Mineração de séries temporais de dados de sensores

Este exemplar corresponde à redação final da Dissertação devidamente corrigida e defendida por Leonardo Elias Mariote e aprovada pela Banca Examinadora.

Campinas, 26 de maio de 2008.

Claudia M. Bauzer Medeiros (Orientadora)

Dissertação apresentada ao Instituto de Computação, UNICAMP, como requisito parcial para a obtenção do título de Mestre em Ciência da Computação.

Substitua pela ficha catalográfica

(Esta página deve ser o verso da página anterior mesmo no caso em que não se imprime frente e verso, i.é., até 100 páginas.)

Substitua pela folha com as assinaturas da banca

Instituto de Computação
Universidade Estadual de Campinas

Mineração de séries temporais de dados de sensores

Leonardo Elias Mariote

Maio de 2008

Banca Examinadora:

- Claudia M. Bauzer Medeiros (Orientadora)
- Geovane Cayres Magalhães
  Instituto de Computação – UNICAMP
- Ricardo Rodrigues Ciferri
  Departamento de Computação – UFSCAR
- Sandro Rigo (Suplente)
  Instituto de Computação – UNICAMP
- Ricardo da Silva Torres (Suplente)
  Instituto de Computação – UNICAMP

Resumo

Redes de sensores têm aumentado a quantidade e variedade de dados temporais disponíveis. Com isto, surgiram novos desafios na definição de novas técnicas de mineração, capazes de descrever características distintas em séries temporais. A literatura correlata endereça problemas diversos, como indexação, classificação, definição de vetores de características e funções de distâncias mais eficazes. No entanto, a maioria dos trabalhos atuais tem como objetivo descrever e analisar os valores de uma série temporal, e não sua evolução. Além disto, vários fenômenos requerem uma análise mais elaborada, capaz de relacionar várias grandezas. Tal tipo de análise não pode ser realizada pela maioria das técnicas existentes hoje.

Esta dissertação apresenta uma técnica que descreve séries temporais sob uma premissa diferente – a de caracterizar a oscilação das séries e não seus valores propriamente ditos. O novo descritor apresentado – TIDES (TIme series oscillation DEScriptor) – utiliza os coeficientes angulares de uma segmentação linear da curva que representa a evolução das séries analisadas, em múltiplas escalas. Com isso, permite a comparação e a mineração de séries utilizando várias granularidades, enriquecendo a análise efetuada.

As principais contribuições são: (I) A especificação de um descritor que caracteriza a oscilação de séries temporais, ao invés de seus valores, utilizando múltiplas escalas; (II) A implementação deste descritor, validada por meio de dados sintéticos e reais; (III) A extensão do descritor de modo a suportar a análise de coevolução em um conjunto de séries.

Abstract

Sensor networks have increased the amount and variety of temporal data available. This motivated the appearance of new techniques for data mining, which describe different aspects of time series. Related work addresses several issues, such as indexing and clustering time series, and the definition of more efficient feature vectores and distance functions. However, most results focus on describing the values in a series, and not their evolution. Furthermore, the majority of papers only characterize a single series, which is not enough in cases where multiple kinds of data must be considered simultaneously.

This thesis presents a new technique, which describes time series using a distinct approach, characterizing their oscillation, rather than the values themselves. The descriptor presented – called TIDES (TIme series oscillation DEScriptor) uses the angular coefficients from a linear segmentation of the curve that represents the evolution of the analyzed series. Furthermore, TIDES suports multiscale analysis, what enables series and series mining under different granularities.

The main contributions are: (I) The specification of a descriptor that characterizes the oscillation of time series, rather than their values, unde multiple scale; (II) The implementation of this descriptor, validated for synthetic and real data; (III) The extension of the descriptor to support the analysis of the coevolution of a set of series.

Agradecimentos

Agradeço inicialmente a Deus, que tornou tudo isso possível.

Agradeço a meus pais e minha irmã, que com todo o apoio e compreensão, me conduziram durante todo esse processo. Eles são os grandes responsáveis por esse resultado.

Agradeço aos meus avós, que apesar de não estarem mais presentes, sempre estiveram ao meu lado.

Agradeço a professora doutora Claudia Bauzer Medeiros, por toda a orientação e auxílio, tantas vezes fundamental.

Agradeço também aos demais docentes do Instituto de Computação, em especial ao professor doutor Ricardo Torres, muitas vezes envolvido diretamente em meus projetos.

Agradeço ainda aos amigos que direta ou indiretamente me ajudaram inúmeras vezes.

E também, agradeço ao Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq) – em especial pelos projetos WebMaps e WebMaps2 –, a Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES), à Fundação de Amparo à Pesquisa do Estado de São Paulo (FAPESP), à Fundação de Pesquisa e Desenvolvimento em Telecomunicações (CPqD) e a empresa Trópico Telecomunicações Avançadas.

Sumário

Lista de Figuras

Capítulo 1

Introdução

A quantidade e a diversidade de aplicações que necessitam gerenciar informações temporais vem crescendo constantemente, gerando vários desafios em análise e gerenciamento. Estes desafios vão desde o nível de interface (visualização) até a busca e armazenamento de informações. A dissertação se concentra em análise e busca por similaridade de séries temporais, sob um novo prisma – enquanto todos os trabalhos na literatura analisam similaridade sob o ponto de vista de valores, a dissertação propõe a comparação em similaridade de oscilações. Este é o grande diferencial desta dissertação, no que se refere a apoiar novas aplicações.

Tais aplicações pertencem a diversos domínios, dentre os quais podemos citar algumas áreas da agricultura, análises econômicas de tendências de mercado, monitoramento de tráfego em grandes cidades. A importância econômica dessas aplicações é muito grande, principalmente em um país como o Brasil, no qual uma parcela significante do Produto Interno Bruto (PIB) é oriundo de atividades agrícolas.

Além disso, a crescente disseminação de dispositivos capazes de coletar informações, como sensores e redes de sensores, tende a aumentar o volume de informações disponíveis para os mais diversos domínios. Isso cria grandes coleções de dados temporais, também conhecidas como séries temporais, que descrevem inúmeros fenômenos ao longo dos anos. Essas séries podem ser vistas como streams de informações.

No entanto, o grande volume das bases que armazenam tais informações torna impraticável análises padrão. Com isso, surge a necessidade de técnicas que auxiliem o processo de análise desses dados, como por exemplo para detectar padrões, identificar erros em coletas ou estabelecer relações entre grandezas distintas. Dentre os tipos de análise mais comuns, um dos mais básicos e fundamentais é a busca por similaridade. Outros tipos de análise mais complexos podem ser contruídos utilizando como base mecanismos de busca por similaridade, como por exemplo predição de valores e detecção de erros.

Várias técnicas têm sido propostas para buscas por similaridade em séries temporais.

Essas técnicas atuam em várias etapas da busca, como definição e extração de um vetor de características, definição de funções de distância apropriadas para um dado fenômeno e mecanismos de indexação eficientes para buscas em séries temporais. Esses trabalhos abordaram uma série de problemas fundamentais para a área. Em geral, as técnicas propostas para o problema de busca por similaridade têm como objetivo descrever os valores de uma série temporal ao longo do tempo, definindo um mecanismo eficiente de recuperação.

No entanto, vários fenômenos não podem ser descritos por apenas um fator isolado, mas sim pela composição de vários fatores. Um exemplo disso é o estudo do clima de uma região, que requer combinar dados de sensores de temperatura, pluviosidade e humidade do ar, entre outros. Com isto, é necessário utilizar técnicas que analisem de maneira conjunta todos esses fatores, estabelecendo uma relação entre eles. Quando a evolução de um fator interfere na evolução de outro, diz-se que existe uma *coevolução de dados*.

A maior parte das técnicas existentes são capazes de avaliar um fator particular isolado, como por exemplo a taxa de pluviosidade em uma região com o decorrer do tempo. Poucas técnicas foram propostas para a análise de múltiplas séries, sendo além do mais voltadas apenas para a predição de valores. Até onde conhecemos, muitas funcionalidades desenvolvidas para séries de dados simples ainda não foram estendidas para coevolução.

A abordagem de considerar apenas os valores das séries pode trazer problemas devido a diferenças de representação e freqüências na obtenção das medidas que descrevem o fator sendo analisado. Essas diferenças podem ser oriundas de, por exemplo:

- uso de sensores de tipos diferentes
- sensoriamento realizado em locais diferentes (como, por exemplo, Campinas e Rio de Janeiro)
- sensoriamento realizado em períodos diferentes (como meses, estações ou anos distintos) ou intervalos não homogêneos
- análise conjunta de fenômenos distintos

Esta dissertação analisa o problema de busca por similaridade sob uma perspectiva diferente, motivada pela necessidade de usuários. Apresentamos um novo descritor para séries temporais, capaz de descrever padrões de evolução de dados e tendências de oscilação, ao invés de se concentrar nos valores das séries com o decorrer do tempo. Nosso descritor, o *TIDES* (TIme series oscillation DEScriptor), permite a sumarização e a comparação de séries distintas, em diferentes escalas temporais. Além disso, é imune a deslocamentos das séries no eixo-y.

A Figura 4.1 ilustra um exemplo de busca baseada na oscilação de duas séries, e não em seus valores propriamente ditos. A curva evolui da mesma forma nas regiões A e B, porém essas duas subseqüências estão deslocadas em relação ao eixo-y. Se essas subseqüências fossem submetidas a um descritor convencional, provavelmente seriam consideradas não similares, pelas próprias regras de comparação ponto-a-ponto utilizadas.

A representação das séries temporais proposta nesta dissertação é bastante adequada para a representação conjunta de várias séries. Desta forma, apresentamos um conjunto de extensões para o descritor, de modo a tranformá-lo em um descritor para séries com coevolução.

Realizamos vários experimentos com o descritor, que validaram as versões mono e multi escala, além de comparar sua eficácia com a de outro descritor bastante conhecido (o *Linear Scan*). Construímos ainda um protótipo para auxiliar na análise visual dos resultados obtidos nos experimentos.

Assim sendo, as principais contribuições são:

- A especificação de um descritor capaz para caracterizar a oscilação de séries temporais, considerando além disso a análise multi-escala dessas séries. Com isto, permite a comparação de séries utilizando granularidades diferentes.
- A implementação do descritor, validada para dados sintéticos e reais.
- A extensão do descritor para análise de coevolução de um conjunto de séries.

Parte das contribuições da dissertação foram publicadas no artigo *Diagnosing similarity of oscillation trends in time series*, publicado no workshop *Spatial and Spatio-Temporal Data Mining (SSTDM)*, na 17^a IEEE International Conference of Data Mining, em 2007 [MMT07].

O restante da dissertação está estruturada da seguinte forma. O capítulo 2 apresenta alguns conceitos e os principais tópicos relacionados com a dissertação. O capítulo 3 apresenta uma análise sobre algumas técnicas de mineração de séries temporais presentes na literatura. O capítulo 4 apresenta o descritor TIDES, em sua versão para apenas uma série, nas versões mono e multi escala do descritor. O capítulo 5 apresenta as extensões no descritor, transformando-o em um descritor para múltiplas séries. O capítulo 6 apresenta os experimentos realizados e os resultados obtidos. O capítulo 7 apresenta conclusões e possíveis trabalhos futuros.

Capítulo 2

Conceitos Básicos

Este capítulo apresenta uma revisão bibliográfica sobre os principais tópicos relacionados ao tema da dissertação. Para tanto, descreve inicialmente conceitos de bancos de dados temporais (seção 2.1) e bancos de streams (seção 2.2). Em seguida, apresenta algumas características sobre recuperação de informações baseada em conteúdo (seção 2.3). Após isto, explora conceitos gerais sobre busca por similaridade, bem como outras técnicas de mineração (seção 3). Na mesma seção, analisa como a literatura tem tratado os problemas de busca por similaridade em séries temporais.

2.1 Bancos de dados temporais

Bancos de dados temporais armazenam informações que variam de acordo com o tempo. Com este histórico, várias análises podem ser realizadas, trazendo uma maior compreensão do fenômeno estudado, como por exemplo tráfego em redes de dados ou evolução de dados financeiros [GzM03].

A área de bancos de dados temporais vem sendo estudada há muito tempo, com um elevado número de publicações e soluções. No entanto, ainda oferece um grande número de desafios.

2.1.1 Conceitos gerais de dados temporais

Para que seja possível uma melhor análise das técnicas e métodos existentes para dados temporais, é fundamental que se defina um conceito claro e preciso do que eles representam. Segundo Jensen e Snodgrass ([JS96]), um dado pode ser caracterizado como temporal quando apresenta mais de um *estado* ao longo do tempo. Um *estado* pode ser compreendido como um valor associado a um objeto durante um certo intervalo de tempo. Um objeto tem seu estado alterado por meio um *evento*. Um evento pode ser visto como

um fenômeno instantâneo. Tais definições nos levam a uma correlação entre estados e eventos.

Em uma análise mais simples, um objeto temporal pode ser visto como um conjunto de atributos que podem assumir diferentes valores em intervalos de tempo distintos. Em um determinado instante de tempo, porém, eles possuem apenas um valor. Esta definição nos leva diretamente ao conceito de seqüência de dados (*time sequence*). Segundo Shoshani e Kawagoe ([SK86]), uma seqüência de dados pode ser vista da seguinte forma: seja t uma tabela de entradas do tipo $< id, (tempo, valor)* >$, onde o asterisco indica repetição, ou seja, uma mesma identificação *id* pode assumir mais de um valor ao longo do tempo. A tupla (tempo, valor) representa uma *seqüência temporal*, e está associada à chave *id*. Este conceito é amplamente utilizado, como pode ser visto em [RM97, YSJ$^+$00]. Outros conceitos de dados temporais podem ser vistos em [JS97a].

A classificação de seqüências temporais de [SK86] permite um estudo mais apropriado para cada tipo de seqüência. Entre as características analisadas, encontram-se:

- *Regularidade*: Indica se o tempo entre as amostras da seqüência de dados é regular ou não.
- *Tipo*: Está vinculado às características de distribuição dos valores observados, podendo ser discreto, contínuo, constante em passos (patamares) ou eventos. É importante notar que as seqüências de dados do tipo contínuo podem necessitar de funções de interpolação, que determinem o conteúdo entre duas amostras obtidas.
- *Estaticidade*: Indica se a coleção de valores já foi completamente coletada, ou se ainda sofrerá acréscimo de valores.
- *Unidade de tempo*: Determina a unidade usada entre os pontos de amostra.

A análise destas características no fenômeno estudado pode auxiliar decisões de projeto. A regularidade de uma série, por exemplo, pode simplificar suas estruturas de armazenamento.

Uma vez compreendido o que é um dado temporal, é necessário analisar o que representa uma marca de tempo (*timestamp*). A marca de tempo é utilizada para indicar quando um atributo assume um novo valor. Nesta análise, serão considerados dois principais tipos de marcas de tempo ([JS96, JS97a, SK86]): o tempo de transação e o tempo válido. O tempo de transação representa o instante no tempo em que um certo registro foi adicionado no SGBD (Serviço Gerenciador de Banco de Dados). O tempo válido, por sua vez, representa o tempo em que um *fato*[1] [SK86] realmente ocorreu no mundo real.

[1]Fato, segundo [SK86], representa um evento armazenado que corresponde a uma verdade no modelo do mundo real utilizado, em um instante de tempo

Segundo [JS96], quando os valores de tempo (válido e de transação) não são iguais, eles guardam uma relação entre si, que pode ser classificada em retroativa ou preditiva. Na relação retroativa, os dados são válidos antes de serem armazenados. Na relação preditiva, os dados armazenados se tornarão válidos em algum momento após terem sido armazenados.

Vários outros tipos de marcas de tempo são citados na literatura, entre os quais destaca-se o tempo de decisão, que representa o momento em que ocorreu a decisão que levou à consumação do fato. Um exemplo de modelo que trabalha com o conceito de tempo de decisão é o modelo de Thompson [JS96]. Segundo [JS96], porém, os diferentes tipos de tempo definidos no modelo, na verdade, representam semânticas distintas atribuídas ao tempo válido e ao de transação, que podem ser obtidos através dos conceitos de especialização/generalização.

Outro conceito vinculado a seqüências de dados são as *séries temporais*. Uma série temporal, assim como uma seqüência de dados, representa uma seqüência de valores associados a um objeto, sendo que o objeto não possui dois valores distintos no mesmo instante de tempo [RM97]. As principais diferenças entre séries temporais e seqüências de dados são relativas ao volume de dados e ao conjunto de operações necessário para realizar todas as manipulações desejadas com os dados brutos.

2.1.2 Modelos de dados temporais

Uma vez definido o conceito de dado temporal, o próximo passo é analisar como ele pode ser visto em um modelo de dados. Segundo Jensen e Snodgrass ([JS97b]), dados temporais são dificilmente expressos pelos modelos de dados padrão (não temporais). Isto faz com que problemas que utilizam informações temporais sejam normalmente resolvidos pelos desenvolvedores das aplicações, acima do nível do SGBD. Em [EGSV98], Erwig et al mostra que alguns fenômenos espaço-temporais, como o movimento, não podem ser representados de maneira pura em um modelo computacional atual. Para alguns fenômenos, no entanto, alguns artefatos podem ser utilizados, como por exemplo a interpolação linear.

O uso de modelos tradicionais para dados temporais pode elevar a complexidade de implementação de um sistema, e ainda levar ao surgimento de soluções proprietárias. Neste contexto, ao menos dois tipos distintos de modelos devem ser analisados: o modelo conceitual e o modelo lógico. Podem ser identificadas duas grandes linhas no que diz respeito ao tratamento de atributos temporais. A primeira defende que um atributo temporal seja visto como qualquer outro, enquanto que a segunda defende que um atributo temporal seja visto de forma diferente dos demais.

Várias soluções distintas são defendidas na literatura para o modelo lógico dos dados [JS96]. Segundo Parent et al ([PSZ99]), qualquer tipo de atributo, objeto ou relação pode

possuir a característica de ser temporal ou espacial, sendo assim tratados da mesma forma que os demais elementos do banco. [SK86] apresenta uma visão similar a esta, defendendo que qualquer atributo ou tabela isolada pode ser visto como uma série de dados, o que possibilita uma simples definição de operadores, como por exemplo a restrição e a composição. [EGSV98], por sua vez, trata os atributos temporais de maneira distinta dos demais. Algumas técnicas e conceitos, como a normalização e as dependências funcionais, apresentam características diferentes para dados temporais [JS97b]. Essas diferenças podem simplificar, e até mesmo justificar, uma representação diferenciada para esse tipo de dados.

