Revista de Sistemas de Informação da FSMA
n. 15 (2015) pp. 30-39

SISTEMAS DE INFORMAÇÃO
http://www.fsma.edu.br/si/sistemas.html

TRILHA ESTUDANTIL

Visualização em Big Data: Uma Ferramenta para Auxiliar no Reconhecimento de Padrões em Fluxo Contínuo de Dados

Victor Hugo Andrade Soares, *Bacharel em Sistemas de Informação, UFV*,
Joelson Antônio dos Santos, *Bacharel em Sistemas de Informação, UFV* e
Murilo Coelho Naldi, *Dr. Prof. Adjunto-III, UFV*
{victorhugoasoares, joelsonn.santos}@gmail.com, murilocn@ufv.br

***Resumo*—O desenvolvimento de novas tecnologias é responsável pela geração e armazenamento de quantidades contínuas e massivas de dados. Esses tipos de dados são conhecidos como fluxo contínuo de dados. A análise dos fluxos de dados pode ser vantajosa para as várias áreas, como bioinformática, medicina e empresas, uma vez que pode resultar em informações importantes sobre os dados. Neste trabalho, propomos uma nova ferramenta para Visualização de Dados, que permite a análise da evolução dos agrupamentos em tempo real, durante a transmissão de dados. A ferramenta de visualização proposta é um complemento para o SAMOA, uma versão do MOA (*Massive Online Analysis*) para a mineração de grandes fluxos de dados e processamento distribuído.**

***Palavras-chave*—Mineração de Dados, Fluxos Contínuos de Dados, Visualização de Dados.**

Visualization in Big Data: A tool for pattern recognition in data stream.

***Abstract*—The development of new technologies is responsible for the generation and storage of continuous and massive amounts of data. Such type of data is known as data stream. The analysis of data streams may be advantageous in many fields, like bioinformatics, medicine, companies and others, as it may result in important information about the data. In this work, we propose a new software tool for Data Visualization that permits the analysis of the evolution of clusters in real time during the data streaming. The proposed visualization tool is add-on for SAMOA, a new variant of MOA (Massive Online Analysis) for massive data streams mining and processing distribution.**

***Index Terms*—Data Mining, Data Streams, Data Visualization.**

I. Introdução

A constante evolução e uso de novas tecnologias de informação mudou a maneira com que as pessoas se comunicam e solucionam problemas pessoais e profissionais. Devido a esse uso, há um acúmulo de dados gerados diariamente, a partir de interações de consumo entre clientes e empresas, relações sociais em rede entre amigos ou transações bancárias [1]. A capacidade de coletar e analisar dados em tempo real é chamada de fluxo contínuo de dados [2].

O desafio de se trabalhar com Visualização de Dados, nesse contexto, é analisar os tipos de dados com velocidade, a fim de encontrar padrões e auxiliar em tomadas de decisões [3]. Além disso, a quantidade de dados em um fluxo contínuo pode ser finita ou infinita, onde essa imprevisibilidade do volume de dados torna a análise ainda mais custosa [4].

Mecanismos capazes de automatizar os processos de análise de dados são necessários e nesse cenário são empregadas técnicas de mineração de dados. A mineração de dados é constituída por técnicas de inteligência artificial e técnicas estatísticas, que visam extrair informações úteis de uma determinada quantidade de dados que podem estar armazenados ou em fluxo contínuo [5].

Uma das técnicas para auxiliar na identificação de padrões em dados é o agrupamento de dados. Trata-se de um conjunto de métodos não supervisionado que visa agrupar objetos de acordo com suas características. Isso pode ser obtido por meio de medidas de dissimilaridades ou similaridades entre os objetos analisados [6].

Uma das características dos fluxos de dados é que eles podem ser infinitos, ou seja, não é possível mantê-los armazenados. Após uma dada análise, os dados devem ser descartados para que novos dados possam ser recebidos. Essa volatilidade dificulta o reconhecimento de novas tendências nos agrupamentos, pois os dados não podem ser reprocessados. Somente o conhecimento final de cada agrupamento é armazenado [4].

Existem técnicas na literatura criadas para detecção de mudanças em fluxos de dados. O M-DBScan [7], por exemplo, compara a entropia, ou seja, o grau de incerteza de um novo grupo com a entropia dos grupos gerados anteriormente. A mudança na entropia caracteriza mudança no padrão dos dados. O objetivo deste trabalho é possibilitar a visualização do comportamento dos dados no decorrer do fluxo, independente da ocorrência de mudanças, portanto, não foram utilizadas técnicas de agrupamento para

detecção de mudanças.

Uma vez que os dados forem agrupados, é necessário analisar os resultados a fim de reconhecer padrões e extrair conhecimento dos mesmos, porém, a análise de dados abstratos não é intuitiva. A visualização de dados surgiu com o objetivo de facilitar o entendimento e percepção de determinadas características em conjunto de dados abstratos. O sistema perceptual humano é sensível a diferenças de cores, tamanhos, formatos e distâncias entre objetos. Por isso, a Visualização de Dados faz uso de recursos gráficos para ilustrar os atributos dos dados. Isso auxilia o usuário na percepção de semelhanças e diferenças nos mesmos [8].

Este artigo tem como objetivo apresentar uma ferramenta de visualização intitulada *2D Data Stream Viewer*, desenvolvida para auxiliar na identificação de padrões e tendências em dados coletados em fluxo contínuo. Existem algumas plataformas para análise de fluxos contínuos de dados, duas delas são: *Apache Spark Streaming* e SAMOA (do inglês *Scalable Advanced Massive Online Analysis*) [9]. A *Spark Streaming* é uma plataforma de análise distribuída de dados e uma recente extensão da plataforma *Apache Spark* [10]. Já a plataforma SAMOA, além de ser uma plataforma distribuída de análise de fluxos contínuos de dados, é também uma biblioteca de algoritmos de mineração de fluxos contínuos de dados. Como vantagem, a SAMOA permite a análise dos dados em tempo real. Essa característica tem impacto na baixa latência quando se refere ao processamento de grandes quantidades de dados. Já a plataforma *Spark Streaming* tem como característica o armazenamento de partes dos dados em *batch* para análise posterior [10].

A ferramenta de visualização proposta neste artigo é um complemento da plataforma SAMOA. A SAMOA contém uma versão distribuída e massiva do algoritmo de agrupamento de dados *KMeans*, que por sua vez é implementado e adaptado para o modelo de fluxo contínuo. Diferente de outros algoritmos como: DBSCAN [11], SLINK e seus variantes [12], que possuem complexidade computacional quadrática, o algoritmo *KMeans* possui complexidade linear [5]. Logo, o tempo de processamento dos dados por esse algoritmo torna-se mais escalável do que vários outros algoritmos de agrupamento de dados. Por esse motivo, utilizamos o algoritmo *KMeans* no agrupamento dos dados em fluxo contínuo. O resultado do agrupamento permite a coleta e armazenamento das informações necessárias a serem ilustradas na ferramenta de visualização proposta.

