomposition* (SVD) (Strang, 1980), e técnicas baseadas em fractais (Traina et al., 2000; Pagel et al., 2000). Recentemente desenvolvemos uma técnica denominada *Nearest-Neighbor Projection* (NNP) para exploração de dados multidimensionais em representações bidimensionais (Tejada et al., 2003). Embora algumas dessas técnicas apresentem um meio para medir a quantidade de informação perdida, não existem garantias de redução de dimensionalidade sem perder uma quantidade considerável de informação. Além disso, diferentes grupos poderiam ser encontrados em diferentes projeções como mostrado na literatura (Aggarwal, 2001, 2002; Aggarwal e Yu, 2000). Portanto, realizar agrupamento em um único subespaço projetado pode levar a um resultado pobre do processo de agrupamento.

O resultado esperado de todo algoritmo de agrupamento é que ele produza um bom agrupamento. Porém, a avaliação do que é um bom agrupamento não é um assunto trivial, e depende fortemente do conhecimento que o avaliador possui sobre os dados, o domínio dos dados e a aplicação para a qual o agrupamento é gerado. Esse conceito não pode ser estabelecido de forma geral para todas as aplicações e conjuntos de dados. Essa é a razão pela qual a geração de um bom agrupamento para uma aplicação não pode ser conseguida sem a interferência direta do usuário. As técnicas de agrupamento visual se baseiam nesse fato para explorar a capacidade dos seres humanos para tomar decisões, bem como a capacidade de processamento dos computadores. Isso também reduz a complexidade do algoritmo substituindo heurísticas usualmente custosas por decisões do usuário.

Para apresentar uma interface apropriada ao usuário, os algoritmos de agrupamento visual devem apoiar uma idéia intuitiva de grupo; isto é, o conceito de grupo usado pelo algoritmo deve coincidir com a noção de grupo que o usuário possui (regiões densas com formatos arbitrários).

Baseados nesses fatos, podemos estabelecer o que um algoritmo de agrupamento em subespaços deve cumprir para ser apropriado para conjuntos de dados reais:

1. encontrar grupos em múltiplas projeções;
2. aproveitar o conhecimento do domínio que o usuário possui;
3. gerar grupos de formatos arbitrários; e
4. ser escalável.

Além disso, é desejável que se forneçam mecanismos para lidar com os *outliers* e definir uma métrica para o quanto um grupo é consistente com as respostas do usuário.

A Tabela 4.1 apresenta um resumo das características das abordagens mais representativas das diferentes famílias de algoritmos de agrupamento[1]. O algoritmo de Aggarwal (Aggarwal, 2001), o qual denominaremos *HC-Cooperative*, cumpre quase todos os requisitos que estabelecemos como necessários para que um algoritmo seja aplicável a conjuntos de dados reais.

	Escalável	Multidim.	Formato arb.	*Outliers*	Int. do usuário	Metrica
Birch	√					
Chameleon			√			
Cure	√		√	√		
Bubble	√					
Orclus		√		√		
DBScan	√		√	√		
HC-Cooperative		√	√	√	√	√
HD-Eye	√		√	√	√	

Tabela 4.1: Características dos algoritmos de agrupamento mais representativos (Tejada e Minghim, 2003).

O algoritmo *HC-Cooperative* inclui interferência do usuário durante a criação dos grupos. Essa interação fornece ao usuário meios para entender os dados ao apresentar uma visualização dos grupos formados durante o processo, assim como para identificar os *outliers* que o conjunto de dados apresenta.

Além disso, Aggarwal apresenta uma métrica para o quão consistente é um grupo com as respostas do usuário, uma forma eficiente de lidar com os *outliers*, um conceito intuitivo de grupo baseado em densidade (que permite a definição de grupos com formatos arbitrários) e o uso de múltiplas projeções para tratar dados com alta dimensionalidade reduzindo a perda de informação.

Porém, o algoritmo *HC-Cooperative* não pode ser aplicado a grandes conjuntos de dados devido ao alto custo computacional dos processos usados para projetar os dados e para estimar a densidade. Portanto, desenvolvemos mecanismos para reduzir o tempo de execução de tais processos. Tais mecanismos foram introduzidos em vários passos do algoritmo. Porém, o ganho mais importante foi conseguido durante o cálculo de estimativa de densidade, onde utilizamos o fato de que nem todas as instâncias são relevantes para o cálculo de densidade de todos os pontos da malha onde se armazena a densidade.

Além disso, introduzimos uma abordagem para melhorar projeções nos casos em que os grupos não podem ser identificados apropriadamente, baseada na técnica proposta por Tejada *et al.* (Tejada et al., 2003). Tal abordagem pode ser utilizada interativamente pelo usuário em cada iteração (projeção do conjunto de dados) durante o processo de agrupamento. Ela foi inserida em um dos passos do algoritmo de agrupamento constituindo em uma opção para o usuário

[1] Ver o Capítulo 2 para maiores detalhes sobre os algoritmos de agrupamento.

para melhorar o mapa de densidade de cada projeção utilizada durante o agrupamento antes da etapa final de cálculo da densidade. Para tornar tal técnica aplicável a grandes conjuntos de dados, construímos una malha utilizando a triangulação de Delaunay (Edelsbrunner, 2001) de forma a indexar através da malha os vizinhos mais próximos à instância atual e utilizar sua vizinhança para calcular o deslocamento ao invés de todas as instâncias como na abordagem original proposta por Tejada *et al.*

O algoritmo *HC-Cooperative* é descrito em maior detalhe na Seção 4.2, juntamente com um exemplo ilustrativo. Na Seção 4.3 discutimos as modificações realizadas no algoritmo *HC-Cooperative* para reduzir o tempo de processamento e melhorar a qualidade do agrupamento. Na Seção 4.4 apresentamos os resultados do tempo de processamento e da qualidade do agrupamento gerado. Finalmente, a Seção 4.5 é dedicada a apresentar as considerações finais do capítulo. O algoritmo foi implementado em C++ e usa um visualizador desenvolvido no Laboratório de Computação de Alto Desempenho, baseado em OpenGL. Software, código e documentação estão disponíveis em http://www.lcad.icmc.usp.br/~powervis/visual.

4.2 O algoritmo *HC-Cooperative*

O algoritmo *HC-Cooperative* é um algoritmo visual baseado em densidade que utiliza múltiplas projeções para agrupar os dados com o objetivo de reduzir a perda de informação, e interação do usuário para lidar com *outliers* e determinar quando o processo de agrupamento deve parar (i.e. todos os grupos significativos foram descobertos).

O algoritmo pode ser resumido como segue[2]:

Algoritmo 6
HC-Cooperative()
inicio
  repetir
    Amostrar l instâncias a_i aleatoriamente e obter sua r-vizinhança $\mathcal{M}_i$.
    Normalizar os grupos $\mathcal{M}_i$ para obter os grupos $\mathcal{O}_i$.
    Determinar o subespaço bidimensional $\mathbb{S}$ onde os grupos $\mathcal{M}_i$ formem grupos bem definidos.
    Projetar os dados no subespaço $\mathbb{S}$.
    Estimar a densidade no subespaço $\mathbb{S}$.
    Especificar um nivel de ruido λ onde os outliers não sejam incluidos na criação dos grupos.

[2]Uma versão mais detalhada pode ser encontrada no Capítulo 3.

Criar e armazenar os grupos criados na projeção atual.
ate *todos os grupos serem identificados.*
Criar os grupos finais.
fim;

Cada um dos l grupos $\mathcal{M}_i$, onde l é o número de âncoras a_i definido pelo usuário, é formado pelas instâncias na r-vizinhança de a_i (i.e., o resultado de uma busca por abrangência com centro a_i e raio r). Esses grupos são normalizados de tal forma que seus centróides z_i virem $\overline{0}$. Isto é feito subtraindo z_i de todas as instâncias em $\mathcal{M}_i$ gerando o grupo normalizado $\mathcal{O}_i$.

O subespaço bidimensional $\mathbb{S}$ onde os $\mathcal{M}_i$ formam grupos bem definidos é aquele onde o momento de todo $\mathcal{O}_i$ é mínimo. Isto é, o subespaço onde $\sum_{i=1}^{l} \mu(\mathcal{O}_i)$ é mínimo. $\mu(\mathcal{O}_i)$ é o momento de $\mathcal{O}_i$ definido pela expressão $\mu(\mathcal{O}_i) = \sum_{j=1}^{k} d(z'_i, m_j)$. $d(z'_i, m_j)$ é a distância entre o centróide z'_i de $\mathcal{O}_i$ e a instância (m_j). Isto é igual a minimizar $\mu(\bigcup_{i=1}^{l} \mathcal{O}_i)$.

Foi provado por Jolliffe (Jolliffe, 1986) que o subespaço d-dimensional onde um grupo de instâncias $\mathcal{G}$ possuem momento mínimo é aquele definido pelos d autovetores associados aos d menores autovalores da matriz de covariância de $\mathcal{G}$. Isto é, as d menores componentes resultantes de aplicar PCA (Jolliffe, 1986) ao conjunto $\mathcal{G}$ (no nosso caso $\mathcal{G} = \bigcup_{i=1}^{l} \mathcal{O}_i$).

O algoritmo *HC-Cooperative* usa esse resultado para encontrar o subespaço bidimensional $\mathbb{S}$, diminuindo iterativamente a dimensionalidade do conjunto de dados pela metade.

Uma vez que o subespaço $\mathbb{S}$ foi determinado, o conjunto de dados é projetado nele, e a densidade $\hat{f}_D$ é estimada utilizado o *kernel* Gaussiano:

$$\hat{f}_D = \frac{1}{nh^d} \sum_{i}^{n} \frac{1}{\sqrt{2\pi}h} e^{-\frac{(x - x_i)^2}{2h^2}} \tag{4.1}$$

onde x_i é a i-ésima instância, h é o fator de suavização, n é o número de instâncias no conjunto de dados e x representa um ponto na malha cartesiana bidimensional onde a densidade será armazenada.

A estimativa de densidade é mapeada em um *heightmap* apresentado ao usuário. O usuário então especifica o valor para o nível de ruído λ de forma que os *outliers* não sejam incluídos nos grupos formados na iteração (projeção) atual. Isso é feito utilizando técnicas de interação tais como planos de corte (Figura 4.1). Com λ estabelecido, os grupos são criados procurando as células da malha que possuem pelo menos três de seus vértices acima do nível de ruído. Tais células formam componentes conexas, cada uma representando um grupo. A cada grupo é atribuído um identificador diferente que será adicionado à $IDString$ de cada instância pertencente ao grupo. A $IDString$ de uma instância armazena os identificadores dos grupos aos

quais a instância pertence nas diferentes projeções. Isso pode ser observado na Tabela 4.2, a qual reproduzimos do Capítulo 3.

Figura 4.1: Plano de corte sobre um *heightmap* representando a estimativa de densidade para especificar o nível de ruído λ sobre uma projeção do conjunto de dados Iris.

