CCICLO:
Componente para cruzamento e
integração de objetos classificados.

LUCIANO VIEIRA DE ARAÚJO

DISSERTAÇÃO APRESENTADA AO
INSTITUTO DE MATEMÁTICA E ESTATÍSTICA DA
UNIVERSIDADE DE SÃO PAULO
PARA OBTENÇÃO DO GRAU DE MESTRE EM
CIÊNCIA DA COMPUTAÇÃO

Área de Concentração: Banco de Dados
Orientador: Prof. Dr. João Eduardo Ferreira

- São Paulo - 2002 -

CCICLO: Componente para cruzamento e integração de objetos classificados

Este exemplar corresponde à redação final da dissertação devidamente corrigida e defendida por Luciano Vieira de Araújo e aprovada pela comissão julgadora.

São Paulo, 29 de julho de 2002

Banca examinadora:

Prof. Dr. João Eduardo Ferreira (orientador)	IME-USP
Profa. Dra. Ana Cristina Vieira de Mello	IME-USP
Prof. Dr. Caetano Traina Júnior	ICMC-USP

Dedico este trabalho aos meus pais.

"Melhor é o fim das coisas do que o seu princípio."

Salomão, sábio e rei de Israel - Livro de Eclesiastes - capítulo 7 - versículo 8.

Aos meus irmãos Sérgio e Livia pelo apoio a cada momento.

Aos amigos de Goiânia pelas celebrações restauradoras a cada visita à terra natal.

Índice

Resumo	xi
Abstract	xiv
1 Introdução	1
2 Trabalhos relacionados	7
2.1 Cache	8
2.1.1 Recuperação de dados no cache	8
2.1.2 Substituição de dados no cache	10
2.2 OLAP	17
2.2.1 Cubos	20
2.3 Visões materializadas	21
2.4 Estruturas de índices para sistemas OLAP	22
2.4.1 Índices de junção	22
2.4.2 Índices bitmap	24
2.5 Soluções para a memória principal	26
3 Descoberta de conhecimento por meio de classificadores	27

3.1	Classificação de dados	30
3.1.1	Processo de classificação de objetos	31
3.2	Dados classificados	34
3.2.1	Estrutura para representação de classificações	34
3.2.2	Formato de dados para utilização do CCICLO	35
4	Cruzamento de objetos classificados	39
4.1	Características do CCICLO	42
4.1.1	Uso da memória principal	42
4.1.2	Representação de objetos classificados	43
5	Estrutura de dados para cruzamento de objetos	45
5.1	Objetos complexos	45
5.2	Representação multidimensional do vetor de características	47
5.2.1	R-Tree	49
6	Operadores para cruzamento de objetos	55
6.1	Operadores simples	56
6.1.1	Operador seleção (σ)	56
6.1.2	Operador projeção (π)	58
6.1.3	Operador negação ($\neg$)	58
6.1.4	Operador diferença entre objetos classificados (-)	59
6.2	Operadores compostos	60
6.2.1	Operador união composta ($\cup \beta$)	61
6.2.2	Operador interseção composta ($\cap \beta$)	62
6.2.3	Operador junção composta ($\bowtie \beta$)	64
6.3	Hierarquia de execução dos operadores	65

7	Estudo de caso	67
7.1	Classificadores para o banco de dados de HIV	71
7.2	Representação dos dados em memória	73
7.2.1	Representação em disco do banco de HIV	75
7.2.2	Representação no CCICLO do banco de HIV	77
7.3	Seleção multidimensional de classificadores para cruzamento	80
7.4	Uso dos operadores	82
7.4.1	Operador seleção (σ)	83
7.4.2	Operador projeção (π)	85
7.4.3	Operador negação ($\neg$)	87
7.4.4	Operador diferença entre objetos classificados (-)	89
7.4.5	Operador união composta ($\cup \beta$)	90
7.4.6	Operador interseção composta ($\cap \beta$)	93
7.4.7	Operador junção composta ($\bowtie \beta$)	95
7.4.8	Inclusão dos resultados obtidos na estrutura em memória	96
8	Conclusão	99
8.1	Contribuições do trabalho	99
8.1.1	Redefinição dos operadores relacionais	99
8.1.2	Integração de conhecimento especializado	100
8.1.3	Representação em estrutura de dados do contexto de classificação	101
8.1.4	Representação multidimensional das características dos classificadores	102
8.2	Aplicação prática do CCICLO	102
8.3	Futuras pesquisas	103
8.3.1	Implementação do CCICLO	103
8.3.2	Interface gráfica para o CCICLO	104
8.3.3	Adaptação do CCICLO para trabalhar com incertezas nas classificações	105

Lista de Figuras

2.1	Diagrama das consultas	9
2.2	Exemplo de afinidade mista – Temporal e regional.	13
2.3	Pseudo-código do algoritmo LRU de dois níveis.	17
2.4	Pré-cálculo das agregações de um cubo	21
2.5	Relação entre instâncias da tabela-fato Venda e as tabelas-dimensão Localização e Produto.	23
2.6	Índice de junção.	24
3.1	Processo de classificação de dados.	32
3.2	Estrutura de dados para representação dos objetos classificados.	37
5.1	Objeto complexo – classificador.	46
5.2	R-tree.	51
5.3	Representação das dimensões avaliadas pelos classificadores.	52
7.1	Objeto complexo - classificador.	74
7.2	Cálculo de valores.	78
7.3	Conjunto de classificadores usados no estudo de caso.	82
7.4	Seleção de pacientes com falha terapêutica.	84

7.5	Seleção de pacientes com tratamento efetivo.	84
7.6	Seleção usando os atributos Id e R.	85
7.7	Projeção dos identificadores dos objetos de C_3	86
7.8	Projeção dos identificadores com valor 1 para o rótulo.	87
7.9	Negação do classificador C_2	88
7.10	Exemplo do operador diferença entre classificadores.	90
7.11	Exemplo de união composta de classificadores.	92
7.12	Exemplo do operador interseção.	94
7.13	Exemplo de junção composta em C_3	96
8.1	Ambiente integrado para descoberta de conhecimento em biologia.	104

Lista de Tabelas

2.1	Dados sobre o conjunto de dados de cada consulta.	16
2.2	Dados sobre o agrupamento semântico.	16
2.3	Tabela de referência.	24
2.4	Índice bitmap para o atributo cidade.	25
2.5	Índice bitmap para o atributo estado.	25
6.1	Funcionamento dos operadores de conjunto.	60
6.2	Funcionamento dos operadores lógicos.	61
7.1	Número de ocorrências para os dados variáveis do paciente.	76
7.2	Comparação entre o espaço ocupado pelas tabelas do primeiro e segundo grupos. .	76
7.3	Comparação entre o espaço estimado e o espaço em uso.	77
7.4	Consumo de espaço dos tipos de dados em Java 1.3.	78
7.5	Demonstrativo de espaço.	79
7.6	Demonstrativo de espaço.	80
7.7	Funcionamento dos operadores lógicos.	91

Resumo

A classificação de dados é uma das tarefas importantes realizadas pela *mineração de dados*. Entretanto, tal classificação não considera a necessidade da comparação automática entre diversas classificações de dados.

Este trabalho apresenta um componente para descobrir as relações entre objetos classificados. Esse componente é uma forma de apoio ao processo de descoberta de conhecimento. Para tanto, tal componente oferece uma estrutura de dados para representação, em memória, dos dados gerados pelo processo de classificação, que inclui dados dos classificadores e dos objetos analisados. Além disso, para a manipulação da referida estrutura de dados em memória foi estendido um conjunto de operadores da álgebra relacional, para otimizar o cruzamento dos objetos classificados.

Abstract

The classification of data is one of the important tasks carried through data mining operation. However, it does not consider the need for automatic comparison between several classifications of data.

This work presents a component to analyse the discovered the relations between classified objects. This component supports the process of knowledge discovery. In such a way, it offers a data structure for main-memory representation, of the data on the classification process, that includes: data of the classifier and data of analyzed objects. Moreover, aiming the manipulation of the related data structure in main-memory, a set of operators of relational algebra was extended to optimize the comparison of the classified objects.

Introdução

O volume de dados armazenado em bancos de dados cresce intensamente movido pela popularização da internet e de dispositivos eletrônicos capazes de gerar e armazenar dados. Com eles é possível reunir um grande volume de dados a cada dia, ou até mesmo a cada hora, abrangendo os mais diversos assuntos dos mais diferentes lugares do mundo ou fora dele. A reunião de tão grande, variado e atualizado volume de dados tem despertado o interesse por analisá-los. Afinal de contas, a geração de tantos dados, em formato digital, é algo inédito na história e com tendência de expansão. Porém, a análise de um conjunto de dados não é uma tarefa trivial, e sua complexidade tende a aumentar de acordo com o volume e a variedade dos dados envolvidos. Nesse sentido, os métodos tradicionais de análise de dados, alicerçados em relatórios e gráficos preconcebidos, têm mostrado pouca eficiência para análise de realidades variadas e muitas vezes pouco conhecidas, principalmente considerando-se a limitação humana para avaliação simultânea de grande quantidade de informações. Esse cenário tem sido o motivador de estudos de técnicas para a descoberta de novos conhecimentos por meio da análise de grandes volumes de dados. Entende-se por novos conhecimentos, informações que não estejam explicitamente representadas nos modelos analisados, como relacionamentos entre grupos de dados ainda não percebidos ou não comprovados, padrões de comportamentos, assim como a influência da variação de um conjunto de dados sobre os demais grupos analisados. De modo geral, novos conhecimentos englobam todas as informações úteis inferidas a partir dos dados analisados, e não de modelos previamente estabelecidos.

O desenvolvimento de métodos para o descobrimento de informações em grandes volumes de dados apresenta-se como uma tarefa multidisciplinar, pois envolve tanto especialistas das diversas áreas da ciência da computação, como especialistas da área de conhecimento estudada. Geralmente, as tarefas para análise de dados envolvem a captação, integração e armazenagem dos dados a serem analisadas, o desenvolvimento de modelos para representação e processamento automático do conhecimento e as ferramentas para geração e avaliação dos resultados de forma compreensível e reproduzível. Essa linha de pesquisa é conhecida como *Descoberta de Conhecimento em Bancos de Dados* [HK01] (*KDD - Knowledge Discovery in Database*) ou *Mineração de Dados (Data Mining)* e tem apresentado um grande crescimento em diversas áreas de pesquisa – a ciência da computação, matemática e estatística, responsáveis pela geração dos recursos necessários para obtenção, integração e análise dos dados; a biologia e a física, entre outras, responsáveis pela geração dos dados e pelo fornecimento de conhecimento para modelagem e avaliação dos resultados obtidos. Além disso, o *KDD* tem se tornado popular nas áreas empresariais, onde dados de empresas e indústrias têm alimentado grandes bancos de dados analíticos, chamados de *Data Warehouse* [CD97b], para posterior análise por sistemas computacionais providos de recursos para análises estatísticas.

As pesquisas em genoma também apresentam um bom campo para utilização das técnicas de *KDD*. Os dados gerados pelos diversos projetos genoma apresentam uma grande quantidade de padrões com valores significativos e que instigam o seu estudo, como os conjuntos de genes que codificam proteínas, os genes responsáveis por determinada doença ou a possibilidade de cruzamento de dados genéticos com informações de diversas outras áreas relacionadas ao organismo estudado. Por exemplo: os dados obtidos por estudos do genoma humano podem ser confrontados com dados clínicos, sociais, epidemiológicos ou terapêuticos do paciente. Os dados sobre estudos do genoma de um vírus podem ser cotejados com informações sobre os indivíduos contaminados pelo vírus e com dados de outros vírus encontrados.

Os exemplos citados são apenas uma pequena amostra das possibilidades e contribuições oferecidas pelos cruzamentos de dados. O interesse pela análise de dados cresce com o volume e variedade de dados acumulados. Contudo, para tornar as análises e os cruzamentos entre os dados

possíveis, é necessário superar alguns obstáculos:

- A demora no processamento de grandes volumes de dados. Normalmente o processamento de grandes volumes de dados é lento, podendo levar dias, semanas ou até meses, dificultando a possibilidade de avaliação de diversas hipóteses ou a utilização de diferentes métodos de análises. Um processo de análise de dados com intervalos de tempo muito longo pode ser algo indesejado por quebrar a continuidade do processo. O cruzamento de grandes volumes de dados tende a ser um processo muito lento por causa da complexidade do processamento realizado e da necessidade de sucessivas trocas de dados entre a memória principal e as memórias secundárias mais lentas. As trocas de dados são provocadas pela diferença de capacidade de armazenamento entre a memória principal e a memória secundária. Como normalmente a memória secundária é muitas vezes maior que a memória principal, o processamento de seu conteúdo exige a troca de dados entre elas. Assim, os dados colocados na memória são processados e depois são devolvidos para a memória secundária para liberar espaço para novos dados e podem ser recuperados novamente se for necessário utilizá-los. A quantidade de trocas de dados pode aumentar caso o processamento gere um extenso volume de dados durante a sua execução, como acontece com operações de *produto cartesiano*, em que cada elemento de um grupo é relacionado com todos os elementos do outro grupo, representando um aumento significativo de dados.
- A numerosa quantidade de resultados espúrios geradas pelas técnicas automáticas para *mineração de dados*. Em alguns casos, tais técnicas buscam avaliar a totalidade de combinações entre os dados que levam à geração de uma vasta quantidade de resultados, porém apenas uma pequena parte deles apresenta valores significativos.
- A complexidade para integração de informações sobre diversas áreas de conhecimento. Modelar uma realidade para ser representada por um conjunto de dados nem sempre é uma tarefa simples e, na maioria das vezes, exige envolvimento de especialistas da área capazes de

descrevê-la com precisão. Expandir uma modelagem para representação de uma realidade envolvendo diversas áreas de conhecimento, sendo algumas delas pouco conhecidas, com o objetivo de cruzar seus dados, torna-se uma tarefa um pouco mais complexa, exigindo mais conhecimento das pessoas envolvidas ou a inclusão de um número maior de profissionais. Isso nem sempre é possível ou desejável, em razão da demanda de recursos e dificuldades para a criação e integração da equipe.

Os obstáculos apresentados apontam para a necessidade do surgimento de trabalhos que apresentem alternativas tanto para melhorar o desempenho dos processos de análises de dados, como para facilitar o aprendizado e a integração do conhecimento sobre diferentes áreas. Este trabalho pretende contribuir para a superação desses obstáculos, apresentando um componente para o cruzamento de objetos classificados na memória principal do computador como apoio à estratégia para o descobrimento de conhecimento em banco de dados. Esse componente é chamado de **CCICLO** (*Component for Crossing and Integration of Classified Objects* ou *Componente para o Cruzamento e Integração de Objetos Classificados*).

A utilização de objetos classificados como base para o cruzamento de dados é justificada por ser a classificação de objetos uma das principais tarefas dos processos de *Mineração de Dados*, em que objetos com características comuns são identificados e passam a formar classes de objetos, permitindo assim a exploração das peculiaridades de seus membros. Uma vez analisados os objetos e criadas as classes, o cruzamento de objetos classificados apresenta o atrativo de oferecer uma nova perspectiva para análise de classes de objetos, aproveitando o processamento existente e simplificando tanto o processo de integração dos dados, como a exploração das relações entre as classes.

O **CCICLO** apresenta um ambiente preparado para trabalhar com objetos classificados abrangendo a reutilização dos classificadores e de suas classificações, a simplificação do processo de integração de dados, a representação multidimensional das informações sobre os classificadores, a estrutura de dados para representação resumida dos objetos em memória e os operadores relacionais

adaptados para realização dos cruzamentos dos dados na memória principal do computador.

Trabalhos relacionados

O uso da memória principal do computador possui a vantagem de aumentar a velocidade de processamento dos dados, em virtude do menor tempo gasto para acessar o dado armazenado na memória, em relação ao tempo gasto para acessar o dado armazenado em disco. Portanto, aplicações que usam a memória para armazenar os dados necessários ao seu processamento tendem a ser mais rápidas que as aplicações que solicitam dados diretamente ao disco.

O *CCICLO* apresenta uma proposta de cruzamento de objetos na memória principal do computador, com o objetivo de aproveitar o recurso oferecido pela representação de dados na memória para viabilizar a análise de grande volume de dados. Para tanto, os estudos que levaram ao seu desenvolvimento avaliaram linhas de pesquisas que usam a memória principal do computador para melhorar o desempenho das aplicações, como as *estruturas de cache*, as *soluções para a memória principal*¹ e os sistemas *OLAP* (On-line Analytical Process/Processo Analítico Imediato)² que apresentam estruturas de índices diferenciadas para realizar consultas e cruzamento de dados. A escolha das linhas de pesquisas baseadas em otimização em memória, como referência para a definição e desenvolvimento do *CCICLO* objetiva atender, da forma mais eficiente possível, às necessidades de cruzamento de dados.

¹Termo traduzido do inglês *main memory applications*.

²A sigla em inglês OLAP será utilizada no restante do trabalho por ser assim utilizada em publicações em português.

2.1 Cache

O cache [DFJ⁺96] é uma aplicação bastante conhecida e utilizada para melhorar o desempenho de aplicações, pelo uso da memória do computador. Ele tem como objetivo principal manter em memória os dados com maior probabilidade de reutilização, evitando acessos ao disco para recuperar dados já processados.

Por causa das limitações de espaço na memória principal, às vezes não é possível conservar em memória todos os dados necessários para um determinado processamento. Nesse caso, surge a necessidade de liberação de espaço em memória, por meio do envio de alguns dados para o disco para que outros possam ser inseridos e processados. Como nem sempre a memória necessita ser completamente esvaziada, há a possibilidade de manter parte dos dados, na expectativa de novas utilizações dos mesmos. Caso os dados mantidos em memória sejam novamente necessários, diminui-se o tempo de recuperação deles, o tráfego na rede e a carga no servidor.

O cache gerencia a memória do computador realizando duas importantes tarefas: a recuperação e a substituição dos dados em memória.

2.1.1 Recuperação de dados no cache

O cache não somente deve manter os dados em memória, como também precisa ser capaz de usá-los para atender às solicitações das aplicações. O processo de identificação dos dados armazenados na memória e seu envio para a aplicação solicitante é chamado de *recuperação de dados* [DFJ⁺96].

Quando uma consulta é feita por uma aplicação, o cache verifica se ele possui os dados necessários para respondê-la. Caso não os possua, a consulta é repassada para o servidor que providencia a recuperação dos dados. Se todos os dados estiverem no cache, a solicitação da aplicação é respondida sem contactar o servidor. Esses dois casos são considerados simples e são atendidos pela maioria dos caches.

A recuperação de dados torna-se mais complexa quando o cache possui apenas parte dos dados solicitados. Nesse caso, o melhor desempenho será obtido se os dados em cache forem utilizados

e somente os faltantes forem solicitados ao servidor. Para tanto, deve ser realizado um processo chamado de *divisão de consultas* [DFJ⁺96], no qual a consulta original e o conteúdo do cache são analisados para a criação de duas consultas: a *consulta exploratória* e a *consulta restante*.

- *Consulta exploratória*³ – consulta responsável pela recuperação dos dados armazenados no cache.
- *Consulta restante*⁴ – consulta a ser enviada ao servidor para solicitar somente os dados não encontrados no cache.

Figura 2.1: Diagrama das consultas

As consultas exploratória e restante devem gerar resultados disjuntos, cuja união responda completamente a consulta original. A figura 2.1 extraída de [DFJ⁺96] apresenta o funcionamento do procedimento de divisão de consultas. Nesse exemplo, são utilizados a relação *Empregados*(id, nome, renda, idade) e um cache cujo conteúdo é representado pelo predicado:

$$(\text{renda} < 100.000 \text{ ou } (\text{idade} > 25 \text{ e } \text{idade} \leq 28))$$

³Tradução do termo em inglês - probe query.

⁴Tradução do termo em inglês - remainder query.

A consulta original enviada para o cache pela aplicação possui o predicado:

$$(\text{renda} > 50.000 \text{ e idade} \leq 30)$$

Baseado no predicado dos dados contidos no cache e no predicado da consulta, o processo de divisão de consultas é realizado, sendo obtidas as consultas exploratória e restante.

Predicado da consulta exploratória:

$$((\text{renda} > 50.000 \text{ e renda} < 100.000 \text{ e idade} \leq 30) \text{ ou} \\ (\text{renda} > 100.000 \text{ e idade} > 25 \text{ e idade} \leq 28))$$

Predicado da consulta restante:

$$((\text{renda} \geq 100.000 \text{ e idade} < 25) \text{ ou} \\ (\text{renda} \geq 100.000 \text{ e idade} \geq 28 \text{ e idade} \geq 30))$$

Conforme o exemplo anterior, a divisão de consultas não é um processo trivial e pode ser inviável nos casos em que seu processamento demanda mais tempo que o necessário para recuperar todos os dados diretamente do servidor. Porém, ela é importante para o bom funcionamento do cache, pois aumenta a utilização dos dados, evitando solicitá-los novamente ao servidor, uma vez que já estão na memória. Trabalhos como [GG99, GG97, GG96, CKM⁺00, KB96, KMPS94, FLS98, CS99, CKPS95] têm estudado a divisão de consultas e apresentam propostas para sua realização e identificação dos casos que as tornam inviável.

2.1.2 Substituição de dados no cache

A substituição dos dados contidos no cache não é uma tarefa simples, porque engloba a escolha de quais dados devem permanecer em memória e quais devem ser retirados. Para decidir sobre a substituição de dados, o cache se apóia em métricas capazes de classificar o seu conteúdo de acordo com o potencial de reutilização dos dados. Para tanto, o histórico de acesso ao dado,

o relacionamento entre os dados e o custo para obtenção do dado são avaliados para definir o potencial de reutilização dos dados [DFJ⁺96].

Os dados com menor potencial de reutilização, conhecidos como "região fria" do cache são os primeiros a serem substituídos quando existe a necessidade de liberação de espaço na memória. Já os dados com maior potencial de reutilização são conhecidos como "região quente". Eles só são substituídos quando todos os dados da região fria foram removidos e ainda existe a necessidade de liberação de mais espaço na memória.

