TÉCNICO LISBOA

VisMillion (and Change)

Visualizando Big Data em Evolução Dinâmica

Tiago Miguel Borralho Pereira

Dissertação para obtenção do Grau de Mestre em

Engenharia Informática e de Computadores

Orientador(es): Prof. Daniel Jorge Viegas Gonçalves
Eng. Daniel Filipe Martins Tavares Mendes

Júri

Presidente: Prof. Francisco António Chaves Saraiva de Melo
Orientador: Prof. Daniel Jorge Viegas Gonçalves
Vogal: Profª. Ana Paula Pereira Afonso

Novembro 2019

Agradecimentos

Gostaria de agradecer à minha mãe por me facultar todos os meios e recursos e ainda ter mantido um apoio constante durante a realização do meu curso, sem ela não seria possível realizar este trabalho.

Agradeço também a todos os meus familiares, amigos e colegas pela coragem, motivação e apoio que me deram ao longo deste percurso.

Reconheço toda a confiança que me foi entregue pelo Prof. Daniel Gonçalves ao conceder-me este trabalho, assim como todo o suporte e auxílio fornecidos pelo Dr. Daniel Mendes, permitindo a sua conclusão.

Este trabalho foi parcialmente apoiado pela FCT através do VisBig PTDC/CCI-CIF/28939/2017 e UID/CEC/50021/2019.

Resumo

A quantidade de informação produzida pelas imensas atividades do mundo atual tem vindo a aumentar exponencialmente. Com esse aumento, também a exploração de diversas técnicas que permitem analisar e visualizar toda essa informação tem vindo a crescer de forma a poder acompanhar a sua evolução. No entanto, com o surgimento da necessidade de visualizar dados em tempo real, tornou-se imprescindível simplificar a sua análise, devido ao enorme volume de informação, para que o utilizador consiga interpretar os diversos padrões existentes nos dados, conforme estes evoluem rapidamente e consiga ainda reagir a possíveis variações das tendências dos fluxos de dados, mantendo sempre o contexto da informação representada na visualização.

Propomos o sistema "VisMillion and Change" que se foca no estudo de transições animadas suaves entre técnicas de visualização diferentes, que permitem analisar grandes quantidades de informação em tempo real. Possibilitando diferentes intervalos de tempo em cada uma, com o objetivo de se obter uma visualização única que contém os dados recém-obtidos e ainda um histórico (cada vez maior) da informação recebida. O sistema pretende facultar a visualização de *Big Data* em tempo real (*streaming*) e fornecer um conjunto de transições suaves que permitem facilitar a análise de grandes quantidades de dados conforme estes evoluem ao longo do tempo.

Palavras-chave: Visualização, Tempo Real, Padrões, *Big Data*, Streaming, Transições Animadas

Abstract

The amount of information produced by the immense activities of today's world has been increasing exponentially. With this increase, the exploration of various techniques that allow the analyzes and visualization of this vast amount of information has been growing in order to follow its evolution. However, with the emergence of real time Big Data, it has become imperative to simplify its analysis due to the huge amount of information so that the user can interpret the various patterns in the data as it evolves rapidly and can still react to possible variations of data flow trends, always maintaining the context of the information that is represented in the visualization.

We propose the "VisMillion and Change" system that focuses on the study of smooth animated transitions between different visualization techniques, allowing the analyzes of large amounts of information in real time. Providing different time intervals in each technique, in order to obtain a single view that contains the most recent data and also a history of the information that has been received. The system is intended to provide a visualization of real-time Big Data and a set of smooth transitions that make it easier to analyze large amounts of data as it evolves over time.

Keywords: Visualization, Real Time, Patterns, Big Data, Streaming, Animated Transitions

Índice

Lista de Tabelas

Lista de Figuras

Capítulo 1

Introdução

A automatização de diversas atividades exercidas na atualidade tem vindo a produzir cada vez mais informação. Este aumento deve-se grande parte à digitalização dos registos já existentes. No entanto, estes são gerados todos os dias e em qualquer lugar, desde registos pessoais sobre rotinas e viagens, à monitorização de largas e complexas redes; desde transações financeiras, a notícias provenientes dos média [17]. Também os sensores e as suas infraestruturas, assim como outros sistemas eletrónicos, estão constantemente a produzir dados, em intervalos de tempo cada vez menores [1]. Como consequência, tem sido potenciado um crescimento rápido e exponencial da dimensão, assim como do espaço em memória necessário para armazenar esta informação, a qual damos o nome *Big Data*.

Comparado com os dados tradicionais, este é caracterizado por 5Vs: enorme Volume, grande Velocidade, alta Variedade, pouca Veracidade e elevado Valor. Além do desafio de processar grandes volumes de informação, os principais obstáculos centram-se na diversificação do tipo de dados aliado ao requisito da velocidade, existindo sempre incerteza relativamente à fiabilidade dos dados [14]. Isto é, as ferramentas responsáveis pelo seu processamento, têm de lidar não só com dados estruturados tradicionais (bases de dados estruturadas) ou semiestruturados (documentos XML, JSON, etc.), como também com os não-estruturados (imagens, vídeos, redes sociais, etc.). Esta tarefa pode tornar-se morosa, dependendo do sistema e dos recursos computacionais no qual é realizado o processamento, de forma a uniformizar todos os formatos e estruturas dos dados existentes.

A exploração desta vasta quantidade de informação tornou-se uma tarefa complicada e muitas pesquisas têm sido feitas para encontrar uma forma que permita extrair toda a informação útil. É cada vez mais importante filtrar e marcar informação relevante dentro de conjuntos de dados conforme o domínio do problema ou a análise que se pretende realizar, para reduzir a quantidade de informação existente e facilitar a sua exploração. Permitindo assim, uma análise mais facilitada, através do reconhecimento de padrões, comportamentos e correlações interessantes entre os diversos dados [1].

1.1 Motivação

Dada a importância e a necessidade de facilitar a interpretação da informação, as visualizações tomam um papel essencial, permitindo uma melhor compreensão e exploração da informação existente. No entanto, a representação de grandes quantidades de dados pode condicionar a capacidade de um utilizador conseguir analisar todo o seu domínio. Assim como, a própria resolução dos ecrãs convencionais, que por si já é limitada, pode ser insuficiente para visualizar toda essa informação.

Algumas soluções para o problema anterior passam por aplicar técnicas de *Machine Learning*, métodos de redução de dados, medidas estatísticas para agregar a informação e ainda, fornecer características de escalabilidade tanto a níveis de detalhe como de interação nas visualizações. Mas, apesar de se possibilitar a observação de maiores quantidades de informação, estas soluções são usualmente aplicadas quando se tem acesso a todo o conjunto de dados de uma vez, para se conseguir processar e aplicar as técnicas desejadas.

No entanto, hoje em dia, muitos dos dados são gerados continuamente e com uma forte ligação ao seu contexto temporal. A estes os dados chamamos dados em *streaming*, que correspondem aos dados que são obtidos de uma forma continua e em tempo real, a partir de inúmeras fontes de informação. O surgimento deste tipo de dados, veio impedir a utilização de grande parte das técnicas que permitiam simplificar e reduzir o volume de dados.

As próprias ferramentas que antes processavam informação estática e permitiam a sua visualização, para agora acompanharem em tempo real a evolução dos dados conforme a sua relação com o tempo, tiveram de adotar tipos de processamento e de visualização diferentes para conciliar a representação dos novos dados recém-obtidos com os já existentes na representação. No entanto, grande parte dos sistemas atuais ainda não se adaptou a esta imposição dos dados em *streaming*, acabando por vezes, por dificultar a análise das visualizações ou mesmo sofrendo de congestão nas transferências entre as bases de dados e as visualizações. O que pode ter consequências como representações demasiado densas para serem analisadas e no pior dos casos, a existência de um "congelamento" do sistema por um certo período de tempo ou até mesmo um "*crash*" [4].

Para evitar estes problemas, devem ser considerados os domínios e os tipos de cada conjunto de dados, de forma a prevenir o sistema de eventuais alterações que possam surgir durante a representação da informação e, no caso de serem necessárias alterações ou modificações na visualização, estas devem ser suavizadas, para evitar choques visuais durante a análise da informação e a perda de contexto da visualização dos dados que o utilizador pretende explorar.

1.2 Objetivo

Com base na necessidade emergente provocada pelo crescimento contínuo e exponencial da escala da informação, assim como a imposição colocada pelos dados em *streaming*, tornou-se essencial facilitar a análise e comparação em tempo real dos dados, facultando a visualização das suas tendências e dos padrões existentes. O objetivo desta tese é:

Estudar formas de transitar grandes quantidades de dados obtidos em tempo real e que estão compreendidos entre múltiplas técnicas de visualização, associadas a diferentes medidas estatísticas e intervalos temporais, de forma a criar uma visualização contínua que evite a perda de contexto de análise por parte do utilizador.

Pretendeu-se facilitar a análise de grandes quantidades de informação em séries temporais, sendo ela obtida em tempo real. Para isso, facultou-se a representação da informação através de elementos que realizam uma "degradação graciosa" da representação visual dos dados, agregando a informação conforme esta se torna cada vez mais antiga na visualização. Ou seja, à medida que o tempo passa e novos dados vão sendo recebidos, os mais antigos dão espaço aos mais recentes, passando os primeiros a ser representados de uma forma mais agregada e abstrata, mas integrada, através da transição da informação para outro tipo de visualização. Desta forma, o utilizador passou a ter uma visão mais detalhada sobre os dados mais recentes, mantendo o contexto e os padrões anteriores sempre disponíveis.

Projetou-se ainda, que o utilizador fosse capaz de analisar um conjunto de módulos adjacentes que constituem a visualização e de comparar diferentes intervalos temporais, de forma a interpretar variações e tendências nos dados mais recentes, relacionando-as com possíveis padrões nos dados contidos em segmentos temporais mais antigos. Cada um desses módulos associado a um tipo de visualização que representa os dados com agregações e medidas estatísticas diferentes e alinhado horizontalmente com outro(s) módulos que representam outros intervalos temporais.

Foi por isso importante a existência de animações que permitissem seguir a evolução dos dados, para sempre que existisse alguma alteração no nível de agregação da informação ou os dados transitassem para um módulo diferente, o utilizador conseguisse entender qualquer tipo de transformação ocorrida para o novo tipo de visualização ao qual esse módulo está associado.

A interface foi desenvolvida com o objetivo de conseguir receber e representar enormes volumes de informação, obtidos por cada segundo de execução. Desta forma, o utilizador pode analisar os padrões existentes e seguir a sua evolução ao longo do tempo sem perder o contexto da informação que pretende analisar.

Para testar a interface foram realizados testes de usabilidade que permitiram compreender quais foram as técnicas de transição que mais se evidenciaram para representar a transformação entre duas técnicas de visualização diferentes, cada uma das quais com intervalos temporais muito distintos. Foram ainda realizados testes de eficiência para verificar qual era o débito máximo de informação que seria possível receber sem comprometer o normal funcionamento do sistema, mantendo as visualizações fluídas e as transições mais suaves possíveis.

1.3 Contribuições

O trabalho desenvolvido nesta dissertação levou às seguintes contribuições, no âmbito da visualização de grandes quantidades de informação em tempo real:

1. **Concetualização de um conjunto de técnicas de transição** que permitem representar enormes quantidades de dados em séries temporais e ao mesmo tempo, agregá-los e transformá-los em representações que podem ser quadrados, linhas ou barras que representam as medidas estatísticas para esse agregado de informação.

2. **Um protótipo que implementa várias técnicas de visualização e de transição** que permitem comprovar a concetualização mencionada no ponto anterior, com o objetivo de representar grandes volumes de informação em tempo real com vários módulos que realizam uma "degradação graciosa" da representação visual dados, tendo ainda a possibilidade de usufruir de grandes discrepâncias de intervalos temporais entre cada módulo.

3. **Testes de eficiência e avaliação do desempenho** das técnicas desenvolvidas no protótipo, apresentando as limitações que são impostas pelas tecnologias utilizadas e ainda pela complexidade das operações utilizadas para desenvolver estas técnicas perante a receção em tempo real de *Big Data*.

4. **Testes de usabilidade** que permitem validar a eficácia das técnicas de transição desenvolvidas para inferir padrões e tendências nos dados e respetiva avaliação. Propondo ainda, um conjunto de técnicas que melhor se adequam para a transição entre pares de técnicas de visualização específicas, através dos resultados obtidos e preferências dos participantes dos testes realizados.

1.4 Estrutura do Documento

A estrutura deste documento divide-se da seguinte forma: No capítulo 2 podemos ler o trabalho relacionado com a visualização de informação que possui diferentes soluções onde se aplicam diversos tipos de visualizações, referindo as suas principais vantagens e desvantagens, e finalmente, uma discussão onde as várias soluções são comparadas. De seguida no capítulo 3, é explicado o conceito VisMillion e são expostas as diversas técnicas de visualização, assim como as transições entre visualizações desenvolvidas. No capítulo 4 é apresentado o protótipo desenvolvido e discutida a arquitetura seguida pelo mesmo, assim como a sua implementação . No capítulo 5 são descritos os testes de desempenho realizados com o objetivo de encontrar as limitações impostas pelos volumes de informação e ainda, os testes de usabilidade, assim como as respetivas conclusões sobre a preferência dos utilizadores relativamente às técnicas de transição desenvolvidas. Por fim, no capítulo 6 é apresentada uma conclusão do trabalho desenvolvido e ainda direções para trabalho futuro.

Capítulo 2

Trabalho Relacionado

Para tornar possível o estudo das transições através das quais uma visualização consiga representar *Big Data* em *streaming*, é adequado compreender o estado da arte. Nos dias correntes, são várias as ferramentas e soluções disponíveis que respeitam os padrões de interação mais importantes, permitindo adaptar dinamicamente as visualizações. No entanto, as técnicas de visualização mais comuns nem sempre se podem aplicar a grandes quantidades de informação, sendo por isso importante adaptá-las conforme o contexto do problema.

Esta secção divide-se em duas partes. A primeira está relacionada com as ferramentas que pretendem responder à visualização estática de grandes quantidades de dados, enquanto a segunda parte se foca nos sistemas que representam *Big Data* em *streaming*.

2.1 Visualização de grandes volumes de dados

De forma a ser possível representar grandes quantidades de informação é necessário compreender o domínio dos dados e arranjar soluções que se adequem a cada técnica de visualização. A Tabela 2.1 sintetiza comparações de alguns tipos de visualização mais populares com a grandeza de volume, a variedade de tipos de dados e o seu dinamismo, em termos de visualizações estáticas ou dinâmicas.

Method Name	Large Data Volume	Data Variety	Data Dynamics
TreeMap	Yes	No	No
Cicle Packing	Yes	No	No
SunBurst	Yes	No	Yes
Parallel Coordinate	Yes	Yes	Yes
Stream Graph	Yes	Yes	Yes
Circular Network Diagram	Yes	Yes	No

Tabela 2.1: Atributos das técnicas de visualização (extraído de [1]).

Ainda que algumas técnicas de visualização necessitem de uma maior quantidade de modificações, estas podem ser ajustadas de forma a ser possível representar maiores volumes de dados, respeitando

a sua dinâmica e variedade. De seguida são apresentados alguns trabalhos relacionados, que pretendem ilustrar sistemas que expõem os vários tipos de soluções possíveis, adaptando algumas técnicas de visualização populares ou criando as suas próprias para visualizar a informação, permitindo desta forma representar *Big Data*.

2.1.1 DeepEye: An Automatic Big Data Visualization Framework

DeepEye [22] é um sistema de visualização que permite obter uma visualização adequada a um conjunto de dados, recorrendo para isso a transformações nos dados através de combinações de operações. Esta ferramenta automatiza essa tarefa através de processos que tentam responder aos seguintes pontos: Verificação da técnica de visualização de forma a obter uma boa combinação dos atributos do conjunto de dados e uma boa representação gráfica; Transformação e seleção eficiente dos dados; Produção de visualizações *On-Time*. Para verificar se uma visualização é adequada para um conjunto de atributos, os autores pediram aos seus utilizadores para classificarem várias visualizações possíveis para os dados existentes, obtendo-se assim o melhor tipo de visualização para cada conjunto de dados, podendo esta ser de diversos tipos: Linechart, Barchart, Piechart, etc.

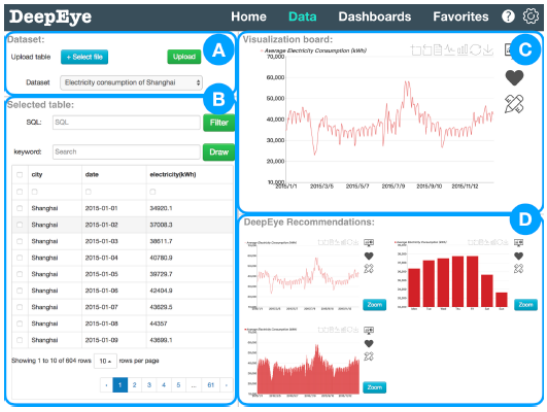


Figura 2.1: DeepEye front-end display [22].

Relativamente à interação, o sistema possibilita a inserção ou seleção de conjuntos de dados através da zona (A) na Figura 2.1. O utilizador pode depois filtrar os dados a serem visualizados na zona (B), obtendo as melhores representações dos dados, escolhidas pelo sistema, na zona (D) e a selecionada pelo utilizador na zona (C). A ferramenta fornece ainda técnicas como o zoom através da seleção de intervalos de dados nas visualizações.

Este sistema contribui bastante no trabalho do utilizador pois facilita a manipulação dos dados e seleciona a visualização mais adaptada para o conjunto de dados pretendido. O desempenho do sistema é, no entanto, agravado quando a dimensão dos dados é demasiado grande.

2.1.2 imMens: Real-time Visual Querying of Big Data

Visualizar grandes quantidades de dados pode levar à sobreposição dos mesmos, condicionando a capacidade de análise de um utilizador. Por outro lado, reduzir o conjunto de dados através de métodos

de *sampling* e *filtering*, pode remover dados importantes e *outliers*. Liu et al. [19] apresentam um conjunto de técnicas que permitem responder aos problemas referidos anteriormente, através de um sistema de interação em tempo real com escalabilidade visual. O imMens é um sistema browser-based de análise de visualizações que utiliza a tecnologia WebGL[1] para processamento de dados e renderização no GPU, o Leaflet[2] para mapas e o D3[3] para renderizar eixos e legendas.

Para a escalabilidade interativa, foram revistos os métodos de *panning*, *zooming* e *brushing* & *linking* em *binned plots*. Como o número de dimensões a visualizar aumenta, o tamanho dos dados de suporte pode crescer rapidamente. Para isso os autores desenvolveram métodos de querying visual em tempo real. O primeiro passa por pré-computar blocos de dados multivariados que são projeções correspondentes a views materializadas, decompondo o conjunto multidimensional de dados em conjuntos com 3 ou 4 projeções dimensionais. O segundo método trata-se de paralelizar o processamento e a renderização dos dados (WebGL).

Para a escalabilidade percetiva foram revistas técnicas de *Filtering* e *Sampling*, mas estes métodos requerem que as dimensões sejam conhecidas previamente causando um pré-processamento caro. Utilizou-se por isso, a técnica *Binned Aggregation* que agrega a informação em vários bins, possibilitando a visualização de densidades através da quantidade de dados que pertencem a cada um. Esta técnica possibilita a visualização de padrões globais (e.g., densidades) e características locais (e.g., *outliers*), o que possibilita ver informação mais rica nas visualizações enquanto permite múltiplos níveis de resolução dependendo do tamanho dos *bins*.

Focando-se nos binned plots, até 2 dimensões, uma das características fundamentais é a Cor. Esta é usada para codificar a densidade dos dados e indicar destaques para *brushing* & *linking*. Variáveis visuais adicionais (como texturas ou tamanho) ficam de parte pois podem interferir com a interpretação da visualização. Os autores escolheram binning retangular (o método mais simples de *binning*) para criar a grelha de pontos e desenvolver o histograma espacial, obtendo desta forma mais consistência e eficiência no processamento das queries. A interação com *Big Data* impõe uma dificuldade – respostas em tempo real. Interagir com uma visualização de grande escala, implica uma maior resolução de dados, para que seja possível realizar técnicas como *Panning* ou *Zooming*. Para *Brushing* & *Linking*, requerem-se agregações computadas filtradas por uma seleção inicial dos dados. Como solução, quando se aplica *binned aggregation* aos X campos desejados para as visualizações do sistema, são formados cubos de dados com X-dimensões, que o sistema decompõe em vários sub-cubos (igualando o número de divisões do mapa) que possuem no máximo 4 dimensões, ou seja, reduz-se a quantidade de dados para depois serem agregados para realizar *Brushing* & *Linking*. Já para interações *Panning* e *Zooming*, são utilizados mapas de dados pré-computados a diferentes níveis da resolução.

Como exemplo de implementação, dada a definição de uma visualização, o sistema imMens carrega os mapas de dados de um servidor e fornece um display de múltiplas visualizações interativas de binned plots e heatmaps geográficos para web browsers. Para testar o desempenho perante grandes quantidades de dados, verificou-se o tempo médio para realizar *Brushing* & *Linking* na visualização. Para

[1] https://www.khronos.org/webgl/
[2] http://leafletjs.com/
[3] https://d3js.org/

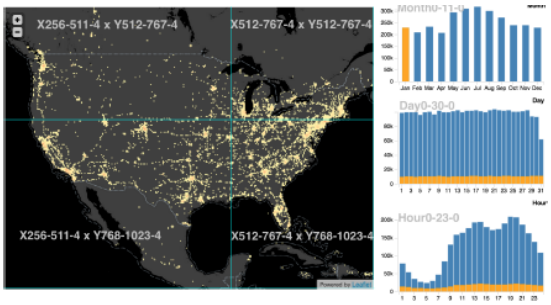


Figura 2.2: Visualização de múltiplas coordenadas: '*Brightkite user checkins in North America*' [19].

isso, utilizaram-se dois conjuntos de dados reais com cerca de 4 (Figura 2.2) e 118 milhões de itens, obtendo-se como tempo para realização da técnica, os valores 17.76 ms e 16.56 ms, respetivamente.

2.1.3 Topic-aware Network Visualisation to Explore Large Email Corpora

Repke et al. [23] propõem uma visualização interativa (em estudo) para redes sociais que posiciona indivíduos em representações bidimensionais, definindo as comunidades como ligações entre eles. Os autores focam-se em coleções de emails como fundamento para esta alternativa de visualização interativa e integrada que pretende encontrar padrões globais nos dados tendo como principais beneficiários os analistas de grandes quantidades de dados não-estruturados e heterogéneos. Os autores tentaram obter um sistema capaz de explorar largas coleções de documentos sem qualquer conhecimento prévio sobre o seu conteúdo. No sistema, utilizaram os nomes e emails dos recetores e remetentes, as redes de comunicação entre eles, representações vetoriais semânticas dos conteúdos e *timestamps* dos emails para criar uma representação gráfica capaz de responder à questão "Quando e quem comunica? Com quem e quando?".

Os emails trocados entre dois indivíduos são reduzidos a uma aresta que os liga na visualização, tornando a sua representação menos complexa e mais fácil de detetar os padrões existentes. A posição inicial de cada ponto é determinada pelo somatório normalizado dos vetores 2D de todos os emails enviados ou recebidos pelo individuo, agrupando os indivíduos em comunidades de comunicações frequentes e ligando-os por arestas que podem ou não ser visíveis conforme o detalhe pretendido, sendo este modificado ao fazer zoom. Os *clusters* de emails são codificados pela Cor que simboliza o número de tópicos existentes. Tanto a opacidade como a espessura das arestas são utilizadas para codificar a frequência das trocas de emails. Os *timestamps* são posteriormente utilizados para criar um heatmap que sobrepõe a visualização e permite ver a atividade num certo intervalo de tempo que pode ser regulado com um slider.

Dado que os autores apresentam uma visualização que se encontra em evolução, numa fase inicial, afirmaram que perante a representação de cerca de 2000 emails, existem indivíduos centrados no ecrã por não se relacionarem com nenhum tópico, no entanto já é possível distinguir indivíduos conforme a sua relação com os tópicos e apresentam uma imagem com um protótipo numa fase inicial - Figura 2.3. A Figura 2.4 representa uma maquete desenhada da visualização pretendida pelos autores.

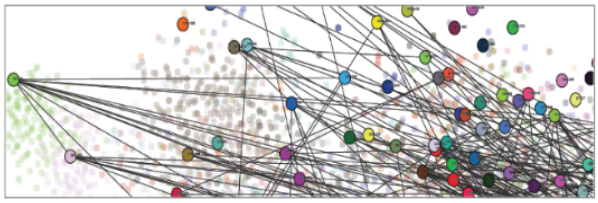


Figura 2.3: Protótipo Inicial Email Corpora [23].

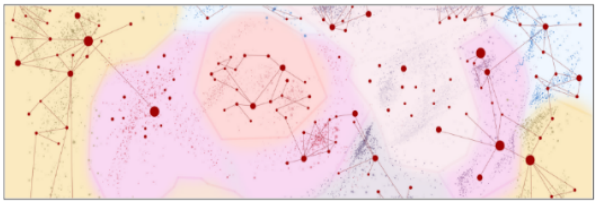


Figura 2.4: Maquete desenhada da visualização pretendida.

2.1.4 An Interactive 3D Visualization for the LOD Cloud

A LOD Cloud[4], contém milhares de conjuntos de dados acessíveis gratuitamente. As visualizações atuais são úteis para obter uma noção do tamanho e a relação existente entre eles, no entanto, Cerquitelli et al. [8] propõem uma visualização 3D que adota a metáfora da área urbana, com intuito de se obter uma representação visual e interativa de mais informação e de forma a ser possível descobrir mais relações entre os conjuntos de dados. Esta nova abordagem suporta várias formas de "edifícios" e são também vários os algoritmos que permitem posicioná-los no plano de visualização (2.5).

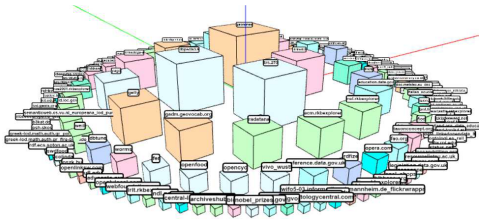


Figura 2.5: Perspetiva do modelo 3D interativo LOD [8].

A representação de cada "edifício" simboliza um conjunto de dados, cujo volume corresponde ao número de triplos do respetivo, ou seja, o número de conjuntos de declarações sujeito-predicado-objeto. Já o seu posicionamento, depende da abordagem desejada, diversificando-se em: *Mountainside layout, Orthogonal Spiral, Cyclic Spiral* e *Similarity-based layout*. Na Figura 2.6 e Figura 2.7 é possível visualizar cada uma destas opções. Se existirem ligações entre conjuntos de dados, é criado um segmento simbolizando uma "rua/ponte" que conecta os "edifícios", variando a largura conforme o grau de ligação.

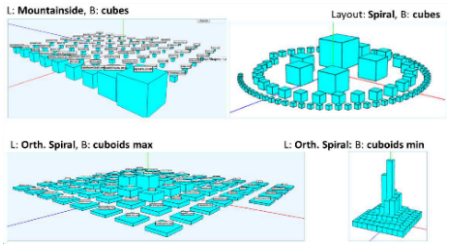


Figura 2.6: *Datasets* em LOD 2014 [8].

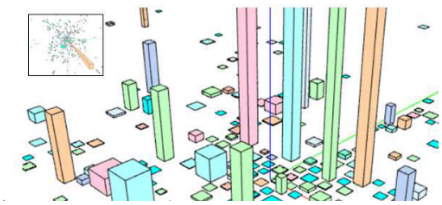


Figura 2.7: Similarity-based Layout [8].

Relativamente à interatividade desta alternativa, esta possibilita ao utilizador a realização de zoom em qualquer parte do modelo, alterar para a perspetiva desejada, assim como a forma dos "edifícios" e o seu posicionamento, para que se possa visualizar todas as ligações existentes.

[4]https://lod-cloud.net/

O sistema foi implementado em JavaScript com Three.js[5] e para o testar, foram aplicados 287 conjuntos de dados, de forma a verificar quantos seria possível visualizar. Com os resultados dos testes, os autores concluem que a visualização consegue representar até milhares de conjuntos de dados.

2.1.5 RBPCP: Visualization on Multi-set High-dimensional Data

Os sistemas de coordenadas paralelas são normalmente utilizados para visualizar diferentes atributos de um mesmo conjunto de dados. Quando se pretende visualizar grandes quantidades multidimensionais, são utilizados métodos de visualização que beneficiam desses sistemas com algumas alterações. No entanto estes métodos atuais não permitem detetar padrões nos dados. Os autores propõem uma alternativa para resolver esse problema – Rearranged Bundled Parallel Coordinates Plot (RBPCP) [28]. Este combina os métodos de justaposição e sobreposição para permitir a visualização de grandes quantidades de dados multidimensionais. Pondo em comparação os métodos referidos anteriormente como de justaposição e sobreposição, o primeiro só permite obter a distribuição de um conjunto de dados de cada vez. Já o segundo [Figura 2.8 (a)], permite obter a distribuição global de todos os dados mas em contrapartida sofre da oclusão das linhas e também torna difícil a comparação entre os mesmos. Por isso nesta alternativa, os autores aproveitam as vantagens de cada um destes métodos e acrescentam bundle points entre os eixos adjacentes ao sistema de coordenadas paralelas original, de forma a ser possível comparar as relações entre os vários sets. Posto isso, cada curva do sistema original irá ter começo no valor correspondente ao do conjunto de dados e terminará nesse bundle point. Resultando num *bundled parallel coordinates plot* como o da Figura 2.8 (b).

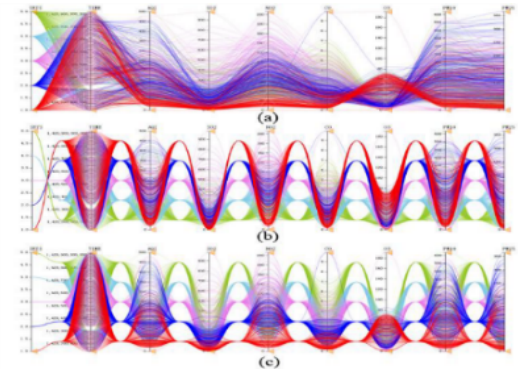


Figura 2.8: (a) Modelo de sobreposição; (b) Bundled Parallel Coordinates Plot; (c) Rearranged Bundled Parallel Coordinates Plot (RBPCP) [28].

Para tornar a visualização obtida com o Bundled Parallel Coordinates Plot menos oclusa, é necessário aplicar o algoritmo de reordenamento dos bundle points, aplicando-se para isso o algoritmo *Median-based Rearrangement* proposto pelos autores de onde se obtém um conjunto de sets ordenados pela prioridade de representação. O resultado da mesma está exemplificado na Figura 2.8 (c), este permite ilustrar a redução da oclusão. O sistema admite *brushing* tornando a visualização mais dinâmica. O utilizador pode alternar entre o modelo de sobreposição e o RBPCP, pode ainda realizar zoom em zonas de maior interesse e desse modo obter padrões nos dados existentes.

[5] http://threejs.org/

2.1.6 Bin-summarise-smooth: A framework for visualising large data

Wickham and Hadley propõem uma ferramenta - Bin-summarise-smooth [26], baseada num processo de quatro passos: processamento de bins (binning), summarize, suavização e visualização. Esta possibilita representar os dados permitindo a exploração interativa na visualização. Os primeiros dois passos do processo, quando eficientes, condensam grandes quantidades de informação em pequenos bins de dados num formato apropriado para a visualização, decompondo depois cada bin na sua quantidade absoluta, valor médio e desvio padrão. A suavização ajuda a resolver problemas provenientes dos passos anteriores, como o excesso de variabilidade dos dados em cada bin. E para esta, pode-se variar o tipo de suavização conforme a escolha do utilizador: *binned mean*, onde se calcula a média de cada *bin*; *running mean*, que é mais complexo e utiliza os pontos mais próximos para a computação; *kernel*, que além dos pontos vizinhos, pondera as distâncias ao ponto alvo e atribui-lhes pesos. O *kernel* divide-se em - *kernel means, kernel regression* ou *robust kernel regression*. Para computar estes últimos basta seguir as técnicas padrão da média, da regressão ou da regressão robusta.

