



**UNIVERSIDADE ESTADUAL DO CEARÁ  
CENTRO DE CIÊNCIAS E TECNOLOGIA  
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO  
MESTRADO PROFISSIONAL EM COMPUTAÇÃO APLICADA**

**BRUNO BEZERRA CHAVES**

**MÉTODOS COMBINATORIAIS PARA PROBLEMAS EM REDES DINÂMICAS:  
ALGORITMOS DE AGRUPAMENTO E PREVISÃO DINÂMICOS**

**FORTALEZA – CEARÁ  
2018**

BRUNO BEZERRA CHAVES

MÉTODOS COMBINATORIAIS PARA PROBLEMAS EM REDES DINÂMICAS:  
ALGORITMOS DE AGRUPAMENTO E PREVISÃO DINÂMICOS

Dissertação apresentada ao Curso de Mestrado Profissional em Computação Aplicada do Programa de Pós-Graduação em Ciência da Computação do Centro de Ciências e Tecnologia da Universidade Estadual do Ceará, como requisito parcial à obtenção do título de mestre em Ciência da Computação. Área de Concentração: Ciência da Computação

Orientador: Prof. Dr. Marcos José Negreiros Gomes

FORTALEZA – CEARÁ

2018

*Deve ser gerada através do preenchimento do Formulário Eletrônico de Elaboração da Ficha Catalográfica, disponível no link:  
[http://www.uece.br/biblioteca/index.php/entrega-de-trabalho.](http://www.uece.br/biblioteca/index.php/entrega-de-trabalho)*

X000x Sobrenome, Nome do 1º autor. (citado na folha de rosto)  
Título principal: subtítulo./Nome completo do 1º autor,  
Nome completo do 2º autor, Nome completo do 3º autor;  
orientação [de]. – Local: ano.  
Nº de folhas.: il.(se houver ilustração); 30 cm.

Inclui bibliografias: f.(nº da folha em que se encontra)  
Trabalho de Conclusão de Curso (Graduação em) –  
Universidade Estadual do Ceará – (UECE).

1. Assunto. 2. Assunto. 3. Assunto. I. Sobrenome, Nome do  
2º autor. II. Sobrenome, Nome do 3º autor. III. Sobrenome,  
Nome do orientador (orient.). IV. Universidade Estadual do  
Ceará – UECE. V. Título.

CDU

BRUNO BEZERRA CHAVES

MÉTODOS COMBINATORIAIS PARA PROBLEMAS EM REDES DINÂMICAS:  
ALGORITMOS DE AGRUPAMENTO E PREVISÃO DINÂMICOS

Dissertação apresentada ao Curso de Mestrado Profissional em Computação Aplicada do Programa de Pós-Graduação em Ciência da Computação do Centro de Ciências e Tecnologia da Universidade Estadual do Ceará, como requisito parcial à obtenção do título de mestre em Ciência da Computação. Área de Concentração: Ciência da Computação

Aprovado em: 10 de Dezembro de 2018

BANCA EXAMINADORA

---

Prof. Dr. Marcos José Negreiros Gomes (Orientador)  
Universidade Estadual do Ceará – UECE

---

Prof. Dr. Albert Einstein Fernandes Muritiba  
Universidade Federal do Ceará – UFC

---

Membro da Banca Três  
Universidade do Membro da Banca Três - SIGLA

## Dedicatória

## **AGRADECIMENTOS**

Agradecimentos

"Epígrafe"

(Autor)

## RESUMO

Arboviroses são doenças causadas pelos chamados arbovírus, que incluem o vírus da dengue, Zika vírus, febre chikungunya e febre amarela. Essas doenças estão cada vez mais voltando a atenção da OMS-TDR e das autoridades de saúde brasileiras para um esforço maior na prevenção e combate a endemias. Esta pesquisa fez importantes esforços para desenvolver, projetar e implementar uma estrutura computacional baseada na web, para ajudar a rastrear e gerenciar os recursos e pessoas no processo de prevenção e combate à arbovírus. Além disso, apresenta uma abordagem para solucionar o problema da previsão de agrupamentos dinâmicos espaço-temporal e combater as arboviroses com o esforço coordenado de uma estrutura de Sistemas de Apoio à Decisão para rastrear simultaneamente o mosquito e os casos em humanos, para prevenir e combater os territórios afetados. Foram implementados 2 métodos para visualização dos grupos formados. O primeiro é uma biblioteca que cria e gerencia grupos de acordo com o nível de zoom. O segundo, chamado algoritmo Convex Hull, consiste em gerar o menor polígono que englobe um determinado conjunto de pontos. O algoritmo ST-DBSCAN foi implementado como base para alcançar o método proposto. Por último, foram desenvolvidos 2 aprimoramentos no software DYNAGRAPH para possibilitar a avaliação dos métodos. Um deles foi a integração com um Editor de Características, que permite alterar os atributos visuais dos vértices e arestas de um grafo dinâmico. Outro novo recurso permitiu a visualização da formação de novos grupos dinâmicos. TODO - predição

**Palavras-chave:** Agrupamento Dinâmicos, Grafos dinâmicos.

## **ABSTRACT**

abstract **Keywords:**

## LISTA DE ILUSTRAÇÕES

<b>Figura 1 – Algoritmo STING: subdivisão de células e construção de árvores . . . . .</b>	<b>22</b>
<b>Figura 2 – Eps-vizinhança de <math>q</math> e Eps-vizinhança de <math>p</math> . . . . .</b>	<b>24</b>
<b>Figura 3 – Editor de Características: menu lateral . . . . .</b>	<b>31</b>
<b>Figura 4 – Editor de Características: criação de um vértice . . . . .</b>	<b>32</b>
<b>Figura 5 – Editor de Características: criação de um vértice - tipos de dados . . . . .</b>	<b>32</b>
<b>Figura 6 – Editor de Características: edição de um vértice - parte 1 . . . . .</b>	<b>33</b>
<b>Figura 7 – Editor de Características: edição de um vértice - parte 2 . . . . .</b>	<b>33</b>
<b>Figura 8 – Editor de Características: edição de um vértice - parte 3 . . . . .</b>	<b>34</b>
<b>Figura 9 – Editor de Características: edição de um tipo de vértice - parte 1 . . . . .</b>	<b>34</b>
<b>Figura 10 – Editor de Características: edição de um tipo de vértice - parte 2 . . . . .</b>	<b>35</b>
<b>Figura 11 – Editor de Características: Tipo de vértice . . . . .</b>	<b>35</b>
<b>Figura 12 – Editor de Características: Aresta . . . . .</b>	<b>36</b>
<b>Figura 13 – Grupos - API Google Maps parte 1 . . . . .</b>	<b>37</b>
<b>Figura 14 – Grupos - API Google Maps parte 2 . . . . .</b>	<b>37</b>
<b>Figura 15 – Exemplo 1 de Convex Hull . . . . .</b>	<b>38</b>
<b>Figura 16 – Exemplo 2 de Convex Hull . . . . .</b>	<b>38</b>
<b>Figura 17 – Arquitetura do Softwate . . . . .</b>	<b>50</b>

## **LISTA DE ALGORITMOS**

<b>Algoritmo 1 – Algoritmo DBScan . . . . .</b>	<b>25</b>
<b>Algoritmo 2 – Algoritmo DBScan - Expandir Cluster . . . . .</b>	<b>26</b>
<b>Algoritmo 3 – Algoritmo ST-DBScan . . . . .</b>	<b>29</b>

## SUMÁRIO

<b>1</b>	<b>INTRODUÇÃO . . . . .</b>	13
1.1	OBJETIVOS . . . . .	14
1.1.1	<b>Objetivo Geral . . . . .</b>	14
1.1.2	<b>Objetivos Específicos . . . . .</b>	14
1.2	HIPÓTESES . . . . .	14
1.3	JUSTIFICATIVA . . . . .	15
1.4	METODOLOGIA . . . . .	15
1.4.1	<b>Etapas metodológicas do projeto . . . . .</b>	15
1.4.1.1	Revisão da literatura e soluções existentes . . . . .	15
1.4.1.2	Análise de requisitos . . . . .	16
1.4.1.3	Arquitetura do software . . . . .	16
1.4.1.4	Modelo e desenvolvimento de software . . . . .	16
1.4.1.5	Ferramentas e Materiais . . . . .	16
1.5	ORGANIZAÇÃO DO TEXTO . . . . .	17
<b>2</b>	<b>CONCEITOS E REVISÃO BIBLIOGRÁFICA . . . . .</b>	18
2.1	AGRUPAMENTOS . . . . .	18
2.1.1	<b>Métodos baseados em particionamento . . . . .</b>	19
2.1.1.1	Agrupamentos K-Médias . . . . .	19
2.1.1.2	Agrupamentos K-Medoids . . . . .	20
2.1.2	<b>Métodos hierárquicos . . . . .</b>	20
2.1.2.1	O algoritmo BIRCH . . . . .	21
2.1.2.2	O algoritmo CURE . . . . .	21
2.1.3	<b>Métodos baseados em estrutura de grade . . . . .</b>	22
2.1.4	<b>Métodos baseados em densidade . . . . .</b>	23
2.2	MÉTODO DBSCAN . . . . .	23
2.2.1	<b>Algoritmo DBSCAN . . . . .</b>	25
2.3	MÉTODO ST-DBSCAN . . . . .	26
2.3.1	<b>Algoritmo ST-DBSCAN . . . . .</b>	27
2.3.2	<b>Agrupamento de dados baseado em predição . . . . .</b>	30
2.4	REDES DINÂMICAS . . . . .	30
2.4.1	<b>O modelo Dynagraph . . . . .</b>	30