O potencial de reutilização de um dado varia dinamicamente de acordo com a inclusão e remoção de dados no cache, exigindo das métricas de avaliação dos dados a capacidade de adequação às mudanças.

As métricas buscam identificar algum tipo de afinidade entre os dados para sugerir a necessidade ou não de mantê-los na memória. A seguir são apresentadas algumas afinidades utilizadas para analisar os dados.

Afinidade temporal

A afinidade temporal [DFJ⁺96] supõe que o potencial de reutilização de um dado está ligado a uma característica temporal atribuída a ele, ou seja, saber quando um dado foi colocado no cache ou quando foi a sua última utilização pode indicar a necessidade ou não de mantê-lo em memória. Para realizar o controle das informações temporais, o cache atribui rótulos de tempo aos dados para registrar quando eles são inseridos na memória ou quando eles são utilizados. Os rótulos de tempo podem ser analisados por diferentes regras:

- *LRU (Less Recently Used - Menos recentemente usado)* – Essa abordagem considera com menor potencial de reutilização os dados menos recentemente usados. Isto é, os dados sem utilização recente são os primeiros a serem substituídos para liberação de espaço em memória.
- *MRU (Most Recently Used - Mais recentemente usado)* – Essa regra considera os dados com rótulo de utilização mais recente como os menos necessários, ou seja, quanto mais recente

foi a utilização de um dado, menor é a chance de sua reutilização. Assim, os dados mais recentemente usados são os primeiros a serem substituídos.

- *FIFO (First In First Out - Primeiro a entrar, primeiro a sair)* – Essa abordagem simula o funcionamento de uma fila. Isto significa que os primeiros dados inseridos na memória são os primeiros a serem substituídos.

Afinidade de frequência de acesso

A afinidade de frequência de acesso [GG99] infere o potencial de reutilização de um dado pela avaliação da quantidade de acessos recebidos por ele. Nesse caso, cada acesso realizado ao dado é contabilizado pelo cache e a análise dessa informação é feita segundo as seguintes:

- *LFU (Less Frequently Used - Menos freqüentemente usado)* – Essa regra qualifica como dados com o menor potencial de reutilização, os dados com a menor quantidade de acesso, ou seja, a abordagem considera que quanto menos usado for um dado, menor a chance de sua reutilização. Portanto, tais dados devem ser os primeiros a serem substituídos para a liberação de espaço em memória.
- *MFU (Most Frequently Used - Mais freqüentemente usado)* – Nessa abordagem os dados com a maior quantidade de acessos são qualificados como os dados com o menor potencial de reutilização. Assim, o MFU considera que um dado que já foi muito utilizado tem a menor chance de ser usado novamente, devendo ser o primeiro a ser substituído.

Afinidade regional

A afinidade regional [RD99] faz uso de métricas de distância para definir agrupamentos entre os dados do cache e a partir deles inferir o potencial de reutilização dos dados. Assim, o gerenciamento do cache é feito baseado no pressuposto de que dados fora dos agrupamentos têm a menor possibilidade de reutilização e, portanto, devem ser os primeiros a serem substituídos.

Figura 2.2: Exemplo de afinidade mista – Temporal e regional.

A noção de proximidade entre dados varia de acordo com a característica analisada. Podem ser considerados próximos os solicitados pela mesma aplicação, os provenientes da mesma relação ou fonte de dados ou os que apresentam valores comuns ou com pouca variação em determinados atributos.

Afinidade mista

A substituição de dados [RD99] por afinidade mista procura aumentar a reutilização dos dados por meio da combinação de abordagens. Com isso, mais parâmetros podem ser avaliados para decidir sobre a substituição dos dados. O uso de múltiplos parâmetros permite que um dado, classificado como parte da região fria do cache por um dos parâmetros, possa ser considerado como parte da região quente pela análise de outro parâmetro. A utilização de um conjunto de afinidades pode tornar a substituição de dados mais precisa, evitando a substituição de dados com alto potencial de reutilização.

A figura 2.2 apresenta um exemplo do funcionamento da afinidade mista. Nela são representados conjuntos de dados utilizados para atender às consultas. Cada conjunto de dados é identificado pela letra *Q* e por um número inteiro seqüencial.

À medida que as consultas são realizadas, os dados recuperados para respondê-las são armazenados no cache e recebem um valor inteiro referente a sua ordem de armazenamento. Usando

uma afinidade temporal como o LRU, a consulta *Q1* seria considerada como parte da região mais fria do cache, por ser a consulta com o rótulo de tempo mais antigo, enquanto a consulta *Q9* seria considerada parte da região quente, por ser a mais recente no cache. Assumindo que na figura 2.2, a distância entre cada consulta represente a afinidade regional entre os dados, então as consultas *Q1* e *Q9* podem ser consideradas próximas, assim como as consultas *Q5* e *Q6*. Dessa maneira, o cache pode ser visto como duas regiões com dados relacionados e duas consultas sem relações regionais com as demais.

Usando duas afinidades para analisar o potencial de reutilização dos dados, o LRU como referência para a substituição e a afinidade regional como auxílio para a definição do potencial de reutilização, a definição do potencial de reutilização dos dados da figura seria feita da seguinte forma: os dados da consulta *Q1*, considerados como candidatos à substituição pelo LRU, passariam a ser considerados como parte da região quente do cache, por apresentar afinidade regional com a consulta *Q9*, considerada com grande potencial de reutilização pelo LRU. Para manter a decisão de substituição a cargo do LRU, os dados da consulta *Q1* recebem o mesmo rótulo de tempo dos dados da consulta *Q9*.

Afinidade semântica

A afinidade semântica é definida por [DFJ⁺96] como uma relação entre dados identificada na especificação de consultas associadas. Ou seja, as condições especificadas nos predicados das consultas são avaliadas para inferir a relação entre os dados.

A afinidade semântica entre dados tem sido bastante explorada nos últimos anos e pode ser encontrada em pesquisas como: [HP02, RD00, LC99, CRS99]. O interesse por essas análises está relacionado à possibilidade de utilizar parâmetros especializados e vinculados ao significado do dado para gerenciamento do conteúdo do cache. As afinidades temporal e freqüencial são parâmetros muito utilizados mas, em alguns casos, se mostram inadequadas por não diferenciar o tipo de dados armazenado no cache. O cache com uso de afinidade semântica permite uma adaptação do seu

gerenciamento ao valor semântico dos dados representados nele. Assim, a avaliação do potencial de reutilização de um dado torna-se mais flexível e ajustada ao valor semântico das consultas realizadas no momento.

O trabalho [RD00] explora a semântica de localização geográfica para gerenciar o cache. Ele supõe que consultas sobre a mesma região de uma cidade, de um mapa ou de uma rodovia estão semanticamente relacionadas por fazerem parte do conjunto de informações sobre aquela região. Seguindo esse raciocínio, uma consulta sobre restaurantes e outra sobre teatros no centro da cidade estão semanticamente relacionadas por fazerem parte do conjunto de informações sobre o centro da cidade. Nesse caso, uma consulta sobre pontos de entretenimento no centro da cidade poderia ser parcialmente atendida pelos dados das duas consultas, evitando que eles fossem solicitados novamente ao servidor.

O trabalho [RD99] usa a afinidade semântica juntamente com a afinidade temporal (LRU) para compor uma afinidade mista de gerenciamento do cache. Para o uso das duas afinidades, o cache é representado por duas regiões: os agrupamentos semânticos e o conjunto de dados de cada consulta. Os agrupamentos semânticos são formados pelos conjuntos de dados das consultas semanticamente relacionadas, ou seja, quando o relacionamento semântico entre duas consultas é identificado, elas passam a compor o mesmo agrupamento semântico. Dessa maneira, os dados semanticamente relacionados podem ser analisados como um agrupamento formado por diversos conjuntos de dados ou como cada conjunto de dados específicos. Para gerenciar o cache, são usadas as informações mostradas nas tabelas 2.1 e 2.2, extraídas de [RD99].

A tabela 2.1 apresenta os dados sobre cada consulta armazenada no cache. Nela estão representados: *Id* o identificador da consulta, *RS* o atributo responsável pela identificação do relacionamento semântico entre as consultas, *AC* as tuplas de atributos de cada consulta, *PC* o predicado da consulta, *NT* o número de tuplas que atendem ao predicado da consulta, *RTC* o rótulo de tempo da consulta e *Ag* o agrupamento ao qual a consulta pertence.

A tabela 2.2 apresenta os dados sobre cada agrupamento semântico identificado no cache. Onde estão representados, *IdA* o identificador do agrupamento, *RTA* o rótulo de tempo do agrupamento

Id	RS	AC	PC	NT	RTC	Ag
S1	Estoque	SNome	$20 < \text{Preco} < 40$	2	T1	1
S2	Empresa	CNumero,CNome	ramo = "computadores"	4	T2	2
S3	Estoque	SNome, Preco	$\text{Preco} > 40$	3	T3	1
S4	Estoque	SNome, Cnome	$\text{Preco} < 20$	8	T4	1

Tabela 2.1: Dados sobre o conjunto de dados de cada consulta.

IdA	RTA	LC
1	T4	$S1 \rightarrow S3 \rightarrow S4$
2	T2	S2

Tabela 2.2: Dados sobre o agrupamento semântico.

e LC representa a lista das consulta que compõem o agrupamento, ordenadas de acordo com o rótulo de tempo das consultas.

Para decidir sobre a substituição de dados no cache utilizando os dados das tabelas 2.1 e 2.2, o trabalho [RD99] propõe um algoritmo chamado de *LRU de dois níveis*, apresentado na figura 2.3, para a substituição dos dados no cache.

O LRU de dois níveis realiza a substituição dos dados em duas etapas. Inicialmente, identifica o agrupamento semântico com o rótulo mais antigo dentro do cache. Em seguida, passa a eliminar, de dentro do agrupamento, os dados das consultas com rótulo de tempo mais antigo. Esse procedimento é repetido até a liberação do espaço necessário ou esvaziamento do cache.

```
LRU-2Níveis{
  Enquanto (existir Agrupamento no cache) faça {
    Ag recebe o agrupamento com o rótulo de tempo mais antigo;
    Enquanto (existir consulta em Ag ) faça {
      XC recebe a consulta de Ag com o rótulo de tempo mais antigo;
      Elimine XC do cache;
      Apague dados sobre XC de Ag;
      Adicione espaço livre;
      Se (espaço livre é suficiente) devolva (sucesso);
    }
    Apague Ag do cache;
  };
  devolva(falha);
}
```

Figura 2.3: Pseudo-código do algoritmo LRU de dois níveis.

2.2 OLAP

Os sistemas OLAP [CD97b, CD97a] têm sido muito utilizados pelas aplicações para análise de dados. Seu crescimento é justificado pela flexibilidade oferecida por suas características multidimensionais para análise de grande volume de dados.

O aspecto multidimensional dos sistemas OLAP tem representado uma extensão para as limitações do modelo relacional. Contudo, os sistemas relacionais não atendem às exigências desse modelo, tornando necessária uma camada intermediária para adaptar os dados às visões multidimensionais.

O esquema no qual os dados são representados para a análise por sistemas OLAP é conhecido

como *esquema estrela*⁵. Ele é formado por dois tipos de tabelas: dimensão e fato. Cada tabela dimensão representa uma abordagem diferente aos dados analisados, ou seja, esses dados podem ser visualizados por meio de diferentes combinações entre as dimensões. A tabela dimensão pode representar um conjunto de tabelas do modelo relacional e seus atributos podem descrever tanto informações sobre seus elementos, quanto modelar uma hierarquia de informações contida na dimensão. A variação da hierarquia dentro de uma dimensão permite modificar os detalhes e o enfoque da análise, por exemplo, em uma dimensão chamada região geográfica, contendo a hierarquia bairro, cidade e estado. Usando essa dimensão, um fato pode ser avaliado considerando o nível hierárquico bairro, o mais detalhado. Caso a análise necessite de dados mais abrangentes e menos detalhados, basta mudar o nível hierárquico, usando como referência uma cidade, formada por diversos bairros, ou um estado, formado por diversas cidades. A tabela fato relaciona as dimensões representadas no modelo às métricas que serão avaliadas. A relação entre as métricas e as dimensões é obtida por intermédio da inclusão das chaves estrangeiras de cada tabela dimensão na tabela fato.

O exemplo a seguir apresenta um esquema estrela extraído de [dRSN98].

Exemplo 1

Tabela Fato

Venda = (vid,pid,lid,tid,vvalor)

Tabelas Dimensão

Produto = (pid,pnome,pcategoria)

Loja = (lid,lnome,lcidade, leestado, lpais)

Tempo = (tid,tdia, tmes, tano)

Cada uma das tabelas-dimensão possui sua hierarquia representada em seus atributos, além de um identificador e um atributo descritivo, por exemplo, a dimensão loja possui *lid* atributo

⁵Tradução do termo em inglês - star schema.

identificador, *lnome* atributo descritivo contendo o nome da loja e as informações hierárquicas sobre a cidade, o estado e o país de sua localização representadas, respectivamente, por *lcidade*, *lestado* e *lpaís*. Essas informações hierárquicas permitem analisar as vendas realizadas pelas lojas, usando os níveis de agregação cidade, estado ou país. A tabela-fato é formada por *vid* identificador do fato, *vvalor* representa a métrica que será avaliada por diferentes consultas, e também possui as chaves estrangeiras *pid*, *tid* e *lid* que referenciam, respectivamente, as tabelas-dimensão Produto, Tempo e Loja.

Normalmente, as consultas em um sistema OLAP são repetitivas e seguem um determinado padrão. O exemplo 2.2 apresenta uma consulta OLAP comum, descrita na linguagem SQL. Nessa consulta, está sendo solicitada a venda mensal de uma certa categoria de produto durante o primeiro semestre do ano.

Exemplo 2

```
Select pnome, tmes, sum(vvalor)
From Venda, Tempo, Produto
Where pcategoria = "roupas"
      and tmes ≥ "jan" and tmes ≤ "jul"
      and Vendas.tid = Tempo.tid
      and Vendas.pid = Produto.pid
Group by pnome, tmes
```

Tipicamente, as consultas OLAP são formadas por uma seleção (*select*) baseada em alguns parâmetros das dimensões e por um agrupamento (*group by*) de acordo com algumas dimensões escolhidas. A execução de consultas OLAP envolvem a junção (*join*) da tabela-fato com as demais tabelas-dimensão, seguida por algumas operações de seleção e agrupamento. Esse padrão de consulta surge como consequência do uso de operações típicas do ambiente OLAP. Como descritas em [MF01], tais operações são:

- *Pivoteamento* – orienta a visão multidimensional dos dados. Ele realiza a rotação entre as dimensões, permitindo escolher quais dimensões devem ser mostradas;
- *Roll-up* – aumenta o nível de agregação, diminuindo o detalhe sobre uma ou mais hierarquias das dimensões;
- *Drill-down* – decresce o nível de agregação, aumentando o detalhe sobre uma ou mais hierarquias das dimensões;
- *Slice-and-dice* – oferece visões de planos do cubo, para enfatizar determinadas informações, ou seja, operações de seleção e projeção são usadas para fixar alguns pontos dentro do cubo e oferecer informações sobre os itens fixados. Por exemplo: fixar um produto e uma cidade para analisar as vendas desse produto, na cidade fixada, durante algum período do ano.

Esse padrão de comportamento das consultas permite otimizações para torná-las mais rápidas. Uma dessas otimizações conhecida como *cubo* é apresentada a seguir.

2.2.1 Cubos

O cubo pré-processado [HK01] é uma das alternativas para explorar o padrão das consultas OLAP e torná-las mais rápidas. Como a maioria das consultas usa algum tipo de agregação, é feito um pré-processamento das agregações possíveis envolvendo as dimensões representadas no esquema. A estrutura de dados obtida é chamada de cubo, ou hipercubo quando envolve mais de três dimensões.

Os diferentes níveis de agregação são chamados de identificadores dos dados no cubo ou *N-D cubo-ids* [HK01], onde *N* representa a quantidade de dimensões envolvidas na agregação. O nível zero (0-D cubo-id), também conhecido como *topo*, representa o mais alto nível de sumarização. À medida que novas dimensões são inseridas na agregação, *N* é incrementado para indicar o nível de agregação atual. O nível mais baixo de agregação é chamado de *base*. Ele contém a agregação de todas as dimensões do esquema. A figura 2.4 mostra as agregações necessárias

para o cálculo do cubo do exemplo 2.2. A geração do cubo demanda tempo e espaço para seu

Figura 2.4: Pré-cálculo das agregações de um cubo

armazenamento, porém, depois de pronto, as consultas OLAP se tornam bem mais rápidas. Para substituir o tempo de processamento dos cubos, os sistemas OLAP usam visões materializadas e estruturas de índices como índices bitmap e índices de junção.

2.3 Visões materializadas

Uma *visão* é um subconjunto dos dados pertencentes às relações de um esquema de dados, ou seja, uma visão pode ser formada por atributos de várias relações, representando uma integração de dados. Em sistemas OLAP, nos quais as consultas são repetitivas e seguem um padrão de processamento, recalculá-la a cada solicitação implica desperdício de recursos computacionais. Para evitar os processamentos repetitivos, são usadas as *visões materializadas* [QGMJ97], previamente calculadas e armazenadas em disco. Assim, as visões materializadas funcionam como um cache para o processamento das visões. Em vez de calcular a visão a cada utilização, ela é processada uma vez, podendo ser reutilizada várias vezes. Como desvantagem, as visões materializadas tornam-se desatualizadas a cada alteração ocorrida nas relações usadas para definí-la. Para resolver esse problema, basta processar novamente a visão, o que pode ser feito quando a atua-

lização ocorre ou pode aguardar o melhor momento para sua realização. A decisão de recalculer a visão materializada é feita de acordo com a necessidade de informações atualizadas. O uso de visões materializadas requer a análise de alguns aspectos como apresentado em [YKL97, IF02]:

1. A realização de um processo de escolha para a definição de quais visões devem ser materializadas.
2. A manutenção autônoma das visões por meio de monitores que captam as alterações ocorridas nas relações bases e decidam o melhor momento para a atualização da visão.
3. A manutenção das visões de forma dinâmica e incremental.

2.4 Estruturas de índices para sistemas OLAP

As estruturas de índices usadas em sistemas OLAP buscam explorar o padrão de suas consultas, caracterizadas pelas operações de junção, seleção e projeção. A seguir, são descritos dois dos índices popularmente encontrados em sistemas OLAP: índices de junção⁶ e índices de bitmap (mapa de bits). O termo bitmap será utilizado no restante do texto em inglês por ser a forma utilizada em publicações em português.

2.4.1 Índices de junção

Os índices de junção são adaptados para tratar junções, o que o torna adequado a consultas OLAP. Diferente dos índices tradicionais que mapeiam o valor de uma coluna para uma lista de instâncias que o utilizam, o índice de junção [HK01] mapeia a junção das instâncias de duas relações. Por exemplo: considerando a junção de duas relações $X(Xid, A)$ e $Y(Yid, B)$, onde Xid e Yid representam os identificadores das relações X e Y , e os atributos A e B são os atributos de ligação, ou seja, os atributos usados para realizar a junção entre as relações. O índice de junção

⁶Tradução do termo em inglês join index.

entre X e Y é representado pela tupla (Xid, Yid). Esse comportamento pode ser observado nas figuras 2.5 e 2.6.

Figura 2.5: Relao entre instncias da tabela-fato Venda e as tabelas-dimenso Localizao e Produto.

A figura 2.5 apresenta algumas instncias das relaes Localizao, Venda e Produto, isto , nela possvel visualizar quais linhas das tabelas esto se relacionando.

A figura 2.6 mostra trs ndices de juno usando as relaes da figura 2.5. A tabela 2.6 a) representa o ndice de juno das relaes Localizao e Venda. A tabela 2.6 b) mostra o ndice de juno das relaes Venda e Produto. E por fim, A tabela 2.6 c) representa o ndice da juno entre as trs relaes. Como pode ser observado, somente so representadas no ndice as tuplas que atendem condio de juno, ou seja, as tuplas que possuem o mesmo valor para o atributo de ligao.

Usando o ndice de juno, as consultas no necessitam processar novamente a juno, evitando a realizao de operaes custosas envolvidas em uma juno [MBK00], como o produto cartesiano entre relaes. Com isso, as consultas so executadas com mais velocidade.

Localização	Venda
⋮	⋮
L9	V45
L9	V57
L9	V205
⋮	⋮

a)

Venda	Produto
⋮	⋮
P7	V45
P7	V120
⋮	⋮

b)

Localização	Produto	Venda
⋮	⋮	⋮
L9	P7	V45
⋮	⋮	⋮

c)

Figura 2.6: Índice de junção.

2.4.2 Índices bitmap

Os índices bitmap são bastante populares em aplicações OLAP por permitir buscas rápidas aos dados do cubo. Eles estão baseados em uma representação alternativa para os identificadores dos atributos de uma tabela. Nos índices bitmap [CI98], para determinado atributo existe um vetor de bits que representa cada valor do domínio de valores do atributo. Se o domínio de valores do atributo possuir n valores, o vetor de bits usado para representá-lo terá o tamanho n . Assim, a presença de um dos valores em uma linha é representada pelo valor 1 na posição do vetor de bits referentes ao valor do atributo. Esse funcionamento está representado na tabelas 2.3, 2.4 e 2.5.

Id	Cidade	Estado
1	São Paulo	SP
2	Rio de Janeiro	RJ
3	São Paulo	SP
4	Campinas	SP
5	Petrópolis	RJ

Tabela 2.3: Tabela de referência.

A tabela 2.3 é formada pelos atributos identificador, cidade e estado. A tabela 2.4 apresenta o índice bitmap para o atributo cidade. Como o domínio de valores do atributo é formado por quatro cidades, o tamanho do vetor de bits para representar cada uma delas é igual a 4. Nesse vetor, o primeiro bit representa a cidade de São Paulo, o segundo a cidade de Campinas, o terceiro a cidade do Rio de Janeiro e, por fim, o quarto bit a cidade de Petrópolis. A representação da cidade de cada linha da tabela é feita atribuindo o valor 1 ao bit da posição correspondente à cidade. A tabela 2.4 repete o mesmo processo de indexação para o atributo estado. Como o domínio de valores do atributo possui dois valores, o tamanho do vetor de bits também é dois.