	Projeção				
Instância	I	II	III	IV	V
1	1	3	2	1	2
2	2	1	1	3	3
3	3	1	3	2	1
4	1	3	2	1	2
5	1	3	2	2	2

Tabela 4.2: Exemplo de $IDStrings$. O elemento em cada célula da tabela identifica o grupo ao qual pertence a instância em tal projeção. É altamente provável que as instâncias 1, 4, e 5 pertençam ao mesmo grupo no espaço multidimensional original.

Uma vez que o usuário decide (usando as visualizações) que todos os grupos foram identificados, os grupos finais são criados procurando subcadeias dentro das $IDString$. Cada subcadeia tem em sua i-ésima posição o identificador de um grupo criado na i-ésima projeção, ou "$*$", onde "$*$" significa que tal posição não será considerada relevante quando as instâncias que coincidam com a subcadeia sejam procuradas. Portanto, cada subcadeia representa um grupo final. Por exemplo, para o caso da Tabela 4.2, a subcadeia "132*2" define o grupo formado pelas instâncias 1,4, e 5, enquanto a subcadeia "13213" define o grupo formado pelas instâncias 1 e 4. Isso fornece um meio para medir o grau de consistência de um grupo com as respostas do usuário. Tal métrica é denominada *razão de interesse* a qual é dada pela expressão:

$$IR(S) = \theta/(\beta_1 \cdot \beta_2 \cdot \beta_3 ... \beta_l) \tag{4.2}$$

onde β_i é o suporte fracionário do grupo m_i na subcadeia $S = m_1 ... m_l$ (i.e., a fração de $IDStrings$ para as quais a i-ésima posição é igual a m_i) e θ é o suporte de S (i.e., o número

de $IDStrings$ que coincidem com S, onde uma $IDString$ coincide com uma subcadeia $S = m_1...m_l$ se elas são iguais exceto nas posições onde $m_i = *$).

Uma razão de interesse maior que 1 indica que o grupo possui uma afinidade significativa entre suas instâncias com base no comportamento do usuário.

Para ilustrar o algoritmo, apresentamos a seguir um exemplo de agrupamento em um conjunto de dados de 15 instâncias 5-dimensionais, mostrado na Tabela 4.3. Esse conjuntos de dados possui três grupos formados pelas instâncias $1-5$, $6-10$, e $11-15$.

Instância	Dim. 1	Dim. 2	Dim. 3	Dim. 4	Dim. 5	Grupo
1	24.805203	70.571593	67.511839	32.702514	15.898942	0
2	22.475693	68.418753	67.826733	34.277486	14.560532	0
3	25.499799	68.917441	66.923558	31.870170	11.870866	0
4	25.507545	72.128011	68.944650	33.214440	17.424723	0
5	20.180426	70.984498	69.368377	35.387339	13.570690	0
6	10.126638	58.299350	29.998860	8.7546100	70.263157	1
7	9.1479970	53.566211	24.655602	10.315620	70.768003	1
8	8.7200150	55.899258	25.430168	12.299597	70.380566	1
9	10.724900	54.662898	26.281604	8.9593930	72.198301	1
10	8.9582850	54.410600	27.514455	12.602682	68.596397	1
11	70.753605	83.065990	44.915836	27.497272	27.297358	2
12	65.586945	79.915609	44.629438	27.833484	27.588379	2
13	70.166474	82.800126	43.811622	29.731629	27.520091	2
14	67.473108	77.690808	43.635263	30.400894	27.359884	2
15	70.247318	77.689025	46.154428	28.584453	29.690863	2

Tabela 4.3: Dados sintéticos de exemplo para ilustrar o algoritmo *HC-Enhanced*.

O resultado da execução do algoritmo sobre esse conjunto de dados será detalhado em cada passo para a primeira iteração.

Passo 1: amostrar l instâncias a_i aleatoriamente e obter sua r-vizinhança $\mathcal{M}_i$

Do conjunto de dados mostrado na Tabela 4.3, amostramos aleatoriamente duas instâncias: $a_1 = 8$ e $a_2 = 9$. A r-vizinhança ($r = 94.3$) de cada uma delas constituem os grupos $\mathcal{M}_1 = \{x/1 \leq x \leq 15\}$ e $\mathcal{M}_2 = \{x/1 \leq x \leq 10\}$ respectivamente.

Passo 2: normalizar os grupos $\mathcal{M}_i$ para obter os grupos $\mathcal{O}_i$

No segundo passo do algoritmo, normalizamos os grupos $\mathcal{M}_1$ e $\mathcal{M}_2$ para obter os grupos $\mathcal{O}_1$ e $\mathcal{O}_2$ formados pelas instâncias normalizadas apresentadas nas Tabelas 4.4 e 4.5 respectivamente. A normalização precisa do cálculo dos centróides $z_i = \frac{1}{N_i}\sum_{j=1}^{N_i} x_{ij}$, onde N_i é o número de instâncias de $\mathcal{M}_i$ e x_{ij} é a j-ésima instância de $\mathcal{M}_i$. As instâncias do grupo $\mathcal{M}_i$ são normalizadas para formar o grupo $\mathcal{O}_i$ subtraindo z_i de cada x_{ij}.

Instância	Dim. 1	Dim. 2	Dim. 3	Dim. 4	Dim. 5
1	-9,22	1,97	21,01	8,41	-21,77
2	-11,55	-0,18	21,32	9,98	-23,11
3	-8,53	0,32	20,42	7,57	-25,8
4	-8,52	3,53	22,44	8,92	-20,24
5	-13,84	2,38	22,86	11,09	-24,1
6	-23,9	-10,3	-16,51	-15,54	32,6
7	-24,88	-15,04	-21,85	-13,98	33,1
8	-25,3	-12,7	-21,08	-12	32,71
9	-23,3	-13,94	-20,23	-15,34	34,53
10	-25,07	-14,19	-18,99	-11,69	30,93
11	36,73	14,46	-1,59	3,2	-10,37
12	31,56	11,31	-1,88	3,54	-10,08
13	36,14	14,2	-2,7	5,44	-10,15
14	33,45	9,09	-2,87	6,11	-10,31
15	36,22	9,09	-0,35	4,29	-7,98

Tabela 4.4: Grupo normalizado $\mathcal{O}_1$.

Instância	Dim. 1	Dim. 2	Dim. 3	Dim. 4	Dim. 5
1	8,19	7,79	20,07	10,66	-26,65
2	5,86	5,63	20,38	12,24	-27,99
3	8,89	6,13	19,48	9,83	-30,68
4	8,89	9,34	21,5	11,18	-25,13
5	3,57	8,2	21,92	13,35	-28,98
6	-6,49	-4,49	-17,45	-13,28	27,71
7	-7,47	-9,22	-22,79	-11,72	28,21
8	-5,89	-8,12	-21,16	-13,08	29,65
10	-7,66	-8,38	-19,93	-9,44	26,04

Tabela 4.5: Grupo normalizado $\mathcal{O}_2$.

Passo 3: determinar o subespaço bidimensional $\mathbb{S}$

O primeiro processo necessário para obter o subespaço bidimensional $\mathbb{S}$ é calcular a matriz de covariância de $\bigcup_{i=1}^{l} \mathcal{O}_i$. A covariância $c(C_1, C_2)$ das dimensões C_1 e C_2 por exemplo, é dada pela expressão $c(C_1, C_2) = \frac{1}{n}\sum_{k=1}^{n}[x_{k1} - m_1][x_{k2} - m_2]$, onde x_{kr} é o valor da dimensão r da k-ésima instância $x_k \in \bigcup_{i=1}^{l} \mathcal{O}_i$, n o número de instâncias em $\mathcal{O}_i$, e m_r a média dos valores da dimensão r.

A matriz de covariância M, onde $M(k, l) = c(C_k, C_l)$, para nosso exemplo é como segue.

$$M = \begin{bmatrix} 1 & 0.869914 & 0.260048 & 0.516132 & -0.503188 \\ 0.869914 & 1 & 0.675438 & 0.837409 & -0.833138 \\ 0.260048 & 0.675438 & 1 & 0.945848 & -0.960029 \\ 0.516132 & 0.837409 & 0.945848 & 1 & -0.991416 \\ -0.503188 & -0.833138 & -0.960029 & -0.991416 & 1 \end{bmatrix} \quad (4.3)$$

Aplicando redução de Householder e a decomposição QL (Press et al., 1992) a M, obtemos os seguintes autovetores e autovalores:

Autovalor 1:	0.0114				
Autovetor 1:	0.3732	-0.3768	0.4286	-0.6403	-0.3534
Autovalor 2:	0.0031				
Autovetor 2:	-0.2640	0.1669	-0.5544	-0.2143	-0.7409
Autovalor 3:	0.0261				
Autovetor 3:	-0.3305	0.7027	0.3001	-0.5161	0.2008
Autovalor 4:	0.9479				
Autovetor 4:	-0.7534	-0.3444	0.4728	0.2023	-0.2214
Autovalor 5:	4.0112				
Autovetor 5:	-0.3375	-0.4664	-0.4416	-0.4864	0.4864

Os autovetores associados aos $\lfloor d/2 \rfloor$ menores autovalores definirão o novo espaço de dados. Com os dados projetados nesse espaço o processo é repetido até $\lfloor d/2 \rfloor = 2$. No nosso exemplo, a primeira iteração cumpre tal condição. Portanto os autovetores 1 e 2 definem o nosso espaço projetado $\mathbb{S}$.

Passo 4: projetar os dados no subespaço $\mathbb{S}$

Nesse passo cada instância projetada $x_i' = x_i \times W^T$ é calculada, onde x_i é a instância no conjunto de dados original e W é a matriz formada pelos autovetores escolhidos.

Para nosso exemplo, o conjunto de dados projetados é mostrado na Tabela 4.6.

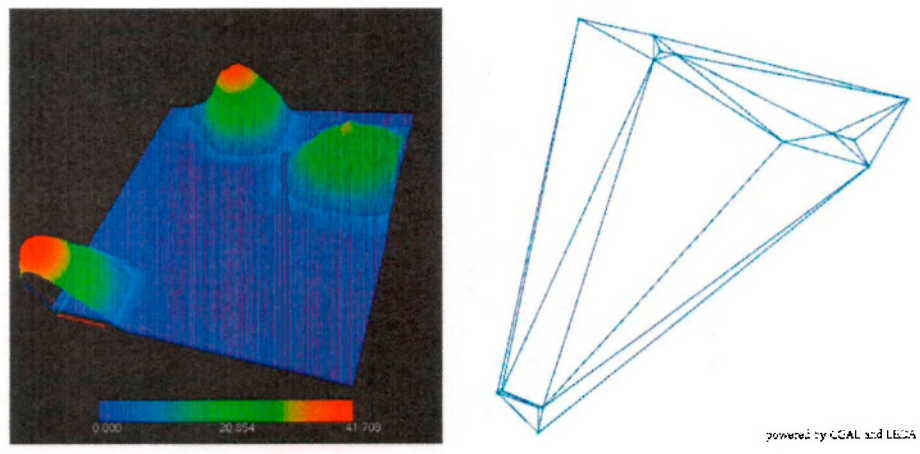
Passo 5: estimar a densidade no subespaço $\mathbb{S}$

A estimativa de densidade para o conjunto de dados é calculada utilizando a expressão 4.1. Para isso construímos uma malha cartesiana que armazenará a densidade estimada para cada um dos seus vértices. Para nosso exemplo utilizamos uma malha 40×40.

