DETECÇÃO DE ANOMALIAS EM TELECOMUNICAÇÕES ATRAVÉS DE UM SISTEMA BASEADO EM CONHECIMENTO QUE UTILIZA CONSULTA POR SIMILARIDADE, DWT E RDR COMO FERRAMENTAS DE APOIO

por

Umberto Maia Barcelos

DISSERTAÇÃO APRESENTADA À
UNIVERSIDADE FEDERAL DE UBERLÂNDIA
UBERLÂNDIA, MINAS GERAIS
COMO PARTE DOS REQUISITOS EXIGIDOS
PARA OBTENÇÃO DO TÍTULO DE
MESTRE EM CIÊNCIA DA COMPUTAÇÃO
SETEMBRO 2005

UNIVERSIDADE FEDERAL DE UBERLÂNDIA
FACULDADE DE FACULDADE DE COMPUTAÇÃO

Os abaixo assinados, por meio deste, certificam que leram e recomendam para a Faculdade de Faculdade de Computação a aceitação da dissertação intitulada "**Detecção de Anomalias em Telecomunicações através de um Sistema Baseado em Conhecimento que utiliza Consulta por Similaridade, DWT e RDR como ferramentas de apoio**" por **Umberto Maia Barcelos** como parte dos requisitos exigidos para a obtenção do título de **Mestre em Ciência da Computação**.

Data: Setembro 2005

Orientadora:

Prof.ª. Drª. Rita Maria da Silva Julia
(Universidade Federal de Uberlândia)

Banca Examinadora:

Prof. Dr. Pedro Paulo Balbi de Oliveira
(Universidade Presbiteriana Mackenzie)

Prof. Dr. Antônio Eduardo Costa Pereira
(Universidade Federal de Uberlândia)

UNIVERSIDADE FEDERAL DE UBERLÂNDIA

Data: **Setembro 2005**

Autor:	**Umberto Maia Barcelos**
Título:	**Detecção de Anomalias em Telecomunicações através de um Sistema Baseado em Conhecimento que utiliza Consulta por Similaridade, DWT e RDR como ferramentas de apoio**
Faculdade:	**Faculdade de Computação**
Grau:	**Mestre**

Assinatura do Autor

À Kleimans, minha querida esposa, que certamente soube esperar e compreender, mas acima de tudo, soube dar carinho, apoio e incentivo. Você caminhou comigo a cada momento deste trabalho, sempre me ajudando a vencer este desafio. Muito obrigado por fazer parte da minha vida!

Agradecimentos

Esta dissertação nunca teria sido concretizada sem a contribuição de várias pessoas para as quais eu tenho o prazer de expressar minha apreciação e gratidão.

Primeiramente, eu gostaria de agradecer à minha Orientadora, a Professora Rita, pela persistência e pelo apoio nos momentos difíceis. Ela foi responsável pela inspiração de muitas das idéias apresentadas aqui.

Também gostaria de agradecer ao meu grande amigo Jony pelo incentivo, amizade e companheirismo.

Aos amigos Adriano, Reginaldo, Eduardo, Apolônio, Germano, Erasmo, Maiymi, Vagner, Michela, Melissa e Lininha agradeço pelo apoio demonstrado ao longo destes anos.

Ao meu professor e amigo John Joe O'Connell agradeço pela paciência e pelo interesse em ajudar.

Agradeço a minha cunhada Susanne, ao meu cunhado Zaidenzinho e ao pequenino Vinícius pelos momentos felizes que vocês me proporcionam quando estamos juntos.

Não poderia deixar de agradecer em especial à Darlene pelo apoio, incentivo e compreensão.

Agradeço também em especial à FAPEMIG e à CTBC Telecom pelo apoio financeiro para apresentação de artigo.

Gostaria de agradecer aos meus pais que nunca mediram esforços em me dar a melhor educação possível e ao meu irmão César pela alegria e pela amizade.

Finalmente, agradeço especialmente à minha esposa Kleimans pelo amor, incentivo e companheirismo.

Resumo

A Fraude em telefonia é um fenômeno mundial. Estimativas correntes contabilizam perdas de 15 A 55 bilhões de dólares por ano na indústria de telefonia, que movimenta negócios na ordem de 1,5 trilhão de dólares [42]. A detecção de fraude é um ponto crítico nestas empresas. Os riscos de fraude forçam as empresas a empreender elevados esforços na análise do tráfego de ligações de clientes. Nesta dissertação, propõe-se utilizar um sistema composto por módulos para efetuar consultas por similaridade em séries temporais com o objetivo de detectar, prematuramente, anomalias no tráfego de ligações de clientes. Para aumentar a eficiência das consultas, uma vez que o volume de dados a ser analisado é muito grande, o sistema utiliza as *Haar Wavelets* como técnica de redução de dados. A fim de indicar as ações que deverão ser desencadeadas em função dos resultados das consultas por similaridade, é utilizado um Sistema Baseado em Conhecimento que corresponde a uma combinação de um sistema de produção e de um sistema baseado em casos. Tal sistema corresponde à estrutura Ripple Down Rules (RDR). O sistema proposto representa uma alternativa tecnicamente eficiente e inovadora na detecção de anomalias em sistemas de telefonia.

Palavras-chave: Wavelet, Consulta por Similaridade, Telecomunicações, RDR, Sistema baseado em conhecimento

Abstract

Telephony fraud is a world phenomenon. Current estimates show losses of from 15 to 55 billion dollars a year in the telephony industry, which runs businesses around 1.5 trillion dollars [42]. Fraud detection is a critical point for these companies. Fraud risks force companies to make a great effort to analyze clients' calls traffic. This paper proposes a Knowledge System (KS) that performs similarity search in time series with the purpose of detecting anomalies, in advance, in the clients' call traffic. Since there is a great amount of data (call) to be analyzed, the system uses *Haar Wavelets* as a dimensionality reduction technique to improve the efficiency of the searches. For the purpose of indicating the actions that should be taken as a result of similarity searches, a knowledge system based on production systems and on case based reasoning system is utilized. This hybrid system is called Ripple Down Rules (RDR).

The system proposed represents a technically efficient and innovating alternative for detecting anomalies in telephony systems.

Keywords: Wavelet, Similarity Search, Telecommunication, RDR, Knowledge System Based

Sumário

Lista de Tabelas

Lista de Figuras

Capítulo 1

Introdução

1.1 Considerações Iniciais e Motivações

A fraude de telefonia é um fenômeno mundial. Estimativas correntes contabilizam perdas de 15 A 55 bilhões de dólares por ano na indústria de telefonia, que movimenta negócios na ordem de 1,5 trilhão de dólares [42]. O volume de ligações é gigantesco, o que dificulta a análise dos dados[1]. A detecção de fraude é um ponto crítico nestas empresas. Atualmente, existem vários sistemas comerciais para detecção de fraudes cada vez mais sofisticados, muitos deles fazendo uso de técnicas de Inteligência Artificial [42]. Os riscos de fraude forçam as empresas a empreender elevados esforços na análise do tráfego de ligações de clientes.

Nesta dissertação, propõe-se um sistema modular para telecomunicações (SMT) que efetua consultas por similaridade em séries temporais para auxiliar na detecção de anomalias de consumo baseadas no perfil de uso de telefone e que propõe ações a serem efetuadas caso tais anomalias se confirmem. Para tanto, o SMT representa os dados referentes às chamadas telefônicas através de séries temporais que são criadas por um módulo Gerador de Séries.

A análise visando a detecção de anomalias é efetuada através de pesquisa de similaridade entre as séries temporais que representam a utilização real das linhas telefônicas e as séries que representam um padrão de uso ideal estabelecido pelos

[1]Estamos considerando apenas informações contidas nos CDR's (Call Detail Records).

proprietários da linha ou um perfil de fraude.

Um outro módulo do SMT é o módulo Gerador de Consultas. Tal módulo tem como função executar as consultas propostas pelos usuários do SMT indagando sobre a normalidade ou não da utilização das linhas telefônicas.

Uma vez terminado o processo de consultas, o SMT aciona um último módulo: o Sistema de Conhecimento (ou Sistema Baseado em Conhecimento), responsável por indicar as ações que deverão ser desencadeadas em função dos resultados das análises de similaridade. Tal módulo corresponde a uma combinação de um sistema de produção e de um sistema baseado em casos (*Case Based Reasoning*), sendo construído nos moldes da estrutura RDR (*Ripple Down Rules*). Este módulo é composto por regras relacionadas ao universo da telefonia, como por exemplo: *Se o perfil de ligações de um determinado telefone (série temporal) estiver SEMELHANTE a um padrão de fraudes (série temporal) conhecido, então este telefone pode ser enviado a uma área responsável por fraudes para ser analisado.*

Um dos problemas que comprometem a eficiência dessas análises é o grande volume de dados a ser analisado. Para resolver este problema, o SMT lança mão de duas ferramentas: as *Haar Wavelets*, como técnica para reduzir os dados sem comprometer os resultados da análise, e a indexação, como técnica de otimização de consulta. Um segundo problema consiste no fato de o conhecimento ser muito dinâmico, exigindo freqüentes atualizações na Base de Conhecimento. É para lidar com este problema que o SMT utiliza o RDR como ferramenta de construção incremental da base de conhecimento.

1.2 Objetivos

O objetivo desta dissertação é propor um sistema modular para telecomunicações (SMT) de auxílio na detecção de anomalias em sistemas de telecomunicação.

1.3 Estrutura da Dissertação

Esta dissertação está estruturada da seguinte forma: No Capítulo 2 são apresentados conceitos teóricos básicos para a compreensão da proposta. No Capítulo 3, é apresentado o estado da arte relativo à utilização das diversas técnicas aqui utilizadas. A dissertação é apresentada e detalhada no Capítulo 4. Por fim, as conclusões, resultados e trabalhos futuros são apresentados no Capítulo 5.

Capítulo 2

Suporte Teórico

2.1 Introdução

O propósito deste capítulo é dar suporte teórico para uma melhor compreensão do sistema proposto neste trabalho. Conforme introduzido anteriormente, tal sistema representa dados dos clientes de uma empresa de telecomunicações através de séries temporais. A grande quantidade de dados a serem armazenados e analisados exige, a título de eficiência, a aplicação de uma técnica de redução de dados que, no caso da presente proposta, consiste nas *Haar Wavelets*. A fim de explorar as informações contidas nesses dados, utiliza-se a estrutura de um Sistema Baseado em Conhecimento (SC) cuja Base de Conhecimento (BC) é construída incrementalmente, através da ferramenta RDR. Tal exploração é conseguida por meio de técnicas de consultas aplicadas às séries temporais. Nas seções que se seguem apresenta-se um resumo das ferramentas que servem de suporte ao sistema.

2.2 As Séries Temporais

Uma série temporal é uma coleção de observações feitas seqüencialmente no tempo [10]. Estas observações, normalmente, são números reais representados em intervalos regulares como, por exemplo, anualmente, mensalmente, diariamente. Freqüentemente, os dados irregulares são interpolados para formar valores regulares antes de

serem analisados [48]. Como exemplo de séries temporais, podemos citar [31]:

1. estimativas trimestrais de PNB ;
2. valores diários de temperatura de uma determinada cidade;
3. valores mensais de vendas de automóveis no Brasil;
4. um registro de marés no porto de Santos.

As Séries Temporais são chamadas *contínuas* se as observações são realizadas continuamente no tempo e, *discretas*, se elas são feitas apenas em determinados pontos. Nos exemplos 1-3 as séries temporais são discretas, enquanto que, no exemplo 4, são contínuas. Muitas vezes, uma série temporal discreta é obtida através de amostragem de uma série temporal contínua em intervalos de tempos iguais, Δt. Em outros casos, tem-se o valor da série acumulado em um dado instante, como ilustra o exemplo 3.

Em geral, uma série temporal é chamada de *regular* se os pontos observados são feitos no mesmo espaço, caso contrário elas são chamadas de *irregulares*. Formalmente, uma série temporal que descreve o estado de um objeto de dados em n pontos pode ser definida como uma seqüência ordenada:

$$S = \{S_i\}_{i=0}^{n-1}, S_i = (t_i, v_i) \tag{2.2.1}$$

onde v_i é o valor do objeto de dados no ponto t_i. Normalmente o valor de um objeto de dados é um valor numérico (como, por exemplo, preços, temperaturas etc.), porém, pode haver inúmeras situações onde o valor do dado não seja um simples inteiro, mas uma seqüência de imagens que variam com o tempo, uma série de pulsos telefônicos, etc. Um modelo clássico para séries temporais supõe que a série temporal $Z_t, t = 1, ...N$ possa ser escrita como a soma de três componentes: uma tendência, uma componente sazonal e um termo aleatório:

$$Z_t = T_t + S_t + a_t, t = 1, ..., N. \tag{2.2.2}$$

A tendência é causada por fatores que são medidos durante períodos longos de tempo. Já a componente sazonal aparece quando as observações são registradas em períodos curtos e apresenta uma periodicidade marcante.

Para facilitar nosso entendimento, segue abaixo exemplos de séries temporais reais [31]:

1. Produção de leite no Estado de São Paulo, composta de 77 dados medidos mensalmente e com periodicidade de 12 meses;

2. Índice de Produto Industrial do Brasil, composta de 139 observações mensais e com periodicidade de 12 meses;

3. Índice de Custo de Vida de São Paulo, com 126 observações mensais e não-sazonal;

O gráfico a seguir mostra uma série temporal citada em [4]: O próximo exemplo,

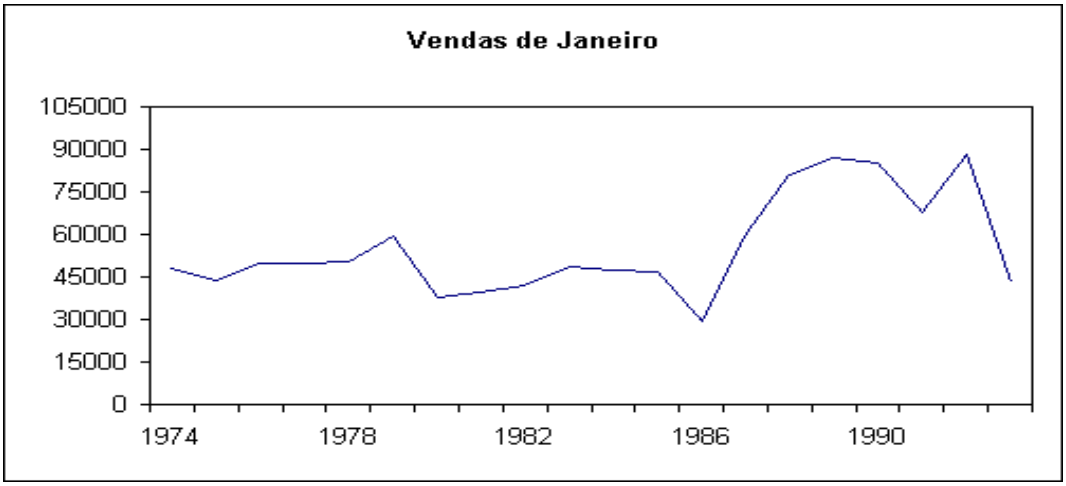

Figura 2.1: Gráfico da Série (apenas o mês de janeiro): Vendas de Carros no Mercado Espanhol

mostra a média de temperatura da cidade de São Paulo durante o ano, seguido de um exemplo que exibe número de ligações telefônicas de um determinado telefone em 1 semana.

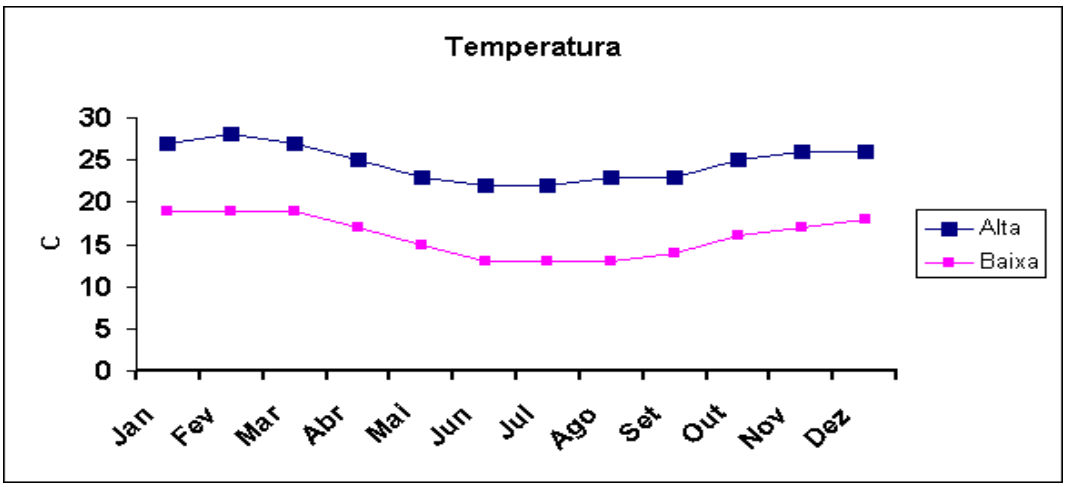

Figura 2.2: Média de Temperatura anual da cidade de São Paulo. Fonte: weather.com

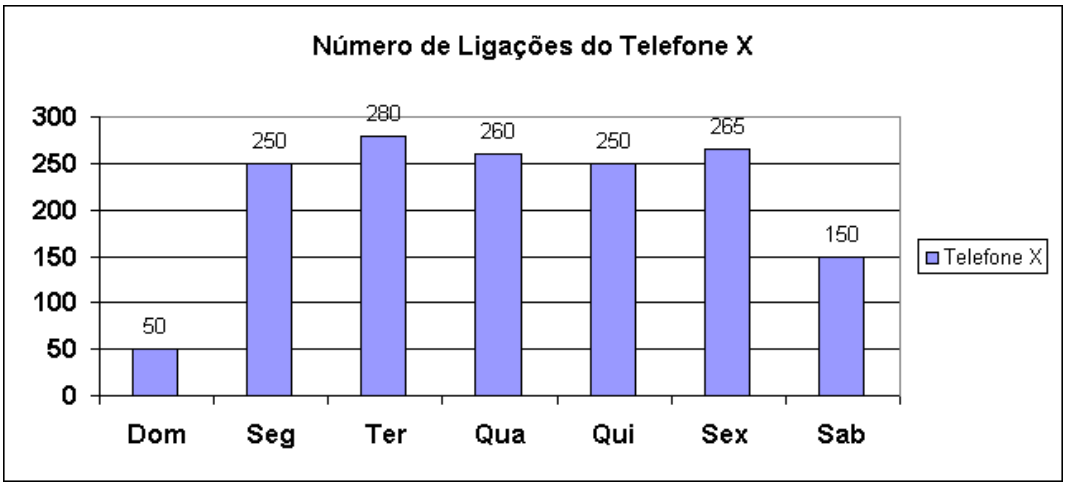

Figura 2.3: Número de ligações de um determinado telefone. Fonte: Ctbc Telecom

2.2.1 Objetivos da Análise de Séries Temporais

A tendência atual em analisar Séries Temporais visa, basicamente [31]:

1. Investigar o mecanismo gerador da série temporal;
2. Fazer previsões de valores futuros da série;
3. Descrever o comportamento da série;
4. Procurar periodicidades relevantes nos dados;
5. Identificar padrões.

2.2.2 Um Modelo para Representar Série Temporal

Baseado na definição (2.2.1), uma série temporal pode ser representada como uma seqüência ordenada de pontos e seus valores. Na representação proposta em [10] foi adicionado um identificador para diferenciar uma séria da outra. Se o *id* é único, uma série pode ser representada como :

$$< id, (t_0, v_0), ..., (t_{n-1}, v_{n-1}) > \tag{2.2.3}$$

Porém, não há nenhuma razão para restringir a série temporal de modo que ela armazene o valor de apenas um objeto de dado. Fazendo uma extensão da definição (2.2.1), podemos considerar que uma série temporal pode consistir em mais de um objeto de dados. Se $v^1, ..., v^k$ são objetos de dados que constituem uma série temporal, e v_i^j o valor do objeto de dados v^i no ponto t_j, uma representação geral é:

$$< id, (t_0, v_0^1, ...v_1^k), ..., (t_{n-1}, v_{n-1}^1, ...v_{n-1}^k) > \tag{2.2.4}$$

Embora seja completa, esta representação deve seguir as propriedades gerais de séries temporais. Estas propriedades são [10]:

1. *Os **tipo de valor** representam os tipos de dados e podem ser tipos simples, como inteiros e números reais, como, também, podem ser objetos complexos, como imagens.*

2. *O* ***domínio do tempo*** *e* ***granularidade****, ou seja, o tipo de pontos e a distância mínima entre eles. Nós podemos ter uma representação ordinal onde pontos sucessivos são representados por inteiros sucessivos (1,2,3,etc.), ou uma representação tipo calendário, onde a hierarquia de tempo (ano, mês, dia, etc.) é usada.*

3. ***Intervalo válido*** *é o intervalo válido dos pontos da série temporal.*

4. ***Regularidade****,ou seja, define se a série temporal tem um valor para cada ponto no intervalo* intervalo válido

2.2.3 Consultas em Séries Temporais

Um dos mais importantes benefícios da aplicação da ciência da computação na área médica é a consulta por similaridade baseada em conteúdo como ferramenta de apoio ao diagnóstico. Esta técnica tem sido usualmente aplicada na manipulação de imagens médicas (tomografia, mamografia, ressonância magnética,etc.) [17]. Ao se trabalhar com um Sistema Gerenciador de Banco de Dados contendo, por exemplo, cadastro de pacientes de um hospital ou cadastro de clientes de uma empresa de Telecomunicações, é comum algum critério de filtragem. Um exemplo simples de consulta seria "Obter os resultados dos exames de sangue de todos os pacientes com dengue que foram atendidos após o início do último verão", ou "Quais foram os clientes que adquiriram uma linha telefônica nos últimos quatro meses". Consultas como estas são muito comuns e são geralmente fáceis de serem realizadas, considerando que os dados são, na sua maioria, números inteiros, data e hora, seqüências de caracteres. No entanto, existem outros tipos de consultas que são realizadas utilizando outras técnicas. Por exemplo, na área médica, quando se trata de imagens, cadeias de DNA etc.., não faz sentido realizar consultas como "obter o cadastro dos pacientes com tumor no cérebro cuja tomografia seja igual à do paciente em estudo". Dificilmente as tomografias de dois tumores serão exatamente iguais, mesmo que os tumores tenham a mesma classificação. Portanto, o critério mais adequado para estes tipos de casos seria o da **semelhança ou similaridade**. Desta forma, a consulta acima faria mais sentido se fosse da forma: "obter o cadastro dos pacientes com tumor no cérebro cuja

tomografia seja bastante similar à do paciente em estudo"[17].

A avaliação da similaridade entre dados complexos, ou seja, objetos, é realizada através de funções que medem a distância entre eles. Então, dadas duas séries temporais $\vec{x} = \{x_0, x_1, ..., x_{n-1}\}$ e $\vec{y} = \{y_0, y_1, ..., y_{n-1}\}$, onde $x_1..x_n$ e $y_1..y_n$ são os valores da série temporal nos instantes $t_1..t_n$ correspondentes. Uma aproximação padrão é computar a distância Euclidiana $\vec{D}(\vec{x}, \vec{y})$ entre as séries $\vec{x}$ e $\vec{y}$ [9]

$$D(\vec{x}, \vec{y}) = \left(\sum_{i=0}^{n-1} |x_i - y_i|^2 \right)^{\frac{1}{2}} \tag{2.2.5}$$

Usando este modelo de similaridade, podemos recuperar séries temporais considerando a distância $D(\vec{x}, \vec{y})$. A distância euclidiana não é adequada quando se quer uma medida flexível entre séries temporais. As razões são as seguintes [48]:

- Duas séries podem ser muito similares mesmo que tenham escalas diferentes de amplitude. Se calculada a distância euclidiana, dará uma discrepância muito grande entre estas séries.

- A distância entre duas séries de tamanhos diferentes é indefinida, mesmo sendo semelhantes.

- Duas séries podem ser muito similares mesmo não estando perfeitamente sincronizadas. Se calculada a distância euclidiana, veremos uma grande divergência.

Devido a esta rigidez da distância euclidiana, algumas transformações nas séries podem ser necessárias [48, 10], tais como as transformações *shifting* e *scaling*.

Transformações *Shifting*

A distância euclidiana, sozinha, não nos dá uma medida intuitiva de similaridade. Se aplicarmos as transformações ignorando a defasagem (offset) no eixo Y (considerando que o eixo Y representa valores da série), conseguiremos uma boa análise de similaridade. Assim sendo, a transformação *Shifting* tenta corrigir defasagens

no eixo Y e é alcançada pela adição um valor a cada ponto y_i da série $\overline{x}$ (o que corresponde a deslocar a série no eixo Y de um valor α), ou seja:

$$\overline{x} = x_1 + \alpha, x_2 + \alpha, ...x_n + \alpha \tag{2.2.6}$$

Transformação *scaling*

É alcançada multiplicando cada ponto y_i da série por $\overline{x}$ um determinado valor, ou seja,

$$\overline{x} = x_1 * \alpha, x_2 * \alpha, ...x_n * \alpha \tag{2.2.7}$$

Normalização

É uma transformação que faz a série ficar invariante em relação à *shifting e scaling*. A forma normal de uma série $\overline{x}$ pode ser dada:

$$x_i = \frac{x_i - avg(\overline{x})}{std(\overline{x})} \tag{2.2.8}$$

onde, *avg* e *std* indicam a média e o desvio padrão respectivamente e x_i representa o i-ésimo elemento da série $\overline{x}$.

Este tipo de normalização é chamado de *normalização padrão* e é baseado em propriedades estatísticas da série. Ela converte a série em outra série, onde a média é igual a 0 e o desvio padrão é igual a 1[1].

Outro tipo de normalização é conhecido como *normalização por faixa fixa*. Ela também deixa a série invariante em relação a *shifting* e *scaling*, porém, força a série ter um valor máximo e um valor mínimo. A fórmula matemática é:

$$x_i = \frac{x_i - m_x}{M_x - m_x} \tag{2.2.9}$$

onde, M_x e m_x são o máximo e mínimo valores da série.

A figura 2.4, ilustra cada uma destas transformações, onde $\alpha = 3$ (*shifting*) e 25 (*scaling*). E a figura 2.5 mostra a mesma série normalizada, ou seja, foi aplicada a

[1]Se as séries temporais estiverem desalinhadas em relação ao eixo x (tempo), existe o *time warping*, que é a forma para calcular esta distância [10].

2.2.9 em cada ponto da série, deixando a nova série com valores entre 0 e 1. Esta série, após a normalização, está invariante a *shifting* e *scaling*, assim sendo, o cálculo da distância euclidiana torna-se mais flexível[2].

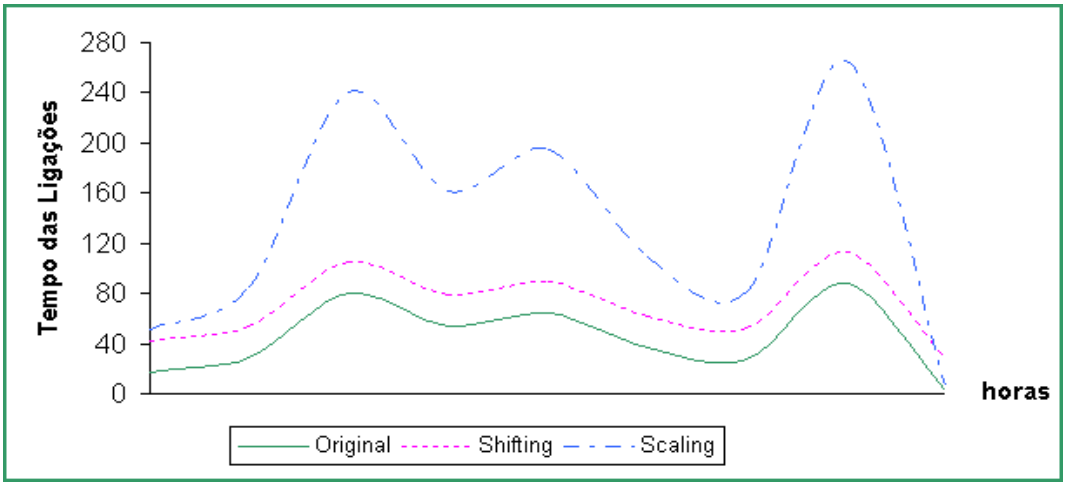

Figura 2.4: Exemplo de Transformações Shifting e Scaling em uma Série Temporal

Tipos mais comuns de consulta por similaridade

O cálculo da medida de similaridade entre duas séries temporais visa detectar o quão uma série é próxima (similar) à outra. Para tanto, ele pode ser efetuado segundo a estratégia *whole matching* ou *subsequence matching* [16]. As *Whole Matching* são aquelas em que, dado uma coleção de N seqüências de números reais $S_1, S_2, ..., S_N$ e uma consulta Q, deseja-se encontrar aquelas seqüências que estão a uma distância $\leq \varepsilon$ de Q. As seqüências e a consulta devem ter o mesmo tamanho, isto é, o mesmo número de pontos. As *Subsequence Matching* são aquelas em que, dada uma coleção de N seqüências de números reais $S_1, S_2, ..., S_N$ de tamanhos arbitrários, uma consulta Q e uma tolerância ε , deseja-se identificar as seqüências $S_i (1 \leq i \leq N)$ que estão a uma distância $\leq \varepsilon$ da consulta Q. Caso uma seqüência S_i e a consulta Q tenham tamanhos diferentes, para efeito de análise de similaridade divide-se S_i em partes que

[2]A utilização destas transformações, dependerá da aplicação.

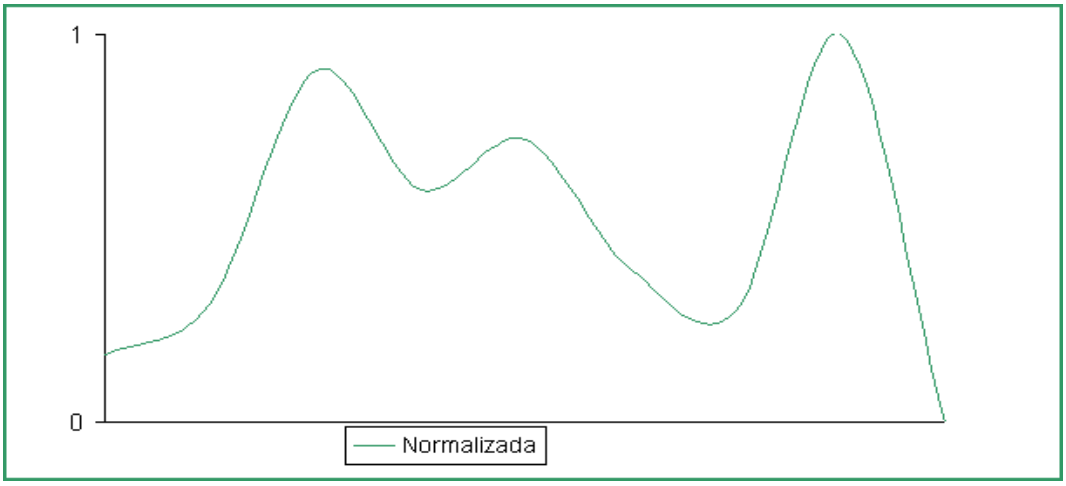

Figura 2.5: Exemplo da Normalização em uma Série Temporal

tenham o mesmo tamanho de Q. Tanto na *whole matching*, quanto na *subsequence matching* podem-se utilizar os tipos de consultas apresentadas a seguir.