2.4.1.1	Estrutura de dados . . . . .	31
<b>2.4.2</b>	<b>Editor de características . . . . .</b>	<b>31</b>
<b>2.4.3</b>	<b>Visualização dos grupos formados . . . . .</b>	<b>36</b>
2.5	TRABALHOS RELACIONADOS . . . . .	39
<b>3</b>	<b>MODELO DE AGRUPAMENTO E PREVISÃO EM REDES DINÂMICAS . . . . .</b>	<b>41</b>
<b>4</b>	<b>AVALIAÇÃO DO MÉTODO PROPOSTO . . . . .</b>	<b>42</b>
<b>5</b>	<b>CONCLUSÕES E TRABALHOS FUTUROS . . . . .</b>	<b>43</b>
5.1	CONSIDERAÇÕES FINAIS . . . . .	43
5.2	LIMITAÇÕES . . . . .	43
5.3	TRABALHOS FUTUROS . . . . .	43
	<b>REFERÊNCIAS . . . . .</b>	<b>44</b>
	<b>APÊNDICES . . . . .</b>	<b>46</b>
	APÊNDICE A – Tecnologias . . . . .	47
A.1	SERVIDOR . . . . .	47
A.2	BANCO DE DADOS . . . . .	47
A.3	CLIENTE . . . . .	48
A.4	ARQUITETURA DO SOFTWARE . . . . .	49

## 1 INTRODUÇÃO

Grandes quantidades de dados estão disponíveis para análise em organizações hoje em dia. Estas enfrentam vários desafios quando se tenta analisar dados gerados com o objetivo de extrair informações úteis. Esta capacidade analítica precisa ser reforçada com ferramentas capazes de lidar com grandes conjuntos de dados sem tornar o processo de análise uma tarefa árdua. Agrupamento de dados normalmente são usados no processo de análise de dados, pois esta técnica não exige qualquer conhecimento prévio dos dados. Contudo, os algoritmos de agrupamento geralmente requerem um ou mais parâmetros de entrada que influenciam o processo de agrupamento e os resultados que podem ser obtidos.

Nos últimos anos, o problema de agrupamento dinâmico tem atraído o interesse de pesquisas, impulsionado pelo aumento da disponibilidade de grandes conjuntos de dados contendo elementos espaciais e temporais. Este problema pode ser analisado como um problema de otimização. Seu objetivo principal é maximizar as diferenças das características dos indivíduos de grupos distintos, e minimizar as diferenças das características dos indivíduos de um mesmo grupo.

Agrupamento de dados ganhou uso muito difundido, especialmente para dados estáticos. No entanto, o rápido crescimento de dados espaço-temporais de inúmeros instrumentos, como os satélites em órbita terrestre, criou uma necessidade de métodos de agrupamento espaço-temporais para extrair e monitorar clusters dinâmicos. O agrupamento espaço-temporal dinâmico enfrenta dois grandes desafios: primeiro, os clusters são dinâmicos e podem mudar de tamanho, forma e propriedades estatísticas ao longo do tempo. Em segundo lugar, vários dados espaço-temporais são incompletos, ruidosos, heterogêneos e altamente variáveis sobre espaço e tempo.

O problema de agrupamento dinâmico com a componente de previsão divide-se em passos. A primeira etapa é obtenção das informações espaço-temporais mapeáveis e características do indivíduo. Neste passo, segue-se três estratégias para resolução do problema: os dados são analisados como um só grupo (Agrupamento Estático); trata-se os dados por intervalos pré-definidos; e mapeamento das evoluções entre intervalos observados. Sendo assim, pretende-se indicar o conjunto de grupos espacialmente correlacionados também no tempo.

Já o problema de previsão de grupos dinâmicos introduz o conceito de indicar os possíveis grupos que serão formados no tempo após um conjunto de eventos serem observados previamente.

O algoritmo proposto para previsão de agrupamentos é uma importante contribuição

deste trabalho, uma vez que poderá ser usado na obtenção de informações, na previsão de movimentação dos grupos e recomendação para o combate a endemias. O serviço proposto se baseia na localização passada dos casos de Dengue e Chikungunya.

## 1.1 OBJETIVOS

A seguir, são expostos os objetivos desta dissertação, definindo o produto final a ser obtido.

### 1.1.1 Objetivo Geral

Estudo e aplicação de métodos existentes e proposta de um método para resolver o Problema de Agrupamento em Grafos Dinâmicos e previsão de evolução destes agrupamentos.

### 1.1.2 Objetivos Específicos

Para que se alcance o objetivo geral, as seguintes metas foram estabelecidas:

- a) Utilizar o software Dynagraph como ambiente de suporte à visualização e interação com os resultados dos métodos de agrupamento espaço-temporal utilizados.
- b) Extração de características de previsão espaço-temporal sobre a evolução dos agrupamentos dinâmicos.
- c) Avaliação dos resultados sobre bases de dados dinâmicas e reais ligadas a evolução de casos de Dengue e Chikungunya.

## 1.2 HIPÓTESES

As hipóteses a seguir conduziram a elaboração desta dissertação:

- a) É possível a criação de um algoritmo capaz de sugerir novos agrupamentos geolocalizados baseados no tempo.
- b) É exequível a integração de um editor de características ao DYNAGRAPH, que é um software extensível.
- c) É realizável a utilização do modelo proposto de agrupamentos em grafos dinâmicos em um ambiente Web.

### 1.3 JUSTIFICATIVA

Esta pesquisa justifica-se por perceber-se a necessidade de ferramentas e estudos relacionando os assuntos abordados: agrupamento, previsão em dados dinâmicos espaço-temporais, grafos dinâmicos e sistemas web de forma integrada. E também, acelerar técnicas de agrupamento em grafos dinâmicos para tomada de decisão.

A relevância da pesquisa está em permitir uma análise dos dados extraídos para apoio à tomada de decisão, onde concentra-se na avaliação dos resultados sobre bases de dados dinâmicas relativas a casos de Dengue e Chikungunya. A pesquisa toma como base as características de evolução dos casos da doença observados entre 2015 e 2018 em Fortaleza. Os dados foram tomados a partir de (SIMDA, 2018), onde um estado é definido como o período de uma semana.

### 1.4 METODOLOGIA

A seguir, são descritas as etapas metodológicas para o desenvolvimento da dissertação.

#### 1.4.1 Etapas metodológicas do projeto

A presente pesquisa pode ser caracterizada quanto ao procedimento quantitativa e comparativa. A pesquisa quantitativa prioriza apontar numericamente a frequência e a intensidade dos comportamentos dos indivíduos de um determinado grupo, ou população. O método comparativo constitui-se em investigar coisas ou fatos e explicá-los de acordo com suas semelhanças e suas diferenças. Possibilita a análise de dados concretos e a dedução de semelhanças e divergências de elementos contínuos, abstratos e gerais, facilitando investigações de caráter indireto (FACHIN, 2001).

##### 1.4.1.1 Revisão da literatura e soluções existentes

As fontes principais de pesquisa foram sites especializados em pesquisas científicas, por exemplo, o portal de periódicos da CAPES, IEEE e outros sites de referências em que possuem livros, periódicos e dissertações disponíveis. Os temas essenciais abordados na pesquisa foram:

- Estrutura de dados em grafos dinâmicos

- Modelos de previsão espaço-temporais
- Algoritmos de agrupamentos dinâmicos

#### 1.4.1.2 Análise de requisitos

Detectou-se as necessidades de informações baseadas na previsão de agrupamentos dinâmicos em grafos. Assim sendo, e após obter os dados a partir de (SIMDA, 2018), houve a necessidade de um software para representação e tratamentos de grafos dinâmicos. Com isso foi escolhido o DYNAGRAPH, que tem como característica a extensibilidade.

#### 1.4.1.3 Arquitetura do software

A arquitetura do software desenvolvido é apresentada, assim como o diagrama de caso de uso (UML).

#### 1.4.1.4 Modelo e desenvolvimento de software

Foi desenvolvido um modelo capaz de representar agrupamentos e previsão dinâmicos. Em seguida, foi desenvolvido a partir do software DYNAGRAPH executar o modelo proposto e apresentar os resultados para validar a aplicação da ferramenta.