A literatura correlata cobre inúmeras propostas de modelos, com implementações usando vários modelos de bancos de dados – relacional, orientado a objetos [Bot95], e outros mais específicos para, por exemplo, objetos móveis [Yi04].

2.1.3 Operações sobre dados temporais

Existem várias técnicas para a implementação de operações sobre dados temporais. Segundo [JS97b], consultas sobre dados temporais utilizam normalmente os operadores de desigualdade, enquanto que consultas tradicionais utilizam normalmente o operador de igualdade. Além disto, a maioria dos dados temporais serão inseridos em ordem crescente de marca de tempo. Essa característica pode ser utilizada para simplificar as estruturas de indexação utilizadas para dados temporais. O próprio conceito de série temporal aproveita o fato das tuplas serem armazenadas em ordem de marca de tempo.

Uma técnica utilizada é a definição de funções de regressão capazes de gerar algum tipo de interpolação ou inferência para processar algumas operações sobre atributos temporais. Nesta linha, Trudel ([Tru97]) apresenta uma representação para predicados de primeira ordem com características temporais. Para tanto, define o uso de funções no plano cartesiano para projetar os valores de cada predicado com o decorrer do tempo, tendo como eixos do plano o momento no tempo e o valor do predicado, utilizando operações de derivação e integração. [EGSV98] também apresenta as aproximações lineares como uma das possíveis formas de ligar pontos de amostras de dados temporais, criando uma representação contínua para tais dados.

É importante notar que as técnicas que utilizam modelos matemáticos para aproximar valores de atributos entre as amostras presentes são compatíveis com séries temporais contínuas. Nesses casos, as técnicas de regressão polinomial tendem a apresentar um melhor resultado.

2.2 Bancos de dados de streams

O conceito básico de bancos de dados de *streams* é reflexo do surgimento de aplicações baseadas em dados coletados de sensores, que produzem *streams* ou *fluxos de dados*. Para estes tipos de aplicação, o valor de um atributo em um momento de tempo é transitório, e apresenta um ou mais fluxos de atualização contínuos, em grande velocidade e de tamanho imprevisível. Estas características trazem uma série de desafios para a área de bancos de dados. Os SGBDs tradicionais não estão aptos a trabalhar com tamanho volume de dados e atualizações. Com isso, torna-se necessária a definição de novas técnicas e estruturas de acesso. Exemplos de aplicações que utilizam fluxos de dados para análise incluem monitoramento ambiental, aplicações financeiras, gerência de redes, entre outras.

2.2.1 Conceitos gerais

Do ponto de vista de armazenamento, um fluxo de dados pode ser visto como uma seqüência de valores associada a um atributo qualquer, variando com o tempo. Porém, muitas vezes não é possível representar o valor de tal atributo em qualquer momento de tempo t. Nesses casos, o fluxo de dados representa amostragens desses valores em diferentes momentos t_i. Em geral, a taxa de amostragem de tais valores é bastante alta, visando a diminuir o intervalo entre as amostras. Streams também são vistos como arquivos do tipo *append-only*, com registros transientes [BBD$^+$03]. Isto significa que geram repositórios de dados nos quais registros são apenas adicionados e, após um novo registro ser inserido, alguns registros antigos tornam-se inválidos (ou descartáveis).

Tal volume de dados e de atualizações satura os SGBDs tradicionais rapidamente [BBD$^+$03], afetando diretamente seus métodos de busca, indexação e estruturas de acesso. Além disto, alguns conceitos intrínsecos aos SGBDs tradicionais não podem ser aplicados a streams de dados, como por exemplo a necessidade de precisão na resposta a uma consulta. Em uma base cujos dados representam streams, torna-se necessário trabalhar com a idéia de resposta aproximada ([BBD$^+$03, CCC$^+$02]). Segundo Carney et al ([CCC$^+$02]), outras funcionalidades são necessárias para um banco de dados que manipule streams, como por exemplo um acesso eficiente ao histórico de um atributo, suporte efetivo a gatilhos, alarmes e operações de tempo real.

Golab e A zsu ([GzM03]), por sua vez, definem vários requisitos com os quais um SGBD voltado para a gerência de streams deve se preocupar. Exemplos são: Operações baseadas em tempo e em ordem; Mecanismos de sumarização para armazenamento de um stream; Obtenção de respostas aproximadas; Impossibilidade de utilizar operadores bloqueantes (por exemplo, algumas junções temporais); Execução paralela de consultas, visando a aumentar o desempenho do sistema.

Alguns dos operadores necessários são comuns aos presentes nos SGBDs tradicionais

(*seleção, agregação aninhada, agrupamento e uniões, junções*). Surgem ainda operações novas, mais voltados para streams (*consulta a itens freqüentes, mineração de streams, consultas orientadas a janelas*).

Segundo Babcock et al ([BBD+03]), as consultas a bancos de dados de streams podem ser classificadas como *consultas instantâneas* e *consultas contínuas*. As consultas instantâneas são muito parecidas às presentes nos bancos de dados tradicionais. Elas são sempre referentes aos valores de um atributo em um dado instante de tempo. As consultas contínuas, por sua vez, são executadas de tempos em tempos em uma base de dados, tendo como resultado um conjunto crescente de tuplas selecionadas [BBD+03, TGNO92, GzM03]. Neste contexto, é possível detectar algumas semelhanças entre este mecanismo e o conceito de *bases de dados ativas*, que se baseia no mecanismo de gatilhos para implementar sua funcionalidade. Segundo o conceito de semântica contínua [TGNO92], o resultado de uma consulta contínua é o conjunto de tuplas que seriam retornadas se a consulta fosse executada a cada momento t_i. Segundo [GzM03], para consultas monotonicamente crescentes, o recálculo de uma resposta a uma consulta contínua necessita apenas calcular as novas tuplas desde a última vez em que o resultado da consulta foi obtido. Consultas não monotônicas precisam ser recalculadas sobre todo o histórico de valores, sempre que novos valores são obtidos, para que o resultado final seja calculado.

[BBD+03] ainda classifica as consultas em *predefinidas* e *ad hoc*. As consultas predefinidas são fornecidas ao SGBD antes que os dados aos quais elas se referem tenham sido obtidos (como ocorre, por exemplo, nas consultas contínuas). As consultas ad hoc são disparadas a qualquer momento, após o fluxo de dados ter sido iniciado. Com isto, é impossível otimizar as consultas previamente. Além disto, os dados envolvidos podem ter sido obtidos há algum tempo, e talvez já não estejam disponíveis para acesso rápido.

2.2.2 Problemas Típicos e Soluções Propostas

Vários problemas surgem devido ao grande volume dos dados analisados. A alta taxa de atualizações, por exemplo, inviabiliza o uso de várias das estruturas de acesso atuais, visto que seria gasto muito tempo atualizando índices e estruturas auxiliares. Desta forma, várias técnicas são propostas visando a diminuir a quantidade de informações utilizadas para uma certa operação.

A forma de acesso ao conjunto de informações não representa o único problema encontrado em bancos de dados de streams. Algumas operações comuns nos bancos de dados tradicionais, como alguns tipos de junção (*theta-join*, por exemplo) ou ainda consultas aninhadas, representam *operações bloqueantes* em bancos de dados de streams. Uma operação bloqueante consiste em uma operação que não consegue produzir a primeira tupla de seu resultado antes de ter percorrido toda a sua entrada [BBD+03, GzM03].

Tais tipos de operação nunca acabariam em um banco de dados de streams, visto que um stream pode ser infinito. Uma maneira de se resolver tal problema é limitar o conjunto de dados analisados ao tamanho de uma janela finita. A impossibilidade de manter um stream em memória, devido ao seu tamanho imprevisível, unida a características de tempo real muitas vezes presentes em bancos de dados de streams tornam necessária a existência de novas técnicas e algoritmos.

Outras técnicas interessantes para o problema do grande volume de informações dizem respeito à construção de tabelas de hashing para auxílio na busca [GzM03, BBD+03]. Para os casos em que tais soluções não sejam interessantes, por exemplo por problemas de desempenho, o uso de algoritmos aproximados pode ser apropriado para a solução. Nesse caso, há um compromisso entre a precisão da resposta e o consumo de recursos computacionais e tempo.

O conceito de janela de dados foi criado visando a contornar o problema do tamanho imprevisível de um stream. Segundo [GzM03], uma janela pode ser caracterizada como um subconjunto de um stream que é utilizado como "fonte de dados"para uma consulta, visto que em uma grande parte das consultas, apenas um trecho do stream de dados é importante para a análise. [BBD+03] considera que a utilização de janelas de dados representa uma maneira natural de empregar um conceito de aproximação de respostas que pode ser compreendido pelo usuário (como por exemplo, utilizar apenas os dados das duas últimas semanas para produzir uma resposta). Tal técnica é determinística e utiliza os dados mais recentes, uma característica desejável para várias aplicações. [GzM03] classifica as janelas de dados de acordo com a direção do limites da janela, do tipo dos limites da janela (físico, como o tempo, ou lógico, como o número de tuplas) e do intervalo de atualização.

Outra técnica utilizada em consultas temporais em streams diz respeito à obtenção de respostas aproximadas [BBD+03, GzM03]. Esta abordagem é aceitável para uma grande parte das aplicações que manipulam dados temporais. Para tanto, surgiram técnicas utilizadas para redução de volume de dados e construção de sinopses, dentre as quais podem ser citados os esboços *(sketches)*, a amostragem aleatória, os histogramas e os *wavelets*, além das próprias janelas de dados. Tais tipos de técnicas se tornam importantes quando o tempo para o armazenamento de um valor acrescido do tempo para recálculo de uma resposta se tornam maiores que o tempo entre amostragens de um stream.

Outro problema característico de bancos de dados de stream diz respeito ao acesso a valores já processados em um stream. Segundo [BBD+03], uma vez que um valor foi processado, ele não deve ser mais acessado. Muitas vezes acessar um dado previamente processado nem é possível, pelo fato dele não ser armazenado. Porém, consultas podem fazer referência a valores antigos, que não se encontram mais presentes na base. Uma técnica desenvolvida para possibilitar tais tipos de consultas é a de *sumarização*, que

armazena algumas informações que resumem o conjunto de dados analisados, visando a possibilitar respostas aproximadas a futuras consultas.

Com relação a consultas contínuas, Terry et al ([TGNO92]) define formas de traduzir uma consulta contínua monotonicamente crescente descrita em linguagem TQL em uma consulta descrita em SQL, que deve ser executada segundo um algoritmo simplificado.

2.3 Recuperação baseada em conteúdo

A quantidade e a variedade de dispositivos capazes de capturar imagens e disponibilizá-las de maneira eletrônica têm crescido significativamente, elevando o número de imagens disponíveis. Esses dispositivos vão de câmeras digitais até sensores capazes de recuperar imagens.

Esse grande volume de imagens fez surgir a necessidade de técnicas capazes de analisá-las, sem depender de uma classificação textual para cada uma delas. Nesse contexto, a área de recuperação de informações baseada em conteúdo cresceu, e se tornou um importante foco de pesquisas [SWS+00, TSMR03, FT07]. Muitos dos conceitos desenvolvidos para recuperação de imagens por conteúdo, no entanto, podem ser utilizados para a recuperação de informações de uma maneira mais ampla. Tais conceitos serão utilizados na dissertação para o tratamento e recuperação de séries temporais.

A recuperação de imagens requer alguns passos básicos [SWS+00, TSMR03]. Esses passos definem um conjunto de blocos funcionais que devem estar presentes em um sistema de recuperação por conteúdo. De uma maneira simplificada, podemos considerar que os principais passos são:

- A partir de uma interface apropriada, um usuário deve ser capaz de especificar uma consulta. Isso pode ser feito, por exemplo, através da indicação de uma imagem de consulta.

- Gerar uma representação computacional para o objeto fornecido na consulta, obtendo um conjunto de características de interesse. Essa representação será utilizada para comparar o objeto de consulta com os objetos armazenados na base de dados.

- Calcular a similaridade entre dois objetos, definindo assim uma métrica de distância entre eles. O cálculo da distância levará em consideração o conjunto de características obtidas para representrar cada um dos objetos (tanto os armazenados na base de dados, como os objetos de consulta).

- Classificar um conjunto de objetos de acordo com um critério de similaridade. Isso significa que, a partir de um objeto de consulta e uma base de dados, é necessário

ordenar os objetos da base tendo como referência o objeto de consulta. Para realizar tal operação, utiliza-se a métrica de distância definida.

- Uma vez que uma consulta tenha sido realizada, e que seu resultado tenha sido ordenado, é necessário apresentar esses resultados para o usuário. A forma como os resultados são apresentados pode alterar muito a percepção do usuário em relação ao mecanismo de busca.

- A última etapa é a de aprimoramento de uma consulta. Nela, o usuário pode refinar indiretamente os parâmetros de configuração dos descritores utilizados. Isso normalmente é realizado por meio da aprovação ou reprovação dos resultados apresentados. Esse mecanismo é conhecido como *relevance feedback*.

Apesar de serem especificadas para recuperação de imagens, essas etapas definem um fluxo, ou um pseudo-algoritmo, capaz de realizar uma função de recuperação de informações por conteúdo mais ampla, independente do tipo do objeto analisado e das características de interesse. A maior parte das etapas requer um grande esforço em pesquisa [TSMR03, FT07].

No entanto, dois passos são bases para o desenvolvimento do nosso trabalho: (1) a representação de um conjunto de características e (2) o cálculo de similaridade entre objetos. Esses passos serão descritos com mais detalhes nas seções a seguir.

2.3.1 Extração do vetor de características

A recuperação de qualquer objeto digital exige sua caracterização, que determina os aspectos principais necessários para descrevê-lo. Na área de processamento de imagens, o conjunto de informações necessárias para caracterizar um objeto, utilizando um determinado conjunto de características, é denominado *vetor de características*. Esse vetor de característica quantifica um ou mais atributos do objeto analisado, criando uma maneira computacional de representar o objeto. O formato do vetor é completamente dependente das características que se deseja quantificar do objeto, e da técnica utilizada para fazê-lo. Desta forma, é muito comum que técnicas diferentes de recuperação de conteúdo definam ou utilizem vetores de características distintos.

Na área de recuperação de imagens por conteúdo, por exemplo, uma série de propriedades podem ser usadas, como por exemplo a cor, a textura e a forma das imagens [SWS+00, OVY01, TFC02]. Dependendo da propriedade utilizada, características diferentes da imagem serão privilegiadas. Em alguns casos, uma determinada propriedade pode não refletir o que uma análise visual refletiria.

Nesses casos, é considerado que existe um *gap semântico* entre o que foi quantizado e o que um usuário gostaria de representar [SWS+00]. Várias linhas de pesquisas buscam

diminuir o gap semântico de uma consulta por uma imagem. Como exemplo disto, temos o mecanismo de *relevance feedback*, que envolve uma interação com o usuário de modo a refinar o mecanismo de busca [FT07].

O gap semântico pode ser originado pelo fato do descritor e do usuário analisarem um mesmo objeto utilizando características diferentes. Isso não significa que um descritor é melhor ou pior, mas sim que o conjunto de propriedades analisado no problema não é o ideal para o conjunto de usuários alvo. O gap semântico não ocorre apenas com imagens. Pode, por exemplo, se manifestar na recuperação de séries temporais, como se verá no decorrer do texto.

2.3.2 Cálculo de distância entre vetores de características

Após extrair o vetor de características, o próximo passo é definir como comparar vetores. Desta forma, é possível indicar se dois objetos são similares em relação às propriedades representadas no vetor de características [SWS+00].

É importante notar que, de acordo com as propriedades analisadas, dois objetos podem ser ou não ser similares. Assim, a função de distância escolhida para uma técnica normalmente está relacionada ao vetor de características gerado por ela.

Alguns algoritmos para cálculo de funções de distância são bastante conhecidos e utilizados, como por exemplo a distância de *Manhattan* (também conhecida como distância *L1*) e a distância Euclidiana (também conhecida como distância *L2*) [TFC02]. Outras técnicas definem funções de distância específicas, que utilizam características do vetor gerado. Outro fator relevante na escolha da função de distância é o seu desempenho. Em alguns casos, torna-se necessário construir uma função de distância que apresente uma complexidade assintótica inferior. Várias simplificações podem ser utilizadas para isso. Na seção 3.4 analisaremos algumas técnicas definidas para séries temporais.

2.3.3 Descritor

O conceito de descritor é bastante utilizado na área de recuperação de imagens por conteúdo. Um descritor pode ser visto como um par $< f_1, f_2 >$, onde f_1 representa uma função de extração de características e f_2 representa uma função de cálculo de distâncias.

A função de extração de características é a responsável por obter um vetor de características V a partir de um objeto qualquer O (seção 2.3.1).

A função de cálculo de distâncias, por sua vez, é responsável por quantizar a distância entre dois vetores de características V_1 e V_2 (seção 2.3.2).

A Figura 2.1 ilustra o modelo conceitual de um descritor. Nela está representada a relação entre as duas funções (f_1 e f_2).

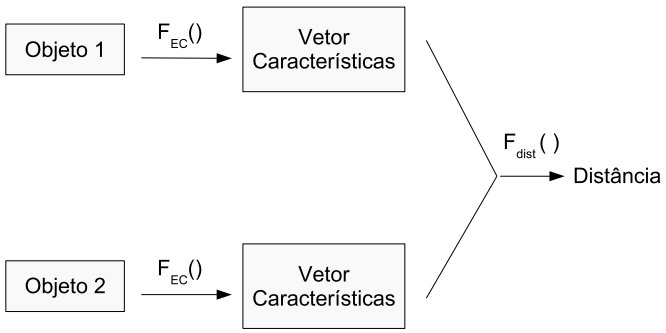


Figura 2.1: Modelo conceitual de um descritor. Nele, estão representadas as funções de extração de características (f_1) e de cálculo de distância (f_2).

2.4 Resumo

Este capítulo apresentou alguns conceitos referentes a séreis temporais e streams, concentrando-se em aspectos que são utilizados na busca por similaridade entre séries. Tais aspectos – função de caracterização e função de distância – serão a base de pesquisa apresentada nos capítulos 4 e 5 e dos experimentos realizados, descritos no capítulo 6.