O resultado do cálculo de densidade para nosso exemplo na primeira iteração é apresentado na Figura 4.2.

Instância	Dim. 1	Dim. 2
1	-50.9915	-14.9528
2	-50.2562	-15.4115
3	-47.9618	-12.3658
4	-52.9518	-15.53
5	-49.5809	-16.9349
6	-63.5135	-35.7693
7	-61.7914	-37.8195
8	-61.8566	-39.6598
9	-63.6954	-36.5843
10	-62.0666	-37.6798
11	-55.842	-12.8939
12	-55.1324	-14.076
13	-55.7631	-14.9956
14	-55.8318	-14.523
15	-59.2992	-12.0678

Tabela 4.6: Conjunto de dados projetados.



(a) (b)

Figura 4.2: (a) Densidade estimada em uma das três projeções realizadas para o exemplo. (b) Triangulação de Delaunay (Edelsbrunner, 2001) da projeção.

Passo 6: especificar um nível de ruído λ apropriado

O nível de ruído foi definido utilizando um *plano de corte* como ilustrado na Figura 4.2.

Os níveis de ruído para as três iterações realizadas foram 1.25124, 5.81586, e 5.81586. A densidade total para cada projeção foi 42.3, 41.708, e 41.708 respectivamente.

Passo 7: criar e armazenar os grupos criados na projeção atual

Nesse passo, procuraram-se as células da malha que possuem 3 dos seus 4 vértices acima do nível de ruído. Essas células são classificadas como *células elementares*. Para construir os grupos na presente iteração começa-se com uma célula elementar semente expandindo o grupo recuperando todos os vizinhos dela que sejam elementares. Para cada uma das novas células elementares adicionadas ao grupo o processo é repetido até que todas as células elementares alcançáveis a partir da semente tenham sido adicionadas. Uma nova semente é escolhida e o processo repetido até que todas as células elementares pertençam a algum grupo.

Uma vez agrupadas as células, as instâncias projetadas são atribuídas ao grupo de células respectivo. O identificador de cada grupo de células é adicionado no final da $IDString$ de cada instância. Para as três iterações do nosso exemplo, as $IDStrings$ das instâncias resultaram como mostrado na Tabela 4.7

Instância	Proj. 1	Proj. 2	Proj. 3
1	0	1	1
2	0	1	1
3	0	1	1
4	0	1	1
5	0	1	1
6	2	0	0
7	2	0	0
8	2	0	0
9	2	0	0
10	2	0	0
11	1	2	2
12	1	2	2
13	1	2	2
14	1	2	2
15	1	2	2

Tabela 4.7: $IDStrings$ resultantes após as três iterações do nosso exemplo.

Os passos 1-7 apresentados são realizados em todas as iterações (projeções). O número de iterações é especificado pelo usuário. No nosso caso, utilizamos 3 iterações para nosso exemplo. Os resultados obtidos de tais iterações são utilizados para gerar os grupos finais. A seguir apresentamos o passo final que cria os grupos que serão considerados como o resultado do algoritmo.

Passo final: criar os grupos finais

A criação final dos grupos é realizada utilizando a informação da Tabela 4.7. Procuramos pelas subcadeias com cobertura maior à especificada pelo usuário. No nosso caso utilizamos

uma cobertura de 2 (isto é, como o tamanho das $IDStrings$ é 3, podemos ter somente uma posição com "*"). Exemplos de subcadeias válidas são:

*	0	0
*	1	2
1	*	0
2	2	1
*	1	2
2	2	0
1	2	*
1	2	2

As subcadeias (ou padrões) 011, 200, e 122, geram os grupos com razão de interesse maior. Tais grupos são $G_1 = \{x/1 \leq x \leq 5\}$, $G_2 = \{x/6 \leq x \leq 10\}$, e $G_3 = \{x/11 \leq x \leq 15\}$ respectivamente. Os três grupos possuem uma razão de interesse igual a 45. Para ilustrar o cálculo da razão de interesse mostramos a seguir o processo para o grupo definido pela subcadeia 011.

A razão de interesse é dada pela equação 4.2. No nosso caso, o suporte $\theta = 5$. Isto é, o número de instâncias cuja $IDString$ cumpre com a subcadeia 011 é 5. β_1 é o suporte fracionário da primeira posição da subcadeia. Isto é, a fração de $IDStrings$ que cuja primeira posição é igual à primeira posição da subcadeia 001. Portanto, $\beta_1 = 1/3$. De forma similar, podemos calcular $\beta_2 = 1/3$ e $\beta_3 = 1/3$. Da expressão 4.2 podemos concluir que IR("001") = 45.

Dessa forma concluímos o exemplo da execução do algoritmo *HC-Cooperative* sobre um conjunto de dados. A próxima seção apresenta as melhorias realizadas nele em dois aspectos: tempo de processamento e qualidade do agrupamento quando os grupos se encontram muito próximos entre si nas projeções.

4.3 O algoritmo *HC-Enhanced*

Nesta seção apresentamos as melhorias realizadas em termos de custo computacional e melhoria da definição dos grupos no algoritmo *HC-Cooperative* originando a versão *HC-Enhanced*[3], a qual possui uma performance radicalmente melhor à do algoritmo original em termos de tempo processamento, enquanto mantém a qualidade do agrupamento obtido. As melhorias foram concentradas em três processos principais: a busca por abrangência realizada para formar os

[3]Disponível em http://www.lcad.icmc.usp.br/~powervis/visual/.

grupos $\mathcal{M}_i$ (passo 1), a busca pelo subespaço $\mathbb{S}$ (passo 3), e o cálculo da estimativa de densidade (passo 5), o qual é o ponto chave na melhora em tempo de processamento obtida. Além disso, dentro do passo 3 introduzimos um mecanismo para melhorar projeções nos casos em que os grupos não possam ser identificados de forma clara.

4.3.1 Passo 1 - Buscas por abrangência com Omni-sequencial

A *Omni-sequencial* é parte de uma família de estruturas de dados proposta por Santos *et al.* (Santos-Filho et al., 2001) que explora o fato de que a dimensionalidade intrínseca[4] dos conjuntos de dados reais é usualmente consideravelmente baixa.

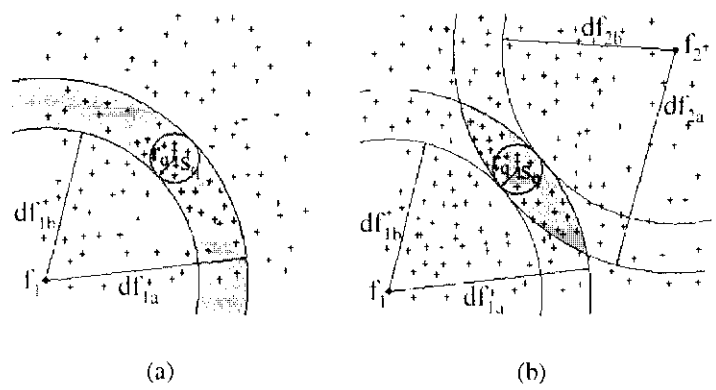
O *conceito Omni* é a base de tal família de estruturas, o qual é basicamente uma transformação de coordenadas na denominada *base focos-Omni*. Essa base é um conjunto $\mathcal{F} = \{f_1, f_2, ..., f_k\}$ de instâncias pertencentes ao conjunto de dados métrico a ser indexado denominadas *focos* ou *pontos focais*. Usando essa base, as *coordenadas Omni* são calculadas para cada instância s como sendo as distâncias entre s e cada foco f_i. O conjunto de dados é então indexado usando tais coordenadas, mantendo os dados originais em um arquivo diferente. O número de focos pode ser estimado como a dimensionalidade fractal, a qual é uma aproximação da dimensionalidade intrínseca, do conjunto de dados. Um algoritmo linear para calcular a dimensão fractal de um conjunto de dados pode ser encontrado no trabalho de Traina *et al.* (Traina et al., 2000).

O resultado de utilizar o conceito Omni é formado por duas partes. Primeiro, a indexação se beneficia da "redução de dimensionalidade" efetuada permitindo que a estrutura de dados indexe conjuntos de dados de alta dimensionalidade sem sofrer da "maldição da alta dimensionalidade". Segundo, quando uma busca por abrangência é efetuada, uma quantidade considerável de cálculos de distância é podada como mostrado na Figura 4.3.

Em uma busca por abrangência com centro s_q e raio r_q, as distâncias df_i de cada foco ao ponto de consulta s_q são calculadas e utilizadas para estabelecer um intervalo para cada foco f_i. Define-se para cada foco uma região calculada subtraindo a bola com centro em f_i e raio $df_ia = df_i - r_q$ da bola com centro em f_i e raio $df_ib = df_i + r_q$, como mostrado na Figura 4.3a. A intersecção das regiões formadas por todos os focos define a *região de fronteira Omni mínima* ($mbOr$). As instâncias pertencentes à $mbOr$ constituem o resultado parcial da busca por abrangência. Portanto, somente as distâncias entre s_q e tais instâncias devem ser calculadas.

É possível usar esse conceito junto a qualquer estrutura de dados. Porém, o ganho resultante de usar estruturas como a R*-Tree (Beckmann et al., 1990) sobre uma abordagem sequencial é

[4] A dimensionalidade do espaço realmente ocupado pelos dados (uma linha em $\mathbb{R}^n$ possui dimensionalidade intrínseca igual a 1).



(a) (b)

Figura 4.3: Busca por abrangência com centro s_q e raio r_q. A região sombreada mostra as instâncias retornadas da poda. (a) Utilizando um foco f_1. (b) Utilizando dois focos f_1 e f_2.

negligenciável quando poucas consultas são realizadas, como no nosso caso. Portanto, usamos a versão sequencial (*Omni-sequencial*) no primeiro passo do algoritmo para acelerar as buscas por abrangência realizadas para gerar os grupos $\mathcal{M}_i$.

4.3.2 Passo 3 - Encontrando e melhorando o subespaço de polarização

No passo 3 do algoritmo *HC-Cooperative*, o subespaço $\mathbb{S}$ (o *subespaço de polarização*) onde $\mathcal{M}_i$ formam grupos bem definidos deve ser encontrado. Para fazer isso, como discutido na Seção 4.2, a dimensionalidade do conjunto de dados é diminuída iterativamente pela metade projetando, na i-ésima iteração, o conjunto de dados no subespaço d_i-dimensional definido pelos d_i autovetores associados aos d_i menores autovalores da matriz de covariância do conjunto de dados, onde d_i é a metade da dimensionalidade atual. Isto é o mesmo do que aplicar PCA (Jolliffe, 1986) ao conjunto $\bigcup_{i=1}^{l} \mathcal{O}_i$[5], escolhendo as d_i últimas componentes como a base do subespaço d_i-dimensional.