#### 1.4.1.5 Ferramentas e Materiais

O desenvolvimento do trabalho foi realizado a partir dos seguintes equipamentos e materiais:

- Macbook Pro 13"modelo 2015 / macOS High Sierra 10.13.2:  
Processador Intel Core i5 2.7 GHz;  
Memória de 8GB 1867 MHzs DDR3;  
HD SSD 128GB;
- Ambiente de desenvolvimento Webstorm;
- Navegador de internet Google Chrome 63.0.3239.132.
- Controle de versão Git (Software e dissertação);
- L<sup>A</sup>T<sub>E</sub>Xpara produção da dissertação;
- Portal de periódicos CAPES;

## 1.5 ORGANIZAÇÃO DO TEXTO

Esta dissertação está organizada em 5 capítulos. O capítulo 1 apresenta uma introdução à necessidade da representação e manipulação do agrupamento espaço-temporal dinâmico, assim como a previsão da formação de novos grupos dinâmicos. Em seguida são apresentados os objetivos, as hipóteses, a metodologia utilizada e as contribuições. O capítulo 2 constitui a revisão bibliográfica sobre modelagem com grafos dinâmicos, métodos de agrupamento por densidade, redes dinâmicas, o DYNAGRAPH, um editor de características e um conjunto de trabalhos relacionados a esta área de conhecimento. O capítulo 3 apresenta o modelo de agrupamento e previsão em redes dinâmicas. O capítulo 4 destaca os resultados e comparação dos algoritmos apresentados. O capítulo 5 apresenta as considerações finais e propostas de trabalhos futuros.

## 2 CONCEITOS E REVISÃO BIBLIOGRÁFICA

Uma visão geral sobre agrupamentos é apresentada na seção 2.1. Na seção 2.2 apresentamos o algoritmo DBSCAN em detalhes e suas características. A seção 2.3 apresenta o ST-DBSCAN, que é utilizada nesse trabalho para auxiliar os métodos de previsão dinâmica. Na seção 2.4 discutimos redes dinâmicas, exibimos o modelo DYNAGRAPH e um editor de características. A seção 2.5 define e apresenta os trabalhos relacionados ao problema de agrupamentos dinâmicos.

### 2.1 AGRUPAMENTOS

A técnica de agrupamento, também chamada de clustering, é uma das técnicas de mineração de dados mais comuns e é usada para descobrir padrões de distribuição nos dados. O agrupamento é feito com base na similaridade das características e na posição dos objetos. Dessa maneira, o objetivo é que os objetos do mesmo grupo sejam muito similares entre si e muito diferentes dos objetos de outros grupos.

Essa técnica é muito utilizada para dados estáticos. No entanto, há pouco trabalho no âmbito espaço-temporal onde os dados estão na forma de campos espaço-temporais contínuos e os agrupamentos são dinâmicos. Além disso, os dados espaço-temporais originados por satélites em órbita terrestre, telefones celulares e outros sensores tendem a ser ruidosos, incompletos e heterogêneos, tornando sua análise especialmente desafiadora (FAGHMOUS; KUMAR, 2013).

Agrupamentos dinâmicos podem mudar seu tamanho, forma, localização e propriedades estatísticas de um único passo para o próximo. Embora os agrupamentos possam se mover ou mudar de forma, existem vários pontos que não alteram as associações de grupos por um período de tempo. Tendo isso em vista é possível extrair de forma autônoma agrupamentos dinâmicos em dados espaço-temporais contínuos que podem conter valores, ruídos ou características muito variáveis.

Os métodos mais tradicionais são os particionais e os hierárquicos. Alguns algoritmos de agrupamento integram as idéias de vários outros, logo é difícil classificar um algoritmo pertencendo a somente uma categoria de método de agrupamento. Além do que, algumas aplicações podem ter critérios que necessitam a integração de várias técnicas de agrupamento. Os principais métodos de agrupamento existentes na literatura podem ser categorizados, como mostram as próximas sessões.

### 2.1.1 Métodos baseados em particionamento

A ideia principal desta classe de algoritmos de agrupamentos é criar  $K$  grupos dos dados, onde  $K$  é inserido pelo usuário. Esse método consiste em escolher  $K$  objetos como sendo os centros dos  $K$  grupos. Os objetos são divididos entre os  $K$  grupos de acordo com algum critério de similaridade estabelecido pelo algoritmo, de modo que cada objeto fique no grupo que tem o menor valor de distância entre o objeto e o centro dele. Os algoritmos de particionamento são muito populares devido à sua facilidade de implementação e baixo custo computacional; no entanto, eles têm essas desvantagens: (1) eles são sensíveis à presença de ruído e outliers, (2) eles podem descobrir apenas grupos com formas convexas e (3) o número de grupos precisa ser especificado.

#### 2.1.1.1 Agrupamentos K-Médias

O algoritmo de agrupamento k-médias é uma das técnicas mais populares dessa abordagem e foi introduzido em (MACQUEEN, 1967). Ele utiliza a média dos objetos que são do grupo em questão, também conhecido como centro de gravidade do grupo. A idéia principal neste algoritmo é usar a média dos objetos para atribuí-los a grupos e também usá-los para representar estes. K-médias é um algoritmo que garante a convergência para um ótimo local, mas não necessariamente um ótimo global. À medida que  $K$  aumenta, o custo de encontrar a solução ótima diminui, e atinge seu mínimo quando  $K$  é igual ao número de objetos (WU *et al.*, 2008). Especificamente, o procedimento é mostrado abaixo:

1. Inserir os objetos para serem agrupados e também o número  $K$  de grupos.
2. Escolher aleatoriamente os objetos  $K$  como centro dos grupos originais.
3. Atribuir cada objeto ao grupo com a média mais próxima.
4. Calcular a nova média de cada grupo.
5. Repetir a partir do passo 3.
6. Parar quando o critério de convergência estiver satisfeito. Outro critério, mais frequentemente usado, é a minimização do erro quadrático, dado por:

$$E = \sum_{i=1}^k \sum_{o \in C_i} |o - \mu_i|^2 \quad (2.1)$$

onde  $o$  é o ponto no espaço representando um dado objeto,  $\mu_i$  é o representante do grupo  $C_i$ , e  $K$  o número de grupos.

### 2.1.1.2 Agrupamentos K-Medoids

O algoritmo de K-Medoids foi introduzido primeiramente em (KAUFMANN; ROUSSEEUW, 1990) e não é tão sensível aos outliers quanto os k-médias. Nesse algoritmo, cada grupo é representado pelo objeto mais próximo ao centro, conhecido como medoid. O processo geral para o algoritmo é o seguinte:

1. Escolher aleatoriamente  $k$  objetos como os medoids iniciais.
2. Atribuir cada um dos objetos restantes ao grupo que possui o medoid mais próximo.
3. Em um grupo, selecionar aleatoriamente um objeto que não seja medoid (*nonmedoid*), que será referenciado como  $O_{nonmedoid}$ .
4. Calcular o custo de substituir o medoid com  $O_{nonmedoid}$ . Este custo é a diferença no erro quadrado se o medoid atual for substituído por  $O_{nonmedoid}$ . Se for negativo, faça  $O_{nonmedoid}$  o medoid do grupo. O erro quadrático é novamente somado ao erro de todos os objetos:

$$E = \sum_{i=1}^k \sum_{o \in C_i} |o - O_{medoid(i)}|^2 \quad (2.2)$$

onde  $O_{medoid(i)}$  é o medoid do grupo  $C_i$ .

5. Repetir a partir do passo 2 até que não haja mudanças.

## 2.1.2 Métodos hierárquicos

Nesta classe de algoritmos os objetos são colocados em uma hierarquia que é percorrida de uma forma bottom-up ou top-down para criar os grupos. A vantagem deste tipo de agrupamento é que ele não requer nenhum conhecimento sobre o número de grupos, e sua desvantagem é sua complexidade computacional (LIN *et al.*, 2004). Muitas vezes, uma estrutura em árvore, um dendrograma, é usada para representar os níveis hierárquicos aninhados.

Os aglomerativos funcionam de uma maneira bottom-up, assumindo inicialmente que cada elemento do conjunto de dados representa um grupo. Em seguida, os grupos são fundidos em grupos maiores até a criação de um grupo único ou qualquer outro critério de parada.

A abordagem divisiva trabalha de maneira top-down, considerando todo o conjunto de dados como um só grupo, que é dividido de maneira recursiva de acordo com a medida de similaridade estabelecida.

### 2.1.2.1 O algoritmo BIRCH

Em (ZHANG; RAMAKRISHNAN; LIVNY, 1996), BIRCH significa Balanced Iterative Reducing and Clustering using Hierarchies e é um algoritmo usado para agrupamentos em bases de dados muito grandes. É considerado um dos métodos hierárquicos mais utilizados na literatura. As vantagens do BIRCH são as seguintes:

- BIRCH trabalha de maneira local. Isso é obtido usando medições que indicam a proximidade natural dos pontos, de modo que cada decisão de agrupamento possa ser feita sem verificar todos os pontos de dados ou grupos existentes.
- Leva em conta a estrutura de dados espacial. Ele trata os pontos em uma região densa como um único grupo, enquanto os pontos em uma região esparsa são caracterizados como outliers e podem ser removidos opcionalmente.
- O algoritmo faz uso total da memória disponível enquanto minimiza os custos de Entrada/saída de dados.