Capítulo 3

Mineração de séries temporais

A área de mineração de dados é responsável por obter informações de um conjunto de dados presentes em uma base usando mecanismos diferentes dos oferecidos por SGBDs para consultas. Um vasto conjunto de operações pode ser classificado como mineração de dados, como por exemplo buscas por similaridade e detecção de padrões. Várias aplicações necessitam de técnicas de mineração, como monitoramento de tráfego, vibrações de pontes, monitoramento ambiental, de tráfego em redes e de acesso a disco [Fal02].

Uma das operações básicas de mineração é a busca por similaridade, que é a ênfase desta dissertação. O restante deste capítulo descreve algumas características comuns da área de mineração, e analisa com mais detalhes algumas soluções e técnicas para a busca por similaridade de séries temporais.

3.1 Conceitos Gerais

A mineração de dados já é um conceito bastante difundido. É responsável por análises que buscam padrões ou características específicas em um conjunto de dados, muitas vezes utilizando um conjunto pré-definido de medidas e padrões de comparação.

As atividades realizadas no contexto de mineração de dados incluem predição de valores, busca por similaridade, detecção de padrões, detecção de *outliers* e classificação, entre outros. Para tanto, há várias técnicas, como aprendizado supervisionado e não supervisionado, técnicas de regressão linear, programação dinâmica e decomposição de sinais [HK02, HPMA$^+$00, FRM94].

Segundo Han e Kamber ([HK02]), a mineração de dados pode ser classificada, segundo seus objetivos, em descritiva ou preditiva. A mineração descritiva tem por objetivo resumir as informações e salientar características interessantes no conjunto de dados analisados. A mineração preditiva, por sua vez, tem por objetivo projetar modelos para prever possíveis futuros valores nas relações analisadas.

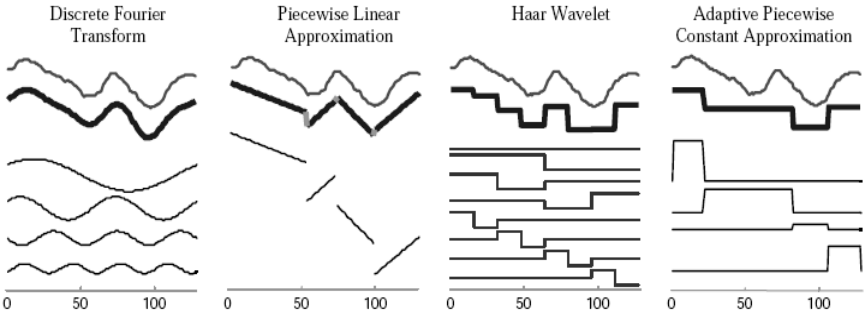


Figura 3.1: Resumo de algumas técnicas de extração de características. Figura obtida de [LKLC03].

A mineração descritiva tem como base o conceito de generalização, utilizando caracterizações analíticas para identificação da relevância de cada atributo. Ela também é usada na determinação de regras de associação, na qual uma das técnicas bastante utilizada é a da família dos algorítmos *Apriori*.

A mineração de dados apresenta um papel bastante importante no caso de análise de streams e de dados temporais, pois é capaz de caracterizar a evolução do comportamento de um elemento, com o passar do tempo. A área de mineração de dados temporais ainda não foi explorada em sua totalidade, uma vez que demanda a captura de associações que envolvam o elemento tempo (vide [JS97b], por exemplo).

A Figura 3.1 resume algumas das principais diferenças entre enfoques de extração de características, nos trabalhos de mineração de séries temporais.

3.2 Mineração de séries temporais

De forma similar ao que ocorre nos demais usos de técnicas de mineração de dados, a mineração de séries temporais prevê um conjunto de operações básicas. Entre essas operações, estão a busca por similaridade, a descoberta de padrões e regras, predição de valores, detecção de *outliers*, classificação, e outras [Fal02].

Segundo Faloutsos ([Fal02]), existe uma grande relação entre várias destas operações. Isto torna possível construir um conjunto básico de técnicas que podem ser utilizadas em várias dessas operações. Como exemplo disto, a combinação de operações como descoberta de padrões e regras com busca por similaridade gera subsídios para a predição de valores.

A predição de valores está diretamente associada a uma busca no passado por padrões de informação similares aos encontrados no momento atual [WSS+05] . Uma forma de se

obter a predição de valores é a partir de técnicas de regressão numérica. Elas são capazes de gerar parâmetros que caracterizam curvas definidas pelos dados que pertencem à série de dados em análise [YSJ+00, RM97, FRM94]. Desta forma, é possível determinar uma função $f(t)$ que aproxima os valores da série analisada. A precisão da predição depende do tipo de regressão utilizada e da característica de distribuição dos dados. Por exemplo, uma regressão linear aplicada a dados que não possuam uma distribuição contínua pode levar a uma má aproximação. Operações como aprendizado supervisionado e não supervisionado também podem ser utilizadas para a predição de valores [BBD+03].

Outra operação bastante importante é a detecção de erros em uma série, por exemplo, devido a problemas na captura ou armazenamento das informações. Uma das maneiras possíveis de se detectar um erro é comparar o valor de um dado com um valor previamente inferido [Fal02]. Caso estes valores sejam muito diferentes, o valor real coletado pode caracterizar um *outlier*, com grande probabilidade de que este valor possua um erro.

A descoberta de padrões, por sua vez, tem por objetivo identificar informações presentes no conjunto dos dados que dificilmente seriam observadas manualmente. Para tanto, ela analisa o comportamento das séries temporais com o passar do tempo, e descobre padrões escondidos, buscando trechos de seqüências que se repetem na série analisada. Várias técnicas podem ser utilizadas para a detecção de padrões, dentre as quais está a programação dinâmica [HPMA+00].

A operação de classificação é uma das operações mais importantes. Ela pode ser utilizada em várias áreas do conhecimento, como na classificação de séries que representam dados médicos ou séries de multimidia, como por exemplo vídeos. Dentre as várias abordagens existentes para a classificação de séries temporais, podemos citar o uso de redes neurais [SGCD06]. Além de utilizar redes neurais para classificar as séries, essa abordagem propõe um algoritmo específico para extrair seus vetores de características.

Como nosso trabalho apresentra uma operação de busca por similaridade, analisaremos esta operação com mais detalhes nas próximas seções. Para tanto, inicialmente verificaremos algumas técnicas e modelos computacionais utilizados para representar séries temporais. Após isto, apresentaremos algumas técnicas bastante utilizadas para realizar buscas por similaridade.

3.3 Caracterização de uma série temporal

Várias abordagens são utilizadas para caracterizar uma série temporal. Dentre as principais, podemos citar: (I) técnicas de processamento de sinais; (II) decomposição em segmentos lineares; e (III) funções de análise de patamares.

Várias formas de processamento de sinais podem ser utilizadas para caracterizar séries temporais [Fal02]. Uma delas é a *transformada discreta de Fourier* (DFT), que é capaz

de eliminar ruídos inseridos no momento da captura das informações. A DFT representa uma função por meio de uma série de coeficientes. Segundo Faloutsos et al ([FRM94]) um pequeno número de coeficientes já é suficiente para a obtenção de uma boa descrição para a maior parte das funções. Outra técnica para a extração de características de curvas é a *decomposição por valor singular* (SVD), na qual apenas as dimensões mais significativas do conjunto de dados analisado são mantidas, diminuindo as dimensões do espaço de dados inicial. Assim, usa-se um espaço de menor dimensionalidade. Esta técnica, porém, pode apresentar um desempenho não satisfatório [Fal02].

Uma série ainda pode ser caracterizada a partir de uma decomposição da curva que a representa em um conjunto de segmentos lineares. Segundo Keogh e Smyth ([KS97]), esta técnica pode se adequar à maior parte dos tipos de dados, mesmo quando eles não são localmente estacionários no tempo, situação na qual as representações espectrais (como a DFT) não apresentam bons resultados. Além disto, a técnica apresenta uma boa adequação a ruídos, sendo conceitualmente mais simples. Alguns algoritmos são capazes de realizar segmentação de curvas de forma bastante eficiente. Keogh et al ([KCHP01]) apresenta três abordagens principais para a segmentação de uma série em seqüências de segmentos: a *top down*, a *bottom up* e a de *janelas deslizantes*.

A abordagem *bottom up* trabalha de uma maneira *gulosa*, seguindo as seguintes etapas:

- Inicialmente, todos os pontos de amostra da série são ligados por segmentos.
- Após isso, é calculado qual erro seria introduzido caso se removesse cada um dos pontos de amostra.
- O ponto que, ao ser retirado, introduzir um menor erro, será removido.

Esse processo é repetido até que se atinja algum critério de parada.

A abordagem *top down*, por sua vez, utiliza um processo reverso. Inicialmente, apenas um segmento é utilizado, ligando o primeiro e o último pontos. É calculado o ganho na representação da série caso cada um dos pontos excluídos seja representado. O ponto que apresentar um maior ganho é escolhido, e passa a ser representado na série. O processo é repetido até que se atinja algum critério de parada. Os critérios de parada escolhidos são normalmente dependentes do domínio dos dados analisados.

A abordagem de *janelas deslizantes* é utilizada para representar séries contínuas, ou seja, que não possuem um conjunto fixo de pontos. Através de testes realizados com as três abordagens, [KCHP01] concluiu que a abordagem *bottom up* apresentou os melhores resultados em um grande número de bases de testes. Uma abordagem híbrida, que combina a *bottom up* e a *janelas deslizantes*, também gerou bons resultados.

A abordagem de Keogh e Pazzani ([KP98]) é um exemplo de técnica que utiliza segmentação linear para representar uma série temporal. Nessa abordagem, após o processo

de segmentação linear, o usuário pode atribuir pesos aos diferentes segmentos gerados. Os pesos atribuídos a cada segmento, bem como seus pontos extremos, são armazenados e utilizados posteriormente no cálculo da distância entre diferentes séries. Com estes segmentos e pesos, define-se uma operação de junção (baseada em um *merge*) de séries de dados, que enfatizam similaridades ou diferenças entre duas séries. Tal operação é capaz de, por exemplo, auxiliar na classificação de séries de dados.

Essa flexibilidade permite ao usuário definir quais segmentos são mais relevantes na representação de sua série, por meio de um mecanismo de *relevance feedback* (ou realimentação de relevância). É importante notar, no entanto, que a abordagem utiliza os pontos extremos de cada segmento. Essa característica não permite resolver alguns problemas comuns em mecanismos de busca por similaridade, como por exemplo problemas de escala e deslocamentos.

Vários estudos estão sendo realizados de modo a representar séries temporais por meio de patamares (funções de degrau) [KCPM01, LKLC03]. A curva que representa uma série é transformada em uma curva aproximada, sendo substituída por uma seqüência de patamares capaz de representar a série original.

Lin et al ([LKLC03]) apresenta uma técnica simples de descrição de séries por patamares. Nela, são coletadas amostras de uma série em intervalos regulares de tempo. Cada amostra define um patamar. Em alguns casos, a amostra pode representar, por exemplo, a média de valores do último intervalo, ao invés de um valor instantâneo na série.

Keogh et al ([KCPM01]), por sua vez, apresenta uma adaptação na técnica de representação por patamares. Nela, os patamares não apresentam sempre os mesmos tamanhos, como o que ocorre em [LKLC03]. Ao invés disso, o tamanho de cada patamar é adequado de modo a melhor representar a série. Essa abordagem, no entanto, apresenta algumas dificuldades para indexação, uma vez que tanto o número de patamares utilizados como o tamanho de cada patamar podem variar. Para resolver esse problema, [KCPM01] propõe uma solução baseada em duas métricas de distâncias distintas. A primeira métrica busca a melhor aproximação possível para a série analisada. A segunda métrica de distâncias garante a geração de uma distância extritamente menor que a distância real. Essa segunda métrica pode ser calculada de maneira mais simplificada.

A Figura 3.2 ilustra uma curva representada por uma seqüência de quatro patamares. O tamanho de cada patamar é ajustado, de modo a introduzir um erro menor ao representar a curva. As setas sobre os patamares indicam os ajustes feitos em seus tamanhos.

A maior parte das técnicas que utilizam a abordagem *APCA* consideram que as séries análisadas são estáticas, ou seja, um conjunto pré-definido de séries armazenadas em uma base. Junkui e Yuanzhen ([JY07]), por sua vez, apresentam algumas adaptações nos algoritmos de segmentação para tratar uma série temporal como um *stream* contínuo de dados. Para tanto, trabalha com algumas aproximações, e permite que novos valores

Figura 3.2: Representação aproximada de uma série utilizando 4 patamares. Figura obtida de [KCPM01].

sejam constantemente adicionados a uma série temporal sendo analisada.

Keogh e Pazzani ([KP00]), por sua vez, exploram o problema de indexação de séries representadas por meio da abordagem *APCA*. Em seu trabalho, [KP00] realiza vários testes comparando a eficácia e eficiência de sua estrutura de indexação com outras conhecidas. Entre as vantagens apresentadas, pode ser citado o fato de que a estrutura de indexação proposta pode se adequar a várias funções de distância.

Uma abordagem bastante utilizada como complemento para descrição de séries temporais é a representação simbólica [LKLC03, KR05, PLC99]. Nessa abordagem, uma série temporal é de alguma forma convertida em uma seqüência de símbolos. A conversão está diretamente ligada à técnica utilizada para caracterizar a curva. Essa representação permite a utilização de algoritmos de casamento de textos para comparação de séries.

Além dessas três grandes vertentes de técnicas de representação de curvas, existem várias outras abordagens conhecidas. Dentre elas, podem ser citadas a classificação de trechos de curvas em figuras geométricas com pesos associados [KP98] e a classificação de trechos da curva em estados de uma máquina de estados que descreva o fenômeno em questão [WSS+05].

3.4 Funções de distância

Como já mencionado, a função de distância está fortemente ligada à caracterização de uma série. Assim como no caso mais amplo analisado na seção de recuperação de informações por conteúdo (veja seção 2.3.2), algumas funções de distância muito utilizadas são a Euclidiana (*L2*) e a Manhattan (*L1*)

No entanto, para alguns vetores de características, essas funções de distância não são apropriadas [KR05, PLC99] – por exemplo, vetores de características que utilizam representação simbólica. Entre elas, podemos citar técnicas baseadas em programação dinâmica ou ainda algumas técnicas utilizadas na área de bioinformática, como distâncias de edição [KR05, PLC99]. Para calcular a distância entre duas séries representadas de maneira simbólica é necessário quantificar a distância entre cada par de símbolos utilizados.

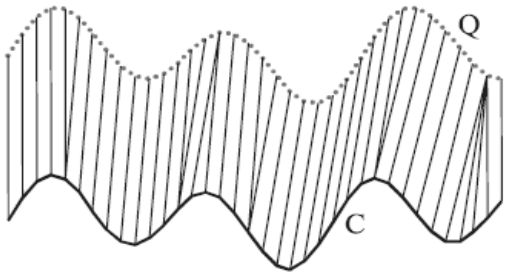


Figura 3.3: Comparação entre duas séries – Q e C. Apesar de similares, apresentam uma pequena diferença de escala entre si. As linhas entre as duas curvas demonstram como seus pontos deveriam ser comparados. Figura extraída de [KR05].

Muitas vezes, no entanto, duas séries que são similares em comportamento podem estar deslocadas no eixo do tempo, ou ainda apresentar pequenas variações de escala temporal. Nessas situações, funções de distância que realizem análise ponto-a-ponto, como por exemplo as métricas *L1* e *L2* classificariam as duas séries como não similares.

Para atacar esse problema, foi proposta uma abordagem capaz de identificar tais deslocamentos ou alterações em escala, denominada *Time Warping Distance* [KR05, PLC99]. Nela, ao invés de se calcular a distância entre duas séries em cada instante de tempo, são calculados os pontos que devem ser comparados. A Figura 3.3 ilustra o caso de duas séries similares, com uma pequena diferença de escala no eixo do tempo.

A abordagem *Time Warping Distance* foi por muito tempo considerada incapaz de permitir um mecanismo exato de indexação [ALSS95]. No entanto, algumas técnicas recentes estão se mostrando capazes de resolver o problema de indexação utilizando o conceito de *Dynamic Time Warping* [KR05]. Assim, restrições relativas a desempenho podem ser superadas, viabilizando o uso desse conceito de distância em vários cenários distintos.

Apesar dos problemas de deslocamentos e escala no eixo do tempo terem sido endereçados pela abordagem *Time Warping Distance*, até onde conhecemos nenhuma abordagem tenta resolver o problema de deslocamentos e escala no eixo dos valores (eixo y). Esse tipo de abordagem seria capaz de caracterizar a oscilação de uma curva, e não seus valores propriamente ditos. Nosso descritor apresenta essa característica. Ela pode ser combinada com a abordagem *Time Warping Distance*, de modo a construir um descritor imune a deslocamentos e diferenças de escala em todas as dimensões analisadas.

3.5 Abordagens específicas

Vários descritores vêm sendo propostos, utilizando as mais diversas formas de caracterização e funções de distâncias. Além das técnicas apresentadas nas seções anteriores, algumas abordagens são propostas para problemas nos quais um conhecimento específico do domínio do problema pode ser utilizado. Em geral, essas técnicas não podem ser totalmente aproveitadas em um cenário mais amplo. Porém, muitas delas apresentam características que podem ser adaptadas. A seguir apresentaremos algumas destas técnicas: Uso de transformações em séries durante o processamento das consultas; Busca por similaridade em dados hierárquicos; Busca por similaridade em séries temporais periódicas.

Busca por similaridade com uso de transformações

A técnica de Rafiei e Mendelzon ([RM97]) visa a disponibilizar ao usuário consultas mais elaboradas, permitindo que ele interfira na comparação de duas séries temporais. Considere, por exemplo, um caso de comparação de vendas de filiais de uma loja. Imagine que uma rede de lojas coleta diariamente o número de vendas realizadas, porém para uma matriz o importante é analisar os números semanais. Neste caso, a própria consulta deveria considerar a média semanal, e não o levantamento diário presente.

