UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
INSTITUTO DE INFORMÁTICA
CURSO DE BACHARELADO EM CIÊNCIA DA COMPUTAÇÃO

JONAS JESKE

Similaridade de Series Temporais na Bolsa de Valores

Trabalho de Conclusão apresentado como requisito parcial para a obtenção do grau de Bacharel em Ciência da Computação

Prof. Dr. Carlos Alberto Heuser
Orientador

Porto Alegre, junho de 2011

CIP – CATALOGAÇÃO NA PUBLICAÇÃO

Jeske, Jonas

Similaridade de Series Temporais na Bolsa de Valores / Jonas Jeske. – Porto Alegre: PPGC da UFRGS, 2011.

31 f.: il.

Trabalho de Conclusão (graduação) – Universidade Federal do Rio Grande do Sul. Curso de Bacharelado em Ciência da Computação, Porto Alegre, BR–RS, 2011. Orientador: Carlos Alberto Heuser.

1. Series Temporais. 2. Dynamic Time Warping. 3. Banco de Dados. 4. Bolsa de Valores. I. Heuser, Carlos Alberto. II. Título.

“You never achieve success unless you like what you are doing.”
— Dale Carnegie

AGRADECIMENTOS

Primeiramente agradaço a Deus pela minha vida, aos meus pais Jorge e Elisa pelo amor, suporte e por sempre terem acreditado no filho e a minha irmã Éllen pelo carinho.

Agradeço o Prof. Carlos Alberto Heuser pela idéia e orientação e a Universidade Federal do Rio Grande do Sul. Também Agradeço a Technische Universität Kaiserslautern pela oportunindade de intercâmbio prestada em 2009.

Aos colegas pelo companheirismo e, em especial, aos colegas intercambistas da turma de 2009 TU-KL.

Agradeço aos professores Renata Galante e Leandro Krug Wives pelas correções e sugestões para este trabalho.

SUMÁRIO

LISTA DE ABREVIATURAS E SIGLAS

DTW	Dynamic Time Warping
IDE	Integrated Development Environment
JDBC	Java DataBase Connectivity
SGDB	Sistema de Gerenciamento de Banco de Dados
CPU	Central Processing Unit
US	Uniform Scaling
SWM	DTW with Uniform Scaling
csv	Comma-separated values

LISTA DE FIGURAS

LISTA DE LISTAGENS

RESUMO

Uma das premissas da Análise Técnica de ações da Bolsa de Valores é a repetição da história, ou seja, dentro do histórico de cotações de preços alguns padrões podem ser encontrados. Logo, tendo uma forma conhecida, é interessante ter alguma espécie de busca para encontrar padrões similares à essa forma. Analisando alguns dos principais softwares para análise técnica de cotações, percebemos que não existe um mecanismo deste tipo de busca implementado.

Em nosso trabalho sugerimos a busca de séries temporais dentro do domínio da Bolsa de Valores. Essa busca é dada por uma função de similaridade chamada *Dynamic Time Warping* (DTW) que implementamos em nosso protótipo. Para a análise de resultados escolhemos algumas ações e alguns padrões para as consultas. Após isso foram feitas as medidas de precisão e revocação para os resultados obtidos.

A busca por similaride DTW na Bolsa de Valores pode ser considerada eficiente quando procuramos somente uma subsérie que seja similar a consulta, ou seja, o melhor caso retornado será quase sempre satisfatório.

Palavras-chave: Series Temporais, Dynamic Time Warping, Banco de Dados, Bolsa de Valores.

Time Series similarity applied to Brazilian Stock Market

ABSTRACT

One of the premisses in which Technical Analysis of the Stock Market is based is "history repeats itself". It means we can find some patterns on the prices of the actions. So, having a well-knowed shape, is interesting to have some kind of search mechanism which can find a similar shape. We look for such kind of search mechanism in some of the most used comercial softwares for Technical Analysis and realized that this feature is not yet implemented.

In this work, we suggest the search of time series within the Brazilian Stock Market. This search is made by a similarity function called Dynamic Time Warping (DTW) which we implemented in our prototype. For the approach of results we pick some stocks and some patters for query. Then we measure the precision and recall for the results.

The similarity search based on DTW and on the Brazilian Stock Market can be considered efficient if we search for just one shape result that can be compared with the query. That means the best result is almost always very similar to the query.

Keywords: Time Serie, DTW, similarity search, stock market.

1 INTRODUÇÃO

Uma das premissas da *Análise Técnica* [1] de ações da Bolsa de Valores é a repetição da história (MURPHY, 1999), ou seja, dentro do histórico de cotações de preços alguns padrões podem ser encontrados. Nossa idéia é encontrar um padrão através de uma busca simples. Uma busca simples é, dado uma consulta, o programa retornar a melhor resposta encontrada. Além disso, analisando alguns dos principais softwares existentes para análise técnica (GrapherOC (OperAção, 2010), ProfitChart RT (NELOGICA, 2011), MetaStock (MCTRADE, 2011) e NinjaTrader (NINJATRADER, 2011)), constatamos que estes não possuem uma função de busca.

Desse modo, para realizarmos esta busca utilizaremos uma função de similaridade de Séries Temporais. Segundo (KEOGH, 2002a), uma *série temporal* é uma coleção de observações feitas sequêncialmente ao longo do tempo. Os valores das séries em nosso caso serão os preços de fechamento das ações, ou seja, não analisaremos volume, preço no *after market* [2], etc. É importante frisar que séries temporais são por definição ambíguas e difíceis de serem manipuladas por ser preciso lidar com subjetividade (KEOGH, 2002a).

Portanto, este trabalho tem como objetivo apresentar um protótipo que realize a busca de uma série temporal em uma outra série temporal maior dentro do domínio da Bolsa de Valores de São Paulo (BM&FBovespa). Para isso, será usado o algoritmo de busca Dynamic Time Warping (DTW) descrito no capítulo 2 deste trabalho. Outros algoritmos de busca foram considerados mas não utilizados. O protótipo é capaz de, dada uma série, a qual chamamos de *consulta*[3], e dado um domínio, retornar subséries do domínio que mais se aproximam dessa *consulta*. Além disso, é retornado o melhor caso, ou seja, a subsérie que é mais similar a *consulta*. O protótipo implementado faz isso buscando subséries menores no domínio e comparando a distância DTW entre essa série e a *consulta*. A implementação do protótipo está detalhada no capítulo 3.

Para saber se o protótipo funciona e como ele funciona, uma análise de desempenho do algoritmo foi feita. Para isso usamos o método de *precisão/revocação* descrito em (MOREIRA, 2006) e também a medida *Mean Average Precision* como parâmetros de comparação (BAEZA-YATES; RIBEIRO-NETO, 1999). Os resultados obtidos com o protótipo, bem como os gráficos de representação, são apresentados no capítulo 4. O protótipo foi validado através de uma série de experimentos com diferentes versões, essas versões antigas foram utilizadas para calibragem.

Por fim, concluímos nosso trabalho no capítulo 5.

[1] Estudo de uma ação de mercado com o propósito de prever preços futuros.

[2] Operações efetivadas em horário após o fechamento do pregão principal da Bovespa.

[3] Consulta é a tradução da palavra *query* do inglês.

2 FUNÇÕES DE SIMILARIDADE PARA SÉRIES TEMPORAIS

A similaridade de uma série temporal em relação a outra é calculada pela distância entre as duas. Segundo (KEOGH, 2002a): *"A definição de similaridade depende do usuário, do domínio e da tarefa, o algoritmo de similaridade deve ser capaz de tratar todos esses pontos"*. Neste capítulo analisaremos algumas funções de similaridade possíveis para o cálculo de distância entre séries temporais.

Uma imagem intuitiva sobre algoritmos de distâncias é mostrada na figura 2.1. Nela são mostradas duas séries temporais com uma forma semelhante, porém não alinhadas no eixo do tempo. A Distância Euclidiana, que assume que o ponto i de uma série deve ser alinhando com o ponto i da outra série, vai retornar uma grau ruim de similaridade. Enquanto que o algoritmo de Dynamic Time Warping (DTW) mostra um cálculo de distância mais intuitivo (KEOGH, 2002b).