Consulta por Faixa (*Range Query*)

Consulta que visa recuperar objetos que se encontram a uma distância máxima r (raio de busca), a partir do objeto de referência O (objeto de busca), como ilustrado na figura [2.6]. Formalmente, dado o conjunto de objetos S e um elemento qualquer deste

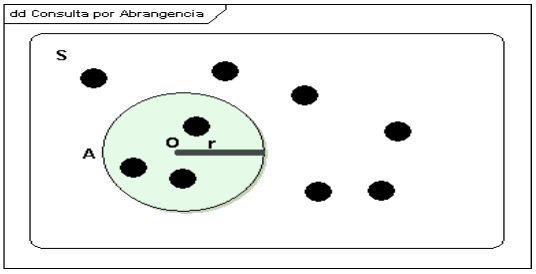

Figura 2.6: Exemplo de consulta por similaridade (Range Query)

conjunto, $s_i \in S$, uma consulta por faixa $RQ(o, r)$, pretende encontrar o subconjunto $A \subseteq S$ em que $A = \{s_i \in S | d(o, s_i) \leq r\}$. Como exemplo, podemos considerar novamente a consulta "obter o cadastro dos pacientes com tumor no cérebro cujo

grau de similaridade entre sua tomografia e a do paciente em estudo seja inferior a r", onde r é obtida através do cálculo da distância euclidiana. O valor aceitável para r deverá ser definido por um especialista da área.

Consulta dos k-Vizinhos mais próximos (*k-Nearest Neighbor Query*)

Consulta que visa recuperar os k objetos mais próximos ao objeto de referência O, como ilustrado na figura [2.7]. Formalmente, dado o conjunto de objetos S e um

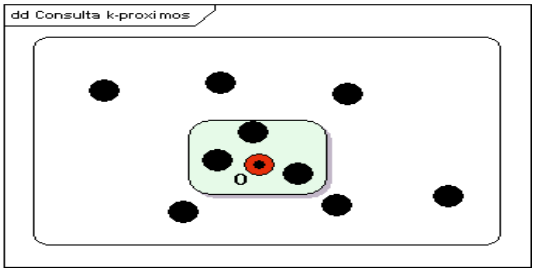

Figura 2.7: Exemplo de consulta por similaridade (K-Vizinhos)

elemento qualquer deste conjunto, $s \in S$, uma consulta dos k-vizinhos mais próximos $kNNQ(o,k)$ pretende encontrar o subconjunto $A \subseteq S$ em que $A = \{s \in S | k = |A|$ e $\forall a_i \in A, \forall x \in [S - A], d(o,a_i) \leq d(o,x)\}$. Como exemplo podemos considerar a consulta "obter o cadastro dos pacientes com tumor no cérebro cujas tomografias sejam as k mais próximas à do paciente em estudo".

Problemas encontrados em consultas sobre Séries Temporais

Consultas por similaridade são freqüentemente usadas para explorar séries temporais em banco de dados, como parte de aplicações *Knowledge Discovery Database* (KDD) como clusterização, classificação e regras de associação em data mining [24]. Normalmente, as séries temporais em banco de dados envolvem uma grande quantidade de dados. Devido a tais volumes de informações, muitas pesquisas estão sendo feitas para melhorar o tempo de processamento necessário para que uma consulta sobre as séries temporais possa ser realizada em tempo aceitável.

Indexação

Um índice é uma estrutura organizada de dados que permite a busca de informações rapidamente. Sem a utilização dos índices, as consultas em séries temporais seriam extremamente ineficientes em termos de custo de CPU e IO [48]. Como a maioria das séries temporais tem uma grande dimensionalidade, ou seja, uma grande quantidade de pontos, seria ineficiente indexá-las diretamente [36]. Os métodos mais promissores são aqueles que utilizam a técnica de reduzir a dimensão do dado (detalhado na próxima seção), e então usa métodos para indexar os dados a partir do novo espaço dimensional [24].

Séries temporais podem ser indexadas usando métodos já conhecidos como *R-tree* e suas variantes [10]. Estes métodos visam aumentar o desempenho das consultas, uma vez que são muito mais rápidos do que aqueles que utilizam a busca linear, isto é, seqüêncial nos dados.

O *R-Tree* estende o popular *B-Tree* (utiliza apenas 1 dimensão) para um número maior de dimensão [48]. Se for bem implementado, ele é uma forma eficiente de indexação n-dimensional incluindo pontos e regiões[3]. Similarmente ao *B-Tree*, o *R-Tree* é uma árvore balanceada com os índices dos dados armazenados nas folhas. Na *B-Tree*, cada nó que não é folha da árvore corresponde a um intervalo. Estendendo esta idéia para multi-dimensões, cada nó não folha no *R-Tree* corresponderá a intervalos multi-dimensionais, chamados de menores retângulos (*minimum bounding boxes-MBR*). O MBR é o objeto base do *R-Tree*. No *B-Tree*, o intervalo associado com os nós incluem todos os intervalos associados com os nós filhos; no *R-Tree*, o MBR associado ao nó também inclui todos os MBR's de seus nós filhos. No *B-tree*, o intervalo associado a um nó, não sobrepõe os intervalos associados com nós irmãos. Desta forma, o número de nós a serem acessados na pesquisa utilizando o *B-Tree* dependerá da profundidade da árvore. No *R-Tree*, entretanto, os MBR's associados aos nós podem ser sobrepostos com MBR's de nós irmãos. Desta forma, a pesquisa

[3] A qualidade da implementação é um fator crítico para o sucesso [48].

no índice poderá ter que percorrer vários caminhos diferentes.

Na figura 2.8, pode-se ver um exemplo de um conjunto de retângulos e seus MBR's. Por simplicidade é apresentada uma figura que considera apenas duas dimensões. A estrutura correspondente a um *R-Tree* pode ser vista na figura 2.9.

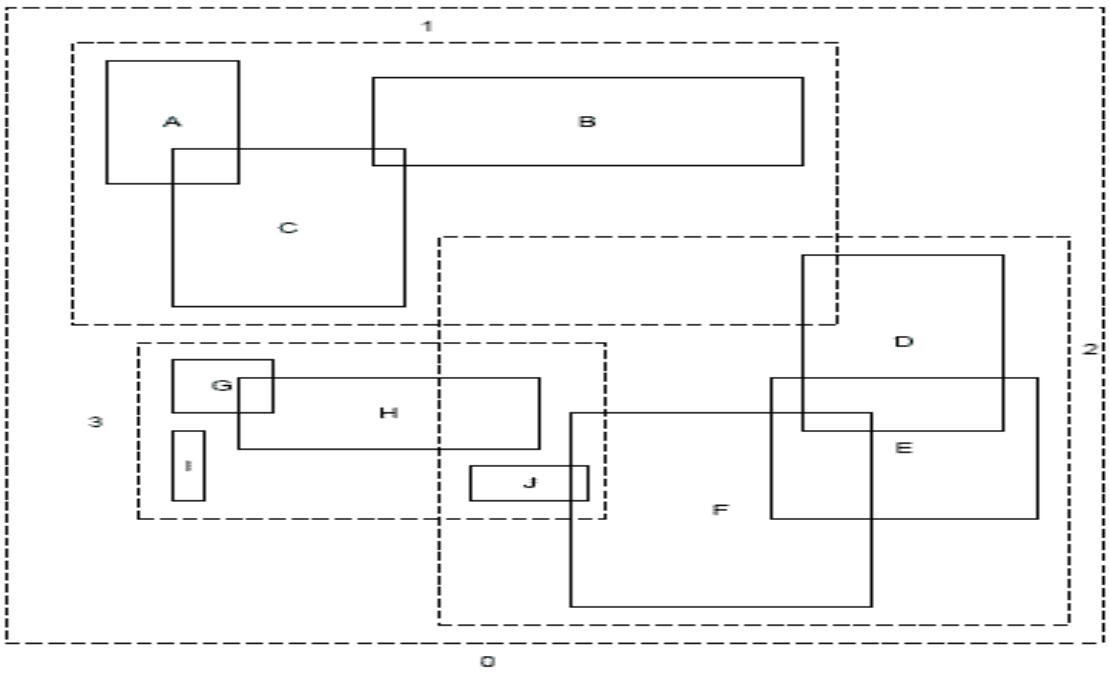

Figura 2.8: Exemplo de MBR's

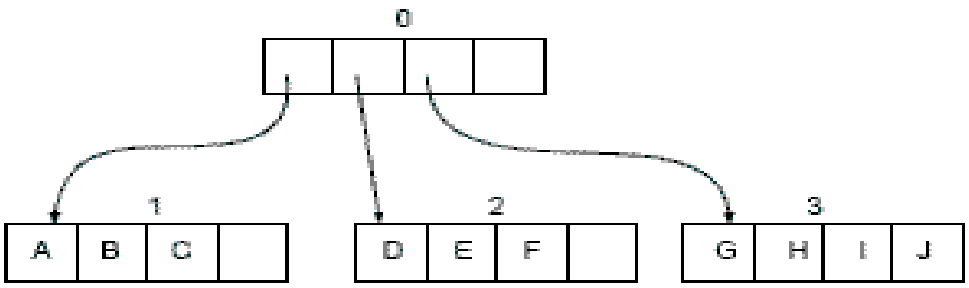

Figura 2.9: A estrutura *R-Tree*

A consulta em um índice *R-Tree* pode ser lenta, pois vários caminhos diferentes poderão ser analisados.

Em geral, qualquer método de indexação deve suportar dois problemas: *Falso-Negativo e o Falso-Positivo*. Na análise de séries temporais, *O Falso-Positivo* ocorre quando, na realização de uma consulta, as séries que parecem ser próximas no índice, na verdade não o são, enquanto que o *Falso-Negativo* ocorre quando as séries não são consideradas na pesquisa por aparentemente estarem distantes no índice, sendo que, na realidade, são próximas. A existência do *Falso-Negativo* implica em perda de informações, fato que não ocorre com o *Falso-Positivo*. Entretanto, o *Falso-Positivo* gera uma perda de desempenho, pois deve haver um pós-processamento para retirar as séries selecionadas por meio do *Falso-Positivo*.
O processo de criação de um índice deve ter as seguintes propriedades [16]:

1. Deve ser rápido, ou seja, mais rápido do que uma busca seqüencial;
2. Deve ser correto, ou seja, não pode permitir *Falso-Negativo*. O *Falso-Positivo* é menos problemático porque pode ser tratado no pós processamento;
3. Deve requerer um pequeno espaço de *overhead*;
4. Deve ser dinâmico, permitindo que a inserção e a exclusão de seqüências sejam feitas facilmente, sem que se necessite reconstruir todo índice.
5. Deve tratar seqüências de vários tamanhos.

Em [24] foram propostas duas novas propriedades:

1. O índice deve ser construído em um tempo aceitável;
2. O índice deverá ser capaz de tratar diferentes medidas de similaridade;

Uma série temporal X pode ser considerada como um ponto em um espaço n-dimensional. Isto significa que um índice pode ser construído para a série temporal utilizando um Método de Acesso Espacial (MAE), como por exemplo, o método *R-tree* e suas variantes. Entretanto, tais métodos começam a se degradar rapidamente

se as dimensões forem maior que o intervalo compreendido entre 7 e 12 [24], e consultas reais podem conter um número muito maior do que estes. Para utilizar o MAE é necessário, primeiramente, que ocorra uma redução de dimensão[4].
Um método para fazer uma redução de dimensão foi proposto em [24]:

1. Estabelecer a métrica da distância (como por exemplo, a distância Euclidiana);
2. Produzir a redução da dimensão que reduz a dimensão do dado de n para N, onde N pode ser eficientemente trabalhado pelo MAE escolhido;
3. Produzir a medida da distância definida no espaço de dimensão N e que obedece a expressão : $D_{esp.reduzido}(A, B) \leq D_{esp.real}(A, B)$ onde A e B são séries temporais.

Como já visto anteriormente, para que tenhamos pesquisas em séries temporais mais eficientes, temos que realizar a redução da dimensão do dado. Diversas técnicas têm sido utilizadas, com êxito, no propósito de reduzir a dimensão dos dados, tais como a *Discrete Fourier Transform* (DFT), *Discrete Wavelet Transform* (DWT), *Singular Value Decomposition* (SVD) e *Piecewise Aggregate* Aproximation (PAA). Tais técnicas são apresentadas, resumidamente, a seguir.

2.3 Técnicas de Redução de Dados

Como dito anteriormente, um grande problema em mineração de dados em séries temporais é a dimensão dos dados. Por esta razão, o desempenho de muitos métodos de indexação em séries temporais se deterioram com grandes volumes de dados [32]. Para que se possa realizar a redução da dimensão do dado, é necessário que se faça uma compactação das séries, onde, através de poucos pontos, consiga-se uma boa representatividade.

[4]Recomendável para aplicações que utilizam séries temporais com muitos pontos. Para séries muito pequenas, pode-se utilizar o MAE sem redução da dimensão.

2.3.1 Discrete Fourier Transform

A primeira técnica proposta para redução de dimensão em séries temporais foi introduzida em [2] e usa a *Discrete Fourier Transform* (DFT) como função de extração e a Distância Euclidiana como a métrica da distância. A definição formal do método pode ser encontrada em [2]. A DFT transforma o dado de seu domínio original para um domínio de freqüências. Uma visão geral do método é descrito a seguir. Os primeiros k coeficientes da DFT de cada seqüência no banco de dados são calculados e é construído o índice multidimensional usando os k coeficientes de cada seqüência transformada. O índice pode ser construído usando qualquer método de acesso espacial.

2.3.2 Wavelets

Nesta seção será apresentada primeiramente uma visão geral sobre *Wavelet*. Posteriormente, será apresentada a técnica de redução dos dados usando as *Wavelets* (DWT).

Transformada *Wavelet*

Wavelets são funções que satisfazem certos requisitos matemáticos e são usadas para representar dados ou outras funções [1]. A análise de dados de acordo com escalas variáveis no domínio do tempo e da freqüência é a idéia básica da utilização da teoria das *Wavelets*. As *Wavelets* são funções matemáticas que ampliam intervalos de dados, separando-os em diferentes componentes de freqüência, permitindo a análise de cada componente em sua escala correspondente. Uma função *Wavelet* adequada a um conjunto de dados permite uma representação esparsa desse conjunto. Isto torna as *Wavelets* uma excelente ferramenta de compressão de dados. A idéia sobre a qual se baseiam as *Wavelets* não é nova. *Wavelets* tiveram uma história científica não muito comum, marcada por descobertas independentes. A partir de 1980, os estudos sobre *wavelets* tiveram um avanço significativo [34]. *Wavelets* estão sendo campo de

Histórico	
Ano	Acontecimento
1807	Jean Baptiste Joseph Fourier diz que qualquer função periódica pode ser expressa como uma soma infinita de ondas seno e coseno de várias freqüências
1909	Alfred Haar, um matemático Húngaro, descobriu uma função base que é reconhecido como a primeira *wavelet*. Ela consiste de pequenos pulsos positivos e negativos
1985	Yves Meyer da Universidade de Paris descobriu a primeira *smooth orthogonal wavelet*
1987	Ingrid Daubechies construiu a primeira *smooth orthogonalwavelet* com suporte de compactação. Possibilitou a utilização prática destes *wavelets*

Tabela 2.1: Breve histórico sobre *wavelets*

pesquisas nas mais diversas áreas, como por exemplo: compactação de imagens, data mining, análise de séries temporais [10] [4][5] , processamento de sinais [26], também em medicina e biologia [47]. Apesar de as DFT's serem uma técnica muito utilizada na redução de dados [9], a vantagem das *Wavelets* sobre as DFT's, é que a base das funções de Fourier são dependentes da freqüência mas não do tempo, ou seja, pequenas alterações no domínio da freqüência produzem alterações em todo o domínio do tempo. As *Wavelets* são dependentes de ambos os domínios, da freqüência (via dilação) e do tempo (via translação), o que é uma vantagem em diversos casos. As bases das funções de Fourier são impróprias para o tratamento local de dados, pois são séries infinitas e não se adaptam à análise de dados descontínuos. As *Wavelets* não somente se prestam à aproximação de funções finitas, como também servem para análise de dados descontínuos. *Wavelet Transform* (WT) ou *Discrete Wavelet Transform* (DWT) estão sendo utilizadas em substituição à DFT em diversas aplicações (computação gráfica,imagens etc). Particularmente, as *Haar Wavelets* têm sido aplicadas

[5]Em [10] as *Wavelets* são usadas para reduzir o espaço dimensional das séries temporais e em [4] as *Wavelets* são utilizadas para fazer a previsão de vendas através de análises da tendência e da sazonalidade de séries temporais.

com êxito, como descrito em [9]. Uma *Haar Wavelet* utiliza a DWT para redução da dimensão de dados em séries temporais. Uma motivação para a utilização das *Haar Wavelets* é o fato de que elas permitem uma boa aproximação da seqüência original com poucos coeficientes, preservam a distância Euclidiana e podem ser computadas de modo fácil e rapidamente. A complexidade das *wavelets* é *O(n)*, enquanto que a complexidade da DFT é $O(nlogn)$. Além disso, elas superam, em relação ao desempenho, os índices criados usando a DFT [9]. Uma desvantagem da utilização das *Wavelets* é referente ao tamanho das seqüências, pois elas devem ser um inteiro que seja potência de dois [9]. Caso não sejam, faz-se necessário um pré-processamento que acrescente pontos com valor igual a 0, até que se consiga uma potência de 2, conforme será explicado no Capítulo 4. A definição formal das *Haar Wavelets* pode ser encontrada em [48].

A título de exemplo, a seguir se apresenta uma aplicação das *Haar Wavelets*. Suponha a seguinte série de tamanho 8:

$$S = \{1, 3, 5, 11, 12, 13, 0, 1\}$$

Para realizar a transformação, primeiramente calcula-se a média entre os pares adjacentes (obtendo os coeficientes de aproximação),ou seja,

$$S_{resolucao3} = (\frac{1+3}{2}, \frac{5+11}{2}, \frac{12+13}{2}, \frac{0+1}{2})$$

A cada nível de resolução calculado, a série (ou sinal) é dividida em duas componentes: a baixa freqüência (coeficientes de aproximação) e a alta freqüência (coeficientes de detalhes). Nota-se que o coeficiente de detalhe indica o quanto o coeficiente de aproximação se distancia do ponto real.

Para se reconstruir a série é necessário armazenar os coeficientes de detalhe, que são calculados como sendo as diferenças entre os números adjacentes dividido por 2, isto é,

$$D_{resolucao3} = (-1, -3, -0.5, -0.5)$$

Este processo se repete até que se encontre a resolução 1, onde aparecerá apenas uma média e um coeficiente de detalhe. Na prática, ao invés de se utilizar a média para se calcularem os coeficientes *wavelets*, utiliza-se um fator de normalização $\frac{1}{\sqrt{2}}$.
Este fator é importante quando os coeficientes *wavelets* são usados na redução dos dados. Ele faz com que a distância euclidiana seja preservada no domínio das *wavelets* e no domínio do tempo [9] (o que não seria garantido com o uso da média, tal como no exemplo anterior). As tabelas 2.2 e 2.3 ilustram o cálculo baseado no fator $\frac{1}{\sqrt{2}}$.

Resolução	a_1	a_2	a_3	a_4	a_5	a_6	a_7	a_8
3	$\frac{a_1+a_2}{\sqrt{2}}$	$\frac{a_3+a_4}{\sqrt{2}}$	$\frac{a_5+a_6}{\sqrt{2}}$	$\frac{a_7+a_8}{\sqrt{2}}$	$\frac{a_1-a_2}{\sqrt{2}}$	$\frac{a_3-a_4}{\sqrt{2}}$	$\frac{a_5-a_6}{\sqrt{2}}$	$\frac{a_7-a_8}{\sqrt{2}}$
2	$\frac{a_1+a_2+a_3+a_4}{2}$		$\frac{a_5+a_6+a_7+a_8}{2}$		$\frac{(a_1+a_2)-(a_3+a_4)}{2}$		$\frac{(a_5+a_6)-(a_7+a_8)}{2}$	
1	$\frac{a_1+a_2+a_3+a_4+a_5+a6+a_7+a_8}{2\sqrt{2}}$				$\frac{(a_1+a_2+a_3+a_4)-(a_5+a6+a_7+a_8)}{2\sqrt{2}}$			

Tabela 2.2: Exemplo da decomposição das *Haar Wavelets*, com fator de normalização $\frac{1}{\sqrt{2}}$. Este fator é usado para levar em consideração a importância dos coeficientes em relação à reconstrução da série temporal [23].

	Médias				Detalhes			
4	1	3	5	11	12	13	0	1
3	2.8284	11.3137	17.6777	0.7011	-1.4142	-4.2426	-0.7071	-0.7071
2	10.0000		13.0000		-6.0000		12.0000	
1	16.2635				-2.1213			

Tabela 2.3: Exemplo da decomposição das *Haar Wavelets*

No exemplo apresentado, a DWT ($c\ d_0^0\ d_0^1\ d_1^1\ d_0^2\ d_1^2\ d_2^2\ d_3^2$), onde c e d são o coeficiente de aproximação e detalhe respectivamente, da série S é: = (16.2635, -2.1213, -6.0000, 12.0000, -1.4142, -4.2426, -0.7071, -0.7071)
Para se utilizarem as *Haar Wavelets* na redução da dimensão dos dados, precisa-se selecionar os coeficientes que melhor representam a série original. Tal seleção pode

ser feita, por exemplo, através da técnica da *Extração de Características (Feature Extraction).* No caso dos *Haar Wavelets*, a *Feature Extraction* mais utilizada é aquela que seleciona os primeiros coeficientes, pois estes tem uma boa representatividade da série original. Desta forma, será considerada a tendência da série temporal, mas perdem-se algumas informações [48]. Uma outra forma de *Feature Extraction* é selecionar os coeficientes com os valores mais significativos. Esta é a melhor maneira de reter a energia da série temporal [48]. Intuitivamente, a energia associa um valor às grandezas que a série temporal representa. Assim sendo, quanto maior a energia, maiores as intensidades das grandezas representadas nas séries. Por exemplo, se as séries representarem em seu eixo das ordenadas o número de ligações telefônicas de um cliente no tempo, quanto maior for o volume de ligações deste cliente, maior será a energia da série que o representa. Um ponto importante a ser considerado é que a transformada *wavelet* é ortogonal e também preserva a energia da série temporal. Dado uma série temporal $\overline{x} = (x(1), x(2), ...x(n))$ e seus coeficientes *wavelets* $\overline{X} = (X(1), X(2)...X(n))$, temos:

$$En(\overline{x}) = En(\overline{X}) = \sum_{i=1}^{n} X^2(i).$$

Então, a melhor maneira de preservar a energia da série com apenas k coeficientes DWT, onde $k < n$, é manter os k coeficientes mais significativos, isto é, os de maiores valores absolutos. A figura 2.10 ilustra exemplos de feature extraction [48].

2.3.3 Principal Component Analysis - Singular Value Decomposition (SVD)

Principal Component Analysis é uma outra técnica de redução de dimensão que transforma um conjunto de m pontos co-relacionados em um conjunto de $k \leq m$ não relacionados e mantém a variância do conjunto original. Este método se diferencia dos demais em um importante ponto: enquanto os outros fazem transformações locais, ou seja, eles aplicam a transformação em cada seqüência de dados independentemente,

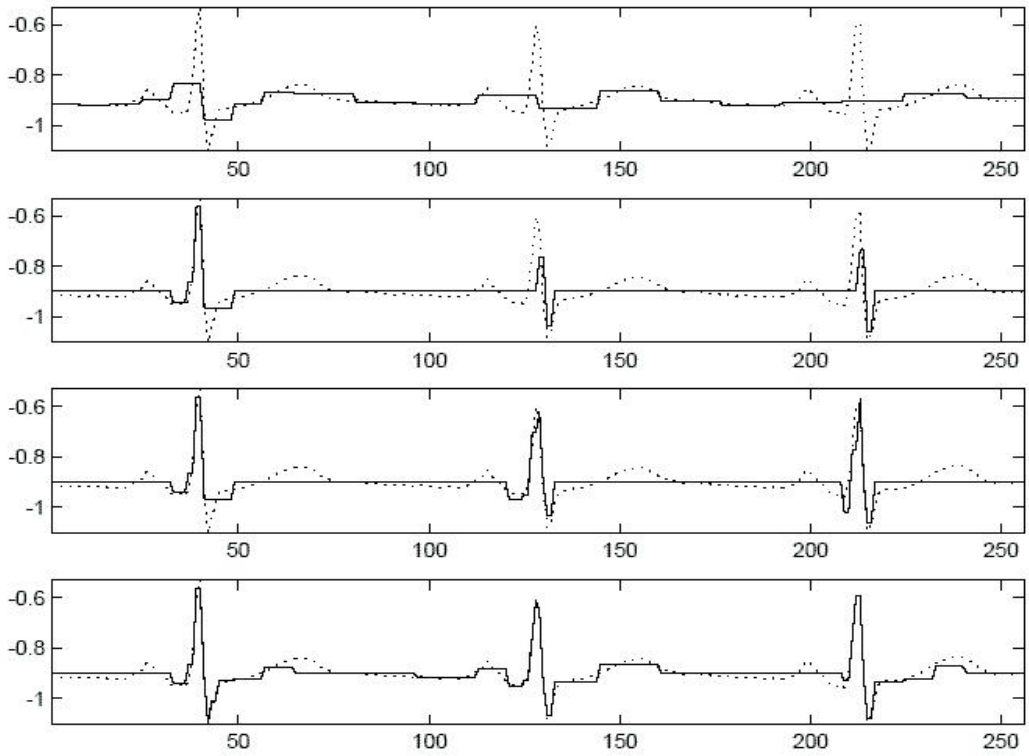

Figura 2.10: Aproximações de uma série temporal (representada na linha tracejada). De cima para baixo, (a)-Primeiros 20 coeficientes (*Haar Wavelet*); (b) - Os 5 coeficientes mais significativos; (c) - Os 10 coeficientes mais significativos; (c) - Os 20 coeficientes mais significativos.

o SVD é global e funciona, simultaneamente, em todo o conjunto de dados. A complexidade deste método em relação ao tempo é $O(m.n^2)$ e, em relação ao espaço, é $O(m.n)$. Uma grande desvantagem deste método ocorre quando uma atualização no banco de dados se faz necessário, pois neste caso todo o índice deve ser reconstruído [10].

2.3.4 Piecewise Aggregate Aproximation (PAA)

O PAA foi introduzido em [24]. Este método foi concebido a partir da observação de que na maioria das séries temporais, os dados podem ser computados segmentando-se a série temporal em segmentos de mesmo tamanho e armazenando-se a média destes segmentos. Os valores das médias podem então ser indexados eficientemente

em um espaço dimensional menor. Suponha-se uma série temporal $\overrightarrow{X} = \{x_1, ...x_n\}$, e um conjunto de séries temporais que constituem o banco de dados $\overrightarrow{Y} = \{Y_1, ...Y_k\}$. Suponha-se, também, que N é a dimensão do dado no espaço transformado. Uma série temporal X de tamanho n é representada no espaço dimensional N pelo vetor $\overline{X} = \overline{x_1}, ..., \overline{x_N}$. O i-ésimo elemento de $\overline{X}$ é calculado pela equação:

$$\overline{X_i} = \frac{N}{n} \sum_{j=\frac{n}{N}(i-1)+1}^{\frac{n}{N}i} x_j$$

Como exemplo, considere

$$\overline{X} = (-1, -2, -1, 0, 2, 1, 1, 0), n = |X| = 8, N = 2$$

Logo, como a dimensão (N) é igual a 2, tem-se: $X =$ (média(-1,-2,-1,0), média(2,1,1,0)) $\Rightarrow X =$ (-1,1).
Em [24], além do *Piecewise Aggregate Approximation (PAA)*, é sugerida a utilização do método *GEneric Multimedia INdexIng* (GEMINI). O GEMINI requer apenas os três passos seguintes:

1. *Estabelecer a métrica da distância (como distância Euclidiana);*
2. *Produzir a redução da dimensão que reduz a dimensão do dado de* n *para N, onde N pode ser eficientemente trabalhado pelo MAE escolhido;*
3. *Produzir a medida da distância definida no espaço de dimensão N e que obedece a expressão:* $D_{esp.reduzido}(A, B) \leq D_{esp.real}(A, B)$*;*

2.4 Processo de Redução da Dimensão dos Dados e de Indexação das Séries Temporais

Como dito anteriormente, um grande problema em mineração de dados em séries temporais é a dimensão dos dados. Por esta razão, o desempenho de muitos métodos de indexação em séries temporais se deterioram para grandes volumes de dados [32]. Para que se possa realizar a redução da dimensão do dado, os seguintes passos devem ser considerados (estes passos dependem da aplicação):

1. Aplica-se um determinado método para geração dos coeficientes sobre as séries, por exemplo, DWT, DFT, PCA etc;

2. Aplica-se um método de *Feature Extraction Vector* sobre os coeficientes obtidos no passo anterior, de modo a selecionar os melhores coeficientes e, conseqüentemente, reduzir a dimensão dos dados;

3. Aplica-se um método de indexação sobre os coeficientes remanescentes da aplicação da função de *Feature Extraction*. Um exemplo de método de indexação é o *R-Tree*, citado anteriormente.

2.5 Sistemas Baseados em Conhecimento

Sistemas de Conhecimento são programas de computador que tentam resolver problemas que os seres humanos resolveriam emulando o raciocínio de um especialista, aplicando conhecimentos específicos e inferências.
Sistemas de Conhecimento são planejados para adquirir e disponibilizar o conhecimento operacional de um especialista humano [3].
Segundo [19], um Sistema Baseado em Conhecimento é um programa inteligente de computador que usa conhecimentos e procedimentos inferenciais para resolver problemas que são bastante difíceis, de forma a requererem, para sua solução, muita perícia humana.

Basicamente, um Sistema Baseado em Conhecimento é composto por um módulo do conhecimento e um módulo de recuperação de informações (máquina de inferência) que serão apresentados a seguir.

2.5.1 Módulo do Conhecimento

O Módulo do Conhecimento é utilizado para significar a coleção de conhecimento do domínio, ou seja, as informações necessárias para resolver um problema. Assim sendo, tal conhecimento precisa ser organizado de uma maneira adequada para que a máquina de inferência consiga tratá-lo convenientemente. Os conhecimentos de um Sistema Baseado em Conhecimento consistem de fatos e heurísticas.

O conhecimento em um Sistema Baseado em Conhecimento consiste de fatos e heurísticas. Os fatos constituem um corpo de informação que é largamente compartilhado, publicamente disponível e geralmente aceito pelos especialistas em um campo. A coleção de fatos disponíveis em um sistema é chamada base de fatos (ou memória de trabalho) do sistema. As heurísticas são, em sua maioria, regras poucos discutidas, de bom discernimento (regras de raciocínio plausível, regras de boa conjectura), que caracterizam a tomada de decisão a nível de especialista na área. A coleção de regras é chamada de base de conhecimento [27]. O nível de desempenho de um Sistema Baseado em Conhecimento é função, principalmente, do tamanho e da qualidade do banco de conhecimento que possui.

Os Sistemas de Conhecimentos provêem estratégias e mecanismos para processarem fatos em relação ao estado de um dado ambiente, derivando inferências lógicas a partir destes fatos [27].

Aquisição do Conhecimento

Uma das tarefas mais importantes, ou pelo menos aquela que demandará maior atenção, é, certamente, a aquisição do conhecimento. Ela não pode limitar-se à adição

de novos elementos de conhecimento à base de conhecimentos, devendo, também, integrar o novo conhecimento ao conhecimento previamente adquirido através da definição de relações entre os elementos que constituem o novo conhecimento e os elementos já armazenados na base. Outro ponto importante é o tratamento de incoerências. Dependendo da forma como o novo conhecimento é adquirido, pode haver erros de aquisição. Estes erros podem ser resultados da própria natureza do conhecimento, como em dados obtidos através de sensores sujeitos a ruídos, ou podem ser gerados pela interface humana existente entre o mundo real e o sistema de representação. Existem técnicas que tentam evitar estes tipos de erros. Uma base de conhecimento pode também ser analisada periodicamente com a finalidade de detectar incoerências eventualmente introduzidas no processo de aquisição. Por fim, deve-se observar que a adequação do formalismo de representação ao tipo de conhecimento do mundo real a ser representado é fundamental para a eficiência do processo de aquisição.

