UNIVERSIDADE FEDERAL DO CEARÁ

CAMPUS DE QUIXADÁ

CURSO DE SISTEMAS DE INFORMAÇÃO

ANÁLISE COMPARATIVA DE MÉTODOS DE COMPRESSÃO APLICADOS A DADOS TEXTUAIS ESTRUTURADOS EM SVG

Projeto de Pesquisa

Anderson Lemos da Silva

Orientador:

Prof. Arthur Rodrigues Araruna

QUIXADÁ

Junho, 2015

SUMÁRIO

1 INTRODUÇÃO

No cenário atual a informação tem se transformado em um bem crucial para qualquer entidade, seja ela uma pessoa física ou organização. Para organizações a posse ou não de uma informação, ou mesmo a demora em obtê-la, pode ser determinante sobre seu sucesso ou fracasso. Assim, há uma necessidade de se obter informação o mais rápido possível.

Os sistemas computacionais são uma alternativa importante para se chegar a essa meta. No entanto, o volume de dados que representam tais informações está cada vez maior, causando problemas ligados a armazenamento e transmissão dos dados.

Para tal problema de armazenamento, pode ser empregada uma solução como compressão de dados e uso de documentos de dados estruturados. A compressão de dados pode ser uma solução para o problema de volumes de dados muito grandes, pois basicamente ela visa transformar uma entrada de dados original em outro conjunto de dados de menor volume, que posteriormente pode ser retornado aos dados originais ou com suficiente similaridade. Já os documentos de dados estruturados se utilizam de uma padronização, organizando-se em blocos semânticos em que um mesmo grupo de dados possui as mesmas descrições, sendo que esses tipos de representações de dados permitem que determinadas técnicas de compressão se aproveitem dessa estrutura, o que pode gerar um ganho ainda maior no resultado da compressão. Os documentos XML e o formato JSON são exemplos de dados estruturados.

Existem vários métodos de compressão de dados utilizados, alguns baseados na codificação de Huffman, alguns em compressores LZ, outros especializados como o XMill — desenvolvido para compressão de documentos XML — muitos deles citados em Salomon (2004). Alguns desses métodos aproveitam-se do padrão conhecido dos dados estruturados para melhorar a eficiência da sua compressão. Neste trabalho, temos como objetivo realizar um comparativo entre algumas dessas técnicas em uma das classes de dados estruturados e eleger uma boa opção de método de compressão para esse domínio.

2 FUNDAMENTAÇÃO TEÓRICA

Nesta seção serão apresentados conceitos sobre dados estruturados e compressão de dados. Em suma, compressão de dados é o processo que transforma um conjunto de dados em outro de tamanho menor, podendo fazer uso do padrão utilizado nos dados estruturados —

organização de dados bastante utilizada na comunicação entre sistemas remotos — para tentar aperfeiçoar a eficiência da compressão.

Na seção de dados estruturados serão apresentados dois tipos de documentos estruturados: o XML — que é um dos mais utilizados padrões de representação externa de dados — e o SVG — que é um tipo de dados derivado do anterior, e que foi escolhido para escopo deste trabalho. Já na seção de compressão de dados abordaremos métodos de compressão de dados, com enfoque em quatro métodos escolhidos para o escopo deste trabalho.

2.1 Dados estruturados

Com o grande crescimento do uso dos meios tecnológicos, surgiu a necessidade de que os sistemas de informação se tornassem cada vez mais robustos e atendessem requisitos como disponibilidade, escalabilidade, concorrência, tolerância a falhas, entre outros. Como uma alternativa a alcançar esse objetivo, os sistemas começaram a ser criados de maneira distribuída.

Um sistema distribuído pode ser definido como um sistema no qual seus componentes estão situados em diferentes partes de uma rede, sejam esses componentes de hardware ou software. A comunicação entre todas as partes de um sistema distribuído e a coordenação de suas ações é feita apenas por troca de mensagens (COULOURIS et al., 2013).