2.1 Propriedades de Medidas de Distância

Todas funções de cálculo de distância entre objetos devem respeitar algumas propriedades a seguir mencionadas. Sendo c_1 e c_2 dois objetos do universo de possíveis objetos. A distância é denotada por $D(c_1, c_2)$. Assim, algumas propriedades devem ser mantidas:

- **Simetria:** $D(c_1, c_2) = D(c_2, c_1)$
- **Similaridade:** $D(c_1, c_1) = 0$
- **Positividade:** $D(c_1, c_2) = 0$ se, e somente se, $c_1 = c_2$
- **Desigualdade triangular:** $D(c_1, c_2) <= D(c_1, c_3) + D(c2, c_3)$

2.2 Distância Euclidiana

Dadas duas séries temporais $Q = q_1...q_n$ e $C = c_1...c_n$ a distância Euclidiana é definida pela fórmula abaixo.

$$D(Q, C) = \sqrt{\sum_{i=1}^{n}(q_i - c_i)^2}$$

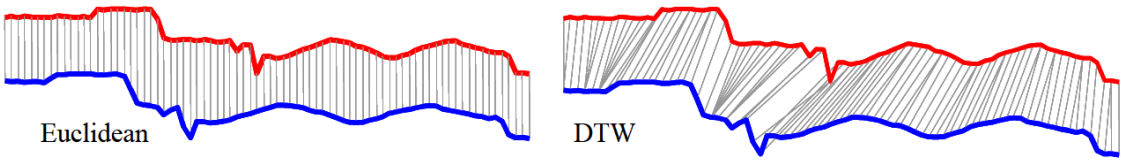


Figura 2.1: Comparação entre Distância Euclidiana e Dynamic Time Warping (figura extraída de (KEOGH, 2002a))

2.3 Distância Euclidiana Quadrática

Porém, conforme descrito em (KEOGH, 2002a), para o cálculo da distância entre séries temporais, ao invés de usar a Distância Euclidiana podemos utilizar a Distância Euclidiana Quadrádica. As distâncias Euclidiana e Quadrática são as mesmas no sentido de que retornam as mesmas classificações. Essa otimização aumenta a performance de CPU.

$$D_{squared}(Q, C) = \sum_{i=1}^{n} (q_i - c_i)^2$$

2.4 Dynamic Time Warping

A função de similaridade entre séries temporais implementada em nosso protótipo é a Dynamic Time Warping[1] (DTW) (FU et al., 2005). Nesta seção vamos descrever como esse algoritmo de similaridade funciona.

Supondo duas séries temporais Q e C, de tamanho *n* e *m* respectivamente, onde: $Q = q_1...q_n$ e $C = c_1...c_n$, queremos alinhar essas duas sequências usando DTW. Para tal, precisamos construir uma matriz *n*-por-*m* onde o elemento (i, j) da matriz contém a distância $d(q_i, c_j)$ entre dois os pontos q_i e c_j (na implementação foi utilizada a Distância Euclidiana Quadrática para o cálculo da distância, que é descrito na seção 2.3). Cada elemento (i, j) corresponde ao alinhamento entre os pontos q_i e c_j. Isso é ilustrado na figura 2.2. Um caminho W é um conjunto contíguo de elementos da matriz que define um mapeamento entre Q e C. Um elemento k de W é definido como $w_k = (i, j)_k$. O caminhamento W é sujeito à algumas restrições:

- **Limites:** $w_1 = (1, 1)$ e $w_k = (m, n)$, isso requer que o caminho comece e termine em cantos diagonalmente opostos da matriz.

- **Continuidade:** Sendo $w_k = (a, b)$ então $w_{k-1} = (a, b)$ onde $a - a <= 1$ e $b - b <= 1$, isso restringe os possíveis passos do caminhamento para células que sejam adjacentes (incluindo diagonalmente adjacentes).

- **Monotonicidade:** Sendo $w_k = (a, b)$ então $w_{k-1} = (a, b)$ onde $a - a >= 0$ e $b - b >= 0$, isso força os pontos de W a serem monotonicamente espaçados no tempo.

[1] A tradução do verbo *to Warp* do inglês é entortar/empenar/deformar.

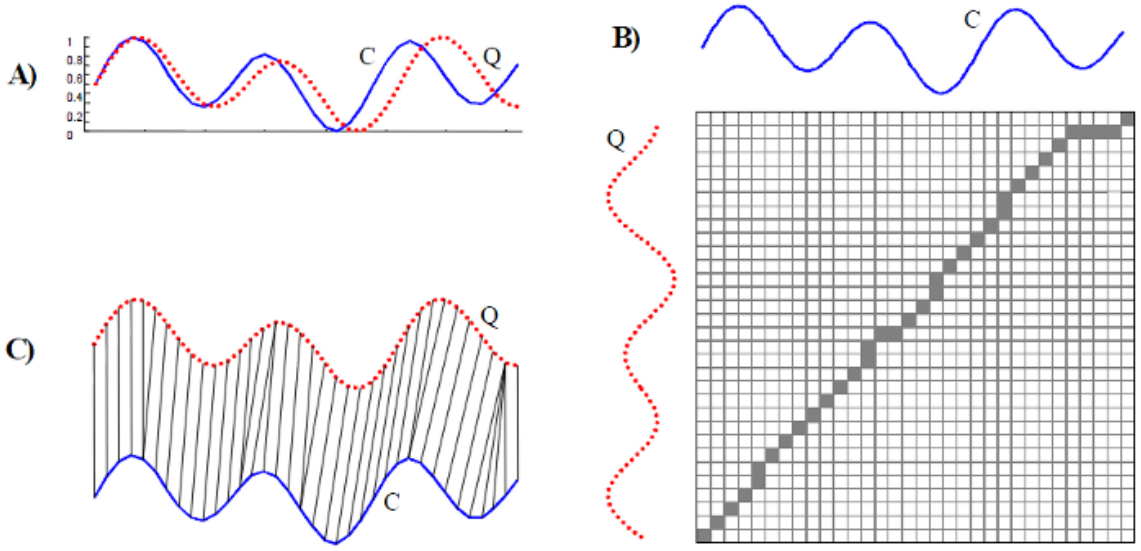


Figura 2.2: **A)** Duas sequências Q e C que são semelhantes, porém fora de fase **B)** Para alinhar as sequências, construimos a matriz e procuramos pelo melhor caminho. **C)** O alinhamento resultante (figura extraída de (KEOGH, 2002a))

Existem vários caminhos que satisfazem as condições acima, entretanto, só nos interessa o caminho que minimize o custo da função:

$$DTW(Q,C) = min\left\{\sqrt{\sum_{k=1}^{K} w_k}\right.$$

Este caminho pode ser encontrado utilizando programação dinâmica para buscar os próximos elementos do conjunto W. Assim, a distância cumulativa definida pela programação dinâmica $\gamma(i,j)$ e o mínimo das distâncias cumulativas das células adjacentes podem ser expressadas como:

$$\gamma(i,j) = d(q_i,c_j) + min\left\{\gamma(i-1,j-1), \gamma(i-1,j), \gamma(i,j-1)\right\}$$

A Distância Euclidiana entre duas sequências pode ser vista como um caso especial de DTW (KEOGH, 2002b). Nesse caso, os elementos k de W são restringidos à $w_k = (i,j)_k$ onde $i = j = k$. Também esse caso especial só pode ser aplicado se as duas sequências possuem o mesmo tamanho, ou seja $n = m$.

2.5 Outras funções

Além dos algoritmos de distância já mencionados anteriormente, consideramos os algoritmos de *Uniform Scaling* (US) e *DTW with Uniform Scaling* (SWM). US é uma técnica que permite que uma série seja globalmente rescalada para depois ser comparada (FU et al., 2005). SWM é uma técnica proposta por (FU et al., 2005) que combina DTW com US. As principais diferenças entre essas propostas para cálculo de distâncias podem ser vistas na figura 2.3. O algoritmo de SWM é muito interessante para o nosso problema, porém ele não foi testado e é proposto como um trabalho futuro.