A Representação do Conhecimento

Uma outra importante etapa no projeto de um Sistema Baseado em Conhecimento é a escolha do método de representação do conhecimento. A linguagem associada ao método deve ser suficientemente expressiva para permitir a representação do conhecimento a respeito do domínio escolhido de maneira completa e eficiente. Há diversas técnicas de representação do conhecimento, tais como [41, 30]:

Regras de Produção: representa o conhecimento como em linguagens de programação lógica, ou seja, em regras *Se Condição Então Conclusão*;

Redes Semânticas: representa o conhecimento como classes de objetos distribuídos como nós em um grafo. Estes nós são organizados em uma estrutura taxonômica, e as ligações (arcos) entre os nós representam ligações binárias;

Quadros (frames): assim como nas Redes Semânticas, o conhecimento é representado como classes de objetos organizadas em uma estrutura taxonômica e interligadas por relações binárias, porém, as ligações entre objetos são feitas por meio de atributos (*slots*);

Sistemas Lógicos Descritivos: representa o conhecimento através da definição de termos e de relações entre eles, baseando-se para tanto, em estruturas taxonômicas.

Normalmente, os Sistemas de Conhecimentos utilizam apenas um único tipo de representação de conhecimento. Alguns Sistemas de Conhecimentos possuem os chamados sistemas híbridos de representação de conhecimento [33, 44] que, além de possuir diversos formalismos de representação, dispõem também de algoritmos de acesso que integram os conhecimentos representados nos diversos formalismos para permitir sua utilização de maneira integrada.

Os Sistemas de Conhecimento que utilizam regras de produção como ferramenta de representação do conhecimento se tornaram popular por razões como, por exemplo, sua natureza modular, o encapsulamento do conhecimento bem como sua expansão, a facilidade de explanação (isso porque os antecedentes de uma regra são necessários para desencadear sua ativação). Além disso, há a similaridade com o processo cognitivo do ser humano, ou seja, as regras parecem ser um modo natural de se modular o raciocínio humano na busca da solução de um problema. A simples representação da notação *Se....Então* para a materialização das regras torna fácil a representação do conhecimento do especialista na forma de regras de produção. Embora as regras de produção sempre tenham sido muito utilizadas na implementação de sistemas de conhecimentos, a maneira com que elas foram utilizadas nos primeiros deles não se mostrou adequada pela falta de uma estratégia de controle das regras que tornasse seu processamento mais eficiente. Daí o aparecimento de técnicas alternativas que visam a aumentar a eficiência da representação e da recuperação do conhecimento por meio

de regras, tal como o RDR apresentado na seção 2.6. Tal ganho de eficiência será enfocado no Capítulo 4.

2.5.2 A máquina de Inferência (Recuperação da Informação)

A máquina de inferência representa o meio pelo qual o conhecimento é manipulado, utilizando as informações armazenadas na base de conhecimento para resolver problemas. Para isto, deve haver uma linguagem ou um formato específico no qual o conhecimento possa ser expresso para permitir o raciocínio e inferência. A máquina de inferência tenta imitar os tipos de pensamento que o especialista humano emprega quando resolve um problema, ou seja, ele pode começar com uma conclusão e procurar uma evidência que comprove, ou pode iniciar com uma evidência para chegar a uma conclusão. No caso dos sistemas baseados em regras de produção, existem, basicamente, dois **Métodos de Raciocínio** aplicáveis às regras: *encadeamento progressivo* e o *encadeamento regressivo*.

No ***encadeamento progressivo***, a parte esquerda da regra é comparada com a descrição da situação atual, contida na memória de trabalho. As regras que satisfazem a esta descrição têm sua parte direita executada, o que, em geral, significa a introdução de novos fatos na memória de trabalho. Tal estratégia é usada, preferencialmente, em prognósticos, monitoramento e controle de aplicações [27].

No ***encadeamento regressivo*** o comportamento do sistema é controlado por uma lista de objetivos. Um objetivo pode ser satisfeito diretamente por um elemento da memória de trabalho, ou podem existir regras que permitam inferir algum dos objetivos correntes, isto é, que contenham uma descrição deste objetivo em suas partes direitas. As regras que satisfazem esta condição têm as instâncias correspondentes às suas partes esquerdas adicionadas à lista de objetivos correntes. Caso uma dessas regras tenha todas as suas condições satisfeitas diretamente pela memória de trabalho, o objetivo em sua parte direita é também adicionado à memória de trabalho. Um objetivo que não possa ser satisfeito diretamente pela memória de trabalho, nem

inferido através de uma regra, é abandonado. Quando o objetivo inicial é satisfeito, ou não há mais objetivos, o processamento termina. Essa estratégia é usada, preferencialmente, em problemas de diagnósticos [27].
Na próxima seção serão resumidas diversas outras alternativas de técnicas de recuperação da informação.

2.5.3 Tipos de Sistemas de Conhecimentos com relação à Representação do Conhecimento e à Recuperação da Informação

Em [41], os sistemas de conhecimento são classificados conforme seu objetivo em resolver diferentes tipos de problemas, e são agrupados em quatro principais categorias:

Provadores de Teoremas e Linguagens de Programação Lógicas: Os provadores de teoremas usam os métodos de resolução para provar sentenças da lógica de primeira ordem, freqüentemente, em tarefas de raciocínio matemáticos ou científicos. As linguagens de programação lógicas, tipicamente, restringem o tratamento que a lógica dispensa à negação, à disjunção e ou ao conceito de igualdade. Elas utilizam o raciocínio regressivo e podem incluir alguns elementos não lógicos, como, por exemplo, entrada e saída. Como exemplo de provadores têm-se: SAM, AURA, OTTER. Como exemplo de linguagens lógicas, têm-se: Prolog, MRS, LIFE;

Sistema de Produção: Como nas linguagens lógicas, estes sistemas utilizam-se das implicações (regras de produção) como sua forma básica de representação. O conseqüente de cada implicação é interpretado como uma ação a ser tomada, ao invés de representar uma simples conclusão lógica. Ações incluem inserções e remoções da base de conhecimento. Sistemas de Produção trabalham com o raciocínio progressivo. Alguns destes sistemas têm mecanismos de resolução de conflitos para decidir quais ações serão desencadeadas quando diversas são

recomendadas simultaneamente. Como exemplo, têm-se: CLIPS, SOAR e OPS-5;

Redes Semânticas e *Sistemas de Quadros (*Frame Systems*):* Os sistemas que utilizam as redes semânticas utilizam-se de objetos distribuídos como nós em um grafo. Estes nós são organizados em uma estrutura taxonômica. As ligações (arcos) entre os nós representam ligações binárias. Nos *Frame Systems*, assim como nos sistemas de redes semânticas, o conhecimento é representado como classes de objetos organizadas em uma estrutura taxonômica e interligadas por relações binárias. Porém, as ligações entre objetos são feitas por meio de atributos (*slots*). Como exemplo de sistemas de rede semântica têm-se: SNEPS, NETL, Conceptual Graphs e como exemplo dos *Frame Systems* têm-se: OWL, FRAIL e KODIAK;

Sistemas Lógicos Descritivos: Estes sistemas evoluíram das redes semânticas. A idéia é expressar e raciocinar com definições complexas e relações entre objetos e classes. Para tanto, representam o conhecimento através da definição de termos e de relações entre eles, baseando-se em estruturas taxonômicas.

Sistemas que lidam com Incerteza: utilizam ferramentas tais como o método Bayesiano, teoria de Dempster-Shafer, teoria dos conjuntos nebulosos etc. De uma maneira geral, os métodos que lidam com incerteza atribuem aos fatos e regras uma medida numérica que represente, de alguma forma, a confiança do especialista nos mesmos. Muitos Sistemas de Conhecimento dispõem de mais de um método, permitindo ao usuário escolher o que melhor atendê-lo.

Conforme já introduzido, o sistema aqui proposto utiliza o RDR como estrutura básica. Na seção 2.6, será visto que o RDR corresponde a um aprimoramento dos Sistemas de Produção visando otimizá-los no sentido de apresentarem melhor desempenho. Devido a isso, na próxima seção será dado um enfoque particular aos Sistemas de Produção.

2.5.4 Sistemas de Produção a Encadeamento Progressivo

A maioria das linguagens lógicas, como o Prolog, trabalham com o raciocínio regressivo. Dado uma consulta $Q_1(query)$, elas procuram por uma prova que estabeleça alguma substituição que satisfaça Q_1. Uma alternativa para esta abordagem é o raciocínio progressivo, onde não há consultas. Neste caso, as regras de inferência são aplicadas na base de conhecimento com o objetivo de produzirem novas inserções. Este processo é repetido infinitamente, ou até que algum critério de parada seja alcançado [41]. Um sistema de produção tem, basicamente, as seguintes características [41]:

- O sistema mantém uma base de fatos conhecida como memória de trabalho (*working memory*). Esta memória, contém um conjunto de literais positivos sem variáveis associadas;

- O sistema também mantém separadamente a base de regras (*base de conhecimento*). Esta base contém um conjunto de regras de inferências, onde cada regra é da forma $p_1 \wedge p_2... \Longrightarrow act_1 \wedge act_2...$, onde p_i são literais, e act_i são ações a serem executadas quando p_i são todos satisfeitos. As ações possíveis são aquelas que inserem ou removem elementos da memória de trabalho ou por exemplo, imprimem um valor;

- Em cada ciclo, o sistema computa o subconjunto de regras cujas condições foram satisfeitas pelo conteúdo da memória de trabalho. Esta fase é chamada de fase de *match*;

- O sistema então decide quais regras devem ser executadas. Isto é chamada de *fase de resolução de conflitos*. Esta fase refere-se ao momento em que o motor de inferência termina o processo de busca e dispõe de um conjunto de regras que satisfazem à situação atual do problema. Este conjunto é chamado de *conflito*. Se o conjunto for vazio, a execução é terminada, caso contrário, é necessário

escolher quais regras serão realmente executadas e em que ordem. Existem vários métodos que resolvem este problema;

- O passo final, em cada ciclo, é executar as ações presentes nas regras escolhidas. Isto é chamado de fase de execução (*act phase*).

A fase de *Match* serve para guiar a pesquisa na memória de trabalho e na base de regras. Em cada ciclo, o sistema computa o subconjunto de regras cujo premissas são satisfeitas pelo conteúdo atual da memória de trabalho. A forma mais simples de realizar unificação [41] é ineficiente. Visando a aprimorá-lo, Charles L. Forgy, na Carnegie-Mellon University, em 1979, em sua tese de PhD, desenvolveu o algoritmo conhecido como RETE. O algoritmo de RETE é um modelo que acelera o processo de inferências do sistema otimizando a maneira com que os sucessivos *matchs* são efetuados. Ao invés de buscar as semelhanças dos fatos em todas as regras, em todos os ciclos de reconhecimento das ações, o algoritmo RETE somente procura por trocas semelhantes em todos os ciclos. Isto aumenta em muito a velocidade da busca das semelhanças entre os fatos e os antecedentes das regras, desde que os dados estáticos, que não mudam de ciclo para ciclo, possam ser ignorados. O RETE elimina redundâncias de avaliações das pré-condições das regras. Por exemplo, se uma regra pode ser disparada com um conjunto de objetos, e uma segunda regra é disparada e não altera nenhum dos objetos do primeiro conjunto, então não há necessidade de se testar novamente as pré-condições desta regra, pois ela ainda está ativa com o mesmo conjunto de instâncias. A idéia por trás do algoritmo é que apenas os objetos modificados no disparo da última regra tenham que ser testados novamente para verificar se tornam alguma regra ativa (ou inativa). O algoritmo Rete é usado na maioria dos sistemas de produção existentes. O RETE apresenta vantagens, como, por exemplo, elimina duplicações entre regras, elimina duplicações ao longo do tempo e minimiza o número de testes requeridos durante a fase de casamento (*match*). A figura 2.11 mostra a idéia geral do algoritmo RETE. Observe que se as três regras fossem executadas isoladamente, as condições A e B, por exemplo, seriam, ambas,

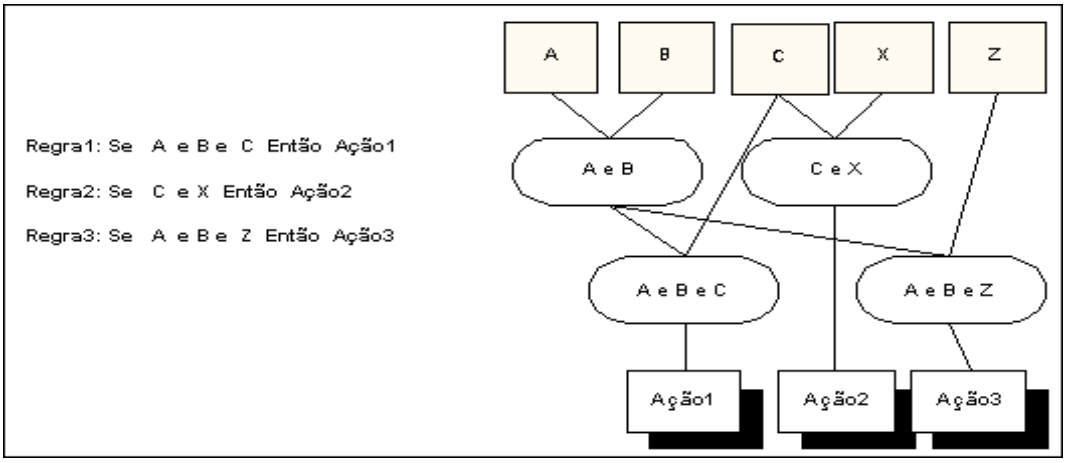

Figura 2.11: Idéia geral do algoritmo RETE

testadas duas vezes. No entanto, após as regras serem reorganizadas (compiladas) segundo o algoritmo RETE, durante a execução das mesmas, as condições A e B serão testadas uma única vez.

2.6 Ripple Down Rule

Ripple Down Rule (RDR) é uma metodologia utilizada para aquisição incremental de conhecimento [40, 46] e baseia-se em contexto e regras. O RDR pode ser categorizado como *Incremental Case-Based*. O RDR é baseado no princípio que "o conhecimento do especialista é essencialmente uma justificativa do porquê ele está correto, não no raciocínio que ele teve para alcançar a conclusão correta"[46]. O conhecimento deve ser adquirido dos especialistas no contexto de casos individuais vistos por eles. Esta metodologia foi proposta em 1989 por Compton, Horn, Quinlan e Lazarus como resultado de estudos realizados em um dos primeiros sistemas especialistas na área médica, o GARVAN-ES1 [40]. O GARVAN foi remodelado utilizando o RDR [13]. Com isso, a aquisição de conhecimento ficou aproximadamente 40 vezes mais rápida que na versão anterior[6]. Nos sistemas tradicionais baseados em regras, o

[6]O Sistema Garvan-ES1 iniciou com 200 regras. Após 11 semanas a base tinha 600 regras[Compton etal., 1991]. Após um ano, 1200 regras[Gaines and Compton, 1992].

engenheiro de conhecimento tem que garantir que novos conhecimentos adicionados à base de conhecimento não gerem conflitos com conhecimentos já existentes; o novo conhecimento deve aumentar a capacidade do sistema, não degradá-lo. Para alcançar este objetivo, o engenheiro de conhecimento deve entender como as regras interagem, saber onde inseri-las etc [38]. Em contraste com o RDR, o sistema tem a decisão de onde as regras serão colocadas [38]. No RDR, uma regra é adicionada ao sistema apenas quando um "caso"é dado como conclusão incorreta. A análise considerando "casos"assemelha-se ao *Case Based Reasoning* (CBR), mas, ao contrário do CBR, o RDR depende diretamente da aquisição do conhecimento do especialista [14].

Nos sistemas tradicionais baseados em regras de produção, os conhecimentos são representados por uma longa lista de regras individuais. No RDR, o conhecimento é representado por árvores, onde cada nó é uma regra. Cada nó pode ter apenas dois nós filhos, o "if-true", caso a condição seja satisfeita, e o "if-false", caso contrário. Além disso, o nó também tem um *cornestone case* associado, isto é, uma justificativa da necessidade da regra ter sido inserida. A figura 2.12 mostra a estrutura de funcionamento de um RDR. Nesta figura, o módulo de casos informa o caso a ser analisado para o módulo de inferência. Este por sua vez, analisa as regras da base de conhecimento e fornece uma resposta. Se a resposta estiver correta o processo é finalizado. Caso contrário, esta informação poderá ser adicionada na base de conhecimento.

O processo de aquisição de conhecimento em um Sistema Baseado em Conhecimento é um processo contínuo, ou seja, novos conhecimentos podem ser adquiridos ou modificados. Uma simples atualização de conhecimentos pode provocar inconsistências nos conhecimentos já adquiridos. O RDR procura facilitar o processo de manutenção da base de conhecimentos, não permitindo a exclusão de regras. Toda manutenção é feita mediante a inclusão de novas regras no final da árvore. Este processo ganha em desempenho, uma vez que não há necessidade de analisar todas as regras da base de conhecimento, já que apenas o ramo da árvore ao qual pertencerá a regra a ser inserida será analisado.

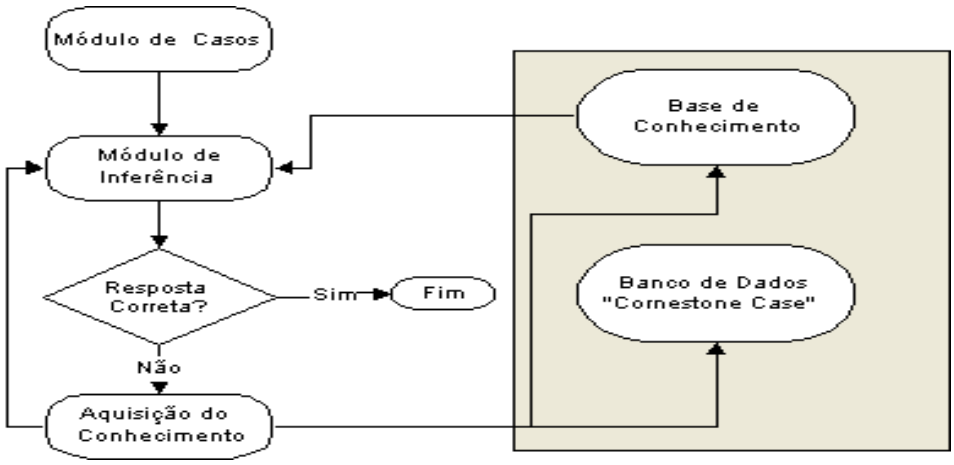

Figura 2.12: Estrutura de um Ripple Down Rule

Uma regra de um RDR pode ser assim definida:

```
if  A and B then C except
      if D then E
else  if F and G then H
```

É interpretado como : Se A e B são verdadeiros, então, conclui-se C, exceto se D for verdadeiro. Neste último caso, conclui-se E. Se A ou B forem falsos, então, se F e G forem verdadeiros, conclui-se H. A figura [2.13] mostra seis regras extraídas do GARVAN-ES1 [18]. Nela, existem 61 combinações de diagnósticos possíveis. Estes diagnósticos são rotulados de 00 a 60. A regra 0.00 é uma regra *default* a qual não possui condições, ou seja, se nenhuma outra regra for disparada, ela será o diagnóstico. Esta regra é sempre verdadeira, então ela tem apenas uma ligação "if-true"para a regra 1.46. Esta regra, por sua vez, tem 4 condições para dar o diagnóstico 46. Se estas condições não forem satisfeitas, o "if-false"para a regra 2.32 é seguido. Se as condições da regra 1.46 forem satisfeitas, o diagnóstico não é indicado imediatamente. O "if-true"para a regra 4.48 é testado, se suas condições forem satisfeitas, o diagnóstico 48 é indicado sobrepondo o diagnóstico 46. As características principais do RDR são [21]:

- O Especialista monitora o sistema durante sua execução. Sempre que o sistema

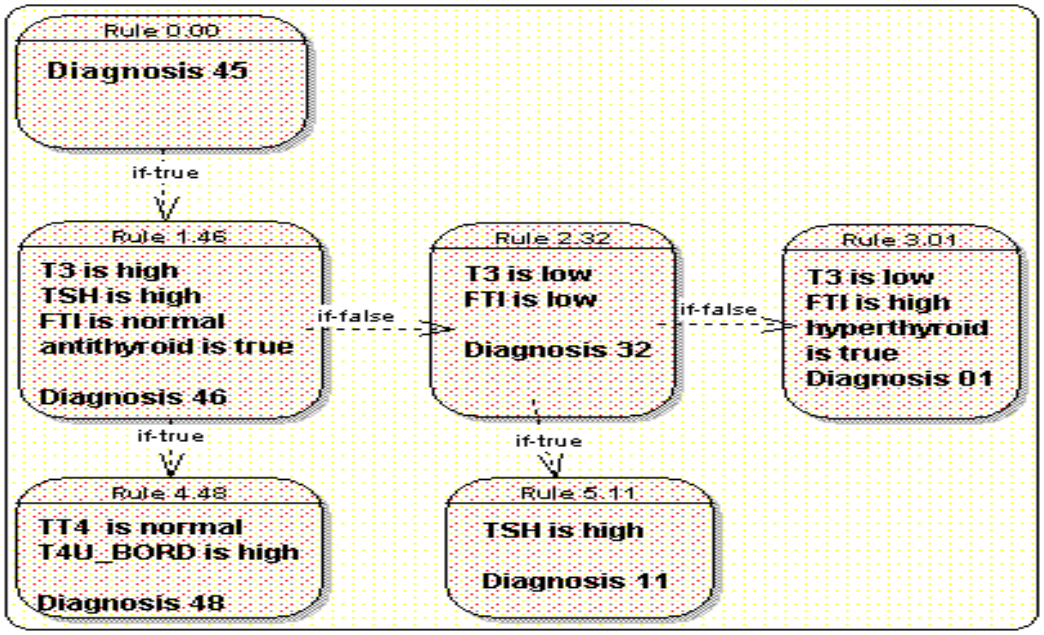

Figura 2.13: Exemplos de Ripple Down Rules

deduzir algum caso que não esteja mais em conformidade com o desejado, o Especialista inclui uma nova regra no RDR para corrigir a análise errada.

- A utilização da estrutura de exceções é usada para corrigir regras erradas. Regras não são editadas. Toda correção no sistema é feito via inserção de novas regras.

- O RDR pode ser construído off-line, mas a técnica é direcionada para correção de erros durante sua execução.

Pode-se notar que, se houver uma necessidade freqüente de adição de novas regras, a árvore ficará muito grande e o desempenho será afetado. Dependendo da aplicação, o RDR pode gerar repetições de conhecimentos, isto é, o mesmo conhecimento pode ser usado em diferentes contextos [15]. Isto pode causar um grande aumento da tarefa de aquisição do conhecimento. Embora seja um problema real, ele não prejudica o método. Métodos tradicionais de indução produzem tamanhos similares da Base de Conhecimento se comparados ao RDR. Este problema pode ser contornado

reorganizando-se o RDR. Desta forma, a redundância da base é eliminada. Em [46] é apresentado um algoritmo que faz esta reorganização da base de conhecimento.

Em [43] é apresentado uma fundamentação algébrica do RDR.

2.6.1 Funcionamento do RDR

O objetivo desta seção é apresentar o funcionamento do RDR [11]. Para isso, será utilizado o exemplo ilustrado na figura 2.14. Inicialmente, deve-se lembrar que o nó raíz (*default)* é sempre o primeiro nó da árvore. Este nó serve para identificar a árvore e informar o ponto de partida. Observa-se, também, que os nós da direita referem-se à nós cuja condições foram satisfeitas, e os nós no sentido vertical, são aqueles nós, cuja condições não foram satisfeitas. Considere que o nó em destaque (regra 4.01, com seu respectivo *cornestone*), inicialmente não pertença à estrutura (conforme será visto no final desta seção, ele deverá corresponder a uma nova regra a ser inserida por decisão do especialista). Por simplicidade, foram omitidos os *cornerstone* das regras, com exceção o da regra 4.01. O RDR ilustrado na figura 2.14 descreve, resumidamente,

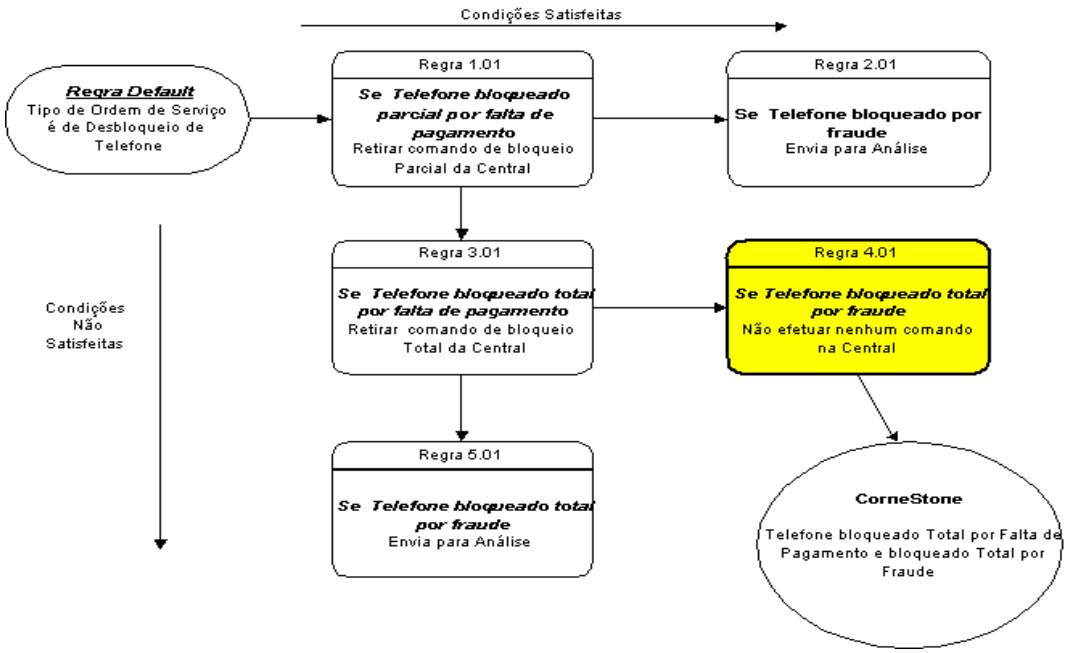

Figura 2.14: Funcionamento do RDR

uma situação típica de empresas de telecomunicações. Mediante a falta de pagamento ou detecção de fraudfes, os telefones são bloqueados, isto é, são impedidos de originar e receber chamadas (bloqueio total por falta de pagamento ou bloqueio total por fraude detectada), ou apenas impedidos de originar chamadas (bloqueio parcial por falta de pagamento). Estes mesmo telefones apenas são desbloqueados através do pagamento da fatura, ou, se for o caso de fraude, através da resolução do problema.

No exemplo, o RDR trabalha apenas com Ordens de Serviços (OS) de desbloqueios, ou seja, mediante o pagamento de uma fatura, esta informação é enviada ao RDR que fará a análise. Os resultados destas análises são aqueles descritos nos nós do RDR. A consulta a ser analisada pelo RDR é: "Qual a ação que deve ser tomada mediante um desbloqueio de telefone?". Neste exemplo, suponha que o telefone A esteja com bloqueio total, tanto por falta de pagamento, quanto por fraude.

Quando a OS de desbloqueio por pagamento efetuado é enviada ao RDR, é feita uma verificação no nó raíz, referente ao tipo de OS. Caso seja de desbloqueio, inicia-se o processo do RDR. Caso contrário, nenhuma conclusão é tirada. Após a regra *default* ter sido analisada como verdade, a regra 1.01 é avaliada. Como o telefone A está com bloqueio total, a regra 1.01 ("*Se telefone bloqueado parcial*") falha. Logo a seguir, a condição da regra 3.01 ("*Se Telefone bloqueado total por falta de pagamento*") é avaliada como sendo verdade. Neste ponto, o sistema pergunta ao usuário se a conclusão *"Retirar comando de bloqueio total da Central"* está correta (note-se que, neste ponto, o nó 4.01 ainda não consta do RDR, conforme explicado anteriormente). Caso a resposta for "sim", o RDR finaliza e o *cornestone* é apresentado como resposta. A vantagem da utilização do *cornestone* é que toda regra inserida deve satisfazer a um caso específico. Caso contrário, o usuário deve inserir uma nova regra para corrigir a conclusão errada e informar o *cornerstone*. Na figura 2.14, o nó 4.01 representa a nova regra adicionada. Esta figura significa que o telefone somente será desbloqueado se ele não estiver com bloqueio total por fraude.

Finalmente, observe que o exemplo da figura 2.14 pode ser enquadrado na estrutura geral do RDR apresentado na figura 2.12, da seguinte maneira: a OS de desbloqueio corresponde ao caso enviado pelo módulo de casos. As regras 1.01, 2.01, 3.01 e 5.01 compõem o módulo de inferência original. A regra 4.01 representa o conhecimento adquirido como conseqüência do fato de o módulo de inferência original não ter produzido uma resposta satisfatória.

Capítulo 3

Estado da Arte

3.1 *Introdução*

Conforme já apresentado, o presente trabalho descreve uma proposta aplicada à área de Telecomunicações que utiliza *Wavelets*, Sistema Baseado em Conhecimento e *Ripple Down Rules* como ferramentas de suporte. Neste capítulo, resume-se o estado da arte ligado à aplicação dessas ferramentas.

3.2 *Wavelets*

A teoria sobre *wavelets* está calcada em conhecimentos matemáticos. Uma boa maneira de entendermos como as *wavelets* trabalham foi descrita em [34], onde as *wavelets* são comparadas com a forma que nossos olhos vêem o mundo. No mundo real, quando olhamos para uma floresta é como se estivéssemos vendo uma fotografia com uma resolução melhor, e mesmo assim não conseguimos ver detalhes, ou seja, não conseguimos diferenciar uma árvore da outra, não conseguimos ver os galhos, folhas etc. Mas, à medida que nos aproximamos, estes detalhes começam a ser cada vez mais identificáveis, isto é, começamos a visualizar nitidamente as árvores, galhos e as folhas. Detalhes que até então não tinham sido observados são encontrados. Se ampliarmos uma fotografia de uma floresta, não veremos as árvores nitidamente, não veremos galhos nem folhas. Com as *wavelets* será possível ver a imagem de uma

floresta interativamente, ou seja, poderemos aumentar a resolução da imagem (zoom) e pegar os detalhes escondidos. Isso será possível porque as *wavelets* são capazes de compactar os dados eficientemente tornando possível armazená-los em um menor espaço físico.

A partir da década de 80, os estudos sobre *wavelets* tiveram seu desenvolvimento acelerado, uma vez que neste período as teorias matemáticas se tornaram mais coerentes.

As *wavelets* começaram, então, a ser utilizadas como técnica de compactação de dados em diversas áreas, tais como, compactação de dados [34][1], na área cinematográfica [34][2], mineração de dados [26], na compactação de imagens [34][3], sendo que, nesta última, permite que as imagens sejam facilmente compactadas, transmitidas via internet e, uma vez recebidas no destino, rapidamente descompactadas. A seguir, apresentam-se algumas aplicações relevantes das *wavelets*.

3.2.1 *Wavelets* em Mineração de Dados (*Datamining*)

A utilização de métodos que utilizam as *Wavelets* em vários processos de *data mining* tem aumentado significativamente. Em [26] é feito um levantamento onde são apontadas algumas áreas de pesquisas que utilizam técnicas que fazem uso das *Wavelets*. O objetivo proposto pelo autor do artigo citado anteriormente foi apresentar uma estrutura de trabalho que modulariza o processo de *data mining* em pequenos componentes,e, então mostra as aplicações de *wavelets* em cada componente. O processo de *data mining* ou mineração de dados é uma tarefa muito trabalhosa, principalmente quando realizada em grandes volumes de dados. Este processo envolve vários campos de pesquisas em Inteligência Artificial, tais como, por exemplo, aprendizado de máquina, reconhecimento de padrões, sistemas especialistas, aquisição de conhecimento e outros. Visando otimizar este processo, a utilização de um sistema modular para

[1]Usado pelo FBI para compactar os enormes banco de dados contendo impressões digitais (1992).
[2]Usada em 1995 no filme Toy Story.
[3]Em 1999 A International Standards Organization aprovou um novo padrão compressão de imagens digitais chamado JPEG-2000.

telecomunicações (SMT) foi proposto [26] em quatro passos:

1. **Gerenciamento de Dados** que consiste em todo o mecanismo de acesso ao dado, da estrutura de acesso ao dado e da forma de armazenamento. Neste passo, é importante usar técnicas de acessar o dado rápida e eficientemente. As transformadas *Wavelets* estão sendo consideradas uma boa técnica, pois podem ser utilizadas para compactação dos dados, podem ser usadas para extrair as tendências e sazonalidade de séries temporais.