Id	São Paulo	Campinas	Rio de Janeiro	Petrópolis
1	1	0	0	0
2	0	0	1	0
3	1	0	0	0
4	0	1	0	0
5	0	0	0	1

Tabela 2.4: Índice bitmap para o atributo cidade.

Id	SP	RJ
1	1	0
2	0	1
3	1	0
4	1	0
5	0	1

Tabela 2.5: Índice bitmap para o atributo estado.

Os índices bitmap apresentam a vantagem de redução de espaço e de operações de I/O, pois

representam uma cadeia de caracteres com apenas 1 bit. Além disso, a indexação de atributos com baixa cardinalidade para seu domínio de valores apresenta a vantagem de realizar operações de comparação, junção e agregação usando a aritmética de bits. Essas operações são realizadas com bastante velocidade na maioria dos processadores, porém o tamanho do vetor de bits reduz a velocidade do processamento do índice, quando a cardinalidade do domínio de valores é alta. Por exemplo: para indexar o atributo nome-produto de uma tabela de produtos, na qual a cardinalidade do domínio de valores do atributo é igual a 200.000 produtos, será necessário um vetor de bits com 200.000 bits. Alternativas para contornar esse problema podem ser encontradas em trabalhos como [Wu99, CI99, WB98].

2.5 Soluções para a memória principal

Com a diminuição do preço da memória RAM, a execução total ou parcial de aplicações em memória têm surgido como boa alternativa para tornar as aplicações mais rápidas. Com isso, aplicações que gerenciam alguns gigabytes de dados podem ser implementadas para serem executadas em memória com custo relativamente baixo. Esse contexto tem potencializado pesquisas específicas sobre o uso da memória para otimização de aplicações, entre elas: gerenciadores de banco de dados totalmente residentes na memória, como o *TimesTen* [Tea99] e aplicações [SCJ01, BMR01, KCK01, RR00] que usam a memória para armazenar estruturas de índice e algumas tabelas para controle de dados. As consultas OLAP também têm sido alvo de otimizações por meio da criação em memória de estruturas de índices para acessar os dados de cubos [MK98, RR99].

O *CCICLO* apresenta uma variação dos trabalhos que abordam índices e banco de dados residentes em memória, por intermédio de estrutura própria para representação e cruzamento de objetos classificados. O *CCICLO* representa em memória uma síntese dos objetos armazenados em disco formada por seus identificadores no disco e pelas classificações atribuídas a eles por diferentes classificadores. Além disso, ele permite o cruzamento das classificações dos objetos por operadores relacionais adaptados para sua estrutura e para a semântica de classificação dos dados.

Descoberta de conhecimento por meio de classificadores

A busca por novos conhecimentos em grandes bancos de dados é uma tarefa em expansão e com tendência de continuar crescendo na proporção em que crescem o volume e a variedade dos dados acumulados. Entre os pontos que instigam a análise desses dados estão: o desejo de monitoramento do comportamento de grupos de indivíduos ou objetos, por meio da análise evolutiva dos dados coletados sobre eles e a identificação da interdependência entre grupos ou classes. Porém, o mesmo volume de dados que instiga a busca por novos conhecimentos, pode se tornar uma grande barreira a ser superada, quando são consideradas as limitações humanas para avaliar, de forma simultânea, grande quantidade de informações, além das limitações de recursos para representação do conhecimento e para análise de dados.

A capacidade de gerar, coletar e armazenar dados unida ao desejo de analisá-los têm estimulado a busca por recursos que permitam avaliar grandes volumes de dados de forma eficiente e compreensível, impulsionando os estudos na área de *mineração de dados* (*data mining*) ou *descobrimto de conhecimento em banco de dados* (*knowledge discovery in database*) [HK01] que é o processo de extração automática de conhecimentos novos e úteis escondidos em grandes conjuntos de dados.

Existem várias técnicas para descoberta de conhecimento por meio da análise de um conjunto

de dados, entre elas encontram-se [HK01]: *análise de regras associadas*, *classificação de dados*, *predição*, *análise de agrupamentos (clusters)*, *análise de exceções (outliers)* e *análise evolutiva*. A técnica escolhida para utilização nesse trabalho é a classificação de dados. Tal escolha foi motivada por três aspectos:

- Por ser uma das principais tarefas para obtenção de novos conhecimentos [Fre02, AEK00, EEvH99],
- Por representar o conhecimento de uma maneira mais aproximada da forma de raciocínio humano, pois os seres humanos estão acostumados a organizar seus conhecimentos dividindo-os em grupos de acordo com a presença ou ausência de determinadas características, e utilizar tais grupos ou classes para abstrair a complexidade que envolve a definição de cada uma delas,
- Por ser muito utilizada em pesquisas de diferentes áreas de conhecimento [WMSW00, Cha01, YRT⁺01, SKATSR01, GKF⁺01, IA00]. Nos mais variados ramos da ciência, como biologia, química e física é possível encontrar o conceito de classificação. Esse ponto em comum pode ser utilizado para facilitar a integração de informações obtidas por cada ramo da ciência e para analisar as possíveis relações entre elas.

Um exemplo para ilustrar a popularidade da classificação é a divisão dos elementos da natureza em reino animal, reino vegetal e reino mineral, além de suas subdivisões em espécies e subespécies. Cada uma dessas classificações pode ser abordada em diferentes níveis de abstração e definida por características diferentes. Ao perguntar a uma criança qual a diferença entre os membros do reino animal e do reino mineral, ela talvez responda que os membros do reino animal possuem vida e os do reino mineral não possuem. A complexidade por trás da definição de vida pode ser tratada por diversos pontos de vista. Um filósofo pode desejar uma definição mais elaborada do conceito de vida, do que aquele oferecido por uma criança. Mas, provavelmente, ambos não teriam dificuldade para dizer a qual dos reinos pertence um cachorro ou uma pedra.

Os exemplos de classificações podem ser encontrados em abundância em nosso dia-a-dia, por causa das facilidades proporcionadas por seus diferentes níveis de abstração para compreensão das "coisas" que nos cercam. Assim, indivíduos com formação acadêmica e quantidade de informações diferentes podem ter uma compreensão aproximada do que é observado, ou seja, em um processo de classificação, é possível que definições ou descrições diferentes de determinada classe gerem classificações idênticas ou bem aproximadas. Isso pode ser exemplificado considerando uma situação hipotética em sala de aula, em que os alunos receberiam a tarefa individual de definir regras para classificação de seres humanos em duas classes: uma com indivíduos do sexo masculino e outra com indivíduos do sexo feminino. Provavelmente, surgiriam diferentes regras para a definição dessas classes, porém a aplicação delas, apesar de suas diferenças, poderia gerar classificações idênticas ou muito aproximadas. Isso pode ser explicado pela diferença de enfoque sobre as características que definem uma classe, por exemplo, o método mais preciso para definição do sexo de um ser humano é a verificação de suas características genéticas, porém outros métodos com menor precisão como o exame dos órgãos sexuais ou de algumas características de fenótipo, como presença de barba, tom de voz, tipo de roupa ou corte de cabelo podem apresentar um resultado satisfatório na maioria dos casos, deixando apenas os casos mais complicados para serem analisados por métodos mais precisos e sofisticados.

É importante notar que as informações baseadas no fenótipo, no genótipo ou em conceitos culturais são informações complementares, e juntas transmitem uma informação mais ampla sobre o indivíduo. Os cruzamentos das informações obtidas por diferentes classificações podem produzir novos conhecimentos sobre o indivíduo analisado e permitem, até mesmo, o aperfeiçoamento das regras para sua classificação.

Exemplos como o anterior podem ser encontrados em diversas áreas do conhecimento, como é o caso das pesquisas sobre a predisposição de um indivíduo a determinadas doenças. Pesquisas sobre esse assunto podem abordar o indivíduo por diferentes pontos de vista, sejam eles genéticos, ambientais ou comportamentais. Classificar tais indivíduos segundo diferentes critérios e cruzar os dados obtidos pode auxiliar na identificação mais precisa dos motivos que aumentaram a vul-

nerabilidade de um indivíduo a uma determinada doença. Assim, diferentes regras de classificação podem ser analisadas em busca de características que aumentem o conhecimento sobre um conjunto de classes ou conceitos. Dessa maneira, o cruzamento dos resultados obtidos por diversos tipos de classificações, considerando a avaliação de grandes volumes de dados, pode ser uma importante ferramenta para descoberta de conhecimento.

Este trabalho não se aprofunda na descrição do processo de classificação de dados e de suas técnicas, pois seu propósito é cruzar os dados classificados por diferentes classificadores, independente da técnica utilizada para projetá-los. Porém, há uma preocupação com a integração dos dados gerados por diferentes classificadores, apresentando um formato padrão e simplificado para a representação dos dados classificados. Dessa maneira, independente da técnica utilizada pelos classificadores, os dados classificados podem ser integrados e analisados. Além disso, esse trabalho apresenta uma visão geral sobre o projeto e a utilização dos classificadores para permitir uma compreensão global do processo.

3.1 Classificação de dados

A definição de classes e a capacidade de identificar os elementos pertencentes a cada uma delas é uma das principais tarefas de *mineração de dados*. Seu objetivo é distinguir classes dentro de um conjunto de dados que representam objetos. As técnicas de *mineração de dados* têm contribuído para a classificação de dados com o desenvolvimento de técnicas e algoritmos, chamados de *classificadores* [HK01], cuja implementação permite analisar um grande volume de dados de forma automática e inferir a classe de cada um deles.

A identificação de classes dentro de um conjunto de dados, entre outras coisas, permite um tratamento diferenciado para cada classe identificada, explorando suas particularidades. Essa personalização de tratamento pode ser encontrada em diversas áreas do conhecimento e com variadas aplicações. Como nas áreas comercial e financeira, em que o comportamento dos clientes é estudado para identificação de características definidoras de classes de consumidores, como bons e

maus pagadores, clientes com potencial interesse ou não por um produto e até mesmo identificar fraude pela variação de comportamento de indivíduos de uma classe. Já na área biológica, o conceito de classes pode ser facilmente identificado. A definição de espécies é uma das características centrais dessa área do conhecimento. Essas aplicações práticas para a classificação de dados ajudam a visualizar a sua importância, porém classificar dados só é uma tarefa simples quando os atributos que indicam a qual classe um dado pertence são conhecidos previamente. Tais atributos são chamados de *atributos de classe* [HK01]. Por exemplo, criar uma classe apenas com animais bípedes é uma tarefa trivial, caso esteja claro que essa classe é formada por animais com dois pés e que o banco de dados analisado possua um atributo informando a quantidade de pés de cada animal. Mas nem sempre as regras ou atributos de classe são conhecidos. Para esses casos, as técnicas de classificação de dados são aplicadas. Por meio delas um conjunto de dados, sem *atributos de classe* [HK01] definidos, tem os valores de seus atributos avaliados para identificar a classe a qual cada um pertence.

3.1.1 Processo de classificação de objetos

A classificação de objetos compreende as seguintes etapas: o estudo dos dados a serem analisados, o projeto e a implementação dos classificadores e seu uso para classificar um conjunto de dados.

Um classificador possui a modelagem de determinada classe e, uma vez definido, é capaz de analisar dados e definir se os mesmos pertencem ou não à classe modelada. A análise dos dados é feita pela avaliação do *vetor de características* do classificador e dos dados submetidos a ele. Um *vetor de características* [HK01] é a representação das características ou dos atributos avaliados por um classificador para identificar a classe de cada dado. Essa representação é feita em forma numérica para permitir sua análise por métodos estatísticos.

A figura 3.1 apresenta um diagrama do processo de classificação de dados, em que, a base de treinamento é retirada da base de dados e enviada para o processo de criação de classificadores. A criação dos classificadores pode ser realizada por técnicas de análise supervisionada ou não-

supervisionada. Uma vez criados, os classificadores são usados para analisar a base de dados e identificar a classe dos objetos. Por fim, as classificações podem ser armazenadas no banco de dados.

Figura 3.1: Processo de classificação de dados.

Criação dos classificadores

Para a criação de classificadores é necessário reunir informações sobre os dados a serem classificados e sobre as possíveis classes a serem formadas. As fontes para se obter essas informações podem variar muito e, em alguns casos, a única fonte disponível são os dados a serem analisados.

A criação ou identificação dos classificadores é feita por meio de técnicas de reconhecimento de padrão que analisam uma *base de treinamento*. Uma *base de treinamento* [HK01] é uma amostra de dados extraída, de forma aleatória, dos dados a serem trabalhados. Ela pode ser analisada usando *técnicas supervisionadas* e *não-supervisionadas*.

A análise usando *técnicas supervisionadas* é feita quando a classe a qual pertence cada dado é conhecida e indicada pelo atributo de classe. A partir desse atributo, a *base de treinamento* é

varrida por algoritmos de aprendizado, em busca de características comuns aos dados da mesma classe. Quando encontradas, tais características são utilizadas para projetar um classificador capaz de identificar indivíduos dessa classe. Os algoritmos de aprendizado e as técnicas para projetar os classificadores não são descritos aqui por fugirem ao escopo do trabalho.

Para exemplificar a análise supervisionada, considere o exemplo de um banco de dados de veículos com um atributo de classe chamado de tipo de veículo e cujo conteúdo indica se um veículo pertence à classe dos veículos terrestres, aéreos ou aquáticos. Nesse caso, os algoritmos de aprendizado iniciam suas análises com a separação dos dados da base de treinamento nas três classes contidas no atributo de classe. Em seguida, cada classe é varrida em busca de características e comportamentos capazes de identificar se um objeto pertence ou não a essa classe.

A análise usando *técnicas não-supervisionadas* ocorre quando não se conhece um atributo de classe. Nesse caso, os atributos dos dados são analisados utilizando métodos estatísticos de definição de distância ou a similaridade entre os dados. Os métodos de distância buscam garantir a maior proximidade ou similaridade entre os dados da mesma classe e a maior distância ou dessemelhança entre dados de classes diferentes. As análises usando técnicas não-supervisionadas também são conhecidas como *técnicas de agrupamento* ou *clustering*.

Validação dos classificadores

Tipicamente, após o processo de classificação, seja ele supervisionado ou não, vários classificadores são obtidos, porém apenas alguns deles produzem resultados satisfatórios. Portanto, cada classificador deve ser testado para verificar a precisão da classificação realizada por ele. Para a realização do teste, uma nova amostra dos dados, chamada de *base de teste*, é extraída dos dados a serem analisados e isso é feito de forma aleatória e independente da *base de treinamento*. Em seguida, a base de teste é analisada pelo classificador. Para medir a eficiência do classificador, a base de teste é classificada de forma confiável e validada. O resultado obtido é comparado com o resultado gerado pelo classificador. Normalmente, a confiabilidade do classificador é medida pela

quantidade de acertos obtidos durante a classificação dos dados.

Os classificadores considerados aceitáveis passam a compor o conjunto de classificadores válidos para análise de bases de dados compatíveis.

3.2 Dados classificados

Um mesmo conjunto de dados pode ser classificado diversas vezes e por diferentes classificadores. Os resultados obtidos em cada momento podem ser utilizados para diferentes fins, tornando seu armazenamento necessário para evitar a repetição das classificações.

A forma como os dados classificados são armazenados e recuperados pode variar de aplicação para aplicação e, dependendo do volume e da complexidade dos dados, pode ser necessário o uso de estruturas de dados e índices especiais.

Este trabalho não abrange o processo de armazenamento e recuperação dos dados armazenados em disco, pois não considera tais tarefas como parte do componente para cruzamento de dados. Portanto, as solicitações e envios de dados para o disco feitas pelo *CCICLO* devem ser tratadas pela camada responsável pelo gerenciamento dos dados em disco.

3.2.1 Estrutura para representação de classificações

A estrutura para cruzamento de objetos em memória foi preparada para trabalhar com classificadores que definem duas classes de objetos: uma classe chamada de *objetos classificados*, contendo os objetos que atendem às regras modeladas no classificador, e outra classe chamada de *objetos não classificados* contendo os objetos que não atenderam às regras do classificador.

Para a classificação de veículos em três classes como veículos terrestres, aéreos e marítimos seriam necessários três classificadores: o primeiro para classificar os veículos como terrestre e não-terrestre. O segundo para classificar os veículos como aéreos e não-aéreos, e o terceiro para classificá-los como marítimos e não-marítimos.

O uso de classificadores especializados em identificar os componentes de uma classe específica ajuda a reduzir a complexidade para a criação dos classificadores, pois só é necessário modelar uma classe, em vez de várias classes. Além disso, as classes de objetos não classificados podem ser utilizadas para cruzamento com outras classes de objetos, oferecendo outro ponto de vista para a análise dos dados. Em alguns casos, saber que um objeto não pertence a uma classe é tão importante quanto saber a que classe ele pertence. Por exemplo: saber que um paciente pertence ao grupo dos pacientes HIV positivo, e não pertence ao grupo de pacientes com comportamento de risco, desperta o interesse sobre a forma de contágio do paciente, pois pode indicar um novo meio de contaminação pelo vírus ou uma falha na definição de comportamento de risco; entre outras possibilidades.

Para representar uma classificação, são necessários dados sobre o classificador usado, sobre o processo de classificação e sobre os objetos analisados.

3.2.2 Formato de dados para utilização do CCICLO

O ponto central para integração dos dados gerados por diferentes classificadores de objeto é a definição de um padrão para representação de seus dados. A partir desse padrão, os dados poderão ser unidos em uma estrutura comum, independente da forma como foram obtidos. Para a utilização do *CCICLO*, devem ser fornecidos três grupos de dados: os dados referentes ao *vetor de característica* do classificador, os dados sobre o processo de classificação e os dados sobre os objetos analisados.

Formato de dados para representação do processo de classificação

A representação do processo de classificação é formada tanto por dados referentes ao classificador, quanto por dados referentes à execução da classificação. Tais dados devem ser representados em um vetor com o seguinte formato:

$\langle Oid, \langle (Característica1, valor1), \dots, (CaracterísticaN, valorN) \rangle, OrigemDados, DataHora, Pt \rangle$

em que *Oid* é um atributo numérico que representa o identificador do classificador, funcionando como um apontador para seus dados em disco. O vetor de tuplas $\langle (Característica1, valor1), \dots, (CaracterísticaN, valorN) \rangle$ representa o vetor de características do classificador, em que cada tupla $(Característica, valor)$ contém uma das características avaliadas pelo classificador com seu respectivo valor. O atributo *OrigemDados* identifica a base de dados de referência dos objetos analisados. O atributo *DataHora* indica a data e a hora da realização da classificação, e o atributo *Pt* representa o ponteiro para o conjunto de dados sobre os objetos avaliados pelo classificador.

Formato de dados para representação dos objetos analisados

A representação dos objetos analisados pelo classificador é feita considerando apenas as informações sobre a identificação do objeto e sobre a classificação recebida por ele, abstraindo seus demais dados.

Os dados analisados devem ser representados como uma lista ligada, ordenada pelo atributo identificador, em ordem crescente. Cada elemento da lista deve possuir o seguinte formato:

$$\langle Id, R, Prox \rangle$$

em que *Id* é um atributo do tipo inteiro que aponta para o dado armazenado em disco, *R* é um atributo binário que recebe o valor 1 (um), caso o dado tenha sido classificado, ou o valor 0 (zero), caso não seja classificado. O atributo *Prox* representa o ponteiro para o próximo elemento da lista.

Uma representação visual dos dados de uma classificação é apresentada na figura 3.2. Na posição horizontal, apresenta os dados do classificador, do processo de classificação e um ponteiro para os dados dos objetos analisados. Na posição vertical, a figura apresenta uma lista ligada contendo os dados sobre os objetos analisados, cada um com seu identificador em disco, classificação recebida e ponteiro para o próximo objeto da lista.

Figura 3.2: Estrutura de dados para representação dos objetos classificados.

Cruzamento de objetos classificados

O cruzamento de objetos classificados surge como mais uma opção para análise dos objetos classificados. Seu objetivo é aproveitar classificações existentes e explorar as possíveis relações entre elas. Encontrar informações contidas na relação entre as classes pode ser tão importante quanto a definição das classes. A análise das relações entre classes permite descobrir como determinadas alterações em uma classe pode afetar as classes relacionadas a ela. Considerando que uma classificação oferece determinada organização de indivíduos de acordo com algumas características. Um mesmo indivíduo ou objeto pode receber diferentes classificações, de tal forma, as classes obtidas a partir delas estão relacionadas entre si, por qualificarem os mesmos objetos. Isso ajuda a perceber a abrangência das relações entre classes. Por exemplo: um conjunto de dados com informações sobre pessoas pode conter informações representando diversas áreas da vida, como vida pessoal, profissional, acadêmica, informações sobre tratamentos clínicos, entre tantas outras informações possíveis. Todavia, as informações pertencem a uma mesma pessoa, e um evento importante em sua vida, como doença ou desemprego, pode impactar todas as outras classificações recebidas por ela.

Para ilustrar um pouco mais as diferentes classificações que um mesmo objeto ou indivíduo pode ter e a importância da relação entre elas, considere um subconjunto de informações que pode ser obtido sobre um paciente: dieta alimentar, dados antropométricos, antecedentes patológicos, medicamentos em uso ou utilizados, antecedentes familiares, exames clínicos e laboratoriais,

características genéticas, estilo de vida (englobando atividade física regular), tabagismo, comportamentos de risco; entre outras informações. De cada um desses grupos de informações podem-se obter diversas classificações, cada uma focando um ponto de vista e obviamente relacionadas, por se tratar dos mesmos indivíduos. Baseadas nas relações sobre os diversos grupos de informações acerca de um paciente, pesquisas médicas como [CGB01] têm descoberto novos e significativos conhecimentos sobre o tratamento e a prevenção de doenças. Pela observação de grupos de pacientes com doenças cardiovasculares, confirmadas por exames clínicos e laboratoriais, e de pacientes com antecedentes familiares de manifestação da doença foi possível inferir a predisposição hereditária para doenças cardiovasculares, o que atualmente é considerado um dos fatores de risco da doença.