De acordo com (KEOGH, 2006), existem outras funções de similaridade de séries temporais baseadas em sua forma, porém, também segundo (KEOGH, 2006), nenhuma

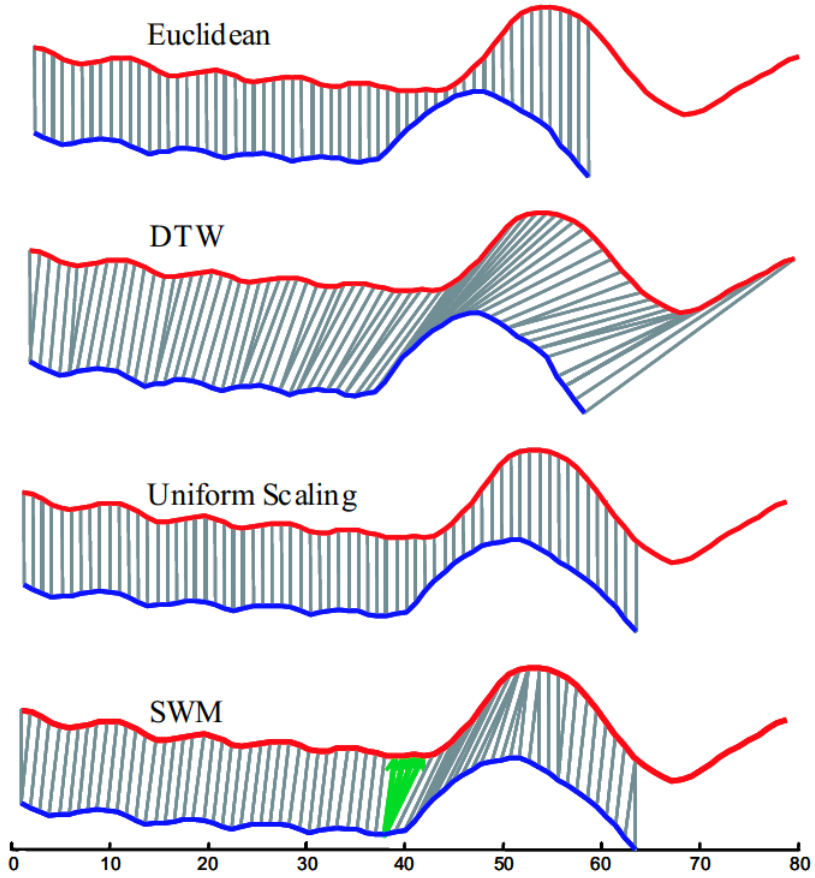


Figura 2.3: Funções de Similaridade (figura extraída de (FU et al., 2005))

delas é melhor do que DTW ou Distância Euclidiana (nesse caso incluímos SWM como uma variação de DTW, também podem existir outras variações de DTW como DTWCID (BATISTA; WANG; KEOGH, 2011)). Existem também dezenas de métodos para acelerar o processo de busca de séries temporais por indexação. Porém, neste trabalho, não utilizaremos nenhum método de indexação de séries temporais.

2.6 Pré-processamento

Segundo (KEOGH, 2002a), se tentarmos medir a distância entre duas séries temporais "cruas"[2], podemos obter alguns resultados não muito intuitivos. Por isso, devemos manipular as séries temporais antes de compararmos elas. A causa disto é a sensibilidade das funções de similaridade à maioria das distorções nos dados. Geralmente essas distorções são insignificantes e devem ser removidas, claro que está não é uma regra geral. Para isso serão utilizados algumas técnicas de pré-processamentos nos dados das séries temporais.

- **Translação de eixo**[3]**:** Alinha as séries no eixo Y removendo a média geral em cada ponto.
- **Escala de Amplitude**[4]**:** Diminui a variação de amplitude dividindo a série por um

[2]"Dados crus"ou "*raw data*"do inglês, é um termo utilizado para dados coletados em uma fonte que não foram sujeitos a nenhuma espécie de processamento ou manipulação.

[3]Tradução do inglês para *Offset Translation*.

[4]Tradução do inglês para *Amplitude Scaling*.

ponto médio.

- **Têndencia Linear**[5]**:** Subtração da linha reta que melhor se encaixa entre os pontos da série.
- **Ruído**[6]**:** Suaviza pontos pelo cálculo de média com seus vizinhos próximos.

Destas quatro distorções indesejadas iremos remover somente duas delas (Translação de Eixo e Ruído) pois, como temos um domínio específico definido, as outras distorções tornam-se desejáveis. Ou seja, encontrar tendências de alta ou baixa quando pensamos em bolsa de valores são altamente desejáveis, pois elas indicam se uma cotação tem maior probabilidade de subir ou cair. Do mesmo modo que uma variação de amplitude, ou seja, um pico de alta ou um vale de baixa em uma cotação são altamente desejáveis de se encontrar. Os detalhes de implementação são descritos na seção 3.3.

[5]Tradução do inglês para *Linear Trend*.
[6]Tradução do inglês para *Noise*.

3 IMPLEMENTAÇÃO

Este capítulo apresentara implementação do algoritmo de busca em forma de um protótipo. Também é apresentado um pouco do histórico de construção do protótipo. A implementação não contém interfaces gráficas, portanto não consideramos o protótipo como ferramenta um programa, e sim como um experimento científico.

Essa implementação tem como objetivo retornar a melhor série temporal encontrada para uma determinada consulta, bem como a segunda melhor série temporal encontrada, a terceira, etc... Através disso, medimos a qualidade do algoritmo no capítulo 4.

3.1 Ambiente de programação

O protótipo foi implementado utilizando a linguagem de programação JAVA (Oracle Corporation, 2011), que foi escolhido devido a facilidade de acesso ao SGDB[1] PostgreSQL (PostgreSQL Global Development Group, 2010a) com conectores do driver PostgreSQL JDBC[2] (PostgreSQL Global Development Group, 2010b). Para a IDE[3] foi escolhido o programa Eclipse (Eclipse Foundation, 2010). Para a construção dos gráficos de saída foi utilizada a biblioteca JFreeChart (Object Refinery, 2010). Para a geração de *logs*[4] foi utilizada a biblioteca log4j (Apache Software Foundation, 2010). O serviço do *Google Code - Project Hosting* [5] foi utilizado para o controle de versões, desse modo, o código fonte completo do protótipo pode ser encontrado no endereço eletrônico: (http://code.google.com/p/jonasthesis/).

3.2 Arquitetura do Protótipo

O protótipo está organizado em etapas sequenciais de processamento. Não está implementada nenhuma forma de paralelismo, porém este é um aspecto que pode ser melhorado como trabalho futuro. Os parâmetros de entrada do protóripo são: uma query de busca e uma lista de ações que compõem o domínio de busca.

Na etapa inicial a função de similaridade é acionada três vezes, uma para cada janela de busca. A relação de tamanho entre a janela de busca e a *consulta* é detalhada na seção 3.4. Após isto os resultados obtidos são armazenados e enviados para o filtro de Remoção

[1]Conjunto de programas de computador responsáveis pelo gerenciamento de um banco de dados.

[2]Conjunto de classes e interfaces escritas em Java que fazem o envio de instruções SQL para qualquer banco de dados.

[3]Programa de computador que reúne características e ferramentas de apoio ao desenvolvimento de software com o objetivo de agilizar este processo.

[4]Processo de registro de eventos relevantes num sistema computacional.

[5]Projeto de hospedagem de desenvolvimento de softwares que fornece sistema de controle de versão.

de Próximos descrito na seção 3.5.1. A próxima etapa aplicada aos resultados é a Mixagem, que é descrita na seção 3.5.2. Finalmente, os resultados são plotados graficamente. O fluxo de execução do protótipo e a arquitetura do mesmo são apresentados na figura 3.1.