Os principais conceitos do BIRCH, que funcionam de maneira incremental, são o Agrupamento por característica(AC) e AC-árvore. Onde AC consiste em todas as informações que precisam ser mantidas sobre um grupo. Já AC-árvore é utilizada para representar a hierarquia de grupos. O agrupamento acontece essencialmente em duas fases. Na primeira delas, o algoritmo lê o conjunto de dados e constrói uma AC-árvore inicial. Essa árvore é utilizada para representar a hierarquia dos grupos. Na segunda fase, um algoritmo de agrupamento selecionado é aplicado às folhas da AC-árvore, removendo grupos esparsos e agrupando os mais densos em grupos maiores.

### 2.1.2.2 O algoritmo CURE

Em (GUHA; RASTOGI; SHIM, 1998), um novo algoritmo hierárquico é proposto para detectar grupos que não são necessariamente convexos. Os grupos do CURE são representados por um número fixo de pontos bem espalhados que são encolhidos em direção ao centro do grupo por uma determinada fração. O CURE difere do algoritmo BIRCH de duas maneiras:

- O CURE começa desenhando uma amostra aleatória em vez de pré-agrupar todos os pontos de dados, como no caso do BIRCH.
- CURE primeiro particiona a amostra aleatória e, em seguida, em cada partição, os dados são parcialmente agrupados. Em seguida, os outliers são eliminados e os dados pré-agrupados em cada partição são agrupados para gerar os grupos finais.

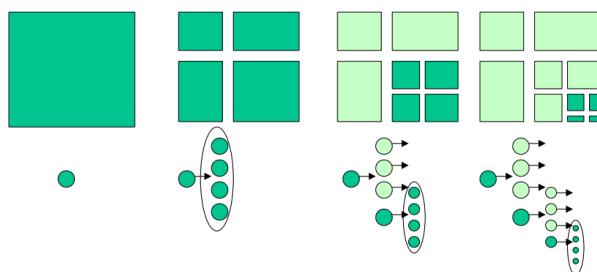
Os resultados experimentais no artigo mostram que o tempo de execução do CURE é sempre menor que o do BIRCH. Mais importante, os resultados mostram que, à medida que o tamanho do banco de dados aumenta, o tempo de execução do BIRCH aumenta rapidamente enquanto o tempo de execução do CURE aumenta muito pouco. O motivo é que o BIRCH varre todo o banco de dados e usa todos os pontos para o pré-agrupamento, enquanto o CURE usa apenas uma amostra aleatória.

### 2.1.3 Métodos baseados em estrutura de grade

Enquanto que os outros métodos de agrupamento discutidos são orientados aos dados, os métodos baseados em grade são orientados ao espaço. Essencialmente, o espaço é dividido em células retangulares, representadas por uma estrutura de grade hierárquica.

(WANG; YANG; MUNTZ, 1997) propuseram um método de agrupamento baseado em grade, STING (STatistical INformation Grid - Informação estatística baseada em grade), para agrupar bancos de dados espaciais e facilitar consultas orientadas à região. Esse algoritmo se baseia na construção de diversas camadas de grade, onde células de uma camada mais alta são subdivididas para a criação de células nas camadas mais baixas, como mostra a figura 1.

**Figura 1 – Algoritmo STING: subdivisão de células e construção de árvores**



Fonte: (BERKHIN, 2002)

O desempenho de STING depende da granularidade do nível mais baixo da estrutura de grade e o resultado dos grupos são limitados, pois só crescem na horizontal ou vertical e sofrem para buscar grupos de formatos complexos.

Os resultados produzidos pelo STING se aproximam do agrupamento produzido pelo DBSCAN a medida que a granularidade da estrutura de grade se aproxima de 0, podendo também ser considerado como um método baseado em densidade. Uma das vantagens do STING é a complexidade linear de tempo em relação ao número de objetos a serem agrupados.

### 2.1.4 Métodos baseados em densidade

Nesta classe de algoritmos, a idéia principal é manter os grupos em crescimento, desde que sua densidade esteja acima de um certo limite. A vantagem dos algoritmos baseados em densidade, em comparação com os algoritmos de particionamento baseados em distância, é que eles podem detectar grupos de forma arbitrária. Isso também fornece uma proteção natural contra outliers. Por outro lado, os algoritmos baseados em distância detectam apenas aglomerados de forma convexa.

Os agrupamentos baseados em densidade analisam a quantidade de elementos dentro de uma vizinhança de acordo com determinados parâmetros. A idéia-chave é que, para cada instância de um grupo, a vizinhança de um determinado raio deve conter pelo menos um número mínimo de instâncias.

A possibilidade de encontrar agrupamentos de forma eventual e o fato de não precisar da definição do número de agrupamentos (YIP; DING; CHAN, 2005) como parâmetro inicial são as principais vantagens dos métodos baseados em densidade. Entretanto, alguns algoritmos podem exigir a definição de outros parâmetros, como o caso do algoritmo DBSCAN (ESTER *et al.*, 1996) abordado na próxima seção.

## 2.2 MÉTODO DBSCAN

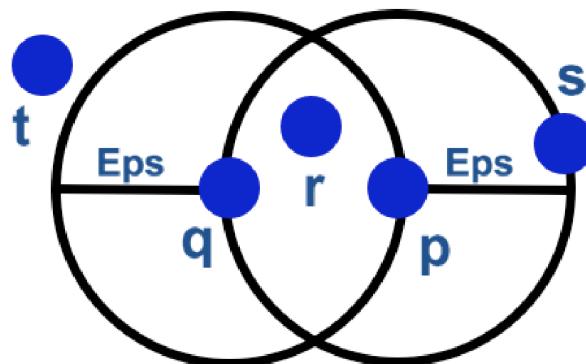
Este método calcula a densidade de uma região contando quantos pontos existem em uma determinada área seguindo uma determinada métrica, geralmente uma medida de distância, como a euclidiana ou manhattan. É um método efetivo para identificar grupos de formato arbitrário e de diferentes tamanhos, separar os ruídos dos dados, requer apenas um parâmetro de entrada, ajuda o usuário na determinação de um valor apropriado para ele e ajuda a detectar grupos e seus arranjos dentro do espaço de dados, sem qualquer informação preliminar sobre os grupos. (ESTER *et al.*, 1996) escrevem que a noção de agrupamentos e o algoritmo DBSCAN se aplicam para espaços euclidianos de duas e três dimensões, como para qualquer espaço característico de alta dimensão. O método DBSCAN é aplicável a qualquer base de dados contendo dados de um espaço métrico, isto é, bases de dados com uma função de distância para pares de objetos. Finalmente, o DBSCAN é eficiente mesmo para grandes bancos de dados espaciais. Para entender o método é necessário conhecer alguns conceitos básicos do algoritmo:

1. Vizinhança: Determinada pela função de distância, que pode ser distância euclidiana ou distância manhattan, para dois pontos  $p$  e  $q$ , dado por  $dist(p, q)$ .

2. Eps: raio ao redor de um ponto.
3. Eps-vizinhança: a vizinhança de um ponto  $p$  com raio Eps, dado por  $N_{Eps}(p)$ , é definido por  $N_{Eps}(p) = \{q \in D | dist(p, q) \leq Eps\}$ . Na figura 2 abaixo os círculos representam respectivamente a vizinhança Eps dos pontos  $q$  e  $p$ .
4. Ponto Central: Se o Eps-vizinhança de um ponto  $p$  contém ao menos um número mínimo, MinPts, de pontos, então o ponto  $p$  é chamado de ponto central. Por exemplo, na figura 2, se adotarmos MinPts = 4,  $p$  é um ponto central e os demais não são pontos centrais.
5. Ponto de Borda: Se a Eps-vizinhança de um ponto  $p$  contém algum ponto central e menos que MinPts então o ponto  $p$  é chamado de ponto de borda. Na figura 2,  $q$ ,  $r$  e  $s$  são pontos de borda.
6. Diretamente alcançável: um ponto  $p$  é alcançável se ele está no Eps-vizinhança de  $q$  e este é um ponto central.
7. Maximalidade: um ponto  $p$  é alcançável se existe uma cadeia de pontos que os liga, que respeita Eps e MinPts.
8. Conectividade: Um ponto  $p$  está conectado a um outro  $q$  se existe um objeto  $o$  tal que tanto  $p$  quanto  $q$  são alcançáveis a partir de  $o$ , com respeito a Eps e MinPts. Logo é uma relação simétrica.
9. Cluster: Seja  $D$  uma base de dados, um grupo  $C$  é um subconjunto não vazio de  $D$ , respeitando Eps e MinPts, onde:
 
$$\forall p, q \text{ se } q \in C \text{ e } p \text{ é alcançável a partir de } q.$$

$$\forall p, q \in C \text{ } p \text{ está conectado por densidade a } q.$$

**Figura 2 – Eps-vizinhança de  $q$  e Eps-vizinhança de  $p$**



Fonte: Elaborado pelo autor

O procedimento para encontrar um cluster é baseado no fato de que um cluster é

inequivocamente determinado por qualquer de seus centros (ESTER *et al.*, 1998). O DBSCAN pode utilizar R\*-tree para obter um melhor desempenho, mesmo assim a complexidade média do tempo de execução do DBSCAN será  $O(N \log N)$ , onde  $N$  é o número de objetos da base de dados (SHEIKHOLESLAMI; CHATTERJEE; ZHANG, 1998).