Explorar a relação existente entre as classes é uma forma de descoberta do conhecimento explorada por diversas áreas da ciência, como a medicina, citada no exemplo anterior. Os classificadores podem contribuir com esse processo por meio da avaliação automática de grandes volumes de dados para identificação de classes. Entretanto, eles não são modelados para identificar os relacionamentos entre as classes. Assim sendo, uma importante tarefa a ser executada é o desenvolvimento de ferramentas capazes de avaliar diferentes classes, explorando a relação entre elas.

Cruzar dados sobre diferentes classificações de objetos implica a capacidade de integração e avaliação desses dados, para que seja possível a realização de operações como seleção e comparação entre os elementos de cada classe. O cruzamento entre dados de objetos classificados apresenta algumas vantagens como:

- Aproveitamento de classificações realizadas – Como a realização de uma classificação pode demorar muito, o uso de classificações já realizadas é um grande atrativo para o cruzamento de objetos classificados, pois oferece uma nova perspectiva para a análise dos objetos, sem a necessidade de classificá-los novamente.
- Integração de conhecimentos multidisciplinares – Cruzar dados classificados por diferentes classificadores representa uma integração dos diversos conhecimentos especializados envolvidos na criação dos classificadores. Esses conhecimentos muitas vezes são difíceis de serem

reunidos em uma equipe de trabalho.

- Diminuição da complexidade do processo de análise – A integração de dados de classificadores permite que os especialistas se dediquem à criação de classificadores específicos e deixem a análise das relações entre as classes para o cruzamento dos dados delas, evitando a criação de classificadores que modelem, de forma simultânea, classes de áreas de conhecimentos distintas.

Apesar das vantagens apresentadas, o cruzamento de dados também possui algumas dificuldades para exploração de seu potencial, tais como:

- Integração dos dados – A integração de conhecimento oferecida pelo cruzamento de objetos classificados só é possível se as informações sobre os objetos estiverem em um formato que permita a leitura e o processamento delas. Porém, muitas vezes, por fazerem parte de projetos diferentes, os dados são armazenados em formatos incompatíveis, apresentando divergências que dificultam sua integração, como níveis de detalhes diferentes, armazenamento em bancos de dados heterogêneos, entre outras.
- Recursos para processamento – Dependendo da quantidade de dados em cada classe, o cruzamento delas pode demandar muitos recursos computacionais, como trocas de dados entre a memória e o disco.
- Compreensão do significado de cada classe envolvida – A busca pelo conhecimento expresso por meio do relacionamento entre diferentes classes exige a compreensão de cada uma delas e uma visão global do relacionamento entre os dados que as representam. Isso não é uma tarefa trivial, principalmente se considerado o aumento de informações advindo da inclusão de cada nova classe no processo.

As vantagens oferecidas pelo cruzamento de objetos classificados, juntamente com a possibilidade de superar as limitações para realizá-lo, motivaram a criação do *CCICLO*.

4.1 Características do CCICLO

Este trabalho foi desenvolvido procurando explorar as vantagens do cruzamento de objetos classificados, propondo soluções para as desvantagens apresentadas. O *CCICLO* foi definido para apoiar a descoberta de conhecimento em banco de dados por meio da análise das relações entre as classes, oferecendo um desempenho que permitisse a avaliação de hipóteses, de maneira rápida e com o envolvimento do pesquisador. Para oferecer esse desempenho, foram estudadas alternativas para os pontos de maior consumo de tempo no processo. Entre eles, foram destacados três pontos: a integração de dados, a troca de dados entre memória principal e disco para o processamento de grandes volumes de dados, e a dificuldade para compreensão dos conceitos envolvidos na definição das classes. Para tratar esses pontos, o *CCICLO* está fundamentado em quatro estratégias: o uso da memória principal, uma representação multidimensional para o vetor de características do classificador e os dados sobre o processo de classificação, uma representação compacta para os objetos classificados, e a definição de operadores para o cruzamento em memória dos dados classificados.

4.1.1 Uso da memória principal

O avanço do desenvolvimento de computadores apresenta dois descompassos cruciais para o desenvolvimento de aplicativos. O primeiro é o crescimento da velocidade dos processadores em relação a velocidade de acesso a dados armazenados em disco. O segundo é o crescimento da capacidade de armazenagem em disco em relação à memória do computador. Esses dois descompassos conjugados geram um desperdício de tempo e processamento, quando é preciso avaliar um grande volume de dados.

Como existem mais dados em disco do que o espaço disponível para armazená-los na memória do computador, não é possível processá-los todos ao mesmo tempo, sendo necessária a liberação de espaço em memória, de tempo em tempo, por meio da troca de dados entre memória e disco.

O *CCICLO* apresenta uma estrutura para representação em memória de dados sobre classifica-

ções. Com essa representação, o *CCICLO* pretende diminuir as trocas de dados entre a memória e o disco, disponibilizando, em memória, os dados necessários para o cruzamento dos objetos classificados.

4.1.2 Representação de objetos classificados

A solução apresentada pelo *CCICLO* é baseada em uma representação compacta dos objetos classificados. Essa representação resume um objeto classificado a dois atributos. Um atributo usado para identificar o objeto e o segundo contendo a classificação recebida por ele. Os demais atributos relacionados ao objeto não são representados na estrutura do *CCICLO*, reduzindo assim, o espaço necessário para representá-los. Representando somente a classificação recebida pelo objeto, o *CCICLO* propõe uma abstração para o processo de classificação, deixando de lado a complexidade de suas regras e focando os resultados gerados por ela.

Essa simplificação do modelo de representação das classificações permite disponibilizar em memória uma maior quantidade de dados, reduzindo a necessidade de troca de dados entre memória e disco. Além disso, quanto menor a quantidade de dados envolvida no processamento, mais rápida e simples pode ser a sua execução. Outro benefício obtido pela simplificação da representação é o aumento da facilidade de compreensão global das classes analisadas. O uso de muitas classes em um cruzamento implica aumentar o grau de dificuldade para compreensão do processo, principalmente, se considerados todos os conceitos e as regras envolvidos na definição de cada classe. No *CCICLO*, os detalhes sobre os objetos são deixados para um momento de refinamento da análise, ou seja, quando os cruzamentos apresentarem resultados que despertem o desejo por avaliações mais detalhadas.

Os capítulos seguintes irão apresentar a estrutura de dados para representação dos objetos, a estrutura para representação multidimensional do classificador e dos dados sobre o processo de classificação, e os operadores relacionais adaptados para realizarem o cruzamento dos objetos classificados.

Estrutura de dados para cruzamento de objetos

Para a descoberta de conhecimento por meio do cruzamento de dados classificados em memória ficar mais abrangente, é necessário agregar aos dados classificados informações sobre o processo de classificação.

Este trabalho busca analisar os dados classificados avaliando-os juntamente com os dados sobre o processo de classificação, tais como: o vetor de característica do classificador, o conjunto de dados avaliados e a data de sua avaliação. Dessa forma, é possível agregar maior valor semântico aos dados classificados, permitindo que a análise seja feita considerando também características comuns ou relacionadas aos classificadores como possuir atributos comuns no vetor de características, classificar a mesma base de dados ou analisar a evolução temporal das classificações.

Para oferecer tal funcionalidade, é apresentada uma estrutura de dados para representação, em memória, dos dados que representam uma classificação.

5.1 Objetos complexos

A estrutura de dados proposta faz uso de um objeto complexo para representação em memória de um classificador.

Objetos complexos [Liu92] são objetos formados por diferentes conjuntos de dados. Nesse caso, o objeto complexo classificador, ou somente classificador, é formado por dados sobre o processo de classificação e pelos dados analisados por ele. A representação desses dados é feita por quatro conjuntos. Um conjunto de dados de controle, um conjunto de dimensões, um conjunto representando os objetos classificados e um conjunto de objetos não classificados como mostra a figura 5.1.

Figura 5.1: Objeto complexo – classificador.

Objeto complexo – classificador:

```
{(<Oid, OrigemDados, DataHora, Pt0, Pt1>),
 (<Dim1, Val1>, ..., <DimN, ValN>),
 (<Id1, Prox>, ..., <IdN, Prox>),
 (<Id1, Prox>, ..., <IdN, Prox>)}
```

- (<Oid, OrigemDados, DataHora, Pt0, Pt1>) em que *Oid* representa o identificador do objeto complexo, *OrigemDados* indica o banco de dados de onde os objetos analisados pelo classificador se originaram, *DataHora* indica a data e a hora de classificação dos objetos, *Pt0*

representa um ponteiro para os dados não classificados e *Pt1* representa um ponteiro para os classificados.

- ($\langle \text{Dim1}, \text{Val1} \rangle, \dots, \langle \text{DimN}, \text{ValN} \rangle$) representa o conjunto de dimensões do classificador, formado pelos atributos e valores obtidos a partir do vetor de característica do classificador, incluindo data e hora de classificação.
- ($\langle \text{Id1}, \text{Prox} \rangle, \dots, \langle \text{IdN}, \text{Prox} \rangle$) representa o conjunto dos dados que atenderam às regras do classificador. Esses dados são chamados de dados classificados. O conjunto é estruturado por uma lista ligada, mantida em ordem crescente dos valores do atributo *Id*, um ponteiro para o dado no banco de dados. O atributo *Prox* é um atributo de controle da lista ligada, cuja função é apontar para o próximo elemento.
- ($\langle \text{Id1}, \text{Prox} \rangle, \dots, \langle \text{IdN}, \text{Prox} \rangle$) representa o conjunto dos dados que não atenderam às regras do classificador. Esses dados são chamados de dados não classificados. Ele possui a mesma estrutura e regras do conjunto de dados classificados.

5.2 Representação multidimensional do vetor de características

Um ponto importante para o cruzamento de dados é a escolha dos classificadores e dos dados para o cruzamento. O método mais simples é a escolha pontual do classificador e dos objetos classificados usando algum tipo de descrição ou identificador. Porém, esse método apresenta pouca eficiência e flexibilidade, pois exige um conhecimento prévio dos classificadores disponíveis e não permite uma busca ampla baseada em algumas das características do classificador.

O *CCICLO* propõe uma forma mais ampla e flexível para a escolha dos dados para cruzamento. Permitindo escolher os classificadores e os objetos classificados baseado em informações do vetor de características do classificador e do processo de classificação. Além disso, a seleção dos dados para

o cruzamento de acordo com o vetor de característica do classificador, a base de dados classificada e o momento da classificação agrega maior valor semântico aos objetos cruzados, pois é possível não somente saber se um dos dados foi classificado ou não, como também conhecer algumas informações sobre o processo de classificação. Por exemplo: a escolha para o cruzamento de objetos complexos representando utilizações, em períodos diferentes, de um mesmo classificador para classificar a mesma base de dados pode denotar um comportamento evolutivo da base de dados. Um exemplo prático é o de uma base de dados de paciente classificada por diferentes classificadores que identifica classes de pacientes com propensão a diferentes doenças. A escolha desses classificadores permite identificar os pacientes classificados em mais de um grupo ou em todos e os fatores de riscos mais comuns encontrados no vetor de características dos classificadores.

Para permitir essas funcionalidades, o *CCICLO* propõe o uso da R-Tree¹.

A R-Tree [Gut84] é uma estrutura de dados hierárquica derivada dos conceitos da Árvore-B (B-Tree) [Com79] para representar, acessar e manipular dados multidimensionais.

A R-Tree possui algumas extensões que apresentam variações, em alguns de seus algoritmos (inserção, remoção, busca e divisão de nós) para tornar a estrutura mais adaptada a determinadas situações ou características dos dados representados. Algumas dessas extensões são:

- R^+ -Tree [SRF87] apresenta uma variação do algoritmo de inserção da R-Tree visando evitar sobreposições de regiões, diminuindo o custo das operações de busca. Ela difere da R-Tree por não garantir que pelo menos a metade da capacidade dos nós seja utilizada. Nela, as regiões não se sobrepõem e um objeto pode ser armazenado em mais de um nó.
- R^* -Tree [BKSS90] é uma extensão da R-Tree baseada no estudo do comportamento de seus algoritmos ao processar diferentes distribuições de dados. Ela possui quatro objetivos de otimização: minimizar a sobreposição das regiões, minimizar a superfície de cada região, minimizar a área coberta por cada nó interno e maximizar a utilização dos nós.

¹O termo em inglês *R-Tree* pode ser traduzido por Árvore-R, porém será utilizado em inglês durante o restante do texto por ser uma forma amplamente usada na literatura científica em português.

- X-Tree [BKK96, BKSS90] é uma extensão da R^* -Tree projetada para trabalhar com objetos de elevado número de dimensões. Ela difere da R^* -Tree em três pontos: a X-Tree possui nós com páginas expandidas oferecendo grande capacidade de armazenagem, não possui sobreposição de regiões e a divisão dos nós é feita de acordo com um histórico de divisão.

A forma tradicional da R-Tree foi escolhida para representar os classificadores, para manter a abrangência da estrutura, evitando sua vinculação a comportamentos específicos dos dados representados como quantidade de dimensões, densidade de dados em cada nó da árvore, sobreposição e cobertura das regiões representadas. Porém, qualquer uma de suas extensões pode ser incorporada no *CCICLO*. Uma das futuras expansões do *CCICLO* pode ser o desenvolvimento de monitores que identifiquem variações no comportamento dos dados analisados e sugiram a melhor estrutura de árvore a ser adotada.

A seguir é apresentada uma breve descrição do funcionamento da R-Tree para auxiliar na explicação de sua contribuição para o ambiente de cruzamento de dados classificados.

5.2.1 R-Tree

A R-Tree, assim como a B-Tree, possui uma estrutura de árvore formada por raiz, nós-folha e nós internos com ponteiros para os próximos nós, permitindo percorrer toda a árvore partindo de seu nó-raiz até atingir os nós-folha. A diferença entre elas encontra-se na forma de representação dos dados em sua estrutura. A B-Tree representa os dados de forma pontual, já a R-Tree representa as regiões nas quais os dados estão distribuídos. Dessa forma, é possível recuperar uma região com todos os seus dados.

Os nós-folha da R-Tree são da forma:

$$\{(Oid, LIMITES)\}$$

em que *Oid* é um identificador do objeto e é usado para apontar um dado no banco de dados, e *LIMITES* contém as dimensões do objeto. Em um espaço bidimensional, com coordenadas x e y , o atributo *LIMITES* possui a forma:

$$(x_{inferior}, x_{superior}, y_{inferior}, y_{superior})$$

em que estão representadas as coordenadas dos limites inferior esquerdo e superior direito da região ocupada pelo objeto.

Os nós intermediários são da forma:

$$(PONTEIRO, LIMITES)$$

em que *PONTEIRO* é um apontador para um nó no nível imediatamente inferior da árvore e *LIMITES* delimita a menor região que cobre todas as regiões representadas no nó apontado por *PONTEIRO*.

Como a descrição do funcionamento da R-Tree não é o foco central deste trabalho, a seguir será descrito somente o algoritmo de busca da R-Tree, por ser o mais importante para a compreensão da colaboração da mesma no processo de análise das dimensões dos classificadores. Os demais algoritmos (inserção, remoção e divisão de nós), propriedades e características da R-Tree, assim como de suas extensões, estão descritos na literatura, podendo ser encontrados em [Gut84, BOSD⁺97, KF93, SC00, BBK01].

Figura 5.2: R-tree.

O algoritmo de busca da R-Tree é descrito a seguir.

Busca(T,R)

Entrada: Onde T é o nó-raiz da R-Tree, e R representa a região dos dados que devem ser recuperados.

Saída: Todos os objetos de dados que se encontram na região R .

Corpo do Algoritmo:

1. Se T é um nó não-folha, faça uma busca recursiva em todos os nós apontados pelo nó T e que possua $R \cap LIMITES \neq \phi$. Ou seja, cuja região coberta pelo nó não-folha $LIMITES$ possua sobreposição com região procurada R .
2. Se T é um nó-folha, verifique todas as suas entradas e devolva todos os objetos contidos na região procurada R .

O algoritmo de busca da R-Tree faz uma avaliação de cada nó da árvore, buscando sobreposições com a área procurada. Caso elas sejam encontradas em nós não-folha, a busca continua recursivamente nas subárvores apontadas pelos nós onde aconteceram as sobreposições. Caso o nó seja uma folha, todos os objetos que se encontram na sobreposição com a área procurada são devolvidos como resultado da busca.

Figura 5.3: Representação das dimensões avaliadas pelos classificadores.

A busca na R-Tree é feita por meio da avaliação das dimensões envolvidas, verificando quais objetos pertencem à região procurada. Considerando uma R-Tree para classificadores, as dimensões da R-Tree representam as dimensões dos classificadores.

A figura 5.3, apresenta o retângulo 1 representando a área dos classificadores que avaliaram dados sobre órgãos do sistema digestivo (3-Intestino, 4-Estômago e 5-Fígado) contendo os classificadores A, B, C, G, J e M. O retângulo 2 representa a área dos classificadores que avaliaram dados

sobre doenças consideradas tumores malignos (2-Carcinoma e 3-Sarcoma) contendo os classificadores C, G e H. O retângulo 3 representa a área de interseção entre os retângulos 1 e 2 contendo os classificadores C e G, ou seja, classificadores que avaliaram, de forma simultânea, órgãos do sistema digestivo e doenças consideradas tumores malignos. A representação desse exemplo na estrutura proposta permitiria selecionar classificadores considerando as duas dimensões, para em seguida realizar o cruzamento dos objetos. Por exemplo: o cruzamento dos objetos dos classificadores do retângulo 3 poderia identificar indivíduos com algum tipo de tumor maligno em mais de um órgão do sistema digestivo. Também seria possível selecionar os classificadores que avaliaram órgãos com tumores malignos e cruzar com os que avaliaram órgãos com tumores benignos procurando indivíduos que manifestaram os dois tipos de tumores ou que manifestaram um tipo e não manifestaram o outro. Esse exemplo com apenas duas dimensões permite vislumbrar a variedade de opções existentes para seleção multidimensional dos classificadores e a quantidade de hipóteses que podem ser testadas via cruzamento de objetos classificados, indicando a necessidade de uma estrutura que ofereça flexibilidade e velocidade para seleção dos dados a serem cruzados.

Operadores para cruzamento de objetos

Os operadores para cruzamento de objetos têm um papel central dentro da estrutura para cruzamento de objetos classificados em memória, pois são os responsáveis pelo processamento dos objetos classificados. Uma vez que os classificadores são localizados na estrutura da R-Tree e os objetos, classificados por eles, são recuperados por meio de seus apontadores, os valores dos atributos, Id e R, passam a ser avaliados pelo conjunto de operadores escolhidos.

Para definir os operadores para cruzamento de objetos, alguns operadores da álgebra relacional foram estendidos e adaptados à estrutura em memória, para que fosse possível a realização de cruzamentos que explorassem as classificações dos objetos e, se necessário, pudessem até definir uma nova classificação a partir da análise realizada durante o cruzamento.

O conjunto de operadores definidos para o *CCICLO* é formado por sete operadores e pode ser estendido de acordo com a necessidade de criação de novos operadores que desempenhem novas tarefas ou que agreguem tarefas de um conjunto de operadores. Os sete operadores do *CCICLO* são: seleção, projeção, negação, diferença entre objetos classificados, união composta, interseção composta e junção composta. A escolha dos operadores foi feita procurando oferecer um conjunto básico de operações para analisar os dados em memória, e também para que pudessem ser aplicados conjuntamente oferecendo maior poder de avaliação dos dados, como acontece com os operadores da álgebra relacional, ou seja, os operadores podem ser aplicados de forma encadeada, em que o resultado de um operador é avaliado por outro. Como por exemplo, uma seleção seguida de uma

projeção ou uma interseção seguida de uma união.

Para auxiliar a exemplificação dos operadores, serão utilizados dois classificadores genéricos, C_1 e C_2 , com seus respectivos conjuntos de dados classificados formados por tuplas do tipo $\langle \text{Id} \rangle$, $\langle \text{R} \rangle$, como mostra o exemplo abaixo.

Exemplo 1

$$C_1 = \{(22,0), (29,1), (42,1), (75,0)\}$$

$$C_2 = \{(15,0), (29,0), (36,1), (75,0), (87,0)\}$$

Para efetuar a descrição dos operadores, eles foram divididos em dois grupos: operadores simples e compostos.

6.1 Operadores simples

O primeiro grupo de operadores a ser definido é o grupo dos *operadores simples*, formado pelos seguintes operadores: seleção, projeção, negação e diferença entre objetos classificados. Eles são chamados de operadores simples por possuírem apenas um operador para análise dos objetos classificados.

6.1.1 Operador seleção (σ)

O *operador seleção* (σ) tem como função gerar um subconjunto dos dados classificados por um classificador, em que os valores dos atributos, Id e R, atendem ao predicado da seleção. Como o resultado de uma seleção possui o mesmo formato dos dados de um classificador, por se tratar de um subconjunto do mesmo, ele pode ser avaliado por outros operadores ou pode ser inserido na memória, passando a representar um novo classificador.

Definição 1 – Sintaxe do Operador Seleção (σ)

$\sigma_{A \text{ op } v} C$ - Onde A é um dos atributos da tupla ($\langle \text{Id} \rangle$, $\langle R \rangle$), op é o operador do predicado, onde $\text{op} \in \{=, \neq, <, \leq, >, \geq\}$, v é um valor inteiro e C representa o conjunto de objetos analisados pelo classificador, o qual será avaliado pelo predicado ($A \text{ op } v$) de seleção.

O operador seleção (σ) pode conter mais de um predicado, permitindo que os atributos, Id e R , sejam avaliados de forma simultânea. Para tanto, basta encadear os predicados utilizando os operadores lógicos E e OU . Além disso, vale a propriedade comutativa entre os predicados do operador, como mostra o exemplo a seguir.

$$\sigma_{A \text{ op } v} (\sigma_{A' \text{ op } v'} C) = \sigma_{A' \text{ op } v'} (\sigma_{A \text{ op } v} C)$$

Como a ordem dos predicados não altera o resultado, a seleção acima pode ser escrita como:

$$\sigma_{A \text{ op } v \text{ e } A' \text{ op } v'} C$$

Exemplo 2

$$\begin{aligned} C_1 &= \{(22,0), (29,1), (42,1), (75,0)\} \\ a) \sigma_{R=1} C_1 &= \{(29,1), (42,1)\} \\ b) \sigma_{Id \geq 42} C_1 &= \{(42,1), (75,0)\} \\ c) \sigma_{Id \geq 42 \text{ e } R=1} C_1 &= \{(42,1)\} \end{aligned}$$

No item a) do exemplo acima, o operador seleção é aplicado sobre o classificador C_1 para selecionar os objetos classificados, ou seja, objetos com valor 1 para o atributo rótulo R , obtendo como resultado o conjunto contendo os identificadores 29 e 42. O item b) seleciona dados cujo identificador seja maior ou igual a 42, obtendo como resultado os identificadores 42 e 75. Por fim, o item c) seleciona os objetos com os atributos $\text{Id} \geq 42$ e $R = 1$, obtendo como resultado o conjunto unitário com o identificador 42.