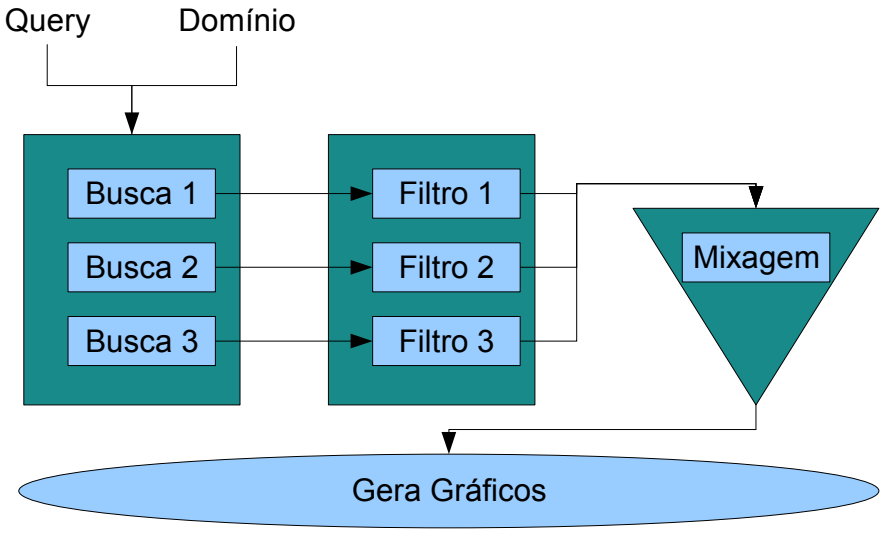


Figura 3.1: Arquitetura do Protótipo

3.3 Pré-Processamento

Para a comparação da *consulta* com um subsequência de dados proveniente da busca, é realizado um pré-processamento antes do cálculo de distância entre as duas séries. Para tal, foram implementados os algoritmos de Translação de Eixo e Remoção de Ruído conforme definidas em (KEOGH, 2006).

A Translação de eixo (KEOGH, 2006) é aplicada removendo a média total dos elementos da série. Sendo a *query* $Q = q_1...q_n$ e uma subsequência a ser comparada $C = c_1...c_n$ a Translação de eixos é definida por

$$Q = Q - mean(Q)$$

$$C = C - mean(C)$$

onde $mean(X)$ é a média aritmética de todos os elementos de X. Um exemplo dessa técnica é mostrado na figura 3.2.

A Remoção de Ruído (KEOGH, 2006) é calculada fazendo-se uma média entre os vizinhos próximos de determinado elemento. Sendo a *consulta* $Q = q_1...q_n$ e uma subsequência a ser comparada $C = c_1...c_n$ a Remoção de Ruído é definida por

$$Q = smooth(Q)$$

$$C = smooth(C)$$

a função $smooth(X)$ é a média aritmética entre um ponto x_n e seus vizinhos c_{n+1} e x_{n-1}. A implementação de $smooth(X)$ em Java é apresentada na listagem 3.1. A apliação desse pré-processamento pode ser visualizada na figura 3.3.

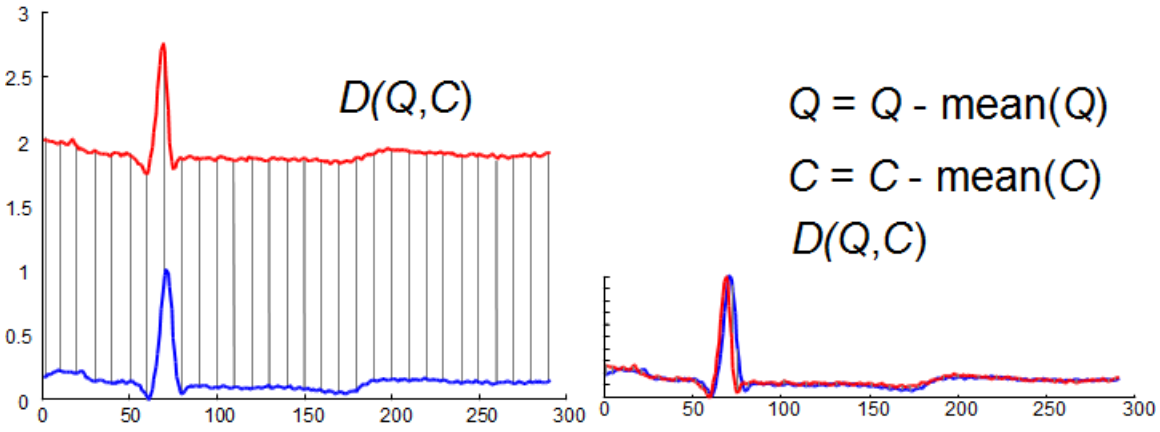


Figura 3.2: Translação de Eixo

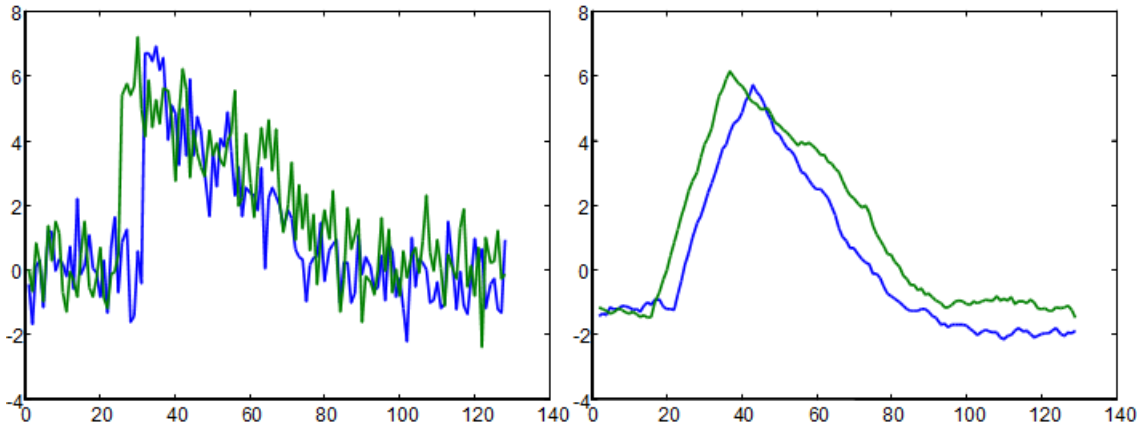


Figura 3.3: Remoção de Ruído

```
1         private void applySmoothToTimeserie(ArrayList<DateAndDouble> T){
2                 double a = 0;
3                 double b = 0;
4                 double c = 0;
5                 //smooth the first point
6                 a = T.get(0).getValue();
7                 b = T.get(1).getValue();
8                 T.get(0).setValue((a+b)/2);
9                 //smooth the other points
10                for(int q = 1; q < T.size()-1; q++){
11                        a = T.get(q-1).getValue();
12                        b = T.get(q).getValue();
13                        c = T.get(q+1).getValue();
14                        T.get(q).setValue((a+b+c)/3);
15                }
16                //smooth the last point
17                a = T.get(T.size()-1).getValue();
18                b = T.get(T.size()-2).getValue();
19                T.get(T.size()-1).setValue((a+b)/2);
20        }
```

Listagem 3.1: Função Smooth

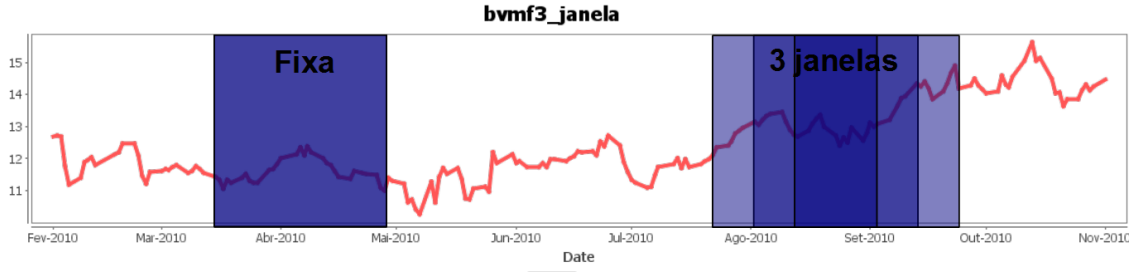


Figura 3.4: Janelas de Busca Fixa e variável com três tamanhos

3.4 Janela de Busca

A busca é realizada comparando uma *consulta* com uma subsérie, que é obtida atráves de uma janela de busca. Essa janela de busca percorre todo o domínio dia após dia. A função de similaridade utilizada para a comparação entre as séries é descrita na seção 2.4. Essa função tem como entrada a *consulta* e uma subsérie do domínio, e retorna como resultado o valor da distância DTW entre as duas.