A Seção II contém a base teórica utilizada no desenvolvimento da ferramenta de visualização e no algoritmo de agrupamento. Na Seção III, será apresentado o desenvolvimento e as funcionalidades da ferramenta proposta. Por fim, a Seção IV, tratará das considerações finais.

II. Fundamentação Teórica

NESTA seção são apresentados os principais conceitos utilizados no desenvolvimento da ferramenta proposta. Dentre eles, conceitos sobre fluxo de dados, técnicas de agrupamento, uma introdução sobre as técnicas de Visualização de Dados e o uso da visão humana na detecção de padrões.

A. Fluxo de Dados

Existem aplicações que ao serem executadas exigem que os dados sejam coletados e processados em tempo real. Nesses casos, o armazenamento e consultas em tabelas persistentes de bancos de dados convencionais são inviáveis nesses casos [13]. Também não é possível utilizar técnicas como a amostragem, pelo fato de não haver uma quantidade definida de dados a serem recebidos. Essas características definem os dados em fluxo contínuo [5].

Formalmente a definição de fluxo contínuo de dados consiste em sequências de pontos *X1,....,Xk*, onde *k* é indefinido. Cada *Xi* tem um tempo de chegada *T1,...,Tk*, onde cada ponto possui *d* dimensões e pode ser denotado por *Xi = (Xi1,...,Xid)* [14].

No processamento de dados em fluxo é preciso lidar com tabelas de valores transitórios. Essas tabelas são preenchidas com os dados coletados em um fluxo contínuo. Essa fase de coleta é chamada de fase *online*. Após o preenchimento dessas tabelas, é feita a análise dos dados coletados, conhecida como fase *offline*. A análise dos dados gera um conhecimento que deve ser armazenado para futuras tomadas de decisões. Encerrada a fase *offline*, os dados são excluídos das tabelas para que novos dados possam ser coletados. As tabelas recebem o nome de transitórias pois abrigam os dados apenas por uma fração de tempo, até que os dados sejam analisados e então descartados ou persistidos em bancos de dados. Esta transitoriedade é o que diferencia a análise em fluxo para a análise em tabelas persistentes [4].

Essa volatilidade dos dados dificulta a análise na obtenção de resultados satisfatórios. Considerando que os dados estão disponíveis por um curto intervalo de tempo e então são descartados ou persistidos, torna-se impossível reprocessá-los em tempo hábil. Isso prejudica na adaptação de novos modelos de decisão que venham a surgir com novas características dos dados recebidos [15]. A fim de amenizar tais dificuldades, serão apresentadas funcionalidades da ferramenta de visualização *2D Data Stream Viewer* voltadas para a percepção de novas características nos dados do fluxo contínuo na Seção III.

B. Agrupamento de Dados

Seres humanos possuem a capacidade de categorizar e agrupar objetos ao seu redor na medida em que são observados. Pode-se citar como exemplo um funcionário de uma plantação que tem por objetivo separar as frutas de acordo com seus tamanhos. Logo, as frutas menores ficarão em um grupo separado das frutas médias e grandes. Essa tarefa se dá com relativa facilidade pela maioria das pessoas. De forma análoga, a mineração de dados possui técnicas capazes de agrupar objetos de acordo com suas semelhanças, conhecidas como agrupamento de dados.

As técnicas de agrupamento de dados visam encontrar semelhanças entre objetos em conjuntos de dados [16]. Esse processo é realizado por meio de cálculos de distâncias entre pontos no espaço n-dimensional dos dados, onde a dimensão dos dados é caracterizada como a quantidade de atributos contidos no conjunto. Embora pareça simples, há subjetividade na definição do que realmente é um grupo [5]. Por exemplo, a Figura 1 mostra que existem diversas maneiras de agrupar um mesmo conjunto de dados, a Figura 1(a) representa o conjunto de dados original, a Figura 1(b) mostra dois grupos distintos, e a Figura 1(c) mostra três grupos distintos. Neste exemplo, cada ponto pertence a um único grupo de maneira integral, porém, existem algoritmos que lidam com sobreposição de grupos, onde um ponto pode pertencer a mais de um grupo. Neste trabalho trataremos apenas de partições rígidas, ou seja, sem sobreposição de grupos.

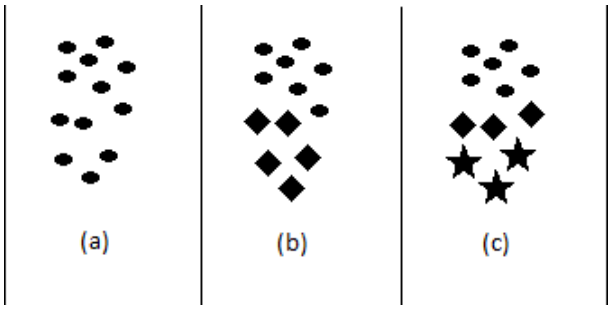

Fig. 1. Formas diversas de se agrupar um mesmo conjunto de dados.
Fonte: Adaptado de [5].

Há diferentes tipos de agrupamentos de dados [17]. Dentre os mais comuns está o agrupamento de dados baseado em protótipos, onde os objetos são mais semelhantes ao protótipo de seu grupo e menos semelhantes aos protótipos de outros grupos. Também existe o agrupamento de dados baseado em densidade, em que os objetos são agrupados de acordo com sua concentração em determinado ponto no espaço n-dimensional, ou seja, a alta densidade dos objetos formam grupos. Outro tipo é o agrupamento de dados baseado em grafos, onde existem nós (objetos) que são conectados por arestas que representam a relação entre eles.

C. Kmeans

O *Kmeans* (ou k-médias) é um algoritmo de agrupamento de dados particional baseado em protótipos [18]. O *Kmeans* visa encontrar grupos por meio de centróides, que são formados pelas médias entre os objetos de um mesmo grupo. Algum tipo de similaridade é usada para definir grupos, de forma que um determinado objeto é mais semelhante aos objetos de seu grupo e menos semelhante aos objetos de outros grupos no conjunto de dados.

O algoritmo *Kmeans* possui características simples e conta com vários métodos de implementação disponíveis na literatura. O parâmetro k, que indica a priori a quantidade de grupos que se deseja obter no agrupamento, deve ser definido pelo usuário ou estimado. Uma das técnica para estimar o valor k utiliza algoritmos evolutivos [19]. Na Tabela I são apresentados os passos para implementação do algoritmo *Kmeans* clássico, onde o atributo k é um parâmetro de entrada definido pelo usuário. A Figura 2 ilustra um exemplo de execução do algoritmo *Kmeans* para $k = 3$.