2. **Pré-Processamento do dado** é o passo onde é feita a "seleção"do dado, ou seja, é aqui, que é retirada toda a informação desnecessária. Neste passo, também se fazem as integrações dos dados, contemplando, se necessário, múltiplas fontes de informações, redução da dimensão do dado, transformações dos dados etc. Nesta etapa, dentre outras atribuições, as *Wavelets* podem ser usadas para reduzir a dimensão do dado,isto é, representar o conjunto de dados original através de um conjunto menor de dados sem perder informações relevantes.

3. **Tarefas e algoritmos** é o passo onde são aplicados os algoritmos de *data mining* e onde são realizadas tarefas, como por exemplo, visualização, classificação, clusterização e pesquisas por similaridade. Como exemplo, pode-se citar a utilização das *Wavelets* nas pesquisas por similaridade, onde os dados são transformados para um domínio *Wavelet*[4] e, a partir daí, aplicam-se as técnicas de consultas sobre a base reduzida.

4. **Pós-Processamento** é o passo onde ocorre o refinamento e avaliação do conhecimento dos passos anteriores. Neste passo, pode-se por exemplo, optar por interpretar o conhecimento e incorporá-lo em um sistema existente.

[4]Redução da dimensão do dado.

3.2.2 Wavelets como Ferramenta de Compactação de Dados em Séries Temporais

Uma série temporal é um conjunto de observações (dados) ordenadas no tempo [31]. Podemos citar como exemplo de séries temporais: índices de bolsa de valores, valores diários de temperaturas, valores mensais de vendas de automóveis, análise de intervenção em séries temporais aplicadas a transporte urbano [7] etc. Extrair conhecimento de séries temporais, apesar de não ser uma tarefa fácil, é imprescendível em áreas financeiras, científicas etc.

As séries temporais podem ter diferentes tamanhos, diferentes posições verticais, diferentes tendências. Todos estes fatores fazem com que a análise se torne complexa. Um problema é: como comparar uma série temporal com outra, eficientemente? Para auxiliar nesta questão, as Consultas por Similaridade estão sendo muito utilizadas. Freqüentemente, o volume de dados envolvendo tais análises é consideravelmente grande. Devido a estes grandes volumes, indexar diretamente a série temporal se tornaria proibitivo, ou seja, a velocidade de acesso ao dado não seria eficiente [36]. Com isso, a redução da dimensão do dado aparece como uma alternativa muito interessante para tentar minimizar este problema, pois, se o tamanho da série for diminuída, menor será o índice e conseqüentemente mais rápido será o acesso à informação. Porém, como podemos diminuir, ou seja, reduzir a série sem perder informações relevantes? Várias técnicas envolvendo a utilização das *Wavelets* foram propostas para auxiliar neste trabalho.

Em [9] é proposto um método usando *Haar Wavelets Transformation* para indexação e recuperação de dados em séries temporais. Neste artigo é demonstrado que: (1) A distância Euclidiana é preservada no domínio *Haar transformed* e previne a ocorrência de *Falso-Negativo* (perdas). (2) Mostra que este método pode superar a *Discrete Fourier Transform* (DFT) através de experimentos, e (3) um novo modelo de similaridade é sugerido para contemplar deslocamentos verticais de séries

temporais. A escolha das *Haar Wavelets* foi baseada nos seguintes fatos: (1) permite uma boa aproximação com apenas um subconjunto de coeficientes,(2) pode ser computado fácil e rapidamente,(3) preserva a distância Euclidiana. A idéia geral do método é a seguinte: Antes da consulta ser realizada na série temporal, é feito um pré-processamento para extrair o vetor de característica com a redução da dimensão do dado e então um índice é construído. Após esta etapa, as consultas por podem ser realizadas de duas diferentes maneiras: *consulta por abrangência* ou *consulta dos k-vizinhos mais próximos*. Outros tipos de *Wavelets* foram testados, mas as *Haar Wavelets* apresentaram um melhor desempenho, quando comparadas às *Daubechies e Coiflet Wavelets*, em relação à precisão. Uma outra importante observação citada é que nem todos os tipos de *Wavelets* são capazes de concentrar energia nos primeiros coeficientes. *Haar, Daub4 e Coif6 Wavelets* foram as melhores famílias de *wavelets*.

Um outro método relacionado à pesquisa por similaridade em séries temporais foi proposto em [36], onde é proposta a utilização de *Wavelets* bi-ortonormais para dar suporte a consultas por similaridade, ou seja, expande a utilização das *Wavelets* nesta área. Uma outra contribuição é o detalhamento do desempenho de várias *Wavelets* em pesquisas por similaridade (mostra que algumas destas *Wavelets* podem superar as *Haar Wavelets* citadas anteriormente).

Em [24] é proposto outra técnica para redução da dimensão do dado para melhorar o desempenho de pesquisas por similaridade em séries temporais em grandes banco de dados. Esta técnica foi chamada de PAA (*Piecewise Aggregate Approximation*. Nos experimentos realizados foi observado algumas vantagens desta técnica em relação a outros citados anteriormente, tais como:

1. Muito rápido para ser computado;

2. Definido para trabalhar com consultas de tamanhos arbitrários, ao contrário da DWT;

3. Permite inserções e remoções de constantes de tempo, ao contrário da de outras

técnicas, como a *Singular Value Decomposition* (SVD).

As *Wavelets* são usadas, também, em pesquisas que envolvem o trabalho de realizar previsões em séries temporais. No modelo clássico, as séries temporais são divididas em quatro padrões:

1. Tendência, que descreve um movimento suave, a longo prazo, dos dados, para cima ou para baixo;

2. Variações sazonais, são variações cíclicas envolvendo prazos relativamente curtos;

3. Variações Cíclicas,

4. Variações irregulares.

Em [4] por exemplo, as *Wavelets* não são utilizadas como ferramenta de compactação de dados, mas, sim, como método para realizar previsões econômicas em séries temporais, considerando a venda de carros no mercado espanhol. Na referência em foco, comparam-se os resultados de tal abordagem com um modelo tradicional, o *Box-Jenkins*. O método de [4] consiste em aplicar a transformada *wavelet*[5] em um conjunto específico de dados; divide os coeficientes obtidos em 2 partes. Cada parte, após aplicada a transformada *wavelet* inversa, captura os movimentos sazonais e a tendência da série. Desta forma, a série temporal é decomposta em 2 componentes. O primeiro componente representa a tendência da série, enquanto que o segundo representa a sazonalidade. Aplicando-se métodos já conhecidos em cada componente, obtém-se a previsão da série temporal.

A análise das séries temporais, conforme dito anteriormente, é complexa. Embora haja uma ampla gama de pacotes estatísticos para uso em computadores que funcionam como ferramentas de apoio ao estudo das séries temporais, a interpretação

[5]É utilizada a *Daubechies Wavelets*.

geralmente cabe ao usuário. Além disso, muitos destes pacotes exigem considerável experiência em estatística [45].

3.3 Sistemas de Conhecimento

Uma das tarefas mais complexas em Ciência da Computação é construir máquinas capazes de aprender, ou seja, máquinas que aprendam automaticamente com experiências acumuladas durante o período de sua utilização. Atualmente, é cada vez mais freqüente o uso de sistemas de conhecimento que auxiliam nas tomadas de decisões, na detecção de intrusões, na elaboração de diagnósticos, em análise, em controles automáticos etc.

Em [5] é apresentado uma estrutura de apoio para sistemas de conhecimento que utiliza uma técnica conhecida como *Case Base Reasoning*, ou *Raciocínio baseados em casos* (CBR). O CBR é um método que visa aproveitar as experiências passadas para tomar futuras ações. A idéia é que ele funcione como os seres humanos, ou seja, normalmente utilizamos de experiências do passado para resolvermos problemas de nosso cotidiano. Cada experiência aprendida é chamada de Caso. Ainda em [5], é sugerida a utilização do CBR em domínios específicos, em que haja apenas casos (experiência aprendida) disponíveis. Também é necessária, para a interpretação dos casos, a noção de similaridade, pois eles serão comparados de modo a apresentar sucessivas evoluções. A análise da base de casos pode ser feita de diferentes maneiras, sendo que a mais simples é a busca seqüencial. Enquanto que em máquinas de aprendizado baseado em indução ocorre a generalização do conhecimento, no CBR é feito o armazenamento de toda a base de casos. Na maioria dos Sistemas de Conhecimento tradicionais, novos conhecimentos são obtidos a partir de inferências efetuadas através de regras de produção, enquanto que nos sistemas que utilizam o CBR, novos problemas são comparados com a biblioteca de casos, e cada caso tem informações específicas de problemas com suas devidas soluções [22].

O CBR pode ser usado em aplicações práticas como sistemas de Help-Desk

e diagnóstico técnico de sistemas [6]. Na área de diagnóstico técnico, o CBR está ganhando espaço rapidamente, pois permite um desenvolvimento e uma manutenção da base de conhecimento mais rápidos e baratos do que em Sistemas de Conhecimentos tradicionais [6]. As *Wavelets* começaram a ser aplicadas, também, nos Sistemas de Conhecimento. Por exemplo, o CBR, combinado com *Wavelets*, pode auxiliar muito bem na análise de imagens, conforme mostrado em [12], onde tal combinação é aplicada na análise previsional. Áreas como turismo e agricultura necessitam de estimativas prévias de precipitações. Atualmente , tais análises são feitas através de fotografias fornecidas por satélites ou radares. Como o volume de dados é grande, torna-se difícil o trabalho para realizar uma análise apurada destas imagens. Através do CBR e usando *Wavelets* para um pré-processamento das imagens, consegue-se melhorar a análise.

Outros tipos de Sistemas de Conhecimento que não utilizam base de casos, mas sim , grandes bases de conhecimentos, são os Sistemas de Conhecimento tradicionais, que geralmente são penalizados por terem um baixo desempenho, por utilizar-se de linguagens complexas e por serem de difícil integração com outros sistemas. Porém, em [27] é apresentado um Sistema Baseado em Conhecimento que tem sido utilizado na análise e detecção de intrusos em redes de computadores. Este sistema apresenta um ótimo desempenho em detecção em tempo real, provê excepcionais formas de integração com bibliotecas nativas de sistemas operacionais e também com outros softwares. O *Production-Based Expert System Toolset (P-BEST)*, como é chamado este sistema, utiliza as técnicas tradicionais de inferências baseadas em *modus ponens*, estratégia de raciocínio *forward-chaining* e regras de produção.

Vários estudos estão sendo realizados para que um dos principais problemas encontrados nos Sistemas de Conhecimento seja amenizado, a geração da base de conhecimentos. Técnicas como Algoritmos Genéticos e Redes Neurais são cada dia mais comuns. Para ilustrar alguns destes estudos, em [28] descrevem-se idéias usadas no desenvolvimento de um sistema de aprendizado de máquina para domínios *fuzzy*,

o qual induz hipóteses representadas por valores *fuzzy*, a partir de um conjunto de exemplos de treinamentos. Já em [25] são utilizados Rede Neurais Artificiais com uma arquitetura complexa e um grande número de entradas de dados conseguindo resultados próximos aos de um analista de mercado[6]. Um outro ponto interessante para um Sistema Especialista é dizer como ele alcançou a resposta do problema proposto,ou seja, não basta resolver o problema, tem que explicar como isso foi feito. Para resolver este problema, um Sistema híbrido foi proposto em [8]. O referido trabalho propõe um sistema que combina dois algoritmos: o primeiro é usado para treinar a rede neural nebulosa na obtenção do conhecimento, e o segundo, para obter a explicação de como a rede neural nebulosa chegou a uma dada conclusão.

Outros Sistemas de Conhecimento usam o RDR como ferramenta de apoio, como por exemplo, o RDR foi usado na reestruturação do sistema GARVAN ES-1 [18]. Este sistema estava em uso desde 1984 e produzia em torno de 6.000 interpretações por ano, possuía 650 regras e fazia 60 diagnósticos. Com o passar dos anos, a manutenção no GARVAN ES-1 tornou-se muito difícil. Quando o sistema foi reestruturado usando o RDR, as regras foram reduzidas para 500, tornaram-se mais simples e o processo de inclusão de novas regras tornou-se muito mais rápido. O RDR tem exatamente a mesma capacidade de dedução e representação que os sistemas conhecidos como sistemas de produção. O RDR difere em relação ao contexto, isto é, os "if-true"e "if-false"garantem que a adição de uma nova regra não afetará nenhuma outra regra [18].

O RDR vem sendo utilizado e apresentado em vários trabalhos nos últimos anos. O sistema PEIRS (*Pathology Expert Interpretative Reporting System*) do *Dept. of Chemical Pathology, St. Vicentt's Hospital Sydney* utiliza o RDR. Este sistema tem mais de 2000 regras [21, 37], não precisa de um engenheiro de conhecimento e nem suporte de programação, ou seja, o próprio e especialista cria as regras. Um outro trabalho que utilizou o RDR foi apresentado em [40], onde o objetivo é simplificar

[6]Neste trabalho é apresentada uma nova proposta de uma rede neural com aprendizado por reforço. Foram realizados testes na Bolsa de Valores de São Paulo.

a aquisição do conhecimento por meio da extração de alguns tipos de conhecimento diretamente de linguagem natural. Em [39] o RDR é utilizado para auxiliar na re-utilização do conhecimento, onde o RDR é utilizado como ferramenta de aquisição de conhecimento para facilitar a re-utilização da informação.

As primeiras versões do RDR foram focadas em tarefas de classificação (*classification tasks*). Primeiramente foi introduzido o *Single Classification Ripple Down Rules* (SCRDR) e posteriormente o *Multiple Classification Ripple Down Rules* (MCRDR). A aplicação do RDR tem sido estendida nos últimos anos, como por exemplo: tarefas de configuração, alocação de recursos, processamento de linguagem natural e processamento de imagens [46]. Um exemplo do uso comercial do RDR é a *Pacific Knowledge Systems (PKS)* [7], o qual é utilizado com o objetivo de produzir comentários clínicos para relatórios patológicos.

[7] http://www.pks.com.au

Capítulo 4

Sistema Modular para Telecomunicações-SMT

4.1 Introdução

Nesta dissertação, propõe-se um sistema modular (SMT) que efetua consultas por similaridade em séries temporais para auxiliar na detecção de anomalias associadas ao perfil de uso de telefone, ou seja, ao tráfego de ligações telefônicas efetuadas por clientes de empresas de telecomunicações. Para tanto, o SMT representa os dados referentes às chamadas telefônicas através de séries temporais que são criadas por um módulo Gerador de Séries.

A análise visando a detecção de anomalias é efetuada através de pesquisa de similaridade entre as séries temporais que representam a utilização real das linhas telefônicas e as séries que representam, ou o padrão de uso ideal estabelecido pelos proprietários da linha, ou um perfil de fraude.

Um outro módulo do SMT é o módulo Gerador de Consultas. Tal módulo tem como função executar as consultas propostas pelos usuários do SMT indagando sobre a normalidade ou não da utilização das linhas telefônicas.

Uma vez terminado o processo de consultas, o SMT aciona um último módulo: o módulo de Conhecimento, responsável por indicar as ações que deverão ser desencadeadas em função dos resultados das análises de similaridade. Tal módulo

corresponde a uma combinação de um sistema de produção e de um sistema baseado em casos (*Case Based Reasoning*), sendo construído nos moldes da estrutura RDR (*Ripple Down Rules*).

O módulo do Conhecimento é composto por regras relacionadas ao universo da telefonia. Incluem-se aí, por exemplo, regras que detectam ocorrência de fraude e que prevêem ações a serem desencadeadas caso isto ocorra. Tais regras são construídas nos moldes das estruturas RDR. Para tanto, a base de conhecimentos contém regras do tipo: *Se o perfil de ligações de um determinado telefone estiver SEMELHANTE a um padrão de fraudes conhecido, então este telefone pode ser enviado a uma área responsável por fraudes para ser analisado.*

Para efetuar pesquisas de similaridade desta natureza, a utilização de métodos tradicionais não apresenta desempenho satisfatório. Um problema enfrentado na tentativa de realizarem tais análises com eficiência, é o grande volume de dados a ser analisado. Diariamente são realizadas milhões de ligações telefônicas, sendo que cada ligação efetuada é registrada pelas centrais telefônicas e armazenada em arquivos, normalmente conhecidos como *Call Detail Record* (CDR), os quais contêm informações detalhadas relacionadas às ligações telefônicas realizadas pelos clientes. No SMT aqui proposto, tais informações são recuperadas a partir do CDR e são convertidas em séries temporais armazenadas em seu banco de dados. Para a realização de consultas em base de dados, normalmente, utilizam-se índices para aumentar o desempenho do processo de consulta. Entretanto, em séries temporais, que são a ferramenta de suporte de representação de dados aqui adotada, os tipos de índices tradicionais *(B-Tree)* são poucos eficientes [48]. Devido a isso, propõe-se a utilização dos índices espaciais, os quais apresentam um melhor resultado. Particularmente, será utilizado o *R-Tree*[1].

Como dito no Capítulo 2, uma série temporal pode ser considerada um ponto de dimensão n no espaço. Logo, ela pode ser indexada utilizando-se índices espaciais.

[1]Podem-se utilizar os índices espaciais de várias formas, como por exemplo: considerando pontos no espaço, formas geométricas etc.

Entretanto, estes tipos de índices têm seu ponto ótimo em relação ao desempenho quando trabalham com um número de pontos entre 7 e 12 [24], o que não representa a realidade das séries temporais tratadas neste sistema, as quais têm um número de pontos muito superior a estes. Para tentar contornar o problema, o SMT utiliza técnica de redução da dimensão dos dados, particularmente, as *Haar Wavelets*. Um outro problema a ser tratado, desta vez incidindo no módulo do sistema de conhecimento que indica as ações que deverão ser desencadeadas em função do resultado das análises de similaridade, consiste no seguinte: as informações (conhecimento) são muito dinâmicas, uma vez que as regras de negócio mudam a toda momento, e isto se agrava quando se considera que o quadro de funcionários é, normalmente, reduzido. Para lidar com tal comportamento, o SMT utiliza o RDR como ferramenta de construção incremental da base de conhecimento.

Para que as consultas por similaridade sejam eficientes, várias atividades precisam ser executadas ou avaliadas, tais como: criação das séries temporais, redução da dimensão dos dados, criação de índices e definição da medida de similaridade.

Para direcionar o entendimento do tema a ser abordado, serão consideradas, apenas, as seguintes anomalias: a anomalia relacionada à detecção de fraude e anomalia relacionada ao perfil de uso dos clientes. Ao longo deste capítulo, alguns termos como "Perfil Real"(PR), "Perfil Desejado"(PD) e "Perfil de Fraude (PF)", serão utilizados, e referem-se ao tráfego de ligações telefônicas representadas por séries temporais. O termo PD corresponde a séries que representam um padrão de uso desejado estabelecido pelo cliente. O termo PF, corresponde a séries que representam um determinado padrão de fraude obtido por meios estatísticos, experimentais etc. O termo PR refere-se ao perfil real de uso das linhas telefônicas, isto é, às séries que representam as ligações efetuadas pelos clientes. As consultas de similaridade poderão ser feitas, por exemplo, entre PR e PD ou PR e PF.

Para ilustrar a atuação do sistema proposto em relação a consultas por similaridade, considere o seguinte caso: suponha que um cliente A tenha um PD de ligações e

um PR de ligações conforme mostrado na figura 4.1. Neste exemplo, o sistema deverá

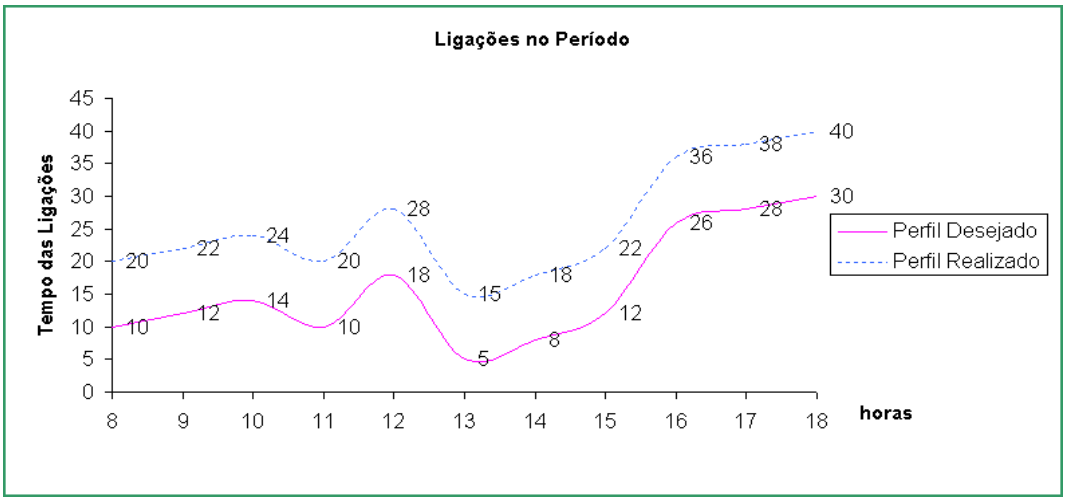

Figura 4.1: Perfil de uso de Ligações Telefônicas de um Telefone

ser capaz de identificar que as curvas presentes na figura 4.1 são similares (observe que a curva que representa o PR é igual à curva que representa o PD, apenas acrescida por um fator de 10).[2]. Outro tipo de questão a ser resolvida pelo sistema poderia ser: identificar que as séries temporais representadas na figura 4.2 são pouco similares. Para obter o grau de similaridade entre as séries temporais, será utilizada a distância euclidiana.

4.2 Detecção de Anomalias em Telecomunicações

Anomalia em Telecomunicações pode ter diferentes significados. Serão consideradas nesta dissertação, a título de exemplo, anomalias relacionadas a fraude e ao perfil de uso de clientes, conforme dito anteriormente. Nas próximas seções, serão apresentados exemplos de anomalias.

[2]Este tipo de análise depende do tipo de aplicação, ou seja, pode-se utilizar a transformação de *shifting* ou não.

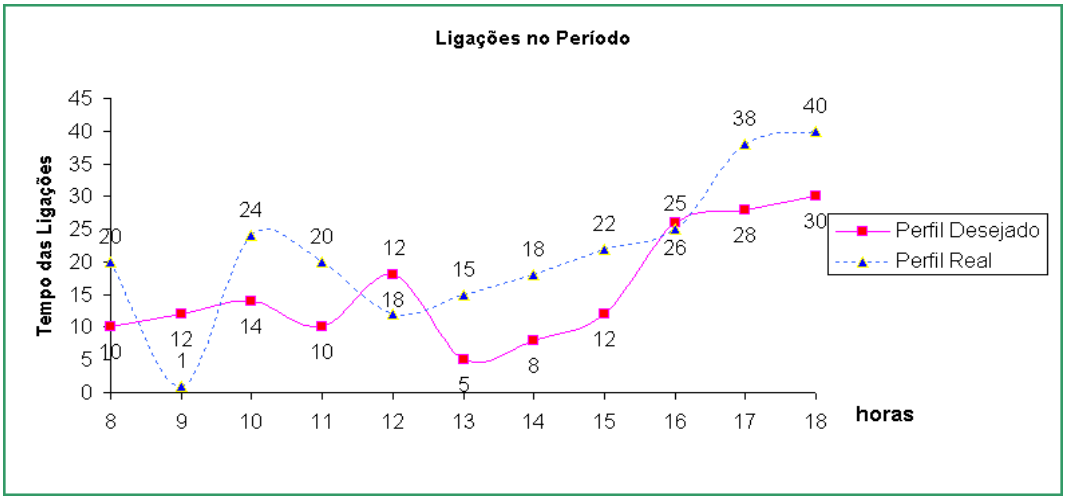

Figura 4.2: Perfil de uso de Ligações Telefônicas de um Telefone

4.2.1 Anomalias referentes à Fraude

Atualmente, existem vários sistemas comerciais para detecção de fraudes, os quais, geralmente, monitoram a utilização ("uso") da rede, para detectar padrões da fraude. A figura 4.3 ilustra a arquitetura genérica de um sistema anti-fraude em redes de telefonia celular [42]. Na figura 4.3, uma chamada é realizada e concluída com

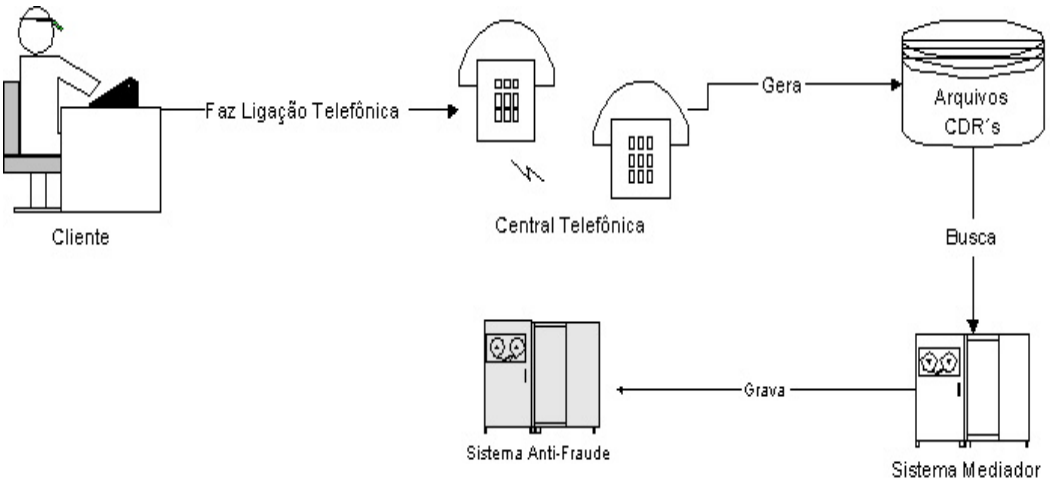

Figura 4.3: Arquitetura de um Sistema Anti-Fraude

sucesso. Quando o assinante desliga o telefone, a Central a qual o telefone estava se comunicando finaliza a conexão e grava o registro de sua chamada gerando um CDR

(*Call Detail Record*). O sistema de mediação, então, coleta este CDR e o entrega a vários sistemas, entre eles, o sistema anti-fraude. Este tipo de análise é chamado de Pós-evento, pois os CDRs são gravados somente depois que a chamada é completada ou terminada. Em caso de fraudes, quanto mais cedo os CDRs alcançarem o sistema anti-fraude, mais rapidamente elas poderão ser detectadas. Os sistemas Anti-Fraudes estão cada vez mais sofisticados e a utilização de técnicas como Rede Neurais já é uma realidade em alguns deles.

Tipicamente, toda atividade incomum ou suspeita resulta em um alarme. A análise destes alarmes, baseada em conhecimentos pré-configurados e adquiridos pelo sistema relativos ao padrão de comportamento do assinante, gera casos de fraude. Estes casos são, então, enviados aos analistas de fraude para análise e conclusão, podendo produzir resultados como, por exemplo, Fraude, Não Fraude e Desconhecido.

Um conceito muito importante é aquele relativo aos casos de Fraudes *Falso-positivas*, em que o sistema detecta um caso de fraude e o analista decide que não é fraude. Por mais que o sistema tenha informações sobre o padrão de comportamento do assinante e sua base de conhecimento esteja corretamente configurada, ainda assim podem existir situações que não se configuram como casos de fraude, após uma análise pormenorizada do analista de fraude.

Os sistemas anti-fraude tratam de volumes muito grandes de dados e são muito sensíveis ao desempenho. Cada vez mais estão recebendo dados de outras fontes de informação para não ficarem limitados a técnicas da detecção em função do "uso". Essas informações adicionais podem ser: Informação de tarifação (Billing), Informações do assinante e informações de associações de proteção ao crédito (Exemplo: Serasa). Podem-se utilizar técnicas estáticas ou dinâmicas na detecção de fraude [42]. Como exemplo de técnicas estáticas de detecção de fraude, têm-se as seguintes:

Detecção da colisão (sobreposição da chamada): determina se duas chamadas, supostamente do mesmo telefone ou serviço, ocorreram ao mesmo tempo;

Verificação (geográfica) da velocidade: determina se duas chamadas, supostamente do mesmo telefone ou serviço, aos serem feitas em duas posições geográficas suficientemente distantes e num período muito curto de tempo, são passíveis de acontecerem;

Desvio de Perfil de Uso: detecta quando um perfil individual recentemente calculado para um dado assinante difere de seu perfil precedente por mais do que uma quantidade especificada pela operadora;

Lista Negra: para cada chamada que chega ao sistema, são verificadas as listas de aparelhos roubados, MIN/ESN, e de IMSI que podem ser usados para a clonagem. Quando existe coincidência de informações, um caso de fraude é gerado para análise.

Como exemplos de técnicas dinâmicas de detecção têm-se:

Detecção de Padrões (Criação de Regras): Incrementa a experiência ou a intuição do Analista de Fraude com a aquisição contínua de conhecimento e de treinamento, o que permite a percepção de novos métodos de fraude e a definição das regras e padrões para encontrar mais fácil e rapidamente as fraudes;

Pontuação baseada em Redes Neurais: para cada novo caso de fraude, o sistema calcula uma pontuação como sendo a similaridade com os casos conhecidos, tanto aqueles que os analistas de fraude classificaram como fraude, como os casos classificados como não fraude. Tal estratégia torna o analista mais confiante na tarefa de decidir se o caso em foco é ou não fraude;

Rastreamento de Chamadas: esta técnica executa a análise detalhada das chamadas recebidas e originadas de/para assinantes suspeitos de fraude. A título de exemplo, um traficante de drogas, muito provavelmente, permanecerá em contato constante com outros criminosos através de aparelhos celulares. Através desta técnica, torna-se possível a visualização de todas as chamadas recebidas

e originadas pelo assinante suspeito. É possível, ainda, em modo gráfico, visualizar o encadeamento em vários níveis de um número desejado. Esta técnica é muito útil para descobrir prováveis novos fraudadores ou para a investigação policial;

Repositório de Dados e Mineração de Dados: oferecem técnicas avançadas de análise através de métodos estatísticos e de inteligência artificial e de refinamentos sucessivos, a partir de dados de alto nível descendo a níveis de detalhes cada vez maiores para uma análise interativa. Através destas técnicas, pode-se chegar à descoberta de novos padrões de fraude e a fraudes existentes ainda desconhecidas;

Assinatura (Impressão Digital) do Assinante: cria uma chave que identifica unicamente cada assinante por uma "assinatura" calculada baseado no "uso" e nos dados de tarifação. Um repositório de assinaturas é criado e é utilizado para verificar se novos assinantes não são fraudadores reincidentes.

4.2.2 Anomalias referente ao perfil de 'Uso'

Além da utilização em sistemas anti-fraude, podem-se detectar anomalias relacionadas ao Cliente. Por exemplo, o Cliente deseja que seu perfil de consumo siga um determinado padrão. Toda vez que o consumo for divergente do padrão desejado(de um fator ε), este fato deve ser informado. É bastante claro que este tipo de análise pode ser feito de diferentes maneiras, entretanto, através do uso de séries temporais, pode-se descobrir padrões, tendências e similaridades de forma mais simples.