Numa versão anterior do protótipo, o tamanho da query e da janela de busca eram os mesmos (fixa), depois isso foi modificado para três tamanhos distintos com o objetivo de melhorar a qualidade dos resultados do protótipo. Segundo (FU et al., 2005), a janela de busca pode ser escalada entre os fatores $0.5 <= 1/l <= 1 <= l <= 2$, sendo l o tamanho da *consulta*. Essa proposta é válida para quase todos os domínios, porém, como estamos interessados no domínio da BM&FBovespa[6] utilizaremos somente os fatores 0.65, 1.0 e 1.3. Estes valores foram escolhidos segundo calibragem do programa para consultas com tamanho médio de 10 dias, assim, o tamanho de 10 dias é escalado para 7 e 13 dias respectivamente. O tamanho da janela de busca sofre um arredondamento quando o valor de dias, após aplicado o fator, não é exato. Todas as três buscas são realizadas para cada dia do domínio e os resultados armazenados em três lugares diferentes.

Por exemplo, para uma consulta de 10 dias, é calculada a distância entre a consulta de 10 dias com uma janela de 7 dias, 10 dias com 10 dias e 10 dias com 13 dias. Esses resultados são descritos detalhadamente no Capítulo 4. Um exemplo de janela de busca fixa e com três tamanhos é mostrado na figura 3.4. Este exemplo é meramente ilustrativo, pois o protótipo faz as mesmas buscas mas não com a ordem e nem com paralelismo conforme descrito na imagem. Contudo, o gráfico é de uma ação real e foi retirado do domínio de BVMF3[7], que será discutido mais adiante no capítulo 4.

No decorrer do desenvolvimento foi descoberto um *bug* [8] na janela de busca. Pois, para uma busca onde ocorrem dias sem pregão, não são retornados os valores desses dias. Isso foi corrigido utilizando uma função recursiva para a obteção da janela de busca. Por exemplo se uma janela de busca de tamanho 5 começa na Quinta-Feira dia 05/05/2011, ela possui dois dias sem valores (Sábado e Domingo). Assim, esse intervalo de tempo será aumentado até o dia 11/05/011, ao invés de 09/05/2011. O mesmo ocorre quando temos feriados.

[6] A Bolsa de Valores, Mercadorias e Futuros de São Paulo é a bolsa oficial do Brasil.
[7] Sigla para a empresa BM&F Bovespa ON NM.
[8] Erro no funcionamento comum de um software.

3.5 Filtragem de Resultados

Após percorrida três buscas para cada dia do domínio, é necessário filtrar e classificar os melhores resultados para a análise posterior. Para isso dois filtros foram utilizados.

3.5.1 Remoção de elementos próximos

Esse filtro ordena os resultados de uma janela de busca em ordem crescente. Após isso, o melhor resultado é selecionado e são descartados todos os resultados que tem datas próximas a este. Isso é repetido para o segundo melhor resultado e assim por diante. Por datas próximas, entende-se os resultados cujas datas não são maiores ou menores que o tamanho da janela de busca.

Por exemplo, se o melhor resultado é no dia 06/05/2011 e a janela de busca tem 5 dias, os resultados entre os dias 01 e 05 e entre 07 e 11 serão desconsiderados, pois esses resultados vão estar sobrepostos. Depois disso, o segundo melhor resultado será selecionado e o algoritmo repete-se até o último resultado.

3.5.2 Mixagem

Quando a etapa da Mixagem chega, temos três séries de melhores resultados já filtrados. Então essas séries deverão ser agrupadas em um só conjunto de melhores resultados para então serem plotados. Nessa etapa, as séries podem conter tamanhos diferentes, o que não ocorria no filtro anterior. Para diferenciarmos os tamanhos, as séries com fator 0.65 serão chamadas pequenas, as com fator 1.0 normais e as de 1.3 grandes. O pseudo-código do algoritmo é descrito abaixo.

1. Para cada série grande verifica-se se existem uma série normal e uma série pequena contidas no espaco de tempo da série grande.
   (a) Se sim, excluimos as séries normal e pequena;
   (b) A série grande passa a ter o menor valor de ordenamento entre as três séries analisadas.
2. Para cada série normal verifica-se se existe uma série pequena contida no espaco de tempo da série normal.
   (a) Se sim, a série pequena é excluida e a série normal recebe o menor número de ordenamento entre as duas.

4 AVALIAÇÃO

Neste capítulo vamos apresentar os resultados obtidos com a análise de desempenho do protótipo. Aqui o desempenho é a qualidade dos resultados obtidos, diferentemente de velocidade para obtenção dos resultados (*throughput* ou tempo de resposta). Para obtenção da base de dados optou-se pelo *website* Yahoo Finance (Yahoo! Inc., 2010) e a extração de dados por arquivos *.csv* [1]. O protótipo conta com um *parser* [2] para leitura dos arquivos *.csv* que contém o histórico de cotações de determinada ação.

Os principais objetivos da avaliação são: verificar se o protótipo funciona para a bolsa de valores e verificar se a qualidade dos resultados obtidos são satisfatórios. Para ver se os resultados são satisfatórios, devemos comparar os gráficos obtidos pelo método de precisão/revocação. Para ver se o protótipo funciona devemos provar a propriedade $D(q, q) = 0$. Essa propriedade foi provada pelo seguinte teste: retiramos uma subsequência de um domínio qualquer, após isso, utilizamos essa subsequência como consulta, então, o protótipo deve retornar como melhor resultado exatamente a subsequência que foi retirada. Esse teste já foi realizado e foi corretamente concluído provando assim a validade do algoritmo.

4.1 Dados Utilizados

Foram escolhidas três *consultas* com cinco, dez e vinte dias; que vamos chamar Q5, Q10 e Q20. Essas *consultas* são mostradas na figura 4.1. Para o domínio foi escolhido o período de Fevereiro de 2010 até Novembro de 2010, ou seja, 274 dias. Foram escolhidos cinco ativos das principais ações do mercado brasileiro para a composição do domínio de busca: BBAS3[3], BVMF3[4], GGBR3[5], MPXE3[6] e OGXP3[7]. Assim, com cinco ativos diferentes e mais três tipos de *consultas* de busca, o domínio total é composto por 15 séries distintas. Ou seja, a Q5 será buscada nos 274 dias de BBAS3, de BVMF3, de GGBR3, de MPXE3 e de OGXP3; do mesmo modo serão procuradas também Q10 e Q20. Um exemplo de domínio com as séries relevantes desejadas já marcadas é mostrado na figura 4.2.

[1] Comma-separated values (CSV), em português Valores Separados por Vírgula, é um formato de arquivo que armazena dados tabelados.

[2] Processo de analisar uma sequência de entrada.

[3] Banco do Brasil S.A. ON

[4] BM&F Bovespa ON NM

[5] Gerdau S.A. ON N1

[6] Mpx Energia ON

[7] OGX Petróleo e Gás ON

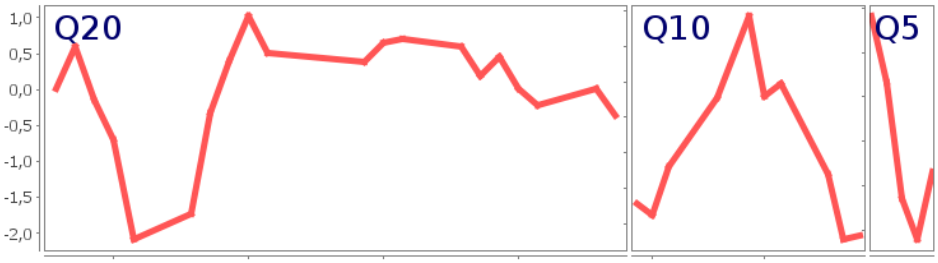


Figura 4.1: Séries Temporais utilizadas para testes

Figura 4.2: Marcação de séries relevantes de acordo com query Q10

4.2 Métricas de Qualidade

A base da avaliação de sistemas de recuperação de informações é a ideia de relevância (MOREIRA, 2006). Com base em uma consulta proposta por um usuário, os documentos são classificados como relevantes ou irrelevantes. A relevância é tratada como binária, ou seja, não existem as categorias de muito relevante, razoavelmente relevante, etc. Os criticos afirmam que a relevância é subjetiva, porém, embora bastante criticada, esta abordagem ainda é o padrão (MOREIRA, 2006).