### 2.2.1 Algoritmo DBSCAN

As etapas envolvidas neste algoritmo são as seguintes:

1. Selecionar um ponto arbitrário  $p$ ;
2. Recuperar todos os pontos que são alcançáveis por densidade a partir do ponto  $p$ ;
3. Se  $p$  é um ponto central, um cluster é formado;
4. Se  $p$  é um ponto de borda, nenhum ponto é alcançável por densidade a partir de  $p$  e DBSCAN visita o próximo ponto do banco de dados;
5. Continuar o processo até que todos os pontos tenham sido processados.

---

#### **Algoritmo 1:** Algoritmo DBScan

---

**Entrada:** D, Eps1, MinPts, Delta, Epsilon

**Saída:** C

**início**

```

para  $i = 0$  até  $n$  faca
    se  $o_i$  não foi visitado então
        marcar  $o_i$  como visitado;
        Vizinhos = BuscarVizinhos( $o_i$ );
        se  $|Vizinhos| < MinPts$  então
            | MarcarRuido( $o_i$ );
        fim
        senão
            | C = próximo Cluster;
            | adicionar  $o_i$  ao cluster C;
            | expandirCluster( $o_i$ , C, Vizinhos);
        fim
    fim
fim
fim

```

---

Para agrupar os pontos levando em conta o fator tempo é necessário uma alteração

---

**Algoritmo 2:** Algoritmo DBScan - Expandir Cluster
 

---

**Entrada:** Vizinhos, C, MinPts

**início**

```

para  $i = 0$  até número de Vizinhos faca
     $p = Vizinhos_{(i)}$ ;
    se  $p$  ainda não foi visitado então
        marcar  $p$  como visitado;
        Vizinhos2 = BuscarVizinhos( $p$ );
        se  $Vizinhos2 \geq MinPts$  então
            |   Vizinhos = Vizinhos  $\cup$  Vizinhos2;
            fim
        fim
        se  $p$  não está em nenhum cluster então
            |   adicionar  $p$  ao cluster C;
        fim
    fim
fim

```

---

no algoritmo DBScan, e com isso detectar os grupos em relação ao tempo. Logo, o algoritmo determinada para esta implementação foi o ST-DBScan (BIRANT; KUT, 2007), abordado a seguir.

### 2.3 MÉTODO ST-DBSCAN

O algoritmo DBSCAN (ESTER *et al.*, 1998) usa apenas um parâmetro de distância Eps para medir a similaridade de dados espaciais com uma dimensão. A fim de suportar dados espaciais bidimensionais, o ST-DBSCAN (BIRANT; KUT, 2007) usa duas métricas de distância, Eps1 e Eps2, para definir a similaridade por uma conjunção de dois testes de densidade. O algoritmo ST-DBSCAN é construído modificando o algoritmo DBSCAN. Em contraste com o algoritmo de agrupamento baseado em densidade existente, o algoritmo ST-DBSCAN tem a capacidade de descobrir clusters de acordo com valores não espaciais, espaciais e temporais dos objetos. As três modificações feitas no algoritmo DBSCAN são as seguintes:

- Permitir o algoritmo ST-DBSCAN descobrir grupos em dados espaciais-temporais.
- Introdução do fator de densidade atribuído a cada cluster, que é o seu grau de densidade, para encontrar objetos de ruído quando existem clusters de diferentes densidades.

- A terceira modificação fornece uma comparação do valor médio de um cluster com o novo valor resultante. Necessária para resolver os conflitos em pontos de borda.

O algoritmo DBSCAN não é satisfatório quando existem clusters de diferentes densidades. Para superar esse problema, o ST-DBSCAN atribui a cada cluster um fator de densidade, que é o grau da densidade do cluster. A função DensityDistance é dada por:

$$\text{DensityDistance} = \frac{\text{DensityDistanceMax}}{\text{DensityDistanceMin}}$$

onde DensityDistanceMax de um objeto  $p$  denota a distância máxima entre o objeto  $p$  e seus vizinhos dentro do raio Eps. Da mesma forma, DensityDistanceMin de um objeto  $p$  denota a distância mínima entre o objeto  $p$  e seus vizinhos dentro do raio Eps.

$$\text{DensityDistanceMax}(p) = \max\{dist(p, q) | q \in D \wedge dist(p, q) \leq Eps\}$$

$$\text{DensityDistanceMin}(p) = \min\{dist(p, q) | q \in D \wedge dist(p, q) \leq Eps\}$$

O fator de densidade de um cluster  $C$  é dado por:

$$\text{DensityFactor}(C) = 1 / \left[ \frac{\sum_{p \in C} \text{DensityDistance}(p)}{|C|} \right]$$

Com esse fator é possível evitar problemas de agrupamentos com grande variação de densidade, onde os grupos são pequenos e muito densos ou grandes e pouco densos.

### 2.3.1 Algoritmo ST-DBSCAN

Uma grande diferença entre o DBSCAN e o ST-DBSCAN é a utilização de um segundo parâmetro EPS, onde é usado para medir a similaridade de valores não espaciais. Nesta pesquisa usa-se o tempo como EPS de valor não espacial. Para dois pontos serem considerados vizinhos eles devem respeitar os limites de espaço e tempo parametrizados por EPS1 e EPS2.

1. O algoritmo começa com o primeiro ponto  $p$  no banco de dados D.
2. Este ponto  $p$  é processado de acordo com o algoritmo DBSCAN e o próximo ponto é tomado.
3. A função BuscarVizinhos( $p$ , Ep1, Ep2) recupera todos os pontos que tem uma distância menor que Eps1 e Eps2 para o ponto  $p$ . Se os pontos recuperados não podem ser alcançados o ponto é atribuído como ruído, onde o ponto selecionado não tem vizinhos suficientes para ser armazenado em cluster.
4. Os pontos marcados como ruído podem ser alterados posteriormente, se não forem diretamente alcançáveis pela densidade, mas eles são alcançáveis por densidade de algum outro

ponto do banco de dados. Isso acontece para pontos de borda de um cluster.

5. Se o ponto selecionado tiver vizinhos suficientes dentro das distâncias Eps1 e Eps2 - se for um ponto central - então um novo cluster será construído.
6. Todos os vizinhos diretamente atingíveis por densidade desse ponto central também são incluídos.
7. O algoritmo reúne iterativamente pontos acessíveis por densidade a partir desse ponto central usando uma pilha.
8. Se o objeto não estiver marcado como ruído ou não estiver em um cluster e a diferença entre o valor médio do cluster e o novo valor é menor do que o valor limite para incluir em um cluster,  $\Delta E$ , ele é colocado no cluster atual.
9. Se dois clusters  $C_1$  e  $C_2$  estão muito próximos um do outro, um ponto  $p$  pode pertencer a ambos  $C_1$  e  $C_2$ . Então o ponto  $p$  é atribuído ao cluster que foi descoberto primeiro.
10. Depois de processar o ponto selecionado, o algoritmo seleciona o próximo ponto em D e o algoritmo continua iterativamente até que todos os pontos tenham sido processados.

---

**Algoritmo 3:** Algoritmo ST-DBScan
 

---

**Entrada:** D, Eps1, Eps2, MinPts,  $\Delta E$

**Saída:** C = ( $C_1, C_2, \dots, C_k$ ) Conjunto de clusters

**início**

**para**  $i = 1$  até  $n$  **faça**

**se**  $o_i$  não está no cluster **então**

            Vizinhos = BuscarVizinhos( $o_i$ , Eps1, Eps2);

**se**  $|Vizinhos| < MinPts$  **então**

                | marcar  $p$  como ruído;

**fim**

**senão**

**para**  $j = 1$  até  $|Vizinhos|$  **faça**

                | RotularPontos( $X_j$ );

                | AdicionarPontosAoCluster( $X_j, C_i$ );

**fim**

**enquanto** existir pontos na vizinhança de  $o_i$  **faça**

            pontoAtual = vizinho( $o_i$ );

            Vizinhos2 = BuscarVizinhos(pontoAtual, Eps1, Eps2);

**se**  $|Vizinhos2| \geq MinPts$  **então**

**para** todo  $o_y$  **faça**

**se** ( $\neg ehRuido(o_y) \vee \neg estaNumGrupo(o_y)$ )  $\wedge$

                        |  $DensidadeMediaDoGrupo(C_i) + o_y > \Delta E$  **então**

                            | AdicionarPontosAoCluster( $o_y, C_i$ );

**fim**

**fim**

**fim**

**fim**

**fim**

**fim**

**fim**

---

### **2.3.2 Agrupamento de dados baseado em predição**

A tarefa de predição visa descobrir o valor futuro de um determinado atributo de dados. É uma área com variedade de aplicações em diversas áreas como meteorologia e detecção de doenças. Por exemplo, um médico gostaria de prever a reação de seus pacientes a um novo medicamento para diabetes, particularmente a duração dos episódios de hipoglicemias. Na previsão de eventos, é desejável prever a ocorrência de um evento ou o número de ocorrências de um evento ou a duração de um evento, dada a existência de certas condições. Por exemplo, um médico acaba de colocar um de seus pacientes epilépticos em uma nova droga que é muito eficaz, mas após o início da terapia pode causar um grave ataque de enxaqueca. O médico gostaria de prever a duração desse ataque, considerando o conhecimento sobre a idade do paciente e o número e duração dos ataques epilépticos no último ano. Nos problemas de previsão de eventos que lidam com a previsão da duração de um evento pode ser modelada usando uma variável contínua e pode-se usar a regressão linear para sua previsão, onde a regressão linear é um dos métodos de regressão mais amplamente disponíveis (MITSA, 2010). Na previsão de séries temporais, os dados são dados históricos obtidos em intervalos de tempo regulares. Informações sobre padrões passados podem ser usadas para prever padrões futuros. No exemplo da enxaqueca, os dados poderiam ser a duração dos ataques de enxaqueca de outros pacientes sobre a droga, juntamente com informações sobre suas idades, condições médicas preexistentes e gravidade da epilepsia.