4.3 Arquitetura do Sistema

A fim de enfrentar as questões citadas anteriormente, o SMT proposto utiliza as técnicas discutidas nos capítulos anteriores. Para saber se um perfil é similar a outro,

lança mão da Consulta por Similaridade. Para a realização da Consulta por Similaridade, os dados são preparados adequadamente, isto é, são convertidos em séries temporais que se submetem aos seguintes processos: transformações (como normalização), redução dos dados (devido ao grande volume de dados) e indexação através de índices espaciais (MAE). Por fim, para que se indiquem as ações a serem desencadeadas em função dos resultados das análises de similaridades, o SMT utiliza um Sistema Baseado em Conhecimento. Devido ao aspecto incremental da base de conhecimento, a estrutura RDR é utilizada. Para a realização destas tarefas, a arquitetura do SMT é modularizada, de modo a ser integrada aos sistemas corporativos de uma empresa, como, por exemplo, Sistemas Anti-Fraudes e *Billing*. Cada módulo será responsável por tarefas independentes. Desta forma, pode-se parametrizar o sistema conforme a necessidade. A figura[4.4] nos mostra um visão geral do SMT proposto. A seguir é

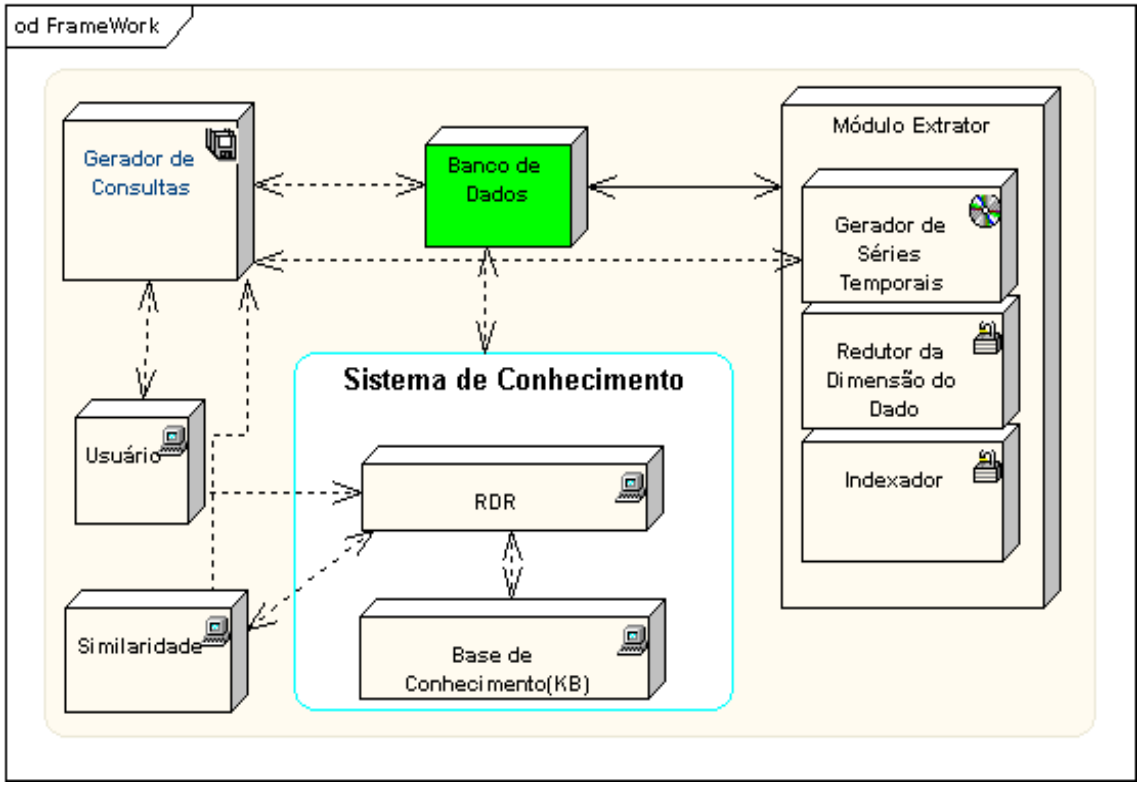

Figura 4.4: Visão Geral

apresentada uma visão geral de cada módulo:

- Banco de Dados (BD): Este módulo armazena dados sobre as ligações telefônicas dos clientes. As ligações são representadas por séries temporais armazenadas em tabela relacionais;

- Usuário: Representa o elemento que deseja fazer uma consulta ao sistema, podendo ser o especialista responsável por manter a base de conhecimento (BC) consistente ou alguém que queira interagir com o sistema;

- Gerador de Consultas: Responsável pela geração de consultas propostas pelo usuário. Este módulo permite a interação entre o usuário e o Sistema Baseado em Conhecimento, apresentando a consulta do usuário de uma forma compreensível pelo sistema;

- Extrator: Responsável pela geração, redução e indexação da Série Temporal;

- Similaridade: Responsável pelo cálculo de similaridade entre as Séries Temporais. A função de similaridade será utilizada pelas regras como um objeto. A função de similaridade será definida como $f(s_1, s_2, \varepsilon)$, onde s_1 e s_2 são séries temporais e ε é o parâmetro indicando a margem do erro. O cálculo da distância entre as séries s_1 and s_2 é estimada por meio da distância euclidiana (Eq.4.3.1).

- Base de Conhecimento (BC): Armazena regras que representam o conhecimento;

- Sistema Baseado em Conhecimento (SC): Responsável pela construção incremental da base de conhecimento e pelo processo de inferência.

Nas próximas seções os módulos da figura [4.4] serão detalhados.

4.3.1 Módulo Extrator

Para que o SMT consiga realizar as Consultas por Similaridade, a primeira atividade a ser desenvolvida deverá ser a transformação dos dados em séries temporais a serem consultadas pelo sistema.

Telefone Origem	Telefone Destino	Data	Tempo
1632115100	1699769196	20050707	39
1632115100	1699769196	20050707	71

Tabela 4.1: Exemplo de arquivo CDR processado pelo Mediador

Conforme dito anteriormente, quando uma ligação telefônica é realizada, as informações são registradas pelas centrais telefônicas e armazenadas em arquivos chamados CDR's. Nas Companhias Telefônicas, existem sistemas específicos que tratam estes arquivos, os quais são conhecidos como Mediadores. A idéia central dos Sistemas Mediadores é capturar os arquivos gerados pelas centrais telefônicas (CDR's) e os converterem em arquivos formatadas cujos *layouts* sejam inteligíveis para os sistemas que os utilizarão. No caso da arquitetura proposta, eles correspondem ao Módulo Extrator. Isto se faz necessário porque as centrais telefônicas são de diferentes fabricantes, como as Centrais Huawei, Siemens, Ericsson etc. Cada central gera seus arquivos de CDR em um formato próprio, como por exemplo, em formato binário. A tabela 4.1 ilustra um arquivo extraído de uma central *Huawei* após passar pelo Sistema Mediador. O Módulo extrator será o responsável pela transformação dos dados relativos às ligações (armazenadas no arquivo gerado pelo Sistema Mediador), em séries temporais, pela redução da dimensão dos dados e pela indexação dos mesmos. Para manter as informações relativas às ligações telefônicas dos clientes atualizadas, o Módulo Extrator constantemente recorre ao Banco de Dados (BD). A seguir, são apresentados os 3 sub-módulos do Módulo Extrator, isto é, o gerador de séries temporais, o redutor da dimensão dos dados e o indexador.

Gerador das Séries Temporais

As Séries Temporais serão construídas a partir dos seguintes dados capturados dos Sistemas Mediadores: número do telefone, duração da chamada, data/hora de

inicio da chamada[3]. Conforme dito anteriormente, tais informações poderão estar armazenadas em arquivos textos ou em tabelas de bancos de dados (bancos relacionais ou orientado a objetos). O Módulo Extrator gerará uma série temporal para cada conjunto de informações relativas a cada linha telefônica. Para tanto, ele poderá trabalhar de duas maneiras diferentes: On-line ou em Lote (*Batch*), dependendo da aplicação. Se for On-line, este processo deverá ficar monitorando a chegada de informações relativas às ligações telefônicas no Sistema Mediador. Se a decisão for processar em Lote, o Módulo poderá buscar as informações em intervalos de tempo pré-determinados.

Assume-se, como premissa, que os tamanhos (número de pontos) das Séries Temporais serão iguais, o que é perfeitamente aceitável, pois todas as séries contém zero ou mais ligações realizadas. A figura 4.5 representa as séries temporais referentes a dois telefones. Observe que o eixo $\boldsymbol{y}$ representa a duração das chamadas em segun-

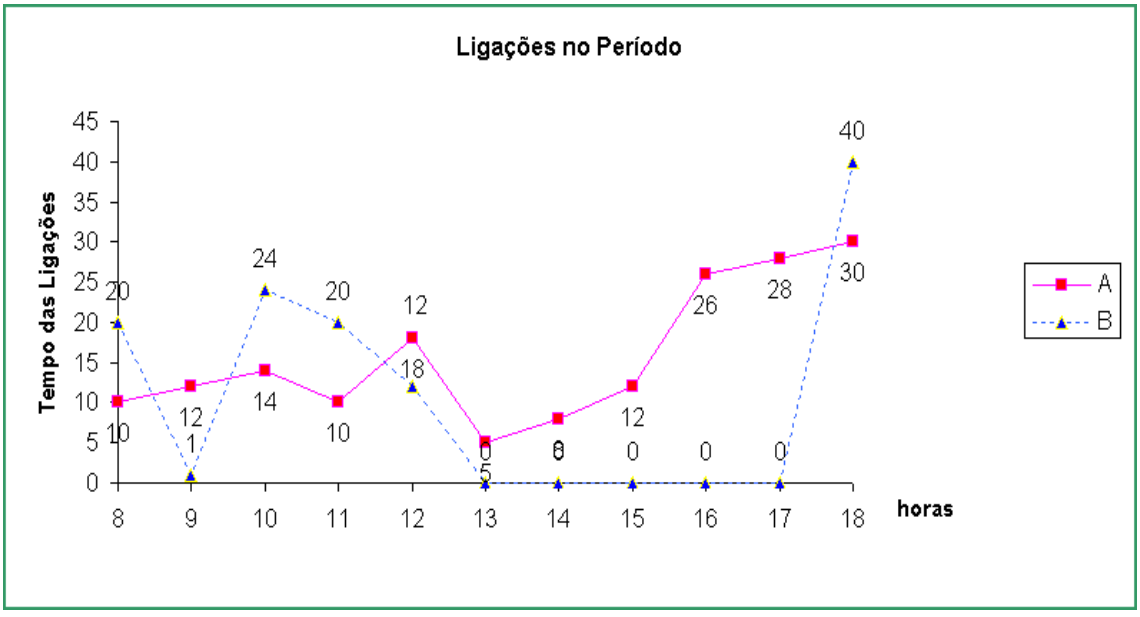

Figura 4.5: Série Temporal representando ligações telefônicas

dos[4] e o eixo $\boldsymbol{x}$ representa a hora do início da chamada. O telefone A fez ligações em

[3]Poderiam ser considerados, também, o valor das chamadas e o total de ligações.
[4]Poderia ser outra unidade de tempo, como minuto, hora etc.

todos os intervalos considerados, enquanto o telefone B não fez ligações nos instantes $x = 13, 14, 15, 16$ e 17 (nesses instantes houve zero ligações). Logo, as duas séries temporais (A e B) têm o mesmo número de pontos, ou seja, 11 pontos. As séries temporais podem ser analisadas considerando diferentes intervalos. Será utilizado o tempo de 1 hora como sendo a Granularidade da série[5], isto é, todas as Séries terão a Duração de Chamadas acumuladas em um período pré-determinado de 1 hora[6]. Desta forma, toda série terá apenas 1 (um) ponto representado por período de 1 hora. Se não houver ligações no período, não haverá necessidade de uso de métodos de interpolação, pois, neste caso, o valor do ponto é 0. Considere-se, por exemplo, que o Telefone T_1 tenha efetuado ligações conforme a tabela 4.2. A série temporal

Telefone Origem	Data e Hora da Chamada	Duração em Segundos
1632115100	01/01/2005 08:00	10
1632115100	01/01/2005 08:30	50
1632115100	01/01/2005 09:00	30
1632115100	01/01/2005 10:00	20
1632115100	01/01/2005 11:00	0

Tabela 4.2: Ligações do telefone T_1

$\overline{T_1}$ que representa as ligações do telefone T_1, terá os pontos (x, y), onde x e y correspondem respectivamente à Data/Hora e a duração da chamada acumulada, ou seja: { (01/01/2005 08:00,60), (01/01/2005 09:00,30),(01/01/2005 10:00,20), (01/01/2005 11:00,0) }. O fluxo do processo de geração das séries temporais pode ser visto na figura 4.6. Para representar as séries temporais, baseou-se na proposta apresentada

[5]A Granularidade é a distância mínima entre dois pontos sucessivos de uma Série.

[6]Este período poderá variar dependendo do perfil os clientes. Se o volume de ligações for muito alto, podem-se considerar períodos menores. Se o volume de ligações for relativamente baixo, pode-se considerar 1 dia, por exemplo.

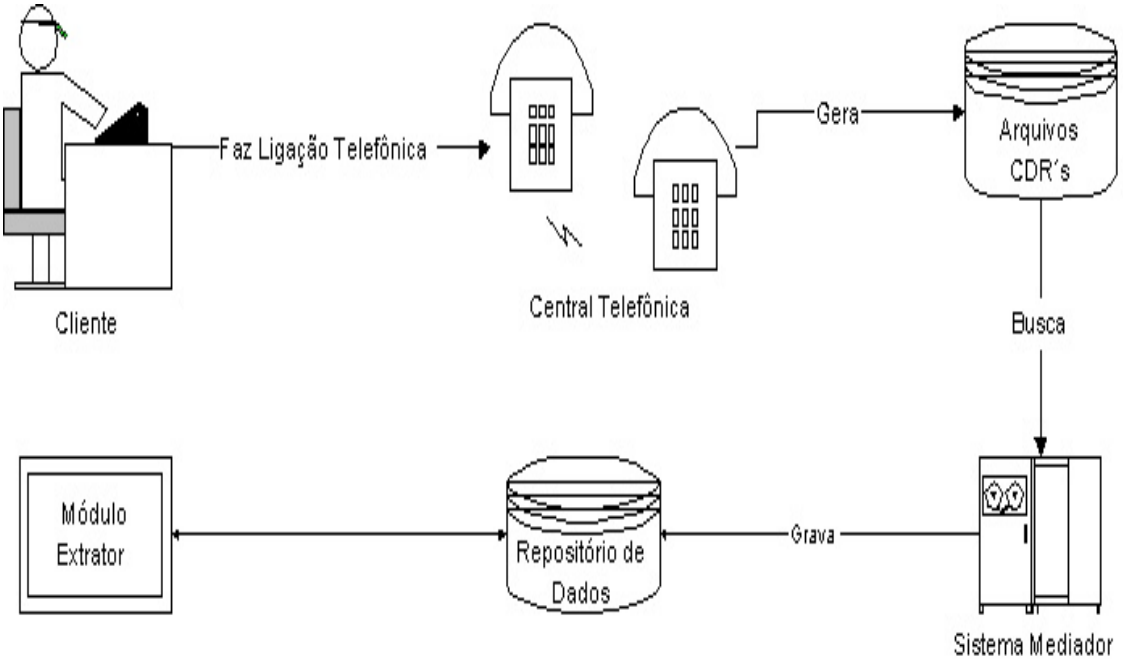

Figura 4.6: Fluxo do processo de Geração das Séries Temporais

em [10].

$$< id, (t_0, v_0^1, ...v_1^k), ..., (t_{n-1}, v_{n-1}^1, ...v_{n-1}^k) > \tag{4.3.1}$$

Esta proposta foi utilizada por ser de fácil implementação em sistemas de computadores. Ela pode ser implementada em Bancos de dados Relacionais, Bancos Orientados a Objetos etc. A tabela 4.3 mostra como a Série Temporal poderá ser representada como uma Tabela. A Série Temporal criada poderá ser identificada pelo atributo

Telefone			
t_0	v_0^1	...	v_0^k
...	...	...	...
t_n	v_n^1	...	v_n^k

Tabela 4.3: Modelo visto como Tabela

"Telefone"que se refere ao número do telefone utilizado pelo Cliente[7], $< t_0, ...t_n >$ são as datas em que ocorreram as ligações e $< v_0^1, ..., v_n^1 >$ são os valores de duração das ligações telefônicas acumulados no período pré-determinado. A representação final é vista na tabela 4.4. A seguir, serão apresentadas as Estruturas que representarão as

Telefone	
Data$< t_0 >$	Duração$< v_0^1 >$
...	...
...	...
...	...
Data$< t_n >$	Duração$< v_n^1 >$

Tabela 4.4: Representação da Série

Séries Temporais. Mostramos apenas as principais propriedades e algumas operações. As propriedades iniciam com a letra ***p*** e as operações iniciam com a letra ***o***. Os nomes das operações são auto explicativos.

A Estrutura TimePoint

```
TimePoint
    Date  pPoint ;
    integer oDay();
    integer oMonth();
    integer oYear();
    integer oHour();
```

Esta estrutura **TimePoint** representa a Data que o ocorreu o evento da Série Temporal, que no caso será a Data /Hora que foi realizada a ligação telefônica. Por exemplo:

```
TimePoint Data ;
    Data.pPoint = '01/04/2004 16:00' ;
```

[7]Não se consideram aqui todas as peculiaridades do processo, como o fato de um mesmo telefone poder pertencer a vários clientes em momentos diferentes.

A Estrutura DataColumn

```
DataColumn
    string  pColumnName ;
    integer pColumnType ;
    real    pColumnValue;
```

A estrutura **DataColumn** representa os dados que serão armazenados na série temporal, que no nosso caso será a Duração das Chamadas. Por exemplo:

```
DataColumn Coluna ;
 // Nome da coluna
 Coluna.pColumnName = 'Duration';
 //Tipo de Dados da coluna
 Coluna.pColumnType = 'Real';
 //Valores acumulados indicando
 //a duracao das chamadas em segundos
 Coluna.pColumnValue=  15.2;
```

A estrutura TimeSerie

```
TimeSerie
 TimePoint  Point
 DataColumn Columns[]
 void    oAddPoint(TimePoint, ColumnValue)
 boolean oDeletePoint(TimePoint)
 void    oSumValue(ColumnName, ColumnValue)
 integer oFindPoint(TimePoint)
```

A estrutura **TimeSerie** representa um (1) elemento da Série Temporal. Por exemplo:

```
TimeSerie  Serie;
 //Data do evento
 Serie.Point = '01/04/2004 15:00' ;
 Serie.Columns = [('Duration','Real',15)];
```

A estrutura TimeSeries

```
TimeSeries
  string    SerieName
  string    NumPhone
  TimePoint DateCreate
```

```
TimePoint LastUpdate
TimeSerie Serie[]
integer oAddRow(TimePoint, DataColumn)
boolean oUpdate(Row)
boolean oInsert(Row)
void    oImport(path, arqname)
integer oFind(NumPhone)
boolean LoadSerie(NumPhone)
integer oLength(NumPhone)
```

A estrutura **TimeSeries** representa uma Série Temporal completa, ou seja, contém informações como o telefone, data/Hora e duração das chamadas.

```
TimeSeries  SerieTemporal;
 //Nome da Serie
 SerieTemporal.SerieName = 'Serie do Umberto';
 //Codigo de Area e Numero do Telefone
 SerieTemporal.NumPhone   = '3432562725'
 //Data de criacao da Serie Temporal
 SerieTemporal.DateCreate=  Date()
 //Data da ultima atualizacao da Serie
 SerieTemporal.LastUpdate=  Date()
 //Primeiro elemento da Serie
 SerieTemporal.Serie[1]=['01/01/2004 15:00',
                      ('Duration','Real',1500)
                        ]
 //Segundo Elemento da Serie
 SerieTemporal.Serie[2]=[ '02/01/2004 16:00',
                      ('Duration','Real',750)
                        ]
```

A figura 4.7 mostra a representação de uma série temporal após passar pelo processo descrito anteriormente. A seguir, serão apresentados os demais componentes do Módulo Extrator.

Módulo Indexador e Módulo Redutor da Dimensão do Dado

O tipo de consulta que será realizada nas séries temporais geradas é uma definição muito importante para o sistema e depende muito do domínio de atuação. Para a detecção de anomalias baseadas no perfil de "uso" do telefone, pode-se analisar a

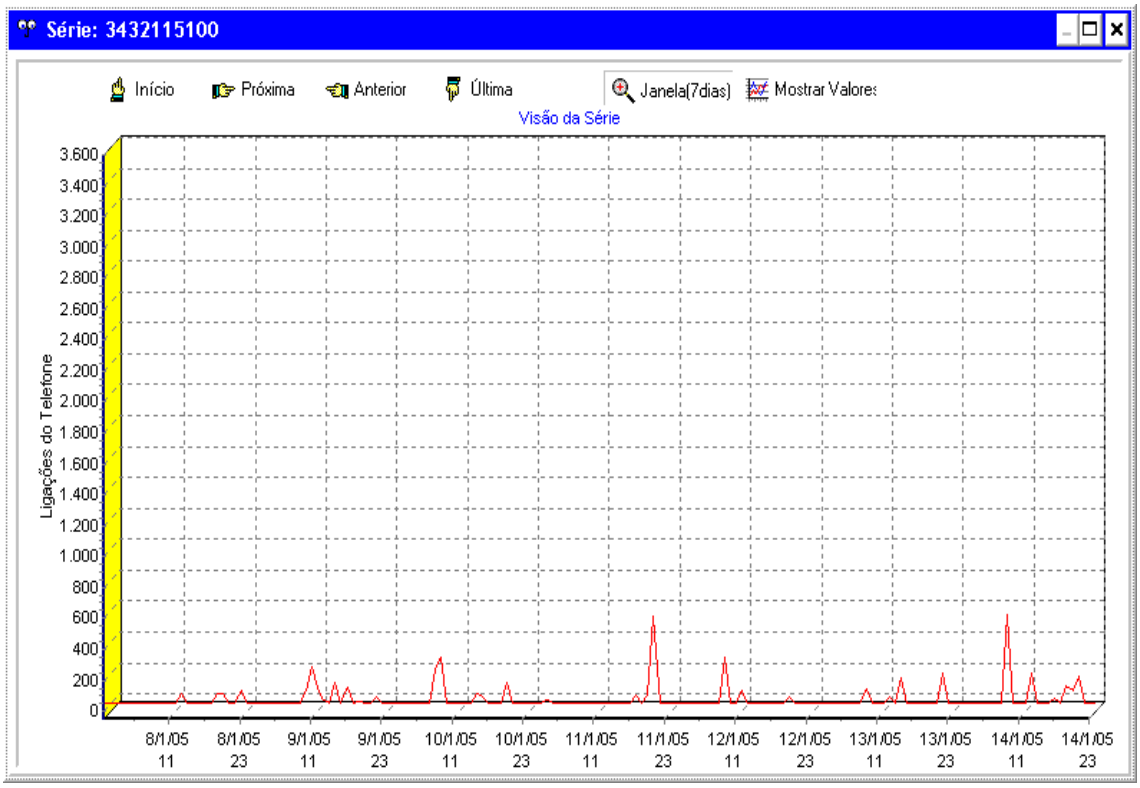

Figura 4.7: Série Temporal obtida dos arquivos extraídos das centrais telefônicas

série do telefone em intervalos semanais, mensais, diários, ou, também pode-se analisar toda a série de uma só vez. Por exemplo, considere-se as consultas Q_1 e Q_2 a seguir:

Q_1: *"As ligações do cliente (PR) seguem um Perfil Desejado (PD) pré-estabelecido por ele mesmo?"*,

Q_2: *"O Telefone X tem, em algum intervalo, um comportamento similar a um padrão de fraude (PF) conhecido?"*,

Para responder às consultas Q_1 e Q_2, primeiramente, é necessário criar uma série temporal que represente o PR do telefone. O PD e PF citados em Q_1 e Q_2 podem ser mensais, semanais, diários etc.

Para resolver consultas como estas, o SMT usa a estratégia *Subsequence Matching*, podendo adotar os tipos de consulta por faixa ou dos k-vizinhos mais próximos (ver seção 2.2.3).

Baseado neste tipo de problema, decidiu-se utilizar no SMT, a *SubSequence Matching*. Além disso, as consultas dos k-vizinhos mais próximos e por faixa podem ser utilizadas sem nenhum problema.

Esta escolha se deveu ao tipo de problema que se está propondo resolver, ou seja, para se analisar se um determinado PR de ligações telefônicas é similar a um PD/PF em um período P de tempo (*slide window* ou janela de pesquisa), serão feitas sucessivas comparações (análise de similaridade) entre o PR e PD/PF, tantas quantas forem as quantidades de períodos P contidos no PR. Suponha-se que a janela tenha 7 dias (P = 7)[8] e o tamanho do PR seja de 11 dias. Neste caso, a PR que representa o consumo do cliente deverá ser passível de análise em sucessivos intervalos P, também de 7 dias, conforme mostra a figura 4.8.

A figura 4.8 mostra as análises de similaridade seqüenciais efetuadas para resolver uma consulta que visa identificar se em algum intervalo P, o PD deixou de ocorrer na série real. Note que o mesmo tipo de pesquisa poderia ser feito visando a detecção de fraude, conforme indagado na consulta Q_2 apresentada anteriormente.
Para pesquisar todos os intervalos P no PR que são similares ao PD, seria necessário compará-los com a janela em cada valor no eixo x, ou seja, 11 (tamanho da serie) - 7(tamanho da janela)+1 vezes. Neste caso, o PD é comparado com cada subseqüência de mesmo tamanho do PR, sendo que uma nova comparação de subseqüência ocorre a cada ponto do PR. Ainda no contexto das buscas seqüenciais, uma outra maneira de se efetuá-las corresponde a executar as comparações de subseqüências a partir de cada ponto do PR que seja múltiplo do tamanho da janela. Isso corresponde a uma análise de anomalias semanais, caso P seja igual a 7 dias, conforme mostra a figura 4.9. Para realizar tais comparações, ou seja, PD e subseqüências do PR, pode-se utilizar a busca seqüencial. Como a busca seqüencial é realizada através de comparações de subseqüências em pontos sucessivos, ela pode não ser uma boa estratégia em alguns casos. Por exemplo, ela representa uma boa técnica quando se trata da consulta

[8]Esta escolha é perfeitamente aceitável, pois corresponde, exatamente, ao número de dias da semana.Este número dependerá de heurísticas feitas previamente e mudará conforme a aplicação.

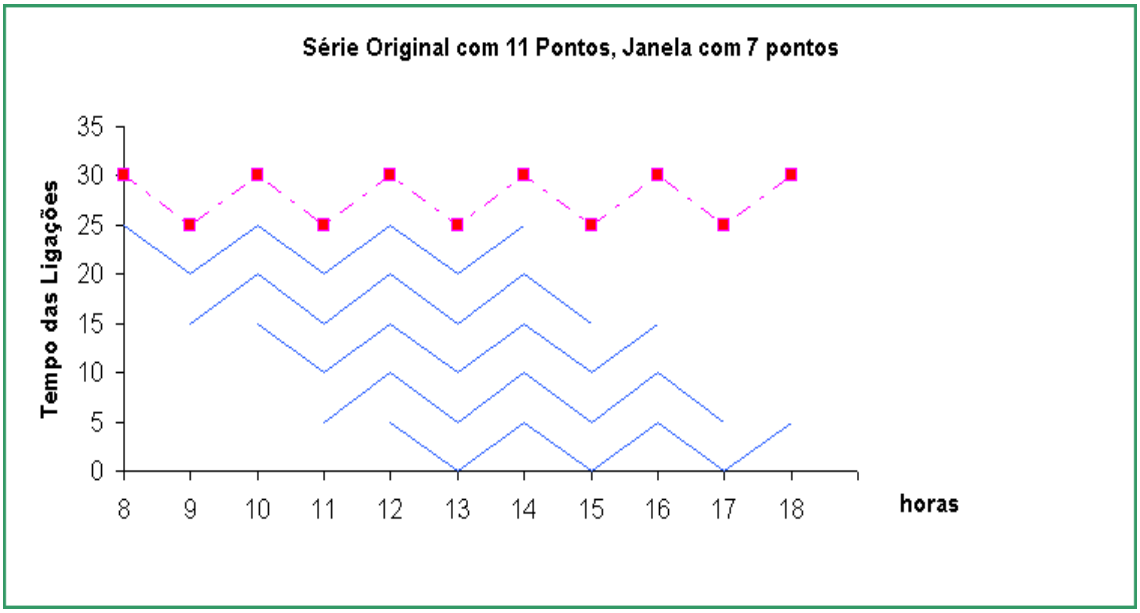

Figura 4.8: Forma de pesquisar uma série deslocando a janela de tamanho 7

expressa em Q_1, mas pode não ser o caso no tratamento de Q_2. Uma alternativa para as situações em que ela não tem bom desempenho, é a criação de índices de busca nas séries temporais. Recomenda-se que a técnica da indexação seja utilizada nos casos em que no máximo 20% dos dados pesquisados precisem ser recuperados [36].

Conforme apresentado nos capítulos anteriores, uma série temporal pode ser considerada um ponto no espaço, logo, pode-se utilizar um índice espacial (MAE). Porém, devido a alta dimensionalidade (número de pontos) das séries temporais que representam as ligações telefônicas, seria inviável a criação de um índice espacial (relembre-se que um índice espacial se torna muito lento à medida que a dimensão aumenta [36]). O caminho para resolver este impasse é a utilização da redução da dimensão do dado. Esta decisão dependerá do tipo de consulta que será feita. No domínio que está sendo proposto, percebe-se, claramente, que será necessária a realização da redução dos dados, pois para cada série temporal T de tamanho lT[9] e *window* de tamanho lW, terão que ser criadas N subseqüências, onde N é calculado

[9]O tamanho da série temporal aumenta conforme passa o tempo, isto é, a cada ligação telefônica a série aumenta de 1 ponto.

pela fórmula a seguir:

$$N = Mod(\frac{lT}{lW}) \tag{4.3.2}$$

onde *Mod* é a função que retorna a parte inteira da divisão. Por exemplo, em caso de análises de anomalias semanais, tal como ilustrado na figura 4.9, considerando-se que a granularidade utilizada seja de 1 hora, cada subseqüência terá $W = 24$ (horas por dia)*7 (dias da semana) = 168 pontos. Para que se consiga a redução dos dados,

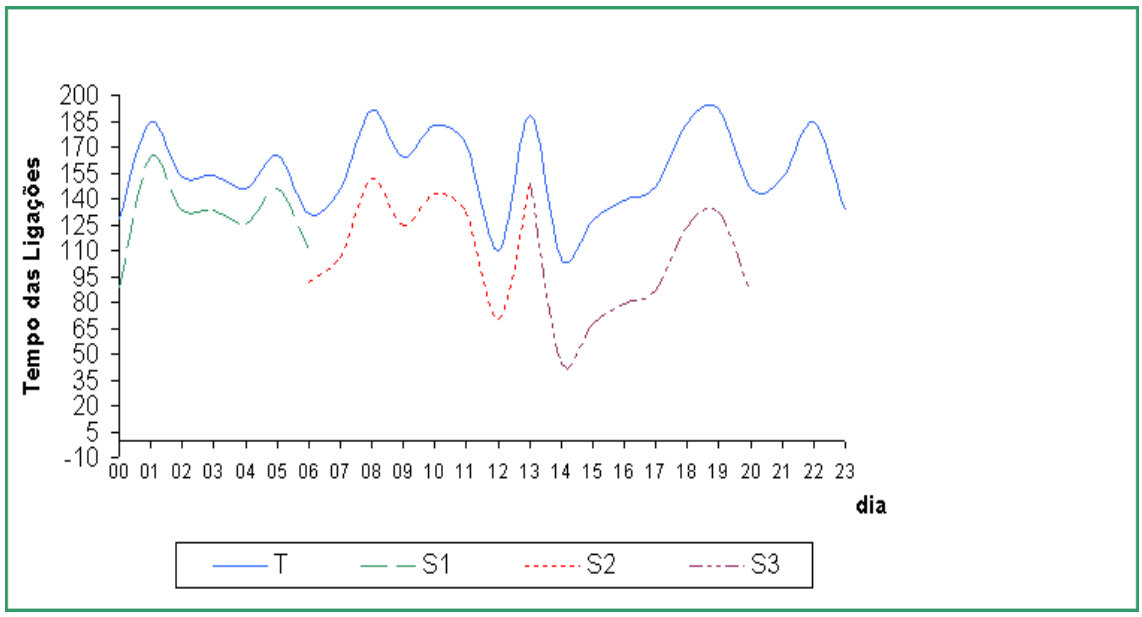

Figura 4.9: Subseqüências geradas para consulta uma determinada série temporal

inicialmente precisa-se definir se haverá necessidade de aplicar alguma transformação nas séries temporais antes de se criar o índice. Para a detecção de anomalias que está sendo proposta, é necessário verificar se existe alguma série na qual ocorra um PD ou PF. Esta análise será feita considerando-se, apenas, a forma geométrica das séries temporais. Mediante estes fatos, tem-se que aplicar a equação 2.2.9 sobre os pontos das séries geradas, ou seja, realizar a normalização[10].