Para tanto, [RM97] define um vetor de transformações do tipo $T = [a^n, b^n]$, aonde a^n representa um vetor de n dimensões, responsável pelo ajuste das dimensões dos coeficientes, e b^n representa um vetor de n dimensões responsável pela realização do deslocamento da curva. A partir de tal vetor de transformações, usa-se a Transformação de Fourier, adequando a curva ao tipo de consulta desejado.

Em seu algorítmo de busca, [RM97] aplica o vetor de transformação a cada elemento a ser comparado, construindo um índice multidimensional, que é utilizado para reduzir o número de candidatos a série similar. Após isto, aplica uma métrica de distância entre a série desejada e os candidatos a série similar, e retorna as séries cuja distância em relação à série da consulta for menor que um dado limite inferior.

Este enfoque possui problemas de desempenho, pois sempre constrói todo o índice de acordo com o tipo de consulta realizada. Ele somente é válido para a busca por similaridade em seqüências completas, não sendo adequado à busca por subseqüências.

Busca por similaridade em dados hierárquicos

O mecanismo de busca por similaridade em dados hierárquicos de Yang et al ([YKT05]) leva em consideração a estrutura hierárquica do conjunto de dados analisado

Para isso, [YKT05] apresenta uma maneira de se transformar uma árvore de dados em uma representação linear, na qual as informações topológicas dos dados são mantidas

por meio de informações específicas. Uma vez linearizados, dois objetos distintos podem ser comparados por meio de uma função de distância convencional, como por exemplo a distância L1. Os objetos para os quais a distância não ultrapassar um limite estipulado serão considerados possíveis soluções para a consulta. Nesse conjunto de objetos deve ser aplicada a função de distância específica, que leva em consideração a topologia hierárquica dos dados.

Uma vez que o cálculo da distância que leva em consideração informações hierárquicas é mais custoso, a redução do número de elementos para os quais esta métrica deve ser aplicada melhora o desempenho de uma consulta.

A abordagem de eliminar falsos candidatos em uma busca por similaridades utilizando representações simplificadas dos objetos pode ser utilizada no contexto do descritor Multi-escala, presente na seção 4.5

Busca por similaridade em séries periódicas

Séries de dados periódicas representam seqüências de valores nas quais trechos são repetidos de tempo em tempo. É possível representar os trechos de tais seqüências por meio de uma máquina de estados, na qual cada estado representa um dos trechos. Dois estados são vizinhos se os trechos que representam são consecutivos.

Com foco neste tipo de dados, Wu et al ([WSS+05]) define uma técnica para mineração de dados, que combina busca por similaridade e predição de valores. Seus resultados são voltados para o auxílio no tratamento médico de pessoas que sofram de um certo tipo de câncer no abdomen. Para tanto, deve prever a localização de um determinado conjunto de células, a partir do movimento que elas sofrem devido ao processo respiratório.

[WSS+05] introduz o conceito de *subseqüência estável*, que representa o ajuste dinâmico do tamanho das subseqüências, levando em consideração a amplitude e freqüência de mudanças na série de dados. O ajuste automático do tamanho do trecho analisado apresenta um melhor resultado do que o uso de subseqüências de tamanhos previamente determinados.

Para otimizar a busca por similaridade, [WSS+05] realiza uma varredura dos streams, e verifica quais deles apresentam um estado inicial igual ao procurado. Com isto, o número de trechos para os quais será aplicada uma função de distância tende a diminuir, melhorando o desempenho da solução.

3.6 Busca de subseqüências

Muitas vezes, as técnicas para busca por similaridade são apresentadas apenas para séries temporais completas. Assim, para comparar duas séries temporais S_1 e S_2, elas exigem

que todos os elementos das séries sejam comparados [RM97]. Essa exigência, muitas vezes, está indiretamente representada na extração do vetor de características. Porém, muitas vezes uma série temporal pode apresentar a característica de ser contínua, ou seja, ela não possui uma quantidade fixa de elementos.

A busca de subseqüências torna-se muito interessante, na medida em que várias espécies de sensores coletam informações de maneira contínua, gerando grandes séries que crescem constantemente. Desta forma, é de grande importância conseguirmos comparar apenas trechos dessas grandes séries. Esses trechos podem, por exemplo, ser referentes a períodos constantes de tempo, como meses ou anos.

Ao analisarmos um processo de busca por subseqüências nos deparamos com um problema: encontrar a posição dentro de uma extensa série temporal S_1 a partir da qual devemos comparar uma subseqüência S_2. Este problema pode ser formulado como segue: Seja $S =< s_1, s_2, ..., s_n >$ uma série temporal e $T =< t_1, t_2, ..., t_j >$ uma subseqüência que está sendo buscada, com $j < n$. Queremos encontrar um índice i para o qual $s_i \approx t_1, s_{i+1} \approx t_2, ..., s_{i+j} \approx t_j$.

Faloutsos et al ([FRM94]) utiliza o conceito de *janelas deslizantes* para resolver o problema da comparação de subseqüências. Isso significa que, ao invés de se comparar uma série temporal completa, serão analisados apenas trechos (ou *janelas*) da série. Assim, as séries analisadas sempre serão referentes a intervalos de, por exemplo, w valores, ou seja, representarão $S[k, k + w - 1]$. Esses intervalos são armazenados de maneira disjunta na base de dados, ou seja, para cada período de tempo, é armazenada uma série com w elementos. A partir do momento em que um novo valor for introduzido na série, um novo intervalo será analisado e armazenado na base, referente aos valores $S[k + 1, k + w]$. Desta forma, ao analisar uma série armazenada, na verdade estamos analisando uma subseqüência de uma série temporal contínua, referente a um dado intervalo de tempo. Essa abordagem permite que utilizemos qualquer técnica desenvolvida para buscas em séries completas em cenários nos quais queremos comparar subseqüências de séries.

O uso direto da abordagem de janelas deslizantes, no entanto, pode aumentar muito o número de séries temporais analisadas, e assim se mostrar inviável em algumas situações. Para resolver este problema, [FRM94] propõe uma estrutura de indexação baseada em árvores R*, capaz de aumentar significativamente o desempenho de um mecanismo de busca.

Já [WSS+05] ataca o problema das buscas por subseqüências delimitando os trechos nos quais serão realizadas as análises, por meio do conceito de *subseqüência estável*. Apesar de apresentar bons resultados, esta técnica necessita que as séries analisadas apresentem características periódicas. Desta forma, dificilmente pode ser utilizada para um caso mais amplo de buscas por similaridade.

3.7 Evolução de dados

Uma série de dados caracteriza a evolução de um elemento com o passar do tempo. A análise de uma série de dados possibilita a compreensão de sua evolução, até mesmo correlacionando fatores associados.

Um fenômeno importante relacionado com a evolução de dados é a *co-evolução*. Ocorre em ambientes com mais de uma série de dados, no qual a evolução de uma série está correlacionada com a evolução de outras [Fal02]. O desafio é realizar uma análise conjunta das séries [YSJ+00], descobrindo relações e aumentando o conhecimento do fenômeno analisado.

Ao analisar a co-evolução dos dados, Yi et al ([YSJ+00]) define uma técnica para predição de valores, detecção de outliers e de correlação entre diferentes streams, denominada MUSCLES (MUlti-SequenCe LEast Squares), que tem como fundamento principal uma regressão linear de múltiplas variáveis. Para tanto, trabalha com um grupo de streams que de alguma forma são relacionados, e utiliza todos eles para detectar possíveis valores de uma destas séries, por meio de uma regressão linear que leva em conta todos os últimos w valores de cada uma das k séries analisadas.

Com a regressão linear, [YSJ+00] é capaz de realizar a predição de um valor de uma série S levando em consideração os últimos valores das $k - 1$ séries relacionadas e os últimos valores da própria série S. Nenhum valor é armazenado para posterior consulta, o que inviabiliza o uso de outras técnicas de mineração.

3.8 Resumo

Este capítulo apresentou alguns conceitos referentes a mineração de dados, voltados para a área de mineração de séries temporais. Dentre eles, se encontram algumas das técnicas mais recentes, utilizadas em descritores de séries temporais.

Várias técnicas tem sido propostas, tanto para caracterizar uma série temporal como para comparar duas séries. No entanto, até onde conhecemos, os descritores atuais tem como objetivo analisar os valores das séries temporais, e não sua oscilação. Além disto, poucos resultados foram apresentados para o problema de análise de séries com coevolução, e nenhum trabalho foi encontrado para a busca por similaridade em séries com coevolução.

Apresentamos nesse capítulo algumas técnicas que serão utilizadas no descritor proposto, dentre as quais podemos citar o algoritmo de segmentação de curvas proposto por [KCHP01] e o uso de representação simbólica nos vetores de características. É importante notar que o uso de segmentação linear para caracterizar as curvas permite facilmente ao nosso descritor analisar a oscilação das séries, ao invés de seus valores.

Capítulo 4

Descritor TIDES para uma única série temporal

Este capítulo apresenta o TIDES (TIme series oscillation DEScriptor) – um descritor que caracteriza uma série temporal baseado na forma como seus dados oscilam com o decorrer do tempo. Isso o diferencia da maior parte dos descritores atuais, que descrevem a seqüência de valores de uma série. Desta forma, podemos endereçar vários problemas não abordados por outros descritores, como deslocamentos de curvas no *eixo-y* e a comparação de séries que apresentam fenômenos distintos, nas quais os valores coletados possuam grandezas diferentes. A Figura 4.1 exemplifica um caso de deslocamento no *eixo-y*. Na figura, as regiões A e B oscilam da mesma forma com o decorrer do tempo, porém estão deslocadas.

O capítulo é organizado da seguinte maneira: a seção 4.1 descreve uma visão simplificada do descritor; as seções 4.2 e 4.3 descrevem as técnicas utilizadas para extrair as informações necessárias das séries temporais brutas; a seção 4.4 descreve como funciona

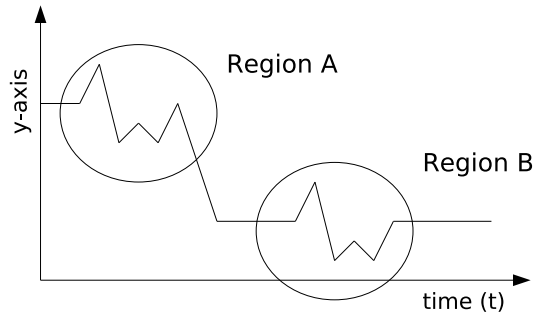


Figura 4.1: Problema de deslocamento de séries no eixo Y.

a função de distância adotada na solução; a seção 4.5 descreve a introdução da análise multi-escala no TIDES.

4.1 Visão Geral

O TIDES é obtido da seguinte forma: Uma série de valores, que representa uma série temporal bruta, é inicialmente transformada em uma seqüência de segmentos de reta que melhor a representa. A informação utilizada de cada segmento de reta é o seu coeficiente angular em relação ao eixo horizontal. Este coeficiente indica quão inclinado é o segmento de reta em relação ao eixo dos y.

A seguir, os coeficientes são submetidos a uma função de classificação. Esta função atribui uma classe, representada por um símbolo, a cada um dos segmentos de reta associados. Essa seqüência de símbolos passa a representar a série de valores. Portanto, a nossa proposta é baseada na representação simbólica da série.

Desta forma, o vetor de características do TIDES é composto por uma seqüência de tuplas do tipo $< s, l >$, onde s é o símbolo e l a projeção do segmento associado no eixo x. A distância entre duas séries de valores passa então a ser a distância entre duas séries de símbolos. A determinação do descritor pode ser especificada pelos passos:

1. Extração de características de uma série

   - A série $S = \gamma_1, ..., \gamma_j$ é transformada em $j - 1$ segmentos $(\gamma_1, \gamma_2), ..., (\gamma_{j-1}, \gamma_j)$;
   - É feita uma redução no número de segmentos, tendo como resultado n segmentos que descrevam a série;
   - Um ângulo a_i é associado a cada um dos segmentos, gerando o vetor: $< a_1, l_1 >, ..., < a_n, l_n >$, onde l_i é o tamanho da projeção do segmento i no eixo x.
   - Os ângulos passam por uma função de classificação, que as classifica dentre n_g possíveis classes. Atribuindo-se um símbolo y_i a cada classe, obtém-se o vetor de características:

   $$V =<< y_1, l_1 >, ..., < y_n, l_n >> \tag{4.1}$$

   Os valores de n e n_g são fornecidos pelo usuário especialistas do domínio.

2. Definição da função de distância

   - Dados dois vetores V_1 e V_2, normalizá-los em NV_1 e NV_2, de forma que tenham o mesmo número de pontos;

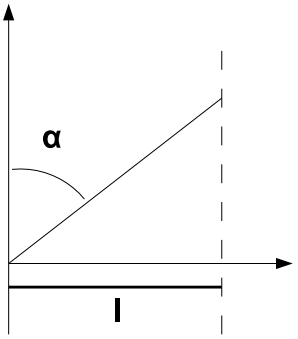


Figura 4.2: Conceitos de ângulo e projeção do segmento no eixo X.

- Calcular a distância entre NV_1 e NV_2, computada como a soma das distâncias entre os símbolos correspondentes;

A Figura 4.2 mostra os conceitos de ângulo de um segmento em relação ao eixo vertical (α) e a projeção de um segmento no eixo X (l).

4.2 Transformação em segmentos

Nosso vetor de características descreve uma série temporal como uma seqüência de segmentos lineares, utilizando para isso um algoritmo de segmentação linear. Cada um dos segmentos possui um ângulo de inclinação em relação ao eixo vertical, representando a oscilação da curva em certo instante de tempo. Nesta forma de representação, dois segmentos de reta são unidos por um único ponto. Estes pontos serão denominados de *pontos de união* no restante deste trabalho.

Para a implementação desta etapa foi utilizada a técnica de segmentação linear proposta por [KCHP01], com a abordagem *bottom-up*. Nesta técnica, inicialmente os pontos da série original são ligados por segmentos de reta. Após isto, a cada etapa da iteração um segmento é removido, da seguinte forma: sejam $\{\gamma_1, \gamma_2, ..., \gamma_n\}$ os pontos de uma série temporal, e $\{\{\gamma_1, \gamma_2\}, \{\gamma_2, \gamma_3\}, ..., \{\gamma_i, \gamma_{i+1}\}, ..., \{\gamma_{n-1}, \gamma_n\}\}$ os segmentos de reta que os interligam. Para remover o segmento γ_i, γ_{i+1}, deverão ser removidos $\{\gamma_i, \gamma_{i+1}\}$ e $\{\gamma_{i+1}, \gamma_{i+2}\}$, e inserido o segmento $\{\gamma_i, \gamma_{i+2}\}$ no lugar. A projeção no eixo x do novo segmento será igual à soma das projeções dos dois segmentos removidos no mesmo eixo. Desta forma, o número de segmentos da seqüência terá sido reduzido em um elemento. A cada iteração, é selecionado o segmento cuja remoção introduzirá menos erro na representação da curva. Erro, neste contexto, foi definido como sendo a variação do eixo Y que ocorre com a remoção de um dado segmento, no ponto de interseção entre os segmentos envolvidos.

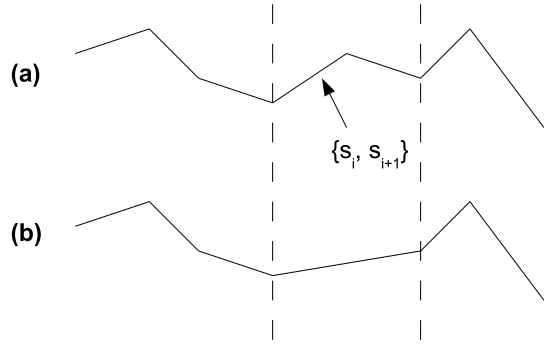


Figura 4.3: Eliminação de um segmento de reta.

A Figura 4.3 ilustra esse processo. Enquanto a série ilustrada em (a) possui ambos os segmentos $\{\gamma_i, \gamma_{i+1}\}$ e $\{\gamma_{i+1}, \gamma_{i+2}\}$, a série em (b) possui o segmento $\{\gamma_i, \gamma_{i+2}\}$ substituindo os dois anteriores.

Uma série de critérios de parada foram implementados, dentre os quais:

- Limiar de erro acumulado introduzido pela eliminação de segmentos
- Limiar de erro introduzido pela remoção de um segmento
- Número fixo de segmentos

As três opções de parada foram testadas, em mais de 1800 consultas, utilizando séries representadas por 60 segmentos. A opção que apresentou um melhor desempenho, levando-se em consideração sua simplicidade, foi a de número fixo de segmentos. Este critério foi adotado na etapa de caracterização das séries.

4.3 Representação do vetor de características

Nosso vetor de características utiliza os coeficientes angulares para caracterizar padrões de oscilação de uma série, representando a inclinação de um segmento em relação ao eixo vertical. Desta forma, uma série S que tenha sido transformada em uma seqüência de n segmentos de reta pode ser expressa como:

$$VS = << a_1, l_1 >, < a_2, l_2 >, ..., < a_n, l_n >> \tag{4.2}$$

onde a_i representa o coeficiente angular do segmento i e l_i representa o comprimento de sua projeção no eixo x.

O uso de ângulos para representar um segmento de reta, no entanto, pode introduzir uma série de problemas no cálculo da similaridade e no armazenamento dos vetores (em geral, por problemas de precisão na representação de seus valores). Para resolver tal problema, adotamos uma técnica cada vez mais utilizada na descrição de séries temporais: a representação simbólica [LKLC03, PLC99]. Nesta técnica, valores são substituídos por símbolos. Isto é feito da seguinte forma: Inicialmente, os coeficientes angulares de uma série são submetidos a uma função de classificação, que atribui uma classe a cada um dos coeficientes.

Assim, o vetor de características V de uma série passa a ser representado como:

$$V =<< y_1, l_1 >, < y_2, l_2 >, ..., < y_n, l_n >> \tag{4.3}$$

onde y_i é o símbolo que representa a classe atribuída ao ângulo a_i.

Utilizamos uma função de classificação simples, que agrupa coeficientes angulares em faixas contínuas. Como um coeficiente angular em uma série temporal pode variar entre 0^o e 180^o (ou 0 e π radianos), a faixa dos possíveis valores (0 a 180) é particionada em n_g grupos. O valor de n_g deve ser definido por especialistas no domínio da série analisada.