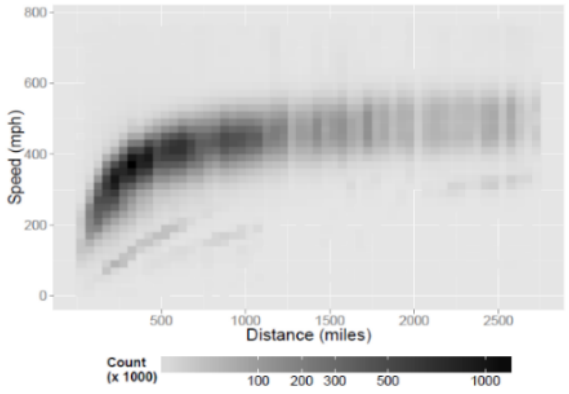


Figura 2.9: Distância e velocidade sumariadas com a quantidade absoluta [26].

Relativamente à visualização, depois de ter sido aplicado o processo referido anteriormente, é adotado um sistema que representa um conjunto de dados com n binned variables e m summary variables numa visualização com (n,m)-dimensões, permitindo a distinção entre as binned variables do conjunto de dados original e as novas variáveis criadas pelo processo. A Figura 2.9 representa uma visualização (2,1)-d (distance + speed, count). Os autores afirmam que normalmente se dá preferência a visualizações onde a quantidade de ambas as variáveis, é maior ou igual a um. Uma visualização (0,1) apenas iria representar uma distribuição das variáveis novas. Uma visualização (1,1) seria representada por uma linha, enquanto uma (2,1) seria algo semelhante a um heatmap ou um tile plot. Já, uma visualização (n,m) originaria small multiples, com ligações entre as múltiplas representações (Linked Brushing). Com o tamanho dos dados a aumentar, também a probabilidade de encontrar valores menos usuais aumenta. Para isso o sistema possui duas ferramentas: peeling e modulus transformation. A primeira remove progressivamente os bins menos preenchidos, permitindo melhorias nas visualizações. Já a segunda ferramenta, opta por reduzir o impacto visual dos outliers.

Os autores testaram esta framework aplicando conjuntos de dadoss com diferentes dimensões concluindo que o tempo de computação do sistema aumenta linearmente com a quantidade de dados mas

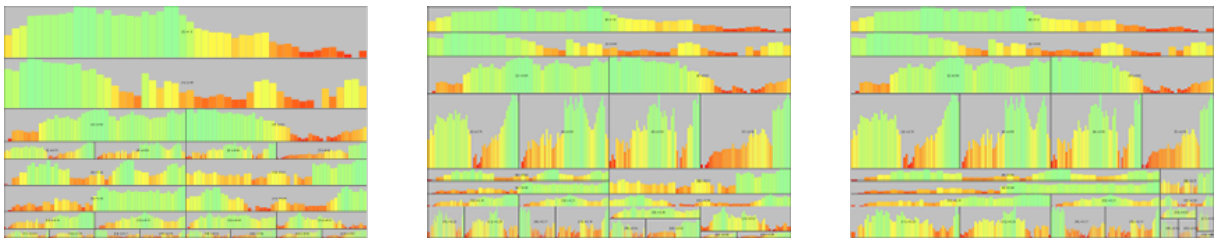
varia pouco com a quantidade de bins provenientes do processamento. Por análise aos indicadores de referência obtidos, verifica-se que o sistema obteve tempos computacionais $\approx$ 0.03s, 0.40s e 4.00s quando o tamanho dos dados variou entre 106, 107 e 108, respetivamente. No entanto, os autores referem que estes tempos podem ser melhorados otimizando o código do sistema.

2.1.7 Importance-Driven Visualization Layouts for Large Time Series Data

Para grandes quantidades de séries temporais devem utilizar-se técnicas de visualização que permitam analisar vastos conjuntos de dados de séries temporais e tendo em conta a importância dos dados e das suas interligações, estes devem ordenar-se hierarquicamente pelas propriedades mais importantes de um ecrã: a relação do tamanho e da posição, fornecendo sempre técnicas que permitam comparar a informação. Hao et al. [13], propõem uma ferramenta que permite visualizar largos conjuntos de dados e ao mesmo tempo responde às necessidades das técnicas de visualização referidas anteriormente. O objetivo principal dos autores é permitir que um analista consiga rapidamente detetar a importância relativa e as relações hierárquicas entre as séries temporais, suportando a sua comparação.

O sistema segue uma abordagem de preenchimento do espaço livre evitando a sobreposição para endereçar a importância dos dados assim como a sua escalabilidade. Um aspeto importante (seguido) foi a regularidade, que consiste em ter uma proporção do ecrã favorável à renderização de um dado número de intervalos entre cada partição da série temporal no ecrã. Esta proporção deve ser homogénea e condicionada pela importância de cada série temporal.

Foi desenvolvido um algoritmo – ID-Map, que mapeia recursivamente na visualização cada série temporal em partições, definindo-as através de *display masks*. Uma *display mask* é um esquema que permite particionar qualquer retângulo em vários sub retângulos, refletindo a importância e as relações pelo tamanho e posição de cada um deles. Para alocar um dado conjunto de séries temporais o *mask chooser*, analisa a distribuição de *i-measures* (que depende do contexto da visualização desejada – média, min, máx, etc.) e depois escolhe a melhor *mask*. Conforme o tipo de distribuição, existem duas masks diferentes: uneven mask, que é apropriada para quando a distribuição das i-measures é desenhada e *even mask*, que é apropriada para distribuições bastante uniformes. O sistema tem 3 opções para dividir os retângulos. Na Figura 2.10, a opção A, divide um retângulo em posições fixas, não se preocupando com as i-measures. As opções B e C, dividem os retângulos para somar as *i-measures*, no entanto oferecem menos regularidade com uma melhor proporção de tamanhos. A escolha das opções é baseada na preferência do utilizador.



(a) Opção A (b) Opção B (c) Opção C

Figura 2.10: Diferentes opções para visualização com sistema ID-Map.

Relativamente à interação, o sistema fornece a técnica *Drill-down*, que permite obter intervalos de dados mais detalhados. Permite que o utilizador defina os atributos para particionar hierarquicamente os dados, assim como as suas cores e os intervalos temporais. Esta ferramenta foi testada perante um conjunto de dados de vendas com 41 778 faturas, formando 35 séries temporais agregadas pelos valores de venda. Foi ainda experimentado representar conjuntos não estruturados de 5 a 200 séries temporais com 48 valores cada. Onde em ambos os casos o sistema obteve bons resultados, concluindo-se que o sistema fornece boa regularidade em termos do alinhamento dos retângulos, com poucas variações nas relações de tamanhos entre eles.

2.1.8 'Circle Segments': A Technique for Visually Exploring Large Multidimensional Data Sets

Ankerst et al. [3], descrevem uma técnica de visualização de grandes quantidades de dados multidimensionais – Circle Segments. Esta técnica, considerada como uma técnica valor-por-pixel, representa as dimensões dos dados em segmentos de um círculo, dividindo-o pelo número de dimensões existentes nos dados. Dentro de cada segmento, os dados são organizados desde o centro do círculo até ao seu exterior, de um lado para outro, ao longo da 'draw line'. Esta é ortogonal à linha que divide o segmento de cada dimensão, começando no centro do círculo a representar pixels desde uma das bordas do segmento até à outra. Sempre que a '*draw line*' alcança uma borda, será movida paralelamente para cima, até ao exterior do círculo, alternando a sua direção. Este processo é repetido até que todos os dados de uma dimensão estejam representados e depois nas restantes dimensões. Os dados mais antigos encontram-se no meio do círculo e os mais recentes, próximos do exterior. O esquema de cores mapeia os valores mais elevados com cores mais claras e os valores mais baixos com cores mais escuras, para que o utilizador tenha uma visualização mais intuitiva. O utilizador pode reatribuir segmentos do círculo às dimensões, possibilitando a sua troca da ordem e ajudando o utilizador a compará-las.

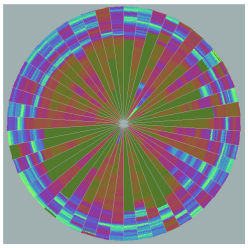


Figura 2.11: Visualização Circle Segments [3].

Para testar esta técnica, aplicaram uma base de dados de ações com cerca de 5328 registos. Comparando-a com gráficos de linhas, ao corresponderem as linhas a 7 preços de ações, apenas conseguem representar cerca de 1000 registos, devido ao tamanho do ecrã. Ao utilizar Circle Segments, obtém-se uma visualização mais completa e mais concisa, facilitando a consulta da evolução das ações e permitindo encontrar tendências entre as várias dimensões. A Figura 2.11 representa esta técnica perante 50 dimensões diferentes, aplicadas a 265000 dados. Os autores concluíram através de várias experiências que esta técnica é apropriada para explorar dados com muitas dimensões, no mínimo 3.

2.1.9 BinX: Dynamic Exploration of Time Series Datasets Across Aggregation Levels

BinX [5], é uma ferramenta que fornece visualizações dinâmicas de extensos conjuntos de dados de séries temporais, que permitem uma melhor compreensão das suas estruturas sem que seja necessário utilizar muitas visualizações nem aplicar transformações demasiado complexas. O conjunto de dados é manipulado através da agregação em binning e posteriormente representado de forma a analisar cada série temporal ao longo do tempo, suportando também a comparação entre elas. Desta forma, obtém-se uma visão global das agregações aos vários níveis de detalhe.

O sistema possui uma componente denominada - Dynamic Time Series Visualization (DTVC) que apresenta um ou dois conjuntos de dados de séries temporais representados como gráficos de linhas com um nível de agregação controlável, assim como a escolha e manipulação da escala do eixo do tempo. Uma DTVC contém no eixo horizontal, o intervalo temporal das séries temporais atualmente carregadas no sistema e rotuladas verticalmente com linhas que demarcam as bordas de cada *bin*. Essas linhas, estendem-se sempre para a parte superior do gráfico, obtendo-se um efeito 'trapézio', onde a largura dele e a sua inclinação interna e externa conectadas à base e topo da DTVC permitem mostrar o nível da agregação. A quantidade de bins corresponde à escala temporal e é representada por espaços horizontais entre as linhas temporais. A BinX fornece um mecanismo que classifica e atualiza as séries temporais dinamicamente quando os níveis de agregação alteram. Finalmente, está disponível uma scatter plot que apresenta as correlações diretas entre dois bins - Figura 2.12, permite observar a aplicação com as duas componentes DTVC, uma scatter plot e ainda uma *cluster view*.

Figura 2.12: Duas componentes DTVC com uma scatter plot e uma *cluster view* [5].

Relativamente à interação, o utilizador pode escolher a quantidade de bins a representar, assim como escolher o nível de detalhe que pretende visualizar. Todas as transições existentes são suaves para ajudar o utilizador a manter o rastro dos elementos da visualização. É ainda possível marcar períodos temporais através de uma caixa retangular colorida que permanece visível ao longo das transições. Estas caixas irão atualizar-se dinamicamente sempre que houver alterações nas agregações. Quando o ponteiro do rato sobrepõe um bin, a secção correspondente na escala inferior é destacada e a informação é mostrada numa caixa amarela. A aplicação fornece ainda linking navigation correspondendo os dois conjuntos de dados. A aplicação BinX, desenvolvida em Java, contém duas componentes DTVC que podem ser independentes e providenciar focos, níveis de agregação ou dados diferentes.

Para testar esta ferramenta, foi escolhido um conjunto de dados sobre as taxas de câmbio diárias de várias moedas ao longo de 15 anos, resultando em cerca de 5 000 dados. O problema inicial que se obteve foi a falta de espaço no ecrã para apresentar todos os dados, pois não havia qualquer extensão visual no modelo. Este foi resolvido assim que as visualizações com o efeito 'trapézio' foram adotadas, fornecendo navegação e sugestões para os dados apresentados fora do ecrã.

2.1.10 Dynamic Reduction of Query Result Sets for Interactive Visualization

Battle et al. desenvolveram o sistema ScalaR [4], um sistema que determina dinamicamente se o resultado de uma query é demasiado grande para ser renderizado para um ecrã. Este insere operações como *Aggregation*, *Sampling* e *Filtering* para reduzir a quantidade de dados a renderizar.

O sistema está dividido, na sua arquitetura, em três camadas. A primeira é a *Web Front-End*, responsável por receber os dados do utilizador, uma query e o tipo de visualização desejada (scatterplots, linecharts, histograms, etc.), para representar os dados resultantes utilizando D3. A segunda camada, é uma camada intermediária no servidor que recebe os dados vindos do *Front-End* e traduz em queries para a DBMS (*Database Managmente System*), obtendo da mesma um plano com informações de onde irá computar o tamanho expectável do resultado. Aplicando as técnicas de redução de dados que forem necessárias. A terceira camada, contendo a DBMS – SciDB [9], executa as queries recebidas.

Esta ferramenta facilita o utilizador no sentido em que elimina qualquer necessidade de escrever uma query que reduza a resolução dos dados, fornecendo-a de forma automática ao Back-End do sistema sempre que se justificar a redução dos mesmos. Desta forma, reduz-se o tempo de renderização da visualização e maximiza-se a utilização da resolução de um ecrã.

2.1.11 LiveRAC: Interactive Visual Exploration of System Management Time-Series Data

LiveRAC [21], um sistema de visualização de grandes quantidades de dados que permite a comparação visual lado a lado da informação a diferentes níveis de detalhe, possibilitando análises rápidas na monitorização de redes complexas ou datacenters. Utiliza *semantic zooming* nas representações e adapta-as conforme o espaço do ecrã. Foi o primeiro a integrar essa técnica com a interação *Stretch* and *Squish Navigation*, que possibilita a expansão de partes da visualização comprimindo as restantes.

A interface do LiveRAC, tem como princípios a utilização de representações familiares ao utilizador como Linecharts e Barcharts, aplicando pequenas visualizações lado a lado que partilham entre si o período temporal. Ao sobrepor o cursor ao longo de um gráfico, uma linha vertical é apresentada sobre as visualizações, criando uma ligação (Linking) que possibilita a sua comparação. Segundo o estudo realizado, concluiu-se que a organização espacial é mais precisa do que as codificações como a cor, tamanho e orientação e, por isso, na matriz de visualização do sistema, uma linha representa um dispositivo e cada coluna apresenta um grupo de parâmetros do mesmo. Finalmente, todas as transições existentes são animadas, fazendo com que o acesso a níveis de detalhe mais elevados num segmento de dados de uma visualização continue a permitir que toda a restante informação continue

visível para o utilizador. Assim, o utilizador não perderá o contexto da informação já existente. A Figura 2.13 ilustra a interação com a interface do sistema que permite ligar todas as visualizações em função da sua comparação. Em termos de implementação, o LiveRAC está ligado a um servidor SWIFT [Koutsofios, Eleftherios E., 1999], que possui a base de dados. E utiliza o PRISAD [24] para renderizar e permitir a navegação entre os conjuntos de dados de forma eficiente.

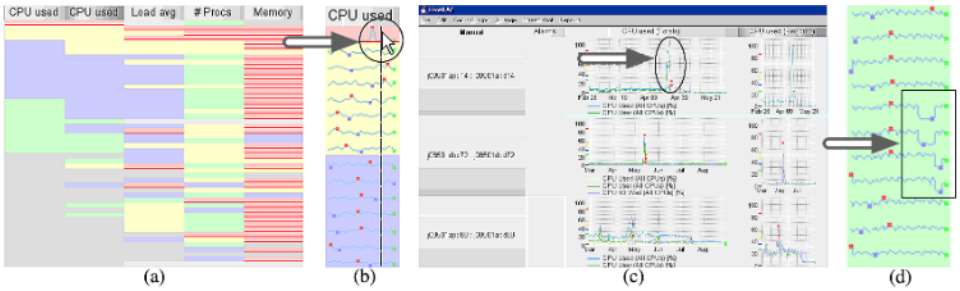


Figura 2.13: Visão global do sistema LiveRAC perante a sua interação [21].

Após um estudo realizado pelos autores com vários testes a utilizadores, esta ferramenta obteve bons resultados quando experimentada com uma grande quantidade de dispositivos e parâmetros, sendo uma boa opção para grandes quantidades de informação.

2.2 Representação de Big Data em Streaming

Enquanto os dados estáticos podem ser representados conforme descrito na Secção 2.1, a solução deste trabalho passa por aplicar uma nova técnica de visualização, ou adaptar as existentes de forma a visualizar grandes quantidades de dados em *streaming*, contextualizando-os temporalmente. Conforme Krstajić et al. [17], muitos dos tipos de visualização existentes poderão ser aplicados para dados em *streaming*, desde que sejam feitas alterações na forma como estes são representados, quer seja por reajustes nas suas métricas ou por reorganização dos atributos na visualização. Nesta segunda parte do Capítulo 2, serão apresentadas algumas soluções para estes problemas, com foco na representação de *Big Data* em *streaming*.

2.2.1 VALID: A Web Framework for Visual Analytics of Large Streaming Data

VALID [18], uma ferramenta baseada no navegador para visualizações dinâmicas de largas quantidades de dados. A escalabilidade visual é inerente à arquitetura *streaming* suportada pela plataforma, que é baseada no modelo servidor-cliente.

Para criar as visualizações dinâmicas é utilizado Javascript, WebGL e D3. Dada a grandeza dos dados, os autores introduzem técnicas de visualização como *Binned Points* e *Binned* & *Bundling Lines*. A primeira técnica (Figura 2.14), agrega pontos que se encontram na mesma posição numa splatter-plot, utilizando a cor como atributo que demonstra a densidade dos pontos. A representação dos novos dados é feita adaptando-os à visualização e correspondendo a sua cor à escala existente. A segunda técnica (Figura 2.15), de forma semelhante à anterior, permite uma visualização estatística que agrupa

Figura 2.14: Visualização Binned Points [18].

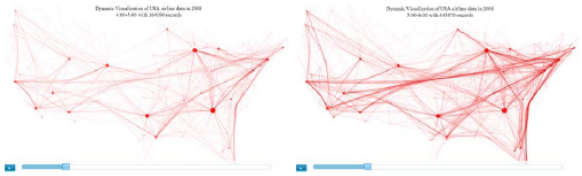


Figura 2.15: Visualização Binned & Bundling Lines [18].

as linhas que se encontram na mesma posição, estendendo o método edge bundling e evitando a sobreposição de linhas. Esta visualização ao receber novos dados procura indexá-los através dos valores bundling pré-calculados poupando assim a repetição do moroso processo inicial de edge bundling.

Para testar o sistema, aplicou-se um conjunto de dados com cerca de 120 milhões de registos reais (representando 20 anos de voos nos EUA), com o qual a ferramenta obteve uma média de 30 quadros por segundo durante a sua renderização, possibilitando uma visualização fluida dos dados existentes.

2.2.2 An Online Visualization System for Streaming Log Data of Computing Clusters

O estudo das visualizações sobre estas monitorizações de *Clusters* contribui para uma melhor e mais rápida decisão em situações críticas. No entanto, estas são demasiado complexas por representarem dados hierárquicos e heterogéneos, de múltiplas frontes e em *streaming*. O objetivo de Xia et al. [27] era apresentar um sistema com duas fases de processamento de dados que permita visualizar estes mesmos, ultrapassando essas dificuldades.

Para tornar possível a representação de diversos tipos de dados (numéricos, caracteres numéricos, sequências de caracteres, etc.), como este tipo de monitorização exige, é necessário um pré-processamento da informação para uniformizar cada um dos tipos de dados recebidos.

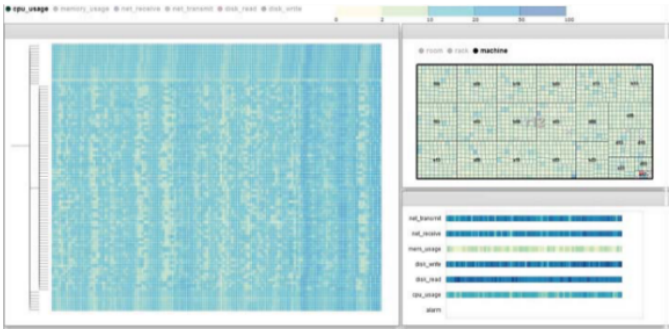


Figura 2.16: Painel de visualização Cluster Insight [27].

O sistema de visualização corresponde a um sistema online de múltiplas visualizações e inclui uma de *Cluster Metric* e outra de *Cluster Insight*, implementados com base na biblioteca Flare[6] que possui

[6]ttp://flare.prefuse.org/

gráficos básicos com transições animadas e ainda utilizando o servidor Red5 Media[7]. A visualização *Cluster Metric* permite uma visão global do conjunto de *clusters* existentes, sendo composta por três painéis. O primeiro é um gráfico de coordenadas paralelas que permite comparar múltiplas métricas de desempenho de *clusters*. O segundo é um painel que permite agrupar ou isolar diferentes *clusters* para criar uma visualização em gráficos de barras. Por último, *Expandable time sequence chart*, com 2 painéis de controlo: um radar chart responsável pelas métricas em tempo-real e um *time sequence chart* que apresenta um registo dos comportamentos dos *clusters* mostrados no *radar chart*. A visualização *Cluster Insight* (Figura 2.16) é composta por um *time sequence graph* que apresenta a hierarquia dos *clusters* e os seus desempenhos. É composta por um treemap, que indica uma visão global do desempenho dos *clusters*, e por um node metric card que indica seis métricas mais importantes para o utilizador.

Este sistema suporta as técnicas básicas e estatísticas dos painéis de controlo, suporta a visualização de dados com diferentes formatos e ainda métodos de visualização de dados hierárquicos, fornecendo sempre interação com o sistema, permitindo alterar as métricas que o utilizador pretende visualizar ou selecionar elementos dos gráficos para que possam ser visualizados com mais detalhe. Todas as transições existentes são fornecidas com suavidade.

Conforme os resultados obtidos na monitorização de 30 *clusters* ao mesmo tempo, entre os quais existem *clusters* com mais de 1500 máquinas, os autores afirmam que é possível localizar problemas existentes nos *clusters* através de visualizações fácil de perceber e simples de utilizar.

2.2.3 Density Displays for Data Stream Monitoring

Hao et al. [12] propõem duas técnicas de visualização em tempo-real para representações densas de informação. A primeira – *circular overlay displays*, passa por visualizar largos volumes de dados sem ter de movimentar a representação quando esta já está completa, evitando assim que os analistas tenham de ajustar a sua imagem mental sempre que novos dados forem acrescentados. A segunda técnica – *variable resolution density* displays, permite que a visualização completa seja obtida mantendo sempre a sua ordem inicial.

Foram aplicadas múltiplas células como séries temporais para a visualização das largas streams de dados, onde a cor de cada célula representa o valor de um atributo. No caso *variable resolution density displays*, o tamanho de cada célula diminui conforme existem mais dados lidos para permitir que o utilizador tenha uma visão global numa só visualização. Isto é, dependendo da quantidade de dados a serem apresentados e o espaço disponível na visualização, esta ajusta a resolução de cada célula. Já para que os padrões descobertos não se percam com a adição de mais dados à visualização, os autores utilizaram *circular overlay displays* para substituir as mudanças convencionais na visualização quando esta já se encontra completa. Ou seja, quando o ecrã está cheio, os dois últimos intervalos de dados são sobrepostos pelo novo intervalo (Figura 2.17). Utilizando esta técnica é possível representar grandes quantidades de dados num formato de séries temporais usando apenas um ecrã e sem ser necessário mover dados. Assim, o utilizador poderá lembrar-se dos padrões já descobertos e o desempenho do

[7] http://red5.org/

sistema é melhorado. É possível interagir com o sistema podendo repetir a chegada dos dados em tempo-real, movimentando um slider.

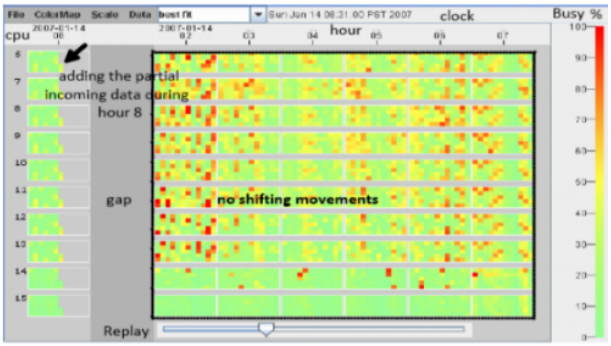


Figura 2.17: Circular Overlay Display, sobrepõe últimos intervalos (00-01 horas) com o novo intervalo (08 horas) [12].

As técnicas apresentadas pelos autores eliminam a limitação das séries temporais na representação de grandes quantidades de dados em *streaming*, possibilitando que mais dados e intervalos de tempo sejam vistos numa só visualização. No entanto, alguns testes com utilizadores, revelaram dificuldade na comparação da coluna do novo intervalo de dados com as restantes.

2.2.4 I^2: Interactive Real-Time Visualization for Streaming Data

I^2 [25], é um sistema de visualização interativa que coordena o funcionamento de aplicações em *Cluster* e apresenta visualizações para o mesmo. O sistema integra visualizações eficientes em tempo-real e disponibiliza um ambiente de desenvolvimento interativo que conecta os programas de análise de dados distribuídos com os resultados e as suas visualizações. Tal como o nome indica I2, são dois os tipos de interação que este sistema fornece: através de alterações no código e através da interação com a interface do sistema. Assegurando-se que apenas dados que estão representados na visualização são processados e transferidos para o *Front-End*, diminuindo a quantidade de dados a representar.

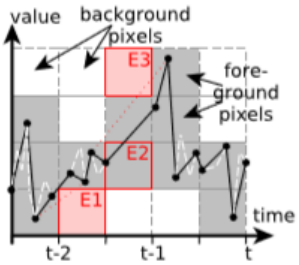


Figura 2.18: Algoritmo M4 [15].

Os autores referem a técnica de agregação M4 [15], que permite reduzir a quantidade de dados, transferindo 4 valores agregados por coluna de pixéis, isto é, encontra o valor máximo e mínimo, assim como, o primeiro e último *timestamp* dos valores, e conecta-os com uma linha, como é possível observar na Figura 2.18. Todos os pixéis que cruzarem essa linha serão coloridos. No final, não existem perdas no gráfico comparativamente ao dos dados originais. Dado que a técnica apenas é considerada para

dados finitos, é ainda necessário, criar um canal de processamento de *streams* para que seja possível suportar *streaming data*. Este consiste em 4 tarefas que se realizam paralelamente. Sendo a primeira o *watermarking*, que define *timestamps* para evitar o processamento de dados que cheguem fora de ordem. Em seguida é realizado o *windowing*, dividindo o fluxo de dados em pedaços de informação para posteriormente se correr o algoritmo M4 e representar-se os resultados obtidos.

O servidor do sistema onde os pipelines para análise são enviados é o Apache Flink[8] *cluster*, que permite processar rapidamente grandes quantidades de dados em *streaming*. O *Front-End* é baseado na ferramenta Apache Zeppelin[9], tendo este sido modificado para suportar a redução automática de dados dependendo dos parâmetros da visualização. Os dois sistemas comunicam entre si de maneira a disponibilizar rapidamente os dados necessários para a sua representação.

Esta ferramenta disponibiliza interações ao utilizador que em tempo-real poderá realizar as alterações desejadas na visualização, sem que este sistema necessite reiniciar. A quantidade de dados a representar é reduzida sem perda de qualidade o que significa que quando se aumenta a quantidade de streams, o sistema irá resolver a quantidade de trabalho independentemente da visualização, de outra forma o *Front-End* iria encravar com a sobrecarga de informação.

2.2.5 StreamSqueeze: A Dynamic Stream Visualization for Monitoring of Event Data

O StreamSqueeze [20] adota uma estratégia que usa posições estáticas dos dados para uma melhor leitura dos mesmos e opta por transições suaves para facilitar a rastreabilidade sempre que a informação for atualizada ou adicionada à visualização. Desta forma o sistema exibe os novos dados numa maior resolução, o que permite uma análise mais detalhada. Favorece a escalabilidade e ainda tem animações suaves e contínuas na visualização resultando num melhor acompanhamento de *streaming* de dados em tempo-real.

A visualização referida utiliza a técnica de preenchimento do ecrã, na qual a informação mais recente ocupa mais espaço. Todas as colunas seguintes contêm o dobro dos itens da anterior e são reduzidas a metade do tamanho. Sempre que um novo item chegar, o registo mais antigo é removido da lista e através de uma transição suave é colocado na coluna seguinte numa posição adjacente à anterior (e assim sucessivamente), representando desta forma o esquema temporal dos eventos. Já a sua organização vertical é computada de acordo com o seu identificador e tamanho. É utilizado o código de cores para identificar as categorias de cada evento.

Relativamente às transições, na primeira metade do tempo em que cada item se encontra numa coluna, a sua posição mantém-se constante no ecrã para uma melhor leitura e interação com o mesmo. A animação, é depois responsável por remover os itens das listas assim que 50% da lista de um, contiver mais dados recentes do que esse mesmo. O utilizador pode interagir com o sistema selecionando eventos que pretenda visualizar (Figura 2.19), alterando a cor do evento para vermelho. Tem a desvantagem de ser complicado selecionar os itens das últimas colunas por serem pequenos. Pode ainda

[8] https://flink.apache.org
[9] https://zeppelin.apache.org

pausar ou resumir a visualização dos dados em *streaming*, sendo que para a última, o sistema opta por aumentar a velocidade dos eventos pausados até que se atinja o período temporal atual (tempo real).

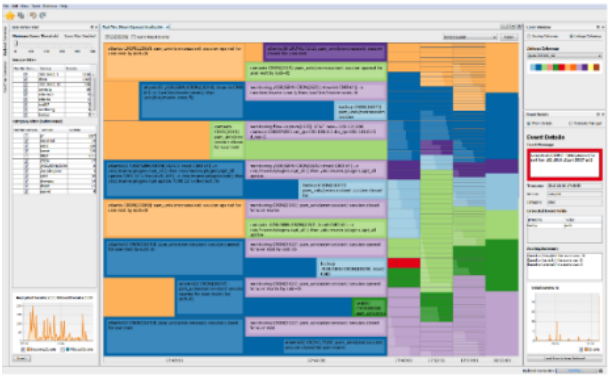


Figura 2.19: Interação com a visualização utilizando a ferramenta StreamSqueeze [20].

O desempenho desta ferramenta depende de três fatores importantes: a frequência da chegada de eventos, o número de categorias monitorizadas e o nível de detalhe de cada evento. Se a frequência da chegada de eventos for muito alta, o nível de detalhe que irá ser percebido pelo utilizador será muito baixo. Tendo sido testada com um sistema de registo de eventos, que ao receber 100 000 eventos por hora, tornou a análise mais complicada devido ao número limitado de itens que podem ser representados em simultâneo, um máximo de 2047 eventos.

2.2.6 Real-time Visualization of Streaming Text with Force-Based Dynamic System

Alsakran et al. [2] apresentam um novo sistema de visualização dinâmica que suporta exploração interativa com escalabilidade, denominado STREAMIT. Este permite analisar streams de documentos de texto, agrupando-os pelas suas similaridades conforme novos documentos são obtidos, estando por isso em constante atualização através de transições graduais que ajudam a não quebrar a imagem mental criada pelo utilizador.