Matematicamente, a autodecomposição ou PCA, se reduz a resolver a equação $RW = W\Lambda$ onde R é qualquer matriz quadrada (Golub e Loan, 1991), como a matriz de covariância de $\bigcup_{i=1}^{l} \mathcal{O}_i$. Devido a que este é um processo custoso, realizamos experimentos usando os algoritmos *online-PCA* (Rao e Principe, 2000) e *Simple-PCA* (Partridge e Calvo, 1998) para encontrar o subespaço $\mathbb{S}$. Ambos os métodos são baseados no fato de que um autovetor w é diretamente relacionado com sua versão rotacionada por R como na equação:

$$w = \frac{Rw}{w^T Rw} \tag{4.4}$$

[5]Como mencionado na Seção 4.2, $\mathcal{O}_i$ é a versão normalizada de $\mathcal{M}_i$ ao redor de $\overline{0}$

Então, é possível estabelecer um processo iterativo como:

$$w(n) = \frac{R(n)w(n-1)}{w^T(n-1)R(n)w(n-1)} \tag{4.5}$$

onde $R(n)$ é a estimativa da matriz de covariância no tempo n.

As menores componentes são calculadas utilizando a técnica de deflação, subtraindo os autocomponentes prévios do sinal original x como na expressão:

$$x_i' = x_i - (w^T x_i)w \tag{4.6}$$

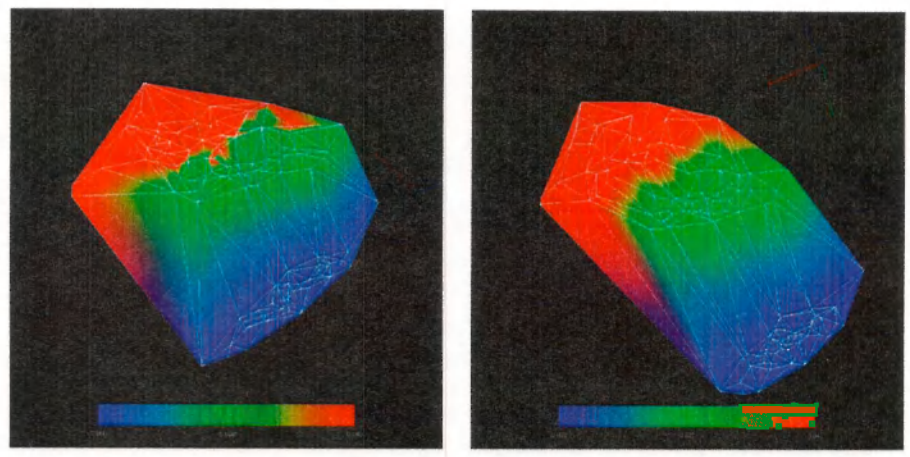
onde x_i é a i-ésima amostra do sinal x (a i-ésima instância).

Dos resultados, pudemos notar que devido ao fato desses algoritmos necessitarem os autovetores prévios para calcular o seguinte, e devido a que precisamos sempre dos últimos, o tempo gasto por eles é maior ao tempo gasto pelo PCA original. Além disso, devido a que o PCA é realizado sobre um amostra do conjunto de dados ($\bigcup_{i=1}^{l} \mathcal{O}_i$), o tempo gasto pelo PCA é negligenciável comparado ao tempo gasto por outros processos. Devido a isso, concentramos nossos esforços em encontrar um meio para diminuir o tempo de processamento no passo de estimativa de densidade, o qual é o processo mais custoso do algoritmo e a matéria da próxima subseção, e em melhorar as projeções nas quais os grupos não estão bem definidos.

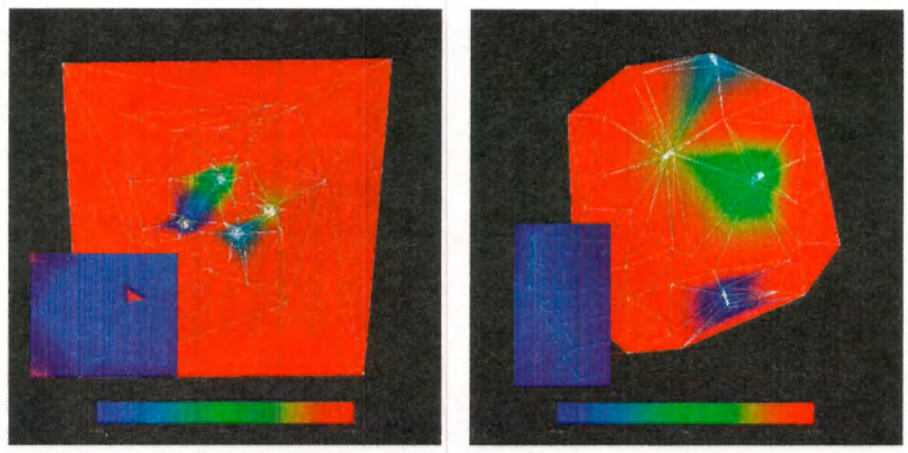
Em alguns espaços de polarização os grupos podem não ser identificados de forma clara, pois não é para todo conjunto de âncoras a_i e todo raio r definindo os grupos iniciais $\mathcal{M}_i$, que o subespaço de polarização obtido é um no qual os grupos no espaço multidimensional original formam grupos bem definidos. Portanto, propomos uma abordagem para melhorar tais projeções baseados na técnica proposta por Tejada *et al.* (Tejada et al., 2003). A abordagem utiliza o fato de que a relação entre as distâncias no espaço original e no subespaço bidimensional deve ser constante para todo par de instâncias (x_i, x_j). Na Figura 4.4, assim como no trabalho mencionado, mostramos que isso gera uma melhor projeção quando aplicado aos resultados de técnicas como Fastmap (Faloutsos e Lin, 1995) ou PCA (Jolliffe, 1986). A técnica é apresentada a seguir.

Para uma instância x_i, calculamos o vetor $\overrightarrow{v_{ij}}$ a partir dela a cada instância x_j em sua vizinhança. Então aplicamos uma perturbação a cada instância na vizinhança na direção do vetor calculado para tal instância. Para fazer com que o processo seja apropriado para grandes conjuntos de dados, utilizamos a triangulação de Delaunay (Edelsbrunner, 2001) para recuperar a vizinhança de uma instância projetada através da estrela do ponto correspondente.

O processo, para cada iteração, pode ser resumido como segue:



(a)



(b)

Figura 4.4: Melhorias das projeções Fastmap usando a abordagem proposta.(a) Conjunto de dados Iris. (b) Um conjunto sintético 10-dimensional com 1000 instâncias. Em ambos os casos a cor mapeia o grupo ao qual a instância pertence. Em (b) mostramos uma ampliação do grupo mapeado em azul. Uma instância classificada como ruído (vermelho) é parte do grupo originalmente. Na projeção melhorada a instância é removida do grupo.

Algoritmo 7

Melhorando-Projeções()

inicio

Calcular a triangulação de Delaunay para a projeção atual.

para *cada instância projetada* p.

inicio

Obter a estrela S *de* p.

para *cada ponto s em S.*
inicio
Calcular $\overrightarrow{v}$ como sendo o vetor de p a s.
Deslocar s na direção de $\overrightarrow{v}$ uma fração de Δ.
fim
fim
Normalizar a projeção ao intervalo [0,1].
fim;

O erro Δ é dado por:

$$\Delta = \frac{d(p^n, s^n) - dmin}{dmax - dmin} - d_2(p, s) \tag{4.7}$$

onde $d_2(p, s)$ é a distância entre p e s no subespaço projetado bidimensional, $d(p^n, s^n)$ é a distância entre os respectivos pontos p^n e s^n no espaço n-dimensional original, e $dmax$ e $dmin$ são as distâncias máxima e mínima respectivamente no espaço n-dimensional.

Esse processo tenta convergir para uma projeção ótima. Porém, devido a que a normalização das distâncias é um processo custoso, ela não pode ser realizada em cada iteração para o espaço projetado. Essa é a razão pela qual as distâncias são normalizadas uma única vez para o espaço original, e realizamos uma normalização das instâncias projetadas ao invés das distâncias projetadas em cada iteração. Portanto, o processo gera realmente uma aproximação do caso ótimo.

Para ilustrar o resultado desse processo, a Figura 4.5 mostra algumas das iterações para uma projeção do conjunto de dados utilizado no exemplo apresentado na Seção 4.2. Podemos notar que na última iteração a projeção apresenta os dados melhor agrupados.

Podemos aproximar o erro $Q(P)$ da projeção P obtida em cada iteração da seguinte forma:

$$Q(P) = \frac{1}{2M} \sum_{k=1}^{M} |\Delta_k| \tag{4.8}$$

Note-se que o erro dado por $Q(P)$ é um erro relativo que é útil para comparar diferentes projeções do mesmo conjunto de dados. Porém, como a equação 4.7 define um deslocamento aproximado, $Q(P)$ sobre-estima o erro da projeção. Isto é, a projeção ótima P' de um conjunto de dados possui erro $Q(P') \geq 0$.

Essa abordagem pode ser utilizada interativamente pelo usuário sobre a projeção gerada em cada iteração (dentro do passo 3 do algoritmo), gerando uma nova projeção e portanto um novo mapa de densidade. O mapa de densidade é gerado de forma melhorada comparado com a abordagem de Aggarwal, como descrito na próxima subseção.

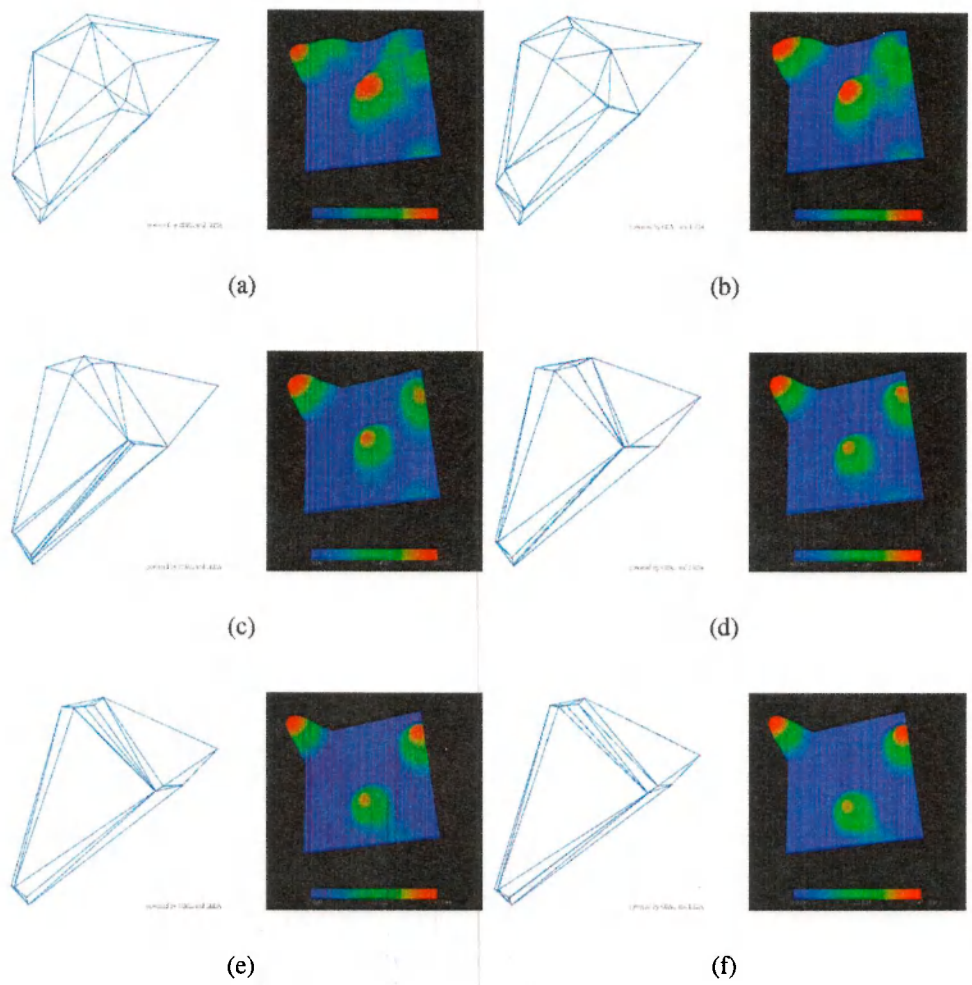


Figura 4.5: Melhoria de uma projeção do conjunto de dados de exemplo utilizado na Seção 4.2. (a) Iteração 1. (b) Iteração 7. (c) Iteração 13. (d) Iteração 19. (e) Iteração 25. (f) Iteração 31.