6.1.2 Operador projeção (π)

O *operador projeção* (π) tem como função gerar um conjunto com tuplas contendo apenas os atributos especificados em sua sintaxe. Como o seu resultado é formado por um subconjunto dos atributos que representam um objeto classificado, ele não pode ser inserido novamente na estrutura em memória como um novo classificador.

Um exemplo de seu uso está no processo de recuperação dos dados junto ao banco de dados. Nesse caso, uma lista contendo apenas identificadores é passada para a camada da aplicação responsável pela armazenagem e recuperação de dados. A lista pode ser criada utilizando o operador projeção para remover o atributo R, deixando a lista apenas com o atributo Id.

Definição 2 – Sintaxe do operador projeção (π)

$\pi_A C$ - Onde A é o conjunto de atributos da tupla ($\langle Id \rangle$, $\langle R \rangle$) que deve ser representado no conjunto resposta do operador, e C é o classificador cujos objetos serão avaliados.

Exemplo 3

$$\begin{aligned} C_1 &= \{(22,0), (29,1), (42,1), (75,0)\} \\ a) \pi_{Id} C_1 &= \{(22), (29), (42), (75)\} \\ b) \pi_R C_1 &= \{(0), (1), (1), (0)\} \end{aligned}$$

No exemplo item a) do exemplo acima, o operador projeção projeta o atributo Id do classificador C_1 que passa a ser o único a aparecer em seu conjunto resposta. No item b), o operador foi utilizado seguindo o mesmo raciocínio, dessa vez projetando o atributo R.

6.1.3 Operador negação ($\neg$)

O *operador negação* ($\neg$) atua sobre as tuplas invertendo o valor do atributo R. Caso o atributo possua o valor 0 (zero), após a ação do operador negação seu valor é mudado para 1 (um) e vice-versa.

Definição 3 – Sintaxe do Operador ($\neg$)

$\neg C$ – Onde C é o classificador cujos objetos terão o valor do atributo rótulo (R) invertido.

Exemplo 4

$$\begin{aligned} C_1 &= \{(22,0), (29,1), (42,1), (75,0)\} \\ a) \neg C_1 &= \{(22,1), (29,0), (42,0), (75,1)\} \end{aligned}$$

No exemplo 6.1.3 item a) o operador negação foi utilizado para inverter o valor do atributo rótulo (R) dos objetos do classificador C_1 .

6.1.4 Operador diferença entre objetos classificados (-)

O *operador diferença entre objetos classificados* (-), ou somente *operador diferença* (-), avalia os objetos analisados por dois classificadores, subtraindo do conjunto de objetos do primeiro classificador representado na sua sintaxe os objetos que também estão presentes no conjunto de objetos do segundo classificador, mantendo o valor do atributo R.

Definição 4 – Sintaxe do operador diferença (-)

$C_1 - C_2$ – Onde C_1 e C_2 são os classificadores cujos objetos são verificados pelo operador diferença.

Exemplo 5

$$\begin{aligned} C_1 &= \{(22,0), (29,1), (42,1), (75,0)\} \\ C_2 &= \{(15,0), (29,0), (36,1), (75,0), (87,0)\} \\ a) C_1 - C_2 &= \{(22,0), (42,1)\} \\ b) C_2 - C_1 &= \{(15,0), (36,1), (87,0)\} \end{aligned}$$

No exemplo 6.1.4 item a) é realizada a diferença entre os classificadores C_1 e C_2 , obtendo no conjunto resposta apenas os objetos de C_1 que não estão presentes em C_2 , sem alteração do valor do atributo R. De modo semelhante o item b) apresenta a diferença entre os classificadores C_2 e C_1 .

6.2 Operadores compostos

Operador composto é definido como um operador capaz de avaliar os objetos analisados por um classificador considerando seus dois atributos, Id e R, usando para isso dois operadores: um operador de conjunto (união ($\cup$), interseção ($\cap$), junção ($\bowtie$)) responsável por avaliar o atributo Id. E um operador lógico (E ($\wedge$), Não E ($\neg$), Ou ($\vee$), Não Ou ($\neg \vee$), Ou-Exclusivo ($\otimes$), Não Ou-Exclusivo ($\neg \otimes$)) responsável pela avaliação do atributo R.

A	B	$A \cup B$	$A \cap B$	$A - B$	$B - A$
{1,2,3 }	{5,3,2}	{1,2,3,5}	{2,3}	{1}	{5}

Tabela 6.1: Funcionamento dos operadores de conjunto.

As tabelas 6.1 e 6.2 exemplificam o funcionamento de cada operador. A tabela 6.1 apresenta os operadores de conjunto avaliando dois conjuntos de números inteiros, e a tabela 6.2, os operadores lógicos avaliando combinações dos valores binários. Elas mostram que, de forma isolada o funcionamento dos operadores é igual ao tradicionalmente conhecido, o que facilita sua compreensão no contexto de cada operador.

O processamento do atributo rótulo (R) é um dos principais diferenciais da estrutura de cruzamento de classificadores em memória, pois por meio dele, o conteúdo semântico atribuído ao rótulo é avaliado, gerando resultados úteis na geração e validação de novos conhecimentos. Além disso, os operadores compostos permitem definir novas classificações para os objetos, sem a necessidade da criação de novos classificadores. As novas classificações podem ser inseridas na memória

R_1	R_2	$R_1 \wedge R_2$	$R_1 ! \wedge R_2$	$R_1 \vee R_2$	$R_1 ! \vee R_2$	$R_1 \otimes R_2$	$R_1 ! \otimes R_2$
0	0	0	1	0	1	0	1
0	1	0	1	1	0	1	0
1	0	0	1	1	0	1	0
1	1	1	0	1	0	0	1

Tabela 6.2: Funcionamento dos operadores lógicos.

como um novo classificador que representa o conhecimento obtido por meio da análise de outras classificações.

Mais informações sobre a importância dos operadores serão descritas no estudo de caso, em que são sugeridas aplicações práticas para eles. A seguir, cada operador será definido e seu funcionamento será exemplificado.

6.2.1 Operador união composta ($\cup \beta$)

O *operador união composta* ($\cup \beta$) é definido para avaliar, dois a dois, os classificadores especificados em sua sintaxe ou os resultados de operadores aplicados em sequência a um conjunto de classificadores. Como resultado, ele gera um conjunto com todos os objetos analisados pelos classificadores especificados em sua sintaxe. A diferença do seu funcionamento em relação ao operador união da álgebra relacional está na possibilidade de se avaliar o conteúdo do atributo rótulo (R), o que é obtido por meio do operador lógico especificado em sua sintaxe e cujo funcionamento é apresentado na tabela 6.2. Uma exceção ao funcionamento tradicional da união ocorre quando um objeto não está representado em ambos os conjuntos avaliados. Nesse caso, como não existem dois valores de R para serem avaliados pelo operador lógico, o objeto é inserido no conjunto resposta mantendo inalterado o valor do atributo R. Esse comportamento pode ser evitado por meio do uso de outros operadores a serem definidos posteriormente, como o operador interseção que pode selecionar somente os objetos comuns entre os conjuntos.

Definição 5 – Sintaxe do operador união composta ($\cup \beta$)

$C_1 \cup \beta C_2$ – Onde C_1 e C_2 são os classificadores verificados pelo operador união composta e $\beta \in \{\vee, !\vee, \wedge, !\wedge, \otimes, !\otimes\}$ representa um operador lógico utilizado na avaliação dos valores do atributo R.

Exemplo 6

	$C_1 = \{(22,0), (29,1), (42,1), (75,1)\}$
	$C_2 = \{(15,0), (29,0), (36,1), (75,0)\}$
a) <i>União-E:</i>	$C_1 \cup \wedge C_2 = \{(15,0), (22,0), (29,0), (36,1), (42,1), (75,0)\}$
b) <i>União-Não E:</i>	$C_1 \cup !\wedge C_2 = \{(15,0), (22,0), (29,1), (36,1), (42,1), (75,1)\}$
c) <i>União-Ou:</i>	$C_1 \cup \vee C_2 = \{(15,0), (22,0), (29,1), (36,1), (42,1), (75,1)\}$
d) <i>União-Não Ou:</i>	$C_1 \cup !\vee C_2 = \{(15,0), (22,0), (29,0), (36,1), (42,1), (75,0)\}$
e) <i>União-Ou-Exclusivo:</i>	$C_1 \cup \otimes C_2 = \{(15,0), (22,0), (29,1), (36,1), (42,1), (75,1)\}$
f) <i>União-Não Ou-Exclusivo:</i>	$C_1 \cup !\otimes C_2 = \{(15,0), (22,0), (29,0), (36,1), (42,1), (75,0)\}$

No exemplo 6.2.1 item a) o operador união composta é utilizado para unir os objetos dos classificadores C_1 e C_2 . Para tanto, ele utiliza o operador de conjunto união ($\cup$) para avaliar o atributo Id e o operador lógico E ($\wedge$) para avaliar o atributo R. O conjunto resposta do operador é formado por todos os objetos analisados pelos dois classificadores, sendo que apenas os representados em ambos tiveram os valores de seus rótulos avaliados pelo operador lógico, conforme tabela 6.2. O item b) difere do item a) na negação dos valores obtidos pelo operador lógico Não E ($!\wedge$). Os demais itens seguem a mesma linha de raciocínio.

6.2.2 Operador interseção composta ($\cap \beta$)

O *operador interseção composta* ($\cap \beta$) é definido para avaliar, dois a dois, os objetos analisados pelos classificadores especificados em sua sintaxe ou os resultados de operadores aplicados em

seqüência a um conjunto de classificadores. O seu conjunto resposta apresenta apenas os objetos formados pelos atributos, Id e R, representados em ambos os conjuntos avaliados. Sendo o valor do atributo R definido pelo operador lógico especificado.

Definição 6 – Sintaxe do Operador interseção composta ($\cap \beta$)

$C_1 \cap \beta C_2$ – Onde C_1 e C_2 são os classificadores cujos objetos são avaliados pelo operador interseção composta e $\beta \in \{\vee, !\vee, \wedge, !\wedge, \otimes, !\otimes\}$ é o operador lógico responsável pela avaliação do atributo R.

Exemplo 7

	$C_1 = \{(22,0), (29,0), (42,1), (75,1), (78,0), (92,1)\}$
	$C_2 = \{(15,0), (29,0), (36,1), (75,0), (87,0), (92,1)\}$
a) <i>Interseção-E:</i>	$C_1 \cap \wedge C_2 = \{(29,0), (75,0), (92,1)\}$
b) <i>Interseção-Não E:</i>	$C_1 \cap !\wedge C_2 = \{(29,1), (75,1), (92,0)\}$
c) <i>Interseção-Ou:</i>	$C_1 \cap \vee C_2 = \{(29,0), (75,1), (92,1)\}$
d) <i>Interseção-Não Ou:</i>	$C_1 \cap !\vee C_2 = \{(29,1), (75,0), (92,0)\}$
e) <i>Interseção-Ou-Exclusivo:</i>	$C_1 \cap \otimes C_2 = \{(29,0), (75,1), (92,0)\}$
f) <i>Interseção-Não Ou-Exclusivo:</i>	$C_1 \cap !\otimes C_2 = \{(29,1), (75,0), (92,1)\}$

O exemplo 6.2.2 item a) apresenta a utilização do operador interseção composta, no qual os atributos Id e R, dos classificadores C_1 e C_2 são avaliados, respectivamente, pelo operador de conjunto interseção e pelo operador lógico E ($\wedge$), produzindo um conjunto resposta com os objetos representados em ambos os classificadores, nesse exemplo formado pelos objetos (29,0), (75,0) e (92,1), cujos valores para o atributo R foram definidos pelo operador lógico E($\wedge$), conforme tabela 6.2. O item b) difere do item a) na negação dos valores obtidos pelo operador lógico Não E ($!\wedge$), já os demais itens seguem a mesma linha de raciocínio.

6.2.3 Operador junção composta ($\bowtie \beta$)

O *operador junção composta* ($\bowtie \beta$) é definido como o operador responsável por avaliar, dois a dois, os classificadores especificados em sua sintaxe ou os resultados de operadores aplicados em sequência a um conjunto de classificadores, gerando como resultado um conjunto com os dados representados em ambos os conjuntos avaliados e que possuem o mesmo valor para o atributo R. Sendo o valor do atributo R, no conjunto resposta, definido pelo operador lógico especificado.

Definição 7 – Sintaxe do operador junção composta ($\bowtie \beta$)

$C_1 \bowtie \beta C_2$ – Onde C_1 e C_2 são os classificadores cujos objetos são verificados pelo operador junção, e $\beta \in \{\vee, !\vee, \wedge, !\wedge, \otimes, !\otimes\}$ é um operador lógico, opcional, utilizado na avaliação dos valores do atributo R.

Exemplo 8

	$C'_1 = \{(22,0), (29,1), (42,1), (75,1), (78,0), (92,1)\}$
	$C'_2 = \{(15,0), (29,0), (36,1), (75,0), (87,0), (92,1)\}$
a) <i>Junção-E:</i>	$C_1 \bowtie \wedge C_2 = \{(92,1)\}$
b) <i>Junção-Não E:</i>	$C_1 \bowtie !\wedge C_2 = \{(92,0)\}$
c) <i>Junção-Ou:</i>	$C_1 \bowtie \vee C_2 = \{(92,1)\}$
d) <i>Junção-Não Ou:</i>	$C_1 \bowtie !\vee C_2 = \{(92,0)\}$
e) <i>Junção-Ou-Exclusivo:</i>	$C_1 \bowtie \otimes C_2 = \{(92,0)\}$
f) <i>Junção-Não Ou-Exclusivo:</i>	$C_1 \bowtie !\otimes C_2 = \{(92,1)\}$

No exemplo 6.2.3 item a) os atributos identificador e rótulo dos classificadores C_1 e C_2 são avaliados, respectivamente, pelo operador de conjunto junção ($\bowtie$) e pelo operador lógico E ($\wedge$), produzindo um conjunto resposta com os objetos representados em ambos os classificadores e com mesmo valor para o atributo rótulo, nesse exemplo formado pelo objeto com o identificador 92, e cujo valor para o atributo rótulo foi definido pelo operador lógico E ($\wedge$), conforme tabela 6.2. O

item b) difere do item a) na negação dos valores obtidos pelo operador lógico Não E ($!\wedge$), já os demais itens seguem a mesma linha de raciocínio.

6.3 Hierarquia de execução dos operadores

Os operadores do *CCICLO* possuem a mesma hierarquia de execução, não existindo precedência entre eles. Os operadores são executados da direita para esquerda conforme são especificados em uma expressão. Porém, o uso de parênteses muda a ordem de execução dos operadores. Operações entre parênteses têm prioridade de execução em relação aos demais operadores. Após sua execução, as demais operações são executadas da direita para esquerda mantendo o funcionamento normal.

Exemplo 9

$$\pi_{Id} \sigma_{R = 1} (C_1 \bowtie \wedge C_2)$$

No exemplo 6.3, inicialmente é realizada a JunçãoE ($\bowtie \wedge$) entre os classificadores C_1 e C_2 . Do resultado obtido são selecionados somente os objetos classificados. Por fim, é feita uma projeção sobre o atributo Id. Como resultado dessa expressão, é obtida uma lista com os identificadores dos objetos classificadores pela junção E dos classificadores C_1 e C_2 .

Estudo de caso

Esse capítulo apresenta um estudo de caso no qual o *CCICLO* é utilizado para avaliar relacionamentos entre classes por meio do cruzamento de dados sobre objetos classificados. Inicialmente é feita uma apresentação do contexto do estudo e, em seguida, são mostradas situações reais em que o *CCICLO* pode ser aplicado.

Os exemplos escolhidos estão voltados para análise de dados biológicos, em especial dados sobre o vírus HIV, pois representam casos encontrados no cotidiano de dois dos projetos desenvolvidos no Laboratório de Banco de Dados do Núcleo de Bioinformática da Universidade de São Paulo (BIOINFO-USP), são eles:

Integração e Análise de dados sobre o vírus HIV – Esse projeto é desenvolvido em cooperação com o Ministério da Saúde do Brasil. Ele visa desenvolver um banco de dados especializado em dados sobre o vírus HIV e integrar ferramentas de análises utilizadas na pesquisa do vírus. No restante do texto ele será chamado de banco de dados de HIV.

CAGE (*Cooperation for Analysis of Gene Expression* – Cooperação para análise de expressões gênicas). Esse trabalho é patrocinado pela FAPESP¹ e estuda a expressão de genes graças à tecnologia de microarrays e à cooperação de um grupo multidisciplinar de pesquisadores. Além de informações sobre experimentos de microarray, assim como o banco de HIV, ele tam-

¹Fundação de Amparo à Pesquisa do Estado de São Paulo.

bém apresenta um conjunto de dados clínicos que complementam as informações necessárias à pesquisa.

O banco de dados de HIV é formado por três grupos de dados: dados sobre o paciente, englobando seu comportamento e sua exposição ao vírus; dados sobre o tratamento recebido pelo paciente, englobando resultados de exames realizados e de medicamentos utilizados; dados genéticos sobre os vírus encontrados nos pacientes, englobando seqüências genéticas, mutações e a relação entre mutações e resistência a drogas.

Cada paciente possui um conjunto amplo de informações sobre seu tratamento e sobre o vírus encontrado em seu organismo buscando atender às necessidades de diferentes linhas de pesquisa sobre HIV.

Os conceitos sobre tratamento médico e genético sobre o vírus HIV apresentados nesse trabalho foram extraídos de manuais [dS01a, dS01c, dS01b] produzidos pelo Ministério da Saúde do Brasil para difundir o tratamento de pacientes infectados pelo vírus HIV.

A seguir são apresentadas as diferentes perspectivas de três profissionais envolvidos em estudos sobre o vírus HIV, mostrando de maneira concisa a especialidade de cada um deles e a necessidade de integração de seus conhecimentos.

A primeira perspectiva está relacionada aos médicos responsáveis pelo tratamento do paciente. Para que os médicos possam acompanhar a evolução do quadro do paciente e tomar as devidas atitudes são necessárias informações tais como:

- Esquema de drogas utilizado pelo paciente [dS01c] – Todos os pacientes em tratamento recebem uma terapia formada por um conjunto de medicamentos conhecido como esquema de drogas ou coquetel anti-AIDS. O esquema de drogas utilizado por um paciente pode ser alterado pelo médico, de acordo com a tolerância do paciente ao medicamento, eficácia da droga, interações medicamentosas ou novas descobertas de drogas. O conhecimento sobre os esquemas utilizados pelo paciente contribui para a compreensão da evolução de sua doença e para a tomada de decisão sobre qual o tratamento mais adequado ao paciente.

- Aderência do paciente ao tratamento [dS01c] – Por causa dos efeitos colaterais do tratamento, muitos pacientes não conseguem tomar a medicação de acordo com a prescrição médica, favorecendo o aumento da quantidade de vírus em seu organismo. A falta de aderência do paciente ao tratamento pode ser significativa para a falha do seu tratamento, pois contribui para a prevalência, em seu organismo, dos vírus resistentes ao esquema de drogas utilizado em seu tratamento.
- Controle dos resultados dos exames do paciente [dS01c] – A forma atual de acompanhar a evolução da doença é por meio de alguns exames laboratoriais. Os dois exames mais conhecidos são o de carga viral, que informa a quantidade de vírus presente no organismo e o CD4 que realiza a contagem das células T-CD4+ encontradas no organismo. As células T-CD4+ são células de defesa do organismo humano que são destruídas pelo vírus HIV. A relação entre os dois exames é simples. Quando acontece um aumento da quantidade de vírus no organismo do paciente é esperada uma redução na quantidade de células T-CD4+. Um tratamento é considerado eficiente quando a carga viral de um indivíduo diminui a ponto de se tornar indetectável e a contagem de T-CD4+ aumenta ou deixa de cair, permanecendo estável.

O segundo ponto de vista está relacionado ao geneticista. Uma de suas pesquisas busca identificar alterações genéticas no vírus, conhecidas como mutações, e seus efeitos no comportamento do vírus. As pesquisas sobre resistência a drogas necessitam de informações como:

- Mutações encontradas no vírus [dS01c] – Mutações são alterações ocorridas no código genético. Por exemplo: em determinada posição da sequência genética é esperado o nucleotídeo adenina, representada pela letra A, porém é encontrada uma timina, representada pela letra T. Algumas dessas mutações podem determinar alterações no funcionamento do vírus, como resistência ou sensibilidade a certa droga ou até mesmo inviabilizar a multiplicação ou cópia do vírus mutante.

- Relação entre mutações e a resistência a drogas [dS01c] – Do conjunto de mutações já encontradas no HIV, apenas algumas foram associadas à resistência a drogas e muitas delas ainda continuam sendo estudadas. O método utilizado para identificar a relação entre a mutação e resistência às drogas é conhecido como fenotipagem. Nesse método, os vírus com mutação são clonados e cultivados em diferentes situações, tais como presença e ausência da droga estudada, para constatar se eles são mesmo resistentes. A fenotipagem ainda é um método caro e demorado, tornando valiosas indicações fundamentadas sobre possíveis relações entre mutações e resistência às drogas. Conhecer essas relações permite melhorar o tratamento de pacientes com vírus resistentes e aumentar o conhecimento sobre a evolução e o comportamento das mutações no vírus.