Em nossos testes, algumas subséries presentes no domínio foram consideras relevantes. Ou seja, no nosso entendimento o protótipo deverá encontrar as séries que consideremos interessantes além do resultado ótimo, que é a série que mais se aproxima da *consulta*. Um exemplo de como essa marcação foi feita está na figura 4.2. Essa figura é parte do histórico de cotações de BBAS3 onde, neste caso, três séries relevantes foram marcadas em azul.

4.2.1 Precisão / Revocação

Considerando um domínio de séries temporais (para exemplificação utilizaremos o domínio BBAS3) e um conjunto de séries temporais relevantes, $|R|$ será o número de séries relevantes e o número de documentos recuperados pelo protótipo será $|A|$. O número de documentos na interseção entre R e A será $|Ra|$. Estes conceitos são baseados na livro (BAEZA-YATES; RIBEIRO-NETO, 1999) e a ilustração do caso está na figura 4.3.

No nosso caso as séries temporais relevantes foram marcadas em azul conforme exemplificado na figura 4.2. As séries consideradas relevantes são as que mais se aproximam da *consulta* Q10 descrita na seção 4.1 e mostrada na figura 4.1, essa *consulta* tem basicamente a forma de uma subida de 5 dias seguida por uma queda de 5 dias, ou seja,

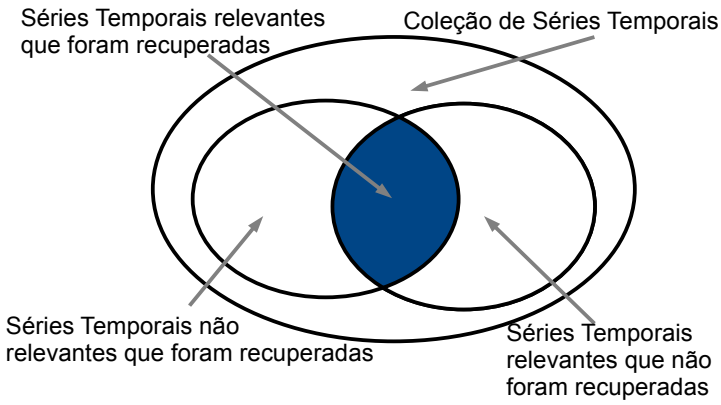


Figura 4.3: Precisão/Revocação para Séries Temporais

um pico. Os dados para estas séries são retirados de um período limitado de tempo do BBAS3. Não é mostrado todo o domínio de BBAS3 para uma melhor visualização, bem como não são mostradas todas as séries marcadas como relevantes.

Assim sendo, as medidas de precisão e revocação são dadas como:

- **Revocação** é a fração entre o número de séries relevantes recuperadas e o número total de séries relevantes existentes.

$$\text{Revocação} = \frac{|Ra|}{|R|}$$

- **Precisão** é a fração entre o número de séries relevantes recuperadas e o número total de séries recuperadas.

$$\text{Precisão} = \frac{|Ra|}{|A|}$$

Como precisão e revocação são medidas baseadas em conjuntos, ou seja, não levam em conta a ordenação do resultado (MOREIRA, 2006), utilizaremos o sistema de 11 pontos de revocação. Esse sistema avalia a precisão nos pontos com revocação 0%, 10%, 20%, ..., 100%. Por exemplo, para o cálculo da precisão do ponto de revocação 10% é levado em conta quantas séries foram recuperadas para encontrar a primeira série relevante. Isso repete-se para os demais pontos de revocação. Desse modo, a noção de *ordem* dos resultados é preservada. Por definição, a precisão no ponto de revocação 0% é 100%. Com esse sistema é possível notar mais intuitivamente as diferenças entre as séries analisadas, pois conseguimos visualizar os resultados em um gráfico.

Um exemplo de tabela juntamente com um gráfico pode ser visto na figura 4.4. O resultado das séries recuperadas ordenadas é mostrado na figura 4.5 e pode ser visualmente comparado com a figura 4.2. Na figura 4.5, que mostra as séries recuperadas, podemos notar as setas indicando um número. Esse número é a ordem de recuperação dada pelo protótipo. Também existe uma seta com o nome de *BEST*, que indica o melhor resultado encontrado, ou seja, a série mais similar a *consulta*.

Segundo (MOREIRA, 2006) e também (BAEZA-YATES; RIBEIRO-NETO, 1999), a curva de precisão/revocação deve sempre ser decrescente. Por isso, os resultados para todos os casos testados foram interpolados. *"A precisão interpolada para um nível de recall j é o maior valor de precisão para qualquer nível de recall maior ou igual a j"*

Recall	Precision	Recuperados	Relevantes
0%	100,00%		
10%	100,00%	1	1
20%	100,00%	2	2
30%	100,00%	3	3
40%	80,00%	5	4
50%	62,50%	8	5
60%	46,15%	13	6
70%	0,00%	0	7
80%	0,00%	0	8
90%	0,00%	0	0
100%	0,00%	0	0

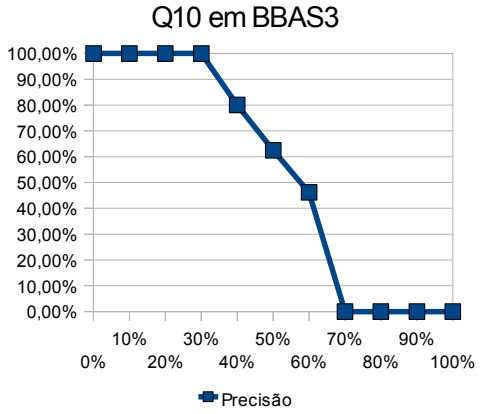


Figura 4.4: Precisão/Revocação de Q10 em BBAS3

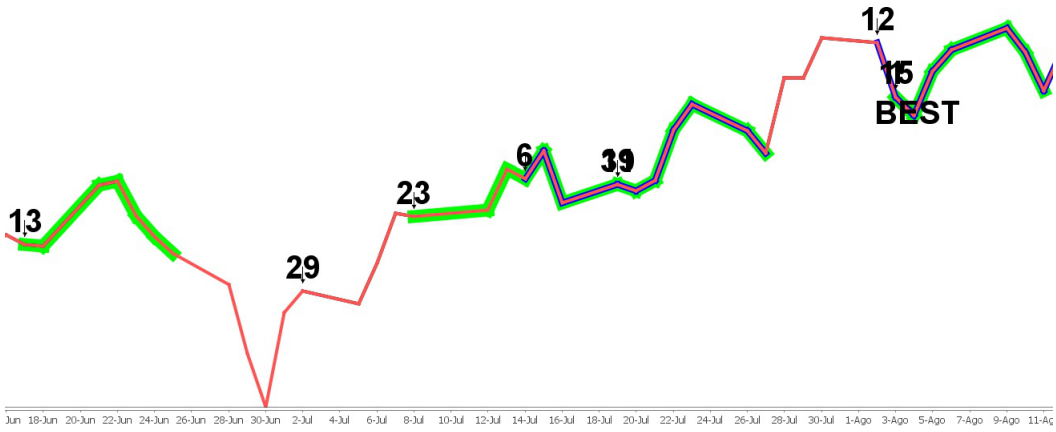


Figura 4.5: Séries relevantes recuperadas de acordo com query Q10

(MOREIRA, 2006).