4.3.3 Passo 5 - Estimativa de densidade

Esse passo é o ponto crítico do algoritmo. Todos os passos do algoritmo possuem complexidade $O(kn)$, onde k é uma constante e n o número de instâncias. Porém, ao contrário de outros passos, na estimativa de densidade k é uma constante alta (o número de pontos na malha que armazenará a densidade).

Propomos uma abordagem para reduzir a constante k baseados na contribuição individual de cada instância à densidade acumulada nos pontos da malha.

Na Figura 4.6 apresentamos a contribuição de uma instância x_i no cálculo da estimativa de densidade de um ponto da malha x como função do fator de suavização h e da distância entre x e x_i. Podemos observar que, após certa distância, e dependendo do fator de suavização, a contribuição da instância x_i ao cálculo da densidade acumulada no ponto da malha pode ser

negligenciado. Podemos observar esse fato melhor na Figura 4.7 onde a contribuição $\hat{f}_D^i$ da instância x_i na densidade acumulada no ponto da malha x é apresentado para diferentes valores do fator de suavização h. Tal contribuição é dada pela expressão:

$$\hat{f}_D^i = \frac{1}{\sqrt{2\pi}h} e^{-\frac{(x-x_i)^2}{2h^2}} \tag{4.9}$$

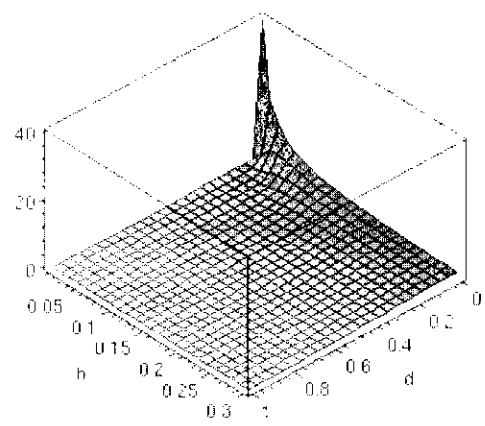
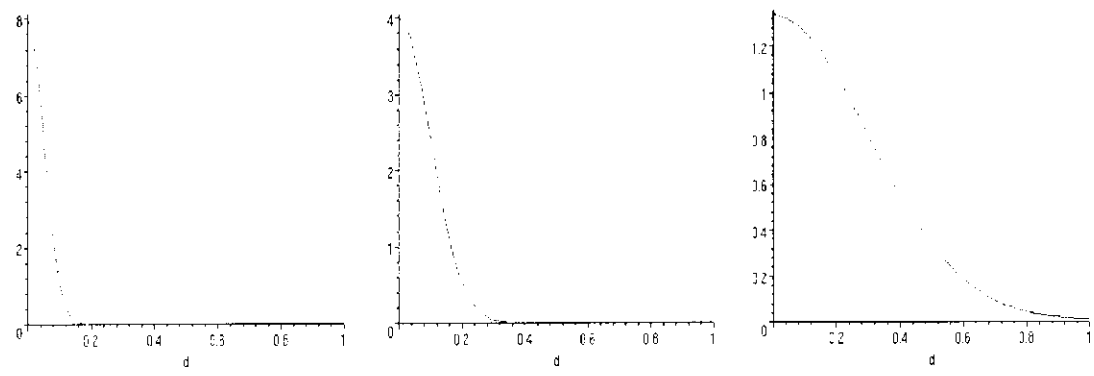


Figura 4.6: Contribuição de uma instância x_i no cálculo da densidade acumulada no ponto da malha x como função do fator de suavização h e a distância d entre x e x_i.



(a) (b) (c)

Figura 4.7: Contribuição de uma instância x_i no cálculo da densidade acumulada no ponto de malha x como função da distância d entre x e x_i para (a) $h = 0,05$, (b) $h = 0,1$ e (c) $h = 0,3$.

Devido ao fato do conjunto de dados ser normalizado ao intervalo $[0, 1]$ em todas as dimensões durante a projeção, o fator de suavização h se encontra entre 0 e 1, sendo que para os conjuntos de dados testados o valor ótimo de h segundo a fórmula apresentada por Silverman (Silverman, 1986) é cerca de 0,1. Podemos observar que, para o valor ótimo de h (0,1), a con-

tribuição de uma instância ao cálculo da densidade acumulada em um ponto da malha diminui rapidamente à medida que a distância aumenta.

Portanto, definimos a vizinhança v das células da malha para as quais uma instância é considerada relevante para o cálculo da densidade acumulada nelas. Na Figura 4.8 mostramos um exemplo da vizinhança $v = 2$ de uma instância da malha 9×9.

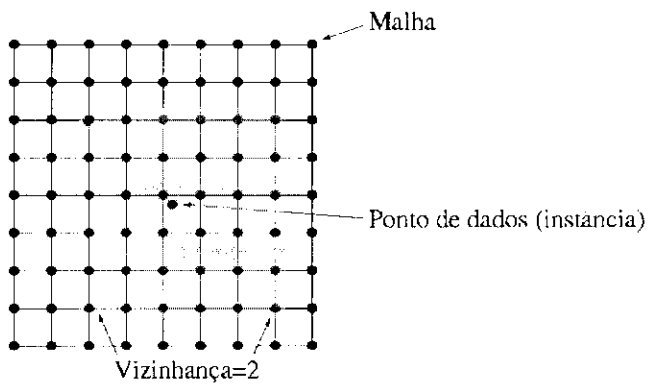


Figura 4.8: Vizinhança de uma instância.

A abordagem proposta para calcular a estimativa de densidade é dada a seguir:

Algoritmo 8
CalcularEstimativaDeDensidadeMelhorado()
inicio
  para *cada instância* x_i *com coordenadas bidimensionais* x_{ix} *e* x_{iy}
  inicio
    $indice_x = \lfloor x_{ix}.m \rfloor$.
    $indice_y = \lfloor x_{iy}.m \rfloor$.
    Testar condições de fronteira ($indice_x + v + 1 > m$,
      $indice_x - v < 0$, $indice_y + v + 1 > m$, *ou* $indice_y - v < 0$).
    Acumular a contribuição de x_i *para todos os pontos na malha com indice no*
      intervalo $[indice_x - v, indice_x + v + 1; indice_y - v, indice_y + v + 1]$
  fim
fim;

onde m é o número de amostras da malha (i.e., para uma malha 30×30, $m = 30$) e v é a vizinhança.

A abordagem original, apresentada aqui para comparação, é como segue.

Algoritmo 9
CalcularEstimativaDeDensidadeOriginal()

inicio
  para *cada ponto da malha* x
  inicio
    para *cada instância* x_i
    inicio
      Acumular a contribuição de x_i *para* x.
    fim
  fim
fim;

A vizinhança ótima depende do número de amostras m da malha, assim como da precisão p requerida e do fator de suavização h. A precisão indica a contribuição mínima que uma instância deve fornecer ao cálculo da densidade acumulada em um ponto da malha para que ela seja considerada relevante.

Tal relação é dada pela expressão:

$$v = m\sqrt{-2h^2 \ln(p\sqrt{2\pi}h)} \tag{4.10}$$

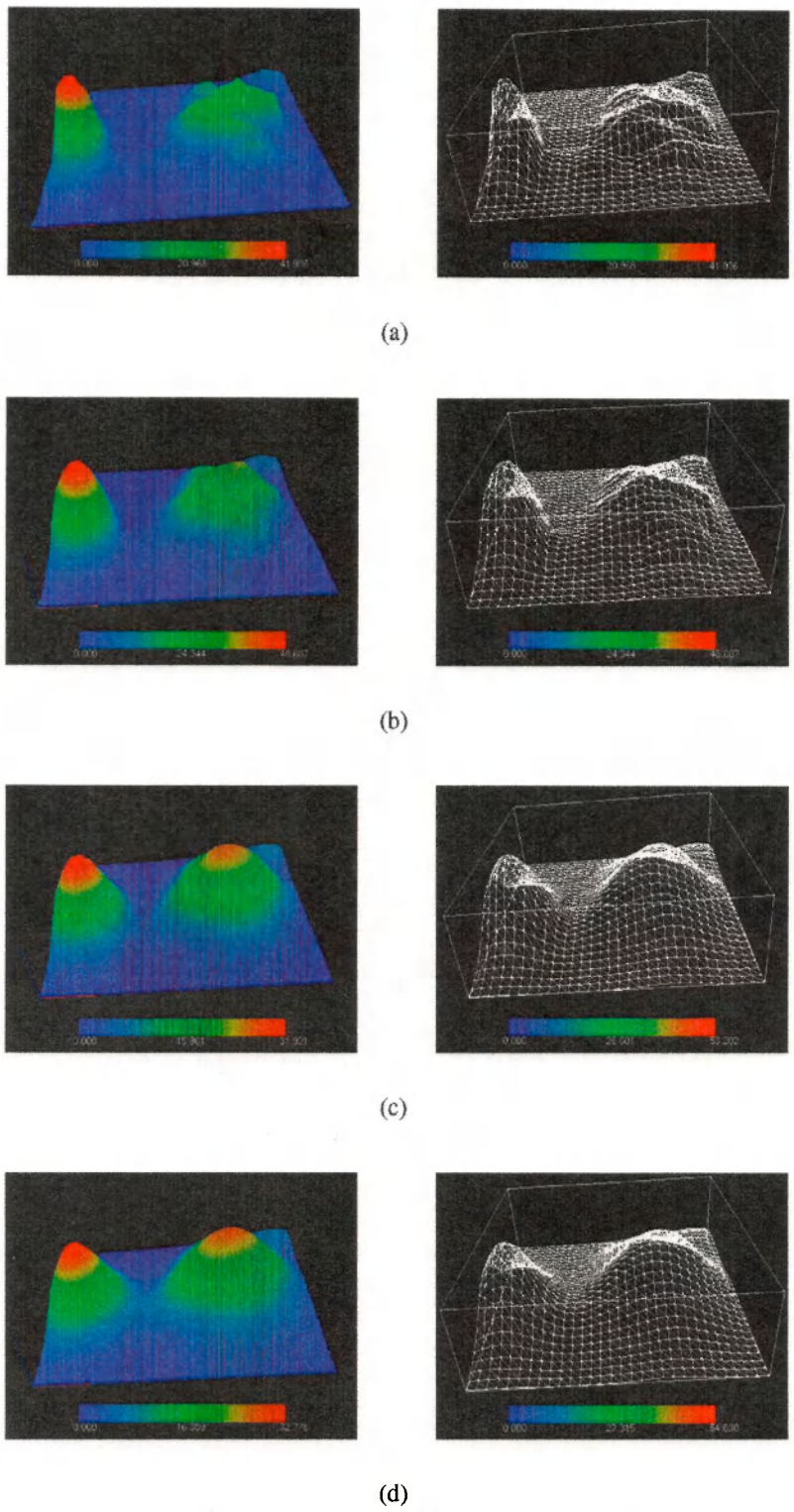
diretamente derivada da equação 4.9.