TABELA I
Algoritmo *KMeans* Clássico

Algoritmo *KMeans* Clássico
1: Escolhe k centróides iniciais aleatoriamente definidos pelo usuário.
2: Calcula a distância de cada objeto aos centróides.
3: Atribui cada objeto a seu centróide mais próximo.
4: Calcula a média de cada grupo e atribui novas posições aos centróides.
5: Repete passo 2, 3 e 4, até que nenhum centróide mude de posição.

Fonte: Adaptado de [18].

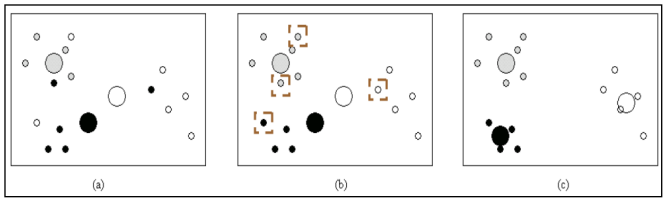

Fig. 2. Exemplo de execução do algoritmo *KMeans*. (a) Cada elemento foi designado para um dos três grupos aleatoriamente e os centróides (círculos maiores) de cada grupo foram calculados. (b) Os elementos foram designados agora para os grupos cujos centróides lhe estão mais próximos (c) Os centróides foram recalculados. Os grupos já estão em sua forma final. Caso não estivessem, repetiríamos os passos (b) e (c) até que estivessem.
Fonte: [6].

Relações de proximidade ou medidas de similaridades são utilizadas para definir a real distância entre os objetos em relação aos centróides [18]. As medidas de similaridade mais conhecidas e utilizadas partem da métrica de Minkowski[6], que é dada pela seguinte fórmula:

$$d(x, y) = \sqrt[q]{\sum_{i=1}^{t} (x_i - y_i)^q} \quad (1)$$

Na métrica de Minkowski, quando $q = 2$, temos a conhecida distância euclidiana, que busca semelhanças entre objetos no espaço euclidiano. Quando $q = 1$, temos a chamada distância de Manhattan (ou também, distância de block city)[5]. Neste trabalho foi implementado um algoritmo *KMeans* adaptado para trabalhar com dados em fluxo contínuo na plataforma distribuída SAMOA, apresentada na Seção II-D. Nos cálculos de similaridade foi utilizada a distância euclidiana. As adaptações implementadas no algoritmo serão descritas na Subseção III-A.

D. SAMOA

O *Scalable Advanced Massive Online Analysis* (SAMOA) é uma plataforma distribuída que também contém

uma biblioteca de algoritmos de mineração de fluxos contínuos de dados [9]. O SAMOA é capaz de se adaptar a diferentes plataformas de processamento distribuídos de dados em fluxos contínuos, são elas *S4* e *Storm* [9].

O SAMOA possui alguns algoritmos implementados, dentre eles está o algoritmo de agrupamento de dados *Clustream*. O *Clustream*, que também é referido como *framework*, possui duas fases de processamento dos dados em fluxo: *online* e *offline*. A fase *online* é o processo de sumarização dos pontos que chegam continuamente para a fase de análise. Já a fase *offline* consiste no agrupamento dos pontos coletados na fase anterior [14]. Adicionalmente, o SAMOA também garante a seus usuários a facilidade de implementação de novos algoritmos de análise de dados e permite desenvolver conexões para outras plataformas [9].

Uma vez que a análise de agrupamento é uma tarefa complexa, somente uma técnica para resolução dos problemas de agrupamento não basta [5]. Logo, um dos objetivos deste trabalho consistiu em aumentar a gama de algoritmos de agrupamento de dados na plataforma SAMOA. Na Subseção III-A é descrita a implementação do tradicional algoritmo *KMeans* adaptado para análise de fluxo contínuo.

A plataforma SAMOA disponibiliza *plugins* para a captura de diferentes fluxos de dados reais e sintéticos. Dados reais demonstram uma situação ou aplicação do mundo real. Geralmente esses dados são estacionários, ou seja, são armazenados em algum repositório de dados. Quando se refere a fluxo contínuo, os dados surgem em intervalos de tempo distintos, de uma ou mais fontes e são processados em tempo real. Um exemplo de aplicação real é da empresa *Yahoo*, que utiliza a plataforma SAMOA para detecção de *spams* em e-mails [20].

Já os dados sintéticos, em tese, simulam uma sequência aleatória e indefinida de pontos. Essa aleatoriedade é configurável, o que garante uma flexibilidade na geração desses dados e permite simular fluxos com diferentes configurações [21]. Logo, a flexibilidade dos dados sintéticos tornou-se interessante nos experimentos deste trabalho, permitindo testar a ferramenta de visualização proposta com diferentes configurações de dados.

A diferença em se utilizar dados artificiais é que eles são gerados normalizados. Com o algoritmo implementado na plataforma SAMOA, os *plugins* para coleta dos dados reais já estão disponíveis. A adaptação do algoritmo para análise de dados reais depende somente do pré-processamento e normalização dos dados brutos coletados. Uma vez pré-processados, os dados reais podem ser agrupados e analisados assim como os dados artificiais.

E. Visualização de Dados

Visualização de Dados é uma sub-área da Computação Gráfica que busca auxiliar o usuário na análise e compreensão de dados abstratos [22]. Esses dados são geralmente obtidos a partir da mineração de um grande conjunto de dados [23]. O uso de tais técnicas tira vantagem do sistema visual humano na tarefa de identificar padrões e tendências presentes nos dados [24]. A Visualização de Dados faz uso de elementos básicos que o sistema perceptual humano consegue assimilar com rapidez, como cor, tamanho, forma, proximidade e movimento [8]. O fato de o ser humano perceber rapidamente tais características, permite utilizá-las para representar cada atributo dos dados. Com isso, é possível não somente assimilá-los com rapidez, mas também, representar uma grande quantidade de dados de uma só vez. Quando grande quantidade de dados são apresentados visualmente, é possível perceber grupos, lacunas, pontos máximos, pontos mínimos e várias outras características [8].

A obtenção de uma representação visual de um conjunto de dados pode ser generalizada em um processo automatizado. Os dados abstratos são coletados; a ferramenta de Visualização de Dados processa esses dados e gera ao usuário uma imagem representativa [25]. A Figura 3 ilustra esse processo.

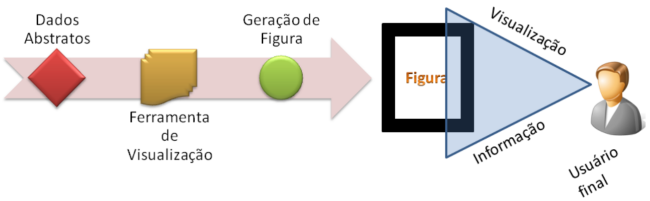

Fig. 3. Processo geral para gerar representação visual de dados. Fonte: [25].

Diversas ferramentas de visualização são encontradas na literatura, com o objetivo de tornar intuitiva a análise de conjuntos de dados abstratos. Isso inclui a utilização de diferentes técnicas de visualização [26], como as orientadas a pixels, hierárquicas, baseadas em gráfos, etc.