Desta forma, a função de classificação F_{class} atribuirá à classe j o coeficiente angular a_i quando:

$$(i-1) * \frac{180}{n_g} \leq a_i \leq i * \frac{180}{n_g} \tag{4.4}$$

4.4 Função de Distância

Esta seção descreve a função de distância proposta, que utiliza o vetor de características V descrito na expressão 4.3.

A representação simbólica permite a utilização de várias funções de distância, como por exemplo as baseadas na técnica de programação dinâmica. Tais funções de distância são comumente utilizadas em áreas como bioinformática e processamento de *strings*. Como exemplo destas funções pode ser citada a distância de edição.

O TIDES, no entanto, utiliza uma função de distância mais simples, baseada na Função de Distância Manhattan (L1). O cálculo da distância entre dois vetores de características é composta por duas fases: a fase de normalização e a fase do cálculo de distância propriamente dito.

Na fase de normalização, dois vetores distintos V_1 e V_2 com diferentes números de elementos ($n_{V_1} \neq n_{V_2}$) são normalizados em dois novos vetores, NV_1 e NV_2, com o mesmo número de elementos. A normalização ocorre da seguinte maneira: suponha que, em um instante de tempo t_i, apenas uma das séries – V_1 – possui um ponto de união. Neste caso,

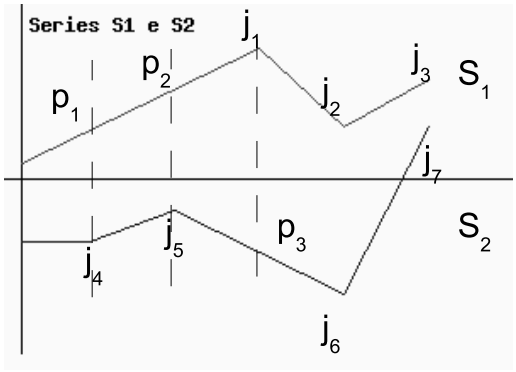


Figura 4.4: Processo de normalização de duas séries.

será inserido um ponto de união artificial na outra série – V_2 – para o mesmo instante de tempo t_i. Este ponto artificial transforma um único segmento de reta em dois segmentos distintos, que possuem a mesma inclinação.

A Figura 4.4 ilustra o processo de normalização para as séries S_1 e S_2, que foram segmentadas e após isto normalizadas. Os pontos $j_1, j_2, ..., j_7$ são os pontos de união originais e os pontos p_1, p_2 e p_3 são os pontos artificiais criados pelo processo de normalização.

É importante notar que é possível realizar o processo de desnormalização, ou seja, gerar V_1 a partir de NV_1. Para tanto, basta remover os pontos de união artificiais. Um ponto de união artificial pode ser identificado pelo fato de unir dois segmentos de reta com a mesma inclinação.

Estando os vetores V_1 e V_2 normalizados em NV_1 e NV_2, o cálculo da distância é dado por:

$$D(NV_1, NV_2) = \sum_{i=1}^{i=m} d(< y_{i,1}, l_{i,1} >, < y_{i,2}, l_{i,2} >) \tag{4.5}$$

onde $d(< y_{i,1}, l_{i,1} >, < y_{i,2}, l_{i,2} >)$ representa a distância entre os i-ésimos segmentos dos vetores normalizados NV_1 e NV_2 e m é o tamanho dos vetores normalizados (n_{NV_1} e n_{NV_2}).

A distância d, por sua vez, é dada por:

$$d(< y_{i,1}, l_{i,1} >, < y_{i,2}, l_{i,2} >) = (y_{i,1} - y_{i,2}) * l_i \tag{4.6}$$

onde a distância entre dois símbolos quaisquer α e β é dada por:

$$d(\alpha^{esimo} simbolo, \beta^{esimo} simbolo) =\mid \alpha - \beta \mid \tag{4.7}$$

Desta forma, a distância entre dois símbolos consecutivos é sempre 1.

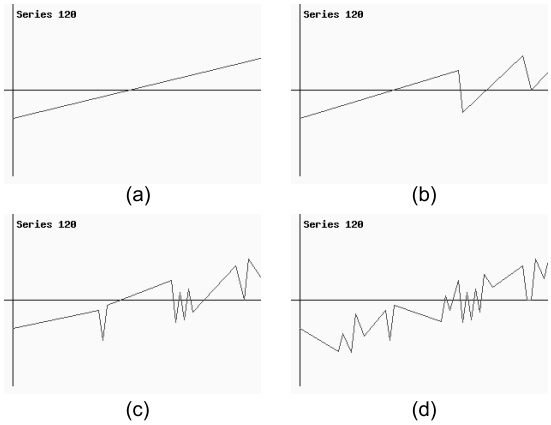


Figura 4.5: Quatro representações distintas da mesma série, utilizando respectivamente 1, 5, 15 e 55 segmentos.

4.5 Análise Multi-Escala

Como descrito nas seções anteriores, o TIDES utiliza um processo de transformações de séries em segmentos lineares para, por meio dos coeficientes angulares de cada segmento, descrever a oscilação da série. Porém, a análise da oscilação de uma série varia muito de acordo com a granularidade adotada (número de segmentos, ou seja, n). De uma maneira geral, quanto maior o número de segmentos utilizados, mais sensível se torna a análise de oscilação da série. Isto significa que, com poucos segmentos, é dada mais importância à tendência global da série, enquanto que com muitos segmentos, as oscilações locais ganham mais importância.

A Figura 4.5 ilustra quatro representações diferentes para uma mesma série, cada representação utilizando um número de segmentos diferente. Descrições realizadas com poucos segmentos permitem análises mais simplificadas das séries, como por exemplo, caracterizá-las como ascendentes, descendentes ou estacionárias.

Desta forma, estendemos nossa solução para uma abordagem multi-escala. Com isto, o usuário pode comparar duas séries analisando-as em mais de uma granularidade. Para tanto, ampliamos o conceito do nosso vetor de características, fazendo com que um vetor multi-escala VM seja representado por:

$$VM =< V_1, V_2, ..., V_k > \tag{4.8}$$

onde V_i representa o vetor de características para a série analisada utilizando n_i segmentos.

Desta forma, VM caracteriza uma série em mais de uma escala (ou seja, com granularidades distintas). Tanto a quantidade de escalas utilizadas, como o número de segmentos

utilizados em cada uma delas devem ser definidos por algum especialista no domínio estudado.

Dado que VM é composto por k vetores de características, é possível computar a distância entre dois vetores VM_1 e VM_2 da seguinte maneira. Para cada par de vetores $V_{1,i} \in VM_1$ e $V_{2,i} \in VM_2$, é utilizada a função de cálculo de distâncias definida na seção 4.4, considerando que V_1 e V_2 têm o mesmo número de segmentos.. Desta forma, podemos construir um vetor de distâncias DM:

$$DM =< d_{n1}, d_{n2}, ..., d_{nk} > \tag{4.9}$$

onde d_{nsi} representa a distância calculada para o vetor de características construído com ns_i segmentos.

Uma vez que o vetor de distâncias DM tenha sido calculado, uma série de análises pode ser realizada. Em alguns casos, duas séries podem parecer completamente distintas quando analisadas em uma escala, porém podem ser similares em outra escala. Por exemplo, duas séries podem ser descritas como 1-similares e 10-não similares (ou seja, possuem padrão de oscilação semelhante se caracterizadas por um único segmento, e padrões diferentes se caracterizadas por 10 segmentos). DM pode ainda ser submetido a alguma função de agregação, como por exemplo uma soma. A análise multi-escala torna o descritor TIDES mais versátil, capaz de se adequar melhor a diferentes domínios de aplicação.

A Figura 4.6 ilustra um exemplo de duas séries com comportamentos de oscilação bastente interessantes. Quando realizamos uma análise global nas séries (por exemplo, utilizando apenas um segmento), elas se mostram completamente diferentes – Figura 4.6 (a). Porém, conforme aumentamos o número de segmentos utilizados na análise, o comportamento oscilatório das duas se torna muito mais similar – Figura 4.6 (c).

Isto pode ser depreendido pela diminuição de distância entre as séries, neste caso. A Figura 4.7 mostra a comparação efetuada entre os ângulos dos segmentos lineares que compõem as séries, para o caso (c).

4.6 Algoritmos

Cada uma das etapas utilizadas pelo descritor foram anteriormente descritas. Nessa seção, apresentamos alguns algoritmos, escritos em uma meta-linguagem, que identificam cada um dos passos e suas seqüências de execução.

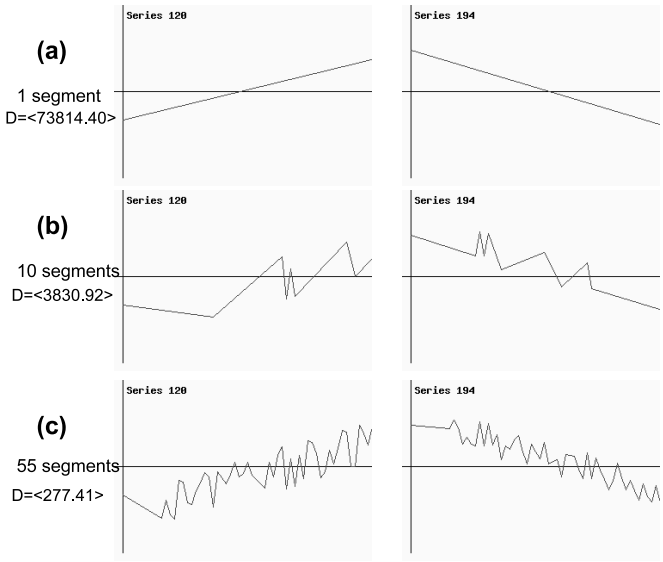


Figura 4.6: Análise da oscilação de duas séries, utilizando quantidades de segmentos distintas.

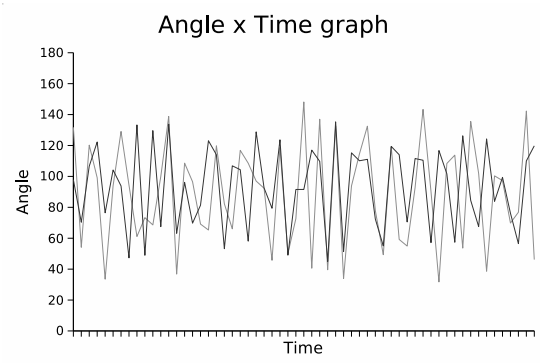


Figura 4.7: Seqüência de coeficientes angulares dos segmentos de reta que descrevem as séries 120 e 194.

4.6.1 Algoritmos de Extração de Características

Apresentamos inicialmente o algoritmo para a obtenção de um vetor de características, a partir de uma série temporal analisando uma única escala. Esta série está no seu formato bruto, ou seja, representa uma seqüência de tuplas, no formato $< tempo, valor >$.

Algorithm 1 ExtracaoUmaEscala(S, n_{seg}, n_g)

1: Segmente a série temporal S, de modo a obter um vetor de segmentos de reta Vr, com n_{seg} segmentos, ou seja:
2: $V_r \leftarrow$ segmentar (S, n_{seg});
3: **for** $i = 1$ to n **do**
4: $< \gamma_i, \gamma_{i+1} > \leftarrow Vr[i]$;
5: $a_i \leftarrow$ coeficienteAngular($< \gamma_i, \gamma_{i+1} >$);
6: $l_i \leftarrow$ projecaoEmX($< \gamma_i, \gamma_{i+1} >$);
7: $y_i \leftarrow$ classificacaoCoeficiente(a_i, n_g);
8: $V[i] \leftarrow < y_i, l_i >$; {$V$ é o vetor de características}
9: **end for**
10: RETURN;

A vertente de múltiplas escalas representa uma extensão da versão de uma única escala do descritor. As modificações inseridas são pontuais, e estão representadas no algoritmo 2.

Algorithm 2 ExtracaoMultiEscalas($S, C_{escalas}, n_g$)

1: **for** $j \in C_{escalas}$ **do**
2: Segmente a série temporal, de modo a obter um vetor de segmentos de reta Vr, com j segmentos
3: $V_r \leftarrow$ segmentar (S, j);
4: **for** $i = 1$ to j **do**
5: $< \gamma_i, \gamma_{i+1} > \leftarrow Vr[i]$;
6: $a_i \leftarrow$ coeficienteAngular($< \gamma_i, \gamma_{i+1} >$);
7: $l_i \leftarrow$ projecaoEmX($< \gamma_i, \gamma_{i+1} >$);
8: $y_i \leftarrow$ classificacaoCoeficiente(a_i, n_g);
9: $VM[j][i] = < y_i, l_i >$; {VM é o vetor de características multi-escala}
10: **end for**
11: **end for**
12: RETURN;

4.6.2 Algoritmo de Cálculo de Distâncias

Nesta seção, apresentamos o algoritmo utilizado para o cálculo da distância entre dois vetores de características. A função de cálculo de distâncias para a versão multi-escalas do descritor apresenta apenas alguns ajustes em relação à versão que analisa uma única escala. Para cada escala analisada, os vetores de características MV_1 e MV_2 podem ser comparados utilizando o algoritmo 3. Desta forma, o valor da distância d calculado para cada uma das escalas de interesse pode ser armazenado em sua respectiva posição no vetor de distâncias DM.

Algorithm 3 DistanciaUmaEscala(V_1, V_2)

1: Aplique o processo de normalizacao de Vetores a V_1 e V_2, produzindo NV_1 e NV_2;
2: $d \leftarrow 0$;
3: **for** $i = 1$ to $|NV_1|$ **do**
4: $\quad < y_{1,i}, l_{1,i} > \leftarrow NV_1[i]$
5: $\quad < y_{2,i}, l_{2,i} > \leftarrow NV_2[i]$
6: $\quad d \leftarrow d +$ DistanciaSimbolos$(y_{1,i}, y2, i) * l_{1,i}$;
7: **end for**
8: RETURN d

4.7 Conclusões

Este capítulo apresentou o descritor TIDES, capaz de descrever a oscilação de uma série temporal simples. Para tanto, apresentou uma forma de representar as séries temporais computacionalmente, basendo-se no conceito de segmentação de curvas e utilizando o conceito de representação simbólica. Desta forma, uma série temporal passa a ser representada por um vetor de características no formato $<< y_1, l_1 >, ..., < y_n, l_n >>$, em que y_i é o símbolo que representa a classe associada ao i-ésimo segmento da série. Após isto, foi apresentada uma função de distância específica para o vetor de características construído. Ainda estendemos o descritor, de modo a descrever e comparar séries temporais em mais de uma escala. Isto pode ser utilizado para analisar um mesmo fenômeno em granularidades diferentes.

Capítulo 5

Introduzindo coevolução no TIDES

O capítulo anterior discutiu a necessidade de um descritor que analise a oscilação de uma série temporal, e não apenas seus valores brutos. Esse tipo de análise pode ser fundamental no estudo de alguns fenômenos relacionados a uma variedade de áreas de conhecimento, como por exemplo agricultura e planejamento metropolitano.

Alguns fenômenos, porém, não podem ser caracterizados com a análise de uma única grandeza. Eles necessitam que séries de informações temporais sejam analisadas de maneira conjunta, de modo a estabelecer uma relação entre elas, verificando como a variação de uma das grandezas afeta a variação das demais. Com isso, muitas vezes se torna necessária a realização de várias operações de mineração agrupando todas as grandezas envolvidas. Quando um conjunto de grandezas distintas afeta um fenômeno, de modo que a evolução de uma das grandezas está diretamente relacionada à evolução de outra, dizemos que as grandezas apresentam *coevolução*.

O capítulo anterior representou uma série temporal como uma curva, na qual o eixo X representa o tempo, e o eixo Y a grandeza medida. Esta representação também é válida para o problema de coevolução, no qual cada uma das séries é representada por uma curva isolada. Neste cenário, é impossível estipular uma unidade para o eixo Y, visto que cada grandeza pode utilizar uma unidade distinta.

A Figura 5.1 ilustra um cenário com três curvas simultâneas, cada uma correspondente a uma grandeza diferente. Para realizar uma busca por similaridade neste cenário, deseja-se encontrar conjuntos de três séries (para as mesmas grandezas) que, no mesmo instervalo de tempo, apresentem o mesmo comportamento que as três séries ilustradas na figura.

O princípio básico do descritor é o mesmo, utilizando as mesmas etapas de segmentação, classificação e cálculo de distância. Ao analisar a coevolução de curvas, o alvo do nosso descritor ainda é sua oscilação, e não seus valores.

O TIDES será adaptado de modo que todas as séries sejam representadas por uma única série temporal, composta por classes capazes de representar elementos m-dimensionais.

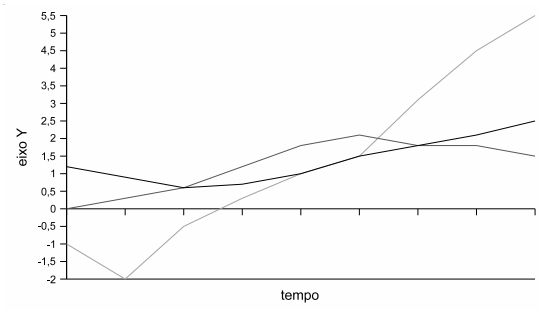


Figura 5.1: Exemplo de três séries, referentes a grandezas diferentes, representadas em um mesmo gráfico.

Para isto, apresentamos uma forma de reduzir o problema da análise de um conjunto de séries utilizando os mesmos princípios apresentados para a análise de uma série isolada. Essa série agrupa, de uma maneira adequada, as informações significativas de cada uma das séries isoladas.

Para realizar tal redução, é necessário adequar as etapas utilizadas pelo descritor.

5.1 Segmentação de múltiplas curvas

Como descrito anteriormente, um fenômeno que apresenta a característica de coevolução é representado por um conjunto de séries, cada uma responsável pela descrição de uma grandeza envolvida no fenômeno. No entanto, quando um conjunto de séries distintas é analisado de maneira conjunta, vários fatores aumentam a complexidade da tarefa. Há diferenças decorrentes da freqüência de coleta de cada uma das séries, bem como da faixa de valores de cada uma das grandezas representada.

Uma das dificuldades encontradas quando analisamos tal cenário é a possível diferença nas taxas de amostragem das séries representadas, bem como o comportamento distinto das séries entre si. Com isso, é possível e esperado que os pontos de união resultantes do processo de segmentação de cada uma das séries sejam distintos entre si.