## **2.4 REDES DINÂMICAS**

Para redes dinâmicas, o foco é tipicamente em uma classe de redes e questões referentes à estrutura dessa classe de rede, como a estrutura evoluiu e como isso afeta sistemas dinâmicos na rede. As redes temporais normalmente são mais orientadas a dados - onde se investiga um conjunto de dados, suas estruturas e como, por exemplo, surtos epidêmicos se comportariam sobre ele. Em seguida, é questionado como essas observações se generalizam comparando os resultados para diferentes conjuntos de dados (HOLME, 2015).

### **2.4.1 O modelo Dynagraph**

O Dynagraph (CALIXTO; NEGREIROS, 2013) é um modelo computacional que permite, a partir de uma estrutura de dados simples, modelo de um Grafo ou Rede Dinâmica,

representar com o mínimo custo de armazenamento a evolução de uma instância de observação e estudos. Ele acompanha a evolução dos conjuntos de um grafo: de nós e ligações (arcos e elos), como inserção/retirada ao longo do tempo, e mudança de suas características (posição, cor, forma, tamanho, e outras), por meio de um editor de características, apresentado na próxima subseção.

#### 2.4.1.1 Estrutura de dados

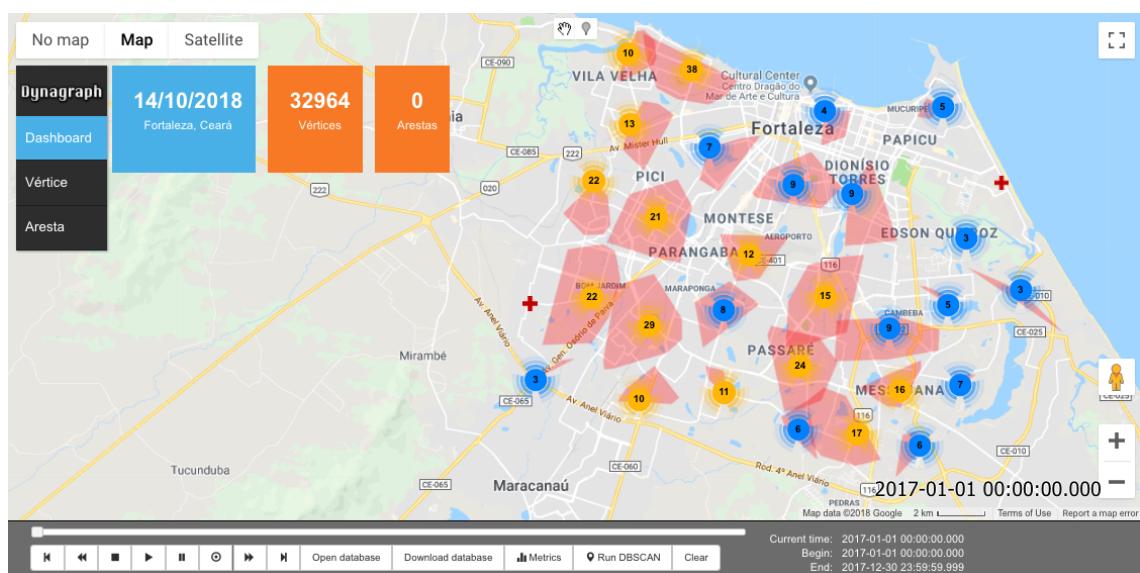
(TODO - anexar imagem da estrutura JSON usada pelo Dynagraph)

#### 2.4.2 Editor de características

O Editor de Características é uma extensão do software DYNAGRAPH, que permite alterar os atributos visuais dos vértices e aresta de um grafo dinâmico. Com ele podemos criar novos tipos de dados e com isso diferenciar, por exemplo, focos de casos de dengue e tipos de dengue. Essa extensão permite criar com os seguintes atributos: Rótulo, Opacidade, Escala, Espessura da Borda, Cor da Borda e Cor de Preenchimento. Outra opção é utilizar uma imagem.

A figura 3 exibe a localização do Editor de Características na aplicação em um menu lateral com as seguinte opções: Vértice, Aresta e um painel, este exibe a data atual e número de vértices e arestas em análise.

**Figura 3 – Editor de Características: menu lateral**

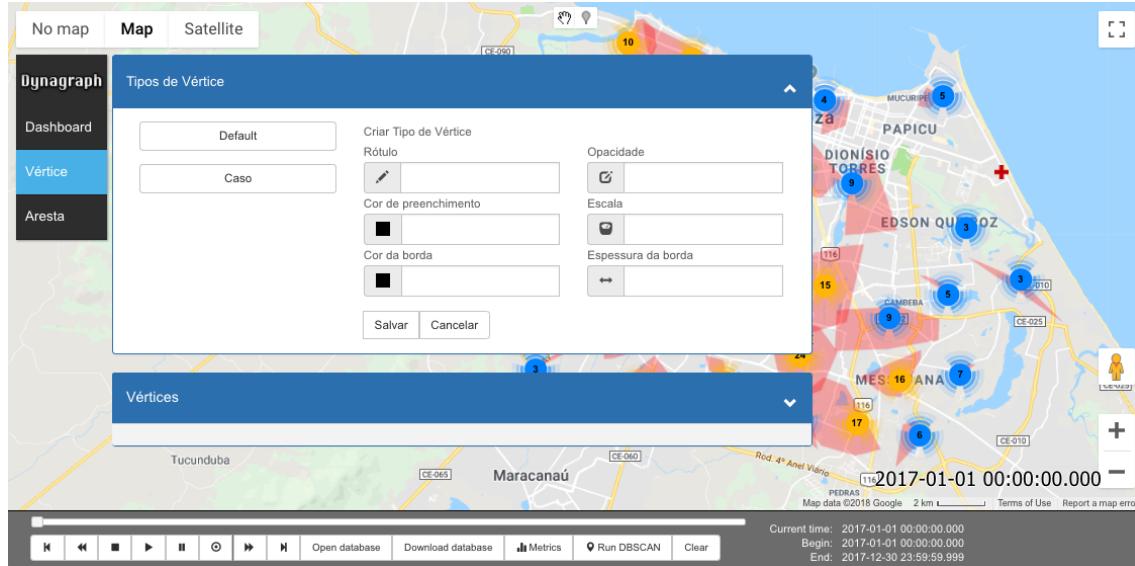


Fonte: Elaborado pelo autor

Para criar um novo vértice deve-se selecionar o submenu relacionado a vértices como

mostra a figura 4.

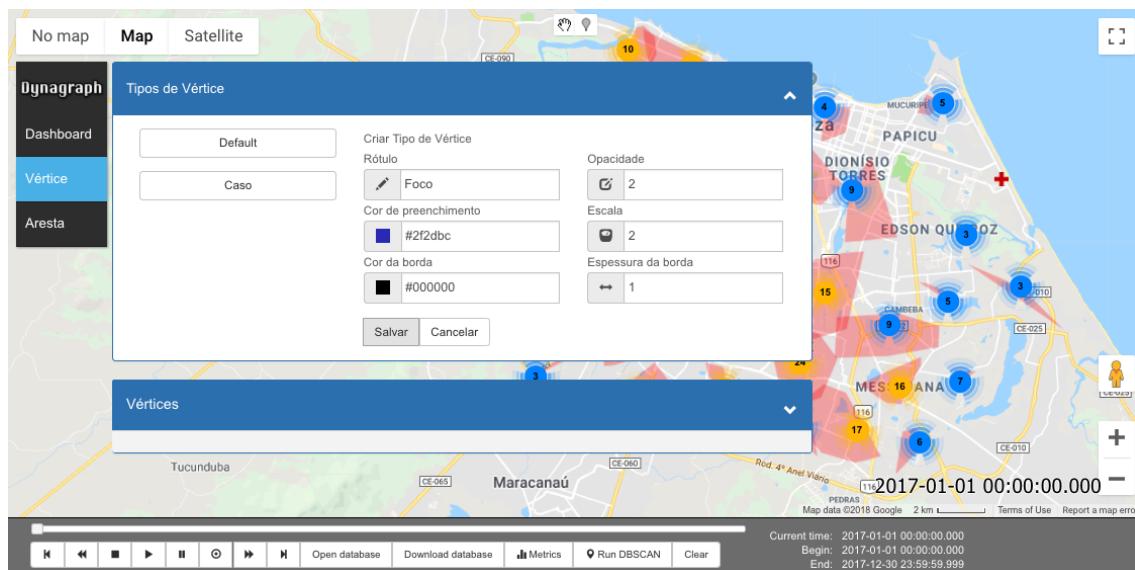
**Figura 4 – Editor de Características: criação de um vértice**



Fonte: Elaborado pelo autor

A figura 5 exibe os tipos de dados permitidos na criação ou edição de um vértice.