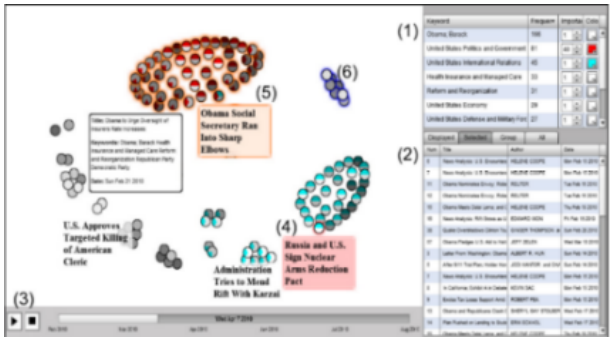


Figura 2.20: Interface do sistema STREAMIT [2].

A representação do sistema é feita em espiral e permite visualizar e analisar os documentos de forma dinâmica. A semântica de cada grupo é examinada por *keywords* e o título de cada documento é apresentado como legenda. As *keywords* são palavras importantes que aparecem com alguma

frequência nos documentos, que são automaticamente agrupados através de planos geométricos, para que os seus grupos possam ser representados. Servem para alterar o foco da visualização, provocando alterações nos grupos existentes. Relativamente à organização desta aplicação (Figura 2.20), a STREAMIT tem uma área principal onde é apresentado o movimento das partículas (documentos) numa visualização 2D. Cada partícula é representada por um gráfico circular, cuja divisão do mesmo simboliza as keywords. As similaridades entre elas refletem-se na proximidade das posições, formando grupos, que variam com a evolução do tempo. É utilizada uma escala cinza para indicar a "idade" de um documento. Existe um painel que permite controlar ou pausar o fluxo, uma tabela que contém as *keywords* e permite alterar a sua importância e cor, e outra que permite consultar todos os documentos.

Toda a visualização permite interatividade, sendo possível pesquisar, rastrear e examinar documentos. O utilizador pode controlar a importância de cada *keyword*. O sistema revela a evolução do fluxo através de um slider que permite animar o sistema, possibilitando aos utilizadores descobrir padrões ao longo do tempo, sendo para esse efeito também possível, selecionar grupos e keywords, sombreando todos os documentos que as contenham.

A capacidade desta ferramenta revelar grupos pode não ser compatível quando existe uma elevada quantidade de *keywords*. Usando para isso o modelo LDA [7], que reduz a quantidade de *keywords* por documento, permitindo extrair tópicos que estejam relacionados com a stream a ser visualizada. Assim, durante a visualização em tempo-real, a aplicação examina dinamicamente se um documento recém-chegado contém esse tópico analisando as suas *keywords*.

O sistema foi testado com conjuntos de dados reais até 10205 documentos, contendo 2036 *keywords* e apresentou uma resposta perante a inserção dos documentos em menos de 1 segundo, sendo por isso suficientemente rápido. Apesar da visualização ser pouco intuitiva, os autores concluíram que o sistema suporta até vários milhares de documentos para análises em *streaming*, oferecendo simulações rápidas com respostas imediatas e controlos convenientes.

2.2.7 Real-Time Visualization of Network Behaviors for Situational Awareness

Os autores apresentam um sistema que combina múltiplas técnicas de visualização, resultando na plataforma MeDICi [6]. Esta, fornece uma análise em tempo-real da atividade na rede para ajudar os analistas a terem respostas imediatas aos problemas existentes. E divide-se em duas ferramentas de visualização: *Correlation Layers for Information Query and Exploration* (CLIQUE) e *Traffic Circle*, fornecendo visualizações complementares que suportam a exploração temporal de dados que podem representar possíveis problemas na rede.

A ferramenta mencionada anteriormente – CLIQUE (Figura 2.21), gera modelos estatísticos de padrões esperados no tráfego da rede, permitindo a comparação dos mesmos com a atividade em tempo-real. Esta permite análises dos dados em *streaming* através de visualizações semelhantes a um heatmap, que representam o comportamento da rede. Dada a quantidade de atividades existentes em cada rede, é necessário reduzi-la e para isso, o sistema utiliza uma técnica, SAX – *Symbolic Aggregate approximation*, que reduz as dimensões da stream e gera uma representação que depois é convertida

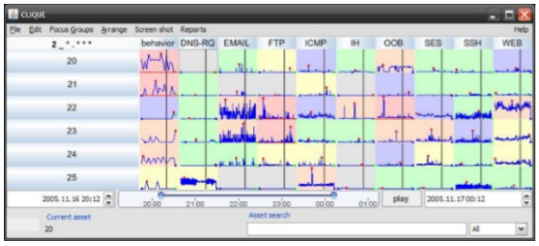


Figura 2.21: Interface CLIQUE [6].

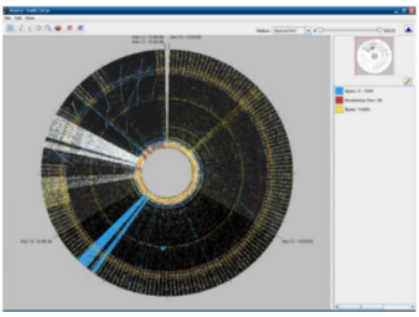


Figura 2.22: Interface Traffic Circle [6].

em conjuntos de *glyphs* (representam as tendências dos dados, ou seja, se estes são constantes, crescem, decrescem, etc). O objetivo da CLIQUE é ajudar os utilizadores a detetar eventos potencialmente malignos nas vastas quantidades de dados. Os modelos são criados em tempo-real e em resposta à interação do utilizador que permite realizar técnicas de *zoom* nas células da visualização para obterem informação mais detalhada, e esta será representada numa visualização do tipo *SparkLine*. É utilizada a codificação da cor para permitir uma rápida identificação do estado corrente de uma categoria.

Já a Traffic Circle (Figura 2.22), que complementa a ferramenta anterior, permite representar o máximo de dados possível numa só interface interativa, suportando uma exploração mais aprofundada das características descobertas com a CLIQUE, permitindo aos analistas visualizarem largos segmentos do tráfego da rede. Esta ferramenta, utiliza a metáfora roda temporal para representar os dados através de arcos. Os dados mais recentes estão localizados no topo do círculo e o tempo é percorrido como um relógio, sendo que os dados mais antigos vão sendo removidos da visualização, e estes encontram-se no topo, mas mais à esquerda. Foi verificado operacionalmente que esta visualização permite representar acima de 125 milhões de fluxos de atividade por análise. Relativamente à interação, o utilizador pode manipular o zoom e controlar a stream rodando o círculo, o que permite ajustar o período temporal. A cor é utilizada para visualizar determinadas atividades, podendo torná-las transparentes de forma a reduzir o ruído e a oclusão. É ainda possível alterar o tipo desta visualização, de circular para retilínea, reduzindo o tempo de renderização da visualização.

A habilidade das duas ferramentas para análise dos dados em tempo-real, foi testada com um conjunto de dados real cujas taxas de fluxo variaram entre 83 e 145 registos por segundo, o que aparentou um bom desempenho do sistema.

2.2.8 Visual Analysis of News Streams with Article Threads

Krstajić et al. [16] apresentam uma ferramenta que permite visualizar a evolução das notícias em tempo-real. Esta mistura os artigos agrupando-os em tópicos e retendo apenas a informação relevante. Assim, reduz-se a sobrecarga de dados a serem visualizados. Cada notícia obtida pela stream contém atributos como *timestamps*, *tags* e outros metadados. As tags são utilizadas pelo algoritmo que compara cada notícia e depois agrupa-as em tópicos. Cada uma delas tem uma classificação associada, baseada no número de apresentações nos artigos, e é utilizada para atribuir um artigo a uma determinada categoria.

Relativamente à visualização, os autores referem que devido à maioria das técnicas existentes asso-

ciarem o eixo-X à dimensão temporal dos conjuntos de dados, optaram por não alterar este princípio, e numa tentativa inicial, estes decidiram definir um tamanho fixo para cada categoria e apenas permitir um posicionamento livre dentro de desse subespaço. No entanto esta solução fornece uma visualização onde se tem uma resolução completa da stream de notícias e todas elas são representadas no ecrã, conforme a sua chegada, estando localizadas mais à direita. A cor de cada item está associada à sua classificação, onde um tom entre verde e vermelho representa itens com classificação positiva e negativa, respetivamente, e um item com tonalidade cinzenta demonstra a sua neutralidade. No entanto, esta alternativa inicial falha na representação das relações entre os vários itens. Para responder a esse problema, é proposto um algoritmo que compara os itens e associa-os a um tópico conforme a sua similaridade, distingue os artigos relevantes dos irrelevantes e otimiza o desempenho da visualização ao remover os mais irrelevantes quando um novo item é obtido. Nesta nova alternativa (Figura 2.23), cada artigo é representado por um retângulo, cuja largura corresponde à duração do tópico e a sua posição corresponde à data em que o primeiro item foi introduzido no tópico. O esquema de cores utilizado mapeia o número de artigos em cada tópico, onde tópicos com poucos itens têm um tom cinza e os restantes um tom avermelhado. Conforme o tempo avança, é possível observar que apenas os tópicos com mais itens, se mantém no ecrã.

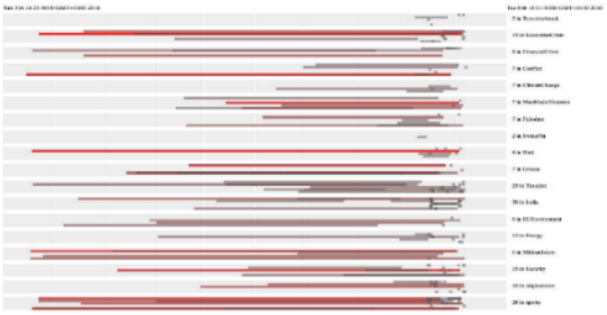


Figura 2.23: Visualização dos dados em *streaming* agregados por tópicos [16].

Como fonte dos dados em *streaming*, foi utilizado o Europe Media Monitor (EMM)[10] que agrega notícias em múltiplos idiomas e processa entre 80 000 e 100 000 artigos por dia.

2.2.9 Real-Time Visual Analytics for Event Data Streams

Fischer et al. [11] apresentam um sistema – Event Visualizer, que tenta responder a alguns problemas existentes quando se pretende visualizar eventos em *streaming*. Este sistema é modular e *loosely coupled*, e permite coletar, processar, analisar e visualizar de forma dinâmica, *streaming* de dados de eventos em tempo-real. O sistema fornece uma framework que possui diversas visualizações interligadas e interativas que se focam nos diferentes aspetos deste tipo de informação e possibilitam o reconhecimento de anomalias num sistema, o mais cedo possível. Esta ferramenta possibilita a visualização de eventos históricos e eventos em tempo-real, apresentando uma linha cronológica para interagir diretamente com as múltiplas streams.

[10] http://emm.newsexplorer.eu/

Para implementar este tipo de visualizações, os autores identificaram que era necessário existir uma exploração interativa com respostas e atualizações em tempo-real. Tornar a frequência das alterações nos dados já visualizados mínima de maneira a manter os padrões existentes sempre visíveis. O sistema necessita de um mecanismo de encaminhamento para enviar os eventos o mais rapidamente possível para as visualizações. Tornando o *Back-End* um ponto crucial para todo o sistema referido. Para fornecer esta flexibilidade, o sistema foi dividido em módulos denominados: Event Service, Event Analyzers e Event Visualizers. Os dois primeiros, têm na sua essência, a receção e registo das streams de dados, o pré-processamento e a conversão dos eventos para eventos mais genéricos que possam ser interpretados da mesma forma. Já o Event Visualizer, é uma interface gráfica que disponibiliza os dados históricos e atuais através de um conjunto de diferentes visualizações. Sendo que para fornecer uma visão global dos eventos, foi optada uma visualização com linha temporal. No entanto, uma linha temporal absoluta iria provocar a oclusão de eventos potencialmente importantes, por isso, foi implementada uma visualização do tipo *Relaxed timeline* (Figura 2.24), em que cada barra horizontal representa as linhas temporais para uma determina categoria e a sua escala é definida pela escala temporal. Um evento é representado por um retângulo colorido (conforme a sua importância) e a sua área depende da quantidade de eventos que ocorreram na respetiva stream, sendo uma desvantagem quando existem muitos eventos num só intervalo. Para resolver esse problema, foi definido um limite de eventos por intervalo, e quando novos eventos ultrapassam esse limite, a aplicação irá remover aqueles que possuem pior resultado na classificação obtida através de algoritmos de classificação baseados na frequência, tipo e definições definidas pelo utilizador para os eventos, sendo esses substituídos pelos novos. Os novos eventos estão posicionados nos intervalos mais à direita, sendo por isso o lado direito composto por eventos mais recentes que o esquerdo.

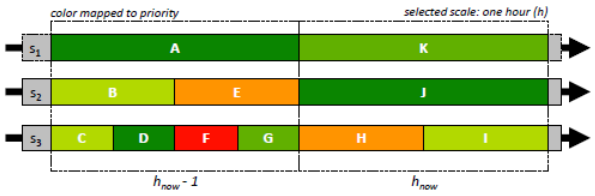


Figura 2.24: Visualização eventos de diferentes streams através de múltiplas *Relaxed timelines* [11].

Relativamente à exploração interativa, o utilizador pode fazer Pan & Zoom em qualquer área para investigar os eventos. É ainda possível selecionar eventos mudando a cor. A alteração entre os eventos históricos e os mais recentes é feito através de um slider, e a transição é feita com animações suaves.

O sistema foi testado com cerca de 100 milhões de eventos, onde em cenários com maior taxa de transferência, foram processados cerca de 225000 eventos por hora.

2.3 Discussão

Nas secções anteriores foi feita uma apresentação resumida de vários artigos que expõem técnicas e ferramentas existentes para a representação de grandes quantidades de informação, criando uma visão

global sobre o estado da arte no que toca às visualizações estáticas e dinâmicas de *Big Data*.

A Tabela 2.2 resume cada um dos artigos presentes no Capítulo 2, relacionando-os com critérios como a Grandeza de Volume de Dados e a sua Dinâmica, como referido na Tabela 2.1 (da Secção 2.1) e também por Ali et al. [1]. Relaciona ainda esses artigos, com as Técnicas de Visualização de cada um, as Técnicas de Redução Dimensional dos dados e outras que sejam relevantes. A Grandeza de Volume de Dados permite entender até que ponto o sistema possibilita representar grandes quantidades de informação sem comprometer o normal funcionamento do mesmo. A comparação com o critério Tempo Real possibilita perceber se este suporta fluxos de dados dinâmicos ou estáticos. Os Tipos de Visualização contêm as técnicas de representação da informação utilizadas pelo sistema. Por fim, os dois últimos critérios são a Redução Dimensional e Técnicas Relevantes, onde o primeiro permite conferir se o sistema executa algum tipo de técnica que permita reduzir a quantidade de dados. E o segundo, as técnicas cujos autores de cada artigo revelaram importantes ou que permitam evidenciar o sistema em causa.

Ainda que muitos dos sistemas existentes consigam representar dados em *streaming*¸ está bastante limitada a quantidade de informação que se pode representar mantendo o estado de coerência da visualização. Isto porque há fontes de dados que enviam enormes quantidades de informação em pequenos intervalos de tempo para esses sistemas e o tipo de visualização optado não permite que tamanhos volumes de informação sejam visualizados. É necessário por isso, conciliar estes tipos de visualização com a quantidade de dados, tendo em conta que os mesmos são obtidos em tempo real. Os sistemas StreamSqueeze [20], MeDICi [6] e Event Visualizer [11], conseguem apresentar volumes de dados na ordem dos 10^5 dados/hora ($\approx$ 28, 63 e 145 dados por segundo, respetivamente), sendo uma limitação para quando se pretende visualizar maiores volumes de informação em intervalos de tempo menores. Em conformidade com a quantidade de informação, também o tipo de dados afeta a forma como estes são representados.

A maioria dos sistemas analisados limita-se à representação de sequências de valores numéricos, ou seja, dados quantitativos. Apesar dos sistemas Topic-aware [23] e Streaming LogData [27] possuírem capacidade de representar dados heterogéneos, ou seja de diversos tipos de dados, estes acabam por impor a existência de um pré-processamento moroso, devido à necessidade de uniformizar todos os formatos recebidos, além da resolução dos potenciais problemas que possam existir, tanto na estrutura, como na formatação de cada um deles.

Como solução para o problema descrito nos parágrafos anteriores, dada a grande quantidade de informação que se pretende visualizar, assim como as limitações impostas pelo seu tipo de dados, há uma necessidade de utilização de métodos de simplificação e redução dos mesmos, já que deste modo se obtêm conjuntos de informação consideravelmente menores. É crucial no entanto, garantir que não haja perdas na visualização desse resultado comparativamente com o do estado original. Por vezes, mesmo após realizar essas técnicas de simplificação, limpeza ou redução dos dados existentes, a grandeza dos mesmos poderá ainda ser demasiado grande, não sendo possível representá-los nos dispositivos atuais, isto devido ao espaço existente no ecrã ser limitado. Como tal, alguns sistemas presentes nos artigos salientam a importância de métodos de agregação e simplificação dos dados,

Tabela 2.2: Relação entre artigos da secção 2 e critérios.

Artigo	Tempo Real	Volume de Dados	Tipos de Visualização	Redução Dimensional	Técnicas Relevantes
DeepEye [22]	x	?	Linechart / Barchart / Piechart	x	-
imMens [19]	x	10^8	Binned Plots & Geographic Heatmap	Binned Aggregation	-
Topic-aware [23]	x	10^3	?	x	-
3DVis-LodCloud [8]	x	?	Barchart 3D	x	-
RBPCP [28]	x	?	Parallel Coordinate Plot	x	Bundled Parallel Coordinate Plot / Median-Based Rearrangement
Bin-summarise-smooth [26]	x	10^8	Binned Plots / Small Multiples	Binning / Summarize / Smoothing	Pelling / Modulus Transformation
ID-Map [13]	x	10^4	ID-Map	x	ID-Map
Circle Segments [3]	x	10^5	Circle Segments	x	Circle Segments
BinX [5]	x	10^3	Linechart / Scatterplot	Aggregation	Trapeze-like effect para níveis de agregação
ScalaR [4]	x	?	Scatterplot / Linechart / Histogram / Mosaicplot / Heatmap / Treemap	Aggregation / Sampling / Filtering	-
LiveRAC [21]	x	?	Linechart / Barchart	x	Semantic Zooming
VALID [18]	✓	10^8	Binned Points / Binned & Bundling Lines	x	Edge Bundling
Streaming LogData [27]	✓	30 clusters	Parallel Coordinate Plot / Selection Widget / Barchart / Radarchart / Time Sequence chart / Treemap	✓N.D.	-
Density Displays [12]	✓	?	Time Series with multiple cells	x	Circular Overlay Display / Variable Resolution Density Display
I^2 [25]	✓	?	Time Series plot	M4	M4
StreamSqueeze [20]	✓	? (10^5/hora)	?	x	Semantic Zooming
Streamit [2]	✓	? (10205 docs c/ 2036 keywords)	Multiple Circular charts	LDA	LDA
MeDICi [6]	✓	10^8 (522 * 10^3/hora)	Heatmap / Sparkline / Time Wheel	SAX - Symbolic Aggregation	Combinação de duas visualizações
News Streams [16]	✓	? (4166/hora)	?	Aggregation	-
Event Visualizer [11]	✓	10^8 (255 * 10^3/hora)	Relaxed Timeline	x	-

enquanto outros optam por utilizar técnicas que fornecem visualizações que se adaptam ao espaço disponível no ecrã. No entanto, estes acabam por remover os dados mais antigos ou os que são menos relevantes para que possam ser substituídos por novos, provocando alguma perda de informação que poderá ser crucial para o padrão mental criado pelo utilizador.

As transições tomam um dos papéis mais importantes no que toca a manter o utilizador atualizado das transformações existentes na visualização e todo o contexto da informação. Alguns dos artigos analisados denotam, por isso, que os respetivos sistemas apresentam transições suaves para que o utilizador não perca o contexto da visualização com o decorrer das alterações que possam existir, como referido anteriormente. Demarcam ainda a importância dessa suavização das transições para que seja possível manter os padrões existentes dos dados já visualizados sempre disponíveis. Esta exigência poderá ser considerada também, através da sobreposição de uma visualização onde apenas se representam os dados novos, como descrito no sistema Density Displays [12], deixando de ser necessário deslocá-los ao longo da visualização. Desta forma, apesar do desempenho do sistema ser melhorado dada a redução do número de movimentações, é possível detetar que a sobreposição gera alguma confusão aos utilizadores que testaram esse sistema. Deve haver por isso, uma preocupação maior com a forma como os novos dados são obtidos e representados para que o utilizador mantenha o "quadro geral" e ao mesmo tempo consiga ver a nova informação, podendo compará-la com o histórico.

Analisando todos estes critérios e após relacioná-los com os artigos, verifica-se que os sistemas até hoje existentes ainda têm dificuldade na representação dinâmica de grandes fluxos de informação, seja porque o seu objetivo é representar quantidades menores de dados, ou porque os sistemas não estão preparados para tais dimensões, isto é, preparados para criar visualizações que na sua constituição têm largas quantidades de informação, obtida em tempo real. Dada a relevância das transições para este tipo de visualizações, denota-se ainda pouco desenvolvimento, pelo que é imprescindível encontrar uma solução que possua técnicas eficientes de redução dimensional dos dados e ainda transições suaves ao longo de toda a visualização para permitir o fácil reconhecimento dos dados recém-obtidos, mas ao mesmo tempo, mantendo com clareza todos os padrões dos registos anteriores, de forma a ser possível compará-los durante a análise da informação em tempo real.

Capítulo 3

Vismillion and Change

Neste capítulo são apresentadas as várias técnicas de transição concebidas, tendo como objetivo transitar os dados entre duas técnicas de visualização. Mas primeiro é explicado o conceito VisMillion [10], assim como as várias técnicas de visualização que foram utilizadas para representar a informação existente.

3.1 Conceito VisMillion

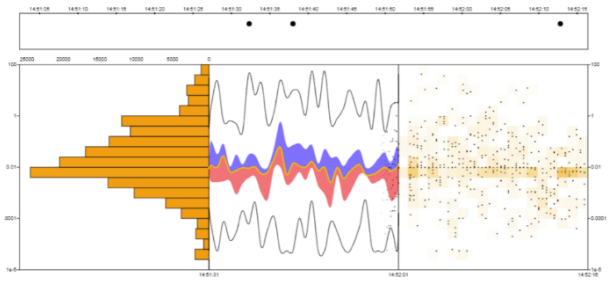
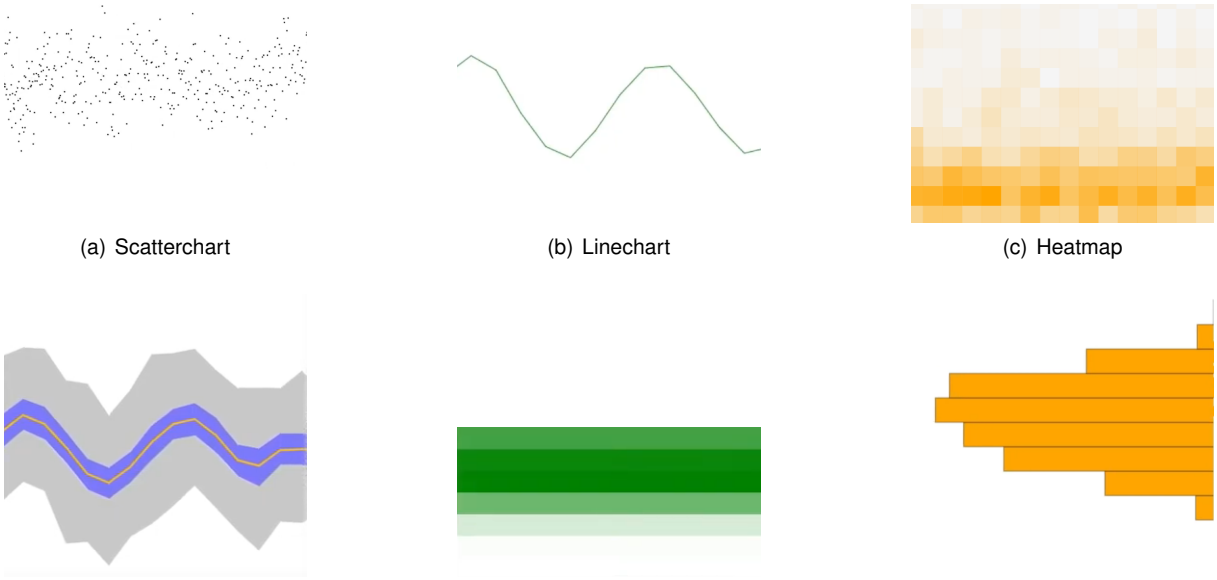


Figura 3.1: Visualização de transações BTC através do sistema VisMillion

O conceito VisMillion [10] permitiu iniciar o desenvolvimento deste trabalho. Este tem como objetivo possibilitar a visualização de grandes quantidades de dados em tempo real, beneficiando de vários módulos lado a lado, que se complementam simultaneamente, criando uma visão contínua da informação que é apresentada da direita para a esquerda, passando de um módulo para outro. A Figura 3.1 apresenta a visualização de transações BTC através do VisMillion e permite representar os módulos alinhados horizontalmente, como descrito anteriormente. Cada um dos módulos representa os dados em função do tempo de maneiras distintas, isto é, quanto mais recentes são, maior é o seu detalhe. Aplicando não só técnicas de visualização diferentes (Linechart, Scatterchart, Barchart), como também diferentes métodos de agregação da informação. Opta ainda por realizar uma degradação graciosa da informação, onde o sistema utiliza medidas estatísticas como a média e quartis para agregar os dados mais antigos, dispondo-os pelos módulos seguintes conforme novos dados vão sendo introduzidos na visualização.

3.2 Técnicas de visualização

Para tornar possível o estudo das transições que melhor se aplicam para cada par de tipos de visualização, aplicaram-se diversas técnicas que permitem representar a informação de formas diferentes para possibilitar a apresentação dos dados com diferentes níveis de agregação e detalhe. Partindo do conceito desenvolvido no sistema VisMillion [10], desenvolveram-se técnicas como Scatterchart, Linechart, Heatmaps (que pode ser representado de dois modos diferentes consoante a agregação desejada), Streamgraph e ainda Barchart. A Figura 3.2 pretende evidenciar cada uma destas visualizações que serão discutidas nas subsecções que se seguem.



(a) Scatterchart (b) Linechart (c) Heatmap

(d) Streamgraph (e) Heatmap Acumulador (f) Barchart

Figura 3.2: Técnicas de visualização aplicadas no sistema Vismillion and Change

3.2.1 Scatterchart

A técnica de visualização Scatterchart (Figura 3.2(a)), de um modo geral, permite relacionar duas variáveis dentro de um conjunto de dados, através de conjuntos de pontos que são posicionados entre os eixos vertical e horizontal, conforme os valores correspondentes a cada um deles. Possibilita ainda configurações adicionais como códigos de cores, dimensões e formas, permitindo representar e relacionar mais variáveis se desejado.

Como neste trabalho se pretende relacionar valores existentes com o seu contexto temporal (duas variáveis), optou-se por utilizar o Scatterchart como técnica de visualização que apresenta os dados no seu formato original, isto é, os dados que estão a ser recebidos em tempo real, sem terem sofrido qualquer agregação. Desta forma, facilitando a análise dos dados recém-obtidos que são visualizados com o máximo detalhe antes dos mesmos transitarem para módulos onde são realizadas as técnicas de agregação da informação. Este tipo de visualização está também preparado para representar informação agregada, não invalidando a sua utilização, caso seja esse o objetivo.

3.2.2 Linechart

A técnica de visualização Linechart (Figura 3.2(b)), representa uma série de pontos unidos através de uma linha que dá uma noção de continuidade aos dados. Esta técnica é normalmente utilizada quando se pretende apresentar tendências e padrões existentes nos dados, dividindo-os em intervalos de tempo para originar pontos interligados por uma linha.

No VisMillion and Change utilizou-se este gráfico com o objetivo de representar informação após a mesma ter sido agregada, dividindo os dados em intervalos de tempo que originam a sua **média**. Desta forma, a linha resultante do Linechart simboliza a ligação dos pontos gerados pelo cálculo da média dos dados em múltiplos intervalos temporais.

3.2.3 Heatmap

O Heatmap (Figura 3.2(c)) representa uma matriz que através de um esquema de cores divide a sua visualização em múltiplas zonas que refletem uma relação entre duas variáveis. Desta forma obtém-se uma visualização que permite comparar diferentes intervalos através das diferenças de cores que compõem cada um dos mesmos. Ao mesmo tempo, faculta a análise de padrões e correlações que possam existir no conjunto de dados a visualizar.

Para este estudo utilizou-se o Heatmap para representar a quantidade de informação existente em cada divisão predefinida, através das diferentes tonalidades de uma escala de cores. Quanto mais dados existirem numa célula do Heatmap, mais intensa será a sua cor. Assim, esta técnica é aplicada após ter sido feita uma agregação dos dados a visualizar, apresentando a concentração de pontos que existe num determinado intervalo de valores em relação a também um intervalo temporal, sendo por isso um **contador**.

3.2.4 Streamgraph

Este tipo de visualização é baseado no conceito da técnica das Boxplots. Uma Boxplot - Figura 3.3, é uma técnica que divide conjuntos de dados nos seus quartis, representando-os através de caixas com um segmento no seu interior simbolizando a mediana. Consoante o espaçamento entre as duas extremidades da caixa, indica a dispersão da distribuição dos dados. Através de linhas que partem das suas extremidades, permite ainda representar os valores máximo e mínimo do seu conjunto. Ao mesmo

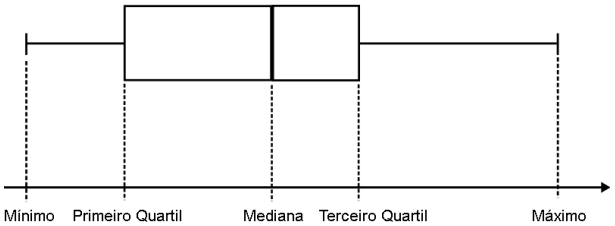
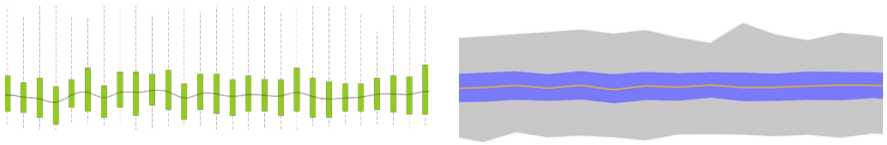


Figura 3.3: Estrutura da técnica de visualização - Boxplot

tempo que visualizar uma boxplot de forma isolada tem a vantagem de representar a distribuição de um conjunto de dados, esta pode ser utilizada ainda para comparar vários intervalos temporais e desta forma identificar distribuições assimétricas ao longo do tempo. Ao juntar múltiplas boxplots, cada uma pertencente a um intervalo de tempo, obtem-se a visualização apresentada na Figura 3.4(a), cuja união entre as diversas boxplots se faz através de uma linha que une os pontos resultantes da mediana de cada uma. No entanto, verificou-se posteriormente que uma alternativa a este tipo de representação seria através de um Streamgraph - Figura 3.4(b).



(a) Boxplots (b) Streamgraph

Figura 3.4: Boxplot para Streamgraph

O Streamgraph é um tipo de visualização que apresenta a informação dividida em várias categorias, demonstrando as diferenças no conjunto de dados ao longo de intervalos de tempo. O próprio nome indica a existência de fluxos contínuos de dados. Estes, são divididos por diversas categorias que são representadas através de diferentes cores para facilitar a sua distinção. Conforme a relação de cada categoria com os intervalos temporais, os vários valores provocam uma variação na dimensão da área de cada categoria, como se pode verificar na Figura 3.5. Este tipo de visualização é útil quando se pretende visualizar muita informação, permitindo descobrir tendências ao longo do tempo.