O processo é realizado da seguinte forma: Sejam $CS =< S_1, S_2, ..., S_n >$ séries para as quais se deseja analisar a coevolução, em um mesmo intervalo de tempo.

1. Inicialmente, cada série deve ser segmentada de maneira isolada. Cada série será representada por n segmentos, onde n é determinado por algum especialista do domínio.

2. Após isso, as séries devem ser normalizadas em conjunto dentre de um mesmo

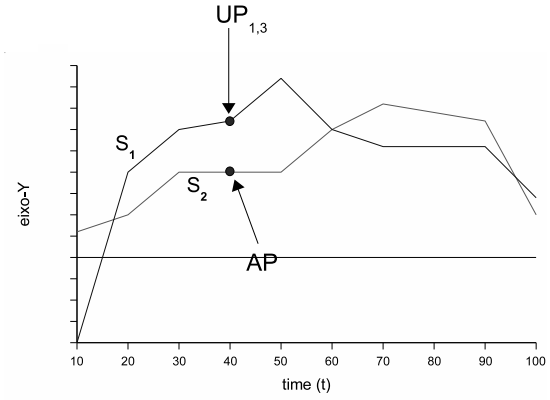


Figura 5.2: Processo de segmentação de várias séries que apresentam coevolução.

intervalo de tempo, por meio da introdução de pontos artificiais de união, como definido na seção 4.4. A diferença, nesse caso, é que os pontos artificiais passam a fazer parte do vetor de características, sendo portanto armazenados. No caso de uma série única (seção 4.4), os pontos eram introduzidos apenas para o cálculo da distância entre duas séries, não sendo portanto armazenados. Para ressaltar essa diferença, o conjunto de pontos de união (artificiais e reais) utilizados para descrever o conjunto de séries que apresentam coevolução (e, assim, armazenados no vetor de características) é chamado de *Conjunto de Instantes de Transição*, ou apenas T_{trans}, e pode ser definido como:

$$T_{trans} = \cup_{i=1}^{i=n} U_i \tag{5.1}$$

onde U_i representa o conjunto de pontos de união (artificiais e reais) da séries S_i.

3. Neste instante, consideramos que o conjunto de séries CS está internamente normalizado. Assim, pode ser submetido às próximas etapas do descritor, compostas pela atribuição de símbolos a cada elemento e pelo cálculo de distância entre objetos diferentes.

A Figura 5.2 ilustra esse processo. Nesta Figura, é possível identificar o ponto $UP_{1,3}$ como sendo um ponto de união presente na série S_1 em um instante de tempo $t_i = 40$ no qual a série S_2 não possui nenhum ponto de união. O ponto AP é o ponto artificial inserido em S_2, no instante $t_i = 40$.

Ao término deste processo de normalização, todas as curvas do conjunto de séries analisado possuirão o mesmo número de segmentos, independente da taxa de amostragem

de cada uma delas. Além disso, o i-ésimo segmento de reta possuirá o mesmo tamanho em todas as séries analisadas.

5.2 Coeficientes Angulares Instantâneos e Classificação de Múltiplas Curvas

Em cada instante de tempo $t_j \in T_{trans}$, é possível construir um vetor $\vec{C}_{t_j}$ que contenha os coeficientes angulares das séries $S_1, S_2, ..., S_n$. Desta forma, $\vec{C}_{t_j}[i]$ representa o coeficiente angular da série i, para um instante de tempo (no caso, t_j). Como um segmento de reta une dois pontos de união, o coeficiente angular de qualquer ponto no seu interior será o mesmo. Assim, o vetor $\vec{C}_{t_j}$ é capaz de caracterizar o coeficiente angular de todas as séries de CS, utilizando a segmentação realizada no passo anterior. A este vetor demos o nome de *CAI (Coeficientes Angulares Instantâneos)*. O *CAI* será usado nos demais passos do processo.

A representação simbólica ainda é interessante, mas no caso de coevolução a função de classificação para atribuir um símbolo a um elemento deve ser sofisticada.

Ao invés de classificar um único coeficiente angular, usaremos a classificação para agrupar um CAI em uma única classe. Isso significa que estaremos representando um vetor m-dimensional de coeficientes angulares por uma única classe, onde m é o número de séries consideradas.

Isso gera uma redução na dimensão do problema analisado, possibilitando o uso de uma técnica similar à apresentada no capítulo 4 para a descrição de um conjunto de séries distintas. Da maneira similar ao processo aplicado a uma única série, cada classe é representada por um símbolo α_i e l_i indica o comprimento da projeção do i-ésimo segmentdo de uma das m séries no eixo dos X, após a normalização. Assim, o vetor de características VC de um conjunto de séries CS pode ser visto como:

$$VC =<< \alpha_1, l_1 >, < \alpha_2, l_2 >, ..., < \alpha_n, l_n >> \tag{5.2}$$

Apesar da definição de VC ser similar à definição de V para uma única série (apresentada na seção 4.3), existe uma grande diferença entre o significado dos dois vetores. Enquanto em V um símbolo y_i representava um coeficiente angular atribuído a um segmento de uma reta, em VC um símbolo α_i representa um elemento m-dimensional. Desta forma, α_i simboliza uma classe que descreve um conjunto de m segmentos de retas, todos no mesmo intervalo de tempo.

A classificação dos coeficientes angulares para o caso de múltiplas séries é realizada em dois passos. No primeiro passo, utilizamos o mesmo princípio aplicado à classificação de uma única série. Cada série S_k de CS é inicialmente analisada de maneira isolada.

Para tanto, em um dado instante de transição t_i, é calculado a qual faixa de ângulos o coeficiente angular de $S_k[t_i]$ pertence. De maneira similar ao que ocorre na classificação de séries simples, a faixa dos possíveis valores para qualquer coeficiente angular (0^o e 180^o, ou 0 e π radianos) é particionada em n_g grupos. Um coeficiente angular é associado a uma classe α_i se ele está contido no intervalo de coeficientes que a dada classe engloba (vide a equação 4.4).

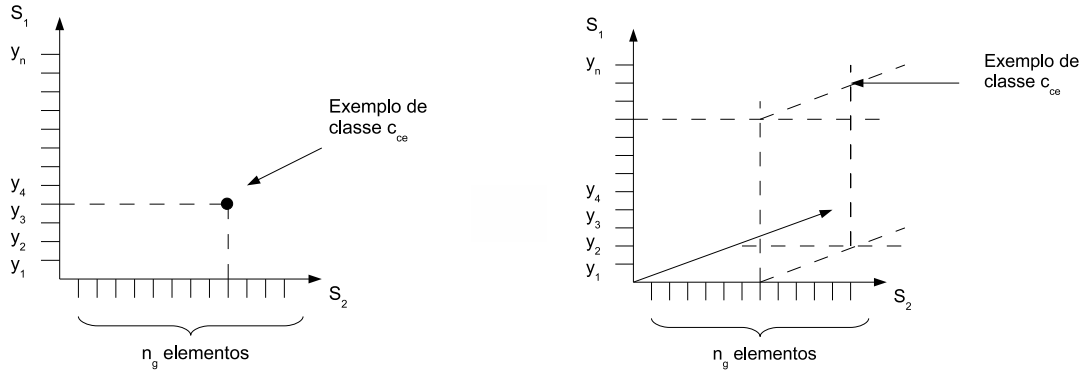
Após esta fase, cada série S_k de CS já está representada como um conjunto de símbolos $<< y_{k,1}, l_1 >, < y_{k,2}, < l_2 >, ..., < y_{k,n}, l_n >>$. Agora, queremos representar o conjunto de séries CS como uma única série de símbolos. Para tanto, é necessário agrupar em uma única classe c_{ce} os m símbolos que representam todas as séries de CS em um dado instante de tempo t_i.

As classes c_{ce} são as responsáveis por realizar a redução de dimensões do problema, ou seja, representar um objeto m-dimensional (o CAI) por meio de um símbolo simples. Cada dimensão de c_{ce} representa a variação possível de ângulos para uma das m séries analisadas. Como inicialmente todas as m séries foram classificadas de maneira isolada, elas já estão representadas por símbolos y_i. Assim, o domínio de cada dimensão de c_{ce} é o conjunto de símbolos gerados no processo de classificação de cada uma das séries. Como todas as m séries são classificadas utilizando o mesmo processo, e o mesmo número de classes (n_g), todas as dimensões de c_{ce} possuem o mesmo domínio.

A Figura 5.3a ilustra um exemplo de classes que descrevem duas séries temporais, enquanto que na Figura 5.3b as classes descrevem três séries. Os eixos das figuras assumem o conjunto de símbolos utilizados para a classificação de cada uma das séries, ainda de maneira isolada. Como descrito na seção 4.3, são utilizados até n_g classes para classificar os segmentos de reta de uma série. Logo, cada eixo das figuras 5.3a e 5.3b pode assumir n_g valores diferentes. Em ambas as figuras, as possíveis classes são pontos em um espaço bi-dimensional (Figura 5.3a) ou tri-dimensional (Figura 5.3b). Elas podem ser vistas como as combinações dos símbolos presentes em cada um dos eixos.

O espaço m-dimensional gerado por todas as possíveis classes capazes de descrever um CAI será chamado, no restante do texto, de *Espaço de conjuntos (ECj)* e depende de duas variáveis: (i) a dimensão de CS (ou seja, o número de séries analisadas – m); (ii) e o número de grupos (n_g) utilizado para classificar uma série de maneira isolada. Isso significa que é um espaço finito, ou seja, cada dimensão é limitada pelo número de grupos n_g utilizado. Mais especificamente, temos que o número total de símbolos que podem ser atribuídos a um CAI é n_g^m.

Da mesma forma que na evolução simples, cada CAI recebe um símbolo que indica a que classe ele pertence. Assim, após o processo de classificação, um conjunto CS pode ser descrito por um descritor Y, como:



(a) Espaço para representar duas séries temporais ao mesmo tempo.

(b) Espaço para representar três séries temporais ao mesmo tempo.

Figura 5.3: Exemplos de espaços m-dimensionais, usados para classificar um CAI. (a) corresponde a duas séries e (b) a três séries.

$$Y =<< \alpha_1, l_1 >, ..., < \alpha_{n'}, l_{n'} >> \tag{5.3}$$

onde $\alpha_i \in ECj$.

Isso nos leva a uma representação muito similar à utilizada na descrição de uma série simples. Assim, o cálculo da distância entre os dois vetores de características Y_1 e Y_2 será similar ao definido para o cálculo de distância entre duas séries simples.

5.3 Função de Distância

Como descrito na seção 4.4, a distância entre dois vetores de características é calculada em dois passos. Inicialmente, é necessário realizar o processo de normalização entre as duas séries. Porém, ao invés de considerar os pontos de união das séries para normalizá-las, serão levados em consideração os instantes de transição definidos para um conjunto de séries CS_k qualquer. Isso significa que, ao normalizar dois CSs distintos, todos os pontos de união de todas as séries envolvidas serão considerados. Esses pontos de união estão representados pelos elementos dos conjuntos T_{trans}.

Da mesma forma que ocorre com uma única série, a normalização dos vetores de características Y_1 e Y_2 gera como resultado NY_1 e NY_2.

Pelo mecanismo de construção dos pontos de união, temos que:

- $|NY_1| = |NY_2|$ Ou seja, o número de símbolos dos dois vetores normalizados é igual. Isso é garantido pelo processo de normalização (visto que $U_{NY1} = U_{NY2}$);

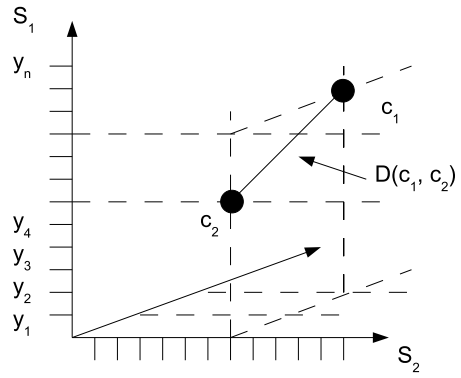


Figura 5.4: Distância entre duas classes de um espaço de três dimensões.

- O comprimento da projeção do i-ésimo símbolo sobre o eixo dos X será igual em ambos os descritores (NY_1 e NY_2) – ou, em outras palavras, $\forall i \in 1, ..., n', l^{=}_{(NY1,i)} l_{(NY2,i)}$.

Essas propriedades nos permitem criar uma função de distância, baseada na métrica Euclidiana, porém com algumas simplificações, como descrito a seguir:

$$D_{CE} = \sum_{i=0}^{i=n} d(< \alpha_i, l_i >_1, < \alpha_i, l_i >_2) \tag{5.4}$$

A aplicação direta das propriedades (1 e 2), no entanto, nos permite simplificar a função de distância para:

$$d(< \alpha i, l_i >_1, < \alpha_i, l_i >_2) = d(\alpha_{i,1}, \alpha_{i,2}) * l_i \tag{5.5}$$

Assim, basta definirmos como calcular a distância entre duas classes. Como descrito na seção 5.2, uma classe ce_i representa um ponto no espaço m-dimensional ECj. Logo, a distância entre duas classes (representadas cada uma por um símbolo em NY_1 e NY_2) será a distância Euclidiana entre os pontos que a representam, no espaço ECj.

Desta forma, a distância $d(\alpha_{i,1}, \alpha_{i,2})$ pode ser definida como:

$$d(\alpha_{i,1}, \alpha_{i,2}) = \sqrt{\sum_{i=0}^{i=m} (ce_{1,i} - ce_{2,i})^2} \tag{5.6}$$

onde $ce_{1,i}$ representa o símbolo associado à i-ésima dimensão da classe presente no primeiro descritor.

A Figura 5.4 ilustra a distância entre duas classes de um espaço de 3 dimensões. Ela pode ser vista como a distância de dois pontos quaisquer em um espaço 3-dimensional.

5.4 Algoritmo

Os princípios básicos utilizados na versão do TIDES que analisa a coevolução de séries são os mesmos que os utilizados para a análise de uma única série. As extensões definidas tem como objetivo descrever um conjunto de séries como uma única séries de símbolos, sendo que cada símbolo representa uma classe $n-$dimensional.

O algoritmo 4 apresenta as extensões aplicadas ao algoritmo de extração de características aplicado a uma única série (algoritmo 1), onde CS é um conjunto de séries.

Algorithm 4 ExtracaoCoevolucao(CS, n_{seg}, n_g)

1: **for** $i = 1$ to $|CS|$ **do**
2: $S_i \leftarrow CS[i]$
3: Segmente a série temporal S_i, de modo a obter um vetor de segmentos de reta Vr_i, com n_{seg} segmentos
4: **for** $j = 1$ to n_{seg} **do**
5: $< \gamma_j, \gamma_{j+1} > \leftarrow Vr_i[j]$;
6: $a_j \leftarrow$ coeficienteAngular($< \gamma_j, \gamma_{j+1} >$);
7: $l_j \leftarrow$ projecaoEmX($< \gamma_j, \gamma_{j+1} >$);
8: $y_j \leftarrow$ classificacaoCoeficiente(a_j, n_g);
9: $V_i[j] \leftarrow < y_j, l_j >$;
10: **end for**
11: **end for**
12: Aplique um processo de normalização envolvendo todos os vetores parciais V_i, gerando assim os vetores normalizados NV_i. {Todos os vetores terão o mesmo tamanho após a normalização}
13: **for** $k = 1$ to $|NV_1|$ **do**
14: $\alpha_k \leftarrow$classificacaoSimbolos($NV_1[k], NV_2[k], ..., NV_n[k]$);
15: $VC[k] \leftarrow < \alpha_k, l_{NV_1} >$; {$VC$ é o vetor de características com coevolução}
16: **end for**
17: RETURN;

Uma vez submetidos ao algoritmo de extração de características (algoritmo 4), dois vetores VC_1 e VC_2 podem ser comparados utilizando a mesma função de distâncias ilustrada pelo algoritmo 3. A única diferença é que, neste caso, a função de distância de símbolos utilizada leva em consideração classes n-dimensionais. Para esta tarefa, utilizamos a distância euclidiana entre as duas classes comparadas, no espaço n-dimensional ao qual elas pertencem.

5.5 Conclusões

Este capítulo estendeu o TIDES para tratar das questões de coevolução de séries. Para isto, foi introduzido o conceito dos vetores CAI. A normalização dentro de um conjunto de séries passa a armazenar todos os pontos de união artificiais. Com isto, um conjunto de séries passou a ser representado por um vetor de características $Y =<< \alpha_1, l_1 >, ..., < \alpha_n, l_n >>$, em que α_i é um símbolo em um espaço m-dimensional. Isto permite usar o mesmo tipo de algoritmos para comparar séries simples e séries em coevolução.

Capítulo 6

Experimentos realizados

De modo a validar os conceitos apresentados e verificar a eficácia do descritor TIDES, realizamos uma série de experimentos utilizando bases de séries que representam fenômenos reais (temperatura) e séries sintéticas. Para tanto, construímos conjuntos de testes que validam cada uma das características que diferenciam o TIDES dos demais descritores na literatura, além de compará-lo a uma abordagem tradicional de busca de similaridade em séries temporais: o Linear Scan.

O restante desta seção está organizada da seguinte maneira: inicialmente, descreveremos as bases de séries utilizadas (seção 6.1. Em seguida descreveremos cada um dos conjuntos de testes realizados, exibindo os resultados obtidos.

6.1 Descrição das bases de séries

Para a realização dos experimentos, foram utilizadas duas bases de séries temporais: uma de séries sintéticas e uma de dados reais. A base de séries sintéticas é disponibilizada em [KXWR06]. Ela é composta por 600 séries, separadas em dois conjuntos: o conjunto de treinamento e o conjunto de testes. Cada uma das séries é composta por 60 pontos, e elas são classificadas em 6 classes distintas. Nosso descritor, no entanto, não necessita de um conjunto de treinamento isolado. Assim, para construir um conjunto de séries maior, unimos os dois conjuntos, construindo uma base com 600 séries sintéticas classificadas.

A base de séries reais contém dados de temperatura referentes aos últimos 30 anos de cinco cidades do estado de São Paulo – Campinas, Jaboticabal, São Carlos, Sorocaba e Taubaté. As séries são referentes a temperaturas máximas e temperaturas médias dessas cidades. Cada ponto representa a média mensal da grandeza medida. Esses dados foram utilizados para construir 1336 séries distintas, cada uma com 48 pontos. Desta forma, as séries representam as variações de temperatura (máxima e média mensal) de uma cidade durante dois anos. Cada série está associada a um mês de início, que corresponde ao mês

representado pelo primeiro ponto da série.