O terceiro ponto de vista está relacionado ao biólogo. Uma de suas linhas de pesquisa estuda a forma de evolução ou filogenia do vírus. A filogenia do vírus busca identificar a origem do vírus e seu padrão evolutivo. Com esse conhecimento é possível acompanhar quais são os subtipos de vírus predominantes nas epidemias de diferentes regiões geográficas e como tem ocorrido a migração dos subtipos.

As abordagens descritas formam apenas um subconjunto dos dados e profissionais envolvidos nos estudos sobre o HIV e mostram a variedade das informações disponíveis e das relações entre elas, em que algumas já são conhecidas e outras ainda necessitam ser estudadas. Além disso, essas abordagens mostram como o conhecimento necessário para o estudo de cada uma das áreas é abrangente e distribuído entre diversas especialidades, demandando a integração tanto dos dados gerados quanto do conhecimento envolvido nas análises. Esse cenário mostra-se apropriado para a utilização do *CCICLO* justamente pela diversidade de informações e necessidade de integração de conhecimento. Para tanto, é necessária a definição de alguns classificadores para que classes de pacientes possam ser criadas e as relações entre elas estudadas pelo cruzamento de seus objetos.

7.1 Classificadores para o banco de dados de HIV

Para o uso no estudo de caso são apresentados alguns classificadores úteis para a análise do banco de dados de HIV. Eles são responsáveis por apresentar diferentes perspectivas sobre os dados, focando contextos específicos acerca dos pacientes e seus tratamentos.

Segue uma relação de oito classificadores para análise do banco de dados de HIV.

- C_1 - Pacientes com diagnóstico de AIDS – A presença do vírus HIV no organismo do paciente não é suficiente para diagnosticar a AIDS. O diagnóstico de AIDS [dS01a, dS01c, dS01b] é fundamentado em critérios estabelecidos pela comunidade científica que atribui pontuações para cada sintoma ou doença associado à AIDS e define uma pontuação mínima para diagnosticar a doença. Esse classificador rotula com valor um os pacientes infectados pelo vírus HIV e com diagnóstico de AIDS, e rotula com o valor zero os pacientes infectados pelo vírus HIV e sem o diagnóstico de AIDS. Lembrando que os objetos com valor um para o rótulo são chamados de classificados e os com valor zero de não classificados.
- C_2 – Pacientes aderentes ao tratamento – Um paciente é considerado aderente ao tratamento quando ele toma o remédio na quantidade e periodicidade correta. Esse classificador rotula com valor um os pacientes aderentes ao tratamento, e com valor zero os pacientes não aderentes ao tratamento.
- C_3 – Pacientes com falha terapêutica – A falha terapêutica indica que o tratamento do paciente não está surtindo o efeito esperado. Segundo [dS01a], ela é definida como a ocorrência de deterioração clínica e/ou piora dos parâmetros laboratoriais. Esse classificador rotula com valor um os pacientes com falha terapêutica, e como valor zero os pacientes com tratamento bem sucedido.
- C_4 – Pacientes virgens de tratamento - Pacientes virgens de tratamento são pacientes que nunca receberam nenhum tipo de medicação para o tratamento da AIDS. Esse classificador rotula

com valor um os pacientes que nunca fizeram uso de medicamentos para o tratamento de AIDS, e com valor zero os pacientes que já utilizaram medicamentos.

- C_5 – Pacientes cujos vírus possuem mutações. Esse classificador rotula com valor um os pacientes cujos vírus apresentaram mutações em sua sequência genética, e com valor zero os pacientes cujos vírus não apresentaram mutação.
- C_6 – Pacientes cujos vírus possuem mutações associadas à resistência a drogas. Algumas mutações foram identificadas como responsáveis por tornar o vírus resistente ou susceptível a algumas drogas. Esse classificador pode ser desenvolvido para identificar pacientes resistentes a drogas específicas como o AZT, porém nesse exemplo ele rotula com valor um os pacientes cujos vírus apresentem mutações associadas à resistência a qualquer droga, e com valor zero os pacientes cujos vírus não apresentem mutações relacionadas à resistência a alguma droga.
- C_7 – Pacientes que utilizaram um determinado esquema de drogas. Esse classificador rotula com valor um os pacientes que já utilizaram um determinado esquema de drogas, e com valor zero os pacientes que nunca fizeram uso desse esquema de drogas.
- C_8 – Pacientes que estão utilizando um determinado esquema de drogas. Esse classificador atenta para o esquema de drogas em uso, rotulando com valor um os pacientes que estão utilizando atualmente um determinado esquema de drogas, e com valor zero os pacientes que atualmente não estão usando esse esquema de drogas.

O processo de classificação envolve a análise de um conjunto de dados e a aplicação de regras que identificam se um objeto pertence ou não à classe definida pelo classificador. O uso dos classificadores proporciona uma avaliação padronizada das informações em vez de uma avaliação pessoal de cada profissional. Com isso, é possível criar um padrão para as classificações e refazê-las sempre que necessário. Em um banco de dados no qual as classificações não são obtidas por um classificador e os dados avaliados durante a classificação não estão disponíveis, o processo de reavaliação e

confirmação das classificações se torna difícil, e em alguns casos até mesmo impossível. Apesar da complexidade do processo de classificação, os classificadores foram apresentados de forma simples e sem entrar em detalhes específicos sobre as regras e sobre o processo para a obtenção de cada classificação. Isso mostra o nível de abstração que pode ser utilizado para o cruzamento de objetos classificados. Processos complexos podem ser descritos em poucas palavras, permitindo abstrair a complexidade de cada classificação e integrar com mais facilidade o conhecimento de diferentes especialistas representado em cada classificador. Dessa maneira, hipóteses sobre o relacionamento entre classes podem englobar um maior número de classes sem inviabilizar a compreensão do processo como um todo.

7.2 Representação dos dados em memória

A forma de representação de dados usada pelo *CCICLO* objetiva manter, em memória, a totalidade dos dados usados no cruzamento ou a maior parte deles, reduzindo a necessidade de troca de dados entre memória e disco. Para alcançar e manter esse objetivo mesmo com o aumento do volume de dados analisados, o *CCICLO* usa uma representação resumida dos dados armazenados em disco. Nela são representados apenas dados com grande valor semântico para o cruzamento dos objetos classificados. Conforme mostra a figura 7.1, o classificador é representado por seu identificador, pelo vetor de características e pelas informações sobre o processo de classificação como data e hora do processamento e identificação da base classificada. O objeto analisado é representado pelo seu identificador e pela classificação recebida por ele. Dessa maneira, é obtida uma boa redução do espaço necessário para a representação dos dados, permitindo que todos ou a maioria dos objetos classificados sejam representados em memória.

A representação resumida de dados não é uma novidade e pode ser encontrada em outras soluções, como ferramentas OLAP [CD97b]. Porém, ela é um ponto importante dentro da estrutura do *CCICLO*.

A opção por representar somente dados com grande valor semântico para o cruzamento de dados

Figura 7.1: Objeto complexo - classificador.

introduz a abstração da complexidade do processo de classificação na qual o *CCICLO* se alicerça. O processo de classificação e suas peculiaridades são analisados e trabalhados durante o projeto dos classificadores, e ao final desse processo, o resultado gerado por um classificador é considerado aceitável e o processo em si pode ser abstraído, facilitando a compreensão da classificação e a inferência das relações entre as classes.

O enfoque de representação de dados em memória do *CCICLO* é adequado ao contexto do estudo do HIV, no qual as informações sobre os pacientes e seus tratamentos são abrangentes e crescem constantemente, tanto pelo avanço da epidemia quanto pelo acompanhamento feito ao paciente, exigindo um ambiente capaz de manter a eficiência e simplicidade mesmo com o aumento do volume de dados.

O banco de dados de HIV atualmente conta com 319 pacientes cadastrados e com informações amplas e integradas sobre o comportamento do paciente, dados epidemiológicos, uso de medicação, exames laboratoriais e, principalmente, as seqüências genéticas dos vírus de cada paciente e as informações sobre resistência a drogas. Essa vasta representação de dados motivou um convênio com o Ministério da Saúde do Brasil que possui um amplo e reconhecido programa de distribuição

de drogas para pacientes HIV positivo. Além desse programa, atualmente o Ministério da Saúde está realizando um programa, chamado RENAGENO ² [dS01a], para estudar as seqüências genéticas dos vírus de pacientes com falha terapêutica, com o objetivo de identificar possíveis mutações que confirmam ao vírus resistência a determinadas drogas e com isso proporcionar um tratamento mais adequado ao paciente.

Com esse projeto, espera-se aumentar o número de pacientes cadastrados de 319 para aproximadamente 100.000 pacientes, que representa o número de pacientes cadastrados e que recebem medicamentos do Ministério da Saúde.

Considerando o constante crescimento dos bancos de dados atuais e a necessidade de adequação da solução proposta a esses crescimentos, a seguir é apresentada uma comparação entre as necessidades de espaço em disco e em memória de acordo com o crescimento do banco de dados de HIV. Essa comparação pretende mostrar a adequação do *CCICLO* à análise de grandes volumes de dados.

7.2.1 Representação em disco do banco de HIV

A quantidade de espaço em disco necessária para armazenar informações sobre um paciente é variável, pois alguns pacientes possuem mais informações sobre o tratamento do que outros. Por isso, para ter um valor aproximado do espaço necessário para armazenar informações acerca de um paciente foi escolhida uma quantidade média para suas informações variáveis como mostra a tabela 7.1. A escolha desses valores fundamentou-se na quantidade de dados encontrada entre os 319 primeiros pacientes cadastrados e na expectativa de possuir uma média de cinco exames de cada paciente e seqüências de duas regiões do genoma do vírus, com tamanho individual aproximado de 1000 de bases ou nucleotídeos.

Após a definição da quantidade média de ocorrências para cada paciente, foi realizado um cálculo de espaço necessário em disco, usando como referência os atributos de cada tabela e

²Rede Nacional de Genotipagem.

<i>Tipo de Informação</i>	<i>Ocorrências</i>
Exames Laboratoriais	5
Esquema de drogas em uso	1
Esquemas de drogas utilizados	5
Seqüências	2

Tabela 7.1: Número de ocorrências para os dados variáveis do paciente.

o espaço consumido por seu tipo de dados, segundo o gerenciador de banco de dados comercial Sybase em sua versão 11. Não foi considerado o espaço consumido por estruturas de gerenciamento do banco de dados, pois as mesmas não necessitam ser representadas em memória.

Para tornar o cálculo mais preciso, as tabelas do banco de dados foram divididas em dois grupos. O primeiro formado pelas tabelas cujo tamanho apresenta pouco ou nenhum crescimento em relação ao número de pacientes cadastrados. Nesse grupo se enquadra a tabela com os nomes das drogas utilizadas no tratamento do paciente, pois ela só cresce quando uma nova droga é disponibilizada no mercado. E isso independe da quantidade de pacientes cadastrados. O segundo grupo é formado pelas tabelas que crescem de acordo com a inclusão de novos pacientes. Como é o caso da tabela com os esquemas de drogas em uso, pois a cada novo paciente inserido no banco, novas informações sobre o esquema de drogas em uso também são incluídas.

<i>Tabelas do Primeiro Grupo</i>	<i>Tabelas do Segundo Grupo</i>
1,08 Mbyte	0,21 Mbyte

Tabela 7.2: Comparação entre o espaço ocupado pelas tabelas do primeiro e segundo grupos.

Feita essa separação e a comparação do espaço ocupado por cada uma das tabelas foi decidido somente considerar para o cálculo do espaço ocupado por cada paciente as tabelas do primeiro grupo, pois o espaço ocupado pelas tabelas do segundo grupo é pequeno, como mostra a tabela

7.2, e se torna pouco expressivo à medida que cresce o número de pacientes cadastrados. Para os 319 pacientes as tabelas do segundo grupo representam aproximadamente 16% do espaço total ocupado em disco. Como o espaço ocupado pelas tabelas não aumenta com o cadastramento de novos pacientes, à medida que mais pacientes são inseridos no banco, essa proporção diminui e se torna pouco relevante. Após a soma dos espaços ocupados por cada tabela do banco de dados de HIV, foi obtida a estimativa de 3720 bytes para cada novo paciente cadastrado.

Para verificar a qualidade da estimativa proposta, a tabela 7.3 apresenta uma comparação entre a estimativa de espaço necessário para o armazenamento em disco e o espaço atualmente ocupado pelos 319 pacientes cadastrados. A diferença entre o espaço estimado e o espaço atual é pequena e pode ser explicada, pois o valor da estimativa é um valor médio e aproximado.

<i>Número de Pacientes</i>	<i>Estimativa de espaço</i>	<i>Espaço Atual</i>
1	3720 bytes	Variável
319	1,14 Mbytes	1,08 Mbyte

Tabela 7.3: Comparação entre o espaço estimado e o espaço em uso.

7.2.2 Representação no CCICLO do banco de HIV

Para o cálculo de espaço para representação dos dados sobre HIV no *CCICLO*, foi considerado o espaço necessário para a representação do objeto complexo (figura 7.1). O espaço ocupado pela R-Tree não foi considerado, pois apresenta pequeno impacto no total do espaço necessário, principalmente quando são representadas classificações com grande número de objetos e poucos classificadores.

A figura 7.1 apresenta o formato do objeto classificador e a figura 7.2 apresenta o espaço ocupado por cada um de seus atributos segundo a linguagem de programação Java. Os atributos do classificador são do tipo inteiro e real (float) como mostra a figura 7.2, em Java, ambos consomem

quatro bytes para sua representação.

<i>Tipo de Dado</i>	<i>Espaço em bytes</i>	<i>Faixa de Representação de Valores</i>
Inteiro	4	-2.147.483.648 a 2.147.483.648
Real(float)	4	$\pm 3.40282347E+38F$

Tabela 7.4: Consumo de espaço dos tipos de dados em Java 1.3.

Objeto complexo – classificador:

$$\begin{aligned}
 \{(<OId, OrigemDados, DataHora, Pt0, Pt1>), \Rightarrow & 4 + 4 + 4 + 4 + 4 = 20 \quad \text{bytes} \\
 (<Dim1, Val1>, \dots, <DimN, ValN>), \Rightarrow & (4 + 4) * D = 8 * D \text{ bytes} \\
 (<Id1, Prox>, \dots, <IdN, Prox>), \Rightarrow & (4 + 4) * M = 8 * M \text{ bytes} \\
 (<Id1, Prox>, \dots, <IdN, Prox>)\} \Rightarrow & (4 + 4) * N = 8 * N \text{ bytes}
 \end{aligned}$$

Figura 7.2: Cálculo de valores.

A figura 7.2 mostra que a primeira parte do objeto classificador formada por $<OId, OrigemDados, DataHora, Pt0, Pt1>$ necessita de 20 bytes para sua representação e a segunda parte formada por $<Dim1, Val1>, \dots, <DimN, ValN>$ necessita de $8 * D$ bytes, onde D é o número de dimensões representadas. Como as duas primeiras partes do classificador não variam com o número de objetos representados, o espaço ocupado por elas não apresenta grande impacto no total de espaço consumido, principalmente quando o número de objetos representados é da ordem de milhares de objetos. Por esse motivo, as duas primeiras partes do objeto não são consideradas para o cálculo de espaço consumido. Assim, o espaço consumido para representar um classificador e seus dados em memória usando a estrutura do *CCICLO* pode ser calculado pela fórmula:

$$\boxed{\text{Espaço} = 8(M + N)}$$

Onde o número oito representa os quatro bytes necessários para representar o identificador do classificador mais os quatro bytes para a representação do ponteiro para o objeto seguinte. E as variáveis M e N representam, respectivamente, o número de objetos não classificados e o número de objetos classificados.

Com base na estimativa de espaço necessário para a inclusão de um novo paciente no banco de dados de HIV e na fórmula para o cálculo de espaço em memória para a representação de cada paciente, a tabela 7.5 apresenta um demonstrativo com o espaço necessário para representar diferentes números de pacientes tanto em disco quanto em memória utilizando a estrutura do *CCICLO*.

<i>Número de Pacientes</i>	<i>Espaço em Disco</i>	<i>Espaço em Memória</i>
1	3.720 bytes (3,65 Kbytes)	8 bytes
319 (Atual)	1.159 Kbytes (1,14 Mbytes)	2,5 Kbytes
1 milhão	3.548 Mbytes (3,47 Gbytes)	7,6 Mbytes
3 milhões	10.643 Mbytes (10,4 Gbytes)	22,9 Mbytes
32 milhões	113.525 Mbytes (110,9 Gbytes)	244,2 Mbytes

Tabela 7.5: Demonstrativo de espaço.

Considerando as diferentes classificações que um objeto pode receber, é necessário estimar não somente o espaço para armazenar e processar em memória uma classificação, mas também prever a utilização da memória para representar diferentes classificações de objetos, tornando o cruzamento uma tarefa mais significativa. Com essa perspectiva, a tabela 7.6 apresenta o espaço necessário em memória para representação de diversas classificações do banco de dados de HIV, considerando que o mesmo possua 3 milhões de pacientes que ocupam 7,6 Mbytes de memória para cada classificação representada.

Caso seja necessário representar 100 classificações desse banco, e considerando que todos os pacientes estejam representados nelas, será necessário um espaço equivalente a 760 Mbytes

de memória RAM. Essa quantidade pode ser considerada baixa, tendo em vista que uma boa parte dos computadores caseiros atuais possui a capacidade de gerenciar mais de 1,5 Gbyte de memória RAM. Se for preciso processar 1.000 classificações desse banco de dados seriam necessários 7,6 Gbytes de memória RAM. Ainda assim, esse valor não foge aos recursos de processamento disponíveis em computadores profissionais utilizados em universidades e grandes empresas.

<i>Número de Pacientes</i>	<i>Número de Classificações</i>	<i>Espaço em Memória</i>
1 milhão	1	7,6 Mbytes
1 milhão	100	760 Mbytes
1 milhão	1.000	7,6 Gbytes

Tabela 7.6: Demonstrativo de espaço.

Os números da tabela 7.6 juntamente com os dados do relatório [dS00], que estima em 597.443 o número de indivíduos entre 15 e 49 anos infectados pelo HIV no Brasil, mostram a adequação do *CCICLO* para a análise do banco de dados de HIV.

Esse exemplo demonstra a vocação do *CCICLO* para representar objetos classificados em memória oferecendo a capacidade de processamento de grande volume de dados por meio de uma representação simplificada dos mesmos.

7.3 Seleção multidimensional de classificadores para cruzamento

A possibilidade de avaliar um conjunto de classificadores exige uma forma flexível para selecioná-los. O *CCICLO*, por meio da representação do vetor de característica dos classificadores em uma R-Tree, permite usar os atributos do vetor de característica para a seleção dos classificadores. E se os valores avaliados em cada característica forem representados, eles também podem ser utilizados

para compor a condição de seleção dos classificadores, ou seja, em vez de informar um a um cada classificador e seus respectivos dados a serem cruzados, eles podem ser escolhidos usando as semelhanças entre eles.

Por exemplo: caso seja necessário avaliar a evolução das classificações realizadas por um classificador em determinada base de dados, é preciso selecionar todas as classificações feitas por ele dentro do período analisado. Realizar essa seleção de forma pontual implica informar uma a uma as classificações realizadas. No *CCICLO*, a seleção poderia ser feita informando o classificador, a base desejada e o período de classificações. Assim, todas as classificações feitas pelo classificador na base específica dentro do período informado seriam recuperadas diretamente pela R-Tree, sem a necessidade de conhecer previamente quantas e quais foram as classificações realizadas.

Outro exemplo da flexibilidade oferecida pela escolha multidimensional pode ser visto quando se deseja avaliar, no banco de dados de HIV, os classificadores que avaliaram o esquema de drogas prescrito ao paciente. Em vez de solicitar pontualmente a busca pelos classificadores C_7 e C_8 , pode-se solicitar a recuperação de todos os classificadores que avaliaram o esquema de drogas utilizado pelo paciente. Com a mesma linha de raciocínio, é possível solicitar todos os classificadores aplicados sobre determinada base de dados, ou ainda, uma busca considerando três dimensões: como recuperar os classificadores que avaliaram as seqüências genéticas de determinada região do genoma do vírus, as mutações que conferem resistência a drogas e as amostras retiradas de certa parte do corpo do paciente. A escolha multidimensional não se limita a três dimensões, mas sim à quantidade de características representadas e das características que se deseja avaliar.

A seleção multidimensional dos classificadores e de seus objetos para o cruzamento é uma das funcionalidades importantes do *CCICLO*, pois permite agregar mais valor semântico ao dados. Cada condição especificada representa um conhecimento a mais sobre os dados obtidos. Apesar da flexibilidade da seleção multidimensional, a seleção pontual também pode ser realizada no *CCICLO*.

Caso exista algum classificador que atenda às condições de busca especificadas, eles são apresentados para que o pesquisador decida quais operações serão realizadas para o cruzamento dos

objetos. Essa interação entre pesquisador e programa é defendida em trabalhos como [AEK00] e ajuda a refinar as análises, reduzindo a quantidade de resultados espúrios gerados por processos totalmente automatizados. Além disso, ela facilita a compreensão dos resultados obtidos, pois o próprio pesquisador foi quem escolheu os classificadores e os operadores utilizados.

7.4 Uso dos operadores

Os operadores adaptados para o *CCICLO* são os responsáveis pelo cruzamento dos objetos classificados. Após a recuperação dos classificadores desejados, as operações a serem realizadas são definidas pelo uso dos operadores que executarão o cruzamento dos objetos classificados de acordo com suas especificações.

O banco de dados de HIV permite analisar o paciente sob diferentes perspectivas, entre elas: o acompanhamento do tratamento recebido pelo paciente, aderência do paciente ao tratamento, análise das mutações encontradas nas seqüências do vírus e estudos filogenéticos das seqüências.

Figura 7.3: Conjunto de classificadores usados no estudo de caso.

Para exemplificar o uso dos operadores do *CCICLO*, são apresentadas situações práticas relacionadas ao banco de dados de HIV, em que eles podem ser utilizados para realizar o cruzamento de objetos classificados como forma de apoio ao processo de descoberta de conhecimento no banco

de dados. Para simplificar a apresentação dos exemplos, será considerado que os classificadores já foram escolhidos e seus dados recuperados para o cruzamento conforme o exemplo na seção 7.3. A figura 7.3 apresenta os classificadores que serão utilizados a seguir, juntamente com as classificações geradas por eles para um conjunto de pacientes representados por seus identificadores (Id).