Nem sempre essa troca de mensagem é feita de maneira simples. Com a grande variedade de tecnologias, os sistemas distribuídos se tornaram cada vez mais heterogêneos, e partes que precisam se comunicar podem se diferenciar em diversos fatores, tais como: implementações de protocolo de rede, representação de dados, plataformas de execução, padrões de invocação de serviço, entre outros (COULOURIS et al., 2013).

Em especial, para o escopo deste trabalho, podemos destacar as diferenças entre as representações de dados. As partes em um sistema distribuído precisam manipular o mesmo padrão de dados que serão trocados nas mensagens que essas partes utilizam para se comunicar. Como uma alternativa para tentar resolver esse problema foram construídos os dados estruturados.

Segundo Almeida (2002), “dados estruturados são dados dispostos em representações rígidas, sujeitas a regras e a restrições impostas pelo esquema que os criou. Programas que produzem tais dados os armazenam em disco, para que possam ser utilizados em formato binário ou texto”. De forma semelhante, Arasu e Garcia-Molina (2003) diz que

dados estruturados podem ser definidos como qualquer conjunto de dados que seguem um esquema ou um tipo em conformidade.

Existem diferentes tipos de dados estruturados, dentre eles os textuais, que são o foco deste trabalho. Em meio aos dados textuais estruturados podemos citar o SGML, o XML, o HTML e o JSON. Apesar de pertencerem ao mesmo domínio de aplicação — linguagens de marcação — o SGML, o XML e o HTML não têm o mesmo propósito (ALMEIDA, 2002). Já o JSON é derivado do JavaScript, uma linguagem de programação. Outro tipo de dado textual estruturado é o SVG, que é baseado em XML e foi criado para representar dados gráficos vetoriais. Como já citado, o SVG é o domínio principal de escopo deste trabalho. A seguir serão definidos e diferenciados dois desses tipos de dados: o XML e o SVG.

2.1.1 XML

Uma linguagem de marcação pode ser definida como um conjunto convenções utilizadas para codificar textos e especificar quais marcas são exigidas e/ou permitidas, diferenciando o que faz parte do texto original e o que faz parte da marcação (ALMEIDA, 2002).

Nesse contexto foi criado um padrão internacional, o *Standard Generalized Markup Language* — em tradução livre, “Linguagem de Marcação Padrão Generalizada” — ou apenas SGML, que define a estrutura e o conteúdo de documentos eletrônicos, documentos esses que podem ser de vários tipos. Pode-se dizer que o SGML é o precursor para outros tipos de documentos estruturados de linguagem de marcação.

O *Extended Markup Language* — em tradução oficial, “Linguagem de Marcação Estendida” —, ou XML, é uma delas, e pode ser visto como uma versão resumida do SGML (ALMEIDA, 2002).

A estrutura de um documento XML é semelhante à de um documento HTML — que é uma linguagem de marcação com *tags*, palavra usada originalmente para especificar uma classe de relatórios técnicos de uso comum em escritórios e que são marcações fixas do documento, sendo, atualmente, o padrão em uso para páginas na Internet —, a diferença é que as *tags* do XML não são pré-definidas. Com essa vantagem, o autor de um documento XML pode especificar a forma de apresentação dos dados, além de poder dar definições semânticas. Um arquivo XML pode conter, além dos dados, a descrição da estrutura do documento. Um exemplo de descrição do XML é através do *Data Type Definitions* — em tradução livre, “Definições de Tipo de Dados” — ou DTD, que são gramáticas que conferem a estrutura do

documento XML. O XML omite as partes mais complexas e menos utilizadas do SGML, o que simplifica sua definição (ALMEIDA, 2002).

2.1.2 SVG

Atualmente o conteúdo na *web* é mostrado através de diversas formas tais como texto, áudios, vídeos e imagens. Em especial, as imagens têm um papel importante nesse conteúdo, pois elas podem esclarecer ou demonstrar melhor o significado de uma informação. Quando falamos de imagens digitais, há basicamente duas formas de representação: matriz de pixels — também conhecida como *bitmap* — e as imagens vetoriais (GOMES, 2005).