Com a existência de inúmeros métodos de coleta, processamento e análise de dados, as ferramentas de visualização ainda não são capazes de ilustrar algumas situações específicas de análise, como é o caso da análise de fluxo contínuo de dados.

Considerando as principais ferramentas de visualização em gráfico 2D encontradas na literatura, como é o caso da *Weka* [27], *MGV* [28] e Polaris [29], as três ferramentas utilizam dados armazenados em tabelas persistentes para o processamento e visualização dos dados. A *Weka* implementa diversos algoritmos para mineração de dados, como classificação, agrupamento, associação e validação de resultados, porém, não permite a coleta e visualização de dados em fluxo contínuo.

Para visualização de dados em fluxo contínuo, [30] propõe uma técnica para escalonamento multidimensional, com o objetivo de reduzir o tempo computacional necessário para se processar os dados coletados e ilustrar o resultado do fluxo contínuo em gráfico 2D. No entanto, a técnica não permite o acompanhamento visual do fluxo, a fim de perceber mudanças e tendências na linha do tempo em que o fluxo foi coletado. O presente trabalho aborda o desenvolvimento de uma ferramenta para Visualização de Dados em gráfico 2D, que é um complemento para

a SAMOA. Essa ferramenta permite a análise de dados coletados e agrupados em fluxo contínuo. Na Subseção III-B deste artigo serão apresentados os detalhes do desenvolvimento dessa ferramenta.

III. Desenvolvimento e Apresentação

NESTA seção serão descritos os passos para a adaptação do algoritmo *KMeans* para o modelo de fluxo contínuo e o armazenamento dos dados necessários que serão utilizados na ferramenta de visualização. Também será apresentado o desenvolvimento da ferramenta de visualização *2D Data Stream Viewer* e as funcionalidades implementadas nesta ferramenta.

A. Algoritmo de Agrupamento

Foi escolhida a plataforma SAMOA para o desenvolvimento do algoritmo *KMeans* adaptado para fluxo contínuo, que proporcionou algumas vantagens. Uma delas, é que a plataforma tem como função manter a abstração da comunicação de rede que é necessária em sistemas distribuídos convencionais. Com isso, os desenvolvedores precisam apenas elaborar a lógica das aplicações, sem se preocupar com a manipulação dos protocolos de redes envolvidos. Outra vantagem é que a plataforma é desenvolvida sobre a linguagem de programação *Java*. Logo, assim como as demais linguagens que utilizam o conceito de reusabilidade do paradigma de orientação a objetos, é possível o desenvolvimento de novas aplicações sem redundância na codificação.

Na Tabela II são apresentados os passos do algoritmo *Kmeans* adaptado para fluxo contínuo de dados. Enquanto no algoritmo clássico o usuário precisa entrar apenas com o parâmetro k, que representa a quantidade de grupos a serem encontrados, nesta versão o algoritmo espera receber três parâmetros: k, x e t.

Assim como no algoritmo clássico, o parâmetro k é a quantidade de grupos a serem encontrados. O parâmetro x é a quantidade limite de dados que podem ser agrupados de uma só vez, antes de serem descartados. Uma ideia seria limitar o tamanho de x ao tamanho da memória de trabalho disponível, mas o valor pode ser definido arbitrariamente. O parâmetro t limita o tempo máximo de espera do algoritmo durante a coleta dos dados. Se a quantidade x de dados não for coletada em tempo t, a fase de agrupamento é executada, sendo utilizados apenas os dados já coletados.

Esses parâmetros possuem papéis importantes no contexto da análise de fluxo de dados. Como não se conhece a quantidade de dados que serão recebidos, o parâmetro x evita que ocorra estouro da memória de trabalho, ou simplesmente, divide o fluxo em lotes de dados com tamanho padronizado. O parâmetro t resolve o possível problema de ociosidade do algoritmo. Caso o fluxo de dados termine ou tenha uma longa pausa, limitar um tempo t evita que o algoritmo fique ocioso por um longo tempo, ou até mesmo que a execução não prossiga. Logo, se o algoritmo fica ocioso por um tempo t, o agrupamento é iniciado.

TABELA II
Algoritmo *KMeans* para Fluxo de Dados

Algoritmo *KMeans* para Fluxo de Dados
1: Coleta os dados do fluxo até que a tabela de tamanho x seja preenchida ou o tempo t seja atingido.
2: Se for a primeira iteração, escolhe k medóides iniciais aleatoriamente. A partir da segunda iteração, considerar os centróides da iteração anterior.
3: Calcula a distância de cada objeto aos centróides.
4: Atribui cada objeto a seu centróide mais próximo.
5: Calcula a média de cada grupo e atribui novas posições aos centróides.
6: Repete passo 3, 4 e 5 até que nenhum centróide mude de posição.
7: Calcula o *SSE* de cada grupo gerado e o *SSE* do agrupamento total.
8: Armazena em arquivo as posições finais dos centróides, a quantidade de objetos em cada grupo, o *SSE* de cada grupo e o *SSE* total do agrupamento.
9: Exclui os dados coletados, volta ao passo 1 e repete todo o algoritmo até que termine o fluxo.

Neste trabalho, as instâncias são geradas pela classe *RandomRBFGeneratorEvents* do SAMOA e seus atributos normalizados em um intervalo de 0 a 1. As instâncias são geradas simulando um fluxo contínuo, mas a quantidade de instâncias pode ser definida pelo usuário. Definido pelos desenvolvedores da plataforma, cada objeto gerado por essa classe possui sempre duas dimensões.

Além dos passos tradicionais que se assemelham aos do algoritmo *KMeans* clássico, na primeira execução do algoritmo as posições dos k centróides são escolhidas pelas posições de k medóides aleatórios, ou seja, pelas posições de k elementos contidos na base de dados. Após a execução do *KMeans*, é feito o armazenamento dos centróides finais de cada grupo, quantidade de objetos, *SSE* (*sum squared error*) de cada dimensão dos grupos individuais e *SSE* do agrupamento total. Com isso, os objetos são excluídos para que novos possam ser coletados e iniciar um novo agrupamento.