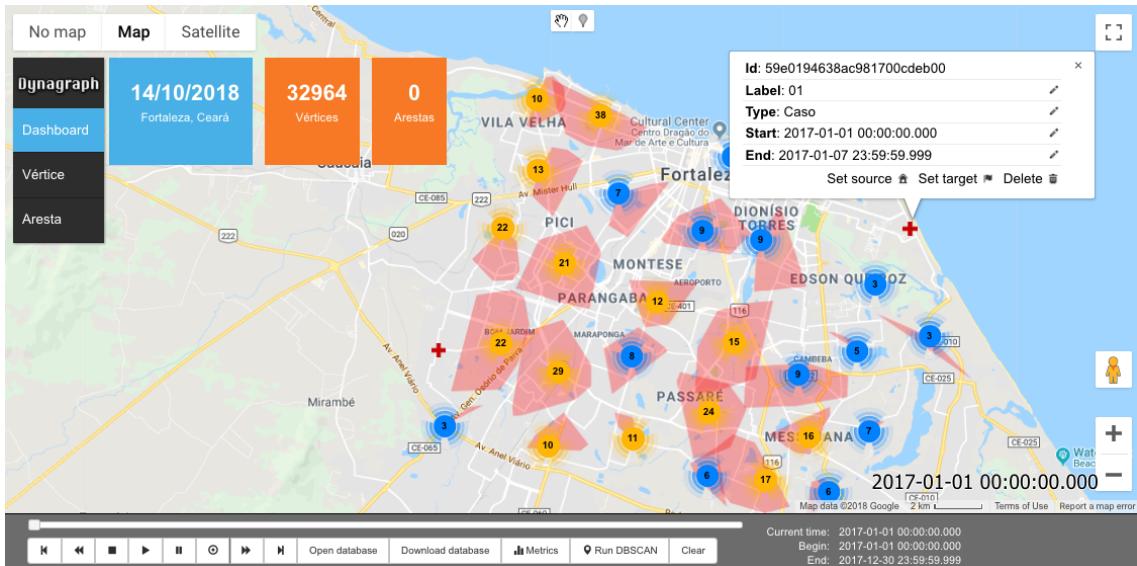
**Figura 5 – Editor de Características: criação de um vértice - tipos de dados**



Fonte: Elaborado pelo autor

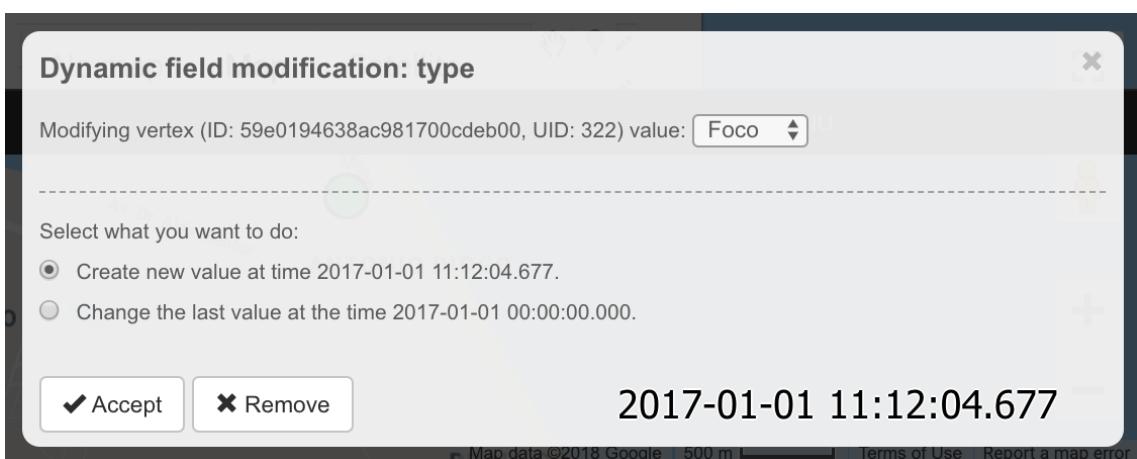
Após criar um novo tipo de vértice podemos usá-lo editando um dado ponto no mapa, como mostram as figuras 6 e 7. A figura 8 exibe o resultado esperado.

**Figura 6 – Editor de Características: edição de um vértice - parte 1**



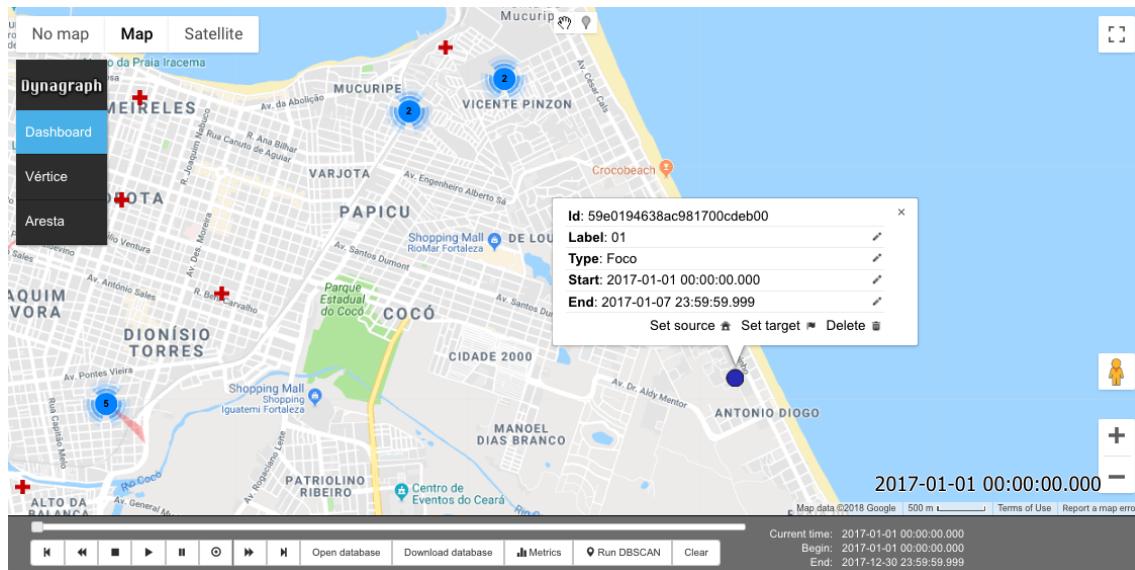
Fonte: Elaborado pelo autor

**Figura 7 – Editor de Características: edição de um vértice - parte 2**



Fonte: Elaborado pelo autor

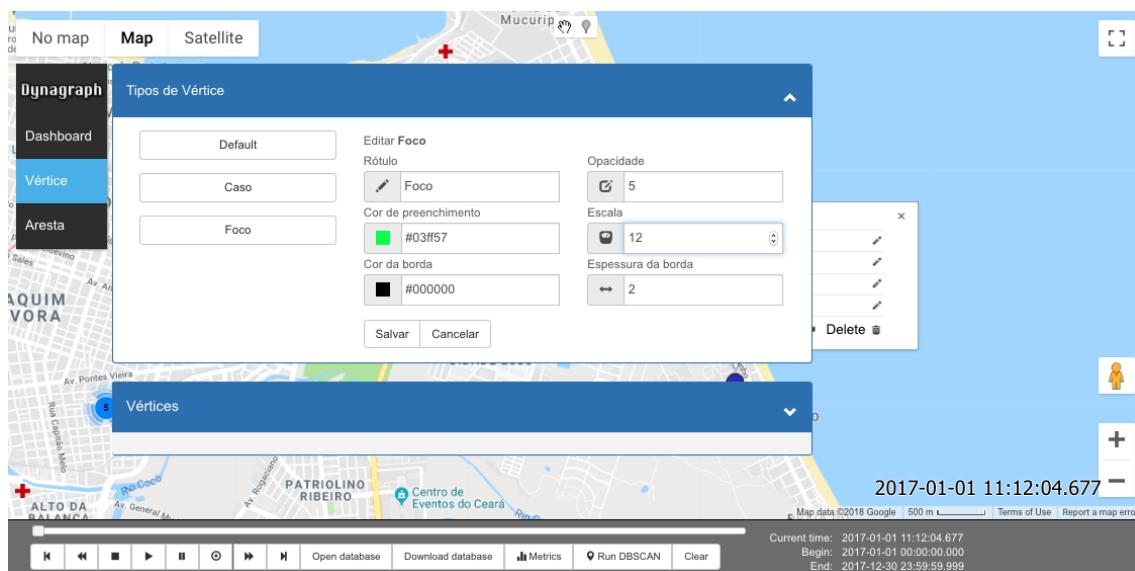
**Figura 8 – Editor de Características: edição de um vértice - parte 3**