A Figura 4.9 mostra os resultados do cálculo da estimativa de densidade para diferentes vizinhanças v para o conjunto de dados Iris sobre uma malha 30×30. Podemos observar que uma vizinhança de $v = 5$ (a vizinhança ótima é $v = 5$ para uma precisão de $0,008$) a estimativa de densidade aproxima a densidade estimada utilizando todos os pontos da malha como vizinhança (abordagem original).

4.4 Resultados

Na Figura 4.10 apresentamos uma comparação entre os resultados obtidos com o algoritmo original *HC-Cooperative* e os obtidos com a versão modificada *HC-Enhanced* em termos de tempo de processamento para diferentes tamanhos do conjunto de dados. Para esse e os demais testes apresentados aqui, o programa foi executado em um computador PC com um processador de 1.0 GHz e 1.0 GB de memória RAM, sob Linux Slackware 9.0.

Podemos ver um considerável ganho no tempo de processamento. Esses resultados foram obtidos sem perda da qualidade do agrupamento. Portanto, o algoritmo *HC-Enhanced* é mais interativo e escalável que o algoritmo *HC-Cooperative*, o qual foi o objetivo do presente trabalho de pesquisa.



(a)

(b)

(c)

(d)

Figura 4.9: Estimativa de densidade para o conjunto de dados Iris sobre uma malha 30×30 com vizinhança (a) $v = 2$, (b) $v = 3$, (c) $v = 5$ e (d) $v = 30$.

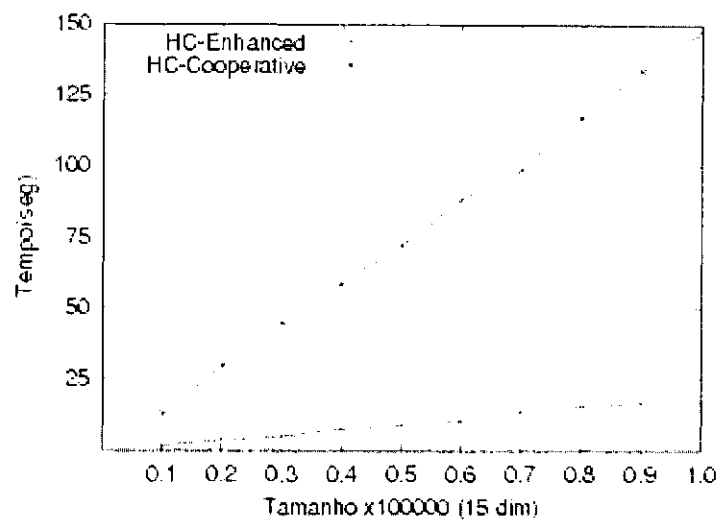


Figura 4.10: Comparação da performance entre o *HC-Cooperative* e o *HC-Enhanced* para conjuntos de dados de diferentes tamanhos e 15 dimensões.

Os algoritmos foram aplicados aos conjuntos Iris e IDH, além de dois conjuntos de dados sintéticos com 1000 instâncias 10-dimensionais e 10000 instâncias 50-dimensionais respectivamente, e ao conjunto de dados Quadrúpedes que serão descritos a seguir[6].

O conjunto de dados Iris é formado por 150 instâncias 4-dimensionais de três classes de plantas classificadas de acordo ao comprimento e largura da sépala, e ao comprimento e largura da pétala. Esse conjunto de dados é usualmente separado pelos algoritmos de agrupamento em dois grupos, um deles formado pelas instâncias da primeira classe de plantas e o outro pelas instâncias das outras duas classes. Usando os algoritmos *HC-Cooperative* e *HC-Enhanced* o primeiro grupo é formado por 92% das instâncias da primeira classe, enquanto o segundo grupo é formado por 85% das instâncias da segunda e terceira classe. É importante notar que não houve falsos positivos nos grupos criados. Esse resultado é usualmente obtido com a maioria dos algoritmos de agrupamento. Além desses dois grupos, os algoritmos HC geram um grupo formado por 86% das instâncias da segunda classe de plantas. Portanto, é possível inferir que existem três grupos de plantas no conjunto de dados Iris e que dois deles são muito similares. Os algoritmos HC são capazes de apresentar tal resultado devido à natureza do processo de criação dos grupos finais baseado em múltiplas projeções. É importante notar que a separação não é exclusiva e uma instância pode pertencer a mais de um grupo.

O tempo de processamento dos algoritmos *HC-Cooperative* e *HC-Enhanced* para o conjunto de dados Iris foi de $0,97$s e $0,14$s respectivamente, para uma malha de 40×40 uti-

[6]Os conjuntos de dados utilizados, com exceção de Sint10d e Sint50d, foram tomados do Repositório de Aprendizado de Máquina da Universidade da Califórnia, Irvine (http://www.ics.uci.edu/~mlearn/MLRepository).

lizando uma vizinhança $v = 6$. As densidades máximas foram $32,648$ para o caso do algoritmo *HC-Cooperative*, e $31,502$ para o caso do algoritmo *HC-Enhanced*.

O conjunto de dados IDH armazena informações de 173 países relativas à qualidade de vida de cada um deles. Esse conjunto de dados possui 9 dimensões além da classificação em termos de qualidade de vida do país, a qual não foi incluída nos testes. Os algoritmos HC separaram esse conjunto de dados em dois grupos. O primeiro grupo foi formado pelas primeiras 25 instâncias na classificação. O segundo grupo foi formado pelas instâncias 48 a 143. As instâncias restantes $(26 - 47)$ foram classificadas como ruído. Da triangulação de Delaunay da projeção Fastmap mostrada na Figura 4.11 podemos notar que esse resultado está de acordo com a distribuição dos dados. Houve um falso positivo no primeiro grupo e 4 e 9 falsos negativos no primeiro e segundo grupos respectivamente.

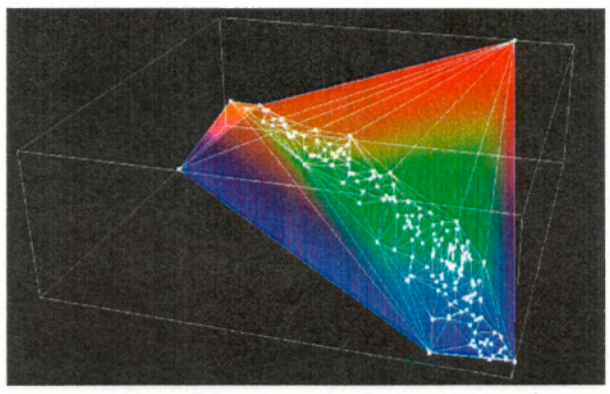
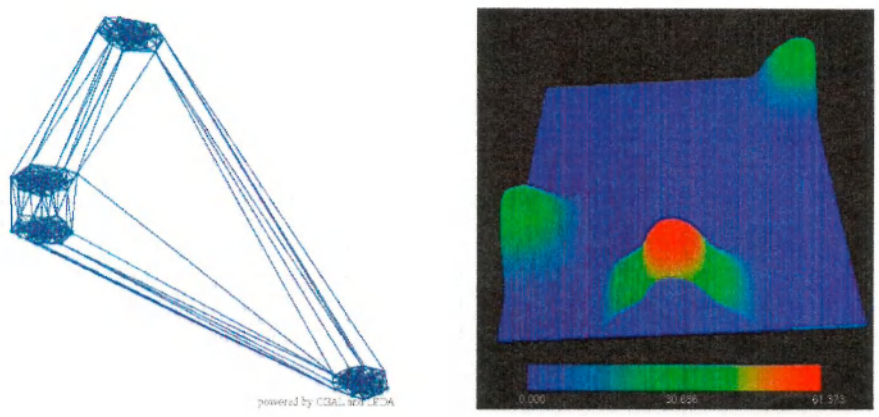


Figura 4.11: Triangulação de Delaunay da projeção Fastmap do conjunto de dados IDH. A altura e a cor mapeiam a classificação do país, sendo que o país melhor classificado é mapeado como azul-baixo e o pior classificado como vermelho-alto.

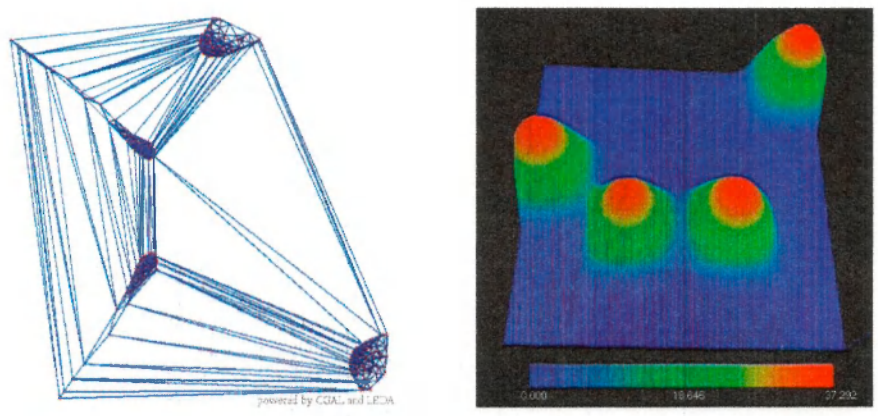
O tempo gasto pelos algoritmos para o conjunto de dados IDH foi de $1,1$s (*HC-Cooperative*) e $0,14$s (*HC-Enhanced*), para uma malha de 40×40 utilizando uma vizinhança $v = 6$. As densidades máximas foram $54,054$ e $51,615$ respectivamente.

O primeiro conjunto de dados sintético, Sint10d, foi criado com 1000 instâncias 10-dimensionais formando quatro grupos, dois deles consideravelmente próximos entre eles, e 20% de ruído. A aplicação dos algoritmos HC a esse conjunto de dados originou os seguintes resultados: quatro grupos foram obtidos, o primeiro com 88% das instâncias originais do primeiro grupo sintético, o segundo com 100% das instâncias do segundo grupo sintético, o terceiro com 88% das instâncias do terceiro grupo sintético, e o quarto com 100% das instâncias do último grupo sintético, não se apresentando falsos positivos em nenhum deles. Os segundo e terceiro grupos apresentaram falsos negativos devido a se encontrarem muito próximos entre eles, o que fez com que as instâncias na fronteira fossem consideradas ruído. Normalmente um único grupo seria gerado a partir desses dois grupos.