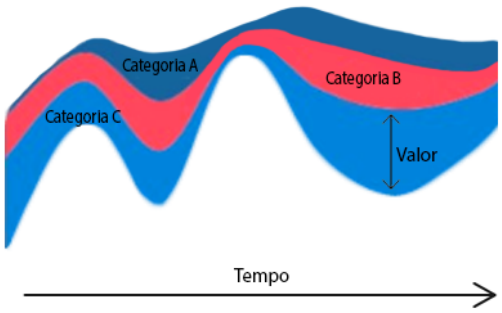


Figura 3.5: Estrutura da técnica de visualização - Streamgraph

Assim, utilizando um Streamgraph para representar múltiplas boxplots ao longo do tempo, foi possível dividir os valores retirados de cada uma por várias categorias, representando a cinzento o intervalo de valores até obter o **máximo e mínimo** de cada intervalo, como se pode observar na Figura 3.2(d), a azul o **1º e 3º quartis** e a **mediana** é representada através de uma linha que se encontra entre os mesmos. Desta forma podemos descobrir tendências e padrões nos dados existentes ao longo da sua evolução, sendo necessário realizar uma agregação dos dados antes de os visualizar.

3.2.5 Heatmap Acumulador

O Heatmap Acumulador (Figura 3.2(e)), é uma variante do Heatmap que permite visualizar a informação sem contexto temporal, isto é, vai acumulando toda a informação recebida e representa-a através de intervalos que dividem o seu domínio de valores em múltiplas barras. Essas barras vão modificando a sua cor consoante a quantidade de dados acumulados. Tal como anteriormente, faculta a comparação dos dados não permitindo neste caso visualizar a evolução ao longo do tempo, mas sim a concentração de dados (**contador**) a ter em conta em cada intervalo no instante em que é realizada a análise,

3.2.6 Barchart

Um Barchart (Figura 3.2(f)), é uma representação que permite dividir um conjunto de dados em várias categorias ou intervalos, que através de barras retangulares que variam a sua largura ou comprimento (conforme a representação horizontal ou vertical da mesma) em proporção com os valores que estas representam. Esta técnica permite comparar os vários intervalos de valores e detetar quais são aqueles que possuem uma maior quantidade de dados. Assim, é ainda possível verificar a distribuição de valores existente ao longo de toda a visualização.

Neste estudo, utilizou-se o Barchart para representar diversos intervalos de valores aos quais os dados que pertencem a cada um, se vão acumulando com o decorrer do tempo (**contador**). É utilizada a estrutura horizontal, com as barras a crescerem para a esquerda, de forma a dar continuidade ao fluxo da movimentação dos dados, também esta com uma direção é horizontal, da direita para a esquerda.

3.3 Transições entre visualizações

Dada a elevada importância da conservação do contexto de análise por parte do utilizador, assim como a necessidade permantente de comparação da informação existente com a que é obtida em tempo real, é essencial fornecer ao utilizador elementos que realizem uma ”degradação graciosa”da representação visual dos dados, em que para isso, é necessário representá-los de forma mais agregada e abstrata conforme estes se vão tornando cada vez mais antigos na visualização. Ao mesmo tempo, os intervalos temporais devem ser maiores nos níveis de agregação superiores, de forma a conseguir conservar a maior quantidade de informação no histórico da visualização.

Aliada à conservação do contexto referido anteriormente e de forma a tornar possível evidenciar padrões existentes nos dados a visualizar, as transições animadas tomam um papel bastante importante dado que tornam a visualização mais apelativa ao utilizador e ainda um melhor acompanhamento da mesma, conforme esta evolui ao longo do tempo, evitando assim uma perda do contexto da visualização por parte do utilizador. Foram então realizadas várias alternativas para possíveis transições entre dois tipos de visualização distintos, alinhados horizontalmente, que permitem responder aos requisitos descritos anteriormente, recorrendo a transições suaves para incorporar dois tipos de visualização que possuem níveis de agregação e velocidades diferentes.

Tendo em conta que o primeiro módulo será o que representa a informação assim que esta é recebida e para os dados serem apresentados sem qualquer agregação, estes serão melhor interpretados quando representados através de pontos que demonstram a relação valor/tempo existente. Assim, optou-se por considerar como primeira técnica de visualização, o Scatterchart. Como segunda técnica de visualização, foram utilizados Heatmaps (normal e acumulador), um Linechart, um Streamgraph e ainda um Barchart, descritos na Secção 3.2. Foram criadas várias alternativas que permitem transitar a informação do Scatterchart para cada uma dessas técnicas referidas. A representação mais à direita representando o Scatterchart, possui um intervalo temporal bastante reduzido, originando por isso uma velocidade superior à da representação da esquerda que por sua vez possui um intervalo temporal muito maior. Desta forma, os dados mais recentes são deslocados ao longo do eixo horizontal a uma velocidade semelhante à original, isto é, a velocidade com que os dados são recebidos, enquanto os dados mais antigos são visualizados durante mais tempo ou de forma infinita (no caso do Barchart e Heatmap Acumulador), funcionando como histórico. Desta forma, torna-se possível manter o estado do contexto criado pelo utilizador, assim como os padrões existentes durante mais tempo.

Em cada uma das secções que se segue será apresentado um modelo sem qualquer tipo de transição para que se possa analisar nos modelos seguintes as diversas alternativas.

3.3.1 Transição Scatterchart - Heatmap

Os quadrados do Heatmap são formados após a transição dos pontos correspondentes aos dados mais recentes e conforme a sua concentração por intervalo de tempo. Isto é, tonificam mais ou menos os quadrados respetivos, consoante a quantidade de pontos que existe naquele intervalo. Foi necessário numa primeira abordagem, encontrar diferentes formas de transformar um conjunto de pontos em quadrados com dimensão maior e pré-configurada, para dar origem a diferentes transições que permitam animar essas transformações.

A: Sem animação

Para ter como referência uma técnica, cujo objetivo é necessariamente transformar pontos em quadrados, foi concetualizada como transição, aquela que não possui qualquer tipo de animação, transitando os pontos para a representação da esquerda sem preocupação no que toca às diferentes velocidades, cores, dimensões e respetivas transformações - Figura 3.6.

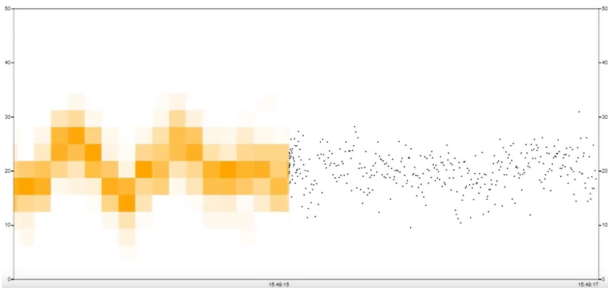


Figura 3.6: Transição não animada entre Scatterchart e Heatmap

B: Fade-in Fade-out

Pretende-se com a técnica Fade-in Fade-out, desvanecer os pontos enquanto se vão formando os quadrados do Heatmap e ao mesmo tempo que os pontos transitam para o interior de cada um dos quadrados. Esta transição (Figura 3.7) foca-se ainda nas diferenças dos intervalos temporais e resultantes velocidades das duas metades da visualização. Existe então uma desaceleração dos pontos permitindo a sua chegada ao Heatmap com a velocidade que o mesmo apresenta e desta forma fazendo corresponder os pontos aos quadrados a que pertencem. Para não existir um choque visual, fazendo desaparecer os pontos no mesmo instante em que os quadrados do Heatmap começam a ser criados, criou-se um desvanecimento gradual dos pontos enquanto os quadrados vão aparecendo também eles gradualmente, acentuando assim a ideia de que os quadrados são formados pelos pontos que desvanecem no momento da transição para o seu interior.

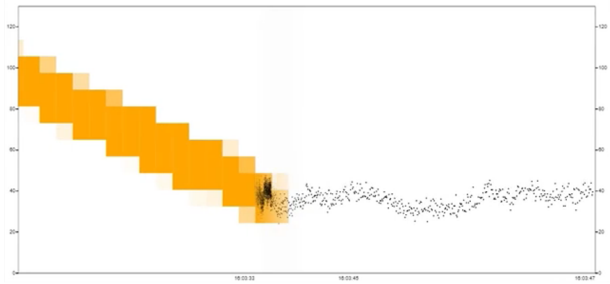


Figura 3.7: Transição Fade-in Fade-out entre Scatterchart e Heatmap

C: Aglomeração em quadrados

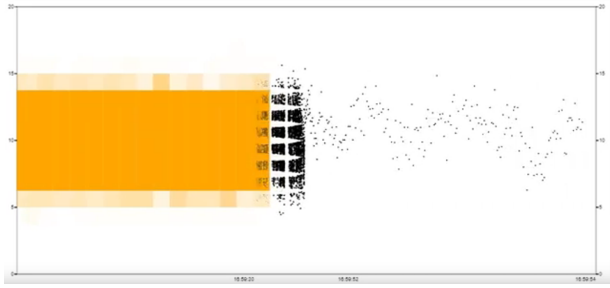


Figura 3.8: Transição de Aglomeração em quadrados entre Scatterchart e Heatmap

Com o objetivo de apresentar de uma forma mais clara a quantidade de pontos que pertence a cada quadrado do Heatmap, esta técnica forma quadrados de múltiplos pontos agregados, ou seja, vai aglomerar os pontos que correspondem a cada um dos quadrados num pequeno núcleo, que por si, formam um quadrado de dimensão semelhante à dos quadrados do Heatmap (Figura 3.9a). Verificando-se então a necessidade de deslocar os pontos com a direção e aceleração desejadas, de forma a fazer coincidir todos os pontos com o interior do quadrado correspondente. Assim que todos os pontos estiverem no seu interior, os quadrados do Heatmap vão aparecendo de forma gradual (Figura 3.9b) até à sua conclusão (Figura 3.9c). Finalmente existe um desvanecimento dos pontos de cada coluna de quadradados já formada, momentos antes da repetição do ciclo anterior, como é vísivel na Figura 3.8.

(a) Aglomeração de pontos

(b) Aparecimento gradual dos quadrados

(c) Aglomerado dos pontos e aparecimento gradual dos quadrados

Figura 3.9: Etapas para aglomeração em quadrados com Transição entre Scatterchart e Heatmap

D: Colunas de dados

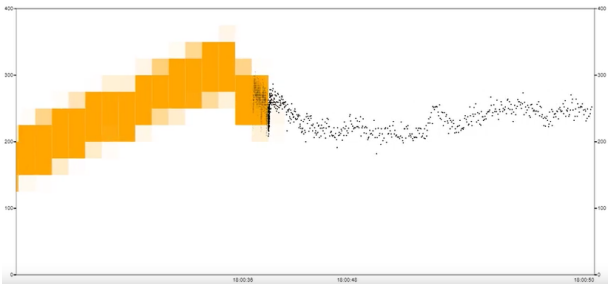


Figura 3.10: Transição de Colunas de dados entre Scatterchart e Heatmap

Semelhante à técnica anterior, esta parte do princípio da formação de colunas de pontos (Figura 3.11a), permitindo observar a posição dos pontos relativa ao eixo vertical e desta forma compreender melhor os seus valores. Assim, os pontos correspondentes a cada coluna (*bin*) são deslocados de forma desacelerada até que todos estejam compreendidos entre o começo e o término de cada uma. Quantos mais pontos entrarem nas colunas de quadrados do Heatmap, mais nítidos estes serão, provocando um aparecimento gradual dos mesmos, como é possível observar nas Figuras 3.11b e 3.11c. Só depois de todos os pontos pertencentes a uma coluna estarem no seu interior é que se passa para o processo do desvanecimento dos pontos, como a Figura 3.10 pretende ilustrar, até à repetição do processo.

(a) Formação de coluna de pontos

(b) Aparecimento gradual dos quadrados

(c) Coluna de pontos e aparecimento gradual dos quadrados

Figura 3.11: Etapas para criação de colunas com Transição entre Scatterchart e Heatmap

E: Granulado

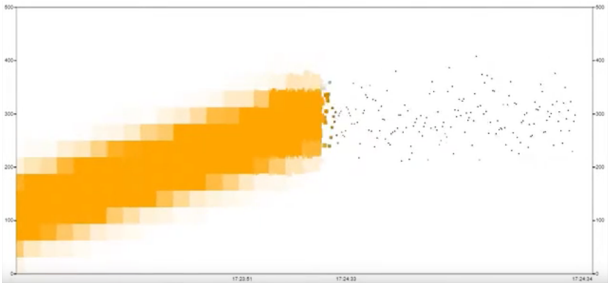


Figura 3.12: Transição Granulado entre Scatterchart e Heatmap

Com a ideia de tornar os pontos cada vez maiores até que estes tenham o tamanho de um quadrado do Heatmap e desta forma induzir visualmente a criação desses quadrados gradualmente, esta possibilidade de transição passa por aumentar os pontos fazendo-os crescer (Figura 3.13a) enquanto se deslocam para a esquerda de forma a criarem núcleos de quadrados cada vez maiores (Figura 3.13b). Quando esses núcleos de quadrados sofrem uma desaceleração, começam a ficar sobrepostos acabando por dar origem a quadrados que se encontram na mesma posição dos quadrados do Heatmap que vão surgir (Figura 3.13c). Resultando por fim, numa transição (Figura 3.12) que vai sofrendo alguns ajustes na posição dos quadrados ao longo do seu deslocamento, com o objetivo de tornar o surgimento dos quadrados do Heatmap menos notável e desta forma não afetando a visualização.

(a) Aumento da dimensão dos pontos

(b) Aparecimento de núcleos de quadrados

(c) Crescimento dos pontos para quadrados cada vez maiores

Figura 3.13: Etapas para criação de quadrados com Transição entre Scatterchart e Heatmap

3.3.2 Transição Scatterchart - Linechart

A linha do Linechart é formada após a transição dos pontos correspondentes aos dados mais recentes e conforme a sua média por intervalo de tempo. Ou seja, quanto maior a média num determinado intervalo temporal, mais elevado (em relação ao eixo vertical) será o ponto de ligação da linha naquele intervalo. Foi necessário numa primeira abordagem encontrar diferentes formas de transformar um conjunto de pontos numa linha cuja representação, sendo uma média, pode apresentar um choque visual no momento da transição.

A: Sem animação

Com o objetivo de transitar pontos para uma linha que representa a média dos mesmos, apresenta-se na Figura 3.14 uma transição não animada, que permite retratar o choque visual que poderá ser causado quando existe uma grande variação nos pontos recebidos. Os pontos transitam para a representação da esquerda sem preocupação, no que toca às diferentes velocidades, cores e respetivas transformações.

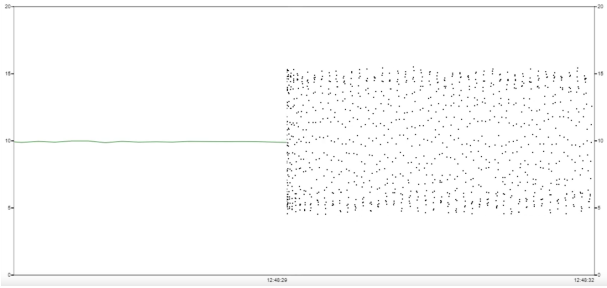


Figura 3.14: Transição não animada entre Scatterchart e Linechart

B: Fade-in Fade-out

Com o objetivo de desvanecer os pontos enquanto a linha que representa a sua média vai sendo formada ilustrando mais facilmente os pontos que vão originar a nova alteração no Linechart, foi estudada a transição Fade-in Fade-out - Figura 3.15. Além de tornar mais nítida a formação da linha, através de um aparecimento gradual, também os pontos vão desvanecendo. A transição foca-se ainda nos intervalos temporais de cada parte da visualização, tentando compensar as velocidades resultantes através de uma desaceleração dos pontos, permitindo a sua chegada ao Linechart no momento em que o mesmo começa a formar a linha correspondente.

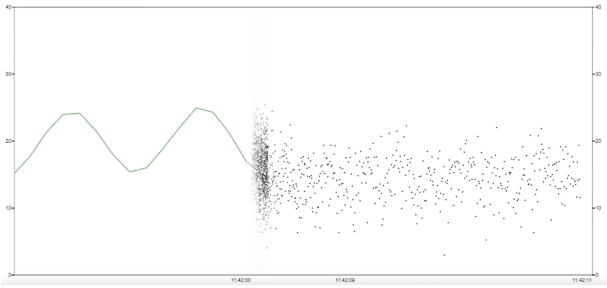


Figura 3.15: Transição Fade-in Fade-out entre Scatterchart e Linechart

C: Afunilamento

Inicialmente foi estudada uma forma que permitisse convergir todos os pontos de forma contínua, fazendo com que os pontos fossem ao encontro da linha resultante do Linechart. A ideia passaria por ir afunilando os pontos conforme estes chegam ao seu momento de transição para uma linha. Como existe uma grande diferença entre os intervalos de tempos das duas metades de visualização, tornou-se necessário, de forma a não comprometer a posição dos pontos, convergir os dados em intervalos,

dado que esta solução torna mais intuitivo compreender quais são os pontos que pertencem a cada valor da média representado no Linechart.

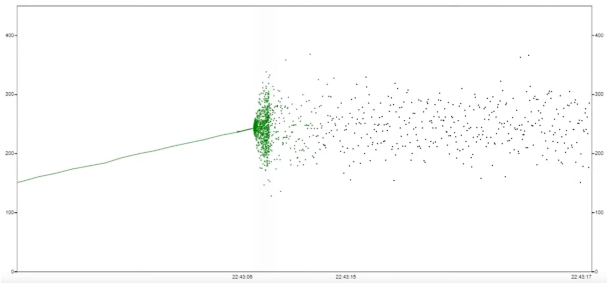
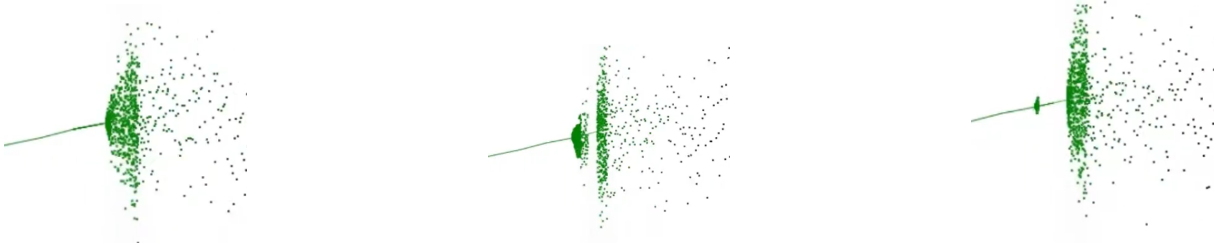


Figura 3.16: Transição de Afunilamento entre Scatterchart e Linechart

Assim, a transição representada na Figura 3.16 pretende ir acumulando os pontos que pertencem a um intervalo de tempo, para posteriormente começar a convergi-los para um ponto do Linechart que representa a média de valores desse intervalo. Apenas quando todos esses pontos estão concentrados e sobre a linha, é que estes começam a desvanecer e pode recomeçar um novo ciclo para o intervalo de dados seguinte - Figura 3.17.



(a) Pontos começam a convergir

(b) Separação dos intervalos e continuação da convergência do primeiro intervalo

(c) Pontos concentrados na linha antes de desvanecerem

Figura 3.17: Etapas para convergir pontos para linha com Transição entre Scatterchart e Linechart

D: Contração de pontos

Com o objetivo de gerar um só ponto que permita juntar-se à linha resultante do Linechart, que representa a média de valores correspondente a um intervalo de dados, foi estudada uma alternativa que pretende contrair cada intervalo de pontos até gerar um único que representa o valor da média. Numa primeira tentativa, foi idealizado formar a linha a partir da união dos pontos, deslocando-os verticalmente até ”caírem”e formarem um conjunto que formaria uma linha. No entanto, esta abordagem mostrou-se pouco eficaz para quando existem poucos pontos dentro de um mesmo intervalo.

A ideia acabou por tornar possível criar a transição (Figura 3.18), que numa fase inicial trata de agregar todos os pontos que pertencem a um intervalo temporal - Figura 3.19(a) e através de diferenças na aceleração para com os intervalos anteriores, provoca uma separação entre os mesmos - Figura 3.19(b). Desta forma, pode começar a concentrar os pontos todos formando um núcleo que vai enco-

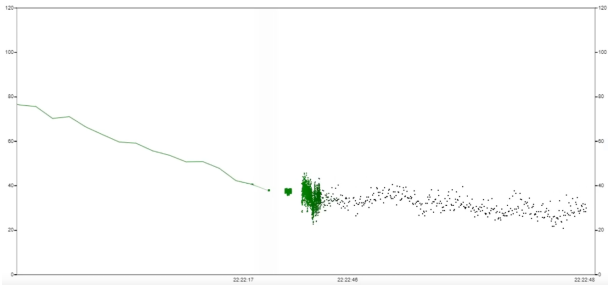
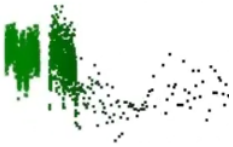


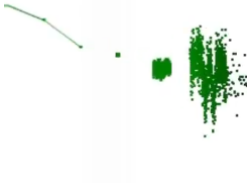
Figura 3.18: Transição de Contração de pontos entre Scatterchart e Linechart

lhendo até dar origem a um só ponto. Quando o Linechart obtiver toda a informação necessária para representar esse novo intervalo, irá unir a linha já existente com esse novo ponto - Figura 3.19(c).

(a) Agregação de pontos de um intervalo



(b) Separação dos intervalos e começo da contração dos pontos



(c) Contração dos pontos para união com a linha existente

Figura 3.19: Etapas de contração de intervalo de pontos para criação de linha com Transição entre Scatterchart e Linechart

3.3.3 Transição Scatterchart - Streamgraph

O Streamgraph é formado após a transição dos pontos e pretende ilustrar os valores máximos e mínimos, mediana, 1º e 3º quartis de cada intervalo de tempo. Para representar as múltiplas áreas, é essencial estudar técnicas para criar transições suaves e coerentes dos pontos para cada uma delas.

A: Sem animação

Com o objetivo de transformar pontos em áreas coloridas, é primeiro apresentada na Figura 3.20, uma transição em que os pontos são transitados para o Streamgraph sem preocupação no que toca às diferentes velocidades das duas metades da visualização, cores e respetivas transformações animadas.

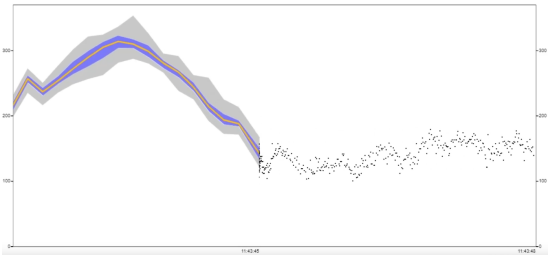


Figura 3.20: Transição não animada entre Scatterchart e Streamgraph

B: Fade-in Fade-out

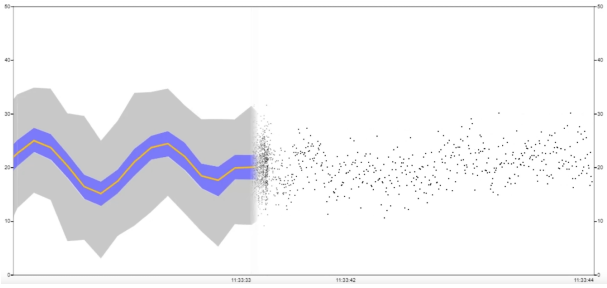


Figura 3.21: Transição Fade-in Fade-out entre Scatterchart e Streamgraph

A transição Fade-in Fade-out pretende suavizar o choque visual que existe como resultado da passagem direta, de um tipo de visualização, para o outro. Fazendo desvanecer os pontos enquanto as áreas que correspondem a cada uma das técnicas referidas anteriormente vão sendo geradas, ilustrando mais facilmente os pontos que originam cada intervalo - Figura 3.21. A transição foca-se também nos intervalos temporais e por sua vez nas velocidades de cada um, provocando uma desaceleração dos pontos antes de transitarem para o Streamgraph, e desta forma permitindo a sua chegada no mesmo momento em que as áreas do Streamgraph são representadas.

C: Estreitamento dos pontos

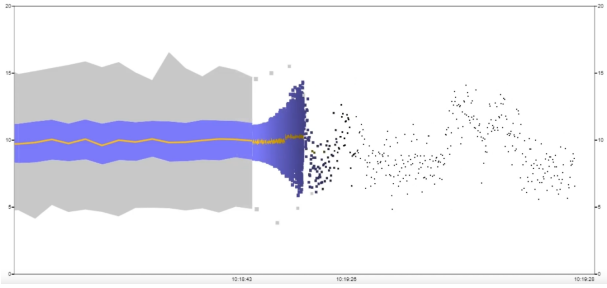


Figura 3.22: Transição de Estreitamento dos pontos entre Scatterchart e Streamgraph

Como alternativa, para este tipo de transição foi estudada uma forma (Figura 3.22) de a animar fazendo uma espécie de ”estreitamento”dos pontos, para delimitar logo na transição quais são os pontos que vão ser relevantes no Streamgraph e deixando os restantes de fora. Assim é provocada uma convergência dos pontos para a área que representa o 1º e 3º quartil de cada intervalo no Streamgraph. Os pontos sofrem numa fase inicial um aumento ligeiro da sua dimensão e vão alterando a cor para a fazer corresponder à cor da área que representam, como é possível observar na Figura 3.24(a). De notar que os pontos cujo valor é igual à mediana do intervalo pertencente, estão demarcados também com a cor que é representada no Streamgraph, permitindo desde cedo criar uma ligação entre as duas visualizações. Os pontos que representam o valor máximo e mínimo de cada intervalo são também eles representados na visualização - Figura 3.23.

Tendo sido experimentada esta técnica, decidiu-se depois adicionar outra animação para evitar o choque visual na correspondência das áreas e cores do Steamgraph com a transição, e enquanto

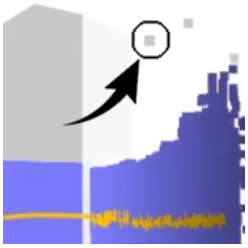
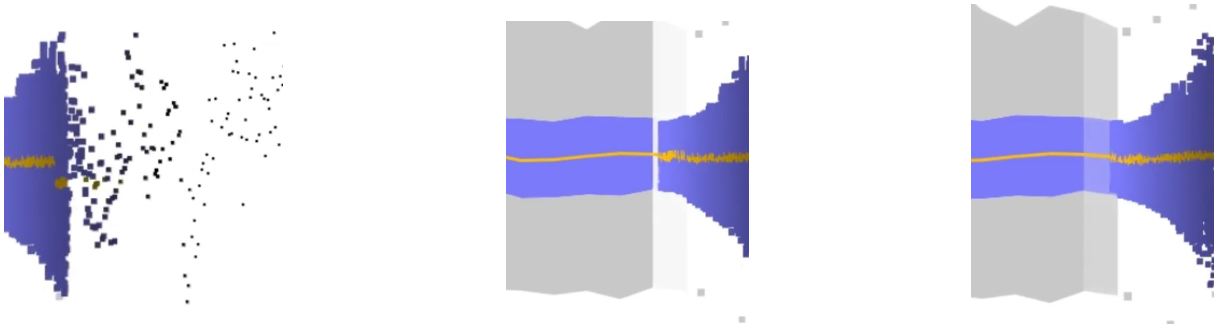


Figura 3.23: Valor máximo assinalado na Transição de Estreitamento dos pontos

um determinado intervalo do primeiro vai sendo gerado, a transição responsabiliza-se por ir fornecendo cada vez mais cor ao mesmo (removendo a transparência), facilitando assim o seguimento da informação ao longo da visualização. O ciclo descrito está apresentado na Figura 3.24.



(a) Aumento da dimensão dos pontos e convergência para quartis

(b) Áreas coloridas começam a aparecer gradualmente

(c) Resultado do passo anterior

Figura 3.24: Etapas de Estreitamento dos pontos com Transição entre Scatterchart e Streamgraph

D: Estampado de pontos

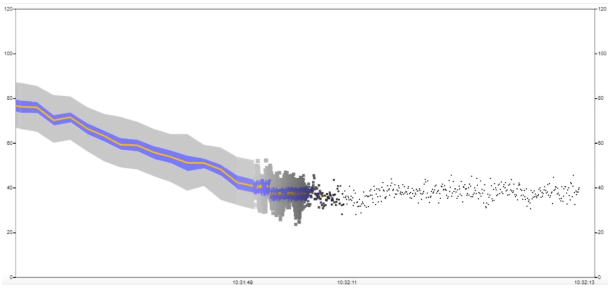
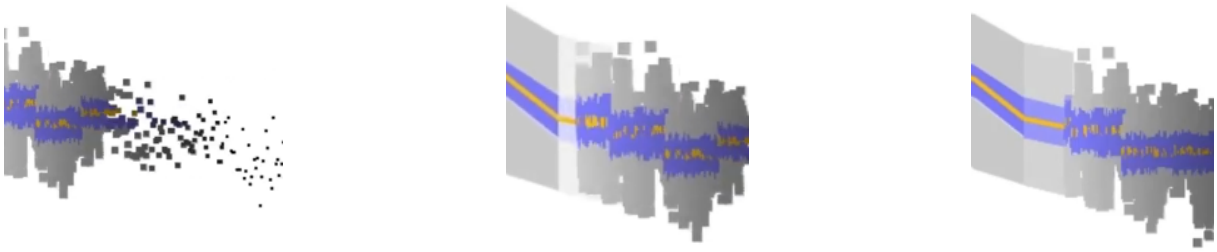


Figura 3.25: Transição de Estampado de pontos entre Scatterchart e Streamgraph

Com o objetivo de manter presentes na transição todos os pontos que são recebidos e ao mesmo tempo tentar adaptá-los ao Streamgraph, mantendo as posições originais. Definiu-se que a melhor solução passaria por adaptar as cores às cores das áreas do Streamgraph, a que cada ponto pertence. Desta forma, criou-se uma técnica (Figura 3.25) que gradualmente vai dando cor aos pontos conforme o valor destes. Assim, um ponto que tenha um valor igual ao valor do 1º e 3º quartis, por exemplo, irá obtendo

a cor correspondente a essas áreas. Posteriormente e com o intuito de se criarem áreas semelhantes às do Streamgraph, ainda durante a transição, e reduzindo o choque visual, optou-se por aumentar gradualmente a área de cada ponto - Figura 3.26(a), facultando algumas modificações e ajudes suaves nas posições de cada ponto, para que na altura da criação das áreas do Streamgraph daquele intervalo, os pontos estivessem alinhados com as mesmas.

Da mesma forma que na técnica 3.3.3, adicionou-se uma área que permite ir adicionando cor à transição de forma gradual para evitar perda de contexto enquanto os pontos vão sendo transitados para o módulo seguinte, como é possível observar na Figura 3.26(b) e Figura 3.26(c).



(a) Aumento gradual da dimensão dos pontos

(b) Junção dos diversos intervalos e aparecimento gradual das áreas

(c) Resultado do passo anterior

Figura 3.26: Etapas do Estampado de pontos com Transição entre Scatterchart e Streamgraph

3.3.4 Transição Scatterchart - Barchart

Cada barra do Barchart é formada e incrementada após a transição dos pontos correspondentes a cada intervalo de valores. Isto é, tendo em consideração que o gráfico de barras foi utilizado como um acumulador, quanto maior a quantidade de pontos que existe naquele intervalo, maior será a barra representante. Foi necessário, numa primeira abordagem, encontrar diferentes formas de transformar os pontos, para facilitar a interpretação, durante a sua transição. Dado que os pontos são de uma dimensão muito menor e a sua cor difere do Barchart, realizaram-se alternativas para diferentes transições.