O *SSE* é uma das métricas mais comuns na avaliação de agrupamentos [5]. O cálculo do *SSE* de um grupo é mostrado da Equação 2.

$$SSE_{grupo} = \sum_{x \in C_i} dist^2(m_i, x) \tag{2}$$

No cálculo, x é um ponto que pertence ao grupo C_i, e m_i é a posição do centróide desse grupo. Já o cálculo do *SSE* do agrupamento completo é dado pela soma dos k *SSE* encontrados nos grupos conforme mostra a Equação 3.

$$SSE_{total} = \sum_{j=1}^{k} \sum_{x \in C_i} dist^2(m_i, x) \tag{3}$$

Na aplicação, o *SSE* é calculado também para cada dimensão dos atributos. Por exemplo: em dados de duas dimensões, é armazenado o *SSE* de cada grupo, mostrado na Equação 2 e também o *SSE* da dimensão 1 de cada grupo e o *SSE* da dimensão 2. A Equação 4 mostra como é feito o cálculo do *SSE* para uma única dimensão. O *SSE*

da dimensão t é o somatório das distâncias ao quadrado da dimensão t do ponto x até a dimensão t do centróide m_i. Isso é feito para todas as d dimensões dos dados.

$$SSE_{Dim_t} = \sum_{x \in C_i} dist^2(m_{i_t}, x_t) \quad \forall \quad t \in d \qquad (4)$$

A métrica é importante para avaliar a qualidade do agrupamento. Quanto menor o *SSE* do grupo, menor é a dispersão dos pontos e melhor é o agrupamento. Os *SSE*'s de cada dimensão também são calculados por permitir perceber em qual dimensão os dados são mais dispersos. Além do *SSE*, existem outras métricas de qualidade de agrupamentos, como homogeneidade, densidade dos grupos, valorização dos máximos e mínimos da distância intra-grupos, entre outras.

A partir do segundo agrupamento do fluxo de dados, os centróides iniciais não são mais aleatórios. Serão utilizadas as posições finais dos centróides no último agrupamento realizado. Isso é feito para que não haja troca de posições entre dois ou mais centróides. Isso torna possível, ao utilizar a ferramenta de visualização, a percepção de movimentação de determinado grupo durante o fluxo. As informações dos agrupamentos são armazenadas sequencialmente em forma de histórico. Ao fim de cada agrupamento, um arquivo de texto com extensão ".vjm" é atualizado com as informações do novo agrupamento realizado. Essa extensão foi criada especificamente para a ferramenta de visualização, permitindo a diferenciação dos arquivos suportados pela ferramenta dos arquivos de textos convencionais.

Cada fluxo de dados gera um único arquivo ".vjm". Durante o fluxo são executados vários agrupamentos, cada um deles com uma porção de dados do fluxo. As informações finais de cada agrupamento são armazenadas sequencialmente no arquivo ".vjm". O arquivo é utilizado na ferramenta *2D Data Stream Viewer*, que possibilita a análise visual do desenvolvimento dos grupos, uma vez que é possível identificar padrões e tendências nos dados. A Figura 4 mostra como os dados devem ser gravados no arquivo de texto ".vjm". Os textos após o caractere '%' referem-se a comentários.

```
% Arquivo "vjm" com informações de um fluxo de dados
% Número de k (grupos)
4;

% Agrupamento 1

% Coordenadas X e Y dos 4 centróides
[[0.225, 0.104], [0.723, 0.256], [0.343, 0.876], [0.978, 0.768]];
% SSE do eixo X dos 4 grupos
[1.341, 0.945, 2.549, 1.023];
% SSE do eixo Y dos 4 grupos
[0.984, 0.835, 2.232, 1.254];
% SSE dos 4 grupos
[3.451, 5.768, 6.295, 3.275];
% Quantidade de objetos por grupo
[233, 167, 356, 244];

% Agrupamento 2

% Coordenadas X e Y dos 4 centróides
[[0.297, 0.209], [0.700, 0.290], [0.402, 0.850], [0.923, 0.723]];
% SSE do eixo X dos 4 grupos
[1.112, 1.254, 2.654, 0.997];
% SSE do eixo Y dos 4 grupos
[0.820, 1.332, 2.101, 0.980];
% SSE dos 4 grupos
[3.002, 5.997, 6.100, 2.978];
%Quantidade de objetos por grupo
[218, 180, 348, 254];
```

Fig. 4. Formato do texto em arquivo ".vjm". O primeiro dado é o número k de grupos. Segundo são as coordenadas dos k centróides do primeiro agrupamento. Terceiro são os *SSE*'s da primeira dimensão dos k grupos. Quarto são os *SSE*'s da segunda dimensão dos k grupos. Quinto são os *SSE* dos k grupos. Sexto são as quantidades de objetos por grupo.

B. 2D Data Stream Viewer

A linguagem de programação *Java* foi escolhida para o desenvolvimento da ferramenta de visualização. Tal escolha baseia-se na portabilidade da linguagem e por ser a mesma utilizada na plataforma SAMOA. A ferramenta possui três objetivos principais: tornar intuitiva a análise dos resultados; ajudar na identificação de mudanças dos dados durante o fluxo e; visualizar as áreas com maior densidade de dados.

Para tornar intuitiva a análise dos resultados, foram utilizados os conceitos de visualização de dados, onde as cores e formatos das representações gráficas dos dados acionam o sistema perceptual humano, facilitando a percepção de diferenças entre grupos. As mudanças nos dados podem ser percebidas pela visualização da movimentação dos grupos ao longo do fluxo de dados. Isso foi implementado através de um automatizador de visualização de vários agrupamentos sequencialmente. A visualização de áreas com maior densidade de dados foi possível através de uma funcionalidade de plotagem global dos dados, ou seja, a plotagem de todos os pontos do fluxo em um só espaço.

As funcionalidades da ferramenta podem ser acessadas em uma única janela, que é ilustrada na Figura 5. A ferramenta trabalha com arquivos de texto com extensão ".vjm". Os arquivos ".vjm" são criados durante a execução do algoritmo *KMeans* no SAMOA, onde cada fluxo de dados gera um arquivo ".vjm" diferente. Cada arquivo é nomeado com a data e hora do início do fluxo de dados.

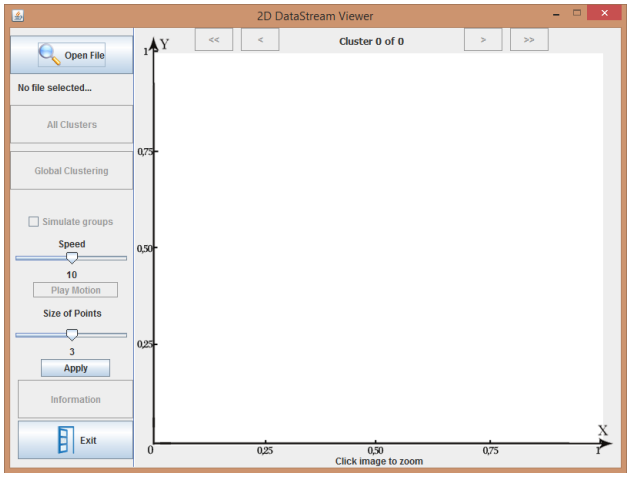

Fig. 5. Janela da ferramenta *2D Data Stream Viewer*.