Esse resultado foi melhorado utilizando o esquema de melhoramento da projeção apresentado na subseção 4.3.2. O agrupamento resultante apresentou 98% de acerto na média depois de que a projeção foi melhorada usando tal esquema. Podemos observar na Figura 4.12a os quatro grupos com dois deles muito próximos. A Figura 4.12b mostra a projeção melhorada onde tais grupos foram separados.



(a)



(b)

Figura 4.12: Triangulação de Delaunay e estimativa de densidade calculada sobre as projeções (a) original e (b) melhorada de um conjunto de dados sintético com 1000 instâncias 10-dimensionais utilizado para os experimentos realizados.

O tempo de processamento para esse conjunto de dados foi de $6,42$s (*HC-Cooperative*) e $0,93$s (*HC-Enhanced*), para uma malha de 40×40 utilizando uma vizinhança $v = 6$. As densidades máximas foram $45,601$ e $43,660$ respectivamente.

Um segundo conjunto de dados sintético, Sint50d, de 10000 instâncias e 50 dimensões foi criado com três grupos e testado com ambos os algoritmos. A saída obtida apresentou três

grupos formados pelas instâncias dos três grupos originais, tendo os seguintes erros: o primeiro grupo teve 1 falso negativo e 4 falsos positivos. O segundo grupo apresentou 9 falsos negativos e 0 falsos positivos. O terceiro grupo teve 0 falsos negativos e 0 falsos positivos. Isso significa uma precisão de $99,9985\%$, $99,9973\%$ e 100% para os três grupos.

O tempo utilizado pelos algoritmos *HC-Cooperative* e *HC-Enhanced* foi de $72,59$s e $20,26$s respectivamente, para uma malha de 40×40 utilizando uma vizinhança $v = 6$. As densidades máximas foram $46,1914$ e $46,190$ respectivamente.

Finalmente, um conjunto de dados formado por 96487 instâncias 72-dimensionais agrupadas em quatro classes de quadrúpedes (cachorros, gatos, girafa, e cavalos) foi também utilizado para testar os algoritmos. Ambos os algoritmos geraram quatro grupos. O primeiro foi formado por $90,80\%$ das instâncias da primeira classe (cachorros). Das 25521 instâncias incluídas neste grupo, 3480 foram falsos positivos correspondentes à segunda classe (gatos). O segundo grupo foi formado por $70,24\%$ das instâncias pertencentes à segunda classe (gatos). O terceiro grupo foi formado por $67,92\%$ das instâncias correspondentes à terceira classe (cavalos). Finalmente, o quarto grupo foi formado por $33,86\%$ das istâncias pertencentes à terceira classe (girafas). Nenhum dos três últimos grupos apresentaram falsos positivos. $30,875\%$ das instâncias do conjunto de dados original foram consideradas ruído. Esse é classicamente um conjunto de dados de difícil solução de agrupamento.

O tempo utilizado pelo algoritmo *HC-Cooperative* para esse último conjunto de dados foi de $391,21$s, enquanto o tempo usado pelo algoritmo *HC-Enhanced* foi de $201,5$s, para uma malha de 40×40 utilizando uma vizinhança $v = 6$. As densidades máximas foram $56,02$ e $55,98$ respectivamente.

A Tabela 4.8 resume esses resultados, apresentando uma comparação entre ambos os algoritmos tanto em termos de tempo de processamento como da qualidade do agrupamento.

	HC-Cooperative				*HC-Enhanced*			
Conjunto de dados	**Precisão** média	**Tempo**	**Falsos positivos**	**Falsos negativos**	**Precisão**	**Tempo**	**Falsos positivos**	**Falsos negativos**
Iris	$87,667$	$0,97$s	0	19	$87,667$	$0,14$s	0	19
IDH	$91,907$	$1,10$s	1	13	$91,907$	$0,14$s	1	13
Sint10d	$94,0$	$6,42$s	0	60	$98,0$	$0,93$s	0	20
Sint50d	$99,9986$	$72,59$s	4	10	$99,9986$	$20,26$s	4	10
Quadrúpedes	$65,518$	$391,21$s	3480	29670	$65,518$	$201,5$s	3480	29670

Tabela 4.8: Resultados dos algoritmos *HC-Cooperative* e *HC-Enhanced* sobre diferentes conjuntos de dados em termos de precisão do agrupamento e tempo de processamento (Tejada e Minghim, 2003).

Podemos observar que o único conjunto de dados ao qual foi necessário aplicar o processo de melhoramento de projeções foi Sint10d. A decisão de utilizar tal melhoramento deve-se

aos resultados visuais apresentados durante o processamento, nos quais pôde-se observar que existiam grupos muito próximos que poderiam ser separados. Os conjuntos de dados restantes apresentaram grupos bem definidos nas projeções originais ou, no caso do conjunto de dados Quadrúpedes, a métrica utilizada não era apropriada para separar os dois últimos grupos devido a se encontrarem "sobrepostos" no espaço Euclidiano. Pela mesma razão, o processo de melhoria das projeções não separou tais grupos, obtendo-se o mesmo resultado ao obtido com o algoritmo *HC-Cooperative*. Além disso, essa sobreposição de grupos teve efeito no desempenho dos algoritmos em termos do tempo de processamento, considerando que o único caso de melhoria de desempenho abaixo de 70% foi esse.

4.5 Considerações Finais

O presente capítulo apresentou os resultados da pesquisa desenvolvida. Apresentamos as melhorias realizadas ao algoritmo original de Aggarwal, *HC-Cooperative*, em termos de tempo de processamento e qualidade do agrupamento, que originaram a versão denominada *HC-Enhanced*. Posteriormente apresentamos comparações entre ambos os algoritmos em termos de tempo de processamento. Os resultados mostraram uma redução do tempo de processamento maior que 87% na maioria dos casos. Unicamente dois dos conjuntos de dados apresentaram melhorias entre 50% e 70%. Finalmente, resultados sobre a qualidade do agrupamento foram apresentados e se observou um melhor agrupamento por parte do algoritmo *HC-Enhanced* quando os grupos se encontram muito próximos nas projeções utilizadas.

O próximo capítulo será dedicado a apresentar as conclusões obtidas durante o desenvolvimento da pesquisa, assim como os trabalhos futuros a serem realizados nessa linha.

CAPÍTULO

5

Conclusões e Trabalhos Futuros

A mineração visual de dados é uma área de pesquisa que vem ganhando importância nos últimos anos. As vantagens geradas do trabalho cooperativo entre o usuário e o computador fazem com que os algoritmos visuais constituam alternativas poderosas frente a algoritmos de mineração clássicos. Os algoritmos de agrupamento visual têm sido o foco de pesquisas recentes, entre eles a abordagem apresentada por Aggarwal (Aggarwal, 2001).

Neste trabalho propusemos um aprimoramento do algoritmo de Aggarwal *HC-Cooperative*, para reduzir seu tempo de processamento, melhorar a qualidade do agrupamento, e permitir uma melhor interação com o usuário. As vantagens do algoritmo *HC-Cooperative* são significativas para o entendimento do usuário sobre a origem dos dados. Durante o processo de agrupamento o usuário ganha um conhecimento mais profundo sobre os dados e, portanto, sobre o domínio do qual os dados foram gerados.

O algoritmo *HC-Cooperative* lida eficientemente com *outliers* e dados multidimensionais, diminuindo a perda de informação originada pelo processo de redução de dimensionalidade realizando agrupamento em múltiplos subespaços. A informação sobre os grupos gerados em cada subespaço é combinada para criar os grupos finais, permitindo a definição de uma métrica para a consistência dos grupos com as respostas fornecidas pelo usuário. Essa métrica é totalmente independente da aplicação, o que é uma vantagem importante desse algoritmo. Porém, devido ao alto custo computacional dos processos envolvidos, o algoritmo não pode ser aplicado a grandes conjuntos de dados sem incrementar o tempo de resposta atingindo níveis proibitivos.

Como a interatividade é um assunto chave nas técnicas de mineração visual, tais como o algoritmo de Aggarwal, o tempo de resposta deve ser suficientemente baixo para permitir um ciclo contínuo entre usuário e computador.

Os aprimoramentos realizados neste trabalho para diminuir o tempo de processamento, conseguiram adicionar essa característica ao algoritmo *HC-Cooperative* melhorando seu nível de escalabilidade. Dos resultados obtidos podemos observar que o ganho em tempo de processamento é consideravelmente alto. A redução do tempo de processamento foi maior que 87% na maioria dos casos. Unicamente dois dos conjuntos de dados apresentaram melhorias entre 50% e 70%. A qualidade do agrupamento gerado pelo algoritmo original é mantida e melhorou-se a interatividade da abordagem.

Também foi desenvolvido um mecanismo para melhorar os resultados do algoritmo quando dois ou mais grupos estão muito próximos entre eles em relação ao domínio total do subespaço projetado. Isso é realizado deslocando os pontos de dados projetados segundo uma heurística de otimização baseada na relação entre as distâncias nos espaços original e projetado. Isso permitiu uma definição de uma medida genérica de qualidade de projeções geradas com técnicas de redução de dimensionalidade. Nos testes, dois conjuntos de dados (Sint10d e Quadrúpedes) poderiam se beneficiar da melhoria da qualidade da projeção. Em um deles (Sint10d) a abordagem proposta foi eficaz, enquanto no outro (Quadrúpedes) não existe característica nos dados capaz de distinguir os grupos existentes.

Como trabalho futuro podemos mencionar a necessidade de mecanismos para melhorar a interação entre o usuário e o computador para aumentar a interferência do usuário durante o processo de agrupamento, tornando a abordagem aplicável a um número maior de situações. Seria também desejável permitir uma especificação dos grupos em múltiplas visualizações utilizando diferentes técnicas de visualização de informação e de interação de forma que se aumente a precisão da especificação de grupos dirigida pelo usuário.

A inclusão de mecanismos rápidos de projeção é também necessária para obter uma maior redução no tempo de processamento. A aplicação de tais mecanismos deve ser estudada considerando a idéia original de Aggarwal de encontrar os subespaços bidimensionais nos quais as instâncias concentram-se em grupos bem definidos.

Também é importante ressaltar que uma das principais dificuldades encontradas no agrupamento baseado em densidade é o tratamento de dados categóricos ou dados não contínuos. O fato da malha bidimensional que armazena a densidade precisar de posições espaciais para calcular a estimativa de densidade origina um problema importante que deve ser estudado. A extensão do algoritmo *HC-Enhanced* para agrupar dados categóricos é um trabalho que deve ser abordado em pesquisas futuras.

Referências Bibliográficas

Aggarwal, C. (2001). A human-computer cooperative system for effective high dimensional clustering. Em *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, páginas 221–226, CA, USA.

Aggarwal, C. (2002). Towards meaningful high-dimensional nearest neighbor search by human-computer interaction. Em *ICDE'02, International Conference on Data Engineering*, páginas 593–604, San Jose, California. IEEE Press.