A: Sem animação

Como abordagem inicial, optou-se por representar uma transição sem animações que permitisse transitar os pontos para o Barchart, sem preocupação relativamente às cores ou dimensões - Figura 3.27.

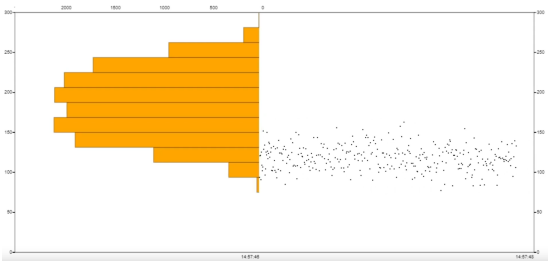


Figura 3.27: Transição não animada entre Scatterchart e Steamgraph

B: Fade-in Fade-out

Com o objetivo de desvanecer os pontos enquanto a o Barchart vai sendo preenchido com os mesmos, foi estudada a transição Fade-in Fade-out - Figura 3.28. Nesta transição os pontos vão desvanecendo consoante são transitados para as barras do Barchart. Além dos pontos, as próprias barras são representadas na transição e possuem um gradiente de cor que permite, também aqui, sugerir um aparecimento gradual do Barchart.

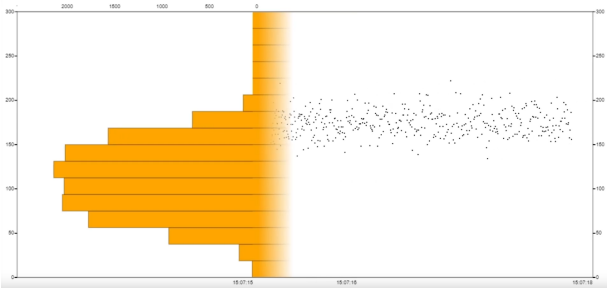


Figura 3.28: Transição Fade-in Fade-out entre Scatterchart e Linechart

C: Guias

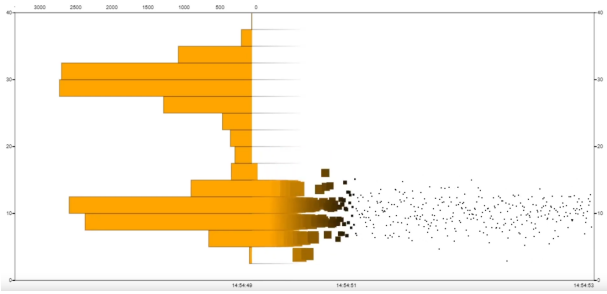


Figura 3.29: Transição de Guias entre Scatterchart e Barchart

Como alternativa, foi estudada uma técnica que permitisse seguir com mais precisão, a transformação dos pontos em barras, seguindo a ideia de que estão a ser encaminhadas para dentro das mesmas. Esta transição (Figura 3.29) trata de, numa fase inicial aumentar as dimensões dos pontos para que os mesmos passem a ter uma dimensão igual à altura das barras do Barchart e assim quando os pontos alcançarem as barras é possível verificar que existe mais continuidade e menos saltos visuais perante o surgimento de novos pontos. A Figura 3.30(a), sugere esse aumento da área dos pontos e ao mesmo tempo a adaptação da cor dos mesmos para corresponder com a cor do Barchart. Enquanto a Figura 3.30(b), demonstra a possibilidade da transição gerar múltiplos pontos seguidos em fila, o que visualmente sugere um incremento das barras. Para tornar mais clara essa representação, foram posteriormente adicionadas guias, para os pontos seguirem no momento da transição.

(a) Aumento da área dos pontos

(b) Formação das barras horizontais

Figura 3.30: Etapas da Transição através de Guias entre Scatterchart e Barchart

D: Consumo de pontos

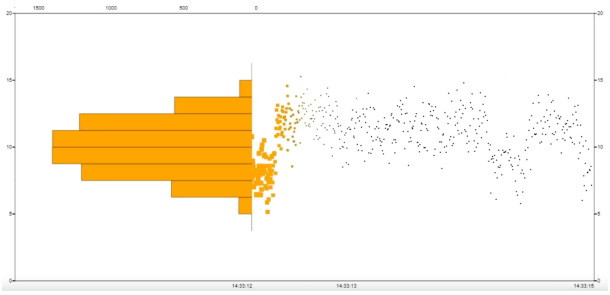


Figura 3.31: Transição de Consumo de pontos entre Scatterchart e Barchart

Com o objetivo de transitar os pontos para o interior das barras, mantendo uma noção da posição de cada ponto no momento que antecede o incremento das mesmas, mas ao mesmo tempo evitar o choque visual da transição direta dos pontos para as barras, como descrito na subsecção 3.3.4. Decidiu-se criar uma transição (Figura 3.31) que provocasse um aumento ligeiro dos pontos, evitando assim que sejam formados grandes núcleos, em que os mesmos ficassem sobrepostos e a sua cor vai sendo adaptada à cor das barras do Barchart, como é possível observar na Figura 3.32a. Na figura 3.32b, verifica-se que os pontos são deslocados para dentro de cada barra, sugerindo um efeito do seu ”consumo”, por parte das mesmas.

(a) Aumento ligeiro dos pontos

(b) Formação das barras horizontais

Figura 3.32: Etapas da Transição de Consumo de pontos entre Scatterchart e Barchart

3.3.5 Transição Scatterchart - Heatmap Acumulador

O Heatmap Acumulador tem como objetivo ser um mero Heatmap com a diferença de possuir um intervalo temporal infinito e, por isso, apenas uma coluna com vários intervalos no eixo vertical, assemelhando-se a barras horizontais que têm o mesmo comportamento do Heatmap. Estas, sofrem alterações após a transição dos pontos, que conforme a sua concentração/quantidade tonificam mais ou menos as barras respetivas ao intervalo a que pretencem. Foi necessário numa primeira abordagem encontrar diferentes formas de transformar um conjunto de pontos em barras com comprimento maior e pré-configurado para dar origem a diferentes transições animadas.

A: Sem animação

Com o objetivo de transformar pontos em barras horizontais, apresenta-se na Figura 3.33 uma transição que não possui qualquer tipo de transição animada, transitando os pontos para a representação da esquerda sem preocupação no que toca às transformações quer a níveis de cor ou de dimensão.

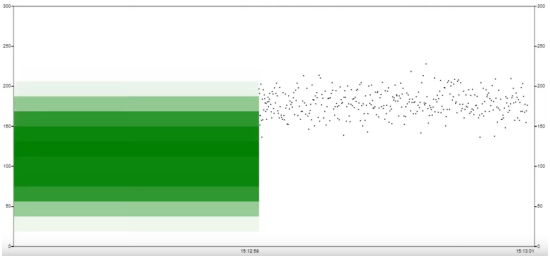


Figura 3.33: Transição não animada entre Scatterchart e Heatmap Acumulador

B: Fade-in Fade-out

A técnica Fade-in Fade-out tem como objetivo desvanecer os pontos enquanto os mesmos transitam para o interior de cada uma das barras horizontais deste Heatmap. Esta transição (Figura 3.34) faz desaparecer os pontos no mesmo instante em que as barras do Heatmap começam a tornar-se mais visíveis, por via da sua cor. Criou-se um desvanecimento gradual dos pontos acentuando assim a ideia de que as barras são formadas pelos pontos que desvanecem no momento da transição para para o seu interior. As próprias barras do Heatmap também têm um gradiente que varia entre transparente e a uma tonalidade de cor verde resultante da quantidade de dados acumulados naquele intervalo.

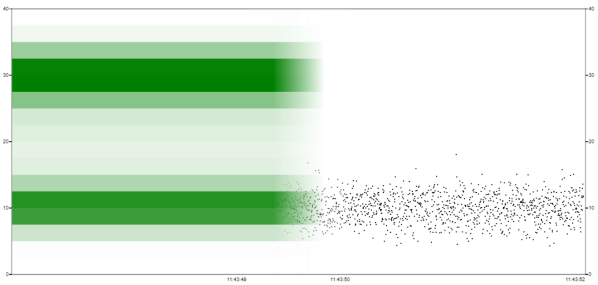


Figura 3.34: Transição Fade-in Fade-out entre Scatterchart e Heatmap Acumulador

C: Encaminhamento de pontos

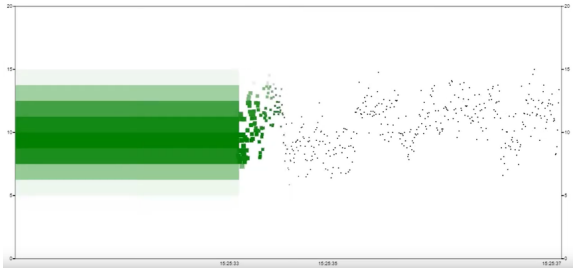
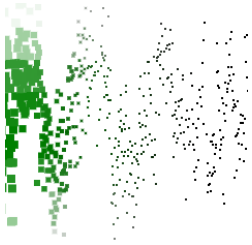
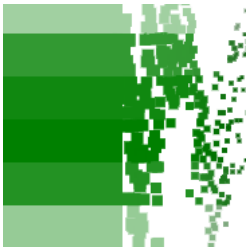


Figura 3.35: Transição de Encaminhamento de pontos entre Scatterchart e Heatmap Acumulador

Com o objetivo de incrementar o tamanho dos pontos, tentando diminuir o choque visual causado pela técnica sem animação que transitava os pontos para as barras do Heatmap, pretende-se com esta transição (Figura 3.35) encaminhar os pontos para dentro de cada barra ao mesmo tempo que vão aumentado o seu tamanho. Para isso, os pontos crescem em direção às barras correspondentes, indo gradualmente alterando a sua cor para a cor coincidente - Figura 3.36(a). Os pontos são deslocados para dentro de cada barra provocando um efeito de ”consumo”por parte das mesmas, como é visível na Figura 3.36(b). É dado um tamanho máximo para cada ponto para que estes não criem núcleos de quadrados muito acentuados e assim mantém-se uma animação suave enquanto os pontos transitam.



(a) Aumento da dimensão dos pontos



(b) Consumo dos pontos

Figura 3.36: Etapas da transição Encaminhamento de pontos entre Scatterchart e Heatmap Acumulador

D: Eletrocardiograma

Esta transição, apresentada na Figura 3.37, foi inicialmente pensada para ligar todos os dados, criando uma espécie de gráfico de linhas que seria depois consumido pelo Heatmap Acumulador. Quando foi

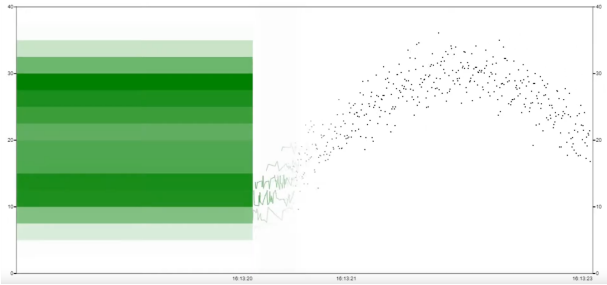
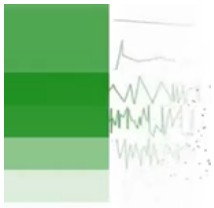


Figura 3.37: Transição Eletrocardiograma entre Scatterchart e Acumulador Heatmap

testada inicialmente, verificou-se que quanto mais pontos e variação existia na dados recebidos, uma única linha teria um resultado pouco percetível tornando a visualização confusa. Optou-se então por manter a ideia, mas dividir os dados nos intervalos do Heatmap, onde por cada intervalo de valores os pontos respetivos se interligam, formando tantas linhas quantas barras existem no Heatmap, como se pode observar na Figura 3.38(a). Desta forma, surgiu também o nome pela sua semelhança a vários eletrocardiogramas, pois passa ser possível visualizar, para cada intervalo, os vários picos e variações existentes nos seus valores, antes dos mesmos coincidirem com as barras do Heatmap (Figura 3.38(b)).

(a) Conversão de pontos para linhas



(b) Formação das barras horizontais

Figura 3.38: Etapas da Transição Eletrocardiograma entre Scatterchart e Acumulador Heatmap

E: Dilatação

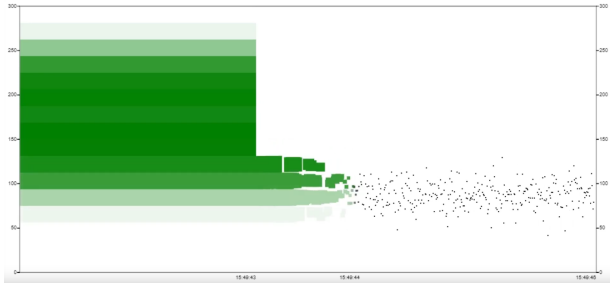


Figura 3.39: Transição de Dilatação entre Scatterchart e Acumulador Heatmap

Tendo como objetivo aumentar os pontos, para fazê-los coincidir com dimensão igual à altura de cada barra do Heatmap, esta transição (Figura 3.39) pretende dilatar cada ponto recebido e ao mesmo tempo ir alterando a sua cor para a fazer corresponder com a cor do intervalo ao qual pertence no Heatmap, como apresentado na Figura 3.40(a).

(a) Dilatação dos pontos

(b) Formação das barras horizontais

Figura 3.40: Etapas da Transição de Dilatação entre Scatterchart e Acumulador Heatmap

Quando existem muitos pontos seguidos, pertencentes ao mesmo intervalo, é projetada a ideia de uma barra, logo na transição. Inversamente, quando existem poucos dados, é apenas apresentado o ponto com dimensão crescente até atingir tamanho igual ao da altura da barra.

3.4 Sumário

Neste capítulo foi explicado o conceito VisMillion, o qual permitiu iniciar o desenvolvimento deste trabalho. Foram ainda apresentadas as várias técnicas de visualização utilizadas para representar grandes quantidades de informação, na Secção 3.2. Tendo depois sido exposto o estudo realizado, das múltiplas técnicas de transição, na Secção 3.3, onde foram explicadas as diferentes abordagens para transformar as representações gráficas dos dados do Scatterchart para uma das outras técnicas de visualização anteriores, tendo em consideração as medidas estatísticas e os tipos de agregação que cada uma delas permite representar.

Capítulo 4

Protótipo

De forma a facultar o estudo de transições suaves que permitem seguir a informação conforme esta transita ao longo das diferentes técnicas de visualização e ainda, para que seja possível representar grandes quantidades de dados, tendo em consideração o seu domínio e a importância da redução do seu volume, foi desenvolvido um protótipo cuja abordagem parte do conceito VisMillion [10].

Neste capítulo é explicado o protótipo desenvolvido, considerando não só os requisitos que foram analisados, como também, as decisões de implementação e desenvolvimento que permitiram realizar o estudo das transições entre diferentes técnicas de visualização.

4.1 Arquitetura

A arquitetura utilizada para o protótipo (Figura 4.1) passou por utilizar um **Gerador de fluxos de informação** para criar e enviar pacotes de dados para o sistema, simulando uma fonte de informação que distribui dados em tempo real para o seu recetor. Na base do Vismillion and Change existe uma entidade **Gestor**, responsável por manter o sistema atualizado e invocar tarefas para um conjunto de **Módulos** que lhe pertencem, transmitindo-lhes ainda, os fluxos de dados que foi recebidos depois de enviados pelo gerador de fluxos de informação. Cada um desses módulos possui então métodos que

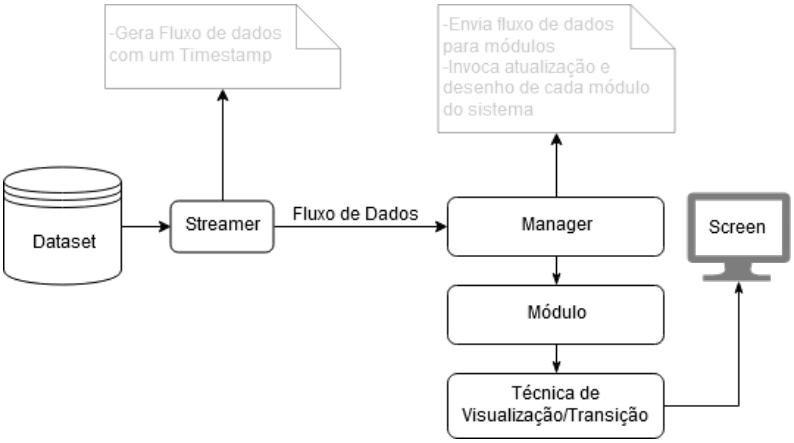


Figura 4.1: Diagrama da arquitetura do protótipo

quando invocados permitem não só atualizar e controlar os domínios dos mesmos, como ainda enviar os dados que compõem esses domínios para a **Técnica de Visualização** ou **Técnica de Transição** à qual está associado, assim como, instruções para que a mesma desenhe na visualização.

Relativamente à estrutura do sistema, o Gestor é composto por um ou mais módulos. A utilização de técnicas que realizam uma degradação graciosa da informação, em que, conforme a visualização vai recebendo os dados mais recentes, os mais antigos vão sendo agregados para reduzir o volume da informação, implicou a aplicação de uma estratégia modular. Isto é, cada técnica de visualização que se pretende utilizar para analisar os dados é atríbuida a um módulo, fornencendo-lhe independência suficiente para o seu funcionamento de uma forma individual, mas ao mesmo tempo, permitindo a interligação de diversos módulos ao longo de um eixo horizontal, que corresponde ao tempo, através de módulos que estão associados a técnicas de transição. Esta estrutura está representada no UML da Figura 4.2. Assim, permitiu-se a criação de uma visão contínua, de múltiplas técnicas de visualização e ao mesmo tempo, permitiu-se a representação da informação com diferentes métodos estatísticos, técnicas de agregação e níveis de detalhe.

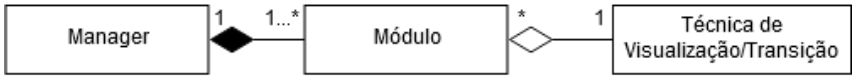


Figura 4.2: UML do protótipo

Com o objetivo de se estudarem as transições entre visualizações, recorrendo a esta mesma estratégia modular, pensou-se na utilização de um módulo extra que se encontra no meio dos dois anteriores. É nele que se concentram todas as técnicas das transições nas quais este trabalho se foca. Na Figura 4.3, encontra-se a estrutura referente a esta estratégia.

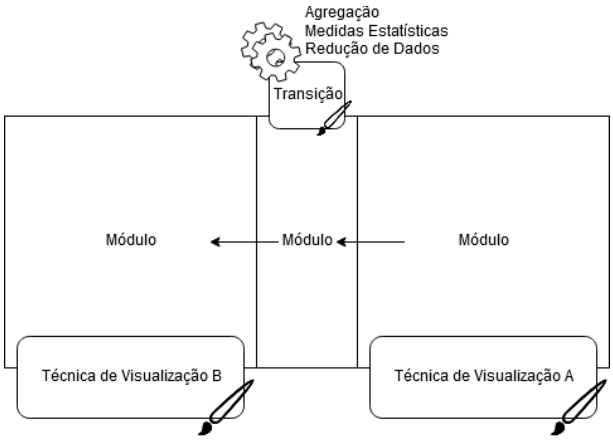


Figura 4.3: Diagrama da estrutura do sistema

As técnicas de agregação e aplicação dos métodos estatísticos realizam-se nas próprias transições, fornencendo-lhes assim toda a responsabilidade para a redução de dados e abstraindo as técnicas de visualização de todas as operações que não se relacionem com o desenho da informação. Por outro lado, cada técnica desenvolvida para transitar a informação entre módulos ficou encarregue do desenho na visualização, da aplicação dos métodos que permitem reduzir o volume de informação, e ainda das

correções da velocidade, de modo a fazer corresponder as velocidades dos módulos.

Esta arquitetura permite que seja realizada uma análise módulo a módulo e desta forma, consegue obter-se uma ideia mais concisa sobre o passado dos dados através da visualização onde estes estão mais agregados, possibilitando a análise dos seus padrões. Ao mesmo tempo, permite comparar os dados que estão a ser obtidos em tempo real com os dados mais antigos da visualização. O que é mais valia para se verificar se uma tendência se mantém ou não constante, ao longo do tempo.

4.2 Interface

O ecrã inicial do protótipo contém um menu - Figura 4.4, onde é possível selecionar uma técnica de visualização para analisar a informação. O Scatterchart é a técnica escolhida por defeito para o módulo mais à direita, onde os dados obtidos em tempo real são representados numa fase inicial. Tendo sido escolhida a técnica pretendida, é ainda fornecida a possibilidade de modificar o intervalo de tempo que se pretende para cada módulo e desta forma fazer variar a velocidade com que os dados de cada um se vão movimentar.

Figura 4.4: Menu inicial do protótipo VisMillion and Change

Sendo o foco principal desde trabalho o estudo das melhores formas para transitar informação de um módulo, cujo intervalo temporal é muito menor do que o outro para onde se destina essa informação (dado que se pretende uma visualização que registe um histórico cada vez maior), as transições realizadas para cada tipo de visualização foram preparadas para os intervalos de tempo que estão predefinidos nesse menu. No entanto, é possível modificar os seus valores.

Após a seleção da técnica desejada e do intervalo de tempo para cada módulo, a interface irá carregar o conjunto selecionado, apresentando os 3 módulos referidos anteriormente. Estes irão surgir alinhados horizontalmente, com os dados a transitar da direita para a esquerda. Resultando num módulo com a técnica de visualização Scatterchart, outro com uma técnica de transição e por fim,

à esquerda, um com a técnica selecionada pelo utilizador no menu anterior.

O protótipo ficou preparado para que se modifique a transição em tempo de execução, através de um menu de seleção (Figura 4.5), permitindo assim alternar entre as várias opções de transição que foram estudadas para cada par de técnicas de visualização. Da mesma forma, ficou também preparado para as modificações das técnicas de visualização escolhidas (módulo da esquerda) e assim verificar as restantes transições desenvolvidas.

(a) Troca de transição

(b) Troca de técnica de visualização

Figura 4.5: Alterações em tempo de execução

4.3 Implementação

Seguindo uma abordagem de implementação que permitisse, de um modo geral, manter a compatibilidade entre sistemas operativos e ao mesmo tempo, sempre disponível, necessitando apenas de conexão à internet, optou-se por seguir uma solução *browser-based* (baseada no navegador), permitindo assim analisar o protótipo desenvolvido através da utilização de um navegador.

A implementação deste protótipo baseou-se no modelo desenvolvido no sistema VisMillion [10]. Este modelo define-se por uma área com dimensões configuráveis num documento HTML, onde através de métodos específicos para cada técnica desejada, se pode desenhar elementos no ecrã, mais especificamente dentro da área do Canvas[1]. Este modelo apesar de permitir inúmeras alterações em relação a figuras, cores, tamanhos, texto, posições, etc., não contém transições na sua estrutura, sendo por isso necessário elaborar todas as transições desejadas através dos tempos de execução, desacelerações, modificações nas cores e tamanhos dos elementos. Ao mesmo tempo que se torna uma tarefa complexa, oferece uma maior liberdade para desenvolver animações, conforme pretendido.

Como referido na arquitetura, Secção 4.1, o sistema parte de um conjunto de entidades que permitem representar a informação no ecrã através de elementos gráficos. Essa informação é recebida via múltiplos fluxos de dados que são emitidos por um Gerador de fluxos de informação. Existe depois um Gestor que ao receber a informação, vai emiti-la para o módulo mais à direita, e por fim, o módulo transmite-a para a técnica de visualização à qual está associado. Transferindo os dados que se encontram fora do seu domínio, para o módulo seguinte (à sua esquerda).

Nesta secção, será explicada a implementação de cada uma destas entidades, seguindo a ordem do diagrama apresentado na Figura 4.1, que representa o ciclo da informação desde a origem dos fluxos de dados até à sua representação na visualização.

[1] https://www.w3schools.com/html/html5_canvas.asp

4.3.1 Gerador de fluxos de informação

O sistema possui um Gerador de fluxos de informação que é responsável por simular um servidor que envia dados de forma contínua para serem representados na visualização, em tempo real. O Gerador de fluxos de informação é uma entidade baseada no conceito VisMillion [10] que foi implementada com a linguagem Python para enviar pacotes de dados para a interface através de *WebSockets*. Ao utilizar esta tecnologia, é criada uma ligação bidireccional que permite uma troca de informação de um modo simples e persistente entre o cliente e o servidor, operando entre um único canal de comunicação. Segundo o estudo realizado no VisMillion, em que são postas em comparação as latências médias entre vários protocolos de comunicação, servidor-cliente, concluiu-se que a latência média dos *WebSockets* tem os valores mais baixos, visto que a quantidade de pedidos realizados para enviar o mesmo conjunto de informação é também ela menor em comparação com os restantes protocolos. Uma vez que os navegadores com as versões mais recentes suportam esta API de *WebSockets*[2], optou-se por aplicar esse protocolo de comunicação para a conexão entre o Gerador de fluxos de informação e o Gestor. Perante a leitura de ficheiros .csv, cujo conteúdo pretende ser conjuntos de dados (*datasets*) de enormes dimensões, são gerados fluxos de dados que são depois enviados para o protótipo.

Uma vez que o foco deste trabalho está na parte da representação da informação, o Gerador de fluxos de informação não é responsável por nenhum tipo de simplificação ou redução de dados. No entanto, como alguns *datasets* possuem *timestamps* (registos de data/hora) que são demasiado antigos para serem representados pelo sistema, pois este está desenvolvido para obter dados em tempo real, foi necessário modificar o registo de data/hora de cada um dos dados (Figura 4.6), de forma a tornar possível a realização deste trabalho. O Gerador de fluxos de informação trata então de modificar ordenadamente as entradas do *dataset*, transformando o *timestamp* do primeiro dado no *timestamp* corrente e calculando a sua diferença. Assim, os restantes dados irão somar esse valor ao seu *timestamp* original, para o fazer corresponder de forma cronológica com o tempo do primeiro *timestamp* registado. Desta forma, torna-se possível visualizar a informação, independentemente da mesma "não pertencer"ao domínio temporal, que tinha sido atribuído inicialmente aos módulos.

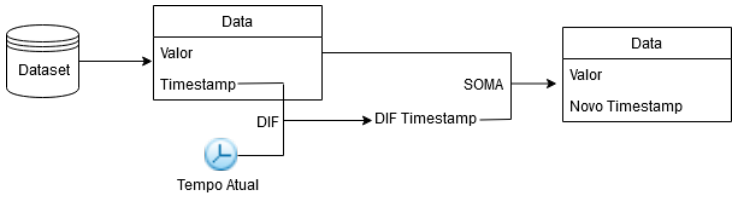


Figura 4.6: Modificação do Timestamp no Gerador de fluxos de informação

Além da comunicação existente entre o servidor e o cliente, onde os pacotes de informação são enviados para o cliente, também este pode comunicar com o servidor através de ações provocadas pelo utilizador, em configurações que estão disponíveis na interface do sistema. Como referido anteriormente, cada linha do *dataset* é enviada para o cliente, não respeitando o seu *timestamp* original, mas sim o gerado a partir da primeira linha. O utilizador poderá modificar através da interface, o intervalo de

[2]https://developer.mozilla.org/en-US/docs/Web/API/WebSockets_API

tempo com que cada uma dessas linha é enviada, provocando um atraso maior ou menor no envio das mesmas. Quanto menor esse valor, mais rapidamente será enviada a linha seguinte, para ser posteriormente renderizado o seu valor na visualização. O utilizador dispõe ainda de comandos que permitem pausar/continuar e recomeçar a transmissão de dados.

4.3.2 Gestor

O Gestor é a entidade principal, visto que é a responsável por gerir todos os módulos do sistema, sendo também através dele que se definem os domínios para os valores dos dados que se pretendem visualizar. Toda a informação que é gerada pelo Gerador de fluxos de informação e enviada através de fluxos de dados, tem como recetor, o Gestor. Sendo por ele que os fluxos de dados passam, antes de serem transferidos para o módulo da direita, mas também é o Gestor que possibilita a posterior transferência dos dados, desse módulo para os módulos seguintes, quando esses deixam de pertencer ao seu domínio temporal.

A sua implementação está preparada para a adição e remoção de mais módulos em tempo de execução, possuíndo para isso métodos que recalculam as dimensões de cada um dos módulos do sistema, dependendo da sua quantidade, de forma a mantê-los sempre com a mesma largura, à excepção dos módulos onde vão decorrer as transições, que possuem uma dimensão configurável apenas antes de iniciar o protótipo. Existem portanto, métodos que permitem adicionar ou remover módulos e estes estão adaptados para a configuração inicial antes de executar o sistema, mas estão preparados para que o seu funcionamento se mantenha mesmo em tempo de execução.

É então, através do Gestor que se unifica os vários módulos, sendo ele também o responsável pela sua atualização. Uma desvantagem do *Canvas* é a necessidade de redesenho de todos os elementos quando se pretende modificar alguma coisa na visualização, visto que cada elemento é ”pintado”no ecrã e num modelo normal, não representa nenhum objeto, não sendo também possível aceder aos seus atributos. Então, por cada iteração em que exista uma atualização que force o sistema a redesenhar os elementos no Canvas, torna-se necessário realizar uma limpeza do mesmo, para de seguida invocar os métodos de *update* e *draw* de cada módulo. No início de cada ciclo/iteração, é criado um *timestamp* do tempo corrente, e o mesmo é enviado para cada um dos módulos no momento em que se invoca o seu método de *update*. Assim, cada um dos módulos irá realizar as operações necessárias, tendo em consideração as atualizações no seu intervalo de tempo. Calculando ainda os **FPS** (quadros por segundo), que serão sempre afetados pela quantidade de dados a renderizar e ainda pela complexidade das operações a realizar.

Finalmente, o Gestor contém as configurações que permitem especificar o *Canvas* que será utilizado, assim como as suas dimensões e a cor do seu fundo. Permite ainda, determinar o intervalo de valores que pertencem ao domínio da visualização e a escala correspondente, que através de ferramentas da biblioteca **D3**, permitem mapear os dados no ecrã, consoante as diferentes opções disponibilizadas pela mesma. A escala utilizada para o mapeamento horizontal (associado ao tempo) está pré-configurada para uma escala temporal, *scaleTime*.

4.3.3 Módulo

Numa primeira fase e seguindo a lógica adotada pelo sistema VisMillion [10], o protótipo era constituído por múltiplos módulos, todos eles distintos, onde cada um era responsável pela atualização e renderização dos dados, através de uma determinada técnica de visualização. No entanto, essa abordagem inicial não se demonstrou muito eficiente e provocava muita repetição em termos de programação, não permitindo desenvolver o estado de abstração dos módulos, visto que estes estavam diretamente ligados a uma técnica de visualização. Então, optou-se por separar os conceitos, separando também a parte lógica da visual, até então unificada e aplicou-se a estratégia modular referida na arquitetura, Secção 4.1, passando a existir módulos passivos a configurações específicas, mas que partem todos de uma mesma entidade - Módulo.

O que diferencia um módulo de um módulo de transição, é a sua inicialização. No momento de criação do módulo, existe uma variável que permite especificar o tipo de módulo pretendido. É também nesse momento que se configura a largura do módulo (variável consoante é ou não um módulo de transição) e o intervalo de tempo que se pretende para representar a informação. É ainda registado o seu índice, que permite a sua identificação perante a lista de módulos armazenada pelo Gestor. A cada módulo é possível atribuir uma técnica de visualização, não sendo esta dependente do mesmo, mas sim dos dados que lhe são transmitidos. Desta forma, garantindo a abstração mencionada anteriormente, simplificando e deixando ainda o sistema preparado para a modificação da técnica de visualização que lhe fora atribuída em tempo de execução, sem ser necessário reiniciar o sistema.