Foi desenvolvida uma funcionalidade para abrir arquivos no formato ".vjm". Isso é feito pelo botão *"Open File"*. É

possível abrir arquivos de fluxos de dados já finalizados ou de fluxos de dados ativos. No caso de arquivos de fluxos de dados ativos, uma *Thread* é ativa para verificar modificações no arquivo aberto e então carregar os novos dados na memória de trabalho. Com o arquivo aberto, e os dados armazenados na memória de trabalho, os valores das coordenadas X e Y dos centróides são normalizados para o tamanho do gráfico, que é ilustrado por uma imagem 600x600 pixels. Com as coordenadas normalizadas, é feita a plotagem no gráfico com cores distintas para cada um dos k grupos. A Figura 6 exibe a plotagem dos 6 centróides finais do primeiro agrupamento de uma execução do algoritmo *Kmeans*. No topo da tela é possível visualizar o número do agrupamento ilustrado no gráfico e o total de agrupamentos disponíveis para visualização.

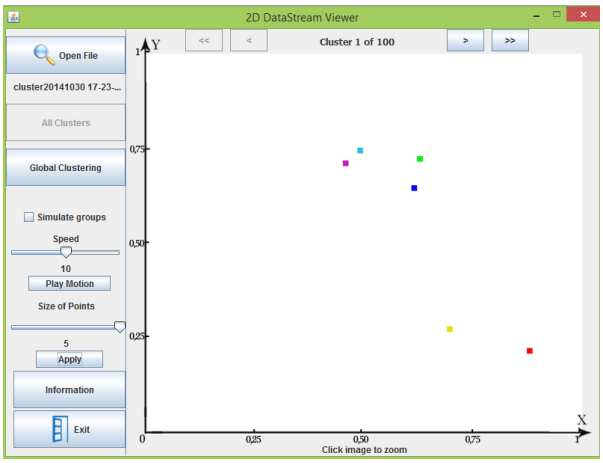

Fig. 6. Exemplo de plotagem dos 6 centróides do agrupamento 1 de 100 disponíveis.

As setas existentes na parte superior da janela foram desenvolvidas para navegar pelo histórico de agrupamentos. Como o algoritmo faz agrupamentos por lotes de dados, são feitos vários agrupamentos durante um único fluxo de dados. Com as setas de navegação, é possível avançar ou retroceder um agrupamento, ou acessar diretamente o primeiro ou último agrupamento disponível no histórico.

O botão *"Play Motion"* funciona como um automatizador na visualização dos agrupamentos sequencialmente. A velocidade em que os agrupamentos são alternados no gráfico pode ser controlada pela barra *"Speed"*. A velocidade mínima entre cada *frame* é de 2 segundos e a máxima é de 0,1 segundo. Ao visualizar os agrupamentos sequencialmente, é possível ter uma ideia de movimentação dos grupos. Isso ajuda na identificação de mudanças e percepção de novas tendências no fluxo de dados. A ferramenta de visualização torna explícita a percepção dessas mudanças, o que não era facilmente percebido quando a análise era feita apenas com os resultados numéricos dos agrupamentos.

A barra *"Size of Points"* altera a quantidade de *pixels* utilizados para representar um dado no gráfico. A barra possui valores normalizados de 1 a 5. Na imagem, cada dado pode ser representado por um ponto de 1 a 81 pixels. Quando a quantidade de dados a ser exibida for relativamente pequena, pontos maiores facilitam a visualização. Quando a quantidade de dados for relativamente grande, pontos menores representam melhor a real densidade dos dados. A Figura 7 ilustra a diferença entre a utilização de pontos pequenos e grandes na visualização dos dados.

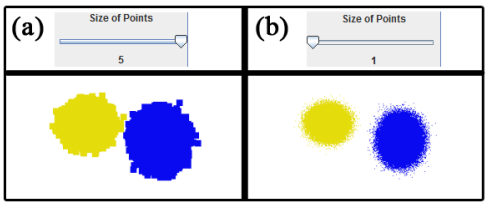

Fig. 7. Exemplo de uso da barra *Size of Points*.

O botão *"Global Clustering"* tem o objetivo de exibir os centróides existentes no histórico de agrupamentos. Tal funcionalidade proporciona a visualização das densidades finais do fluxo de dados. Com essa funcionalidade é possível perceber a tendência real dos dados em um longo período, elevando a confiabilidade da análise para o reconhecimento de padrões. A Figura 8 demonstra o agrupamento global de um histórico completo.

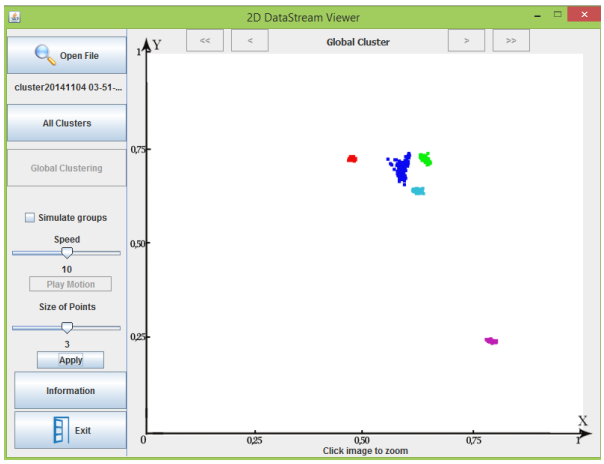

Fig. 8. Agrupamento global de um histórico completo.

O botão *"Information"* exibe as informações dos agrupamentos. Se ele for pressionado enquanto a ferramenta exibe um agrupamento global, uma janela de mensagem surge exibindo a quantidade de agrupamentos existentes no histórico; o número de k e; a quantidade total de dados agrupados no fluxo. Na Figura 9 são exibidas as informações globais de um histórico.

Quando o botão *"Information"* é pressionado enquanto a ferramenta exibe apenas um agrupamento, uma janela surge exibindo os dados de cada centróide; quantidade de dados e variância de cada grupo; e a variância total do agrupamento. Na Figura 10 são exibidas as informações de um único agrupamento do histórico.

A visualização dos centróides como representantes dos dados pode não expressar a real tendência e densidade dos mesmos. Isso ocorre quando os grupos possuem quantidades de dados ou variâncias discrepantes. Como todos

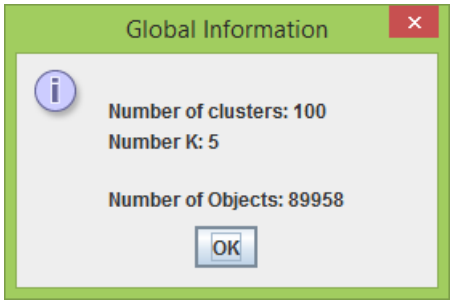

Fig. 9. Informações do agrupamento global de um histórico.