4.2.2 Mean Average Precision

Outro modo de avaliar o desempenho de funções de similaridade é por *Mean Average Precision* (MAP). O MAP basei-se na Precisão Média de uma consulta, que é a média das precisões após cada documento relevante recuperado (MOREIRA, 2006). A medida MAP, para uma série de consultas, é a média dessas Precisões Médias de cada consulta. Para o exemplo de Q10 em BBAS3 temos uma Precisão Média de 53,51%. Neste caso, como exemplo, o MAP de toda Q10 seria calculado pela média das Precisões Médias de BBAS3, BVMF3, GGBR3, MPXE3 e OGXP3.

A fórmula para o cálculo de Mean Average Precision é descrita como

$$MAP = \frac{\sum_{q=1}^{Q} AveP(q)}{Q}$$

onde Q é o número total de *consultas* e AveP(q) é a Precisão Média de uma *consulta*.

4.3 Resultados

A principal função do protótipo é encontrar a série mais similar à *consulta*. Conforme vimos na figura 4.5, uma seta com a marcação *BEST* é designada para isso. Aqui mostraremos alguns desses resultados obtidos, porém, o número que melhor representa esse resultado é a precisão do ponto de revocação 10% da média geral. Pois esse ponto indica que a série ótima é quase sempre uma série relevante! Devemos lembrar que o domínio pode, às vezes, não conter um resultado muito semelhante à *consulta*, isso ocorre pela própria natureza da bolsa de valores, pois é possível que nada similar seja encontrado. A comparação entre *consultas* e domínios encontrados é ilustrada nas figuras 4.6 e 4.3. Nessas imagens as respectivas *consultas* foram plotas em azul enquanto que os melhores resultados estão em vermelho.

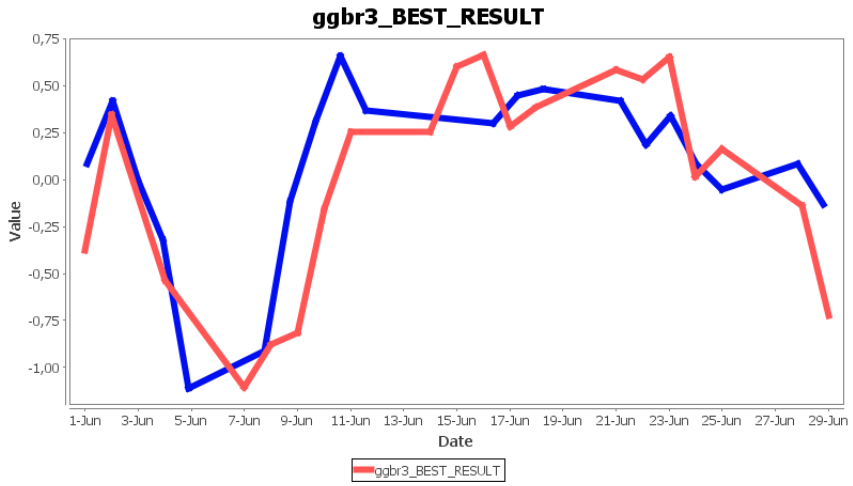
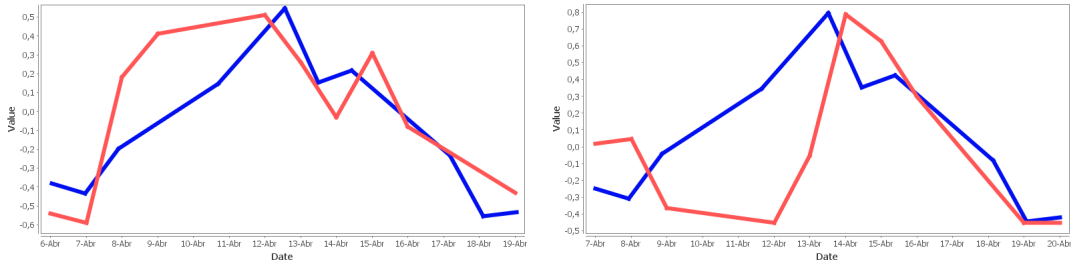


Figura 4.6: Melhor resultado da *consulta* Q20 para GGBR3



(a) BBAS3 (b) MPXE3

Figura 4.7: Melhores resultados de (a) e (b) comparados com a Q10

O segundo objetivo da validação é testar a precisão/revocação e o índice MAP para cada ação analisada. Para isso foi feita uma média simples de todas as ações, desse modo, podemos analisar o desempenho do algoritmo para cinco, dez e vinte dias. Por fim, uma média geral de Q5, Q10 e Q20 foi feita, e podemos analisar o desempenho geral do protótipo. Um exemplo de tabela é descrito na figura 4.4. Esses resultados são mostrados nas figuras 4.8, 4.9 e 4.10.

Para cada gráfico, duas linhas são plotadas, a linha com terminação **_100_i** representa a coleção de resultados obtidos somente com a janela de busca de mesmo tamanho da *consulta*. As linhas com terminação **_var_i** representam os resultados obtidos com três

	Q5	Q10	Q20	MEDIA
FIXO	63,00%	46,86%	28,15%	46,00%
VAR	76,23%	50,01%	29,99%	52,08%

Tabela 4.1: Mean Average Precision

janelas de buscas variáveis, conforme descritos no capítulo 3. Estes ajustes de parâmetros mostram uma diferença significativa. A terminação **i** vem de interpolada.

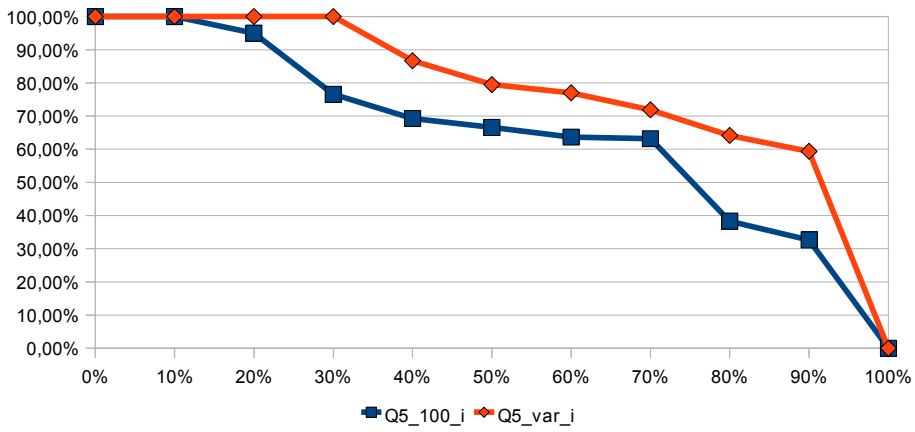


Figura 4.8: Precisão/Revocação para uma *consulta* de 5 dias

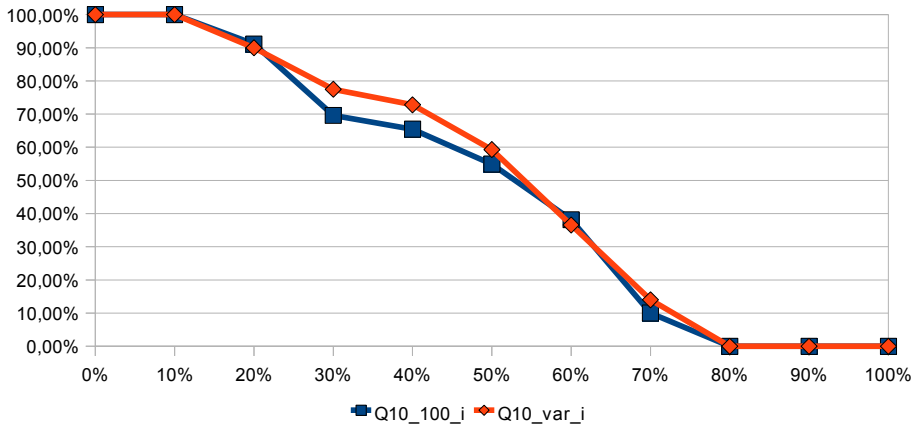


Figura 4.9: Precisão/Revocação para uma *consulta* de 10 dias

Analisando os gráficos podemos chegar a conclusão de que a variabilidade da janela de busca é benéfica para o desempenho do algoritmo. Pois a linha **_100_i** que representa os resultados de janela fixa é sempre ligeiramente melhor do que a linha **_var_i**, que representa a janela de busca para três casos. Além disso, os resultados de MAP com janela fixa ou com três janelas de buscas (variável), apresentados na tabela 4.3 nos mostram a mesma coisa. Essa afirmação também é válida para a média geral. Outro ponto a ser observado é que quanto menor o tamanho da *consulta* buscada, melhor é a precisão/revocação!