As imagens do tipo *bitmap* são representadas por uma matriz de pontos — os *pixels* — e sua resolução depende da quantidade de pontos existentes nessa matriz, ou seja, tem resolução fixa. Isso se torna uma desvantagem, pois ao ser ampliado, este tipo de imagem perde qualidade e ao ser reduzida a imagem pode perder nitidez (GOMES, 2005).

Com o intuito de contornar este problema, foram criadas as imagens vetoriais. Diferentemente das imagens *bitmap*, não é armazenada a matriz de pontos que formam a imagem, que na verdade é gerada a partir de descrições geométricas de formas (GOMES, 2005).

Uma forma de representar imagens vetoriais é com o uso do *Scalable Vector Graphics* — em tradução livre, “Gráficos Vetoriais Escaláveis” —, ou SVG, que é na verdade um tipo de dados textual baseado em XML. Ele pode descrever, de forma vetorial, gráficos bidimensionais através de código XML (GOMES, 2005). Em outras palavras, o SVG não desenha a imagem, ele fornece informações de como desenhar a imagem.

A definição oficial dada pela W3C — *World Wide Web Consortium* —, desenvolvedora da linguagem SVG, é a seguinte: “SVG é uma linguagem para descrever gráficos bidimensionais. SVG permite três tipos de objetos gráficos: gráficos vetoriais — por exemplo, caminhos que consistem em linhas retas e curvas —, imagens e texto. O conjunto de recursos do SVG inclui transformações aninhadas, caminhos de recorte, máscaras alfa, efeitos de filtro e *templates* de objetos” (FERRAIOLO; JUN; JACKSON, 2000).

Com o aumento do uso de imagens SVG principalmente na *web* se tornou importante a rápida obtenção dessas imagens. O tamanho de um arquivo SVG pode influenciar na velocidade de carregamento de uma página web, por exemplo. Uma alternativa a resolver esse problema é a compressão de dados, que pode auxiliar para que as imagens sejam transferidas mais rapidamente, tanto por meios locais e, principalmente, por meios remotos.

2.2 Compressão de dados

Ao longo do desenvolvimento tecnológico, surgiu a necessidade de armazenar dados em volumes cada vez maiores. Armazenar dados vem se tornando um problema à medida que o volume de dados aumenta, e o tamanho da memória de armazenamento pode não ser mais capaz de suportar tantos dados. Como uma alternativa de resolver este problema foi criada a compressão de dados.

Como foi dito por Teixeira (2011) “compressão de dados é o processo de converter um conjunto de entrada de dados — dados originais — em um conjunto de dados de saída — dados comprimidos —, que possui um menor tamanho comparado aos dados originais”. Salomon (2004) fornece uma definição semelhante ao dizer que “dado um fluxo de dados de entrada — entrada ou fluxo original —, a compressão é o processo de conversão desses dados em outro fluxo de dados que possui tamanho menor que o original — saída ou fluxo comprimido”. Para esse autor, um fluxo é considerado como sendo um arquivo ou um *buffer* na memória. Ele utiliza a palavra “fluxo” ao invés de “arquivo” simplesmente porque os dados podem ser transmitidos por uma rede e mandados diretamente ao mecanismo que compacta ou descompacta esses dados, ou seja, nem sempre precisam estar armazenados em um arquivo para que possam ser processados.

A compressão de dados torna-se importante por várias razões. Dentre elas podemos citar duas mais relevantes: primeiro, porque as pessoas não gostam de perder o que têm armazenado — elas preferem guardar estes dados em algum dispositivo —, o que pode se tornar um problema, pois mesmo que a capacidade de armazenamento de um dispositivo seja grande, em algum momento a memória vai acabar, e a compressão de dados pode retardar esse esgotamento de memória. Em segundo, as pessoas não gostam de esperar muito tempo enquanto estão fazendo uma transferência de dados, seja essa transferência feita entre dispositivos locais ou remotos. (SALOMON, 2004).