7.4.1 Operador seleção (σ)

O operador seleção (σ) permite escolher os objetos a serem cruzados através da avaliação de seus atributos ($\langle \text{Id} \rangle$, $\langle R \rangle$) e como apresentado no capítulo 6 (ver seção 6.1.1 na página 56), sua sintaxe é :

$$\sigma_{A \text{ op } v} C$$

A seleção dos objetos classificados pode ser feita avaliando os atributos em conjunto ou cada um separadamente. Caso a seleção ocorra utilizando o atributo R, o resultado da operação será uma das classes criadas pelo classificador. Se a condição do operador seleção for do tipo $R = 0$ ou $R \neq 1$, serão selecionados os objetos classificados. Se a condição for $R = 1$ ou $R \neq 0$, então os objetos não classificados serão selecionados. Quando a seleção é baseada somente no atributo Id, os objetos são selecionados se os valores de seus identificadores estiverem na faixa especificada na condição, independente de sua classificação. A seleção avaliando ambos os atributos permite escolher determinados objetos dentro de uma classe, ou seja, a condição com atributo R indica qual a classe de objetos deve ser selecionada e a condição com o atributo Id informa quais os objetos dentro da classe devem ser selecionados.

Como exemplo prático, é utilizado o classificador C_3 que classifica os pacientes com falha terapêutica. Existem pesquisas que desejam utilizar somente os pacientes sem falha terapêutica, buscando identificar os motivos da eficiência do tratamento. Já outras pesquisas trabalham com pacientes com falha terapêutica, tentando identificar a causa da falha para propor tratamentos alternativos. Usando o *CCICLO* a escolha desses pacientes poderia ser feita aplicando o operador

seleção no classificador C_3 . A figura 7.4 apresenta a sintaxe para seleção dos pacientes com falha terapêutica e o diagrama da seleção:

Figura 7.4: Seleção de pacientes com falha terapêutica.

A sintaxe para seleção dos pacientes com tratamento efetivo e o diagrama da seleção são apresentados na figura 7.5.

Figura 7.5: Seleção de pacientes com tratamento efetivo.

A figura 7.6 apresenta a sintaxe e o diagrama da seleção dos pacientes com falha terapêutica cujos identificadores são maiores que 5.

Figura 7.6: Seleção usando os atributos Id e R.

7.4.2 Operador projeção (π)

O operador projeção (π) permite escolher qual dos atributos do objeto classificado, Id ou R, será representado no conjunto resposta e como apresentado no capítulo 6 (ver seção 6.1.2 página 58), sua sintaxe é:

$$\pi_A C$$

A finalidade prática do operador é permitir escolher com qual dos atributos se deseja trabalhar e descartar o atributo desnecessário. Como exemplo prático, é usada a recuperação de dados armazenados em disco.

Após a realização de cruzamentos com os objetos classificados, pode ser necessário recuperar informações complementares sobre os objetos analisados. Para recuperar os atributos de cada objeto, uma lista com os identificadores dos objetos deve ser enviada para a aplicação responsável pela comunicação com o disco. A figura 7.7 apresenta a sintaxe, usando o operador projeção para

a criação de uma lista formada pelos identificadores dos objetos avaliados pelo classificador C_3 . Ela mostra também um diagrama da execução do operador.

Figura 7.7: Projeção dos identificadores dos objetos de C_3 .

A especificação do atributo Id na condição do operador indica que esse atributo deve ser o único representado no conjunto resposta, levando ao descarte do atributo R. Nesse caso, o resultado seria uma lista com os identificadores dos objetos classificados e dos não classificados. Para escolher uma lista com somente uma das classes, por exemplo, a dos pacientes com falha terapêutica, o operador seleção deve ser utilizado em conjunto com o operador projeção, como mostra a figura 7.8.

Figura 7.8: Projeção dos identificadores com valor 1 para o rótulo.

Na operação conjunta dos operadores projeção e seleção, a expressão entre parênteses é executada obtendo os objetos classificados e, em seguida, o operador projeção gera a lista contendo somente os identificadores.

7.4.3 Operador negação ($\neg$)

Com a função de inverter o valor do atributo rótulo dos objetos classificados, o operador negação ($\neg$) pode ser utilizado para mudar a classificação de uma classe de objetos de classificados para não classificados ou o contrário. Essa operação permite mudar o valor semântico do classificador sem mudar suas regras. Como apresentado no capítulo 6 (ver seção 6.1.3 na página 58), sua sintaxe é:

$$\neg C$$

Para exemplificar o seu uso prático, considere uma pesquisa que deseje estudar pacientes não aderentes ao tratamento. Para obter esses pacientes utilizando o classificador C_2 , que classifica pacientes aderentes ao tratamento, é necessário trabalhar com a classe dos objetos não classificados pelo classificador, representados pelos objetos com rótulo igual a zero. Esse tipo de situação pode causar confusões, portanto seria melhor trabalhar com um classificador que classificasse os pacientes não aderentes ao tratamento. No *CCICLO* isso pode ser obtido sem a necessidade de

escrever um novo classificador, bastando utilizar o operador negação aplicado ao classificador C_2 , como mostra a figura 7.9.

Figura 7.9: Negação do classificador C_2

Nesse exemplo, os pacientes aderentes ao tratamento passariam a ter rótulos com valor zero, formando a classe dos objetos não classificados, e os pacientes não aderentes passariam a formar a classe de objetos classificados, representados pelo rótulo com valor um. Essa mudança de valores do rótulos, além de uma mudança semântica, pode ser necessária para adequar o valor do rótulo aos operadores lógicos disponíveis nos operadores compostos. Por exemplo: o uso do operador lógico E ($\wedge$), para rótulos com valor zero e um, produz como resultado o valor zero. Caso os rótulos possuam valor um, o operador produzirá como resultado o valor um. Dependendo do comportamento desejado, o operador negação pode ser usado para alterar o valor do rótulo, adequando-o ao operador lógico.

7.4.4 Operador diferença entre objetos classificados (-)

O operador diferença entre objetos classificados (-) ou somente operador diferença, permite remover de uma classe, objetos que estão representados em outras classes e, como apresentado no capítulo 6 (ver seção 6.1.4 na página 59), sua sintaxe é:

$$C_1 - C_2$$

Uma análise que procure avaliar relação entre mutações no vírus HIV, resistência a drogas e falha terapêutica pode, em um primeiro momento, não desejar analisar os pacientes não aderentes ao tratamento, por se tratar de uma classe de pacientes na qual o motivo da falha terapêutica está, provavelmente, mais vinculado ao uso incorreto dos medicamentos do que a uma mutação no vírus que cause resistência à droga.

A exclusão dos pacientes não aderentes ao tratamento do conjunto de pacientes com falha terapêutica pode ser feita utilizando o operador diferença (-) aplicado aos classificadores C_3 e C_2 . Entretanto, antes de usar o operador diferença, é necessário escolher, em cada um dos classificadores, as classes de objetos a serem cruzadas. Do classificador C_3 , a classe desejada é a dos objetos classificados, que corresponde aos pacientes com falha terapêutica, e do classificador C_2 deve ser escolhida a classe dos objetos não classificados, que corresponde aos pacientes não aderentes ao tratamento. Para tornar clara a aplicação do operador diferença no passo final do exemplo, os resultados intermediários, obtidos durante a seleção das classes dos classificadores C_3 e C_2 , serão atribuídos aos operadores C'_3 e C'_2 , respectivamente, como mostra a sintaxe a seguir.

$$C'_2 = \sigma_{R=0} C_2$$

$$C'_3 = \sigma_{R=1} C_3$$

Utilizando C'_3 e C'_2 é possível abstrair as seleções anteriores, bastando utilizar C'_3 , representando os pacientes com falha terapêutica, e C'_2 para indicar os pacientes não aderentes ao tratamento. Dessa maneira, a separação dos pacientes não aderentes ao tratamento dos que possuem falha

terapêutica pode ser feita pela subtração do conjunto de objetos de C'_2 do conjunto de objetos de C'_3 , como mostra a figura 7.10.

Figura 7.10: Exemplo do operador diferença entre classificadores.

Se não fosse feita a separação do processo de seleção e diferença, a sintaxe do operador seria:

$$(\sigma_{R=1} C_3) - (\sigma_{R=0} C_2)$$

7.4.5 Operador união composta ($\cup \beta$)

O operador união composta ($\cup \beta$) é utilizado para unir dois conjuntos de objetos classificados. Além disso, o operador lógico agregado a ele permite processar os rótulos de cada objeto para a definição do valor de rótulo que o objeto terá ao final da operação. Como resultado final, o operador apresenta um conjunto formado por todos os objetos de cada classificador. Os valores dos rótulos são tratados de acordo com a seguinte regra: o valor do atributo rótulo é mantido inalterado, preservando a classificação do objeto, quando o mesmo está representado em somente um dos classificadores. Caso o objeto esteja representado em ambos os classificadores, o operador lógico especificado em sua sintaxe é usado para definir o valor do atributo rótulo no conjunto resposta.

Como apresentado no capítulo 6 (ver seção 6.2.1 na página 61), sua sintaxe é:

$$C_1 \cup \beta C_2$$

Uma das aplicações da união de conjuntos é a reunião de subconjuntos. No caso do banco de dados de HIV, por causa das diferentes abordagens de cada pesquisa, os classificadores podem ser aplicados em subconjuntos dos dados de pacientes levando em consideração uma característica específica do objeto, como sexo, idade, região onde mora, região onde foi infectado, tipo de tratamento, estágio da doença.

R_1	R_2	$R_1 \wedge R_2$	$R_1 \neg \wedge R_2$	$R_1 \vee R_2$	$R_1 \neg \vee R_2$	$R_1 \otimes R_2$	$R_1 \neg \otimes R_2$
0	0	0	1	0	1	0	1
0	1	0	1	1	0	1	0
1	0	0	1	1	0	1	0
1	1	1	0	1	0	0	1

Tabela 7.7: Funcionamento dos operadores lógicos.

Como exemplo da utilização do operador união composta, considere que o classificador C_1 foi utilizado para classificar dois bancos de dados de regiões diferentes, um com pacientes da cidade de São Paulo e outro com pacientes da cidade do Rio de Janeiro, representados respectivamente por $C_1(SP)$ e $C_1(RJ)$, e que será necessário integrar esses dados para uma análise conjunta. Em razão da proximidade e do grande fluxo de pessoas entre essas cidades, existe a possibilidade de que alguns pacientes estejam representados nos dois grupos. Além disso, as informações registradas sobre esses pacientes pode variar entre as cidades. Por exemplo: o paciente pode ter realizado alguns exames em São Paulo e os resultados não foram enviados para o Rio de Janeiro. Esse tipo de diferença entre os dados pode gerar uma divergência entre as classificações obtidas em cada cidade, apesar de usar o mesmo classificador.

Para resolver esse impasse, é necessário definir o procedimento usado para avaliar diferentes classificações, ou seja, definir qual a classificação que o paciente receberá quando as classificações

forem iguais e quando forem diferentes. Após essa definição, a tabela 7.7 deve ser consultada em busca do operador lógico que produz o comportamento desejado em cada um dos casos.

No caso do classificador C_1 que classifica pacientes com diagnóstico de AIDS, as classificações iguais devem ser mantidas, ou seja, se o valor do rótulo do paciente for zero em ambas as cidades, ele deve permanecer zero. O mesmo deve ocorrer se o valor for um em ambas as cidades. Quando ocorrer diferença entre as classificações recebidas em cada cidade, deve ser mantido o valor um para o rótulo do paciente. Isso significa que, se o paciente possui o diagnóstico de AIDS em alguma cidade, essa informação deve ser mantida. Simplificando, o comportamento desejado é manter o valor um para o rótulo sempre que ele aparecer em alguma das classificações. Caso contrário, o valor do rótulo deve ser zero. Como mostra a tabela 7.7, esse comportamento é produzido pelo operador OU ($\vee$), assim, esse operador deve ser escolhido como o operador lógico do operador união composta formando o operador união-ou ($\cup \vee$). A sintaxe dessa expressão é apresentada na figura 7.11:

Figura 7.11: Exemplo de união composta de classificadores.

O operador união composta também pode ser utilizado para reunir classificadores diferentes,

bastando utilizá-lo de acordo com os resultados semânticos desejados.

7.4.6 Operador interseção composta ($\cap \beta$)

O operador interseção composta ($\cap \beta$) cruza os dados de dois classificadores identificando os objetos comuns a eles. Além disso, o operador lógico, definido em sua sintaxe, realiza o processamento dos valores dos rótulos definindo as classes dos objetos classificados e dos não classificados. Como resultado final, o operador apresenta um conjunto formado somente pelos objetos encontrados na interseção entre os dois conjuntos, classificados de acordo com o operador lógico especificado.

Como apresentado no capítulo 6 (ver seção 6.2.2 na página 62), sua sintaxe é:

$$C_1 \cap \beta C_2$$

Para exemplificar o uso do operador no banco de dados, serão usados os classificadores C_2 (Pacientes aderentes ao tratamento), C_3 (Pacientes com falha terapêutica), C_6 (Pacientes cujas mutações no vírus conferem resistência a drogas). Esse exemplo procura pacientes com falha terapêutica, aderentes ao tratamento e com vírus cujas mutações não são conhecidas como causadoras de resistência a drogas. Com esse perfil, o paciente não se enquadra nos casos comuns de falha terapêutica, o que pode indicar falha no processo de classificação, falha na coleta de dados ou até mesmo uma nova mutação que confere resistência à droga. Para identificar tais pacientes, pode ser feito o cruzamento entre os dados dos três classificadores. Como o *CCICLO* realiza o cruzamento dos classificadores dois a dois, inicialmente são cruzados dois classificadores e, em seguida, o resultado obtido é cruzado com o terceiro operador, completando a operação de cruzamento. A sintaxe do operador é apresentada na figura 7.12.

$$(C_6 \cap \wedge C_2 \cap \wedge C_3)$$

Figura 7.12: Exemplo do operador interseção.

A operação é realizada utilizando o operador interseção E para obter os pacientes cujos vírus não apresentam mutações e que são aderentes ao tratamento. Nota-se que não foi feita uma seleção das classes de objetos classificados ou não classificados antes da realização da interseção. Com o operador interseção E a seleção não é necessária, pois é realizada pelo operador lógico E ($\wedge$). Ele atribui o valor um somente para os rótulos dos pacientes que possuam inicialmente rótulos com valor um nos dois classificadores. Caso um dos rótulos possua valor zero, ele receberá o valor zero no final da operação.

Finalizando a execução da expressão, o operador interseção E identifica a interseção entre o resultado da parte inicial da expressão e o classificador C_3 . Mais uma vez, o operador lógico E foi utilizado para classificar somente os objetos com valor um para o rótulo nos dois conjuntos, ou seja, os pacientes com falha terapêutica que são aderentes ao tratamento e com vírus sem nenhuma mutação genética. Se o resultado obtido ao final da execução dos operadores não for vazio, os pacientes encontrados podem ser investigados a fundo em busca de novos conhecimentos sobre o caso, demonstrando o papel do *CCICLO* no apoio à descoberta de conhecimento.

7.4.7 Operador junção composta ($\bowtie \beta$)

O operador junção composta ($\bowtie \beta$) funciona como uma especialização da interseção composta, pois seleciona somente os objetos representados em ambos os classificadores e com o mesmo valor para o rótulo, independente de o valor do rótulo ser zero ou um, desde que seu valor seja o mesmo nos dois classificadores. Como resultado final, o operador apresenta um conjunto formado pelos objetos encontrados na interseção entre os dois conjuntos e que possuam o mesmo valor para o atributo rótulo nos dois classificadores. Dessa maneira, o operador junção composta apresenta uma restrição maior para a representação de objetos em seu conjunto resposta do que o operador interseção composta.

Como apresentado no capítulo 6 (ver seção 6.2.3 na página 64), sua sintaxe é:

$$C_1 \bowtie \beta C_2$$

O exemplo prático apresentado para a utilização do operador junção é a avaliação do comportamento evolutivo de uma classificação ao longo do tempo. Com ele é possível identificar os objetos que mantiveram o comportamento constante durante diferentes classificações.

Os classificadores propostos apresentam características importantes para análise temporal, tais como: acompanhar o surgimento de mutações em vírus que não as possuíam, a variação de aderência do paciente ao tratamento, as mudanças dos esquemas de drogas em uso e utilizadas. Como exemplo é analisada a evolução do tratamento dos pacientes com falha terapêutica.

Como a falha terapêutica pode variar com o tempo, conseqüentemente, a classificação dos pacientes também deve sofrer alteração. Essa mudança na classificação do paciente é um importante fator a ser analisado, pois contém importantes informações sobre a evolução da doença e que podem ajudar a desenvolver métodos para combatê-la.

As classificações realizadas pelo classificador C_3 ao longo do tempo contêm as informações sobre os pacientes que tiveram o quadro de falha terapêutica alterado. Para identificar tais pacientes a fim de investigar os motivos da mudança de classe, o operador junção composta pode ser utilizado para avaliar diferentes classificações de C_3 .

A análise da evolução temporal das classificações pode ser feita no *CCICLO* por meio da análise de todas as classificações realizadas em determinado período de tempo, podendo ser analisadas centenas de classificações. Porém, com o objetivo de manter clara a explicação sobre o uso do operador, será considerado que foram selecionadas apenas as duas classificações mais recentes. Com a mais nova sendo representada por C'_3 e a anterior por C''_3 , e que a classificação dos objetos será mantida. A sintaxe dessa expressão é apresentada na figura 7.13.

Figura 7.13: Exemplo de junção composta em C_3 .

Para manter a classificação inalterada, foi escolhido o operador lógico OU ($\vee$). Porém, como mostra a tabela 7.7, tanto o operador OU ($\vee$) quanto o operador E ($\wedge$) poderiam ser usados para manter a classificação. Ao final, o resultado armazenado em C'''_3 pode ser utilizado para separar os pacientes para estudo específico, como mostra a sintaxe a seguir.

Pacientes cujo tratamento permanece falhando $= \sigma_{R=1} C'''_3$

Pacientes cujo tratamento permanece funcionando $= \sigma_{R=0} C'''_3$

7.4.8 Inclusão dos resultados obtidos na estrutura em memória

Alguns dos exemplos apresentados durante o estudo de caso mostraram a necessidade da utilização encadeada de mais de um operador. Em muitos casos, o resultado obtido pela execução de um operador representa apenas um resultado intermediário da análise total desejada, tornando necessária

a utilização de mais operadores para dar continuidade à análise. Além disso, algumas análises podem possuir partes comuns que podem ser compartilhadas ou apresentarem a necessidade de modificação em parte do processo, criando a possibilidade de aproveitamento da parte do processamento já realizado. Para evitar processamentos desnecessários, os resultados obtidos durante as análises podem ser inseridos na estrutura em memória, passando a representar um novo classificador, possibilitando a sua participação em novos cruzamentos. Com exceção dos resultados gerados pelo operador projeção, os resultados dos demais operadores possuem formato compatível com a estrutura em memória, bastando apenas completar os dados necessários para sua representação, como a geração de um identificador e do vetor de característica para sua representação na memória. O identificador é obtido por meio de solicitação a aplicação responsável por armazenar o classificador em disco. Com esse identificador, o classificador passa a ser identificado na estrutura em memória e, posteriormente, em disco, quando o mesmo for gravado.

A definição do vetor de características é feita com a participação do pesquisador. Para tanto, o vetor de característica de cada classificador analisado é apresentado para que sejam escolhidas as características que vão representar em memória o novo classificador.

Caso os classificadores utilizados no cruzamento tenham sido obtidos por meio de uma consulta multidimensional, o *CCICLO* sugere que parte do vetor de característica do novo classificador seja formada pelas características utilizadas na seleção dos classificadores, pelo rótulo de tempo do cruzamento e pela referência à base de dados dos objetos.

Com a inserção do resultado em memória, o cruzamento de dados passa a ser visto como um classificador composto, ou seja, um classificador formado pela análise do resultado de outros classificadores. Essa característica adiciona grande flexibilidade ao processo de análise, permitindo que os resultados de diferentes cruzamentos entre objetos classificados sejam cruzados entre si, gerando resultados cada vez mais específicos. Sem essa alternativa, os classificadores necessitariam ser adaptados ou existiria a necessidade da produção de consultas SQL complexas e pouco eficientes.

Conclusão

8.1 Contribuições do trabalho

O *CCICLO* tem como objetivo apoiar o processo de descoberta de conhecimento por meio da análise de possíveis relações entre classes de objetos e da análise da evolução das classificações realizadas. A classificação de dados é uma das importantes tarefas realizadas pela *mineração de dados*. Entretanto, tal classificação não considera a necessidade da comparação automática entre diversas classificações de dados. Para explorar em alta escala e de forma automatizada essas duas importantes fontes de dados, o *CCICLO* oferece um ambiente para a análise das relações entre as classes e a evolução temporal das classificações como mais uma opção para análise do conjunto de dados gerados pelas classificações. A seguir são apresentadas as principais contribuições deste trabalho.

8.1.1 Redefinição dos operadores relacionais

Uma importante contribuição do trabalho é a adaptação dos operadores relacionais. Tal adaptação permite cruzar um grande número de classes sem a necessidade de desenvolver uma aplicação específica para analisar cada conjunto de classe. Além do mais, o uso dos operadores compostos permite a definição de uma nova classificação para os objetos, sem a necessidade de projetar novos

classificadores.

Os operadores permitem explorar ainda as diferentes formas de composição e cruzamento existentes nas análises de dados. Com o uso de tais operadores, análises mais aprofundadas dos objetos são realizadas por meio de cruzamentos encadeados de diferentes classificações. A utilização de tais operadores pode ser entendida como um refinamento das classificações iniciais.

8.1.2 Integração de conhecimento especializado

O desenvolvimento de um classificador não é um trabalho simples, pois demanda conhecimento específico de determinada área e a abordagem de vários tipos de métodos de descoberta de conhecimento (matemáticos, estatísticos, redes neurais) para tornar factível a identificação de classes.