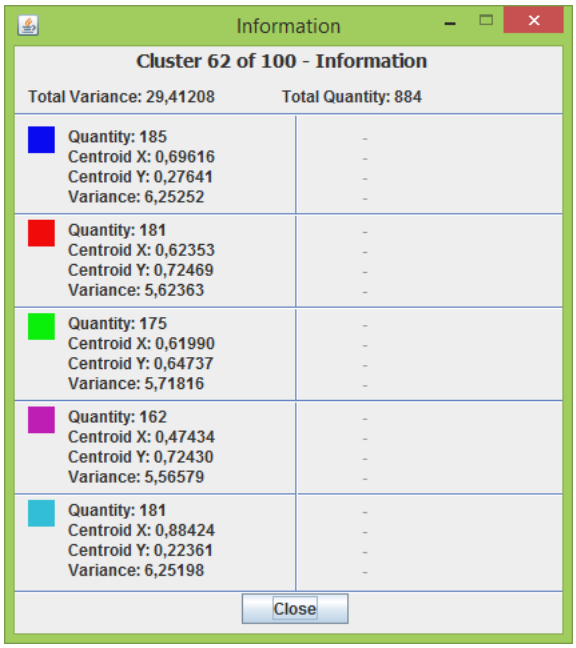

Fig. 10. Informações de um agrupamento do histórico com 5 grupos.

os centróides são plotados no gráfico com as mesmas dimensões, não é possível identificar essa discrepância através deles. Como solução a esse problema, foi criada uma caixa de seleção chamada *"Simulate groups"*. Essa função, quando ativada, utiliza o conceito estatístico de distribuição normal para simular os dados dos grupos. Os pontos são obtidos aleatoriamente, obedecendo os limites da distribuição normal, que é dada pela média (posição dos centróides) e a variância de cada grupo. A simulação pode ser aplicada em um agrupamento individual ou no agrupamento global. Na Figura 11(a) é exibida uma simulação dos 5 grupos de um agrupamento individual. A Figura 11(b) exibe uma simulação do agrupamento global de um histórico completo.

Como é possível ver na Figura 11, os grupos representados pelas cores vermelho e roxo se misturaram na simulação do agrupamento global. Devido a essa unificação e considerando a densidade dos dados, pode-se assumir que os dois grupos representam apenas um. Logo, a quantidade de k considerada cairia para apenas 4 grupos distintos. Como o valor de k é definido pelo usuário antes da coleta dos dados, pode haver diferenças na representação dos

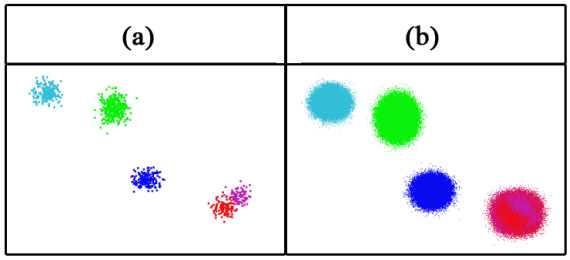

Fig. 11. Simulação de um agrupamento simples e global com k=5

grupos, devido à quantidade pré-definda.

Em um primeiro caso, quando o valor de k é baixo, pode haver grupos com variâncias elevadas, ocasionadas pela presença de sub-grupos ocultos, que estejam representados como um só. Em um segundo caso, quando a quantidade de k é alta, pode ocorrer a criação de grupos muito próximos, que poderiam ser ilustrados como apenas um. A ferramenta *2D Data Stream Viewer* não proporciona a percepção de sub-grupos para correção do primeiro caso, mas a simulação dos grupos e agrupamento global permitem a visualização de interseção entre os grupos, permitindo ao usuário considerar ou não a unificação dos mesmos.

Na Figura 12 é ilustrada a simulação do agrupamento global de um mesmo fluxo de dados para o valor de k=1 e para o valor de k=5. Na Figura 12(a) é possível perceber um único grupo de dados, sem lacunas nem espaçamentos entre eles. Na Figura 12(b) é possível perceber que os 5 grupos tornaram-se apenas 2, com uma grande lacuna entre eles. Embora seja a mesma base de dados, somente o segundo caso tornou possível a identificação de um segundo grupo. Pode-se afirmar que quanto maior o valor de k, maior será a precisão na identificação de diferentes grupos. Embora objetos similares sejam representados por cores diferentes, a visualização permite a percepção de similaridade pela proximidade dos mesmos.

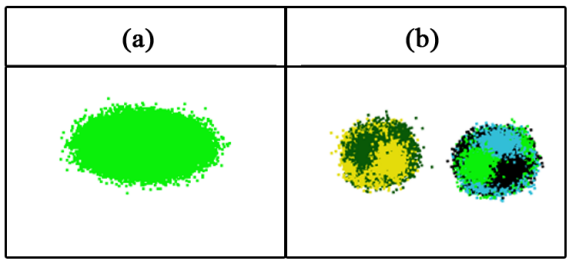

Fig. 12. Visualização de uma mesma base de dados para k=1 e k=5.

IV. Considerações Finais

UMA vez que as plataformas e algoritmos de análise de fluxos contínuos de dados existentes ainda estão em aprimoramento, a adaptação do algoritmo *KMeans* contribuiu para o estudo de novas formas de agrupamento de dados em fluxos contínuos. A adaptação do algoritmo possibilitou o armazenamento do histórico dos dados agrupados que são representados pelos centróides dos grupos.

Essa funcionalidade teve como objetivo auxiliar em futuras tomadas de decisão, baseadas no comportamento dos fluxos de dados analisados pelo algoritmo.

A ferramenta *2D DataStream viewer* facilitou a análise dos agrupamentos e identificação de mudanças durante o fluxo de dados gerado pelo SAMOA. Com ele, é possível visualizar tendências e simular os dados do fluxo, no qual, apenas os centróides são utilizados ao longo do desenvolvimento do fluxo. Como os algoritmos de agrupamento são desenvolvidos para lidar com um grande volume de dados, obter conhecimento dos mesmos sem a necessidade de tê-los armazenados é algo satisfatório. Como a ferramenta faz uso apenas dos dados numéricos finais dos agrupamentos, qualquer algoritmo de agrupamento pode ser adaptado para que os resultados possam ser visualizados pela ferramenta.

Como proposta para trabalhos futuros, pretende-se aplicar testes práticos de usabilidade, a fim de tornar a ferramenta de visualização mais intuitiva, além de perceber possíveis funcionalidades a serem implementadas como complemento e com isso, tornar a ferramenta mais útil. Pretende-se também criar uma API dentro da ferramenta de visualização para conexão com fluxos de dados reais, permitindo que a ferramenta seja capaz de coletar, agrupar e visualizar os resultados com a menor latência possível.

Referências

[1] D. Parikh e P. Tirkha, "Data mining & data stream mining – open source tools ", *Nature*, vol. 2, pp. 5234–5239, 2013, ISSN: 2319-8753.

[2] S. Guha, A. Meyerson, N. Mishra, R. Motwani e L. O'Callaghan, "Clustering data streams: Theory and practice.", em *IEEE Trans. Knowl. Data Eng.*, 2003, pp. 515–528.