Outro fator interessante na compressão de dados é que ela pode ser aplicada no processamento de dados de diversos formatos, tais como binário, texto, pixel ou outra forma de representação de dados (TEIXEIRA, 2011). Neste trabalho, observaremos métodos no contexto de tipos textuais, mais precisamente, dados textuais estruturados do tipo SVG.

2.2.1 Técnicas de compressão de dados

Para que dados sejam comprimidos é necessário um processamento desses dados. Neste trabalho definimos como técnicas de compressão qualquer método conhecido que realize, ou forneça instruções de como realizar esse processamento.

Existem diferentes técnicas de compressão de dados, algumas delas genéricas para qualquer tipo de dados, outras dedicadas a dados específicos. Em especial, existem técnicas dedicadas à codificação de dados textuais e outras para dados binários — que também são eficazes ao serem usadas para comprimir dados textuais, embora não necessariamente eficientes. Dentre as abordagens de compressão de dados textuais, há algumas que se especializam ainda mais, e exemplo das dedicadas especialmente a dados textuais estruturados — para documentos XML, por exemplo.

Neste trabalho serão usadas quatro técnicas de diferentes categorias: uma dedicada a dados binários, uma para dados textuais quaisquer e duas dedicadas a dados XML. Tais técnicas serão apresentadas nas subseções seguintes.

2.2.1.1 Codificação de Huffman

A codificação de Huffman é uma técnica de compressão de dados popular muito utilizada como base para programas de compressão de diversas plataformas (SALOMON, 2004). A codificação de Huffman se baseia na frequência com que conjuntos de bits do arquivo aparecem. Basicamente ela substitui um conjunto de bits mais frequentes por outro conjunto de bits menor, e faz um mapeamento desses bits para permitir o processo de descompactação.

Esta técnica foi desenvolvida por David Huffman e é baseada em duas informações: primeiro que, em um código ótimo, símbolos que ocorrem mais frequentemente serão substituídos por blocos de *bits* menores que os dos símbolos que ocorrem menos frequentemente. E segundo que, em um código ótimo, os dois símbolos que ocorrem menos frequentemente terão o mesmo tamanho (SAYOOD, 2012).

Métodos baseados em codificação de Huffman são bastante aplicáveis a dados binários, mas se comportam muito bem para dados textuais. Pretendemos observar um método baseado nessa técnica em nossos comparativos.

2.2.1.2 LZ77

Dados textuais se comportam bem quando comprimidos por métodos dedicados a dados binários, no entanto há propriedades nos dados textuais que normalmente não são exploradas por esses tipos de métodos, o que pode causar uma perda na eficiência da compressão. Para isso, existem métodos específicos para compressão de dados textuais. Os compressores LZ são dedicados à compressão de texto e há várias abordagens baseadas nesse tipo de técnica. Dentre eles, o LZ77.

> A compressão LZ77 recebe como entrada um fluxo de caracteres e produz um fluxo que intercala caracteres literais e ponteiros. Cada ponteiro indica uma frase e tem duas partes: um deslocamento e um comprimento. O deslocamento dá a distância de volta para a expressão, e o comprimento identifica o número de caracteres na frase. (FRASER, 2002).

Por exemplo, se o método recebe como entrada a frase: “Amanhã de manhã”, ele produzirá como saída o seguinte texto: “Amanhã de 9,5”, em que 9 é o número de caracteres de distância entre o ponteiro até o início da expressão e 5 é o comprimento da expressão compactada.

Para o escopo deste trabalho, é importante observarmos pelo menos um método de compressão de dados textuais simples, pois não raramente o tipo de arquivo que será analisado — SVG — pode conter uma grande sequência de caracteres que não estão diretamente ligadas à estruturação do arquivo.