[10] A normalização é opcional, dependerá da aplicação.

A próxima etapa consiste em optar por um método de redução de dados. Decidiu-se utilizar as *Haar Wavelets* por se tratarem de uma técnica bastante promissora, rápida computacionalmente e respaldada por artigos que demonstram sua superioridade em relação a outros métodos [36, 9]. Na verdade, a técnica de redução a ser utilizada poderia ser parametrizada, ou seja, poderiam ser consideradas, além das *Haar Wavelets*, outras técnicas como a PAA citada anteriormente. A próxima etapa é utilizar a função de *Feature Extraction*. Esta função serve para selecionar os coeficientes *Wavelets* que melhor representarão a série temporal em questão. Uma boa maneira de conservar a energia de uma série temporal com apenas k coeficientes *wavelet*, $k < lT$, é manter os k coeficientes mais significativos, isto é, os coeficientes com os maiores valores absolutos [48]. A criação do índice ocorre durante o seguinte processo:

1. Para cada novo ponto inserido no PR, verifica-se se $Mod\ (\frac{lT}{lW}) = 0$, com $lT \geq lW$. Caso seja verdade, uma nova subseqüência deverá ser inserida no índice. Ela terá seu primeiro ponto em $lT - lW + 1$

2. Aplica-se a transformação (normalização) na subseqüência, usando a equação 2.2.9.

3. Caso necessário, completa-se a subseqüência com zeros até chegar em um número de pontos que seja potência de 2 (As *wavelets* devem ser potências de dois).

4. Calculam-se os coeficientes utilizando *Haar Wavelets*

5. Realiza-se a *Feature Extraction* considerando os k coeficientes, $k \geq 7$ e $k \leq 12$ mais significativos (o número de *Feature Extraction* ideal deverá ser obtido através de experimentos, o que será ilustrado na seção 4.3.2).

6. Inserem-se os coeficientes no MAE (no presente caso, usando-se o *R-Tree*)

Estes passos podem ser vistos no fluxo ilustrado na figura 4.10. Para exemplificar

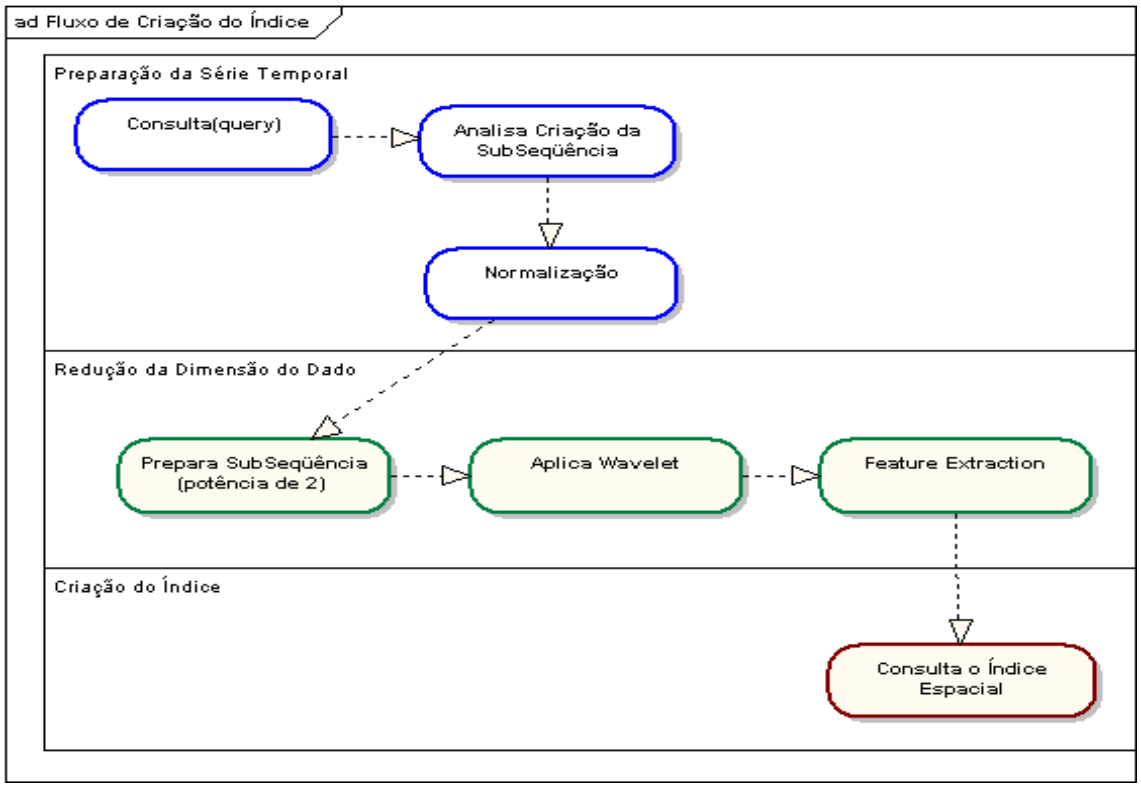

Figura 4.10: Fluxo de criação do Índice

o fluxo descrito, suponha-se a seguinte consulta Q_3:*"Existe algum Cliente que tenha suas ligações telefônicas, em alguma semana, similar ao perfil de fraude PF?"*. Consultas desta natureza ilustram a necessidade da criação de índices, pois, para responder a tal pergunta, o SMT deve varrer toda a base de dados a procura de clientes cujos PR's sejam similares ao PF em questão. Se não houvesse índices, o SMT deveria efetuar a pesquisa de similaridade entre PF e as subseqüências de cada PR seguindo a ordem em que as últimas estão dispostas em PR. Como existem várias PR's na base de dados, um para cada telefone, a busca seqüencial se torna inviável. Os gráficos apresentados nas figuras 4.11 e 4.12 mostram o resultado de alguns experimentos realizados que comprovam esta afirmação. Para isso, realizaram-se várias vezes a mesma consulta utilizando a busca seqüencial e a busca utilizando o índice espacial. O resultado apresentado é referente à consulta Q_3 sendo realizada várias vezes no

mesmo computador e com a mesma configuração[11], porém, a cada teste foram-se adicionando novas séries à base de dados. Observa-se que à medida que aumenta a quantidade de séries a serem analisadas, a busca seqüencial se torna cada vez mais lenta. Note que, tanto na figura 4.11 quanto na figura 4.12, são mostrados resultados

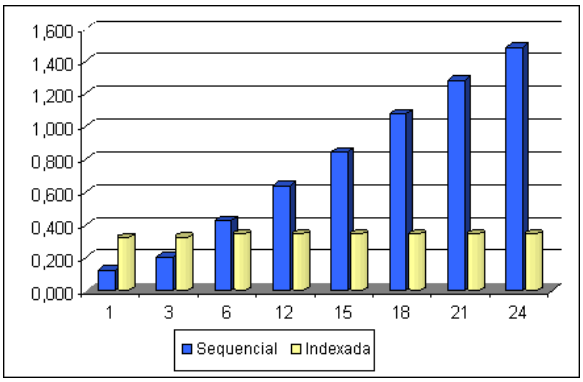

Figura 4.11: quantidade de séries X tempo da consulta em segundos

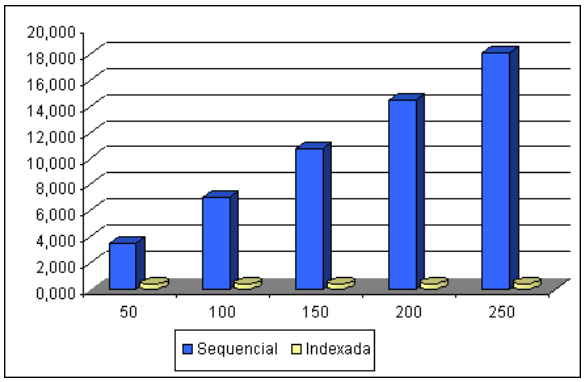

Figura 4.12: quantidade de séries X tempo da consulta em segundos

de diversos testes utilizando busca seqüencial e busca indexada. Para cada novo teste, há um incremento na quantidade de séries analisadas. Contudo, na figura 4.12 os testes são efetuados para quantidades de séries bem maiores (maiores volumes de chamadas).

Prosseguindo com o exemplo, para a criação do índice que auxiliou a execução da consulta Q_3, considerou-se: tamanho da janela de 7 dias, isto é $lW = 7$ e subseqüências semanais, ou seja, apenas intervalos múltiplos de lW.

A primeira atividade a ser feita, conforme o fluxo da figura 4.10, é a preparação das PR's. O sub-módulo gerador de séries temporais captura as informações relativas aos CDR's, e insere as informações no banco de dados formatando-as como séries temporais. A visualização de uma destas séries temporais, S_1, é vista na figura 4.13. A geração das séries temporais é um processo com desempenho aceitável. Em experiências realizadas, a geração de 5.000 séries, com uma base de dados inicialmente vazia, foi de aproximadamente 2,5 Horas[12]. Conforme ocorre a geração do PR, as

[11] Computador com 1.8Hz, 256Mb.

[12] Observe que este processo deve ser realizado apenas 1 vez, depois haverá apenas atualizações na

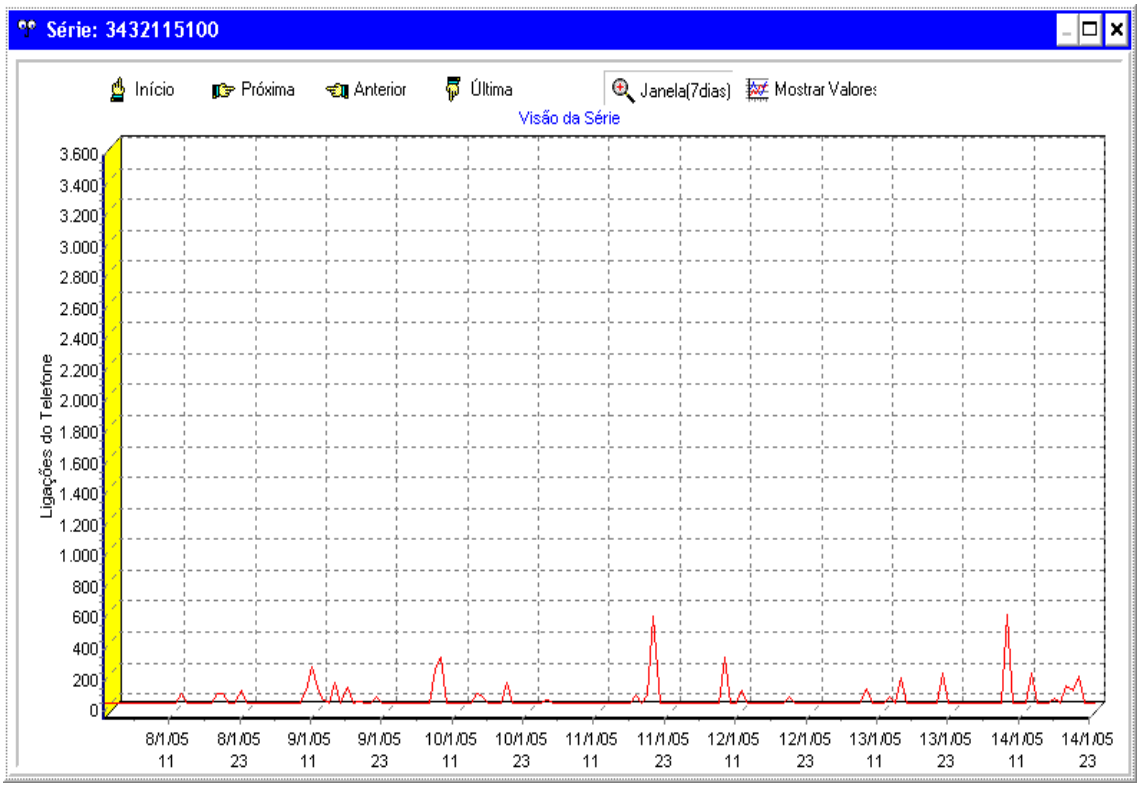

Figura 4.13: Série Temporal obtida dos arquivos extraídos das centrais telefônicas

subseqüências de tamanho lW também são analisadas e geradas, ou seja, a cada janela de 7 dias é gerado 1 subseqüência. A figura 4.14 mostra o mesmo PR, S_1, à frente, e ao fundo, as subseqüências geradas. Observe que as subseqüências são idênticas ao PR, porém, têm tamanho lW. O próximo passo é normalizar as subseqüências. Este passo é necessário para fazer as subseqüências invariáveis ao *shifting e scaling*. Para isso, basta aplicar a equação 2.2.9 em todos os pontos da série. Na figura 4.15 é mostrada a série S_1 com as subseqüências normalizadas. A normalização dependerá muito da análise que se deseja realizar. Por exemplo, para analisar a similaridade entre o PD/PF e o PR, têm-se que definir quais informações serão consideradas, ou seja, serão consideradas as diferenças relacionadas ao deslocamento no eixo y? Serão consideradas as diferenças em relação à escala? Na detecção de anomalia proposta, o interesse está na similaridade entre PR em relação ao PD/PF, considerando apenas

medida em que as ligações telefônicas são realizadas.

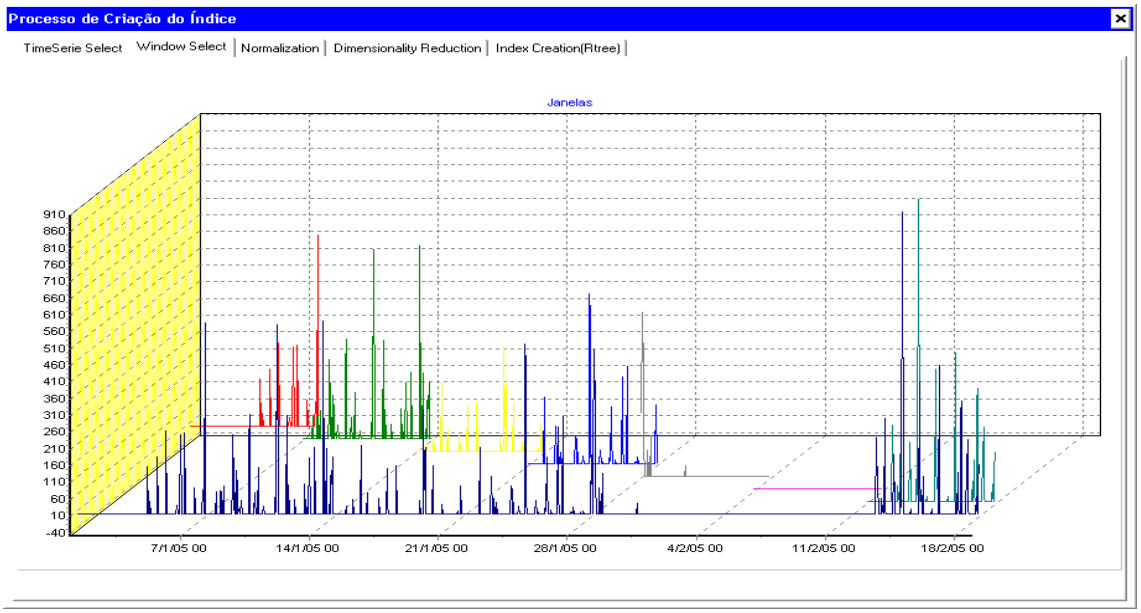

Figura 4.14: Processo de criação do Índice

a forma geométrica destas séries. Se o PR e o PD/PF apresentarem similaridade, não importa se houve mais ou menos ligações de uma série em relação a outra. Para ilustrar esta análise, suponha a seguinte situação: "O cliente definiu seu perfil desejado da seguinte maneira: Aos sábados e domingos o número de ligações de minha empresa é baixo, na segunda o número de ligações é alto, na terça e quarta o volume de ligações é médio, e na quinta e sexta o volume de ligações é de médio para baixo", ou seja, o cliente está interessado no comportamento das ligações (não na quantidade), se ultrapassar um limite muito diferente em relação ao estabelecido, alguma ação deve ser tomada. Após a normalização das subseqüências, elas devem ser preparadas para a indexação com o índice espacial. O primeiro passo é realizar a redução dos dados. Para tanto, deve-se verificar se o tamanho das subseqüências é potência de 2 (exigência para poder utilizar as *wavelets*). Como o tamanho das subseqüências é de 7 dias, o que corresponde a $7 * 24 horas = 168 pontos$, observa-se que 168 não é potência de 2. Através de fórmulas matemáticas calcula-se a próxima potência de 2 maior do que 168. Neste exemplo usou-se a fórmula: $pot = round((\ln(168)/\ln(2)) + 0.5)$ para calcular a potência de 2, e complementa-se a subseqüência a partir do último ponto, com o valor

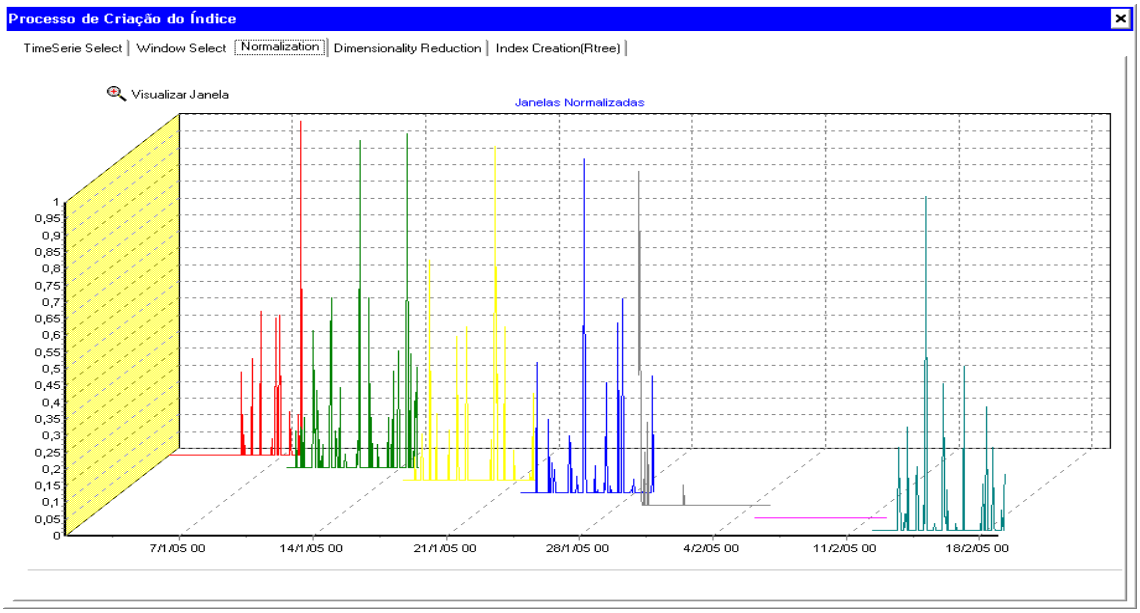

Figura 4.15: Subseqüências normalizadas (PR)

zero (0) até 2^{pot}. A partir da subseqüência complementada com zeros, aplicam-se as *Haar Wavelets*, obtendo-se os coeficientes desejados. Utilizou-se o algoritmo a seguir para calcular os coeficientes *wavelet*[13].

```
j   = Nro de Pontos da Serie;
vet = Pontos da Serie;
//vetWavC corresponderá aos coeficientes de aproximacao
//vetWavD corresponderá aos coeficientes de detalhe
//sqrt = raiz quadrada, fator de normalização
Enquanto (j > 1 )
   J := j div 2 ;
   Para  k = 0 .. j-1
      vetWavC[i,k] := vet[2*k] + vet[2*k+1]) / sqrt(2.0) ;
      vetWavD[i,k] := vet[2*k] - vet[2*k+1]) / sqrt(2.0) ;
   Fim Para;
   vet = vetWavC;
Fim Enquanto;
```

Conforme explicado no capítulo 2, a utilização das *Haar Wavelets* garante que a distância euclidiana no espaço reduzido é preservada e que o *Falso-Negativo* não irá ocorrer, conforme demonstrado em [9]. Estas propriedades são fundamentais para a utilização dos coeficientes *wavelets* durante o processo de indexação. Para comprovar

[13] O algoritmo está em pseudo-código para um melhor entendimento.

estas propriedades, foram testadas diversas consultas segundo duas estratégias: a primeira, utilizando *wavelets* e, a segunda, considerando os pontos com maiores valores absolutos, sem utilizar as *wavelets*[14]. As figuras 4.16 e 4.17 representam um desses testes. No caso, o objetivo é detectar, dentre um conjunto de PR's, os dois que mais se aproximam do PF apresentado na figura 4.16.

O gráfico da figura 4.16 representa o PF, não normalizado, com todos os pontos da série a ser comparado com os PR's (também não normalizados e com todos os pontos). O gráfico da figura 4.17 apresenta uma consulta dos k-vizinhos mais próximos, onde $k = 2$, realizada em uma série utilizando *wavelet* (parte inferior) e mesma consulta sem a utilização das *wavelets* (parte superior). Observe que a distância euclidiana de valor 1989.2215 calculada entre o PF e a segunda série apresentada como solução na estratégia que não usa *wavelets*, (figura 4.17, parte superior à direita), é maior do que a distância de valor 1760.6442 calculada entre o PF e a segunda série apresentada como solução na estratégia que usa *wavelets* (figura 4.17, parte inferior, à direita). Isto indica que houve um *Falso-Negativo*, pois o resultado da consulta (parte superior) apresentou uma série candidata que, na realidade, não deveria fazer parte da solução, pois, a distância real entre o PF e a série candidata, na figura superior, é maior do que a distância entre o PF e a série candidata apresentada como solução na figura inferior (cabe relembrar que a consulta visa obter as duas séries candidatas mais próximas do PF).

O próximo passo do fluxo descrito na figura 4.10 é a utilização da função de *Feature Extraction* para selecionar os coeficientes *wavelets* que melhor representarão a série temporal. Utilizou-se, ainda na execução da consulta Q_3, os 7 coeficientes mais significativos em valor absoluto (na seção 4.3.2 são apresentados alguns resultados obtidos variando-se o número de coeficientes e a função de *Feature Extraction*). Desta forma, os dados foram reduzidos de 168 pontos para apenas 7 pontos.

Após a redução da dimensão do dado, a subseqüência contendo apenas 7 pontos

[14]Esta heurística foi utilizada porque as séries apresentam muitos valores iguais a zero.

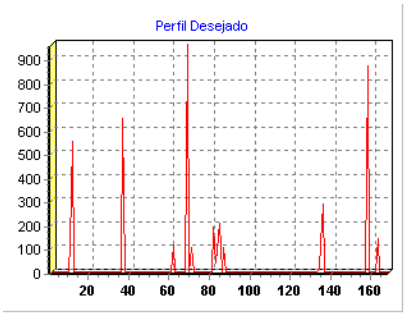

Figura 4.16: PF, Tempo X Pontos

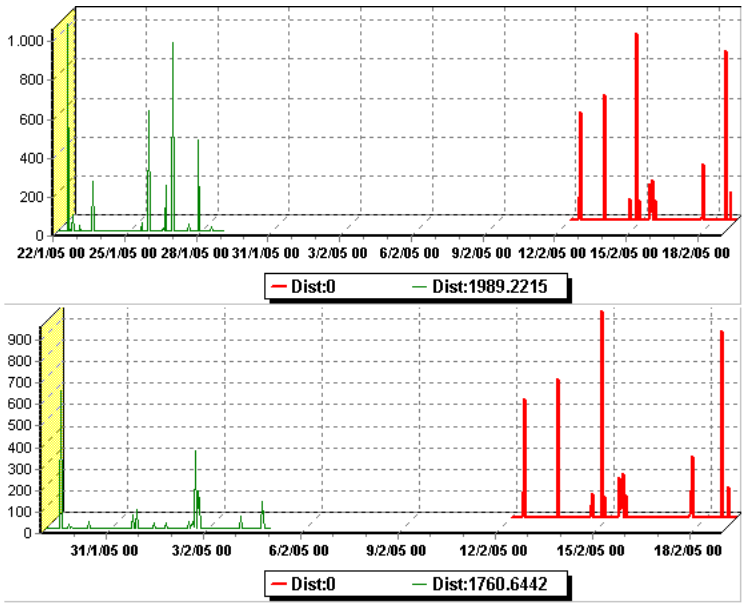

Figura 4.17: distâncias euclidianas reais entre PF e PR

pode ser vista como um ponto de dimensão 7 no espaço. Tal ponto representa uma série temporal. No último passo, faz-se a indexação deste ponto em um índice espacial (no exemplo foi utilizado o *R-tree*). A figura 4.18 mostra um gráfico que representa o desempenho do processo que realiza a criação do índice, considerando dimensão de 7. Neste gráfico, observa-se que o processo pode tornar-se muito lento se o número de séries for muito grande. Uma outra observação importante diz respeito ao tempo

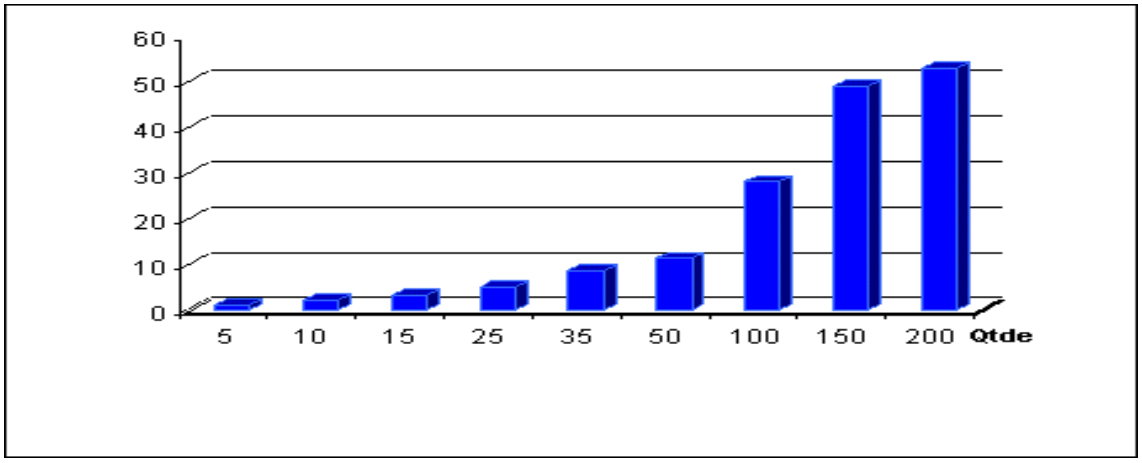

Figura 4.18: Qtde de Séries X Tempo Gasto na geração dos índices (segundos)

gasto pelo processamento referente à criação do índice, porém, separando o tempo de cada processo, ou seja, à normalização, redução da dimensão, função de *Feature Extraction* criação do índice separadamente. Esta observação pode ser vista nas figuras 4.19 e 4.20. A figura 4.19 representa o tempo gasto por cada processo (em segundos) em relação ao número de dimensões, e a figura 4.20 representa o percentual de cada processo no tempo total do processo[15]. Por meio destes experimentos, nota-se que a normalização é responsável por mais de 50% do tempo total gasto. Nota-se, também, que o tempo gasto com o cálculo dos coeficientes *wavelets* e a função de *Feature Extraction* é relativamente pequeno. Como estes processamentos envolvem apenas cálculos, acredita-se que, em computadores mais robustos, este tempo de processamento seja muito menor.

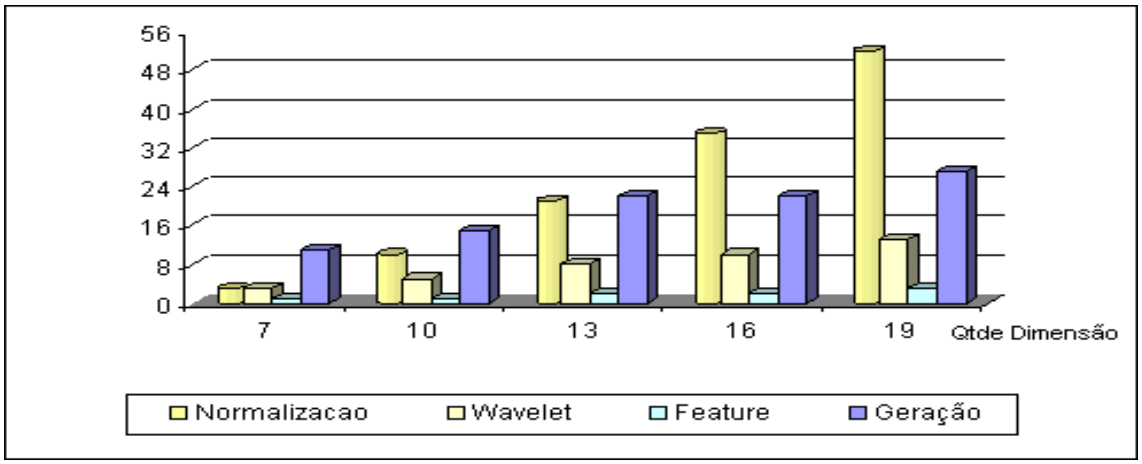

Figura 4.19: Tempo gasto na geração do índice por dimensão

Após a série temporal ter sido gerada e um índice ter sido criado, o SMT está preparado para o processo de consultas. Para tanto, será acionado o Módulo Gerador de Consultas apresentado na próxima sub-seção.

[15]Tempos calculados considerando que foram indexadas 500 séries para cada dimensão.

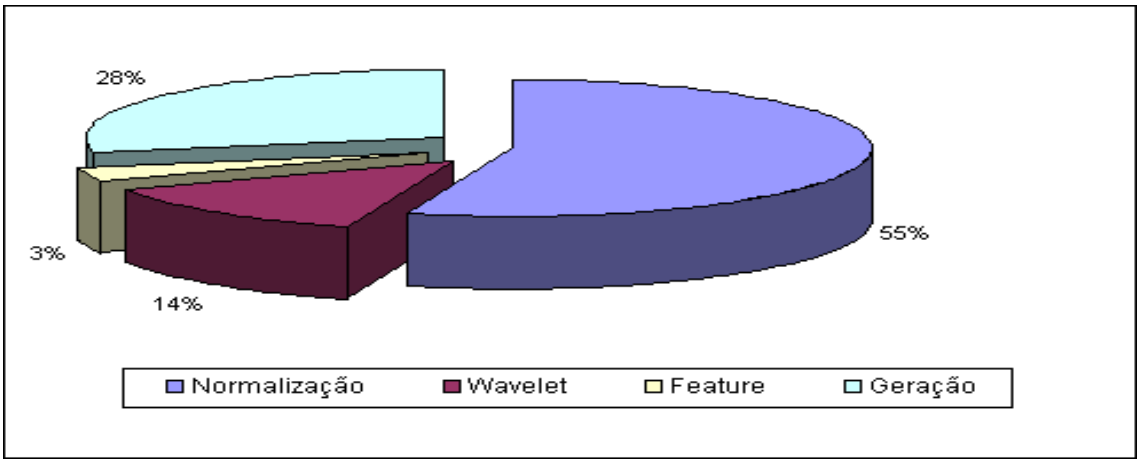

Figura 4.20: Percentual de tempo gasto na geração do índice por dimensão

4.3.2 Gerador de Consultas

Este módulo é o responsável por realizar as consultas nas séries temporais geradas pelo Módulo Extrator. Assim sendo, dado uma consulta como entrada, e um fator ϵ (distância euclidiana máxima permitida), o Gerador de Consultas pesquisará na base de dados e encontrará todas as séries que satisfaçam as restrições estabelecidas na consulta. Nesta fase, todas as séries já estão devidamente indexadas pelo processo descrito na seção 4.3.1. Como exemplos de consultas podem-se citar: "*Quais clientes têm seu PR defasado de seu PD de um fator* $> \epsilon$?", ou, sendo S_1 a série que representa um PF, "*Quais clientes têm algum intervalo de seu PR próximo de* S_1 *de um fator* $< \epsilon$?"