Ao atribuir uma técnica de visualização ao módulo, são emitidas para essa mesma, todas as características necessárias para o seu funcionamento, como é o caso da dimensão, intervalo temporal e a posição que se encontra no eixo horizontal. Cada módulo calcula também o intervalo de tempo para o qual a visualização correspondente precisa de ser atualizada, para que apenas se realizem atualizações quando realmente existirem modificações na mesma, aumentando desta forma o desempenho do sistema. Este cálculo é feito através da divisão entre o intervalo temporal do módulo e a sua largura, pois irá retornar o intervalo de tempo necessário para que exista deslocamento de 1 pixel na visualização (o valor mínimo de deslocamento). Da mesma forma, a limpeza da área do *Canvas* correspondente ao módulo e o seu redesenho só são invocados quando houver realmente necessidade.

Cada módulo possui um *endTime* e um *startTime* que são calculados através das Equações 4.1a e 4.1b, com *deltaRange* e *modulesLength* a corresponder ao intervalo de tempo do módulo e à quantidade de módulos existentes, respetivamente.

$$endTime = timestamp - \sum_{i=index}^{modulesLength} module_i.deltaRange \tag{4.1a}$$

$$startTime = endTime - deltaRange \tag{4.1b}$$

O *endTime* corresponde ao tempo mais recente e o *startTime* ao tempo mais antigo de cada módulo. A Figura 4.7 demonstra a relação entre as duas variáveis.

A transmissão de dados é feita entre módulos, através da invocação do método de transferência

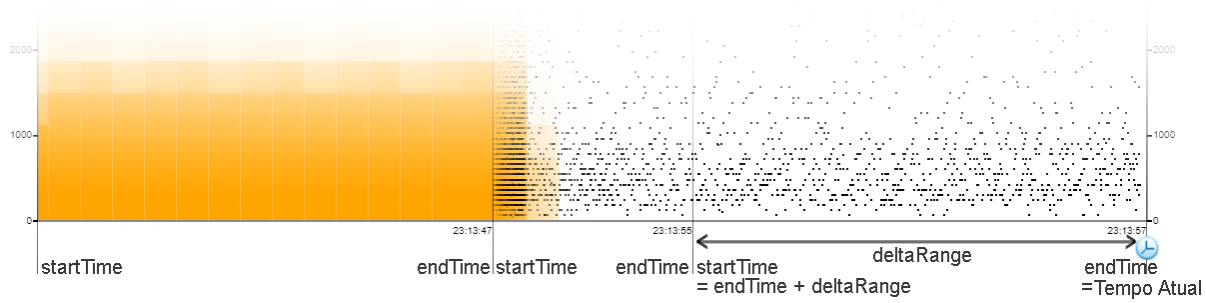


Figura 4.7: Relação entre *startTime* e *endTime*

de informação existente no Gestor. Existindo também um método em cada módulo que lhe permite associar os dados que foram recebidos a uma lista, que é guardada por cada um. No entanto, para iniciar a receção da informação, o primeiro módulo (mais à direita) recebe os dados originais enviados pelo Gestor, sem qualquer agregação e consoante os mesmos se vão tornando mais antigos dentro do módulo, é feita uma verificação dos *timestamps* correspondentes, filtrando os dados que ainda pertencem ao módulo, isto é, se o seu *timestamp* ainda se compreender entre os seus *endTime* e *startTime*. Quando um dado se encontrar fora desse intervalo, será enviado para o módulo seguinte (à sua esquerda), utilizando o método do Gestor de transferência de informação, que se irá responsabilizar pelo envio dos respetivos dados. Ao mesmo tempo, serão removidos da sua lista todos aqueles que se encontrarem fora do seu intervalo temporal, para evitar que persistam dados repetidos ao longo dos vários módulos e por conseguinte haja um decréscimo do desempenho do sistema devido ao excesso de dados armazenados na memória do navegador.

Perante os módulos transicionais, há um método que permite atualizar a velocidade da transição, possibilitando o aumento da duração do mesmo, através da soma de um intervalo temporal inicial (associado à velocidade inicial), o intervalo temporal da desaceleração pretendida e ainda um intervalo temporal final (associado à velocidade final). O funcionamento deste método será explicado detalhadamente na Subsecção 4.3.5, uma vez que o Módulo apenas tem a função de concretizar a definição do intervalo temporal inicial e final, que é feito consoante as configurações iniciais.

4.3.4 Técnica de Visualização

Cada módulo existente no sistema necessita de uma técnica de visualização associada para que seja possível representar os dados recebidos no ecrã. As técnicas de visualização implementadas neste trabalho são: Scatterchart, Heatmaps (normal e acumulador), Linechart, Streamgraph e Barchart. Estas técnicas estão descritas com maior detalhe na Secção 3.2.

Quando se cria uma nova instância de uma técnica de visualização, utiliza-se um método que permite consoante a técnica desejada, selecionar um conjunto de predefinições necessárias para a criação de cada um. O Scatterchart, por exemplo, necessita de uma configuração que permita definir o tamanho de cada ponto. Já o Heatmap, requer uma dimensão para os quadrados correspondentes a cada célula

da sua matriz. As cores de cada composição de elementos da técnica de visualização são também aqui configuradas, de forma a criar uma visualização personalizada, conforme as preferências do utilizador.

Ao atribuir uma técnica de visualização a um módulo, esta vai receber configurações adicionais e ainda os dados provenientes desse módulo para que possa começar a representar a informação pertencente ao seu intervalo temporal. Além desse intervalo, recebe ainda a largura e posicionamento em pixels no eixo horizontal de cada um. Como mencionado na Subsecção 4.3.3 referente à entidade Módulo, quando o módulo invoca o método de atualização e desenho, o mesmo transmite para a técnica de visualização associada o *timestamp*, que já tinha sido transmitido da invocação do método de atualização do Gestor. Criando assim um encadeamento do instante temporal em que se realizou a chamada para atualização no Gestor, até alcançar cada uma das técnicas de visualização aplicadas.

Quando é invocado o método de desenho de uma técnica de visualização, esta, numa primeira fase, limpa todo o espaço que lhe é atribuído no Canvas, prosseguindo depois com o desenho dos elementos que representam a informação em si e com o redesenho dos seus eixos (para evitar que partes dos mesmos tenham sido sobrepostos com resíduos das animações realizadas).

No caso do desenho do Scatterchart, que na versão desenvolvida no VisMillion renderizava os pontos através de círculos que eram desenhados com a função *arc* do D3, este passou a representar os pontos na visualização através de quadrados com 2 pixels, já que visualmente não se denota diferenças significativas, mas em termos de otimização do sistema, passou a poupar-se recursos de processamento e cálculo para o seu desenho.

Como o eixo horizontal corresponde a uma escala temporal, e o tempo vai passando de forma a manter sempre a relação com o tempo real, todos os elementos desenhados na visualização vão também eles evoluindo, e por isso movimentando-se para a esquerda para corresponderem sempre com o seu tempo original. Essa movimentação, foi estudada de diversas formas com o objetivo de arranjar uma fórmula que permitisse a melhor otimização para o sistema.

Numa fase inicial, utilizou-se a API *d3-scale*[3], que permitia deslocar os dados (representados) através da identificação das extremidades da visualização (*leftPosition* e *rightPosition*) e do seu intervalo de tempo (*startTime*, *endTime*), desta forma relacionando a posição dos dados com a evolução do tempo. A sua implementação, em parte, seria algo como a seguinte:

```
position = d3.scaleLinear()
            .range([leftPosition, rightPosition])
            .domain([startTime, endTime]);
```

A abordagem anterior, apesar de apresentar um funcionamento correto, demonstrou algumas quebras de desempenho perante a análise de quantidades elevadas de informação, pois a mesma necessita diversas comparações entre as extremidades da visualização e o instante temporal antes de devolver um resultado. Foi por isso, abordada a Fórmula do Movimento Retilíneo Uniforme[4], apresentada na Equação 4.2a, que traduzida para o modelo do sistema necessita do registo do tempo decorrido, ou seja, a diferença (*deltaTime*) entre o *timestamp* no momento de invocação do método e o instante

[3] https://github.com/d3/d3-scale

[4] Physics for Scientists and Engineers with Modern Physics, 9th Edition - Raymond A. Serway, John W. Jewett

em que a visualização recebeu o primeiro dado. Tomando o *deltaRange* como o intervalo de tempo da visualização e o *ownWidth* como a sua largura, a Equação 4.2b permite entender a transformação necessária para a implementação do movimento no sistema.

$$x = x_0 + v_i * t \tag{4.2a}$$

$$x = x_0 + \frac{ownWidth}{deltaRange} * deltaTime \tag{4.2b}$$

Cada técnica de visualização é responsável por desenhar todo o conteúdo necessário para a análise da mesma. No entanto, dada a forma como cada elemento gráfico é renderizado através da tecnologia *Canvas* nas técnicas de visualização cuja representação se divide em múltiplos intervalos temporais e existe agregação nos seus dados (Heatmap, Linechart e Streamgraph), é necessário manter o último intervalo de dados agregados para que não sejam perdidas partes da visualização antes de todo esse intervalo ter sido desenhado e dispensado do módulo. Essa exigência levou à criação de uma lista de dados secundária e auxiliar, que é substituida em cada nova iteração pela último intervalo de dados agregados da lista original. Para estas técnicas de visualização e perante o momento de desenho da mesma, é necessário invocar os métodos respetivos não só para a lista original, mas também para a auxiliar referida anteriormente.

As técnicas de visualização que permitem acumular informação - Barchart e Heatmap Acumulador, possuem uma particularidade relativamente ao eixo horizontal, visto que este eixo não representa o tempo, mas sim a quantidade de dados que foi acumulada por cada intervalo de valores, especificamente para o caso do Barchart. Ou chegam mesmo a não ter qualquer relação com o eixo horizontal, como é o caso do Heatmap Acumulador. No caso do Barchart, como a quantidade de dados pertencente a um intervalo pode provocar um grande aumento da barra respetiva, podendo esta exceder a sua escala horizontal, definiu-se um reajustamento do domínio horizontal da técnica em cada iteração da atualização, fazendo incrementar o seu valor atual com um valor percentual configurável.

4.3.5 Técnica de Transição

Quando existem 2 ou mais módulos no sistema para representar a informação com múltiplas técnicas de visualização, são aplicados módulos transicionais entre os anteriores, como referido na Subsecção 4.3.3. A estes módulos associam-se técnicas de transição, de forma a criar animações que permitam seguir a informação enquanto esta transita de uma técnica de visualização para outra.

Estas técnicas são responsáveis pela agregação da informação que é obtida pelo módulo do Scatterchart, onde os dados correspondem aos dados originais, com o intuito de transitá-los para a técnica de visualização seguinte onde se pretende que a informação seja visualizada de forma agregada. Desta forma, permitindo representá-la através de intervalos que demonstrem a concentração / densidade, ou medidas estatísticas como a média, mediana, intervalo interquartil, valor máximo e mínimo, etc.

A lógica para a criação dos intervalos está presente nestas técnicas, não tendo sido separada da parte visual, visto que estas são técnicas com métodos bastante exclusivos para o efeito que se

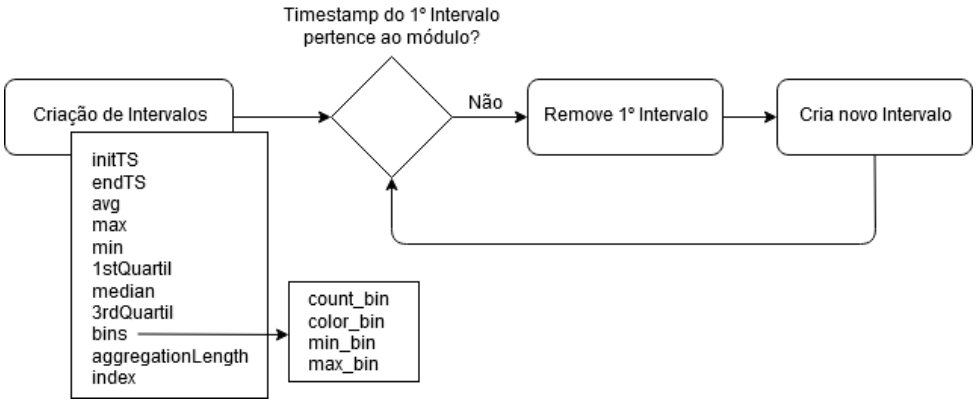


Figura 4.8: Lógica para criação de Intervalos

pretende. A Figura 4.8 pretende ilustrar esta lógica, sendo necessário numa primeira etapa criar os intervalos. A quantidade de intervalos que existem na transição depende de vários fatores. O primeiro é o tamanho da agregação que se pretende realizar, sendo para isso necessário pré-configurar uma dimensão em pixels que irá dividir a largura desta visualização em múltiplos intervalos de tamanho igual ao valor dessa dimensão. Outro fator, é a largura da visualização. Esta, não se pode utilizar diretamente quando os módulos à sua esquerda e direita possuem intervalos de tempo diferentes, pois iria causar sobreposição de alguns intervalos criados pela transição. É, por isso, necessário criar uma largura de visualização auxiliar que é calculada através da fórmula da velocidade média $v = \Delta x/\Delta t$, onde utilizando a velocidade do módulo da esquerda e a duração da transição, se obtém um Δx que corresponde a essa largura. Assim, o número de intervalos em que se pode dividir a transição é igual a esse valor a dividir pela dimensão em pixels fornecida para realizar a agregação.

Cada intervalo contém um *initTS* e *endTS* que simbolizam o começo e término em relação à escala temporal. Contém também um conjunto de *bins* (Figura 4.9) que são utilizados para transitar os dados do Scatterchart para um Heatmap, possuíndo um contador que juntamente com uma escala de cores permite intensificar a tonalidade de cada célula da matriz do Heatmap. Os restantes atributos de cada intervalo servem para o cálculo das medidas estatísticas já mencionadas. Quando o intervalo mais antigo do conjunto possuir um *initTS* que já não pertence ao intervalo temporal da transição, este é transferido para o módulo seguinte e removido. O que implica a criação de um novo intervalo com uma nova relação temporal, que irá para a posição dos intervalos mais recentes do conjunto.

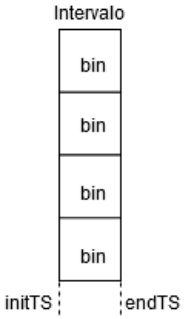


Figura 4.9: Conjunto de *Bins* de um Intervalo

No final de cada iteração da atualização da transição, é então, realizada a agregação (Figura 4.10). Os *timestamps* dos dados provenientes do módulo anterior são verificados para determinar o conjunto de dados que pertence a cada intervalo existente. Para cada intervalo são calculadas as medidas estatísticas daquele conjunto de dados através de métodos do D3, encontrando os seus valores e associando-os aos mesmos. Supondo que um dado obtido pertence ao último intervalo, este irá percorrer o conjunto de intervalos até encontrar aquele cujos *initTS* e *endTS* permitem envolver o seu *timestamp*. Assim, esse dado irá, juntamente com os restantes que pertencem ao intervalo, definir a média, valor máximo e mínimo, primeiro e terceiro quartil e a mediana. Se se tratar de uma transição para um Heatmap, então será também detetado o *bin* a que o dado pertence, para incrementar o contador de dados do mesmo e atualizar a cor correspondente através de uma interpolação de cores com uma escala linear da API *d3-scale*. Se esse dado corresponder ao valor máximo ou mínimo do *bin* em questão, então o seu valor será utilizado para redefinir esses atributos.

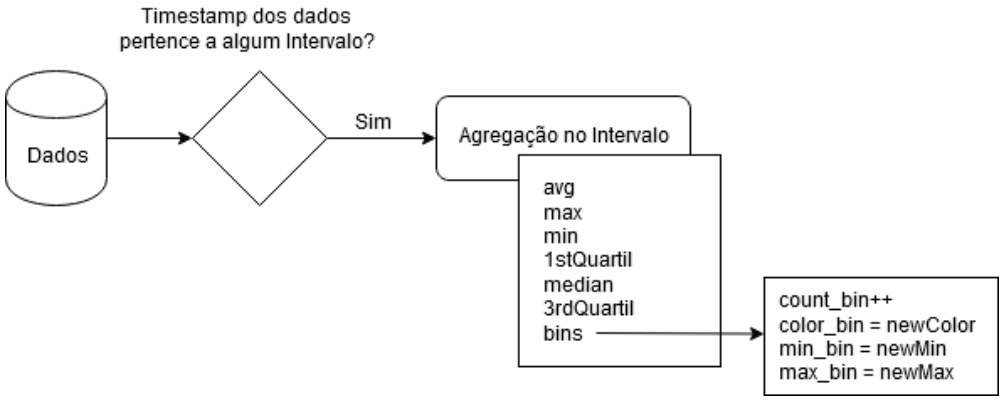


Figura 4.10: Lógica para Agregação em Intervalos

Relativamente ao desenho das transições animadas, foram estudadas diversas técnicas que permitem transitar pontos para linhas, áreas, barras, etc. conforme analisadas na Secção 3.3. Tal como nas técnicas de visualização, existem múltiplos métodos que são invocados para representar os elementos gráficos conforme pretendido. Por exemplo, numa transição entre um Scatterchart e um Linechart, existem diversas técnicas, no entanto, como a ideia é criar uma linha, então será necessário invocar, de entre os vários métodos existentes, aquele que permite desenhar uma linha. Mas como não queremos que a linha ocupe toda a transição, aplicam-se gradientes, áreas transparentes ou coloridas disponibilizadas pela tecnologia *Canvas*, para ocultar as zonas da transição que não se pretende que tenham a linha visível, neste caso. Depois resta animar os pontos, fornecendo-lhes variações na aceleração e movimento, mudanças de cor ou tamanho, entre outros.

A movimentação da representação gráfica dos dados na transição é dada através de uma fórmula que calcula a posição dos pontos, conforme a sua relação com o tempo. Da mesma forma que nas técnicas de visualização existe uma fórmula para calcular a posição relativa a movimentos retilíneos uniformes, nestas, existe a necessidade de calcular a posição com aceleração, já que se pretende fazer variar a velocidade, para fazer corresponder a velocidade dos pontos que chegam à transição com a velocidade da representação dos dados no módulo para onde os dados vão transitar. Inicialmente,

desenvolveu-se esta ideia utilizando a fórmula proveniente dos Movimentos Uniformemente Variados, apresentada na Equação 4.3a, que traduzida para o modelo do sistema, necessita do registo do tempo decorrido, ou seja, a diferença (*deltaTime*) entre o *timestamp* no momento de invocação do método e o instante em que a visualização recebeu o primeiro dado. Tomando a $veli$, como a velocidade do módulo anterior e a *acel*, como a aceleração, a Equação 4.3b permite entender a transformação necessária para a implementação do movimento no sistema, nesta primeira abordagem.

$$x = x_0 + v_i * t + \frac{1}{2} * at^2 \tag{4.3a}$$

$$x = x_0 + veli * deltaTime + \frac{1}{2} * acel * deltaTime^2 \tag{4.3b}$$

A aceleração, referida anteriormente, não poderia ser calculada através da fórmula $a = \frac{\Delta v}{\Delta t}$, pois a duração da transição não possui um valor predefinido. No entanto e uma vez que o tempo necessário para provocar a desaceleração é desconhecido, partindo da Equação 4.3b e substituindo a sua aceleração pela fórmula já referida: $a = \frac{\Delta v}{\Delta t}$, obtém-se a Equação 4.4a, que uma vez em ordem ao tempo (Equaçao 4.4b), nos fornece os dados necessários para prosseguir com o cálculo da aceleração.

$$position = v_i * t + \frac{(v_f - v_i) * t}{2} \tag{4.4a}$$

$$t = \frac{2 * position}{v_i + v_f} \tag{4.4b}$$

Dada a complexidade das transições que se pretendia realizar, a utilização da Equação 4.3b para calcular a posição dos dados, apesar de funcionar corretamente, só permitia que a velocidade do módulo seguinte fosse atingida no final da transição, o que acabou por ser uma desvantagem para transições em que se pretendia obter essa velocidade mais cedo. Como solução, pensou-se em aplicar um intervalo antes e depois do intervalo temporal em que iria ocorrer a desaceleração dada pela equação anterior. Assim, seria possível ter um intervalo inicial em que a velocidade era igual à velocidade inicial, depois um momento de desaceleração e novamente um intervalo final com a velocidade igual à velocidade final. A Figura 4.11 permite relacionar os 3 intervalos anteriores (intervalo inicial, intervalo de aceleração/desaceleração e intervalo final) com a velocidade pretendida.

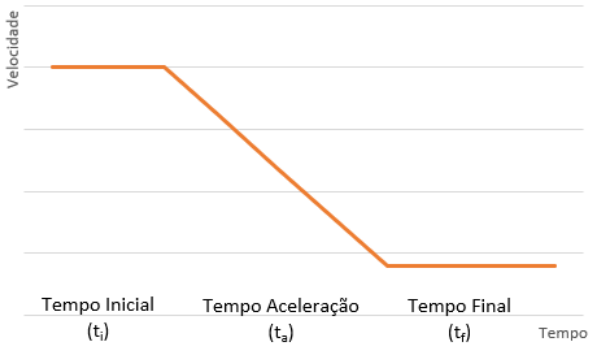


Figura 4.11: Relação Velocidade - Tempo, tendo em consideração os intervalos temporais para o movimento nas transições

Desta forma, a duração do módulo de transição passou a ser igual à soma dos 3 intervalos e a posição dos dados (relativamente ao eixo horizontal). Além de ser dada pelo cálculo da posição definida anteriormente com a Equação4.3b, dos Movimentos Uniformemente Variados (pos_a), passa a somar uma posição relativa a um intervalo inicial e final, como se pode verificar na Equação 4.5. t_f e t_i correspondem ao tempo decorrido no intervalo inicial e final e são calculados através da verificação do mínimo: $t_i = min(deltaTime, t_{i0})$, $t_f = min(deltaTime - t_i - t_a, t_{f0})$, com t_{i0} e t_{f0} a corresponder ao intervalo inicial e final, predefinidos no Módulo, ao qual a técnica está associada.

$$position = v_i * t_i + pos_a + v_f * t_f \tag{4.5}$$

4.4 Sumário

Neste capítulo foi explicado o protótipo desenvolvido, tendo em consideração a necessidade de visualizações que permitam explorar as tendências e padrões existentes na informação, quando esta alcança enormes proporções em termos do seu volume. Foi feita uma apresentação da arquitetura do protótipo, expondo cada uma das entidades que lhe pertencem: Gerador de fluxos de informação, Gestor, Módulo, Técnicas de visualização e Transição, relacionando-as depois com as considerações e decisões de implementação, pela qual cada uma delas passou ao longo do desenvolvimento deste trabalho. Foi ainda detalhado o seu objetivo, para tornar possível a representação de grandes quantidades de dados, tendo em consideração as técnicas de agregação utilizadas, as medidas estatísticas de cada técnica de visualização, as operações mais complexas do sistema e as fórmulas utilizadas para calcular o movimento das reprentações gráficas de cada dado, nas técnicas de visualização associadas a cada módulo.

Capítulo 5

Avaliação

Neste capítulo são abordadas os testes que foram utilizados para avaliar o desempenho e a usabilidade do protótipo. Na Secção 5.1 são reportados os testes de eficiência aplicados a cada uma das técnicas de transição desenvolvidas, com o objetivo de compreender as suas limitações. Foram também realizados teste de usablidade, sendo apresentada toda a metodologia e resultados obtidos. Com os resultados concluíu-se quais são as transições que mais se destacam para cada par de tipos de visualização, seguindo-se uma discussão onde é então realizada uma análise global dos resultados obtidos perante a avaliação desenvolvida.

Para testar o protótipo, utilizou-se o navegador *Google Chrome* (versão *77.0.3865.90* 64 bits) instalado num sistema com o Windows 10 Pro com um processador *Intel® Core™ i7-4770K (3.50 GHz)*, 16 GB de memória RAM e ainda uma resolução 1920x1080. A visualização renderizada (1110x512 px) foi dividida em três módulos, com larguras iguais a 455, 200 e 455 px.

5.1 Testes de Eficiência

A realização de testes de eficiência ao protótipo permitiu identificar as limitações da mesma perante a receção de enormes quantidades de dados, visto que a mesma poderá influenciar a velocidade, o processamento e ainda a fluidez com que as representações dos dados se movimentam na visualização. Podendo levar à diminuição do número de quadros por segundo (FPS - *frames per second*), o que provoca maior lentidão e menor desempenho no sistema.

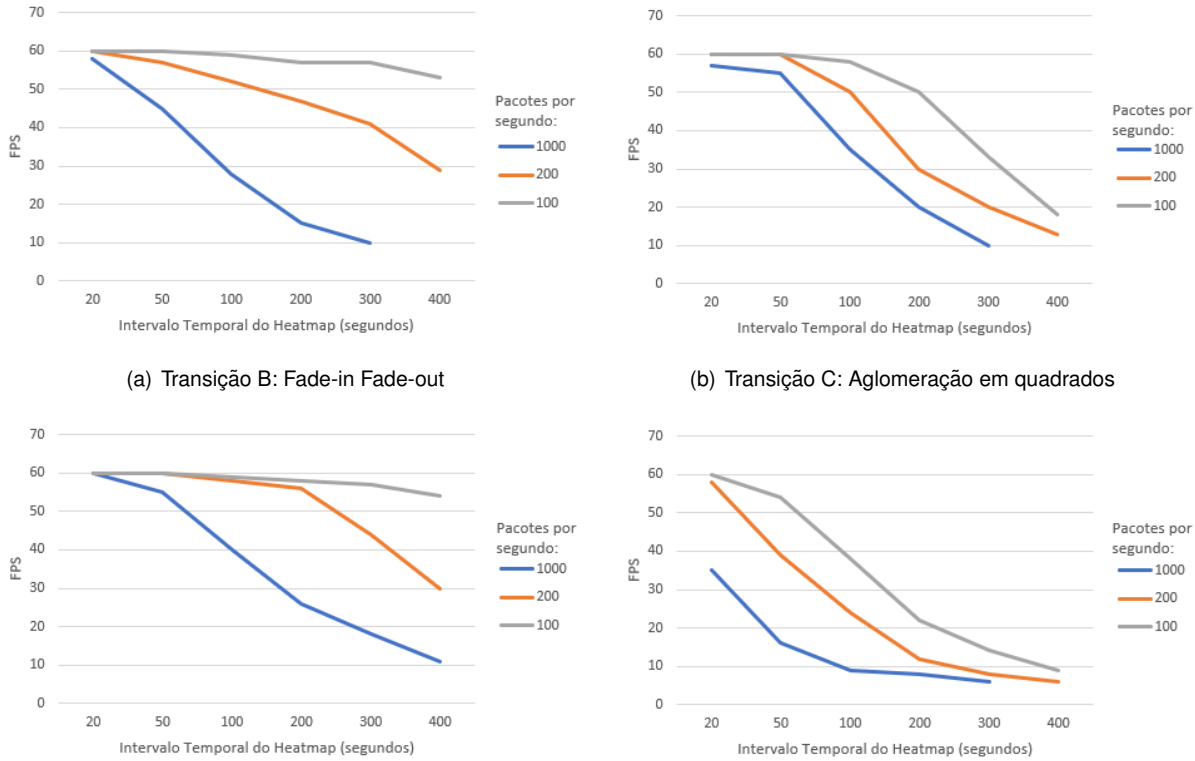
Para testar a eficiência do sistema desenvolvido, foi utilizada uma ferramenta do *Google Chrome* que permite calcular e visualizar o desempenho de aplicações *browser-based*, registando as durações da execução de cada método que foi invocado ao longo da análise.

Foram escolhidas as várias técnicas de visualização que permitem atribuição de intervalos de tempo (duração): Heatmap, Linechart e Streamgraph. Depois, foram escolhidas as várias transições desenvolvidas neste trabalho, com o objetivo de se verificar os valores dos FPS resultantes e o débito de dados durante 20 segundos de execução após todos os módulos estarem totalmente preenchidos com informação. Fazendo depois variar os intervalos de tempo, correspondentes ao módulo da esquerda e

ainda a velocidade com que os pacotes eram recebidos pelo sistema. Os módulos associados a uma técnica de visualização dispunham de 455 px de largura, enquanto o módulo da transição apresentava 200 px. Já à altura da visualização, correspondiam 512 px.

5.1.1 Transições entre Scatterchart e Heatmap

Perante a transição entre o Scatterchart e o Heatmap, a técnica A: Sem animação, quando pretendia transitar os pontos para os quadrados do Heatmap, possuindo este último um intervalo temporal de 6.7 minutos (400 segundos), obteve uma média de 58 FPS, para um débito igual a 1000 pacotes por segundo. Com o objetivo de se verificar e analisar os FPS das restantes técnicas de transição que foram estudadas, testou-se para diferentes débitos (1000, 200 ou 100 pacotes por segundo), uma relação dos FPS com os intervalos temporais do módulo do Heatmap, os resultados estão apresentados na Figura 5.1. A transição que se mostrou como a menos eficiente no que toca ao seu desempenho para com o sistema, foi a técnica E: Granulado. Esta ao sobrepor muitos pontos e incrementá-los ao longo da sua evolução com o eixo horizontal, acabou por causar um impacto negativo na fluidez das animações, não sendo a mais indicada para quando se tem uma grande diferença de intervalos temporais entre o Scatterchart e o Heatmap. No entanto, se o débito de pacotes por segundo for inferior, esta técnica poderá ser utilizada, obtendo-se cerca de 30 FPS quando se tem uma duração de cerca de 3 minutos no módulo do Heatmap. A técnica D: Colunas de dados, demonstrou um bom desempenho, no entanto



(a) Transição B: Fade-in Fade-out

(b) Transição C: Aglomeração em quadrados

(c) Transição D: Colunas de dados

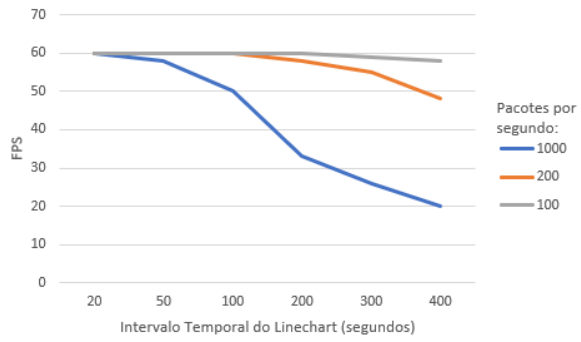
(d) Transição E: Granulado

Figura 5.1: Relação de FPS com Intervalo de tempo do Heatmap e débito recebido em pacotes por segundo

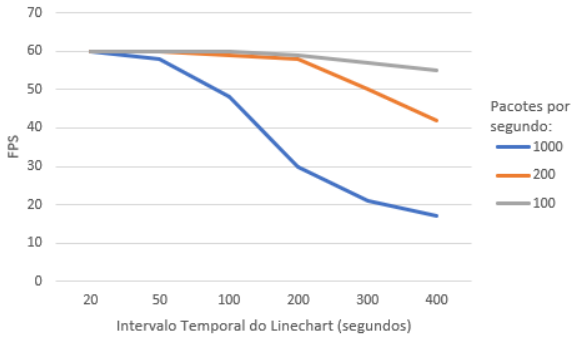
também acaba por limitar um pouco o sistema quando se tem um débito de 1000 pacotes por segundo e se pretendem intervalos temporais superiores a 3 minutos. As técnicas B: Fade-in Fade-out e D: Colunas de dados apresentaram os melhores resultados permitindo seguir a informação conforme esta transita entre as duas técnicas de visualização.