2.2.1.3 Compressão de XML via DAGs

Os dados estruturados possuem propriedades específicas que podem ser aproveitadas para uma melhor eficiência na compressão. Como o objetivo deste trabalho é fazer um comparativo entre as técnicas de compressão que melhor se aplicam ao domínio de arquivos SVG, é importante considerar o fato de que ele se baseia em XML, e possui uma estrutura que pode ser aproveitada.

Um dos métodos desenvolvidos especificamente para tratar de arquivos XML é o presente no trabalho de Lohrey et al. (2013). Nesse trabalho, a compressão é observada levando-se em consideração estruturas de dados que sejam capazes de representar de forma compacta o conteúdo de um arquivo XML e, por consequência, gerar como saída serializada uma versão reduzida do arquivo de entrada.

O autor se baseia no fato de documentos XML possuírem uma estrutura convenientemente representável por árvores que chamaremos de *árvores ordenadas não*

limitadas[1], que são árvores enraizadas onde, apesar de necessariamente finito, não há um limite para o número de filhos que um nó possa possuir, além de obrigatoriamente esses estarem dispostos em uma sequência conhecida.

Foi mostrado que grafos direcionados e acíclicos (os DAGs) podem ser uma forma bastante compacta de representar a estrutura de documentos XML mais comuns (BUNEMAN; GROHE; KOCH, 2003) e, através do que o autor define como DAG mínimo, é possível remover redundâncias encontradas nesses documentos ao remover repetições de subárvores comuns.

Geralmente, tais árvores são representadas usando-se árvores binárias através de codificações que convertem a estrutura original numa estrutura binária. Para documentos XML uma codificação bastante comum é a *primeiro filho/próximo irmão* (KOCH, 2003) — do inglês, *first child/next sibling* — onde, para cada nó *v* da árvore original, na árvore codificada o filho esquerdo de *v* será seu primeiro filho e seu filho direito será o próximo irmão de *v*.

O autor observa os DAGs mínimos das árvores de documentos XML e conclui que em certas situações um DAG possui tamanho menor que o outro e em outras ocorre o contrário. Assim, é proposta uma abordagem híbrida, na tentativa de obter uma estrutura tão eficiente quando a melhor das duas anteriores.

Neste trabalho, pretendemos observar o comportamento dessa técnica aplicada a documentos SVG.

2.2.1.4 XMill

Outra abordagem de compressão dedicada ao XML escolhida para este trabalho foi o XMill (LIEFKE; SUCIU, 2000).

> O compressor XMill é designado para minimizar o tamanho dos documentos equivalentes ao XML. XMill incorpora algumas novas ideias de compressão específica do XML, que também são seguidos por outros compressores XML. O mais importante é separar a estrutura de dados e agrupar itens com significado relacionado. (LI, 2003).

Uma definição semelhante do XMill é dada por Salomon (2004), acrescentando que o principal objetivo dos desenvolvedores do XMill era criar um codificador que comprime arquivos XML mais eficientemente que um codificador típico. Segundo Salomon (2004), o XMill baseia-se nos seguintes princípios:

[1] Em tradução livre de *ordered unranked trees*.

- O XMill não é um compressor por si só. Ele é como um pré-processador que analisa o arquivo XML e utiliza diferentes compressores para comprimir as partes do arquivo, dependendo de qual forma seja melhor comprimir cada parte do arquivo, sendo que o Gzip é o compressor mais utilizado pelo XMill.
- As *tags* e os atributos do XML são comprimidos separadamente.
- Os itens relacionados do documento XML são agrupados em “recipientes”, sendo que itens do mesmo tipo — numérico, textual, etc. — são agrupados no mesmo recipiente.
- O XMill usa compressores de acordo com o que cada recipiente guarda. Por exemplo, um recipiente pode guardar números de telefone, outro as *tags* do XML, entre outros possíveis tipos de dados. O XMill utiliza um compressor específico tentando utilizar o compressor mais eficiente para cada recipiente.