Quando uma análise envolve múltiplos classificadores, que abordam diferentes áreas do conhecimento, ela demanda especialistas de cada área envolvida. A criação de uma equipe tão heterogênea e especializada também não é uma tarefa fácil, pois exige tempo e recursos financeiros. Contudo, muitos classificadores são criados para atenderem às necessidades específicas de algumas pesquisas, formando uma base de conhecimento de diversas áreas.

O *CCICLO* possibilita a integração do conhecimento já representado em forma de classificadores. Para tanto, ele usa seus operadores para cruzar os resultados das classificações efetuadas. Essa integração de conhecimento apresenta uma alternativa para continuação da análise das classificações já existentes, o que permite aproveitar investimentos e processamentos realizados anteriormente em vários tipos de classificações.

Integração de dados de classificadores heterogêneos

Como apresentado no capítulo 3, os classificadores podem usar diferentes técnicas para identificar classes. Além disso, os dados analisados por eles podem estar armazenados em formatos e ambientes heterogêneos, causando problemas para a integração dos dados. O *CCICLO* trata a integração dos resultados de classificadores com o uso de um formato padrão para disponibilizar

os dados da classificação. A partir desse formato, os resultados são inseridos na estrutura em memória ficando solucionados os problemas de integração de dados.

8.1.3 Representação em estrutura de dados do contexto de classificação

A representação do contexto de classificação é uma das contribuições centrais do *CCICLO*, pois a partir dela, o ambiente é definido para realizar o cruzamento de objetos classificados e explorar a relação entre as classificações. Essa representação busca suprir a necessidade de dados para análise de um contexto de classificação de objetos e simplificar a avaliação das relações entre as classes. Como apresentado pela figura 5.1, a estrutura de dados do *CCICLO* incorpora o contexto das classificações por meio da representação dos dados sobre os classificadores e dos dados sobre os objetos classificados por eles, oferecendo dois níveis de informação. Um nível apresenta as características usadas para a definição dos classificadores e algumas informações sobre o processo de classificação, e o outro expõe os dados sobre os objetos classificados.

A contribuição dessa estrutura de dados está relacionada à abstração dos conceitos envolvidos no processo de classificação. Tal abordagem permite focar a atenção à classificação recebida pelos dados, reduzindo o volume de dados processados e facilitando a compreensão das análises realizadas.

Representação resumida dos objetos

A estrutura de dados do *CCICLO* propõe uma representação resumida dos objetos classificados. Assim, o espaço ocupado em memória é reduzido, viabilizando o uso da memória principal do computador como fonte para o cruzamento de dados. Dessa maneira, o cruzamento de dados pode ser realizado de forma mais rápida, aproveitando o menor gasto de tempo para recuperar o dado em memória, em relação ao tempo gasto para recuperar o dado em disco.

8.1.4 Representação multidimensional das características dos classificadores

A escolha dos classificadores a serem utilizados no cruzamento de objetos pode ser feita de maneira pontual, em que os classificadores são indicados um a um. Porém, o *CCICLO* oferece a possibilidade de realizar a escolha dos classificadores usando uma consulta multidimensional. Para tanto, o vetor de característica do classificador e algumas informações sobre processo de classificação são representados como dimensões de uma R-Tree. Assim, os classificadores podem ser escolhidos pelas condições a serem executadas sobre as dimensões da R-Tree.

Por exemplo: escolher todos os classificadores que avaliaram determinada característica dos pacientes ou selecionar todas as classificações realizadas por determinado classificador em uma base de dados específica.

Por ser realizada de forma automática por meio de regras, a seleção multidimensional ajuda a prevenir erros no momento da informação dos classificadores a serem utilizados. Além disso, as condições especificadas para a seleção passam a agregar maior valor semântico aos objetos escolhidos. Dessa forma, os objetos podem ser analisados não somente como classes, mas também como classes definidas de acordo com determinadas regras e parâmetros. Portanto, a representação multidimensional das características dos classificadores e informações do processo de análise contribui para o cruzamento de objetos classificados.

8.2 Aplicação prática do CCICLO

O estudo de caso apresentou necessidades das pesquisas sobre o vírus HIV que podem ser atendidas com relativa facilidade pelo uso do *CCICLO*. Porém, os exemplos apresentados podem ser generalizados para aplicações em diferentes áreas, sejam elas acadêmicas ou comerciais.

A apresentação dos exemplos, tanto da seleção multidimensional dos classificadores quanto do uso dos operadores, trabalhou com um pequeno número de classificadores para manter a clareza

das explicações, por meio da diminuição dos conceitos envolvidos. Porém, o *CCICLO* foi definido para trabalhar com grande quantidade de classificadores e de objetos. Nesse contexto, cresce a importância da flexibilidade oferecida pela seleção multidimensional dos classificadores e da simplicidade proposta pelos operadores para o cruzamento dos objetos analisados. Com uma grande quantidade de classificadores, a escolha individual de quais deles devem ser analisados é trabalhosa e pouco eficiente. A seleção multidimensional facilita e torna mais dinâmica a seleção dos classificadores, além de evitar erros na escolha deles. Uma vez selecionado um grupo com diversos classificadores, a indicação dos cruzamentos por meio dos operadores fica mais simples e padronizada.

8.3 Futuras pesquisas

Os estudos para o desenvolvimento do *CCICLO* apontaram necessidades adicionais que não foram abordadas por este trabalho, por fugirem ao escopo proposto. Essas necessidades são apresentadas a seguir, como sugestão para futuras pesquisas.

8.3.1 Implementação do CCICLO

A implementação do *CCICLO* é uma consequência direta das contribuições que podem ser obtidas por meio de seu uso. O ambiente do *CCICLO* apresenta a reunião de estruturas de dados e conceitos conhecidos e bastante utilizados pela ciência da computação, tornando sua implementação uma tarefa viável. As estruturas de dados utilizadas como vetores, listas ligadas e árvore R-Tree, são encontradas de forma bem detalhada na literatura ou em implementações disponíveis. A implementação dos operadores relacionais adaptados também pode ser realizada de maneira simples e eficiente, em razão da representação das classes em duas listas ligadas distintas e ordenadas.

Como parte de sua implementação, O *CCICLO* será incluído na arquitetura geral para descoberta de conhecimento descrita em [BJFG02], como mostra a figura 8.1.

Figura 8.1: Ambiente integrado para descoberta de conhecimento em biologia.

8.3.2 Interface gráfica para o CCICLO

A vocação do *CCICLO* para trabalhar com grande volume de dados torna importante a criação de uma interface gráfica para facilitar e potencializar sua utilização pelo usuário. A seguir, são apresentados duas sugestões para a interface gráfica do *CCICLO*.

Escolha dos classificadores

A representação multidimensional de dados exige bastante flexibilidade e simplicidade da interface gráfica para facilitar o uso e a compreensão do usuário.

Uma abordagem importante para a escolha dos classificadores é a apresentação de agrupa-

mentos deles de acordo com algumas características representadas. Essa visão oferece ao usuário uma noção sobre a quantidade de classificadores que avaliam determinadas características e como as características de tais classificadores se relacionam. A figura 5.3 apresenta os classificadores representados em um espaço bidimensional, no qual foram representadas as dimensões: doença e órgão afetado. Essa representação permite visualizar com relativa facilidade os classificadores que analisam de forma simultânea órgãos de um mesmo sistema do organismo e os tipos de doenças que afetaram esses órgãos.

A motivação para a representação gráfica dos classificadores é aumentar a intuição do usuário, por meio da seleção visual dos classificadores.

Linguagem visual para aplicação dos operadores

Os operadores relacionais oferecidos pelo *CCICLO* podem ter seu funcionamento representado de maneira gráfica. Por se tratar de operações de conjunto, a representação gráfica de operações como interseção, união e junção pode ser feita com a delimitação da área de cada classe e com o uso de cores para identificar os elementos que compõem cada classe e o resultado da execução dos operadores.

8.3.3 Adaptação do CCICLO para trabalhar com incertezas nas classificações

A classificação binária é bastante utilizada na *mineração de dados*, contudo nem sempre é possível definir com exatidão a classificação de alguns objetos. Normalmente, tais objetos são rotulados como não classificados para garantir uma classificação segura. Para tratar as incertezas no processo de classificação, o *CCICLO* pode ser adaptado para representar e trabalhar com valores que indiquem a precisão das classificações realizadas. Assim, os objetos não somente possuiriam o indicador da classe a qual eles pertencem, como também possuiriam um valor informando qual é a precisão dessa classificação. Para tal adaptação, seriam necessárias duas alterações. A primeira

alteração seria a inclusão de um atributo, na estrutura de dados que representa o objeto analisado, para indicar a precisão da classificação recebida por ele. A segunda alteração seria redefinir os operadores de modo a incluir uma opção para selecionar os objetos cuja classificação possua uma determinada precisão.

Referências Bibliográficas

- [AEK00] Mihael Ankerst, Martin Ester, and Hans-Peter Kriegel. Towards an effective cooperation of the user and the computer for classification. *ACM SIGKDD*, pages 179–188, 2000.
- [BBK01] Christian Böhm, Stefan Berchtold, and Daniel A. Keim. Searching in high-dimensional spaces: Index structures for improving the performance of multimedia databases. *ACM Computing Surveys (CSUR)*, 33(3):322–373, 2001.
- [BJFG02] Junior Barrera, Roberto M. César Júnior, João E. Ferreira, and Marco D. Gubitoso. An environment for knowledge discovery in biology. *Submetido ao Journal of Computers in Biology*, 2002.
- [BKK96] Stefan Berchtold, Daniel A. Keim, and H. P. Kriegel. *The X-tree: An index structure for high-dimensional data*. *VLDB*, 1996.
- [BKSS90] Norbert Beckmann, Hans-Peter Kriegel, Ralf Schneider, and Bernhard Seeger. The R*-tree: an efficient and robust access method for points and rectangles. In *Proceedings of the 1990 ACM SIGMOD international conference on Management of data*, pages 322–331. ACM Press, 1990.
- [BMR01] Philip Bohannon, Peter Mclroy, and Rajeev Rastogi. Main-memory index structures

with fixed-size partial keys. In *Proceedings of the 2001 ACM SIGMOD international conference on Management of data*, pages 163–174. ACM Press, 2001.

- [BOSD⁺97] Elisa Bertino, Beng Chin OOI, Ron Sacks-Davis, Kian lee Tan, Justin Zobel, Boris Shidlovsky, and Barbara Catania. *Indexing Techniques for advanced database systems*. Kluwer Academic Publishers, 1997.
- [CD97a] Surajit Chaudhuri and Umeshwar Dayal. Data warehousing and OLAP for decision support. In *Proceedings of the 1997 ACM SIGMOD international conference on Management of data*, pages 507–508. ACM Press, 1997.
- [CD97b] Surajit Chaudhuri and Umeshwar Dayal. An overview of data warehousing and OLAP technology. *ACM SIGMOD Record*, 26(1):65–74, 1997.
- [CGB01] James I. Cleeman, Scott M. Grundy, and Diane Becker. *Executive Summary of the Third Report of the Nacional Cholesterol Education Program (NCEP) Expert Panel on Detection, Evaluation and Treatment of High Blood Cholesterol in Adults*. *JAMA - Journal of American Medical Association*, 285(19):2486–2497, 2001.
- [Cha01] Tony Y. T. Chan. Unsupervised classification of noisy chromosomes. *Bioinformatics*, 17(5):438–444, 2001.
- [CI98] Chee-Yong Chan and Yannis E. Ioannidis. Bitmap index design and evaluation. In *Proceedings of the 1998 ACM SIGMOD international conference on Management of data*, pages 355–366. ACM Press, 1998.
- [CI99] Chee-Yong Chan and Yannis E. Ioannidis. An efficient bitmap encoding scheme for selection queries. In *Proceedings of the 1999 ACM SIGMOD international conference on Management of data*, pages 215–226. ACM Press, 1999.

- [CKM⁺00] Jens Claussen, Alfons Kemper, Guido Moerkotte, Klaus Peithner, and Michael Steinbrunn. Optimization and evaluation of disjunctive queries. In *TKDE*, pages 238–260, 2000.
- [CKPS95] Surajit Chaudhuri, Ravi Krishnamurthy, Spyros Potamianos, and Kyuseak Shim. Optimizing queries with materialized views. In *11th Int. Conference on Data Engineering*, pages 190–200, Los Alamitos, CA, 1995. IEEE Computer Soc. Press.
- [Com79] Douglas Comer. Ubiquitous B-Tree. *ACM Computing Surveys (CSUR)*, 11(2):121–137, 1979.
- [CRS99] Boris Chidlovskii, Claudia Roncancio, and Marie-Luise Schneider. *Semantic Cache Mechanism for Heterogeneous Web Querying*. *8th World Wide Web Conference(WWW8)*, 1999.
- [CS99] Surajit Chaudhuri and Kyuseok Shim. Optimization of queries with user-defined predicates. *ACM Transactions on Database Systems*, 24(2):177–228, 1999.
- [DFJ⁺96] Shaul Dar, Michael J. Franklin, Bjor T. Jonsson, Divesh Srivastava, and Michael Tan. Semantic data caching and replacement. In *Proc. VLDB Conf., Bombay, India*, pages 330–341, 1996.
- [dRSN98] Prasad M. deshpane, Karthikeyan Ramasamy, Amit Shukla, and Jeffrey F. Naughton. Caching multidimensional queries using ckunks, 1998.
- [dS00] Ministério da Saúde. *Estimativa do Número de Indivíduos de 15 a 49 anos infectados pelo HIV*. Coordenação Nacional de DST e AIDS, Brasil, 2000.
- [dS01a] Ministério da Saúde. *Guia de tratamento clínico da infecção pelo HIV em crianças*. Coordenação Nacional de DST e AIDS, Brasil, 2001.

- [dS01b] Ministério da Saúde. *Recomendações para profilaxia da transmissão materno-infantil do HIV e terapia anti-retroviral em gestantes*. Coordenação Nacional de DST e AIDS, Brasil, 2001.
- [dS01c] Ministério da Saúde. *Recomendações para terapia anti-retroviral em adultos e adolescentes infectados pelo HIV*. Coordenação Nacional de DST e AIDS, Brasil, 2001.
- [EEvH99] Jeroen Eggermont, Agoston E. Eiben, and Jano I. van Hemert. A comparison of genetic programming variants for data classification. In *Advances in Intelligent Data Analysis, Third International Symposium, IDA-99*, volume 1642, pages 281–290. Springer-Verlag, 9–11 1999.
- [FLS98] Daniela Florescu, Alon Levy, and Dan Suciu. Query containment for conjunctive queries with regular expressions. In *Accepted for publication, PODS 98.*, pages 139–148, 1998.
- [Fre02] A.A. Freitas. A survey of evolutionary algorithms for data mining and knowledge discovery. In *To appear in: A. Ghosh and S. Tsutsui. (Eds.) Advances in Evolutionary Computation*. Springer-Verlag, 2002.
- [GG96] Parke Godfrey and Jarek Gryz. A framework for intensional query optimization. In *Proceedings of the Workshop on DDLP Deductive Databases and Logic Programming at JICSLP'96*, pages 57–68, 1996.
- [GG97] Parke Godfrey and Jarek Gryz. Semantic query caching for heterogeneous databases. In *Proceedings KRDB at VLDB'97 – Knowledge Representation Meets Databases*, pages 6.1–6.6, 1997.
- [GG99] Parke Godfrey and Jarek Gryz. Answering queries by semantic caches. In *Database and Expert Systems Applications*, pages 485–498, 1999.

- [GKF⁺01] Birgitta Geier, Gabi Kastenmüller, Matthias Fellenberg, Hans-Werner Mewes, and Burkhard Morgenstern. The HIV databases of annotated unigene clusters. *Bioinformatics*, 17(6):571–572, 2001.
- [Gut84] A. Guttman. R-trees: A dynamic index structure for spatial searching. In *SIGMOD'84 14(2)Haunold, P.S*, 1984.
- [HK01] Jiawei Han and Micheline Kamber. *Data Mining – Concepts and Techniques*. Morgan Kufmann Publishers, 2001.
- [HP02] Vagelis Hristidis and Michalis Petropoulos. Semantic caching of XML databases. *Fifth International Workshop on the Web and Databases – WebDB 2002*, 26(1), 2002.
- [IA00] Alfred Inselberg and Tova Avidan. Classification and visualization for high-dimensional data. In *Proceedings of the sixth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 370–374. ACM Press, 2000.
- [IF02] Isabel Cristina Italiano and João Eduardo Ferreira. *Classificação dos domínios transacionais e analíticos para avaliação de alternativas de sincronismo em projetos de Data Warehouse*. Dissertação de mestrado, Instituto de Matemática e Estatística da Universidade de São Paulo, São Paulo, 2002.
- [KB96] Arthur M. Keller and Julie Basu. A Predicate-based Caching Scheme for Client-Server Database Architectures. *VLDB Journal*, 5(2):35–47, 1996.
- [KCK01] Kihong Kim, Sang K. Cha, and Keunjoo Kwon. Optimizing multidimensional index trees for main memory access. In *Proceedings of the 2001 ACM SIGMOD international conference on Management of data*, pages 139–150. ACM Press, 2001.

- [KF93] Ibrahim Kamel and Christos Faloutsos. On packing R-trees. In *Proceedings of the second international conference on Information and knowledge management*, pages 490–499. ACM Press, 1993.
- [KMPS94] A. Kemper, G. Moerkotte, K. Peithner, and M. Steinbrunn. Optimizing disjunctive queries with expensive predicates. In *Proc. of the ACM SIGMOD Conf. on Management of Data*, pages 336–347, 1994.
- [LC99] Dongwon Lee and Wesley W. Chu. Semantic caching via query matching for web sources. In *Proceedings of the eighth international conference on Information and knowledge management*, pages 77–85. ACM Press, 1999.
- [Liu92] Ling Liu. *A formal approach to Structure, Algebra and Communication of Complex Objects*. PhD thesis, Proefschrift Katholieke Universiteit Brabant Tilburg – the Netherlands, 1992.
- [MBK00] S. Manegold, P. A. Boncz, and M. L. Kersten. What happens during a join? - dissecting CPU and memory optimization effects. In *Proceedings of the International Conference on Very Large Data Bases (VLDB)*, Setembro, 2000.
- [MF01] Leonardo A. Mantovani and João Eduardo Ferreira. *Uma alternativa para simplificação do sincronismo de dados históricos dos ambientes operacionais e analíticos*. Dissertação de mestrado, Instituto de Ciências Matemáticas e de Computação da Universidade de São Paulo, São Carlos, 2001.
- [MK98] Seigo Muto and Masaru Kitsuregawa. Improving main memory utilization for array-based datacube computation. In *Proceeding of the first ACM international workshop on Data warehousing and OLAP*, pages 28–33. ACM Press, 1998.
- [QGMJ97] D. Quass, A. Gupta, I.S. Mumick, and J. Window. Making views self-maintainable

for data warehousing. In *Proceedings of the eighth international conference on Information and knowledge management*, 1997.

- [RD99] Qun Ren and Margaret H. Dunham. Using clustering for effective management of a semantic cache in mobile computing. In *Proceedings of the ACM international workshop on Data engineering for wireless and mobile access*, pages 94–101. ACM Press, 1999.
- [RD00] Qun Ren and Margaret H. Dunham. Using semantic caching to manage location dependent data in mobile computing. In *Proceedings of the sixth annual international conference on Mobile computing and networking*, pages 210–221. ACM Press, 2000.
- [RR99] Jun Rao and Kenneth A. Ross. Cache conscious indexing for decision-support in main memory. In *The VLDB Journal*, pages 78–89, 1999.
- [RR00] Jun Rao and Kenneth A. Ross. B+-trees cache conscious in main memory. In *Proceedings of the 2000 ACM SIGMOD international conference on Management of data*, pages 475–486. ACM Press, 2000.
- [SC00] Thomas Schreck and Zhengxin Chen. R-tree implementation using branch-grafting method. In *Proceedings of the 2000 ACM symposium on Applied computing 2000*, pages 328–332. ACM Press, 2000.
- [SCJ01] Sadagopan Srinivasan, Vinodh Cuppu, and Bruce Jacob. Transparent data-memory organizations for digital signal processors. In *Proceedings of the international conference on Compilers, architecture, and synthesis for embedded systems*, pages 44–48. ACM Press, 2001.
- [SKATSR01] Elizabeth Shoop, James E. Johnson Kevin A. T. Silverstein, and Ernest F. Retzel. Metafam: a unified classification of protein families. II. schema and query capabilities. *Bioinformatics*, 17(3):262–271, 2001.

- [SRF87] Timos K. Sellis, Nick Roussopoulos, and Christos Faloutsos. The R+-tree: A dynamic index for multi-dimensional objects. In *The VLDB Journal*, pages 507–518, 1987.
- [Tea99] CORPORATE TimesTen Team. In-memory data management for consumer transactions the timesten approach. In *Proceedings of the 1999 ACM SIGMOD international conference on Management of data*, pages 528–529. ACM Press, 1999.
- [WB98] Ming-Chuan Wu and Alejandro P. Buchmann. Encoded bitmap indexing for data warehouses. In *Proc. 14th ICDE*, pages 220–230, 1998.
- [WMSW00] Jason T. L. Wang, Qicheng Ma, Dennis Shasha, and Cathy H. Wu. Application of neural networks to biological data mining: a case study in protein sequence classification. In *Proceedings of the sixth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 305–309. ACM Press, 2000.
- [Wu99] Ming-Chuan Wu. Query optimization for selections using bitmaps. *SIGMOD Record (ACM Special Interest Group on Management of Data)*, 28(2):227–238, 1999.
- [YKL97] Jian Yang, Kamalakara Karlapalem, and Qing Li. A framework for designing materialized views in data warehousing environment. In *17th International Conference on Distributed Computing Systems (17th IDCS'97)*, pages 458–465, Baltimore, MD, maio 1997. IEEE.
- [YRT⁺01] Chen-Hsiang Yeang, Sridhar Ramaswamy, Pablo Tamoyo, Sayan Mukherjee, Ryan M. Rffkin, Michael Angelo, Michel Reich, Eric Lander, Jill Mesirov, and Todd Golub. Molecular classification of multiple tumor types. *Bioinformatics*, 17(1):316–322, 2001.