5.1.2 Transições entre Scatterchart e Linechart

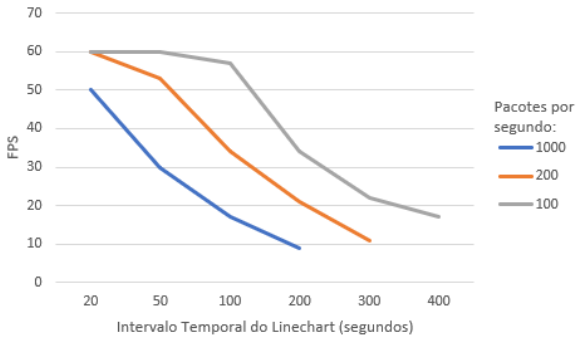
Já para a transição entre o Scatterchart e o Linechart, quando não se tem qualquer animação (técnica A: Sem animação) que permita seguir os dados que transitam entre as duas técnicas, sendo que o intervalo de tempo correspondente ao Linechart era igual a 6.7 minutos (400 segundos), obteve-se uma média de 60 FPS, para um débito de 1000 pacotes por segundo. De forma a verificar os FPS de cada uma das técnicas de transição desenvolvidas, foram testadas perante diferentes débitos (1000, 200 ou 100 pacotes por segundo), fazendo variar os intervalos temporais do módulo do Linechart. Os resultados obtidos estão presentes na Figura 5.2 e permitem verificar que a transição D: Contração de pontos é a que provoca menor desempenho no sistema. O mesmo se deve à complexidade das operações existentes para a renderização da animação existente na sua técnica transicional. As transições B: Fade-in Fade-out e C: Afunilamento, por sua vez, demonstraram uma capacidade melhor para representar as transformações entre o Scatterchart e o Linechart, obtendo uma média de FPS semelhante. A técnica que possui melhores registos de FPS para débitos iguais a 1000 pacotes por segundo, pe-



(a) Transição B: Fade-in Fade-out



(b) Transição C: Afunilamento



(c) Transição D: Contração de pontos

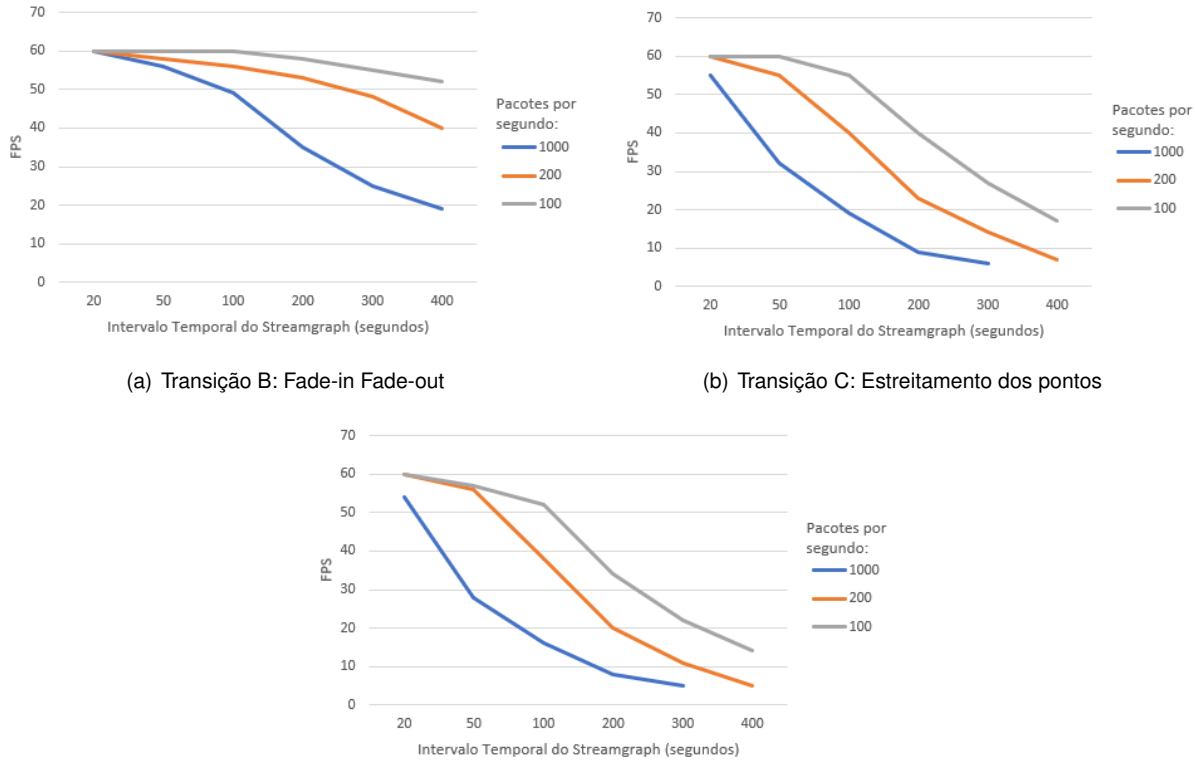
Figura 5.2: Relação de FPS com Intervalo de tempo do Linechart e débito recebido em pacotes por segundo

rante o maior intervalo temporal do Linechart (testado), é a técnica B: Fade-in Fade-out, no entanto a sua renderização poderá afetar a fluidez do sistema, sendo por isso recomendado diminuir o débito de pacotes recebidos ou a duração do intervalo correspondente ao módulo do Linechart.

5.1.3 Transições entre Scatterchart e Streamgraph

Por último, a transição entre o Scatterchart e o Streamgraph, perante a técnica A: Sem animação, que transita a informação entre as duas técnicas de visualização sem animações que permitam seguir as transformações existentes, foi testada utilizando fluxos de informação com um débito igual a 1000 pacotes por segundo, e com o intervalo temporal do módulo associado ao Streamgraph igual a 400 segundos. Como resultado, conseguiu-se cerca de 56 FPS, o que em comparação com os testes anteriores denota a complexidade da renderização das múltiplas áreas associadas a cada medida estatística representada pelo Streamgraph.

As restantes técnicas de transição para Streamgraph testadas permitiram constatar que a transição B: Fade-in Fade-out mantém os melhores valores de FPS ao longo das várias análises, isto é, perante os maiores intervalos temporais do Streamgraph e maior débito de informação por segundo. Sendo possível através da mesma, representar a transição da informação mantendo um sistema com valores de FPS quase sempre maiores ou iguais a 30, o que permite visualizar toda a informação com mais fluidez, à excepção das durações superiores a 5 minutos no módulo do Streamgraph. Já as técnicas



(a) Transição B: Fade-in Fade-out

(b) Transição C: Estreitamento dos pontos

(c) Transição D: Estampado de pontos

Figura 5.3: Relação de FPS com Intervalo de tempo do Streamgraph e débito recebido em pacotes por segundo

restantes, devido à sua semelhança na complexidade tanto em termos visuais (desenho) como em termos de operações (atualização), manifestaram quebras antecipadas nos FPS quando se têm débitos de 1000 pacotes por segundo. De forma a tornar possível a sua utilização com maior suavidade, é necessário aplicar intervalos temporais menores ao módulo do Streamgraph, ou reduzir o débito para menos pacotes por segundo, tal como é possível analisar na Figura 5.3.

5.1.4 Restantes transições

Não foram testadas as transições para o Barchart nem para o Heatmap Acumulador, pois essas técnicas de visualização não possuem uma relação com a escala temporal, não havendo por isso, necessidade de experimentar diferentes intervalos de tempo nos seus módulos, já que não iria influenciar a relação entre os quadros por segundo e o débito da informação recebida. Verificou-se para estes, 60 FPS indepentemente da técnica de transição atribuída.

5.1.5 Discussão

De um modo geral, os testes de eficiência realizados demonstraram que as técnicas de transição implementadas estão otimizadas para a receção de pacotes em grande escala, sendo que em média os melhores resultados foram obtidos quando o débito não era maior do que 200 pacotes por segundo, o que corresponde a 720,000 pacotes por hora. Já a duração do módulo correspondente ao Heatmap, Linechart e Streamgraph, foi normalmente melhor quando não excedia os 3 minutos. No entanto, o aumento do débito e duração do módulo, poderá influenciar a capacidade do sistema manter a sua fluidez na movimentação das representações gráficas dos dados a visualizar. Isto porque, a complexidade das animações desenvolvidas, quando focadas no conjunto de dados a transitar, agrava a quantidade das operações que são necessárias para animar as transformações visuais. Apesar das técnicas poderem ser utilizadas, as transições que se destacaram pelas baixas quebras nos FPS, ou seja, que menos influenciaram o desempenho do sistema, foram as técnicas Fade-in Fade-out, perante as transições para Heatmap, Linechart e Streamgraph. O motivo para a semelhança no conjunto destas transições, deve-se ao facto das suas técnicas não aplicarem métodos para movimentar verticalmente os pontos do Scatterchart, nem incrementarem o seu tamanho ou variarem as cores, o que evita uma série de operações que as restantes técnicas desenvolvidas não conseguem poupar.

5.2 Testes de usabilidade

A realização de testes de usabilidade, permitiu comparar as várias transições desenvolvidas entre diferentes tipos de visualização. O objetivo da sua realização era entender quais eram as técnicas que os utilizadores preferiram para cada transição existente, verificando o sucesso de percepção de padrões e tendências relevantes perante a existência de grandes quantidades de dados. Ambicionou-se ainda, entender através do grau de satisfação de cada utilizador, quais eram as técnicas de transição que

permitiam uma melhor análise dos *datasets* em questão e aquelas que os utilizadores mais gostavam para representar as transformações entre o Scatterchart e as restantes técnicas de visualização.

5.2.1 Metodologia

Os testes consistiram na realização de um questionário (Anexo A) disponibilizado através da ferramenta *Google Forms*[1]. Foram realizados testes presenciais e *online*, para aumentar o número de participantes, já que cada sessão demorou em média 40 minutos.

Os participantes foram introduzidos ao questionário que iam realizar, tendo sido explicada a ordem das tarefas e respetivas perguntas que iam responder. Depois foi dada uma breve introdução às técnicas de visualização existentes, explicando com ajuda de uma imagem, o significado dos eixos que compõem a visualização, assim como, a sua divisão em diferentes módulos e a direção do movimento dos dados. Depois os participantes foram encorajados a realizar perguntas, seguindo-se a explicação de cada uma para então poderem iniciar o questionário.

No começo do questionário pediu-se aos participantes para responder a algumas perguntas sobre o seu perfil, e com o decorrer das secções seguintes, foi apresentado um vídeo por cada técnica de transição existente do Scatterchart para uma determinada técnica de visualização, com a duração aproximada de 1 minuto cada. Seguido a cada vídeo era pedido aos participantes para responder a um conjunto de perguntas sobre o mesmo e a transição em causa.

No final do questionário foi pedido aos participantes para sugerirem modificações ou novas técnicas além das apresentadas ao longo do questionário.

5.2.2 Conjuntos de dados (*Dataset*) e Configurações

De forma a testar cada uma das técnicas de transição desenvolvidas, foram escolhidos diferentes conjuntos de dados (Anexo B), evitando a utilização do mesmo para as várias técnicas.

Através de um gerador[2] de séries temporais, foram criados 9 conjuntos de dados diferentes, onde a evolução segundo o domínio de cada um permitiu analisar diferentes tendências. Este gerador permitiu a modificação dos períodos de tempo correspondentes ao domínio e ainda a adição de diferentes comportamentos que possibilitam a criação de cíclos e intervalo de valores para a variação dos dados (de forma aleatória), a incorporção de ruído, inclinação/declive e a base dos valores do conjunto de dados. O período de tempo e o domínio utilizado para criar cada um desses conjunto de dados, foi escolhido de forma a serem criados vídeos de 1 minuto onde se verificassem os padrões desejados, sendo apenas uma parte dos dados aproveitada para os testes realizados. A Figura 5.4 ilustra as tendências existentes em cada um dos conjuntos de dados gerados.

A ordem das secções correspondentes a cada par de visualizações e das técnicas de transição foi definida utilizando um desenho *Latin Square*[3] que permite criar várias sequências não repetidas, onde a posição de cada elemento só pode ocorrer uma vez.

[1] https://www.google.com/forms/about/
[2] http://denised.github.io/generate-time-series/
[3] https://hamsterandwheel.com/grids/index2d.php

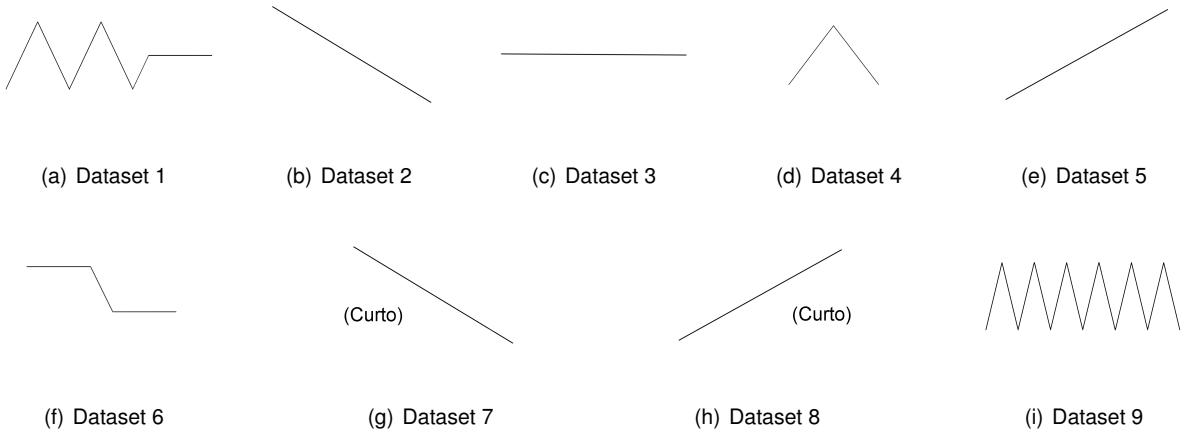


Figura 5.4: Tendências dos conjuntos de dados criados para testes de usabilidade

5.3 Tarefas

Durante a realização do questionário, foi pedido aos participantes que respondessem a um conjunto de perguntas sobre cada vídeo analisado. Os vídeos permitiam visualizar tendências e compará-las com a evolução dos dados ao longo do tempo. Cada conjunto de perguntas era composto por 4 perguntas de escolha múltipla sobre as tendências observadas e 2 perguntas cuja resposta era dada segundo uma Escala de Likert com valores compreendidos entre 1 e 5. Os conjuntos de perguntas foram repetidos ao longo das várias transições existentes para cada par de técnicas de visualização. Cada pergunta correspondia a aspetos diferentes:

1. **Encontrar tendências** - Este aspeto dividiu-se em dois, de forma a permitir a análise dos padrões existentes na informação nos dois módulos do sistema:

   (a) **Dados recém-obtidos** - Consistiu na análise da variação do valor dos dados, conforme estes iam evoluindo ao longo do primeiro módulo (Scatterchart). Os participantes tinham de identificar o tipo de padrão existente no fluxo de dados, reportando se estes se mantinham constantes, oscilavam, aumentavam ou diminuam o seu valor.

   Pergunta - O que estava a acontecer aos dados que iam chegando?

   (b) **Dados antigos** - De forma semelhante ao ponto anterior, este consistiu na análise das tendências existentes nos fluxos de dados que eram visíveis no módulo da esquerda, onde consoante o tipo de visualização associado e o *dataset* correspondente, permitia obter diferentes padrões.

   Pergunta - O que aconteceu aos dados mais antigos?

2. **Comparar tendências** - Os participantes tinham de analisar o fluxo de dados obtido para que depois fosse possível comparar a evolução do padrão visualizado ao longo de 1 minuto de observação e concluir se a tendência existente no início da análise se manteve até ao fim, ou se existiram alterações.

Pergunta - A tendência dos dados mais recentes mantém-se igual à dos dados anteriores?

3. **Identificar medidas estatísticas da agregação** - Este ponto focou-se na análise das medidas estatísticas resultantes da agregação. Durante a transição, os dados que tinham sido obtidos do Scatterchart foram sendo agregados de forma a serem visualizados no módulo da esquerda. Os participantes tinham então de selecionar, entre das várias opções, aquela ou aquelas que eram geradas na transição e depois visualizadas na técnica de visualização da esquerda.

   Pergunta - O que é mostrado no lado esquerdo da visualização?

4. **Classificar transições** - Contrariamente aos pontos anteriores, que permitiam verificar se o participante tinha entendido o que estava a ser visualizado, este (dividido em dois), pretendia obter uma classificação e por sua vez entender a sua preferência relativamente a cada técnica de transição desenvolvida:

   (a) **Compreensão da evolução do fluxo de dados** - Foi pedido aos participantes para classificarem, através de uma Escala de Likert (1-5), a facilidade de análise da evolução do fluxo de dados com a técnica de transição em questão.

   Pergunta - Concorda que esta transição ajudou a entender a evolução do fluxo de dados?

   (b) **Classificação geral da transição** - Consistiu na classificação, também ela com uma Escala de Likert (1-5), da transição analisada, com o objetivo de entender o que o participante achou sobre a transição e se gostou ou não da mesma.

   Pergunta - No geral, como classifica esta transição?

No final de todas as transições relativas a um par de técnicas de visualização, pediu-se ao participante para classificá-las segundo a sua ordem de preferência. Para isso, tinha de fornecer um número de 1 a 4 (ou 1 a 5 no caso de existirem 5 técnicas diferentes, como acontece com os Heatmaps), a cada uma das técnicas analisadas, sendo 1 a melhor.

5.4 Participantes

Os questionários foram realizados por um conjunto de 28 participantes (11 dos quais do sexo feminino e 17 do sexo masculino), com idades compreendidas entre 17-26 e 51-56 anos (82% entre 17-26 anos, 18% entre 51-56 anos). 89% já conhecia o termo *Big Data*. 10 participantes fizeram o teste presencialmente, enquanto os restantes 18 fizeram remotamente.

5.5 Resultados

Através dos questionários, e com o objetivo de realizar as tarefas anteriores, as 4 primeiras perguntas foram contabilizadas como certas ou erradas perante as soluções propostas para cada uma e as 3 perguntas seguintes, onde os utilizadores forneceram classificações a cada técnica conforme os valores de

uma escala de Likert de 1 a 5, foram contabilizadas consoante o valor numérico associado. Cada uma destas perguntas foi testada, com o intuito de verificar os aspetos referidos na Secção 5.3, comparando as várias técnicas de transição para cada par de técnicas de visualização existente.

5.5.1 Transições entre Scatterchart e Heatmap

A Tabela 5.1 relaciona cada técnica de transição para Heatmap desenvolvida com os vários aspetos abordados nas perguntas do questionário, apresentando a percentagem de respostas certas até ao quarto aspeto e a partir do quinto, os valores da mediana e do intervalo interquartil.

	TA	**TB**	**TC**	**TD**	**TE**
Comparação de tendências *	43%	79%	93%	79%	79%
Identificação da tendência nos dados novos	68%	61%	61%	43%	64%
Identificação da tendência nos dados antigos *	75%	75%	82%	32%	82%
Identificação de medidas estatísticas	43%	39%	46%	43%	54%
Compreensão da evolução do fluxo de dados *	4 (2)	4 (1)	3.5 (3)	4 (1)	4 (0)
Classificação da transição *	4 (2)	4 (1)	4 (3)	4 (1.75)	4 (1)
Classificação geral *	4 (2)	3 (2)	4 (2.75)	2 (2)	2 (2.75)

Tabela 5.1: Respostas ao questionário, considerando a percentagem de respostas certas até ao 4º aspeto e a partir do 5º, a mediana e intervalo interquartil para cada técnica de transição para Heatmap. * indica diferenças estatísticas significativas.

Verificando a relação de respostas certas e erradas para os quatro primeiros aspetos considerados no questionário, aplicou-se o teste Cochran, que mostrou diferenças estatísticas significativas na comparação de tendências (χ^2(2)=19.088, p=.001) e identificação de tendências nos dados antigos (χ^2(2)=27.840, p$<$.0005). Comparando os vários pares de transições, ao realizar-se testes de McNemar com a aplicação das correções de Bonferroni, concluiu-se que a transição C (rate=93%) foi melhor que a transição A (rate=43%, p=.01) para comparar tendências, a transição D (rate=32%) foi pior que a transição C (rate=82%, p=.02) e transição E (rate=82%, p=0.01) para identificar tendências nos dados mais antigos. Já para os três aspetos finais, aplicou-se o teste Friedman, que mostrou diferenças estatísticas significativas na classificação dada sobre o grau de compreensão da evolução do fluxo de dados (χ^2(2)=18.598, p=.001), classificação da transição (χ^2(2)=11.278, p=.024) e classificação geral segundo a ordem de preferência do participante (χ^2(2)=12.853, p=.012). Comparando os vários pares de transição, ao realizar-se testes de Wilcoxon com a aplicação das correções de Bonferroni, concluiu-se que a transição C foi pior que a transição B e D (transição B: Z=-3.079, p=.02; transição D: Z=-3.325, p=.01) na classificação dada pelos participantes para o grau de compreensão do fluxo de dados e pior que a transição B e E (transição B: Z=-2.863, p=.04; transição E: Z=-2.917, p=.04) na classificação da transição. A transição D foi melhor que a transição A (Z=-2.883, p=.04) na classificação geral, sendo por isso a preferida dos utilizadores em comparação à transição A.

5.5.2 Transições entre Scatterchart e Linechart

A Tabela 5.2 relaciona cada técnica de transição para Linechart desenvolvida com os vários aspetos abordados nas perguntas do questionário, apresentando a percentagem de respostas certas até ao quarto aspeto e a partir do quinto, os valores da mediana e do intervalo interquartil.

	TA	**TB**	**TC**	**TD**
Comparação de tendências	93%	68%	75%	71%
Identificação da tendência nos dados novos	86%	79%	71%	75%
Identificação da tendência nos dados antigos	86%	68%	86%	89%
Identificação de medidas estatísticas	79%	57%	61%	57%
Compreensão da evolução do fluxo de dados	4 (1.75)	4 (2)	4 (2)	4 (2)
Classificação da transição	4 (1.75)	4 (1.75)	4 (2)	4 (2)
Classificação geral	3 (1.75)	2 (2.75)	2.5 (1.75)	2.5 (2)

Tabela 5.2: Respostas ao questionário, considerando a percentagem de respostas certas até ao 4º aspeto e a partir do 5º, a mediana e intervalo interquartil para cada técnica de transição para Linechart.

Verificando a relação de respostas certas e erradas para os quatro primeiros aspetos considerados no questionário, aplicaram-se os testes Cochran e de McNemar, para verificar se existiam diferenças estatísticas significativas. Ao comparar os vários pares de transições, não foi possível retirar conclusões, o que pode ser justificado pela relação entre as taxas de acerto das perguntas que correspondem a cada aspeto, por cada técnica de transição. Já para os três aspetos finais que permitiam obter uma classificação segundo o participante, aplicou-se o teste Friedman, que também não demonstrou diferenças estatísticas significativas suficientes para se concluir quais foram as técnicas de transição que os utilizadores melhor classificaram.

5.5.3 Transições entre Scatterchart e Streamgraph

A Tabela 5.3 relaciona cada técnica de transição para Streamgraph desenvolvida com os vários aspetos abordados nas perguntas do questionário, apresentando a percentagem de respostas certas até ao quarto aspeto e a partir do quinto, os valores da mediana e do intervalo interquartil.

	TA	**TB**	**TC**	**TD**
Comparação de tendências	68%	71%	75%	61%
Identificação da tendência nos dados novos	36%	61%	61%	57%
Identificação da tendência nos dados antigos *	32%	43%	75%	79%
Identificação de medidas estatísticas	32%	36%	32%	29%
Compreensão da evolução do fluxo de dados	4 (2)	4 (1)	4 (2)	4 (1.75)
Classificação da transição	4 (2)	4 (1)	4 (2)	4 (1)
Classificação geral *	3 (1)	3 (1.75)	3 (2)	1 (0)

Tabela 5.3: Respostas ao questionário, considerando a percentagem de respostas certas até ao 4º aspeto e a partir do 5º, a mediana e intervalo interquartil para cada técnica de transição para Streamgraph. * indica diferenças estatísticas significativas.

Verificando a relação de respostas certas e erradas para os primeiros aspetos considerados no questionário, aplicou-se o teste Cochran, que mostrou diferenças estatísticas significativas na identificação

de tendências nos dados mais antigos (χ^2(2)=18.439, p<.0005). Comparando os vários pares de transições, ao realizar-se testes de McNemar com a aplicação das correções de Bonferroni, concluiu-se que a transição A (rate=32%) foi pior que a transição C (rate=75%, p=.048) e transição D (rate=79%, p=.012) para indentificar as tendências nos dados antigos. Já para os três aspetos finais, aplicou-se o teste Friedman, que mostrou diferenças estatísticas significativas na classificação geral segundo a ordem de preferência do participante (χ^2(2)=26.527, p<.0005). Comparando os vários pares de transição, ao realizar-se testes de Wilcoxon com a aplicação das correções de Bonferroni, concluiu-se que a transição D foi a que teve melhor classificação geral, vencendo à transição A, B e C (transição A: Z=-3.281,p=.006; transição B: Z=-4.167, p<.0005; transição C: Z=-3.676, p=.0014), sendo por isso a preferida dos utilizadores para representar a transformação dos pontos do Scatterchart para as as áreas correspondentes às medidas estatísticas do Streamgraph (mediana, intervalo interquartil e máximos e mínimos). A identificação de medidas estatísticas, demonstrou uma baixa percentagem de respostas certas, que pode ser justificado pela complexidade de interpretação das áreas do Streamgraph, sendo esta uma técnica de visualização pouco comum.

5.5.4 Transições entre Scatterchart e Barchart

A Tabela 5.4 relaciona cada técnica de transição para Barchart desenvolvida com os vários aspetos abordados nas perguntas do questionário, apresentando a percentagem de respostas certas até ao quarto aspeto e a partir do quinto, os valores da mediana e do intervalo interquartil.

	TA	**TB**	**TC**	**TD**
Comparação de tendências	57%	57%	61%	82%
Identificação da tendência nos dados novos *	68%	71%	54%	82%
Identificação da tendência nos dados antigos	64%	61%	61%	32%
Identificação de medidas estatísticas	57%	57%	71%	71%
Compreensão da evolução do fluxo de dados	4 (1.75)	4 (2)	4 (2)	4 (2)
Classificação da transição	4 (1.75)	4 (2)	4 (1.75)	4 (1.75)
Classificação geral *	3 (1.75)	3 (2)	1 (1)	3 (2.75)

Tabela 5.4: Respostas ao questionário, considerando a percentagem de respostas certas até ao 4º aspeto e a partir do 5º, a mediana e intervalo interquartil para cada técnica de transição para Barchart. * indica diferenças estatísticas significativas.

Verificando a relação de respostas certas e erradas para os quatro primeiros aspetos considerados no questionário, aplicou-se o teste Cochran, que mostrou diferenças estatísticas significativas na identificação de tendências nos dados mais recentes (χ^2(2)=14.417, p=.006). Comparando os vários pares de transições, ao realizar-se testes de McNemar com a aplicação das correções de Bonferroni, concluiu-se que a transição D (rate=82%) foi melhor que a transição C (rate=61%, p=.048) para identificar tendências nos dados recém-obtidos. Já para os três aspetos finais, aplicou-se o teste Friedman, que mostrou diferenças estatísticas significativas na classificação geral segundo a ordem de preferência do participante (χ^2(2)=18.927, p<.0005). Comparando os vários pares de transição, ao realizar-se testes de Wilcoxon com a aplicação das correções de Bonferroni, concluiu-se que a transição C foi melhor que a transição A e B (transição A: Z=-3.460, p=.006; transição B: Z=-3.385, p=.006) na classificação

geral dada pelos utilizadores às várias transições, sendo por isso melhor classificada em comparação com a transição A.

5.5.5 Transições entre Scatterchart e Heatmap Acumulador

A Tabela 5.5 relaciona cada técnica de transição para Heatmap Acumulador desenvolvida com os vários aspetos abordados nas perguntas do questionário, apresentando a percentagem de respostas certas até ao quarto aspeto e a partir do quinto, os valores da mediana e do intervalo interquartil.

	TA	**TB**	**TC**	**TD**	**TE**
Comparação de tendências *	54%	79%	82%	68%	39%
Identificação da tendência nos dados novos	64%	54%	71%	89%	54%
Identificação da tendência nos dados antigos	68%	54%	61%	75%	57%
Identificação de medidas estatísticas	50%	54%	57%	50%	43%
Compreensão da evolução do fluxo de dados	4 (2)	4 (1.75)	4 (2)	4 (2)	4 (1.75)
Classificação da transição	3.5 (2)	4 (2)	4 (1)	4 (1.75)	4 (2)
Classificação geral *	4 (3)	3 (1)	1 (1)	3 (2)	3 (2.75)

Tabela 5.5: Respostas ao questionário, considerando a percentagem de respostas certas até ao 4º aspeto e a partir do 5º, a mediana e intervalo interquartil para cada técnica de transição para Heatmap Acumulador. * indica diferenças estatísticas significativas.

Verificando a relação de respostas certas e erradas para os quatro primeiros aspetos do questionário, aplicou-se o teste Cochran, que mostrou diferenças estatísticas significativas na comparação de tendências (χ^2(2)=16.393, p=.003). Comparando os vários pares de transições, ao realizar-se testes de McNemar com a aplicação das correções de Bonferroni, concluiu-se que a transição C (rate=82%) foi melhor que a transição E (rate=39%, p=.02) para comparar as tendências ao longo da visualização do fluxo de dados. Já para os três aspetos finais, aplicou-se o teste Friedman, que mostrou diferenças estatísticas significativas na classificação geral considerando a ordem de preferência do participante (χ^2(2)=36.652, p$<$.0005). Comparando os vários pares de transição, ao realizar-se testes de Wilcoxon com a aplicação das correções de Bonferroni, concluiu-se que a transição C foi melhor que a restantes transições em termos das classificações gerais dadas (transição A: Z=-4.107, p$<$.0005; transição B: Z=-4.364, p$<$.0005; transição D: Z=-4.378, p$<$.0005; transição E: Z=-4.365, p$<$.0005), sendo por isso a preferida dos utilizadores para representar a transformação do Scatterchart para um Heatmap Acumulador.

5.5.6 Discussão

Através dos resultados obtidos nos questionários, e ainda as respostas às perguntas abertas onde era pedido aos participantes que referissem as maiores dificuldades que sentiram ao longo da realização das tarefas, alguns participantes referiram dificuldades na interpretação das medidas estatísticas representadas em algumas das técnicas de visualização utilizadas, havendo também casos em que a própria técnica de visualização era desconhecida, nomeadamente o Streamgraph, onde se verificou a maior dificuldade na interpretação das medidas estatísticas representadas pelo mesmo, como foi referido nos

resultados na Secção 5.5. O facto de não ter sido associado qualquer tipo de significado aos valores representados, tornou a experiência da análise da evolução do fluxo de informação mais complexo, não facilitando na compreensão de algumas técnicas. Houve ainda quem sentisse mais dificuldade em seguir os dados, de módulos onde existe uma relação com o tempo para outros associados aos tipos de visualização onde não existe contexto temporal, como é o caso do Barchart e Heatmap Acumulador.