3 TRABALHOS RELACIONADOS

Nesta seção, serão mencionados dois trabalhos relacionados. O primeiro é um artigo da dissertação de mestrado em que Teixeira (2011) apresentada duas novas abordagens de compressão de documentos XML. O segundo é um artigo publicado no *Journal of Computer and System Sciences* em 2009, que faz um levantamento de métodos de compressão específicos para documentos XML e realiza um experimento comparativo entre eles (SAKR, 2009).

3.1 Novas abordagens para compressão de documentos XML

Teixeira (2011) propôs duas novas abordagens de compressão de documentos XML. Tais abordagens foram testadas e seus resultados foram avaliados levando em consideração os seguintes fatores: taxa de compressão, tempo de compressão e tolerância dos métodos a baixa disponibilidade de memória.

Para comparativo, o autor selecionou técnicas de compressão de arquivos XML existentes que, segundo ele, se destacam na literatura. Os resultados dos experimentos foram comparados, e os comparativos demonstraram que a utilização de técnicas de compressão de documentos XML pode reduzir os impactos de desempenho criados pela linguagem.

Teixeira (2011) fez os testes com documentos XML, que estão dentro do grupo de dados textuais estruturados, e o presente trabalho tem como um de seus objetivos fazer algo semelhante ao que foi feito por Teixeira (2011), comparando métodos de compressão para

dados SVG, que são baseados em XML. No entanto, pretendemos também utilizar técnicas que não são específicas para XML.

3.2 XML compression techniques: A survey and comparison

Sakr (2009) fez um levantamento completo do estado da arte das técnicas de compressão específicas para XML e realiza um estudo experimental de nove dessas técnicas. Basicamente, são implementadas e com um grande conjunto de dados XML são feitos estudos comparativos, com o objetivo de ajudar desenvolvedores e usuários a escolher qual técnica de compressão de dados XML utilizar dependendo da necessidade.

Para selecionar os métodos a serem estudados, o autor levou em consideração os seguintes fatores: se a ferramenta de compressão era pública e livremente aberta, se era independente de esquema e se era capaz de executar no sistema operacional Ubuntu 7.10. Já para comparar as técnicas o autor levou em consideração taxa de compressão — que representa entre o tamanho do arquivo comprimido e o tamanho do arquivo original —, tempo de compressão — que é o tempo que cada método levou para comprimir totalmente o arquivo de entrada — e tempo de descompressão — que é o tempo que cada método levou para descomprimir totalmente o arquivo de que havia sido comprimido.

Sark (2009) realizou um comparativo entre técnicas de compressão específicas para documentos XML, semelhante ao que será feito neste trabalho — comparação de métodos de compressão de diferentes categorias ao domínio de dados do tipo SVG. No entanto, assim como Teixeira (2011), ele não levou em consideração técnicas de compressão não dedicadas a dados XML, além do domínio de dados ser diferente.

4 OBJETIVOS

4.1 Objetivo geral

Temos como objetivo comparar métodos de compressão de dados textuais estruturados aplicando-os a arquivos SVG, levando em consideração fatores como custo de tempo, de espaço, taxa de compressão, dentre outros aplicáveis, de forma a situar uma boa opção de método.

2.2 Objetivos específicos

- Fazer estudo conceitual e técnico sobre a área de compressão de dados.
- Selecionar quatro métodos de compressão de dados, de categorias distintas, a serem estudados.
- Implementar os métodos selecionados sob as mesmas condições de tecnologias.
- Realizar comparativos de performance entre os algoritmos escolhidos, levando em consideração os fatores já citados: custo de tempo, espaço, taxa de compressão, entre outros.
- Definir a melhor opção sobre o domínio das classes de dados textuais estruturados escolhidas.