O módulo de geração de consultas é muito suscetível a fragilidades de desempenho, pois, normalmente, realizará consultas em base de dados contendo grandes volumes de informações. Conforme já dito na seção 4.1, um ponto importante a ser considerado é a definição do número de dimensões a serem utilizados durante o processo de criação dos índices. Esta decisão influenciará diretamente no desempenho das consultas e no tamanho do arquivo de índices. Um outro ponto não menos importante é a decisão de qual função de *Feature Extraction* será adotada. Esta escolha, influirá diretamente no desempenho das consultas[16].

O processo de realização de consultas em séries temporais não é um processo simples, ou seja, várias etapas devem ser seguidas para que se tenha um bom desempenho. Conforme apresentado no capítulo 2, basicamente, dois tipos de consultas são realizados em séries temporais: Consulta por faixa (*Range Query*) e Consulta dos k-vizinhos mais próximos (*k-Nearest Neighbor Query*). A estratégia utilizada para o primeiro tipo de consulta segue os seguintes passos [9, 24], considerando que $\mathbb{A}$ representa o conjunto de todas as séries da base de dados que representam os PR's dos clientes:

1. Projeta-se a série S_q da consulta q no mesmo espaço dimensional do índice;

[16]Estes dois pontos citados, dependem do tipo de aplicação.

2. Utilizando-se o índice, recupera-se em $\mathbb{A}$ todas as séries com distância $\leq \epsilon$ de S_q. Seja $\mathbb{B}$ o subconjunto de $\mathbb{A}$ que contém tais séries;

3. Um pós processamento é aplicado nas séries do conjunto $\mathbb{B}$, visando a eliminação de *Falsos-Positivos*.

O passo 1, busca as séries candidatas à solução da consulta. Para isso, os dados são recuperados através da utilização dos índices espaciais. O pós-processamento citado no passo 3 é necessário porque a redução da dimensão do dado garante apenas que, no caso de consulta por faixa, *Falsos-Negativos* não irão ocorrer [24][17], isto é, que dados significativos não serão excluídos, o que impede que séries candidatas que deverão fazer parte da solução não sejam eliminadas. Contudo, a redução não impede que, no conjunto solução $\mathbb{B}$, apareçam séries candidatas que não deveriam fazer parte da solução do problema, ou seja, podem ocorrer *Falsos-Positivos*. O pós-processamento consiste em se realizar o cálculo da distância euclidiana entre S_q, considerando todos os pontos da série (dimensão total), e as séries candidatas contidas em $\mathbb{B}$ (também, dimensão total). Após este cálculo, as séries candidatas cuja distância euclidiana foi maior do que o fator ϵ estabelecido, são eliminadas da resposta.

O segundo tipo de consulta, chamada k-vizinhos mais próximos, tem uma estratégia mais complexa, e é mais lento do que a consulta por faixa. Os passos a seguir mostram a estratégia utilizada[18], considerando que $\mathbb{A}$ representa o conjunto de todas as séries da base de dados.

1. Projeta-se a série S_q da consulta q no mesmo espaço dimensional do índice;

2. Todas as k-séries candidatas mais próximas de S_q são recuperadas utilizando o índice; Seja $\mathbb{B}$ o subconjunto de $\mathbb{A}$ que contém tais séries;

3. Calcula-se a distância entre as séries candidatas contidas em $\mathbb{B}$ e S_q, utilizando para tanto, todas as dimensões das séries. Considere a maior distância (max);

[17] Garante apenas se for consulta por faixa (*Range Query*).
[18] O algoritmo é baseado no GEMINI [24].

4. Realizar a consulta por faixa, considerando o fator $\epsilon = max$;

5. Calcule novamente as distâncias e as k-séries mais próximas são obtidas;

Como a redução de dados, no caso de consulta pelos k-vizinhos mais próximos, não garante a exclusão de *Falsos-Negativos* nem de *Falsos-Positivos*, torna-se necessária a execução dos passos 4 e 5.

Para que o Módulo Gerador de Consultas possa utilizar os índices criados, a consulta de entrada deve passar por um processo cujo objetivo é deixá-la com as mesmas características das séries que foram indexadas. A primeira delas é considerar a janela de pesquisa como sendo 1 semana, conforme utilizado na criação do índice[19]. Outra característica importante é a normalização, que, assim como foi utilizada no processo de indexação, deverá ser utilizada, também, no processo de consulta (usada para evitar distorções em relação ao deslocamento no eixo y, isto é, *shifting*).

Resumindo, as atividades necessárias para a realização de consultas, utilizando índices, são mostradas no algoritmo seguir:

1. As séries S_q das consultas devem ter o mesmo número de pontos da janela pré-definida, ou seja 1 semana[20];

2. As séries S_q devem ser normalizadas assim como feito no processo de indexação;

3. Deve-se aplicar a redução da dimensão S_q, usando a mesma técnica utilizada na criação do índice, ou seja, *Haar Wavelets*;

4. Aplicar a função de *Feature Extraction* nos coeficientes obtidos através do passo anterior;

5. Realizar a pesquisa no índice, utilizando os coeficientes obtidos pela função de *Feature Extraction*;

[19]Esta premissa pode ser alterada dependendo da aplicação.

[20]Uma implementação mais robusta poderia ter janelas de tamanhos variáveis, mas isso causaria impactos no processo de criação de índices em relação ao desempenho.

6. Realizar o pós processamento, ou seja, retirar os *Falsos-Positivos*, e retornar as séries corretas;

O algoritmo anterior refere-se à consulta por faixa, sendo necessárias poucas alterações para que se consiga realizar a consulta dos k-vizinhos mais próximos.
O Módulo gerador de consultas gera, como solução, um conjunto **S**. Cada elemento de **S** é formado por uma tupla (α, β, γ), onde α refere-se ao PR, β ao PD ou PF, e γ refere-se à distância euclidiana entre α e β.

Devido à relativa complexidade do módulo gerador de consultas, seu funcionamento será detalhado mediante exemplos ilustrados através de gráficos. Para isso, considerou-se uma série temporal (PR) S_1 representando o telefone "3432115100",onde, o primeiro ponto válido é "01/01/2005 00:00"e o último ponto é "22/02/2005 23:00", tendo um tamanho S_t de 1272 pontos. A consulta (*query*) a ser solucionada é: "*Quais os 3 períodos em que o telefone* 3432115100 *teve seu PR mais similar ao PD?*". Para resolver esta *query*, além do PR, S_1, também foi definido o PD, S_q.
A figura 4.21 ilustra uma janela de 7 dias da série S_1. A série S_1 está devidamente

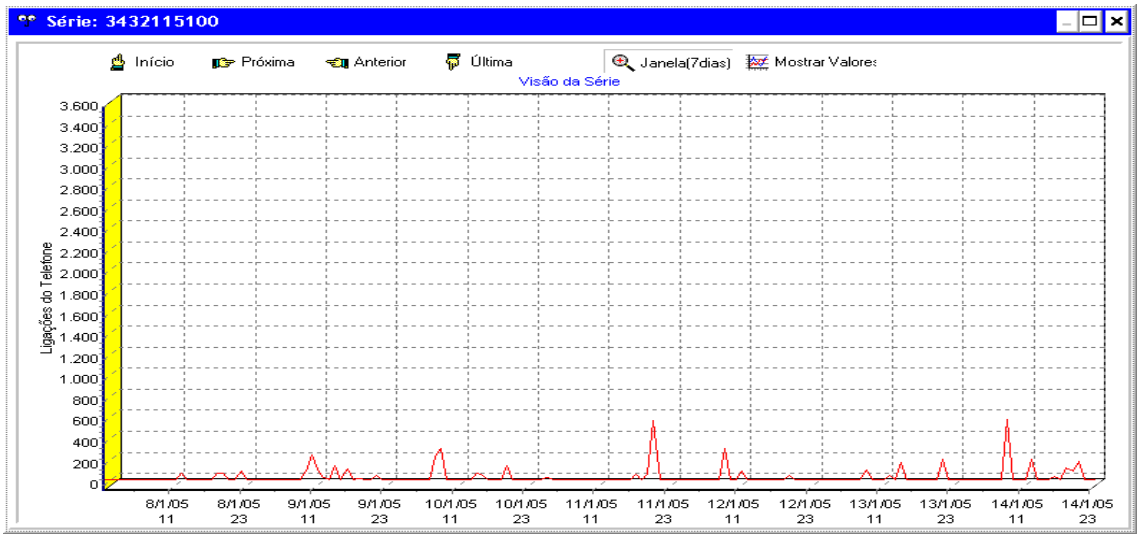

Figura 4.21: Janela de 7 dias da Série S_1 vista a partir do ponto '08/01/2005 00:00' e finalizando no ponto '14/01/2005 23:00'. A série já está indexada.

indexada, e considerou-se o tamanho da janela j de 168 pontos, o que corresponde a

1 semana. Logo, percebe-se que foram indexadas 7 subseqüências, pois, utilizando a equação 4.3.2, têm-se Mod ($\frac{1272}{168} = 7$). Todas as subseqüências foram normalizadas utilizando a equação 2.2.9.

Para a série PD da *query*, isto é, S_q, considerou-se o primeiro ponto válido em "01/01/2005 00:00"e o último ponto em "07/01/2005 23:00", tendo um tamanho lQ de 168 pontos (mesmo tamanho da janela j). S_q pode ser vista na figura 4.22. Uma vez definida S_q, o próximo passo (passo 2 do algoritmo) é realizar a normalização

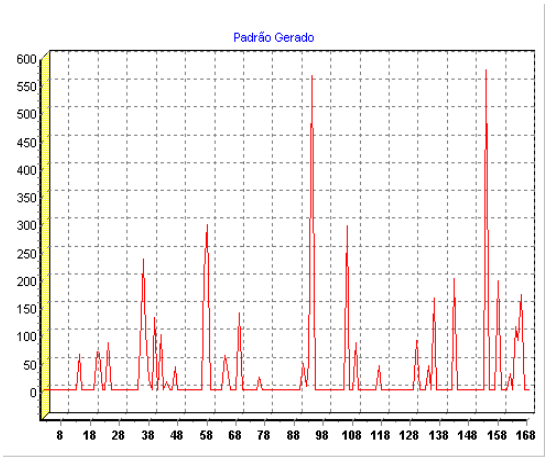

Figura 4.22: PD S_q

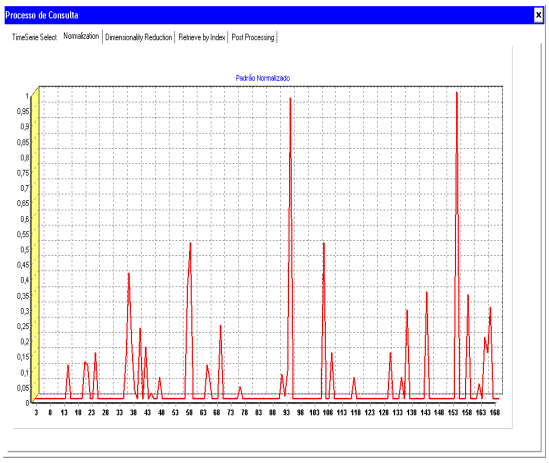

Figura 4.23: S_q normalizada

de S_q. Para isso, a equação 2.2.9 foi aplicada a cada ponto de S_q. S_q normalizada pode ser vista na figura 4.23.

Os próximos passos a serem realizados são a redução da dimensão do dado e a pesquisa no índice. Neste passo, a mesma técnica de redução dos dados que foi utilizada na criação do índice, é aplicada na série S_q, no caso, as *Haar Wavelets*. Após esta etapa, S_q está pronta para ser utilizada. Assim sendo, a pesquisa no índice é feita e as séries candidatas são recuperadas. Como *Falsos-Positivos* podem ocorrer, um pós processamento é necessário. A figura 4.24 ilustra o resultado após este processo ter sido realizado. Nesta figura, é mostrada a série S_q que se refere ao PD e, na parte inferior, as séries PR's candidatas. Foi realizada a consulta dos k-vizinhos mais próximos, com $k = 3$. Ainda nesta figura, pode-se ver as distâncias reais calculadas

entre o PD e as séries candidatas.

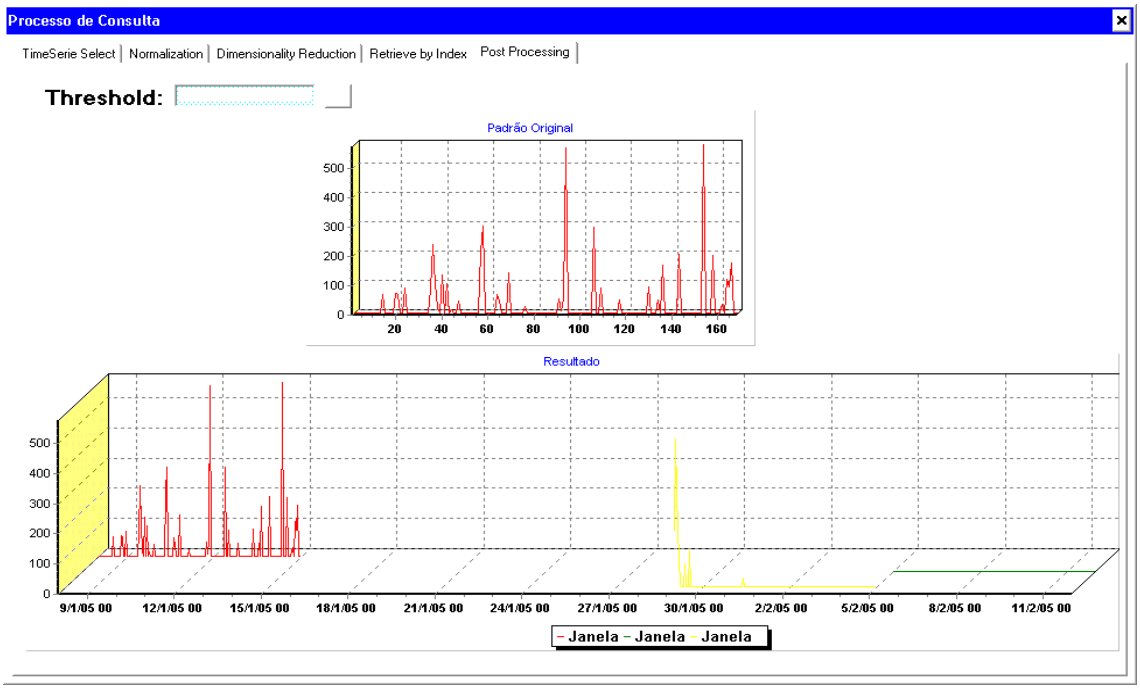

Figura 4.24: A consulta na base de dados, retornou 3 subseqüências mais próximas. A primeira subseqüência, que está em destaque, tem distância euclidiana igual a 0 em relação ao PD

A fim de verificar a eficiência do processo, foram realizadas algumas experiências. A primeira delas analisa a consulta utilizando diferentes dimensões. Outra experiência foi trocar a função de *Feature Extraction*. Esperava-se que, quanto maior fosse a dimensão, maior seria a seletividade das séries candidatas e, conseqüentemente, o desempenho da consulta seria melhor, pois, haveria um pós processamento menor. Esperava-se, também, que, à medida em que o número de dimensões fosse incrementado, o tamanho do arquivo de índices aumentaria e, conseqüentemente, o processo de busca no índice perderia em desempenho [24]. Os gráficos ilustrados nas figuras 4.25 e 4.26 mostram os resultados obtidos[21]. A experiência confirmou parte do que se esperava destas análises. Notou-se que a seletividade não depende apenas do número de dimensões, mas, também, da função de *Feature Extraction* empregada. Como as

[21]Foi realizada uma mesma consulta para as diferentes dimensões. A base de dados tinha 500 séries indexadas. Foram utilizadas duas funções de Feature Extraction: os primeiros coeficientes, e os coeficientes mais significativos.

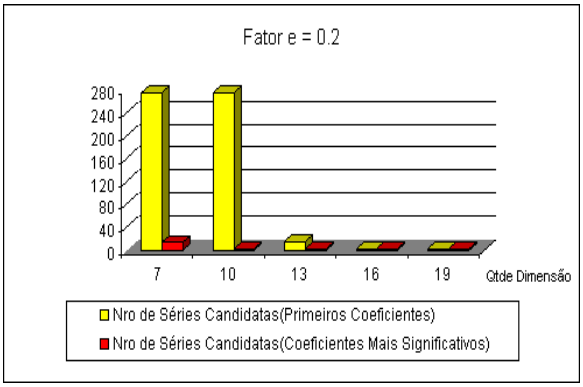

Figura 4.25: Seletividade por Dimensão e diferentes *Feature Extraction*

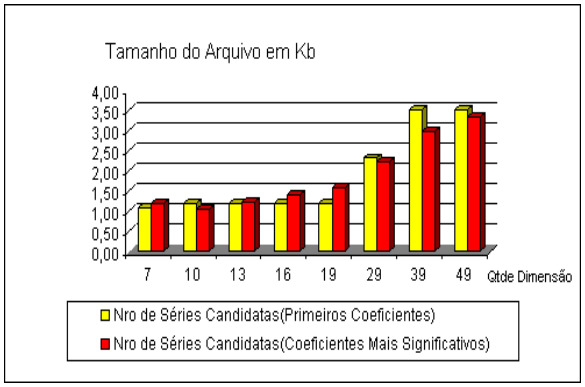

Figura 4.26: Tamanho do arquivo por Dimensão e diferentes *Feature Extraction*

séries utilizadas nas análises têm muitos pontos com o valor 0 (representando intervalos em que não houve ligações telefônicas), foi freqüente a ocorrência de séries com poucos coeficientes com valor maior do que 0. Isso significa que, dependendo da série, apesar do aumento no número de dimensão, pode-se ter a mesma seletividade. Por exemplo, pode-se ter uma série que tenha os seguintes coeficientes *wavelets*: (-0.1284, -0.1211, 0.0397, 0 ,-0.1713, 0.0635, -0.0765, 0, 0, 0), cuja dimensão seja 10. Observe que, neste caso, os últimos coeficientes são 0. Isso significa que, se a dimensão for 8, 9 ou 10, a eficiência da busca no índice será a mesma. O tamanho do arquivo de índices tende a aumentar conforme aumenta a dimensão, porém, dependerá do tipo de dados. Isso pode ser observado na figura 4.26, onde o tamanho do arquivo não aumentou constantemente conforme a dimensão aumentava. Note-se que, apenas nas dimensões maiores do que 16 ocorreu este fato.

Uma outra importante análise realizada foi em relação ao desempenho das consultas, considerando diferentes dimensões e diferentes *Feature Extractions*. A figura 4.27 mostra um gráfico com o resultado. Nele observa-se que, quanto menor for a dimensão, dependendo da função de *Feature Extraction*, menor será a seletividade. Assim sendo, a busca no índice se torna lenta e o pós-processamento fará com que o processo fique ainda mais lento. Após estas experiências, observa-se que a melhor dimensão a ser utilizada neste tipo de consulta é a dimensão 13 e que é conveniente

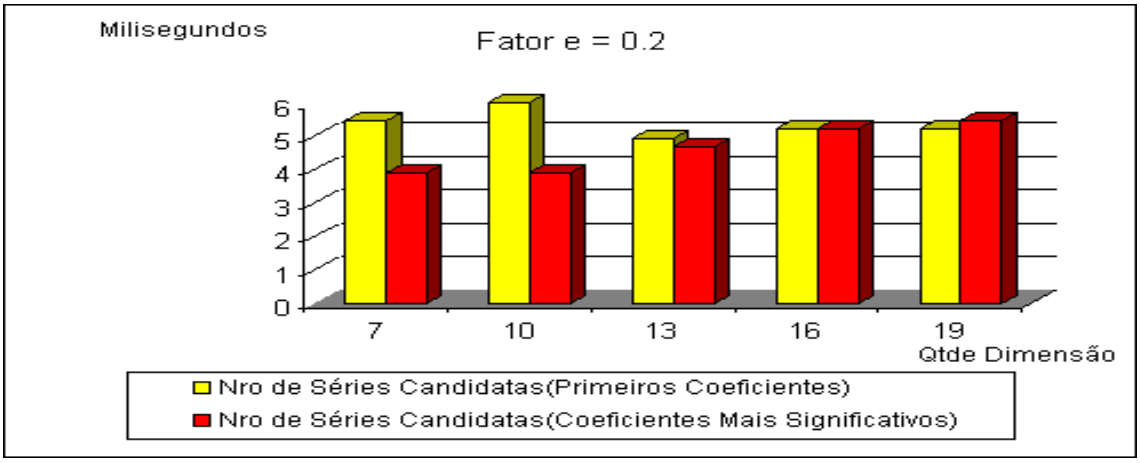

Figura 4.27: Tempo gasto por Dimensão

o uso da função de *Feature Extraction* considerando os coeficientes mais significativos. Observa-se claramente que, quanto maior a dimensão utilizada, menor será o número de séries candidatas recuperadas pelo índice. No entanto, quanto maior for a dimensão, maior será o tamanho do índice, e isso prejudica o desempenho das consultas.

Nos experimentos realizados, quando utilizou-se a dimensão 13, um melhor custo benefício foi alcançado, ou seja, o tamanho do índice não prejudicou o desempenho das pesquisas, e também o número de séries candidatas recuperadas pelo índice foi pequeno.

4.3.3 O Módulo de Conhecimento

As constantes mudanças que ocorrem no meio comercial, como regras de negócios, processos e rotatividade de pessoal, normalmente geram uma grande demanda de pessoal para a realização de manutenção nos sistemas envolvidos. Mas mesmo equipes numerosas e competentes têm dificuldades em acompanhar essas mudanças na velocidade necessária. Uma única regra de negócio pode demandar horas de desenvolvimento para que seja implementada. Em [20] é relatado que a maioria dos usuários corporativos se surpreendem ao descobrir de que maneira muitas regras de negócios estão espalhadas pela organização. Muitas dessas regras estão embutidas em códigos de programas ou estão apenas na cabeça das pessoas. Com as informações espalhadas, torna-se difícil a tarefa de se efetuar as análises, podendo ocorrer erros graves. A análise correta destas regras é fundamental para as empresas. Pensando nisso, algumas empresas estão procurando juntar essas regras em um único repositório onde possam ser facilmente identificadas e analisadas.

Como alternativa de solução, o SMT utiliza um Módulo de Conhecimento (MC) baseado na técnica RDR. Com o MC espera-se diminuir o tempo gasto com desenvolvimento e manutenção de sistemas. O próprio especialista de negócio poderá implementar suas regras, em muitos casos, dispensando o engenheiro de conhecimento[22]. Outra vantagem, é que, através do MC, pode-se, a qualquer momento e facilmente, listar todas as regras cadastradas, visualizar aquelas que estão sendo utilizadas e as conclusões que estão sendo propostas pelo sistema.

O RDR baseia-se no seguinte princípio: *"O conhecimento que o especialista provê é, essencialmente, a justificativa do porquê ele está certo, não o raciocínio que ele teve de fazer para alcançar a conclusão correta"* [35]. Este princípio foi observado em decorrência do fato de o conhecimento obtido de especialistas e incorporado em sistemas baseados em conhecimentos ser muito trabalhoso e difícil. Foi observado que parte desta dificuldade era porque o especialista raramente informava como era o

[22] O sucesso de tal objetivo dependerá da implementação do RDR.

raciocínio que ele fazia para alcançar a solução, mas, em contrapartida, ele justificava o porquê a solução estava correta.

Neste caso, a justificativa é baseada em *features* que são identificadas pelo caso atual. O RDR é estruturado como uma árvore, conforme detalhado no capítulo 2. No RDR, o especialista não é envolvido na estruturação interna do conhecimento nem na verificação de consistência de regras. O especialista apenas provê o conhecimento.

A estrutura RDR proporciona organização das regras, uma vez que se pode dividi-las por assuntos. Cada assunto poderá ter seu próprio RDR ou pertencerá a um RDR maior (como se fossem classes de RDR). A estrutura do Sistema Baseado em Conhecimento proposta é composta de dois módulos: RDR e a Base de Conhecimento (BC). A BC armazena as regras e o RDR é o responsável pelas consultas e atualizações na BC.

A utilização do RDR foi motivada pela participação do autor desta dissertação em um projeto realizado em uma grande empresa de telecomunicações. No referido projeto, foi construído um módulo semelhante à estrutura RDR o qual processa mais de 100.000 análises por mês. Contudo, a técnica utilizada, apesar de eficiente, disponibiliza menos recursos que o RDR. Por exemplo, não permite a correção de regras sem a análise de todas as regras existentes.

O formato das regras construídas pelo RDR segue uma gramática particular, conforme apresentado a seguir [46].

Atributos e *Features*

Os atributos representam características das classes (conceitos), enquanto que as *Features* são representadas por funções que mapeiam atributos em abstrações mais altas. Por exemplo, na regra $\mathcal{C} = 1 \leftarrow \mathcal{A}' \wedge \mathcal{B}'$, $\mathcal{A}'$ *e* $\mathcal{B}'$ são *Features*. Já no exemplo $\mathcal{A}' = A > 5$ e $\mathcal{B}' = MaiorQue(B, 5)$, A *e* B são atributos. Um atributo é característica de uma classe ou conceito. Pode haver quatro tipos de atributos: nominal, boleano, ordinal e numérico. Um atributo nominal tem uma possibilidade finita de

valores, por exemplo, um atributo *cor* pode ter 4 valores possíveis, (*verde, amarelo, azul, branco*). Um atributo ordinal também tem valores finitos, porém, existe uma ordem entre eles, por exemplo, o atributo *temperatura* pode assumir 3 valores, (*frio, temperado, quente*), onde *frio* $<$ *temperado* $<$ *quente*. Um atributo numérico pode assumir valores contínuos ou discretos. Quando um atributo pode assumir valores infinitos, normalmente, utiliza-se uma faixa de valores. Por exemplo, um atributo numérico que represente a altura de um homem deve ter uma faixa de valores que varia entre 50cm e 250cm. O valor de um atributo qualquer é representado por um par *atributo-valor*, por exemplo, *temperatura=frio*. Uma *Feature* é retornada por uma função, a qual pode ter atributos, constantes e *Features* como parâmetros, por exemplo, *MenorQue(y, Media(x1, x2, x3)), MenorQue(x,5) e (x $<$ 5)*. Um atributo tem pelo menos um valor possível. No RDR, o especialista constrói regras combinando *Features*. Estas *Features* são construídas pelo engenheiro de conhecimento por meio dos atributos. Um caso consiste de uma seqüências de valores de atributos e os atributos são extraídos dos casos.

Considere A como sendo um conjunto consistindo dos atributos de um determinado caso. Seja E o conjunto de todas as *Features construídas* pelo engenheiro de conhecimento. O engenheiro de conhecimento cria as *Features* E utilizando os atributos. Um exemplo de uma *Feature* e_1, onde $e_1 \in E$, é *Media*(a_1, a_2), onde a_1 e $a_2 \in A$. O engenheiro de conhecimento cria estas funções após entrevista com o especialista ou através de outros métodos. Estas *Features* devem ser de fácil entendimento, por exemplo, a *Feature distancia(x1,y1,x2,y2)* calcula a distância entre dois pontos, x e y com coordenadas x1,y1 e x2,y2, respectivamente.

As *Features* do tipo *boleano*, ou seja, funções particulares do tipo *boleano* podem ser usadas nas condicionais das regras.

Regras

Uma regra no RDR, consiste de um conseqüente e de um antecedente, onde o antecedente é uma conjunção de *Features* do tipo *boleano*. Pode ser assim representada: *Regra* $\Rightarrow$ *Conseqüente* $\leftarrow$ *Antecedente*.

O RDR Proposto

O RDR proposto será responsável pela análise das informações referentes à similaridade entre as séries temporais. Para isso, as regras deverão ser criadas. *As regras* serão definidas pelo Especialista do domínio. É através destas regras que o mecanismo de inferência conseguirá determinar as ações a serem executadas. As regras serão representadas conforme gramática citada na seção 4.3.3.

A seguir é apresentado um exemplo de RDR cujas regras são enfocadas no domínio de anomalias referentes à ligações telefônicas. As regras estão representadas em linguagem natural para uma melhor visualização, mas serão reescritas no momento oportuno considerando a gramática usada. O objetivo principal é analisar as séries temporais considerando-se o cálculo da distância euclidiana entre elas. Ou seja, a estrutura RDR responderá qual ação será tomada a partir de resultados obtidos na análise de similaridade que visa a detectar anomalias. Como nas seções 4.3.1 e 4.3.2, serão considerados anomalias referentes à fraude, e relacionadas ao perfil de uso do cliente. Para isso, as regras foram divididas em duas categorias: *Fraude e Cliente*, desta forma, as regras ficaram dividas por assunto (contexto). As regras da categoria *Fraude* irão se referir à anomalias ligadas às fraudes, ao passo que as regras da categoria *Cliente* irão se referir a anomalias ligadas ao perfil de uso dos telefones[23].

```
If Categoria = 'FRAUDE' Then
   ..........
Else
    If Categoria = 'CLIENTE' Then
          ..........
```

[23]Os números de telefone usados nas regras abaixo são fictícios.