Após ter sido feita uma comparação estatística entre as respostas fornecidas pelos participantes às perguntas de todas as técnicas de transição desenvolvidas, denotou-se que as respostas dadas para as transições do Heatmap, foram as que permitiram maior número de diferenças significativas, não permitindo no entanto, tirar grandes conclusões relativamente às suas transição preferidas. Já nas transições desenvolvidas para Streamgraph, Heatmap Acumulador e Barchart, os participantes marcaram através das classificações fornecidas, aquelas que eram as suas transições preferidas e determinaram através da ordem escolhida, que as melhores técnicas são:

- **D: Estampado de pontos**, para transitar para **Streamgraph**
- **C: Encaminhamento de pontos**, para transitar para **Heatmap Acumulador**
- **C: Guias**, para transitar para **Barchart**

Relacionando os resultados obtidos com os resultados dos testes de desempenho, concluíu-se que a técnica preferida dos participantes para transitar para Streamgraph, não demonstrou ser suficiente para representar débitos superiores a 200 pacotes por segundo, mantendo um histórico superior a 3 minutos para o Streamgraph. Já para transitar para o Heatmap Acumulador e para o Barchart, não existe qualquer limitação imposta pelo desempenho do protótipo. Conclui-se que é preciso trabalho futuro para tornar algumas técnicas preferidas dos utilizadores mais eficientes, como é o caso da transição para o Streamgraph.

As tendências e padrões visualizados no módulo da esquerda demonstraram ser melhor interpretados conforme a sua evolução com o tempo, nas transições onde existem animações para tipos de visualização que mantém uma relação com o eixo horizontal (tempo). Relacionando-se com a dificuldade de análise às representações sem contexto temporal, mencionada anteriormente. À exceção das transições para Linechart, a maioria das técnicas de transição demonstrou iguais ou melhores resultados nas comparações das tendências, do que as técnicas que não possuem qualquer tipo de animação.

Quanto às sugestões mencionadas pelos participantes, foi sugerido como melhoria para o sistema, a adição de um mecanismo de troca de cores da representação dos dados, de forma a sinalizar (ou alertar) quando existir uma alteração nas tendências dos fluxos de informação. Para permitir uma melhor análise de algumas técnicas de transição, foi proposto ainda um aumento da largura associada à mesma, tornando-se mais fácil a interpretação de algumas transformações visuais e, por sua vez, maior clareza na identificação das medidas estatísticas representadas no módulo seguinte. Além dessas opiniões, verificou-se que todos os participantes concordam que a utilização de transições animadas permite perceber melhor a evolução do fluxo de dados.

Capítulo 6

Conclusões

Esta dissertação apresentou o estudo realizado sobre formas de visualizar grandes quantidades de informação em tempo real, recorrendo a múltiplas técnicas de visualização que ao serem alinhadas horizontalmente, permitem seguir os fluxos de dados, enquanto estes são transitados para as técnicas seguintes, que por sua vez, possibilitam a representação dos dados através de diferentes níveis de detalhe fornecidos por métodos que os agregam. O foco principal deste trabalho passou pela concetualização de diversas técnicas de transição que permitam animar de forma suave, as transformações necessárias para transitar a informação entre as diferentes técnicas de visualização, facultando uma melhor análise e seguimento dos dados e evitando a perda de contexto da informação, por parte to utilizador.

Através da análise do estado da arte, verificou-se que grande parte dos sistemas existentes se limita à representação de domínios de dados cujo tipo não é adaptável, sendo também o volume de informação a representar, um dos principais desafios encontrados pelos autores dos vários trabalhos. É notável a aplicação de diferentes técnicas de processamento, redução e agregação dos dados, com o objetivo de minimizar o impacto dessa enorme quantidade de informação, já que a sua representação poderá levar à congestão e a quebras no desempenho dos sistemas. Tendo como consequência, visualizações menos fluidas.

Uma vez que a exploração de *Big Data* é considerada cada vez mais importante no mundo atual, a necessidade de visualizar padrões, tendências e correlações entre os dados, tornou-se ainda mais importante, uma vez que facilita a análise e permite compará-las ao longo da evolução do tempo. Uma das vantagens do conceito desenvolvido neste trabalho, passa precisamente pela representação da informação através de técnicas de visualização dispostas lado a lado, permitindo a divisão dos dados ao longo de intervalos de tempo e de tipos de agregação que permitem ver as suas tendências e padrões, e por sua vez compará-los com os outros intervalos existentes.

O objetivo passou por encontrar abordagens que permitissem suavizar os choques visuais causados pela transição dos dados de um tipo de visualização para outro, ou seja, de uma técnica de visualização cujo intervalo temporal, velocidade e técnica de agregação, difere da outra técnica, para onde a informação vai transitar. Foram, para isso, estudadas diferentes técnicas com animações que permitissem suavizar transições entre um Scatterchart e outro tipo de visualização, preservando as

tendências e padrões já visualizados e ao mesmo tempo, evitando a perda de contexto da informação.

Para realizar essa tarefa, foi criado um protótipo para correr num navegador *web*, através da tecnologia *Canvas*, que permitisse, desenvolver as técnicas de transição estudadas. O protótipo desenvolvido segue uma arquitetura cliente-servidor, onde é utilizado um servidor para enviar grandes fluxos de informação para o cliente. Este último, sendo o ponto principal, está organizado segundo uma estrutura modular, onde é utilizada uma entidade que permite controlar um conjunto de módulos aos quais estão associadas técnicas de visualização e transição. É realizada uma ”degradação graciosa”da informação, sendo esta representada através de diferentes níveis de detalhe e agregação, ao longo das técnicas de visualização, existindo transições entre as mesmas, que permitem realizar as animações e ainda, as operações de agregação, antes de seguirem para a técnica de visualização seguinte.

Além do protótipo permitir o desenvolvimento das transição concetualizadas, possibilitou a criação de diversos testes de eficiência e usabilidade, com o intuito de se verificar quais das técnicas propostas eram as que menos impacto tinham no desempenho do sistema, e por sua vez, na sua fluidez, fazendo variar do débito de pacotes recebidos por segundo e os intervalos de tempo correspondentes a cada tipo de visualização. Esta foi uma das principais preocupações ao longo do trabalho, tendo em conta o volume de dados que se pretende representar e a capacidade do utilizador os conseguir analisar em tempo real. Foram ainda apuradas as técnicas que permitiam uma melhor identificação e comparação de tendências ao longo da evolução dos fluxos de dados, assim como, as transições que os utilizadores consideravam melhores, para representar as transformações entre pares de visualizações.

Verificou-se, depois de analisados os resultados aos testes de usabilidade com 28 participantes, que os mesmos preferiram as técnicas de transição: D: Estampado de pontos, C: Encaminhamento de pontos e C: Guias, para transitar para o Streamgraph, Heatmap Acumulador e Barchart, respetivamente. No entanto, segundo os testes de eficiência realizados, as transições que menos impacto negativo tiveram no desempenho do sistema, foram as transições Fade-in Fade-out.

O protótipo foi testado com 100, 200 e 1000 pacotes por segundo, e obteve os melhores registos para débitos não superiores a 200 (720000 pacotes por hora), com um histórico de 3 minutos. Concluíu-se que, a quantidade de operações, movimentos complexos e variações de tamanhos e cores, está diretamente relacionada com as quebras dos quadros por segundo e por conseguinte, com a fluidez do sistema. A técnica preferida pelos participantes, nos testes de usabilidade para o Streamgraph, apesar de não obter os melhores resultados em termos de desempenho, pode ser utilizada, assim como qualquer uma das restantes técnicas, desde que seja estabelecido um equilíbrio entre o débito de pacotes por segundo e o intervalo correspondente ao tipo de visualização para onde se destinam os pontos do Scatterchart.

Perante um dos grandes desafios deste tipo de trabalhos, que é a quantidade de informação existente e a dificuldade em manter com clareza os padrões e tendências dos dados, ao longo da evolução do tempo e conforme novos dados vão sendo obtidos em tempo real, chegou-se ao objetivo, que era criar múltiplas técnicas de transição com animações suaves que permitissem seguir a transformação e movimentação dos dados, ao longo de várias técnicas de visualização associadas a um nível de agregação, velocidade e intervalo de tempo diferentes do anterior, de forma a manter um estado coe-

rente da visualização e das enormes quantidades de informação cujas tendências e padrões já tinham sido visualizados.

A realização deste trabalho resultou numa aprendizagem sobre diferentes técnicas para representar informação, quer utilizando tipos de visualização mais comuns, ou outros mais complexos, através de tecnologias como Javascript, D3, WebSockets e Canvas, que permitiram ainda, o desenvolvimento de animações que se baseiam na movimentação e alterações de tamanho e cor dos elementos gráficos que compõem a visualização. Procurando, sempre que possível, métodos que não influenciem negativamente o desempenho do sistema.

6.1 Trabalho Futuro

Concedeu-se a possibilidade de desenvolvimentos futuros sobre o protótipo, mais abrangentes e possibilitando não só melhorias e o aperfeiçoamento do sistema, como a criação de novas técnicas de visualização e transição. Além disso, é sugerida ainda a passagem do sistema para outra tecnologia que permita utilizar o GPU (unidade de processamento gráfico) para acelerar o processamento, evitando que o mesmo seja realizado apenas pelo processador e navegador. Facultando mais quadros por segundo, quando se pretende representar maiores débitos de informação. Um exemplo de uma boa tecnologia para este propósito, é o webGL[1], para o qual existe muita documentação e a sua abordagem não difere muito da que foi utilizada neste trabalho. Permitindo deste modo, aumentar os fluxos de informação, assim como a capacidade de representação de mais dados nos módulos. Permite também, aumentar a duração dos módulos cuja técnica visualização associada, tem como objetivo representar um histórico cada vez maior.

Propõe-se ainda, como trabalho futuro, a criação de transições que permitam ao utilizador seguir com facilidade a alteração da técnica de visualização associada a um módulo, para quando este muda o seu domínio ou se sabe que as tendências naquele intervalo de tempo vão mudar. E como tal, o tipo de visualização associado, poderá também ele ter de ser ajustado. Complementando-se a esse ajuste, a mudança consecutiva da transição entre módulos, visto que esta irá ser afetada pela alteração das técnicas de visualização adjacentes.

Outro aspeto a introduzir, é a representação de outros tipos de dados, além dos dados quantitativos, que foram os abordados neste trabalho. Sendo necessário realizar um novo estudo sobre as melhores técnicas de visualização para esses tipos de dados a introduzir.

[1] https://www.khronos.org/webgl/

Referências

[1] S. M. Ali, N. Gupta, G. K. Nayak, and R. K. Lenka. Big data visualization: Tools and challenges. In *2016 2nd International Conference on Contemporary Computing and Informatics (IC3I)*, pages 656–660. IEEE, 2016.

[2] J. Alsakran, Y. Chen, D. Luo, Y. Zhao, J. Yang, W. Dou, and S. Liu. Real-time visualization of streaming text with a force-based dynamic system. *IEEE Computer Graphics and Applications*, 32(1):34–45, 2011.

[3] M. Ankerst, D. A. Keim, and H.-P. Kriegel. Circle segments: A technique for visually exploring large multidimensional data sets. In *Visualization*, 1996.

[4] L. Battle, M. Stonebraker, and R. Chang. Dynamic reduction of query result sets for interactive visualizaton. In *2013 IEEE International Conference on Big Data*, pages 1–8. IEEE, 2013.

[5] L. Berry and T. Munzner. Binx: Dynamic exploration of time series datasets across aggregation levels. In *IEEE Symposium on Information Visualization*, pages p2–p2. IEEE, 2004.

[6] D. M. Best, S. Bohn, D. Love, A. Wynne, and W. A. Pike. Real-time visualization of network behaviors for situational awareness. In *Proceedings of the seventh international symposium on visualization for cyber security*, pages 79–90. ACM, 2010.

[7] D. M. Blei, A. Y. Ng, and M. I. Jordan. Latent dirichlet allocation. *Journal of machine Learning research*, 3(Jan):993–1022, 2003.

[8] T. Cerquitelli, E. Di Corso, S. Proto, A. Capozzoli, F. Bellotti, M. G. Cassese, E. Baralis, M. Mellia, S. Casagrande, and M. Tamburini. Exploring energy performance certificates through visualization. In *EDBT/ICDT Workshops*, 2019.

[9] P. Cudré-Mauroux, H. Kimura, K.-T. Lim, J. Rogers, R. Simakov, E. Soroush, P. Velikhov, D. L. Wang, M. Balazinska, J. Becla, et al. A demonstration of scidb: a science-oriented dbms. *Proceedings of the VLDB Endowment*, 2(2):1534–1537, 2009.

[10] G. Fialho. Vismillion and change (thesis). 2018.

[11] F. Fischer, F. Mansmann, and D. A. Keim. Real-time visual analytics for event data streams. In *Proceedings of the 27th Annual ACM Symposium on Applied Computing*, pages 801–806. ACM, 2012.

[12] M. Hao, D. A. Keim, U. Dayal, D. Oelke, and C. Tremblay. Density displays for data stream monitoring. In *Computer Graphics Forum*, volume 27, pages 895–902. Wiley Online Library, 2008.

[13] M. C. Hao, U. Dayal, D. A. Keim, and T. Schreck. Importance-driven visualization layouts for large time series data. In *IEEE Symposium on Information Visualization, 2005. INFOVIS 2005.*, pages 203–210. IEEE, 2005.

[14] X. Jin, B. W. Wah, X. Cheng, and Y. Wang. Significance and challenges of big data research. *Big Data Research*, 2(2):59–64, 2015.

[15] U. Jugel, Z. Jerzak, G. Hackenbroich, and V. Markl. M4: a visualization-oriented time series data aggregation. *Proceedings of the VLDB Endowment*, 7(10):797–808, 2014.

[16] M. Krstajić, E. Bertini, F. Mansmann, and D. A. Keim. Visual analysis of news streams with article threads. In *Proceedings of the First International Workshop on Novel Data Stream Pattern Mining Techniques*, pages 39–46. ACM, 2010.

[17] M. Krstajić and D. A. Keim. Visualization of streaming data: Observing change and context in information visualization techniques. In *2013 IEEE International Conference on Big Data*, pages 41–47. IEEE, 2013.

[18] C. Li and G. Baciu. Valid: A web framework for visual analytics of large streaming data. In *2014 IEEE 13th International Conference on Trust, Security and Privacy in Computing and Communications*, pages 686–692. IEEE, 2014.

[19] Z. Liu, B. Jiang, and J. Heer. immens: Real-time visual querying of big data. In *Computer Graphics Forum*, volume 32, pages 421–430. Wiley Online Library, 2013.

[20] F. Mansmann, M. Krstajic, F. Fischer, and E. Bertini. Streamsqueeze: a dynamic stream visualization for monitoring of event data. In *Visualization and Data Analysis 2012*, volume 8294, page 829404. International Society for Optics and Photonics, 2012.

[21] P. McLachlan, T. Munzner, E. Koutsofios, and S. North. Liverac: interactive visual exploration of system management time-series data. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 1483–1492. ACM, 2008.

[22] X. Qin, Y. Luo, N. Tang, and G. Li. Deepeye: An automatic big data visualization framework. *Big data mining and analytics*, 1(1):75–82, 2018.

[23] T. Repke and R. Krestel. Topic-aware network visualisation to explore large email corpora. In *EDBT/ICDT Workshops*, pages 104–107, 2018.

[24] J. Slack, K. Hildebrand, and T. Munzner. Prisad: A partitioned rendering infrastructure for scalable accordion drawing. In *IEEE Symposium on Information Visualization, 2005. INFOVIS 2005.*, pages 41–48. IEEE, 2005.

[25] J. Traub, N. Steenbergen, P. Grulich, T. Rabl, and V. Markl. I2: Interactive real-time visualization for streaming data. In *EDBT*, pages 526–529, 2017.

[26] H. Wickham. Bin-summarise-smooth: a framework for visualising large data. *had. co. nz, Tech. Rep*, 2013.

[27] J. Xia, F. Wu, F. Guo, C. Xie, Z. Liu, and W. Chen. An online visualization system for streaming log data of computing clusters. *Tsinghua Science and Technology*, 18(2):196–205, 2013.

[28] W. Xie, Y. Wei, H. Ma, and X. Du. Rbpcp: visualization on multi-set high-dimensional data. In *2017 IEEE 2nd International Conference on Big Data Analysis (ICBDA)(*, pages 16–20. IEEE, 2017.

Anexo A

Questionários

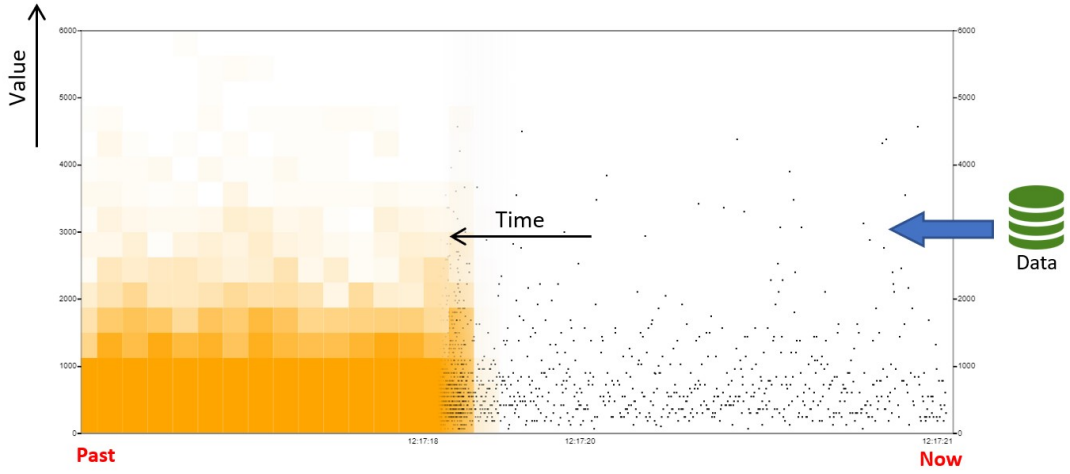
Este capítulo contém apenas uma parte do questionário que foi utilizado para os testes de usabilidade, dada a sua larga extensão. Foi extraída 1 de 22 técnicas de transição presentes no questionário, onde se apresentou um vídeo e um conjunto de perguntas relativas à técnica de transição Scatterchart - Heatmap C: Aglomeração em quadrados.

VisMillion and Change

* Required

With VisMillion and Change we intend to study the best way of representing large amounts of data in real-time, making use of smooth animated transitions. The goal here is to provide the best visual techniques allowing not only data exploration but also the comparison between the past and present. In order to do that, we created a prototype based on multiple chart types, each one representing data in a different way according to the corresponding time intervals.

The next image shows how the visualization works, the right side (present) has dots - scatterchart, representing the raw data and as the time goes by, the data starts to aggregate into another chart - heatmap, which is in the left side (past). This last one has a bigger time interval in order to sustain and show more data. Between both charts, we have a transition which is exactly where this study is focused.



We are testing 5 distinct transitions between charts and for each one we have 4 or 5 options.
During this form, you will be asked to answer multiple-choice questions based on short videos (1 minute each) according to different options for each transition type.
Please watch the videos until the end and take your time to answer the questions.

This form will take you about 30-40 minutes to answer.
Your data will remain anonymous and the results of your form will only serve for statistical measures.
Thank you very much in advance.

VisMillion and Change - Form

1. **Age** *

2. **Gender** *
   Mark only one oval.

   - ◯ Female
   - ◯ Male
   - ◯ Other

3. **Have you ever heard about Big Data?** *
   Mark only one oval.

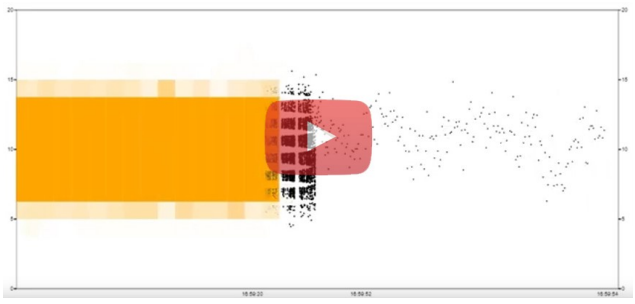
   - ◯ Yes
   - ◯ No

4. **Please select the topmost symbol** *
   Mark only one oval.

   - ◯ ¥
   - ◯ æ
   - ◯ ‡
   - ◯ ¢
   - ◯ ε

Transition Scatterchart - Heatmap

Transition Type C



5. **[TH3] How do you compare the latest data with the previously received?** *
Mark only one oval.

- ◯ Continued with the same trend
- ◯ The trend changed

6. **[TH3] What was happening to the incoming data points?** *
Mark only one oval.

- ◯ Oscillate (wave)
- ◯ Constant
- ◯ Increase value
- ◯ Decrease value

7. **[TH3] What happened to the previously received data?** *
Mark only one oval.

- ◯ Oscillate (wave)
- ◯ Constant
- ◯ Increase value
- ◯ Decrease value
- ◯ Increase and then decrease value
- ◯ Decrease and then increase value

8. **[TH3] What is shown on the left side of the visualization?** *
Check all that apply.

- ☐ Median
- ☐ Sum
- ☐ Count
- ☐ Interquartile Range
- ☐ Mean
- ☐ Maximum and minimum

9. **[TH3] Do you agree that this transition helped to understand the evolution of the data flow?** *
   Mark only one oval.

	1	2	3	4	5	
Strongly Disagree	◯	◯	◯	◯	◯	Strongly Agree

10. **[TH3] Overall, how do you rank this transition?** *
    Mark only one oval.

	1	2	3	4	5	
Awful	◯	◯	◯	◯	◯	Excellent

Please answer all the previous questions before scrolling down.

11. **[TH] Considering that the meaning of squares' colors can be customized, please rank each transition according to its suitability if we wanted to convey the count of data points.** *
    Please select a different rank for each transition type (5 - worst, 1 - best)
    Mark only one oval per row.

	5	4	3	2	1
Type C	◯	◯	◯	◯	◯
Type B	◯	◯	◯	◯	◯
Type A	◯	◯	◯	◯	◯
Type E	◯	◯	◯	◯	◯
Type D	◯	◯	◯	◯	◯

12. **What were your biggest difficulties during the analysis of the transitions?**

13. **Do you have any suggestions for new transition types or modifications to the presented ones?**

14. **Do you believe that animated transitions help to perceive the data flow?** *
Mark only one oval.

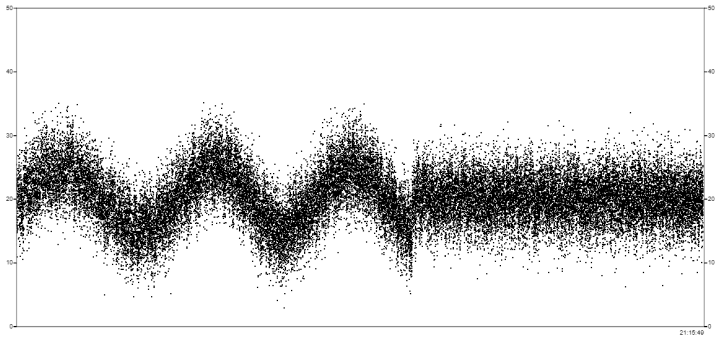
- Yes
- No
- Not sure

Anexo B

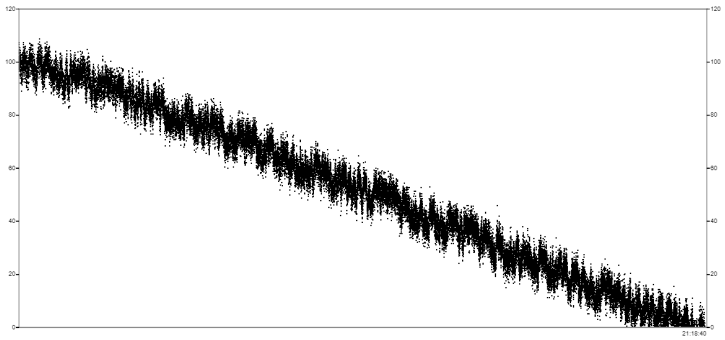
Conjuntos de dados

Este capítulo contém os conjuntos de dados utilizados nos testes de usabilidade. Foram efetuadas várias capturas de ecrã ao sistema, perante a utilização da técnica de visualização Scatterchart para representar todo o domínio dos dados. A Tabela B.1 relaciona cada uma das técnicas de transição testada com o conjunto de dados utilizado.

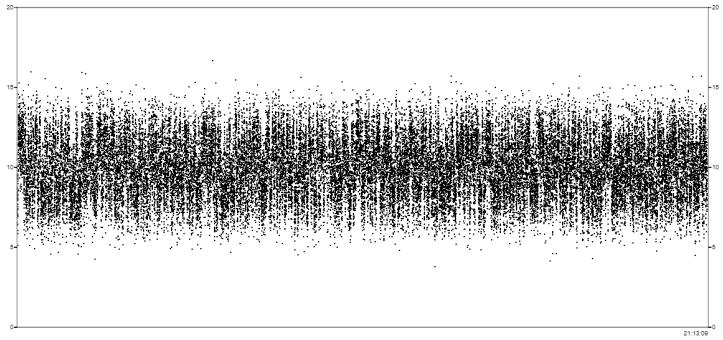
Dataset 1 - Duração = 100 segundos, Domínio Y = [0, 50]



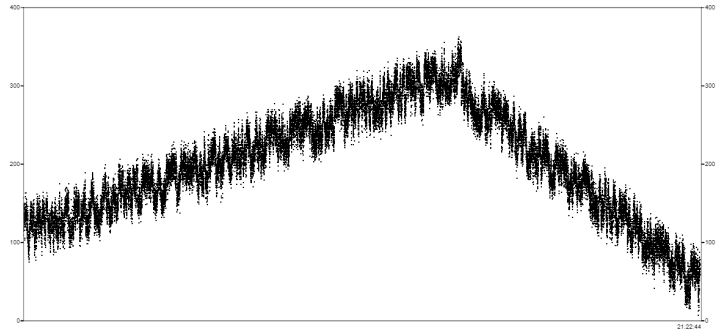
Dataset 2 - Duração = 100 segundos, Domínio Y = [0, 120]



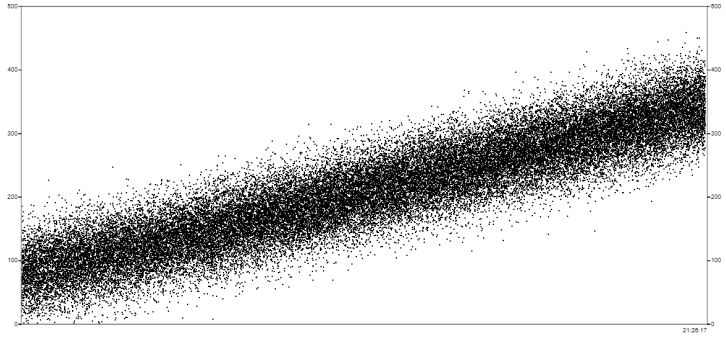
Dataset 3 - Duração = 100 segundos, Domínio Y = [0, 20]



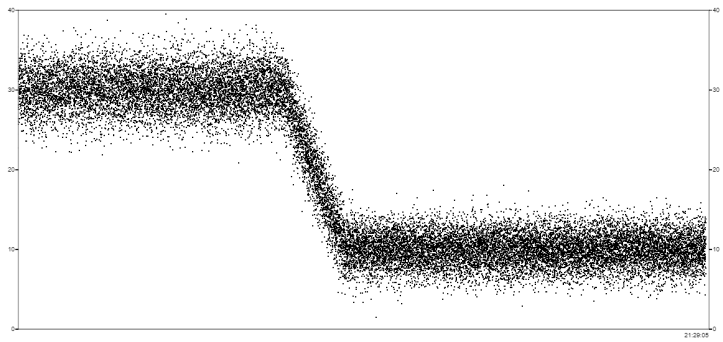
Dataset 4 - Duração = 100 segundos, Domínio Y = [0, 400]



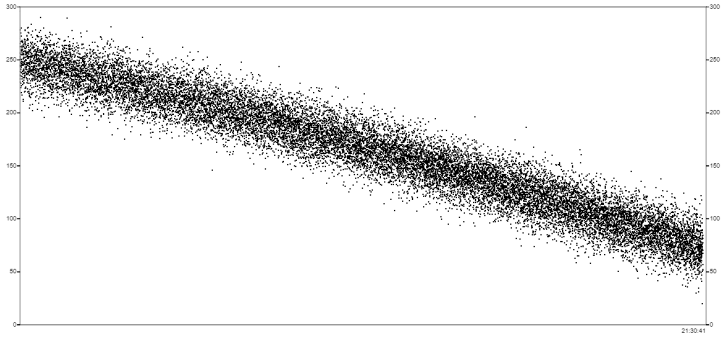
Dataset 5 - Duração = 100 segundos, Domínio Y = [0, 500]



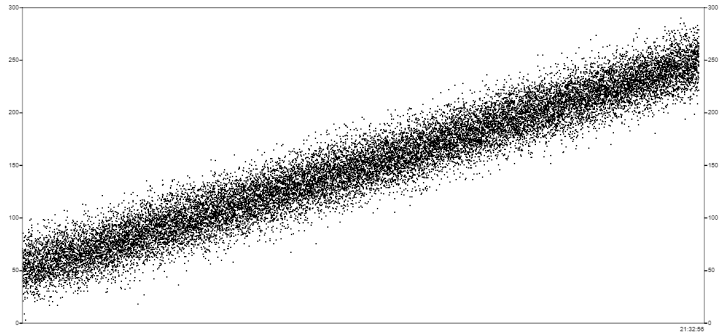
Dataset 6 - Duração = 50 segundos, Domínio Y = [0, 40]



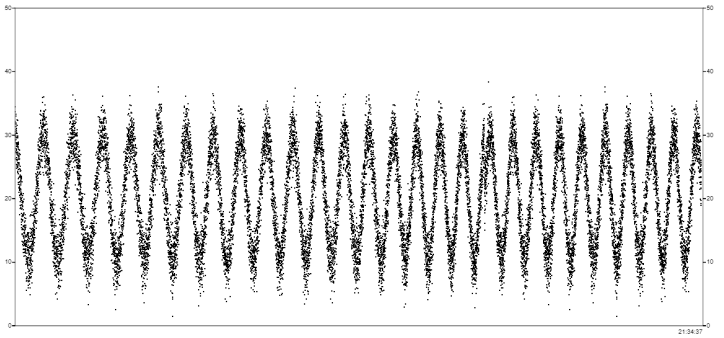
Dataset 7 - Duração = 50 segundos, Domínio Y = [0, 300]



Dataset 8 - Duração = 50 segundos, Domínio Y = [0, 300]



Dataset 9 - Duração = 50 segundos, Domínio Y = [0, 50]



Transição	Dataset
Scatterchart-Heatmap	
Transição A: Sem animação	Dataset 1
Transição B: Fade-in Fade-out	Dataset 2
Transição C: Aglomeração em quadrados	Dataset 3
Transição D: Colunas de dados	Dataset 4
Transição E: Granulado	Dataset 5
Scatterchart-Linechart	
Transição A: Sem animação	Dataset 3
Transição B: Fade-in Fade-out	Dataset 9
Transição C: Afunilamento	Dataset 5
Transição D: Contração de pontos	Dataset 2
Scatterchart-Streamgraph	
Transição A: Sem animação	Dataset 4
Transição B: Fade-in Fade-out	Dataset 1
Transição C: Estreitamento dos pontos	Dataset 3
Transição D: Estampado de pontos	Dataset 2
Scatterchart-Barchart	
Transição A: Sem animação	Dataset 7
Transição B: Fade-in Fade-out	Dataset 8
Transição C: Guias	Dataset 6
Transição D: Consumo de pontos	Dataset 4
Scatterchart-Heatmap Acumulador	
Transição A: Sem animação	Dataset 8
Transição B: Fade-in Fade-out	Dataset 6
Transição C: Encaminhamento de pontos	Dataset 4
Transição D: Eletrocardiograma	Dataset 9
Transição E: Dilatação	Dataset 7

Tabela B.1: Conjunto de dados utilizado por cada técnica de transição nos testes de usabilidade.