Como a base real foi construída automaticamente, utilizando um grande conjunto de dados previamente coletados, suas séries não foram classificadas por nenhum especialista do domínio. Assim, para classificar uma série s_1 como similar ou não-similar a uma série s_2, foi analisado o mês de início das duas. Caso as duas séries apresentem o mesmo mês de início, ou a diferença entre os meses de início das séries seja de apenas uma unidade, as séries são consideradas similares. Caso contrário, elas são consideradas não-similares.

A hipótese é que a oscilação de temperatura no decorrer do ano seja similar entre diferentes cidades e diferentes anos. Além disto, algumas vezes a temperatura média oscila de maneira similar à temperatura máxima de um mesmo período.

6.2 Invariância a translações no eixo Y

Este conjunto de testes utilizou as séries sintéticas. Seu objetivo foi validar a característica de invarância a translações no eixo Y. Para cada série S_i da base de séries sintéticas, foi inserida uma série ruído S'_i que representa a própria série S_i deslocada de um valor aleatório δ no eixo Y. Os pontos de S'_i foram construídos da seguinte forma:

$$S'_i[j] \leftarrow S_i[j] + \delta, \forall j \in \{1, ..., n\} \tag{6.1}$$

Desta forma, a oscilação de uma série qualquer S'_i é sempre igual à oscilação da série original S_i. O objetivo do teste foi validar que as séries S_i e S'_i são consideradas similares pelo descritor.

Para tanto, cada série S_i foi utilizada como série de consulta, e era esperado que em todos os casos, a série S'_i fosse considerada a mais similar, com $D(S_i, S'_i) = 0$. As 600 consultas obtiveram o resultado esperado, validando a invariância do descritor a deslocamentos no eixo Y. Testes realizados com as séries reais de temperatura também capturaram a invariância a deslocamentos no eixo Y.

6.3 Testes utilizando análise Multi-Escala

Este conjunto de testes visa a validar visualmente a análise multi-escala realizada pelo descritor TIDES. Para tanto, várias consultas foram realizadas utilizando cinco escalas diferentes. Em seguida, foram selecionados para análise visual os casos nos quais a distância entre as duas séries analisadas dependia muito da escala utilizada. O objetivo destes experimentos foi validar como o uso de escalas distintas pode afetar a análise de similaridade de séries, quando estamos interessados em suas oscilações. Esta seção apresenta um caso no qual foi feita análise visual. Os testes utilizaram a base de séries sintéticas.

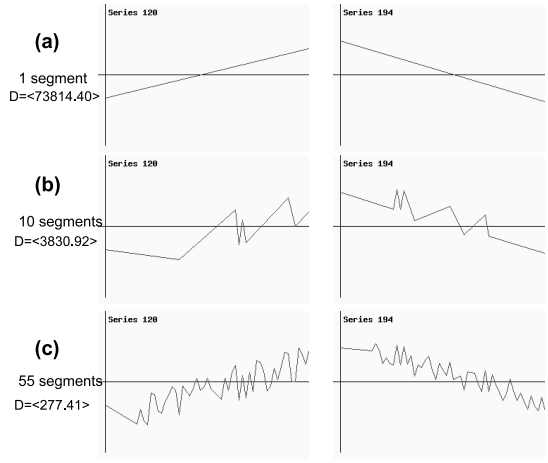


Figura 6.1: Comparação multi-escala entre as séries 120 e 194 da base de séries sintéticas, utilizando três escalas distintas: 1, 10 e 55 segmentos.

Os experimentos mostraram que a análise multi-escala é capaz de detectar relações escondidas em conjuntos de dados conhecido; duas séries consideradas completamente distintas podem se mostrar similares dependendo da escala.

A Figura 6.1 ilustra o caso da análise multi-escala das séries 120 e 194 da base sintética, utilizando 1 segmento (NV_1), 10 segmentos (NV_{10}) e 55 segmentos (NV_{55}), respectivamente as Figuras 6.1(a), 6.1(b) e 6.1(c). Na coluna esquerda aparece o valor de distância entre as séries obtido para cada escala; $73814,4$ quando computado para 1 segmento e $277,4$ quando computado para 55 segmentos.

A Figura 6.2 mostra a variação na distância entre as séries 120 e 194 de acordo com o número de segmentos utilizados na análise. Esta curva mostra que, quanto menor o número de segmentos utilizados para representar as séries, mais distintas elas se tornam. Porém, ao aumentar a quantidade de segmentos, as características locais começam a se fazer sentir. Para um limiar de similaridade utilizando o valor de distância de 300, por exemplo, as séries 120 e 194 são 1-não similares, 10-não similares mas 55-similares.

A Figura 4.7 ilustra a análise realizada pelo TIDES, para 55 segmentos. Ela representa as seqüências de coeficientes angulares das duas séries. É importante notar que seus padrões de oscilação são bastante similares, quando utilizamos uma quantidade de segmentos maior.

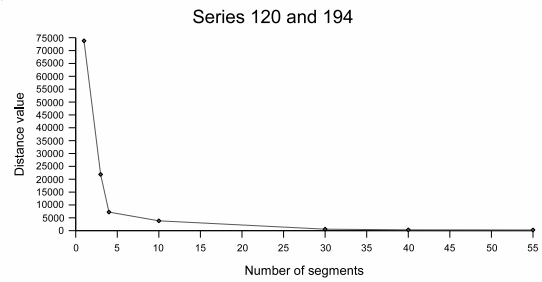


Figura 6.2: Curva de distância entre as séries 120 e 194 em relação ao número de segmentos utilizados para descrevê-las.

6.4 Comparação do descritor com o Linear Scan

As seções anteriores mostram algumas características do TIDES, como a invariância a deslocamentos no eixo Y e a função multi-escala; Esta seção compara sua eficácia com a de alguns descritores existentes. Escolhemos o Linear Scan o conjunto de experimentos. Esta opção de escolha é porque trata-se de um descritor apropriado, bastante utilizado como base na comunidade, já que nenhum dos descritores na literatura, até onde conhecemos, considera a oscilação de séries.

O TIDES caracteriza a similaridade da oscilação de uma série. Assim, algumas bases comumente usadas para comparação de descritores são inadequadas, pois elas são previamente classificadas levando em consideração a similaridade dos seus pontos, e não de suas oscilações. Desta forma, descritores que se baseiam nos valores das séries tendem a permitir análises mais condizentes com as realizadas na fase de classificação.

Utilizamos a base de dados de temperaturas descrita na seção 6.1, que apresenta algumas características interessantes de oscilação. Em primeiro lugar, as temperaturas variam de maneira similar em uma mesma estação do ano para uma mesma região, independente da cidade analisada. Além disso, uma mesma cidade pode apresentar, um determinado padrão de evolução de temperatura, que se repete a cada ano – em variação de valores e em oscilações. Como nossa base é composta por dois tipos de temperaturas (médias e máximas), é ainda comum observarmos uma oscilação similar entre a temperatura máxima e a temperatura média de uma região, em um mesmo período de tempo. Isso nos leva a uma busca por similaridade em séries que podem apresentar deslocamento dos valores no eixo Y.

Com base nisso, realizamos dois tipos de testes de comparação entre os dois descritores. A série de entrada, para uma consulta, é alguma série de valores de temperatura de uma das cinco cidades analisadas:

- O primeiro tipo de testes usa toda a base para calcular a eficácia da consulta.
- O segunto tipo de testes considera apenas os resultados referentes a cidades distintas da cidade de consulta para calcular a eficácia da consulta.

Em ambos os tipos de testes, o critério de classificação de séries adotado foi o descrito na seção 6.1, que considera duas séries *similares* se foram iniciadas no mesmo mês ou em meses vizinhos. É importante notar que este critério de análise apenas classifica uma série como um *acerto* ou um *erro*. Assim, a partir de uma busca por k vizinhos mais próximos, um descritor será avaliado a partir do número de acertos entre as k primeiras séries retornadas.

Para a execução dos testes, todas as séries foram utilizadas como séries de consulta. Em cada consulta realizada, foi computado o número de acertos de cada um dos descritores. Após o término dos testes, a taxa de acertos final foi utilizada para analisar a eficácia de cada um dos descritores.

O primeiro tipo de testes parte da hipótese que o padrão de oscilação da variação de temperaturas é, em geral, intrínseco às características geográficas de uma região. A hipótese é que uma mesma região, em anos distintos, tende a apresentar valores de temperaturas similares para os mesmos meses. Isso faz com que, mesmo considerando apenas os valores de uma série, e não sua oscilação, um descritor seja capaz de recuperar séries iniciadas no mesmo mês, para anos distintos. Desta forma, a eficácia de um descritor baseada em valores tende a aumentar neste tipo de testes.

O segundo tipo de testes, por sua vez, diminui a probabilidade de séries com a mesma distribuição de valores serem retornadas como semelhantes à série de entrada, uma vez que as séries daquela cidade não são consideradas na resposta. Aqui, a oscilação passa a ser mais significativa. Este tipo de testes evidencia a eficácia do TIDES.

Para exemplificar a situação, sabemos que a cidade de São Carlos apresenta normalmente temperaturas mais elevadas que a cidade de Campinas, devido às suas características geográficas. Porém, em um dado verão, ambas as cidades podem apresentar padrões de oscilação de temperatura similares, apesar de possuirem valores brutos de temperaturas distintos. Técnicas que analisam os pontos das séries para realizar as comparações, como por exemplo o *Linear Scan*, tendem a considerar todas as séries de Campinas mais similares entre si, do que em relação a séries de São Carlos. O TIDES, por sua vez, é capaz de analisar a oscilação das séries, e assim eventualmente considerar duas séries de cidades distintas (Campinas e São Carlos, por exemplo) mais similares que duas séries de uma mesma cidade (Campinas, por exemplo) em anos diferentes.

Os testes foram realizados por meio de buscas kNN (k vizinhos mais próximos), para os valores de $k \in \{30, 50, 70, 100\}$. É importante notar que, quanto maior o número de k, menor a chance de se encontrar séries com a mesma distribuição de pontos da série

Tabela 6.1: Comparação entre TIDES e Linear Scan.

k	TESTE 1			TESTE 2		
	Lin.Scan	TIDES	Ganho Perc.	Lin.Scan	TIDES	Ganho Perc.
30	90	81(-9%)	(-11%)	78	77(-1%)	(-1%)
50	85	78(-7%)	(-9%)	73	74(1%)	1%
70	78	76(-2%)	(-3%)	69	72(3%)	4%
100	70	73(3%)	4%	62	69(7%)	10%

de consulta. Porém, a oscilação das séries ainda será similar para aquelas pertencentes a períodos equivalentes em diferentes anos.

Os resultados dos testes realizados foram resumidos na Tabela 6.1. As colunas *TESTE 1* e *TESTE 2* representam respectivamente o percentual de acertos dos dois descritores para os testes do primeiro e do segundo tipos. A taxa de acertos a pode ser calculada segundo a fórmula 6.2, onde NSS é o número de séries similares retornadas pelo descritor.

$$a = \frac{NSS}{k} * 100 \tag{6.2}$$

A Tabela 6.1 mostra que TIDES é mais apropriado que o Linear Scan para caracterizar oscilações. Na coluna TIDES, o valor percentual apresentado entre parênteses representa quanto a eficácia de TIDES é melhor em relação ao Linear Scan. É possível observar que, nos testes do tipo 2, que caracterizam mais a oscilação das séries, TIDES apresenta uma eficácia melhor que o Linear Scan para um número relativamente baixo de k (a partir de $k = 50$). Quanto maior o número de séries analisadas, melhor o desempenho de TIDES (por exemplo, veja $k = 100$). A coluna de ganho percentual foi preenchida de acordo com o percentual de ganho ou perda do TIDES em relação ao Linear Scan. Ela foi calculada de acordo com a equação 6.3.

$$comp_{perc} = \frac{Acertos_{LinScan} * 100}{Acertos_{TIDES}} \tag{6.3}$$

onde $comp_{perc}$ assume um valor menor que 100 quando a eficácia do TIDES é maior, e maior que 100 quando a eficácia do Linear Scan é maior.

Para os testes do tipo 1, a eficácia de TIDES supera a eficácia de Linear Scan apenas para valores maiores de k. À medida em que o número de séries analisadas aumenta, a oscilação passa a exercer maior significado na análise das séries.

6.5 Protótipo

Nesta seção descrevemos brevemente um protótipo construído para realizar buscas utilizando o descritor TIDES [1]. O protótipo foi desenvolvido para possibilitar a análise visual do resultado de uma busca qualquer, auxiliando assim na validação do descritor. Ele realiza uma busca do tipo kNN, com $k = 10$. Desta forma, a partir de uma série de consulta fornecida pelo usuário, o protótipo executa o algoritmo do TIDES, obtendo as 10 séries mais similares. Após isso, exibe graficamente cada uma das séries obtidas.

O protótipo é composto por dois blocos principais: a camada de apresentação e a camada de busca. A camada de apresentação é composta por um conjunto de scripts, escritos em linguagem *PHP*, para apresentar graficamente uma série temporal. Utiliza o eixo X para representar o tempo, e o eixo Y para representar os valores.

A camada de busca contém os algoritmos para criar o descritor. Essa camada está escrita em linguagem *C++*, e pode ser utilizada como um aplicativo isolado, ou seja, independente da camada de apresentação. As buscas estão utilizando a base de séries de temperaturas, descrita na seção 6.1.

As duas camadas se comunicam por meio de um mecanismo de persistência em memória secundária. Ao receber uma consulta, a camada de apresentação invoca a camada de buscas. Essa, por sua vez, executa a busca e gera os resultados em arquivos, utilizando um formato de codificação proprietário. A camada de apresentação decodifica os resultados, gerando a representação gráfica das séries selecionadas no resultado da consulta.

A camada de apresentação ainda está muito simplificada, não permitindo ao usuário selecionar parâmetros para a camada de buscas. Desta forma, as comparações usando TIDES são executadas utilizando os valores padrão para seus parâmetros de configuração. Dentre os parâmetros, podemos citar o número de escalas, o número de classes (símbolos) utilizado na fase de classificação e o número de segmentos utilizado na fase de segmentação.

A Figura 6.3 ilustra a tela de apresentação de resultados de uma consulta para o usuário, tendo como entrada a série 120. Os resultados são exibidos no formato de *grade*, sendo que a série mais similar à série de consulta está no extremo superior esquerdo. A ordem de similaridade está expressa da direita para a esquerda, de cima para baixo e, como esperado, a série 120 é a mais similar a si própria. Para cada série presente no resultado, são exibidos sua cidade e seu mês de início.

A tela de resultados ainda possui um apontador para uma página que exibe as mensagens geradas pelo descritor durante o processamento da consulta. Essas mensagens podem ser utilizadas para depuração do algoritmo, bem como para uma melhor compreensão do resultado obtido.

[1] Este protótipo se encontra disponível para consultas no endereço: *www.lis.ic.unicamp.br/ lmariote/teste_series.php*

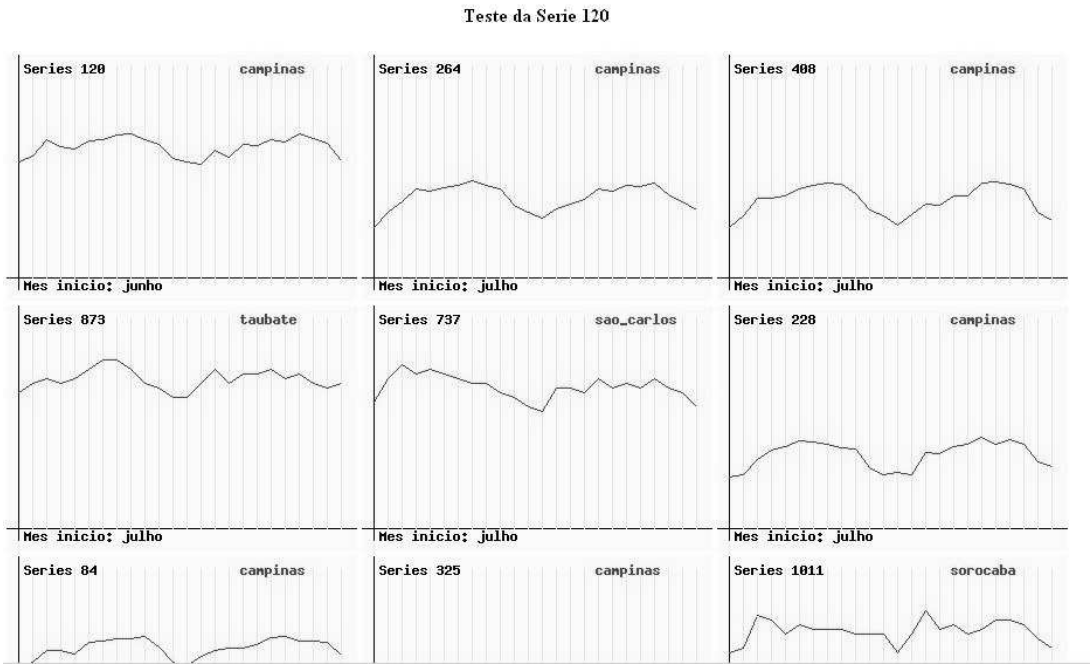


Figura 6.3: Tela do protótipo exibindo os resultados da busca a partir da série 120.

O conjunto de ferramentas utilizado na implementação da camada de apresentação permite que o protótipo seja utilizado através de um *Web-Browser* padrão. Atualmente, o protótipo se encontra disponível para acessos a partir de qualquer local da internet.

6.6 Resumo

Este capítulo apresentou alguns experimentos executados com dados sintéticos e reais usando o TIDES, para séries isoladas. Os resultados mostram que o descritor é imune a variações no eixo Y e que apresenta resultados melhores que o *Linear Scan* quando se aumenta o número de segmentos analisados. O descritor ainda foi submetido a um conjunto de testes para medir sua eficácia. Foram utilizadas séries que descrevem as temperaturas máxima e média de cinco cidades do estado de São Paulo, no decorrer dos últimos trinta anos. Cada ponto representa o valor médio da dada temperatura (máxima ou média) durante um mês em uma das cidades. As séries foram compostas por 48 pontos, representando 2 anos. Para esses testes, foi utilizada uma versão mono-escala do descritor, representando as séries com 47 símbolos (cada um referente a um segmento de reta). Os resultados do TIDES foram comparados aos resultados apresentados por um descritor bastante comum, o *Linear Scan*. Os testes mostraram que o TIDES descreve melhor a

oscilação de uma série temporal.