[3] IBM e S. B. School, "Analytics: the real-world use of big data - how innovative enterprises extract value from uncertain data", em *Executive Report*, 2012, pp. 1–19.

[4] K. Faceli, J. Gama, A. C. P. L. d. Carvalho e A. C. Lorena, *Inteligência Artificial, Uma Abordagem de Aprendizado de Máquina*. LTC, 2011, vol. 1, pp. 1–378, ISBN: 9788521618805.

[5] P. N. TAN, M. STEINBACH e V. KUMAR, *Introdução ao Data Mining, Mineração de Dados*, 1ª ed., C. Moderna, ed. 2009, pp. 1–978, ISBN: 9788573937619.

[6] R. Linden, "Técnicas de agrupamento", *Revista de Sistemas de Informação da FSMA*, nº 4, pp. 18–36, 2009. endereço: http : / / www . fsma . edu . br / si / edicao4/FSMA_SI_2009_2_Tutorial.pdf.

[7] R. M. M. Vallim, J. A. A. Filho, R. F. de Mello, A. C. P. L. F. de Carvalho e J. Gama, "Unsupervised density-based behavior change detection in data streams", *Intell. Data Anal.*, vol. 18, nº 2, pp. 181–201, mar. de 2014, ISSN: 1088-467X.

[8] E. G. E. Carreon, "Técnicas de visualização de informação para a análise de dados de sensores e biossensores", diss. de mestrado, ICMC/USP, São Carlos/SP, 2013.

[9] G. De Francisci Morales, "Samoa: A platform for mining big data streams", em *Proceedings of the 22Nd International Conference on World Wide Web Companion*, sér. WWW '13 Companion, Rio de Janeiro, Brazil: International World Wide Web Conferences Steering Committee, 2013, pp. 777–778, ISBN: 978-1-4503-2038-2.

[10] M. Zaharia, M. Chowdhury, T. Das, A. Dave, J. Ma, M. McCauley, M. J. Franklin, S. Shenker e I. Stoica, "Resilient distributed datasets: A fault-tolerant abstraction for in-memory cluster computing", em *Proceedings of the 9th USENIX Conference on Networked Systems Design and Implementation*, sér. NSDI'12, San Jose, CA: USENIX Association, 2012, pp. 2–2.

[11] M. Ester, H.-P. Kriegel, J. Sander e X. Xu, "A density-based algorithm for discovering clusters in large spatial databases with noise", em *Proc. of 2nd International Conference on Knowledge Discovery and*, 1996, pp. 226–231.

[12] R. Sibson, "Slink: An optimally efficient algorithm for the single-link cluster method.", *Comput. J.*, vol. 16, nº 1, pp. 30–34, 1973. endereço: http://dblp.uni-trier.de/db/journals/cj/cj16.html#Sibson73.

[13] B. Babcock, S. Babu, M. Datar, R. Motwani e J. Widom, *Models and Issues in Data Stream Systems*, sér. PODS '02. Madison, Wisconsin: ACM, 2002, pp. 1–16, ISBN: 1-58113-507-6.

[14] C. C. Aggarwal, J. Han, J. Wang e P. S. Yu, "A framework for clustering evolving data streams", VLDB '03, pp. 81–92, 2003.

[15] J. Beringer e E. Hüllermeier, "Online clustering of parallel data streams.", Data Knowl. Eng., 2006, pp. 180–204.

[16] C. A. R. Pinheiro, *Inteligência Analítica*, 1ª ed., C. Moderna, ed. 2008, pp. 1–414, ISBN: 9788573937077.

[17] J. Leskovec, A. Rajaraman e J. D. Ullman, *Mining of Massive Datasets*, 1ª ed., Cambridge, ed. EUA, 2011, pp. 1–336, ISBN: 1107015359.

[18] D. T. Larose, *Discovering Knowledge in Data: an introduction to data mining*, 1ª ed., Wiley-Interscience, ed. 2005, pp. 1–240, ISBN: 9780471687542.

[19] M. C. Naldi, R. J. G. B. Campello, E. R. Hruschka e A. C. P. L. F. Carvalho, "Efficiency issues of evolutionary k-means", *Appl. Soft Comput.*, vol. 11, nº 2, pp. 1938–1952, mar. de 2011, ISSN: 1568-4946.

[20] A. Murdopo, A. Severien, G. d. F. Morales e A. Bifet, *Samoa - Developer's Guide*, 0.0.1, Y. L. Barcelona, ed. 2012, pp. 1–48.

[21] J. P. Reiter, "Satisfying disclosure restrictions with synthetic data sets", *JOURNAL OF OFFICIAL STATISTICS-STOCKHOLM-*, vol. 18, nº 4, pp. 531–544, 2002.

[22] G. Grinstein, M. Trutshl e U. Cvek, "High-dimensional visualizations", San Francisco/CA: 7th Data Mining Conference KDD Workshop, 2001, pp. 1–14.

[23] E. F. Iepsen, "Estudo e desenvolvimento de uma ferramenta de visualização de informações temporais para banco de dados estruturados no formato mestre/detalhe", diss. de mestrado, ESIN/UCPEL, Pelotas/RS, 2008.

[24] S. K. Card, J. D. Mackinlay e B. Shneiderman, "Readings in information visualization - using vision to think.", San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1999, pp. 1–17.

[25] A. L. S. Moraes, "Visualização de informações fuzzy para auxílio em diagnósticos médicos", diss. de mestrado, ESIN/UCPEL, Pelotas/RS, 2009.

[26] P. C. Wong e R. D. Bergeron, "30 years of multidimensional multivariate visualization.", em *Scientific Visualization*, 1994, pp. 3–33.

[27] I. H. Witten, E. Frank e M. A. Hall, *Data Mining: Practical Machine Learning Tools and Techniques*, 3rd. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2011, ISBN: 0123748569, 9780123748560.

[28] J. Abello e J. L. Korn, "Mgv: A system for visualizing massive multidigraphs.", *IEEE Trans. Vis. Comput. Graph.*, vol. 8, nº 1, pp. 21–38, 2002. endereço: http://dblp.uni-trier.de/db/journals/tvcg/tvcg8.html#AbelloK02.

[29] C. Stolte e P. Hanrahan, "Polaris: A system for query, analysis and visualization of multidimensional relational databases", em *Proceedings of the IEEE Symposium on Information Vizualization 2000*, sér. INFOVIS '00, Washington, DC, USA: IEEE Computer Society, 2000, pp. 5–, ISBN: 0-7695-0804-9.

[30] P. C. Wong, H. Foote, D. Adams, W. Cowley e J. Thomas, "Dynamic visualization of transient data streams", em *Proceedings of the Ninth Annual IEEE Conference on Information Visualization*, sér. INFOVIS'03, Seattle, Washington: IEEE Computer Society, 2003, pp. 97–104, ISBN: 0-7803-8154-8. endereço: http://dl.acm.org/citation.cfm?id=1947368.1947389.