Aggarwal, C. e Yu, P. (2000). Finding generalized projected clusters in high dimensional spaces. Em Chen, W., Naughton, J. F., e Bernstein, P. A., editores, *ACM SIGMOD International Conference on Management of Data*, páginas 70–81, Dallas, Texas, USA. ACM.

Ankerst, M. (2001). Visual data mining with pixel-oriented visualization techniques. Em Keim, D. e Eick, S., editores, *ACM SIGKDD Workshop Visual Data Mining*, páginas 85–88, San Francisco, USA. ACM.

Becker, B. (1997). Using mineset for knowledge discovery. *IEEE Computer Graphics and Applications*, 17(4):75–78.

Beckmann, N., Kriegel, H., Schneider, R., e Seeger, B. (1990). The R*-Tree: An efficient and robust access method for points and rectangles. Em *ACM SIGMOD International. Conference. on Management of Data*, páginas 322–331.

Berkhin, P. (2002). Survey of clustering data mining techniques. Technical report, Accrue Software, San Jose, CA.

Beyer, K., Goldstein, J., Ramakrishnan, R., e Shaft, U. (1999). When is nearest neighbor meaningful? *Lecture Notes in Computer Science*, 1540:217–235.

C.-D.Wann e Thomopoulos, S. (1997). A comparative study of self-organizing clustering algorithms DIGNET and ART2. *Neural Networks*, 10(4):737–743.

Card, S. K., Mackinlay, J. D., e Shneiderman, B. (1999). *Readings in Information Visualization: Using Vision to Think*. Morgan Kaufmann Publishers.

Collier, K., Medidi, M., e Sautter, D. (2002). *Visualization in the Knowledge Discovery Process.*, chapter 8, páginas 121–122. Em Information in Data Mining and Knowledge Discovery. Ed. U. Fayyad, G. Grinstein, e A. Wierse. Morgan-Kaufmann.

Cormen, T. H., Leiserson, C. E., e Rivest, R. L. (1990). *Introduction to Algorithms*. MIT Press.

DuMouchel, W., Volinsky, C., Johnson, T., Cortes, C., e Pregibon, D. (1999). Squashing flat files flatter. Em *5th International Conference on Knowledge Discovery and Data Mining (KDD-99)*, páginas 6–15, San Diego, CA. ACM Press.

Edelsbrunner, H. (2001). *Geometry and Topology for Mesh Generation*. Cambridge Monographs on Applied and Computational Mathematics. Cambridge Univ. Press.

Ester, M., Kriegel, H.-P., Sander, J., e Xu, X. (1996). A density-based algorithm for discovering clusters in large spatial databases with noise. Em Simoudis, E., Han, J., e Fayyad, U., editores, *Second International Conference on Knowledge Discovery and Data Mining*, páginas 226–331, Portland, Oregon. AAAI Press.

Ester, M., Kriegel, H.-P., e Xu, X. (1997). Density-connected sets and their application for trend detection in spatial databases. Em *3rd International Conference on Knowledge Discovery and Data Mining (KDD-97)*, páginas 10–15, California, USA. AAAI Press.

Faloutsos, C. e Lin, K. (1995). FastMap: A fast algorithm for indexing, data-mining and visualization of traditional and multimedia databases. Em Carey, M. J. e Schneider, D. A., editores, *ACM SIGMOD'95 International Conference on Management of Data*, páginas 163–174, San Jose, California. ACM.

Fayyad, U. M. (1998). Mining databases: Towards algorithms for knowledge discovery. *Data Engineering Bulletin*, 21(1):39–48.

Fayyad, U. M., Piatetsky-Shapiro, G., e Smyth, P. (1996). Knowledge discovery and data mining: Towards a unifying framework. Em *Knowledge Discovery and Data Mining*, páginas 82–88.

Freitas, C. M. D. S., Luzzardi, P., Cava, R., Winckler, M., Pimenta, M., e Nedel, L. (2002). Evaluating usability of information visualization techniques. Em *V Simposio sobre Fatores Humanos em Sistemas Computacionais - IHC'02*, Fortaleza, Brazil.

Ganti, V., Ramakrishnan, M., Gehrke, J., Powell, A., e French, J. (1998). Clustering large datasets in arbitrary metric spaces. Technical report, Department of Computer Science, University of Wisconsin at Madison, Madison, Wisconsin, USA.

Ganti, V., Ramakrishnan, R., Gehrke, J., Powell, A., e French, J. (1999). Clustering large datasets in arbitrary metric spaces. Em *International Conference on Data Engineering ICDE'99*, páginas 502–511, Sydney, Australia.

Golub, G. H. e Loan, C. V. (1991). *Matrix Computation*. Then John Hopkins University Press.

Grinstein, G., Trutschl, M., e Cvek, U. (2001). High-dimensional visualizations. Em *7th Data Mining Conference KDD Workshop 2001*, páginas 7–19, San Francisco, CA.

Guha, S., Rastogi, R., e Shim, K. (1998). CURE: An efficient clustering algorithm for large databases. Em Haas, L. M. e Tiwary, A., editores, *ACM SIGMOD International Conference on Management of Data (SIGMOD'98)*, páginas 73–84, Seattle, Washington, USA. ACM Press.

Hinneburg, A. e Keim, D. (1998). An efficient approach to clustering in large multimedia databases with noise. Em *4rd Int. Conf. on Knowledge Discovery and Data Mining (KDD-98)*, páginas 58–65, New York City, New York, USA. AAAI Press.

Hinneburg, A., Keim, D., e Wawryniuk, M. (1999). HD-eye: Visual mining of high himensional hata. *IEEE Computer Graphics and Applications*, 19(5):22–31.

Inselberg, A. (1985). The plane with parallel coordinates. *Special Issue on Computational Geometry: The Visual Computer*, 1:69–91.

Inselberg, A. e Dimsdale, B. (1987). Parallel coordinates for visualizing multidimensional geometry. Em *Computer Graphics International '87*, Tokyo, Japan.

Jolliffe, I. T. (1986). *Principal Component Analysis*. Springer-Verlag.

Karypis, G., Han, E., e Kumar, V. (1999). CHAMELEON: A hierarchical clustering algorithm using dynamic modeling. *COMPUTER*, 32:68–75.

Keim, D. (2002). Information visualization and visual data mining. *IEEE Transactions on Visualization and Computer Graphics*, 8(1):1–8.

Keim, D. A. e Ankerst, M. (2001). Visual data mining and exploration of large databases. Em *Tutorial at PKDD'2001*, Freiburg, Germany.

Knapp, B. (1998). Covariance matrix. *http://www.engr.sjsu.edu/knapp/HCIRODPR-/PR_Mahal/cov_mat.htm*, [accessed in 17/09/2002].

Kreuseler, M. e Schumann, H. (2002). A flexible approach for visual data mining. *IEEE Transactions on Visualization and Computer Graphics*, 8(1):39–51.

M.-S. Chen, J. Han, P. S. Y. (1996). Data mining: An overview from a database perspective. *IEEE Transactions on Knowledge and Data Enginnering*, 8(6):866–883.

Minghim, R. e de Oliveira, M. (1997). Uma introdução à visualização computacional. Em *XVI JAI - Jornada de Atualização em Informática, XVII Congresso da SBC*, páginas 85–131, Brasília, Brasil.

Oliveira M.C. e Levkowitz H. (1997). From visual data exploration to visual data mining: a survey. *IEEE Transactions on Visualization and Computer Graphics*, 9(3):378–394.

Pagel, B.-U., Korn, F., e Faloutsos, C. (2000). Deflating the dimensionality curse using multiple fractal dimensions. Em *International Conference on Data Engineering - ICDE'00*, páginas 589–598, San Diego-CA, USA.

Partridge, M. e Calvo, R. (1998). Fast dimensionality reduction and simple pca. *Intelligent Data Analysis*, 2(3).

Piatetsky-Shapiro, G. e Frawley, W. J. (1991). *Knowledge Discovery in Databases*. AAAI/MIT Press.

Press, W., Teukolsky, S., Vetterling, W., e Flannery, B. (1992). *Numerical Recipes in C: The Art of Scientific Computing*. Cambridge Univ. Press.

Rao, Y. N. e Principe, J. C. (2000). A fast, on-line algorithm for PCA and its convergence characteristics. Em *IEEE Workshop on Neural Networks for Signal Processing*, páginas 299–308. IEEE Press.

Ribarsky, W. (1999). Discovery visualization and visual data mining. Technical report, College of Computing - Georgia Tech.

Ribarsky, W., Katz, J., Jiang, F., e Holland, A. (1999). Discovery visualization using fast clustering. *IEEE Computer Graphics and Applications*, 19(5):32–39.

Sander, J., Ester, M., Kriegel, H., e Xu, X. (1998). Density-based clustering in spatial databases: The algorithm GDBSCAN and its applications. *Data Mining and Knowledge Discovery*, 2(2):169–194.

Santos-Filho, R. F., Traina, A., Jr, C. T., e Faloutsos, C. (2001). Similarity search without tears: The omni-family of all-purpose access methods. Em *17th IEEE International Conference on Data Engineering (ICDE)*, páginas 623–630, Heidelberg, Germany.

Silverman, B. W. (1986). *Density estimation for statistics and data analysis*. Chapman and Hall.

Strang, G. (1980). *Linear Algebra and its Applications*. Academic Press.

Tejada, E. e Minghim, R. (2003). Improved visual clustering of large multidimensional data sets. *Submetido ao ACM SIGKDD Explorations Journal*.

Tejada, E., Minghim, R., e Nonato, L. G. (2003). On improved projection techniques to support visual exploration of multi-dimensional data sets. *Submetido ao Information Visualization Journal, Special Issue on Coordinated & Multiple Views in Exploratory Visualization*, 2(4).

Thomopoulos, S., Bougoulias, D., e C.-D.Wann (1995). DIGNET: An unsupervisedlearning clustering algorithm for clustering and data fusion. *IEEE Transactions on Aerospace and Electronic Systems*, 31(1):21–38.

Traina, C., Traina, A., Wu, L., e Faloutsos, C. (2000). Fast feature selection using fractal dimension. Em *Simpósio Brasileiro de Banco de Dados - SBBD'00*, João Pessoa-PA, Brazil.

Witten, I. e Frank, E. (1999). *Data Mining: Practical Machine Learning Tools and Techniques with Java Implementations*. Morgan Kaufmann.

Wong, P. C. (1999). Visual data mining. *IEEE Computer Graphics and Applications*, 19(5):20–21.

Xu, X., Ester, M., Kriegel, H., e Sander, J. (1998). A distribution-based clustering algorithm for mining in large spatial databases. Em *14th Int. Conf. on Data Engineering (ICDE'98)*, páginas 324–331, Orlando, FL.

Zhang, T., Ramakrishnan, R., e Livny, M. (1996). BIRCH: an efficient data clustering method for very large databases. Em *ACM SIGMOD'96 International Conference on Management of Data*, páginas 103–114, New York, USA. ACM.

Zhang, T., Ramakrishnan, R., e Livny, M. (1997). BIRCH: A new data clustering algorithm and its applications. *Data Mining and Knowledge Discovery*, 1(2):141–182.