Dentro da categoria 'FRAUDE' teremos regras do tipo:

```
If( Similaridade(PF,PR,ClienteX) estiver entre 2 e 3)*
  Then
    (case1.01:Cliente Suspeito)
     Enviar Email para responsaveis
     Registra Ocorrencia
    Except
      If Cliente Isento de analise
        Then
          (Case1.02:Cliente Suspeito mas Isento)
Else
  If(Similaridade(PF,PR,ClienteY) < 1)*
     Then
       (Case1.03:Cliente Fraudatario)
       Enviar Email para responsaveis
       Bloqueia Telefone Imediatamente

Obs.: Os números que representam a similaridade,
      nas regras acima assinalados com (*),
      são fictícios.
```

Dentro da categoria 'CLIENTE' teremos regras que serão capazes de analisar o cliente conforme seu perfil previamente cadastrado:

```
If(Cliente for do Segmento Empresarial)
  Then
    (case3:Cliente OK)
    Except
      If Similaridade(PD,PR) > 1 e Similaridade(PD,PR) < 4
        Then
          (Case3.01:Perfil difere)
          Avisar Cliente
      Else
         If Similaridade(PD,PR) >= 5
           Then
             (Case3.02:Perfil difere totalmente)
             Avisar Cliente
             Bloquear Cliente Para Nao Originar
           Except
              If Cliente nao Quer Aviso
                Then
                  (Case3.03:Perfil difere
                  totalmente,sem aviso)
```

```
                Bloquear Cliente
Else
  If(Cliente for do Segmento Residencial)
     Then
       (Case4:Cliente Ok)
     Except
       If Similaridade(PD,PR) > 3
         Then
             (case4.01:Perfil difere)
             Bloqueia Cliente
         Except
             If Cliente for Vip Then
                (case4.02:Perfil Ok)
```

As regras descritas anteriormente podem ser visualizadas na forma gráfica através da figura 4.28. Considerando o diagrama de regras apresentado na figura 4.28, pode-se

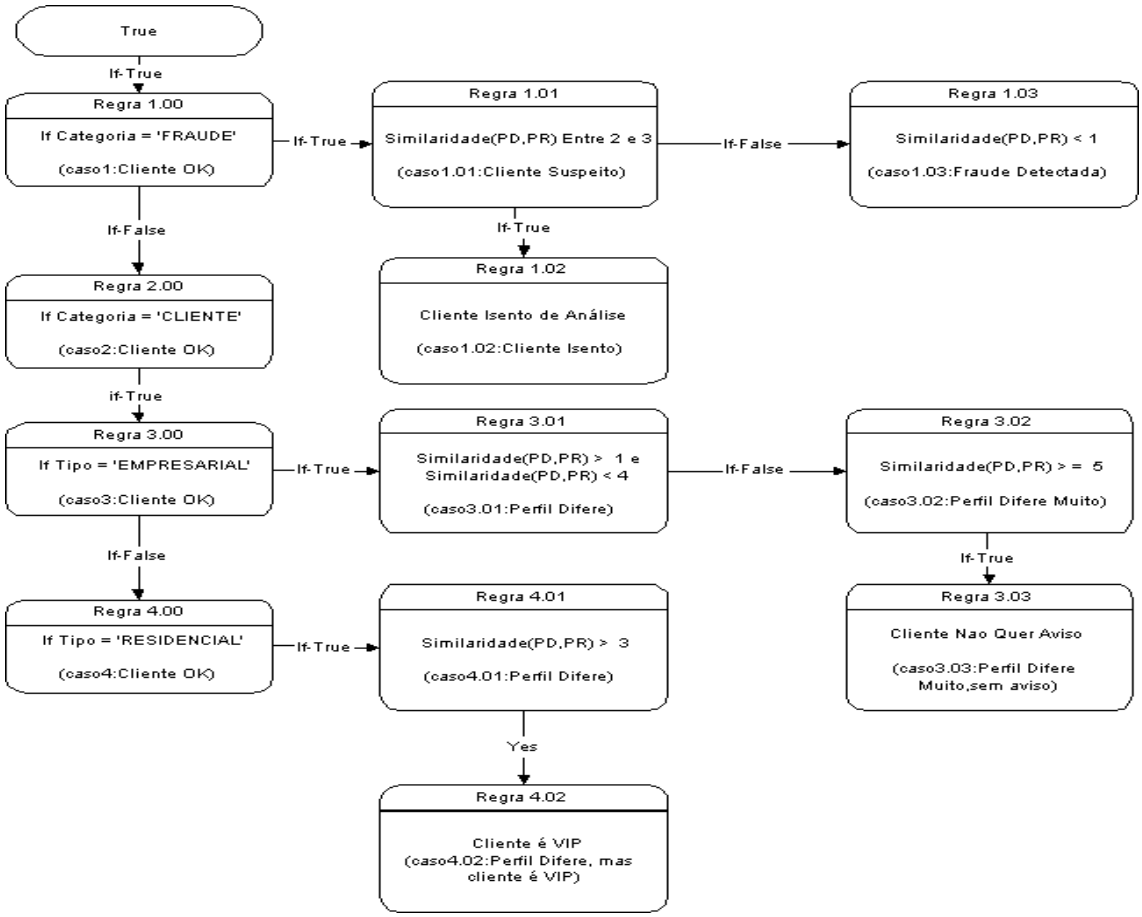

Figura 4.28: Exemplo de Regras na Estrutura RDR

destacar alguns atributos e *Features*. São exemplos de atributos: *Categoria* é um atributo nominal e pode assumir os valores (FRAUDE, CLIENTE),*Tipo de Cliente* é um atributo nominal e pode assumir os valores (VIP, Normal, Empresarial, Residencial). Como exemplo de *Feature*, pode-se destacar a função *similaridade(PD,PR)*, onde PD e PR são atributos numéricos, e assumem o valor correspondente aos códigos das séries do PD e do PR. Neste caso, a função *Similaridade* é uma *Feature* que realiza o cálculo da distância euclidiana considerando os parâmetros de entrada. Como mostrado, o RDR permite que as regras sejam vistas de maneira clara e que sejam analisadas corretamente, facilitando a tarefa de rastreamento da análise feita. Além disso, a estratégia "case based"do RDR permite a melhoria do desempenho do sistema na fase de recuperação da informação. Tal ganho se deve à otimização obtida pela redução na quantidade de unificações entre metas e condicionais das regras efetuadas a cada vez que uma consulta é proposta. Esta redução se deve ao fato de que uma mesma condição é testada uma única vez em qualquer ramo da árvore, mesmo que mais de uma regra deste ramo a tenha como integrante de sua parte condicional (efeito semelhante é obtido pelo algoritmo RETE, conforme apresentado no final da seção 2.5.4). A estrutura RDR proposta pode ser estendida para que seja utilizada em diferentes domínios. Por exemplo, podem-se criar regras que não tenham nenhum relacionamento com anomalias, mas que sejam regras úteis à parametrização do sistema. A título de exemplo, uma categoria nova chamada 'EXTRAÇÃO' (regras referentes ao Módulo Extrator) seria útil em uma implementação do Sistema Modular, onde diferentes técnicas fossem utilizadas. Neste caso, o RDR poderia auxiliar o sistema a decidir quais parâmetros seriam utilizados. Como:

```
If(TipoExtracao = 'SERIES TELEFONE')
  Then
    (case1:Wavelet)
     Reducao = Wavelet
     Tamanho Janela = 6
     Granularidade  = 1
     Utiliza Transformacoes = false
   Except
```

```
      If DataAtual > 01/01/2004
        Then
          (Case1.1:Wavelet Corrigido)
          Reducao = Wavelet
          Tamanho Janela = 6
          Granularidade  = 1
          Utiliza Transformacoes = true
Else
  If(TipoExtracao = 'SERIES PRODUTOS')
     Then
       (Case3:PAA)
       Reducao = PAA
       Tamanho Janela = 6
       Granularidade  = 1
       Utiliza Transformacoes = false
```

4.3.4 Fluxo de Trabalho do SMT

A título de esclarecimento, a figura 4.29 mostra um diagrama que representa o fluxo do SMT proposto. Nela, é ilustrado em que momento os módulos interagem e também em que momento o módulo de conhecimento é acionado. O módulo gerador de séries temporais e o módulo que cria os índices são independentes deste fluxo de trabalho, ou seja, quando se aciona o processo para responder a uma consulta, as séries envolvidas já deverão ter sido criadas pelo Módulo Gerador de Séries, bem como os índices. Para entender melhor o funcionamento do sistema, vamos supor um banco de dados D_1 que armazena padrões de fraudes (PF) referentes à ligações telefônicas dos clientes de uma determinada companhia telefônica. Suponha, também, que exista uma estrutura RDR (R_1) conforme apresentada na figura 4.30. Se a companhia telefônica propuser a seguinte consulta: "Quais clientes possuem um perfil (PR) similar, considerando um fator ϵ, àqueles encontrados em D_1?", os passos de 1 a 7 apresentados na figura 4.29 serão realizados. Mas, se a Companhia Telefônica desejar efetuar a seguinte consulta: "Quais ações deverão ser tomadas, para aqueles clientes que possuem um perfil (PR) similar, considerando um fator ϵ àqueles encontrados em D_1?"os passos de 1 a 9 apresentados na figura 4.29 serão realizados.

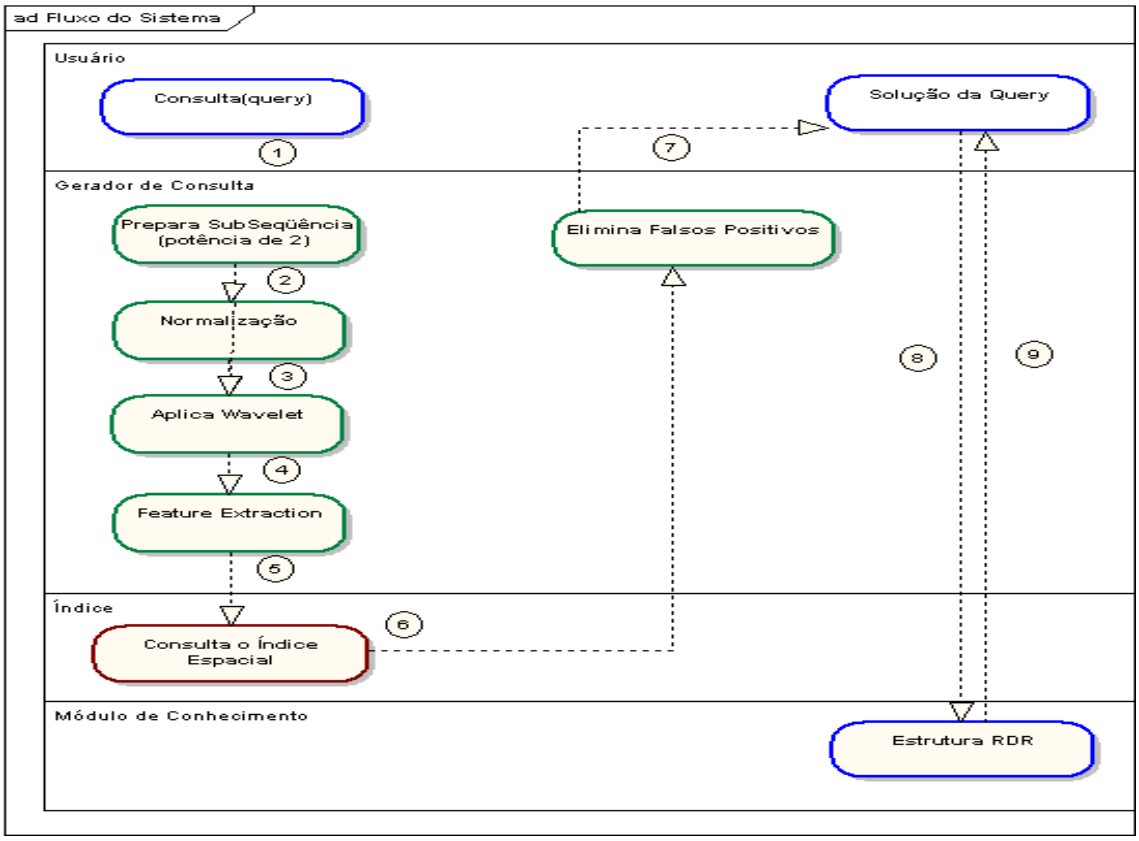

Figura 4.29: Fluxo de Trabalho do Sistema

Para ilustrar o fluxo de trabalho do SMT proposto, considere a execução da primeira das consultas acima, isto é: "Quais clientes possuem um perfil (PR) similar, considerando um fator ϵ, àqueles encontrados em D_1?".

Primeiramente, o usuário aciona o Módulo Gerador de Consultas. Este, por sua vez, buscará em D_1 todos os PF's e, para cada um deles, executará os passos de 1 a 7 apresentados na figura 4.29. Após a execução desses passos, o usuário terá uma lista com todos os telefones que são similares, pelo fator ϵ, das séries de D_1. O usuário, de posse das séries retornadas pelo gerador de consultas, acionará o módulo de conhecimento para que ele, por sua vez, execute as análises através da BC. O Módulo gerador de consultas gera como solução, um conjunto $\mathbf{S}$, onde cada elemento de $\mathbf{S}$ é formado por uma tupla (α, β, γ), onde α refere-se ao PR, β refere-se ao PD e γ refere-se à distância euclidiana entre α e β.

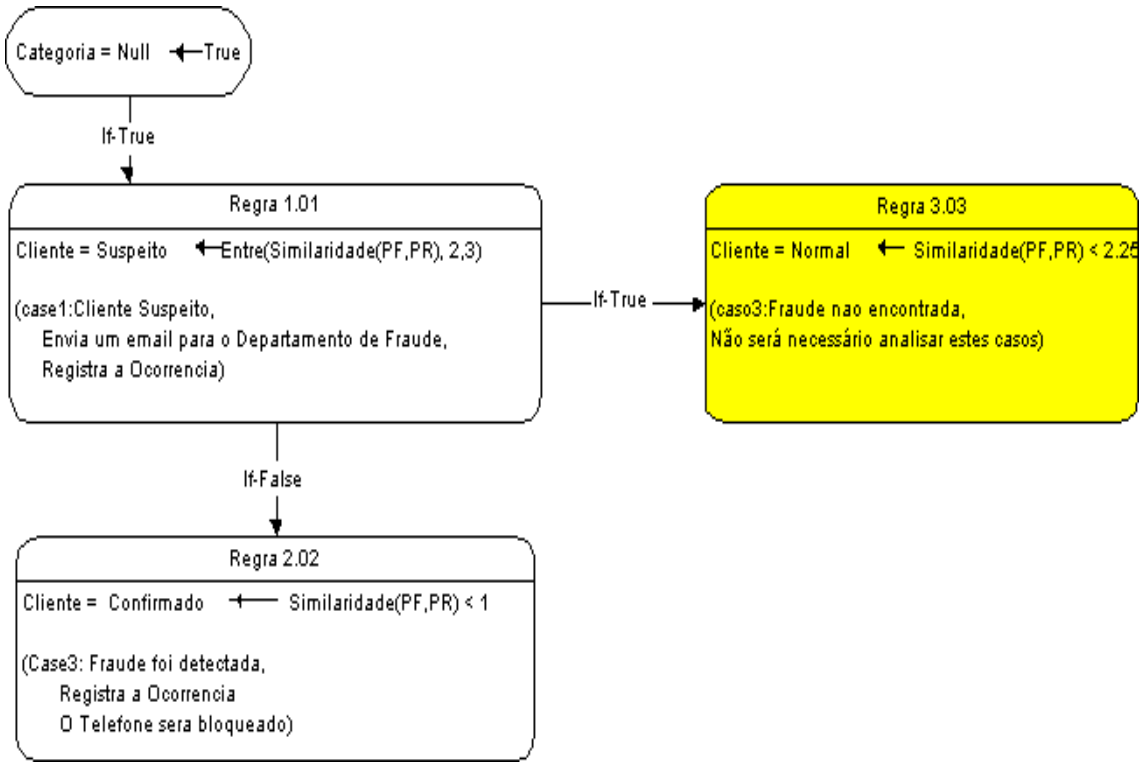

Figura 4.30: Estrutura RDR, R_1

O Módulo de Conhecimento acionará a estrutura RDR (R_1), aqui representada pela figura 4.30, informando o valor dos atributos definidos, no caso, (α, β, γ).
Considere, inicialmente, que a regra 3.03 não exista em R_1.

O RDR inicia sua análise pela regra *default*, como esta regra é sempre verdadeira, a próxima regra a ser analisada será a 1.01. Nesta regra, apenas uma condicional está presente, e é composta de apenas uma *Feature* do tipo *boleano*, chamada *Entre*, que recebe 3 parâmetros. O primeiro parâmetro é outra *Feature*, chamada *Similaridade* do tipo numérica, que é responsável por acionar o módulo de similaridade, onde ocorrem os cálculos da similaridade. O segundo e terceiro parâmetros são termos constantes. Se a *Feature Entre* retornar verdadeiro, a conclusão é apresentada ao usuário, no caso seria: "Cliente Suspeito". Se o usuário concordar com a solução, o processo termina. Caso contrário, se ele julgar que é preciso efetuar análises complementares, ele terá que corrigir a regra 1.01. Isto é alcançado inserindo uma nova regra, a 3.03

(em destaque na figura 4.30). Se a condicional da regra 1.01 for falsa, a regra 2.02 é analisada e o mesmo processo de análise é repetido.

Observe que os números *1*, *2*, *3* e *2.25* usados nestas regras, indicam um grau de similaridade arbitrário. Estes números deverão ser conseguidos através de experimentos, e podem variar com o contexto. Por exemplo, pode-se considerar que um telefone seja suspeito, se a distância euclidiana for X, caso este seja comercial, e Y caso seja residencial.

Capítulo 5

Conclusões e Resultados Obtidos

Nesta dissertação, foi proposto um sistema modular para telecomunicações (SMT) que deve ser capaz de representar conhecimento e dados relativos à telefonia, bem como detectar anomalias ligadas ao uso indevido de linhas telefônicas (Perfil de Fraude). Para tanto, o SMT representa os dados referentes às chamadas telefônicas através de séries temporais que são criadas por um módulo Gerador de Séries.

A análise visando a detecção de anomalias é efetuada através de pesquisa de similaridade entre as séries temporais que representam a utilização real das linhas telefônicas e as séries que representam, ou o padrão de uso ideal estabelecido pelos proprietários da linha, ou um perfil de fraude.

Como o SMT deverá efetuar consultas nessas séries, visando um bom desempenho, o Gerador de Séries, além de criar as séries na base de dados, também utiliza uma técnica de redução de dados que diminui a enorme quantidade de dados sem permitir que se percam informações relevantes. Isso é obtido pela aplicação das *Haar Wavelets* nas séries. Além disso, o Módulo Gerador de Séries, para obter uma melhor análise de similaridade (isso depende da aplicação, ou seja, é opcional), normaliza as séries antes de submetê-las ao processo de redução dos dados.

Ainda a título de melhoria no desempenho das pesquisas de similaridade, concluído o processo de redução dos dados, o SMT cria índices espaciais para as séries reduzidas.

Um outro módulo do SMT é o módulo Gerador de Consultas. Tal módulo tem como função executar as consultas propostas pelos usuários do SMT indagando sobre a normalidade ou não da utilização das linhas telefônicas (para responder a tais consultas, o módulo Gerador de Consultas aciona o módulo de similaridade).

Uma vez terminado o processo de análise de similaridade, o SMT aciona um último módulo: o módulo de Conhecimento, responsável por indicar as ações que deverão ser desencadeadas em função dos resultados das análises de similaridade. Tal módulo corresponde a uma combinação de um sistema de produção e de um sistema baseado em casos (*Case Based Reasoning*), sendo construído nos moldes da estrutura RDR (*Ripple Down Rules*).

O módulo do Conhecimento é composto por regras relacionadas ao universo da telefonia. Incluem-se aí, por exemplo, regras que detectam ocorrência de fraude e que prevêem ações a serem desencadeadas caso isto ocorra. Tais regras são construídas segundo o modelo da técnica RDR. A utilização da estrutura RDR foi motivada pela participação do autor em um projeto de uma grande empresa de telecomunicações, no qual foi desenvolvido uma estrutura semelhante ao RDR, porém, menos eficiente.

O SMT apresentado pode ser utilizado em qualquer domínio onde os dados possam ser representados em séries temporais, podendo ser facilmente parametrizado para trabalhar com diferentes medidas de similaridade, diferentes métodos de indexação e diferentes técnicas de redução da dimensão dos dados.

Para realizar as análises de similaridade nas séries temporais, é necessário que se façam várias consultas em banco de dados, pois, para calcular a distância euclidiana entre as séries, precisam-se de todos os pontos das mesmas. Normalmente, as séries temporais envolvem uma grande quantidade de pontos (dimensão), isto faz com que as consultas utilizando índices tradicionais como B-Tree sejam ineficientes. Uma tentativa de solução para esse problema é utilizar índices espaciais, como por exemplo o R-Tree. A idéia é bastante simples, mas muito interessante. As séries temporais são consideradas um ponto no espaço com n dimensões, podendo, portanto, ser indexadas

em um índice espacial. Entretanto, devido a alta dimensionalidade das séries, mesmo utilizando índices espaciais, as consultas não seriam eficientes, pois, estes índices tem seu ponto ótimo, quando a dimensão está entre 7 e 12 [24]. Em função disto, a utilização das *Haar Wavelets*, como ferramenta de redução de dados, mostrou-se eficiente, uma vez que elas são computacionalmente rápidas e seus coeficientes apresentam uma boa representatividade [9]. Além disso, através das *Haar Wavelets* pode-se conseguir uma análise em vários níveis de resolução. A análise multi-resolução permite descobrir padrões que aparentemente não estão presentes na série original, porém, estes padrões podem ser encontrados considerando níveis de diferentes resoluções. Para utilizar essas *wavelets*, as séries precisam ser preparadas, pois uma exigência, é que as séries tenham um número de pontos que seja potência de 2. Após o preparo das séries, aplicam-se as *Haar Wavelets* obtendo-se os coeficientes. A redução é conseguida utilizando uma função que seleciona os melhores coeficientes, chamada *Feature Extraction*. Uma vez realizada a *Feature Extraction*, o índice espacial pode ser criado.

A título de comparações, foram realizados alguns experimentos utilizando outra técnica de redução dos dados, a PAA. Esta técnica é mais rápida computacionalmente do que as *Haar Wavelets*, pois ela não exige que as séries sejam potência de 2, e nem precisam utilizar a *Feature Extraction*, uma vez que apenas os k coeficientes são calculados (onde k indica a dimensão escolhida). Entretanto, a diferença é pequena, pois nas *Haar Wavelets* o tempo gasto na *Feature Extraction* e no preparo das séries é muito pequeno, conforme apresentado na figura[1] 4.27. Outro fato importante é que a PAA não faz análise multi-resolução, enquanto que as *Haar Wavelets* permitem isso de uma maneira clara e intuitiva, pois os coeficientes *wavelets* já são calculados em vários níveis de resolução.

Uma outra observação importante feita foi em relação à dimensão dos dados, ou seja, a definição de qual número de pontos deve ser utilizado na redução dos dados, levando em consideração o desempenho. Na experiência realizada, o número ótimo

[1] Considerando o tipo de implementação que foi utilizado.

para ser utilizado foi 13. Porém, observou-se que este número pode variar conforme o tipo de série a ser analisada, ou seja, se a série apresentar uma grande quantidade de 0's, pode-se ter uma grande quantidade de coeficientes com valor zero, e isso pode influenciar no número de pontos que será considerado ideal na redução dos dados.

Com os experimentos realizados, as *Haar Wavelets* se mostraram eficientes em relação ao desempenho e também em relação à seletividade. Estes resultados foram ilustrados com gráficos no capítulo 4.

Para realizar as consultas, o Módulo de Consultas dispensa à série dada como entrada (Perfil Desejado ou Perfil de Fraude), o mesmo tratamento dispensado às séries reais no processo de criação dos índices, ou seja, preparação da série em potência de 2, normalização e redução dos dados (*Feature Extraction*). Após esta etapa, as séries são recuperadas utilizando o índice espacial gerado pelo Módulo de Geração de séries.

O Módulo de Conhecimento utiliza uma estrutura RDR para facilitar as análises. Com este módulo, o sistema tem suas regras facilmente identificadas e o próprio especialista é responsável por inserir novas regras, bem como por manter a base de regras atualizada.

Como resultado deste trabalho, publicou-se um artigo completo na conferência *IEEE ICT 2005* [29]. Também foi feito um protótipo do Módulo Gerador de Série e do Módulo de Consultas, que foi utilizado para que as observações e experimentos fossem possíveis. Apesar de o módulo de Conhecimento não ter sido implementado, implementou-se a estrutura RDR para fins de entendimentos de seu funcionamento.

Como trabalho futuro, pretende-se introduzir quantificadores temporais da lógica temporal para tratar aspectos temporais do domínio. Pretende-se, também, incrementar o poder de análise da estrutura, utilizando análises em vários níveis de coeficientes *wavelets*, o que extenderia significativamente a análise da similaridade. Pretende-se, também, estudar a automatização da rotina de inserção de regras na estrutura RDR.

Referências Bibliográficas

[1] Graps A., *An introduction to wavelets*, IEEE Computational Science and Engineering **2** (1995), no. 2, 51–61.

[2] Rakesh Agrawal, Christos Faloutsos, and Arun N. Swami, *Efficient similarity search in sequence databases*, FODO '93: Proceedings of the 4th International Conference on Foundations of Data Organization and Algorithms (London, UK), Springer-Verlag, 1993, pp. 69–84.

[3] G. Araribóia, *Inteligência artificial - um curso prático. livros técnicos e científicos*, 1988.

[4] Miguel A. Arino, *Time series forecasts via wavelets: An application to car sales in the spanish market*, Discussion Paper 95-30, ISDS, Duke University.

[5] B. Bartsch-Spörl and K.-D. Althoff, *Decision support for case-based applications*, Wirtschaftsinformatik 1/96, Special Issue on Case-Based Decision Support, edited by D. Ehrenberg (1996), 6–14.

[6] Brigitte Bartsch-Spörl, Klaus-Dieter Althoff, and Alexandre Meissonnier, *Learning from and reasoning about case-based reasoning systems*, XPS, 1997, pp. 115–128.

[7] Adriano Ferreti Borgatto, *Análise de intervenção em séries temporais : Aplicações em transporte urbano*, Dissertação de Mestrado, Universidade Federal de Lavras, Agosto 2000.

[8] LM Brasil, de Azevedo FM, and Barreto JM, *A hybrid expert system for the diagnosis of epileptic crisis*, Artificial Intelligence in Medicine **585** (2000), 1–7.

[9] K. Chan and A.W.-C. Fu, *Efficient time series matching by wavelets*, In Proc. of the ICDE Conf. (Sydney, Austrália), 1999, pp. 126–133.

[10] A. Chortaras, *Efficient storage, retrieval and indexing of time series data*, Dissertação de Mestrado, Imperial College of Science, Technology and Medicine (University of London), Department of Computing, Setembro 2002.

[11] Veronica Clark, *To maintain an alarm correlator*, Dissertação de Mestrado, The University of New South Wales, School of Electrical Engineering And Telecommunications, Novembro 2000, http://www.hermes.net.au/pvb/thesis/index.html.

[12] Clarimar José Coelho, *Identificação de imagens com análise Multiresolução Wavelet em sistemas baseados em casos*, Dissertação de Mestrado, Universidade de Brasília, Fevereiro 1999.

[13] P. Compton, G. Edwards, B. Kang, L. Lazarus, R. Malor, T. Menzies, P. Preston, A. Srinivasan, and C. Sammut, *Ripple down rules: possibilities and limitations.*, 6th Bannf AAAI Knowledge Acquisition for Knowledge Based Systems Workshop, Banff (1991), 6.1–6.18.

[14] P. Compton, P. Preston, G. Edwards, and B. Kang, *Knowledge based systems that have some idea of their limits*, nov 1996, http://citeseer.ist.psu.edu/compton96knowledge.html.

[15] Paul Compton, Byeong Kang, Phillip Preston, and Mary Mulholland, *Knowledge acquisition without analysis*, Proceedings of the 7th European Workshop on Knowledge Acquisition for Knowledge-Based Systems (London, UK), Springer-Verlag, 1993, pp. 277–299.

[16] C. Faloutsos, M. Ranganathan, and Y. Manolopoulos, *Fast subsequence matching in time-series databases*, Proceedings of the ACM SIGMOD International Conference on Management of Data (Minneapolis, USA), 1994, pp. 419–429.

[17] Roberto Santos Filho, Elaine P. M. de Sousa, Agma Traina, and Caetano Traina Jr., *Desmistificando o conceito de consultas por similaridade: A busca de novas aplicacões na medicina*, Anais do segundo Workshop de Informática Médica, 4p. Simpósio Brasileiro de Engenharia de Software (SBES) da Sociedade Brasileira de Computação (2002), CD–ROM.

[18] Brian R. Gaines and Paul Compton, *Induction of ripple-down rules applied to modeling large databases*, Journal of Intelligent Information Systems **5** (1995), no. 3, 211–228, citeseer.ist.psu.edu/gaines95induction.html.

[19] D. Harmon, P. e King, *Sistemas especialistas - a inteligência artificial chega ao mercado*, Editora Campos, 1988.

[20] Sue Hildreth, *Passando a companhia a limpo*, ComputerWorld **435** (2005), 12–13.

[21] A. Hoffman, R. Kwok, and P. Compton, *Simulations for comparing knowledge acquisition and machine learning*, AI 2001: Advances in Artificial Intelligence, Eds. Markus Stumptner; Dan Corbett; Mike Brooks, Berlin (2001), 273–284.

[22] P.K. Humphreys, R. McIvor, and F. Chan, *Using case-based reasoning to evaluate supplier environmental management performance*, Expert Systems with Applications **25** (2003), no. 2, 141–153.

[23] E. Keogh, K. Chakrabarti, M. Pazzani, and S. Mehrotra, *Locally adaptive dimensionality reduction for indexing large time series databases*, In proceedings of ACM SIGMOD Conference on Management of Data. Santa Barbara, CA (2001), 151–162.

[24] Eamonn J. Keogh, Kaushik Chakrabarti, Michael J. Pazzani, and Sharad Mehrotra, *Dimensionality reduction for fast similarity search in large time series databases*, Knowledge and Information Systems **3** (2001), no. 3, 263–286.

[25] Leilton Scandelari Lemos and Karl H. Kienitz, *Aprendizagem autônoma para gerenciamento de uma bolsa de valores simplificada*, SBAI2001 - V Simpósio Brasileiro de Automação Inteligente (2001).

[26] Tao Li, Qi Li, Shenghuo Zhu, and Mitsunori Ogihara, *Survey on wavelet applications in data mining*, SIGKDD EXplorations **4** (2003), no. 2, 49–68.

[27] Ulf Lindqvist and Phillip A. Porras, *Detecting computer and network misuse through the production-based expert system toolset (p-best)*, IEEE Symposium on Security and Privacy, 1999, pp. 146–161.

[28] F.O.S.S. Lisboa and M.C. Nicoletti, *O uso de possibilidade como medida de similaridade na classificação baseada em exemplares em domínios fuzzy*, Anais do V Simpósio Brasileiro de Automação Inteligente. Canela - RS (2001).

[29] Umberto Maia and Rita Maria Julia Silva, *Detection of telecommunication anomalies by means of a knowledge system that uses similarity search, Haar wavelet and RDR as support tools*, IEEE ICT 2005: 12th International Conference on Telecommunication (CapeTown, South Africa), 2005.

[30] Michael McCord, John Sowa, and Walter G. Wilson, *Knowledge systems and prolog: a logical approach to expert systems and natural language processing*, Addison-Wesley Longman Publishing Co., Inc., 1986.

[31] Pedro Alberto Morettin and Clélia Maria de Castro Toloi, *Previsão de séries temporais*, Atual Editora, 1985.

[32] F. Mörchen, *Time series feature extraction for data mining using dwt and dft*, OCT 2003.

[33] Bernhard Nebel and Kai von Luck, *Issues of integration and balancing in hybrid knowledge representation systems*, GWAI-87. 11th German Workshop on Artificial Intelligence (Berlin, Heidelberg, New York, Tokyo) (K. Morik, ed.), Springer-Verlag, September-October 1987, pp. 114–123.

[34] National Academy of Sciences, *Wavelets: Seeing the forest and the trees*, Dezembro 2001, http://www.beyonddiscovery.org/content/view.article.asp?a=1952.

[35] R. Jansen P. Compton, *Knowledge in context: a strategy for expert system maintenance*, Proceedings of the second Australian joint conference on Artificial intelligence (Adelaide,Australia), 1990.

[36] Ivan Popivanov and Renée J. Miller, *Similarity search over time-series data using wavelets*, ICDE '02: Proceedings of the 18th International Conference on Data Engineering (ICDE'02) (Washington, DC, USA), IEEE Computer Society, 2002, http://csdl.computer.org/comp/proceedings/icde/2002/1531/00/15310212abs.htm, p. 212.

[37] P. Preston, E. Edwards, P. Compton, and D. Litkouhi, *An expert system interpreter for time course data with refinement in context*, AAAI Spring Symposium: Artificial Intelligence in Medicine (1994), citeseer.ist.psu.edu/375274.html.

[38] P. Preston, G. Edwards, and P. Compton, *A 2000 rule expert system without knowledge engineers*, 1993.

[39] Deborah Christina Richards, *The reuse of knowledge in ripple down rule knowledge based systems*, Ph.D. thesis, The University of New South Wales, Dezembro 1998.

[40] J.M. Ruiz-Sanchez, R. Valencia-García, J.T. Fernández-Breis, R. Martínez-Béjar, and P. Compton, *An approach for incremental knowledge acquisition from text*, Expert Systems with Applications (2003), 77–86.

[41] S.J. Russell and P. Norvig, *Artificial intelligence: A modern approach*, Prentice Hall, 1995.

[42] Marcos Vinicius Pinto Salomon, *Fraude nas redes de telefonia celular*, Nov 2004, http://www.teleco.com.br/tutoriais/tutorialfraude/Default.asp.

[43] Tobias Scheffer, *Algebraic foundation and improved methods of induction of ripple down rules*, Proceedings of the Pacific Knowledge Acquisition Workshop PKAW '96, 1996, pp. 279–292.

[44] Rita Maria Julia Silva, Fernanda Emília Muniz de Rezende, and Antônio Eduardo Costa Pereira, *A hybrid KRS to treat fuzzy and taxonomic knowledge*, In ES2002: The twenty-second Annual International Conference of the British Computer Society's Specialist Group on Artificial Intelligence(SGES) **2** (2002).

[45] William J. Stevenson, *Estatística aplicada à administração*, 1986.

[46] Hendra Suryanto, *Learning and discovery in incremental knowledge acquisition*, Ph.D. thesis, The University of New South Wales, Janeiro 2005.

[47] M. Unser and A. Aldroubi, *A review of wavelets in biomedical applications*, Proc.IEEE, Special issue on Wavelets **84** (1996), no. 4, 626–638.

[48] Yunyue Zhu, *High performance data mining in time series:techniques and case studies*, Ph.D. thesis, New York University, Janeiro 2004.