5 PROCEDIMENTOS METODOLÓGICOS

5.1 Realizar revisão bibliográfica e estudar a área de compressão de dados

Nesta fase faremos uma revisão bibliográfica sobre a área de compressão de dados, pesquisando trabalhos publicados nos principais eventos e periódicos nessa área. A pesquisa também será estendida a livros sobre o assunto ou qualquer outra fonte confiável.

Essa revisão bibliográfica tem como objetivo um aprofundamento do estudo na área de compressão de dados, frisando principalmente os dados textuais estruturados, que são o foco deste trabalho.

5.2 Elencar os principais métodos de compressão aplicáveis ao domínio escolhido

Inicialmente, faremos um levantamento da literatura científica sobre métodos de compressão de dados existentes, levando em consideração se tais abordagens são aplicáveis ao domínio escolhido — dados do tipo SVG. Desse conjunto de métodos, elencar os principais segundo análise de simplicidade e disponibilidade.

Esse conjunto de técnicas selecionadas será analisado e desse conjunto serão escolhidos alguns, provavelmente quatro, de diferentes categorias — um específico para dados textuais simples, dois dedicados a documentos XML e um para dados binários — para que possam ser feitos os testes comparativos e assim obtidos os resultados necessários para a escolha do melhor.

5.3 Analisar as técnicas e tecnologias

A partir do conjunto de métodos selecionados na fase inicial, serão analisadas as técnicas e tecnologias empregadas para implementá-los. Para que a comparação seja justa, deve-se colocar os métodos em igualdade. Consideramos colocar os métodos em igualdade quando obedecemos às seguintes condições:

- Implementá-los utilizando os mesmos componentes de hardware e software — mesma máquina, mesmo sistema operacional e mesma linguagem de programação.
- Utilizar as mesmas instâncias de entrada para os testes comparativos.

5.4 Montar ou utilizar repositório de testes disponível

Após a análise das técnicas, e obtenção do conhecimento de todas as suas propriedades, serão obtidas as entradas para que possam ser utilizadas como instâncias para os métodos. O repositório conterá uma quantidade significativa de arquivos do tipo SVG que poderão ser usados para os comparativos.

O repositório poderá ser montado de duas formas:

- Arquivos de testes buscados de repositórios livres já existentes.
- Arquivos de testes gerados para tentar aproveitar todas as propriedades do arquivo SVG e das técnicas que irão comprimi-los.

5.5 Implementar os métodos

Os métodos escolhidos serão implementados em uma mesma linguagem de programação, serão testados utilizando a mesma máquina, entre outras restrições, mas sempre os colocando em igualdade de condições de execução.

A linguagem em que as técnicas de compressão escolhidas serão implementadas ainda não está definida, no entanto a opção mais provável é C++. A escolha da linguagem deverá levar em consideração a facilidade de implementação, a disponibilidade de ferramentas auxiliares e o impacto gerado sobre o desempenho dos métodos.

5.6 Executar os métodos

Após os métodos implementados e as entradas geradas, serão feitos os experimentos comparativos. Cada um dos métodos será executado com todas as entradas e os dados de desempenho serão coletados.

Para isso, pretendemos realizar várias execuções do método com a mesma entrada e contabilizar as médias dos experimentos, de forma a minimizar os efeitos da influência de fatores alheios ao nosso controle.

5.7 Eleger melhor método

Nesta fase, serão analisados e comparados todos os dados coletados na fase de execução. Os métodos serão comparados em função dos alguns quesitos pré-estabelecidos, dentre eles podemos citar custo de tempo, espaço e taxa de compressão.

Depois de feita a comparação, elegeremos um dos métodos, o que melhor se adeque ao domínio escolhido, junto a uma análise crítica obtida dos resultados conseguidos em cada quesito analisado.