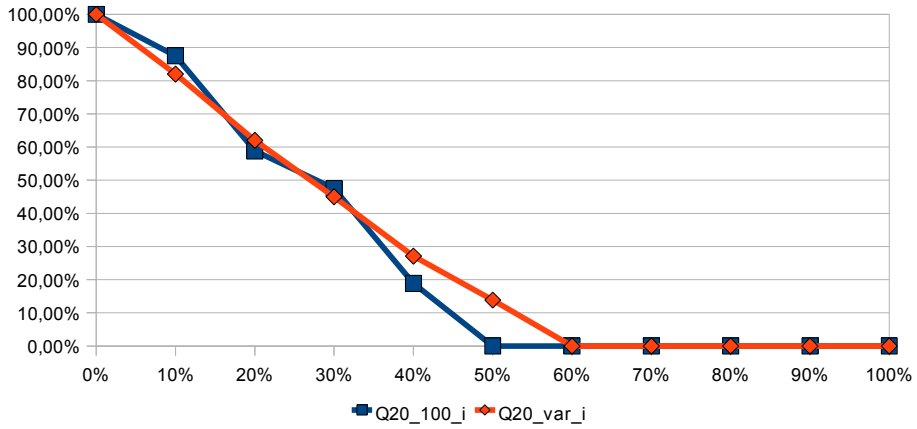


Figura 4.10: Precisão/Revocação para uma *consulta* de 20 dias

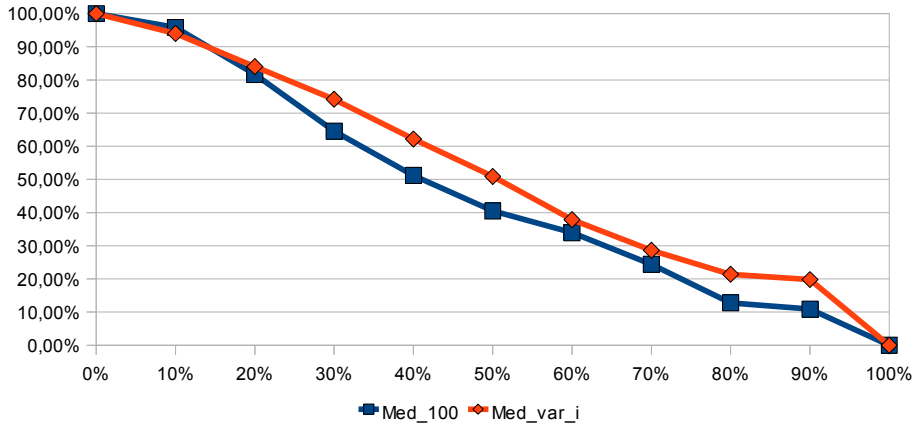


Figura 4.11: Média de Precisão/Revocação para *consultas* de 5, 10 e 20 dias

5 CONCLUSÃO

Este trabalho discutiu sobre algumas funções de similaridade para Séries Temporais. Também foram revisados alguns conceitos sobre distâncias entre séries. Com o objetivo de realizar buscas na Bolsa de Valores, foi escolhido o algoritmo de *Dynamic Time Warping* para ser implementado em um protótipo.

A implementação do protótipo foi realizada em JAVA e utiliza o Sistema de Gerência de Banco de Dados POSTGRESQL. A implementação teve duas arquiteturas distintas e os resultados dessas duas maneiras de buscar por séries temporais foram analisados. Além disso, algoritmos extra de filtragem de resultados foram também implementados.

Por fim, foi feita uma avaliação de desempenho do protótipo. Essa avaliação utilizou o método de precisão/revocação para analisar a eficiencia do programa implementado. Nesses resultados descobrimos que o algoritmo é bom para encontrar a *consulta* desejada, ou seja, dada uma *consulta* de busca o protótipo retornará, em primeiro lugar, um resultado que é satisfatório (relevante). Porém, quando buscamos por mais séries semelhantes no mesmo domínio, a eficiência do protótipo diminui. Uma das causas desse fenômeno é devida à ambiguidade da própria bolsa de valores, ou seja, nem sempre existem padrões repetitivos dentro de uma determinada ação de mercado.

5.1 Trabalhos Futuros

Por realizar uma pesquisa sequencial, um trabalho futuro de implementação seria o paralelismo da função de busca, ou seja, melhorar a velocidade de processamento do algoritmo. Também é interessante a construção de uma interface gráfica para ser usado por um eventual usuário final.

Outra ideia é testar a eficiência com outra função de similaridade, que pode ser, por exemplo, a função de SWM descrita em (FU et al., 2005). Essa função usaria de outros fatores de escala para realizar a comparação.

REFERÊNCIAS

Apache Software Foundation. **log4j**. Disponível em: <http://logging.apache.org/log4j/>. Acessado em: agosto/2010.

WESLEY, A. (Ed.). **Modern Information Retrieval**. [S.l.]: New York, 1999.

BATISTA, G.; WANG, X.; KEOGH, E. J. **A Complexity-Invariant Distance Measure for Time Series**. 2011.

Eclipse Foundation. **Eclipse Software**. Disponível em: <http://www.eclipse.org>. Acessado em: agosto/2010.

FU, A. W.-C. et al. Scaling and Time Warping in Time Series Querying. **Very Large Data Base Journal**, [S.l.], v.17, p.899–921, 2005.

KEOGH, E. J. **A Tutorial on Indexing and Mining Time Series Data**. Gramado, Brazil.: The XVII Brazilian Symposium on Databases, 2002.

KEOGH, E. J. **Exact indexing of dynamic time warping**. Hong Kong, 2002. 406-417p.

KEOGH, E. J. **VLDB06 Time Series Tutorial**. 32.ed. The Convention and Exhibition Center, Seoul, Korea.: Very Large Data Bases, 2006.

MCTRADE. **MetaStock**. Disponível em: <http://www.mctrade.com.br>. Acessado em: abril/2011.

MOREIRA, V. P. **Avaliação de Desempenho de Sistemas de Recuperação de Informações**. Slides da disciplina CMP254 (Bancos de Dados, Web e Recuperação de Informações) do Institudo de Informática da UFRGS.

MURPHY, J. J. **Technical Analysis of the Financial Markets**. [S.l.]: New York Institute of Finance, 1999.

NELOGICA. **ProfitChart RT**. Disponível em: <http://www.nelogica.com.br>. Acessado em: abril/2011.

NINJATRADER. **NinjaTrader**. Disponível em: <http://www.ninjatrader.com.br>. Acessado em: maio/2011.

Object Refinery. **JFreeChart**. Disponível em: <http://www.jfree.org/jfreechart/>. Acessado em: agosto/2010.

OperAção. **GrapherOC**. Disponível em: <http://www.operacaoconsultoria.com.br>. Acessado em: janeiro/2011.

Oracle Corporation. **Java Programming Language**. Disponível em: <http://www.oracle.com/technetwork/java/>. Acessado em: abril/2011.

PostgreSQL Global Development Group. **PostgreSQL Software**. Disponível em: <http://www.postgresql.org>. Acessado em: agosto/2010.

PostgreSQL Global Development Group. **PostgreSQL JDBC driver**. Disponível em: <http://jdbc.postgresql.org/>. Acessado em: agosto/2010.

Yahoo! Inc. **Yahoo Finance**. Disponível em: <http://br.finance.yahoo.com/>. Acessado em: setembro/2010.