Capítulo 7

Conclusões e Trabalhos futuros

7.1 Conclusões

Esta dissertação apresentou um novo descritor para séries temporais – o TIDES – que propõe uma nova abordagem para o problema de busca por similaridade em séries temporais. Ele apresenta três grandes contribuições em relação aos demais:

- Descrição da oscilação de uma série temporal, e não seus valores propriamente ditos;
- Análise multi-escala de séries temporais;
- Análise de co-evolução de um conjunto de séries;

O principal diferencial do TIDES em relação aos demais descritores é o fato de caracterizar a oscilação de séries temporais. Os demais descritores, até onde conhecemos, caracterizam uma série temporal analisando apenas seus valores, e não como esses valores oscilam. Com isso, permitimos novas análises em séries temporais.

Ao analisar a oscilação de uma série, é possível detectar padrões de oscilação, bem como tendências nas séries. Essa abordagem se mostra mais interessante na análise de alguns problemas relacionados a diversas das áreas, como economia e agricultura. Além disso, ela resolve alguns problemas vinculados à recuperação de séries temporais, dentre os quais podemos citar a imunidade a translações no eixo dos valores (eixo y).

Para tanto, o descritor transforma uma série temporal em uma seqüência de segmentos lineares, utilizando técnicas definidas na literatura. Em seguida coleta os coeficientes angulares de cada um dos segmentos. Esses coeficientes são capazes de representar a oscilação da série em um dado instante de tempo. O vetor de características é gerado a partir de uma representação simbólica dos coeficientes angulares obtidos, o que permite a utilização de funções de distância bastante conhecidas na literatura.

A análise multi-escala de um conjunto de séries temporais permite que elas sejam comparadas utilizando mais de uma granularidade. Essa flexibilidade faz com que o descritor se adeque a vários domínios de aplicação distintos. Além disso, é possível estabelecer novas relações de similaridade entre duas séries, visto que elas podem ser similares quando as analisamos com uma granularidade, porém distintas quando as analisamos com uma granularidade diferente.

A última contribuição diz respeito à análise conjunta de séries que apresentam a característica de co-evolução. Se pensarmos que cada série representa uma grandeza física diferente, isso significa que a oscilação de uma série está diretamente relacionada à oscilação de outras séries. Essa abordagem permite o estudo de fenômenos mais complexos, que para serem corretamente caracterizados, necessitam de uma análise de várias grandezas distintas. Como exemplos de grandezas relacionadas, podemos citar a taxa de pluviosidade, a temperatura e a humidade de uma região. Vários fenômenos necessitam da análise conjunta dessas grandezas, como por exemplo a predição de colheitas de uma dada cultura em uma região.

O fenômeno de co-evolução entre séries tem sido pouco explorado, e as propostas apresentadas estão mais voltadas para a predição de valores e detecção de outliers. Até onde sabemos, nenhuma técnica foi proposta para busca por similaridade que combine tipos de séries distintos.

Os testes realizados foram capazes de averiguar a imunidade do descritor em relação a deslocamentos no eixo dos valores (eixo y). Por meio da inserção de séries artificiais, pudemos verificar que o descritor foi capaz de identificar as séries com oscilações similares, mesmo quando elas se encontravam deslocadas entre si.

Além disso, foram realizados testes com a versão multi-escala do descritor. Esses testes foram capazes de capturar relações entre séries, quando submetidas a analises utilizando escalas diferentes, demonstrando que a escala utilizada afeta muito a similaridade entre séries distintas.

Durante o desenvolvimento da dissertação, pudemos estabelecer um paralelo entre as áreas de recuperação de imagens por conteúdo e mineração de séries de dados temporais, analisando a operação de busca por similaridade em séries temporais por meio de conceitos definidos para recuperação de imagens. A relação entre essas duas áreas torna-se bastante interessante, visto que ambas apresentam recentemente muitos resultados, e podem ser facilmente combinadas.

Outro resultado do projeto foi a construção do protótipo de um sistema de busca por similaridade para séries temporais. Nele, o usuário pode, de maneira gráfica e por meio de ferramentas WEB, realizar buscas de séries que mais se assemelham com uma série de consulta. O sistema realiza as buscas em um repositório de séries, que atualmente contempla as séries de temperaturas. Além de representar uma ferramenta importante para

buscas de séries, esse protótipo foi utilizado durante os testes de validação do descritor.

7.2 Trabalhos Futuros

Há várias extensões para a dissertação. A primeira consiste em aplicar a versão multi-escala do descritor em problemas de análises econômicas de tendências de mercado. Esse domínio apresenta uma base de dados de séries reais bastante rica em informações, com um grande volume de dados. Tal conjunto de testes deve auxiliar o processo de validação do descritor em problemas práticos. Além disso, o contato com especialistas do domínio pode identificar estensões para o descritor, mais voltadas para a área econômica.

Outra extensão seria construir uma base de dados composta por séries capazes de descrever o fenômeno de co-evolução. Para tanto, as séries devem representar grandezas distintas, coletadas em um mesmo período de tempo. Essas séries devem evoluir de maneira relacionada.

É necessário também realizar testes para validar o TIDES para a co-evolução de séries temporais. Para tanto, deve ser utilizada uma base de séries que descrevam várias grandezas relacionadas, como a discutida no parágrafo anterior. Como o volume de trabalhos que atuam na área de co-evolução de séries ainda é pequeno, não será possível validar o descritor por comparação com outros descritores.

No entanto, é possível validar a eficácia do TIDES utilizando outras estratégias, como por exemplo a validação de predições. Para tanto, basta submeter o descritor a séries históricas de fenômenos conhecidos e coletar as predições de valores geradas. Essas predições podem ser comparadas com os valores realmente obtidos. A eficácia do descritor estará diretamente relacionada à taxa de acertos obtida.

Outro ponto em aberto requer criar um mecanismo de indexação de séries para o problema de co-evolução. Apesar de vários resultados terem sido apresentados para a indexação de séries temporais simples, nada se conhece a respeito de indexação de buscas que relacionam mais de uma série.

O uso de representação simbólica no vetor de características utilizado pelo TIDES é um indício de que alguma técnica de indexação para séries simples que também utilizam representação simbólica pode ser adaptada para a versão com co-evolução do descritor. Porém, são necessários mais estudos para definir e validar tal adaptação, verificando sua eficiência e sua eficácia.

Quando analisamos um fenômeno que apresenta co-evolução entre grandezas distintas, sabemos que a oscilação de uma série gera uma oscilação em outra. Em alguns casos, mesmo que duas grandezas sejam relacionadas, esse reflexo não ocorre instantaneamente. Isso significa que uma alteração em uma das séries somente será refletida na outra série após um certo período de tempo. Essa latência ocorre pois os efeitos da grandeza não são

imediatos.

Um exemplo disso é a relação entre a precipitação de chuvas em uma região e os níveis de colheita obtidos na mesma região. Períodos de estiagem surtirão efeitos negativos apenas no próximo ciclo de colheita. Desta forma, uma extensão importante para o TIDES é possibilitar a análise relacionada de várias séries, porém com uma defasagem no tempo entre elas. Para tanto, é necessário que o descritor detecte como deve agrupar o conjunto das m séries relacionadas, de modo que a função de distância seja capaz de computar a similaridade entre conjuntos distintos.

A necessidade de definir o parâmetro de entrada n_g dificulta o uso do nosso descritor, uma vez que obriga os especialistas do domínio a conhecer alguns conceitos do descritor. Uma possível extensão diz respeito a definir técnicas que, através de conjuntos de treinamento, detectem o valor ideal para n_g em cada domínio. Além disso, é interessante possibilitar que os grupos gerados possuam tamanhos diferentes, e com isso concentrar mais grupos nas faixa de oscilação de curva nas quais se encontram uma maior quantidade de segmentos. Com isso, podemos tornar o descritor mais sensível aos domínios, sem ter que utilizar um número muito elevado para n_g.

As extensões aqui propostas para a classificação de séries com coevolução apresentam uma explosão no número de classes geradas, pois são geradas todas as possíveis combinações entre os grupos de cada uma das séries analisadas. Porém, é esperado que o número de combinações de fato utilizadas em um conjunto de séries seja muito menor que o gerado pela nossa técnica. Dessa forma, uma extensão interessante diz respeito ao estudo de uma função de classificação mais eficaz para o problema de coevolução, que crie um número de classes mais gerenciável.

Finalmente, outro ponto em aberto é introduzir melhorias no protótipo para buscas de séries temporais. Dentre elas está a flexibilização da ferramenta, permitindo ao usuário escolher o número de escalas a ser utilizado, o número de classes utilizadas no processo de classificação dos coeficientes angulares e o número de segmentos gerados no processo de normalização. Além disso, o protótipo deve ser adaptado para trabalhar com a coevolução de séries. Para tanto, deve permitir a entrada de múltiplas séries como séries de consultas, e ainda representar todas as séries envolvidas na busca quando os resultados forem exibidos.

Referências Bibliográficas

[ALSS95] R. Agrawal, K.-I. Lin, H. S. Sawhney, and K. Shim. Fast similarity search in the presence of noise, scaling, and translation in time-series databases. In *Twenty-First International Conference on Very Large Data Bases*, pages 490–501, Zurich, Switzerland, 1995. Morgan Kaufmann.

[BBD+03] B. Babcock, S. Babu, M. Datar, R. Motwani, and J. Widom. Models and issues in data stream systems. Technical report, Department of Computer Science, Stanford University, 2003.

[Bot95] M. Botelho. Incorporação de facilidades espaço temporais em um banco de dados orientado a objetos. Master's thesis, UNICAMP, 1995.

[CCC+02] D. Carney, U. Cetintemel, M. Cherniack, C. Convey, S. Lee, G. Seidman, M. Stonebraker, N. Tatbul, and S. Zdonik. Monitoring streams : A new class of data management applications. Technical report, Brown Computer Science, 2002.

[EGSV98] M. Erwig, R. H. Giiting, M. Schneider, and M. Vazirgiannis. Abstract and discrete modeling of spatio-temporal data types. In *ACM GIS 98*, 1998.

[Fal02] C. Faloutsos. Tutorial: Sensor data mining: Similarity search and pattern analysis. In *28th International Conference on Very Large Data Bases*, Hong Kong, China, August 2002.

[FRM94] C. Faloutsos, M. Ranganathan, and Y. Manolopoulos. Fast subsequence matching in time-series databases. In *Proceedings 1994 ACM SIGMOD Conference, Mineapolis, MN*, pages 419–429, 1994.

[FT07] C. D. Ferreira and R. S. Torres. Image retrieval with relevance feedback based on genetic programming. Master's thesis, UNICAMP, August 2007.

[GzM03] L. Golab and A zsu M. Issues in data stream management. In *ACM SIGMOD*, volume 32, pages 5 – 14, June 2003.

[HK02] J. Han and M. Kamber. Data mining: Concepts and techniques. In *ACM SIGMOD*, volume 31, June 2002.

[HPMA+00] J. Han, J. Pei, B. Mortazavi-Asl, Q. Chen, U. Dayal, and M. Hsu. Freespan: frequent pattern-projected sequential pattern mining. In *KDD '00: Proceedings of the sixth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 355–359, New York, NY, USA, 2000. ACM Press.

[JS96] C. S. Jensen and R. T. Snodgrass. Semantics of time-varying information. *Information Systems*, 21:311 – 352, March 1996.

[JS97a] C. S. Jensen and R. T. Snodgrass. Semantics of time-varying attributes and their use for temporal database design. Technical report, TimeCenter, 1997.

[JS97b] C. S. Jensen and R. T. Snodgrass. Temporal data management. Technical report, TIMECENTER, June 1997.

[JY07] L. Junkui and W. Yuanzhen. Advances in data and web management. In *Advances in Data and Web Management*, volume 4505, pages 554–565. Springer Berlin / Heidelberg, 2007.

[KCHP01] E. J. Keogh, S. Chu, D. Hart, and M. J. Pazzani. An online algorithm for segmenting time series. In *ICDM '01: Proceedings of the 2001 IEEE International Conference on Data Mining*, pages 289–296, Washington, DC, USA, 2001. IEEE Computer Society.

[KCPM01] E. Keogh, K. Chakrabarti, M. Pazzani, and S. Mehrotra. Locally adaptive dimensionality reduction for indexing large time series databases. In *SIGMOD '01: Proceedings of the 2001 ACM SIGMOD international conference on Management of data*, pages 151–162, New York, NY, USA, 2001. ACM Press.

[KP98] E. Keogh and M. Pazzani. An enhanced representation of time series which allows fast and accurate classification, clustering and relevance feedback. In R. Agrawal, P. Stolorz, and G. Piatetsky-Shapiro, editors, *Fourth International Conference on Knowledge Discovery and Data Mining (KDD'98)*, pages 239–241, New York City, NY, 1998. ACM Press.

[KP00] E. J. Keogh and M. J. Pazzani. A simple dimensionality reduction technique for fast similarity search in large time series databases. In T. Terano, H. Liu, and A. Chen, editors, *Knowledge Discovery and Data Mining, Current Issues*

and New Applications, 4th Pacific-Asia Conference, PAKDD 2000, volume 1805, pages 122–133, Kyoto, Japan, 2000. Springer.

[KR05] E. Keogh and C. A. Ratanamahatana. Exact indexing of dynamic time warping. *Knowl. Inf. Syst.*, 7(3):358–386, 2005.

[KS97] E. Keogh and P. Smyth. A probabilistic approach to fast pattern matching in time series databases. In D. Heckerman, H. Mannila, D. Pregibon, and R. Uthurusamy, editors, *Third International Conference on Knowledge Discovery and Data Mining*, pages 24–30, Newport Beach, CA, USA, 1997. AAAI Press, Menlo Park, California.

[KXWR06] E. Keogh, X. Xi, L. Wei, and C. A. Ratanamahatana. The ucr time series classification/clustering homepage: www.cs.ucr.edu/ eamonn/timeseriesdata/, 2006.

[LKLC03] J. Lin, E. Keogh, S. Lonardi, and B. Chiu. A symbolic representation of time series, with implications for streaming algorithms. In *DMKD '03: Proceedings of the 8th ACM SIGMOD workshop on Research issues in data mining and knowledge discovery*, pages 2–11, New York, NY, USA, 2003. ACM Press.

[MMT07] L. Mariote, C. Medeiros, and R. Torres. Diagnosing similarity of oscillation trends in time series. In *Spatial and Spatio-Temporal Data Mining - 17' ICDM*. IEEE, October 2007.

[OVY01] B. S. Manjunathand J. R. Ohm, V. V. Vasudevan, and A. Yamada. Color and texture descriptors. *Circuits and Systems for Video Technology, IEEE Transactions on*, 11(6):703–715, Jun 2001.

[PLC99] S. Park, D. Lee, and W. W. Chu. Fast retrieval of similar subsequences in long sequence databases. In *KDEX '99: Proceedings of the 1999 Workshop on Knowledge and Data Engineering Exchange*, page 60, Washington, DC, USA, 1999. IEEE Computer Society.

[PSZ99] C. Parent, S. Spaccapietra, and E. ZimÃ¡nyi. Spatio-temporal conceptual models: Data structures + space + time. In *Seventh ACM international symposium on Advances in geographic information systems*, pages 26–33, November 1999.

[RM97] D. Rafiei and A. Mendelzon. Similarity-based queries for time series data. In *SIGMOD '97: Proceedings of the 1997 ACM SIGMOD international conference on Management of data*, pages 13–25, New York, NY, USA, 1997. ACM Press.

[SGCD06] L. Shou, G. Gao, G. Chen, and J. Dong. Classifying motion time series using neural networks. In *Advances in Multimedia Information Processing - PCM 2006*, volume 4261, pages 606–614. Springer Berlin / Heidelberg, 2006.

[SK86] A. Shoshani and K. Kawagoe. Temporal data management. In *Twelfth International Conference on Very Large Data Bases table of contents*, pages 79 – 88, 1986.

[SWS+00] A. W. M. Smeulders, M. Worring, S. Santini, A. Gupta, and R. Jain. Content-based image retrieval at the end of the early years. 22:1349–1380, December 2000.

[TFC02] R. S. Torres, A. X. Falcao, and L. F. Costa. Shape description by image foresting transform. *Digital Signal Processing, 2002. DSP 2002. 2002 14th International Conference on*, 2:1089–1092 vol.2, 2002.

[TGNO92] D. Terry, D. Goldberg, D. Nichols, and Oki. Continuous queries over append-only databases. In *ACM SIGMOD*, pages 321 – 330, June 1992.

[Tru97] A. Trudel. A temporal knowledge representation approach based on elementary calculus. *Computational Intelligence*, 13:465 – 487, 1997.

[TSMR03] R. S. Torres, C. G. Silva, C. B. Medeiros, and H. V. Rocha. Visual structures for image browsing. In *CIKM '03: Proceedings of the twelfth international conference on Information and knowledge management*, pages 49–55, New York, NY, USA, 2003. ACM.

[WSS+05] H. Wu, B. Salzberg, G. C. Sharp, S. B. Jiang, H. Shirato, and D. Kaeli. Subsequence matching on structured time series data. In *SIGMOD '05: Proceedings of the 2005 ACM SIGMOD international conference on Management of data*, pages 682–693, New York, NY, USA, 2005. ACM Press.

[Yi04] B. Yi. Um modelo de dados para objetos móveis. Master's thesis, UNICAMP, 2004.

[YKT05] R. Yang, P. Kalnis, and A. K. H. Tung. Similarity evaluation on tree-structured data. In *SIGMOD '05: Proceedings of the 2005 ACM SIGMOD*

international conference on Management of data, pages 754–765, New York, NY, USA, 2005. ACM Press.

[YSJ+00] B. Yi, N. D. Sidiropoulos, T. Johnson, H. V. Jagadish, C. Faloutsos, and A. Biliris. Online data mining for co-evolving time sequences. In *ICDE '00: Proceedings of the 16th International Conference on Data Engineering*, page 13, Washington, DC, USA, 2000. IEEE Computer Society.