5.8 Cronograma de execução

ATIVIDADES	2015													
	Mai		Jun		Jul		Ago		Set		Out		Nov	
Revisão bibliográfica, definição de escopo de pesquisa e estudo da área de compressão de dados e métodos de compressão de dados textuais estruturados.	x	x	x	x	x	x								-
Elencar os principais métodos de compressão aplicáveis ao domínio escolhido			x	x										
Defesa do projeto.				x										
Analisar as técnicas e tecnologias				x	x	x								-
Montar ou utilizar repositório de testes disponível				x	x	x	x							-
Implementar os métodos						x	x	x	x	x	x			
Iniciar redação da monografia									x	x	x			
Executar os métodos										x	x			
Eleger melhor método											x	x		
Revisão final da monografia											x	x	x	-
Defesa do Trabalho Final													x	-

REFERÊNCIAS

ALMEIDA, Mauricio Barcillos. **Uma introdução ao XML, sua utilização na Internet e alguns conceitos complementares.** Ciência da Informação, v. 31, n. 2, p. 5-13, 2002.

ARASU, Arvind; GARCIA-MOLINA, Hector. **Extracting structured data from web pages.** In: Proceedings of the 2003 ACM SIGMOD international conference on Management of data. ACM, 2003. p. 337-348.

BUNEMAN, Peter; GROHE, Martin; KOCH, Christoph. **Path queries on compressed XML**. In: Proceedings of the 29th international conference on Very large data bases-Volume 29. VLDB Endowment, 2003. p. 141-152.

COULOURIS, George F.; DOLLIMORE, Jean; KINDBERG, Tim; BLAIR, Gordon. **Sistemas distribuídos: conceitos e projeto.** 5. ed. Porto Alegre: Bookman, 2013. xvi, 1048 p. ISBN 9788582600535 (broch.).

FERRAIOLO, Jon; JUN, Fujisawa; JACKSON, Dean**. Scalable vector graphics (SVG) 1.0 specification.** iuniverse, 2000.

FRASER, C. **An instruction for direct interpretation of LZ77-compressed programs.** In: MSR-TR-2002-90, Microsoft Research. 2002.

GOMES, Elma Pereira**. ANÁLISE DOS RECURSOS DE ACESSIBILIDADE DO SCALABLE VECTOR GRAPHICS**. 2005. Monografia (Graduação em Sistemas de Informação) – Centro Universitário Luterano de Palmas, Universidade Luterana do Brasil, Palmas. 2005. Disponível em: < http://arquivo.ulbra-to.br/ensino/43020/artigos/relatorios2005-2/Arquivos/Elma%20P%20G%20-%20Trabalho%20de%20Conclusao%20de%20Curso.pdf>. Acesso em: 15 jun 2015.

KOCH, Christoph. **Efficient processing of expressive node-selecting queries on XML data in secondary storage: A tree automata-based approach.** In: Proceedings of the 29th international conference on Very large data bases-Volume 29. VLDB Endowment, 2003. p. 249-260.

LI, Weimin. **Xcomp: An XML compression tool.** 2003. Tese de Doutorado. University of Waterloo,[School of Computer Science].

LIEFKE, Hartmut; SUCIU, Dan. **XMill: an efficient compressor for XML data.** In: ACM Sigmod Record. ACM, 2000. p. 153-164.

LOHREY, Markus; MANETH, Sebastian; NOETH, Eric**. XML compression via DAGs.** In: Proceedings of the 16th International Conference on Database Theory. ACM, 2013. p. 69-80.

SAKR, Sherif. **XML compression techniques: A survey and comparison.** Journal of Computer and System Sciences, v. 75, n. 5, p. 303-322, 2009.

SALOMON, David. **Data compression: the complete reference.** Springer Science & Business Media, 2004.

SAYOOD, Khalid. **Introduction to data compression.** Newnes, 2012.

TEIXEIRA, Márlon A. C. **Novas Abordagens para Compressão de Documentos XML.** 2011. Dissertação (Mestrado em Engenharia Elétrica) - Departamento de Comunicações, Universidade Estadual de Campinas, Campinas. 2011. Disponível em: <http://www.bibliotecadigital.unicamp.br/document/?code=000841713&fd=y>. Acesso em: 15 jun 2015.