Fonte: Elaborado pelo autor

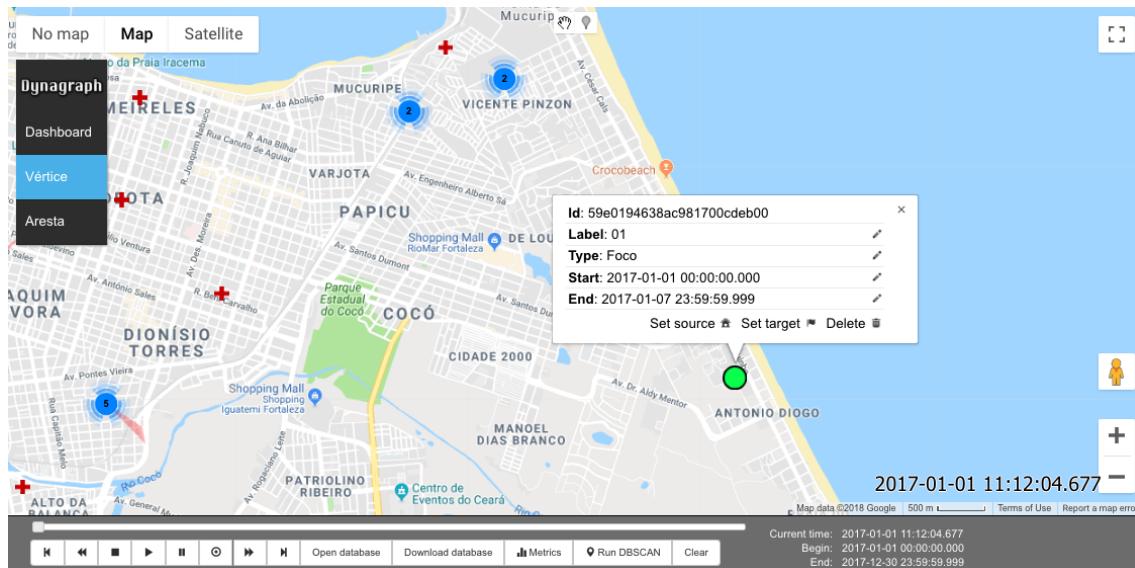
Para editar um tipo de vértice basta selecioná-lo a partir da lista de tipos de vértices e então aplicar as mudanças, como mostra a figura 9 e o resultado na figura 10.

**Figura 9 – Editor de Características: edição de um tipo de vértice - parte 1**



Fonte: Elaborado pelo autor

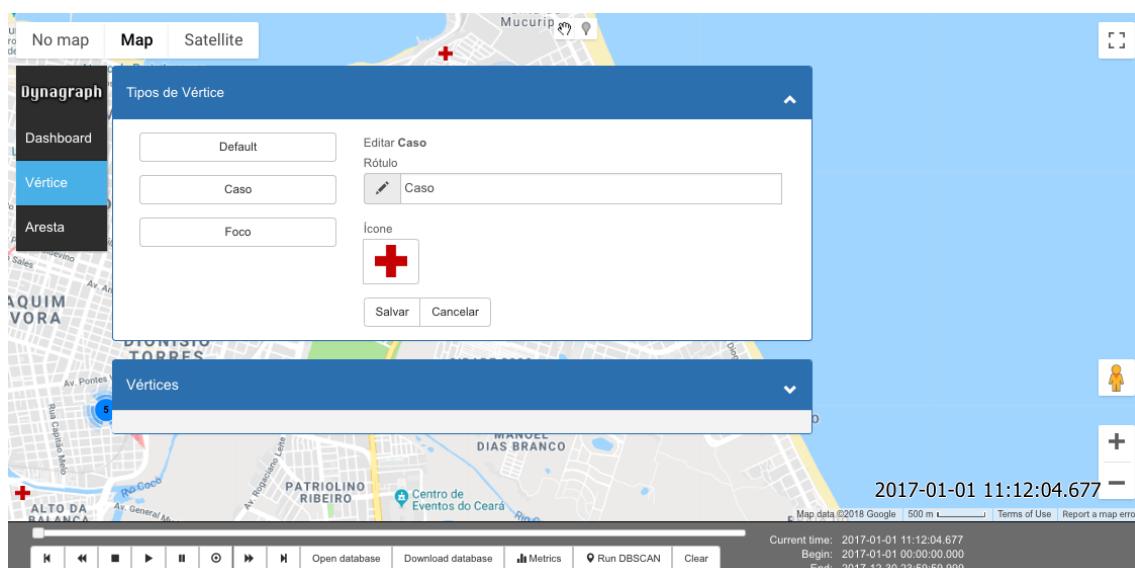
**Figura 10 – Editor de Características: edição de um tipo de vértice - parte 2**



Fonte: Elaborado pelo autor

Outra opção de edição de um vértice é dado na figura 11. Nesta opção podemos usar uma imagem pronta.

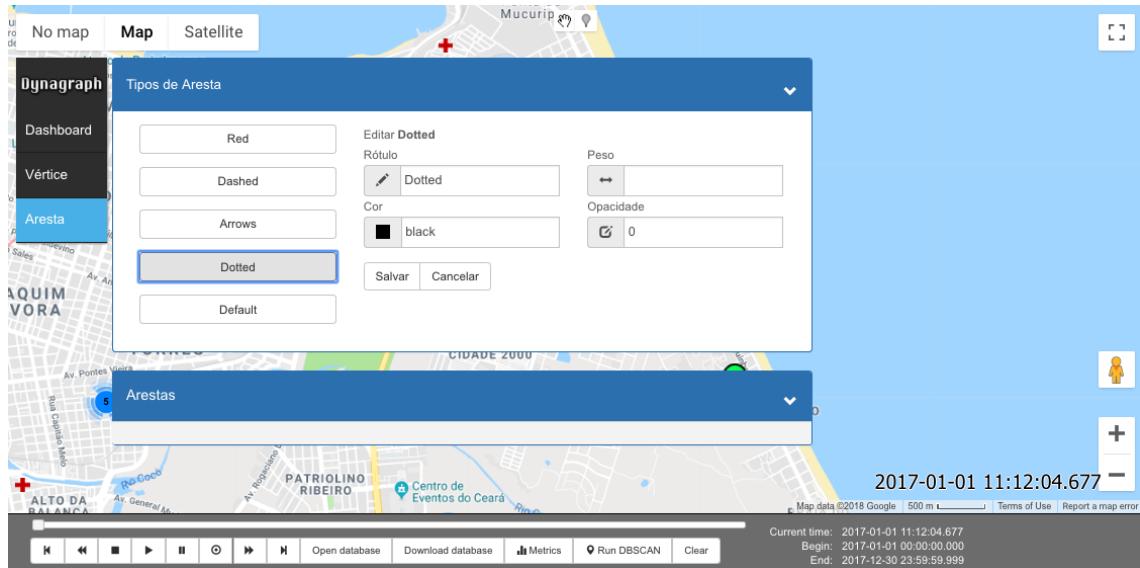
**Figura 11 – Editor de Características: Tipo de vértice**



Fonte: Elaborado pelo autor

O Editor de Características permite também a criação e edição de arestas (figura 12), mas essa funcionalidade não foi necessária nesta pesquisa.

**Figura 12 – Editor de Características: Aresta**

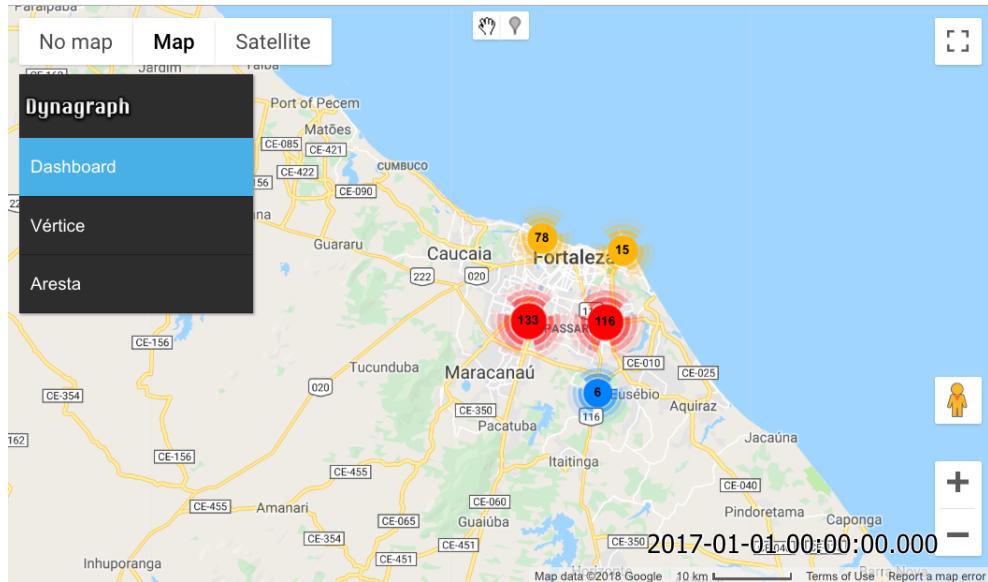


Fonte: Elaborado pelo autor

#### 2.4.3 Visualização dos grupos formados

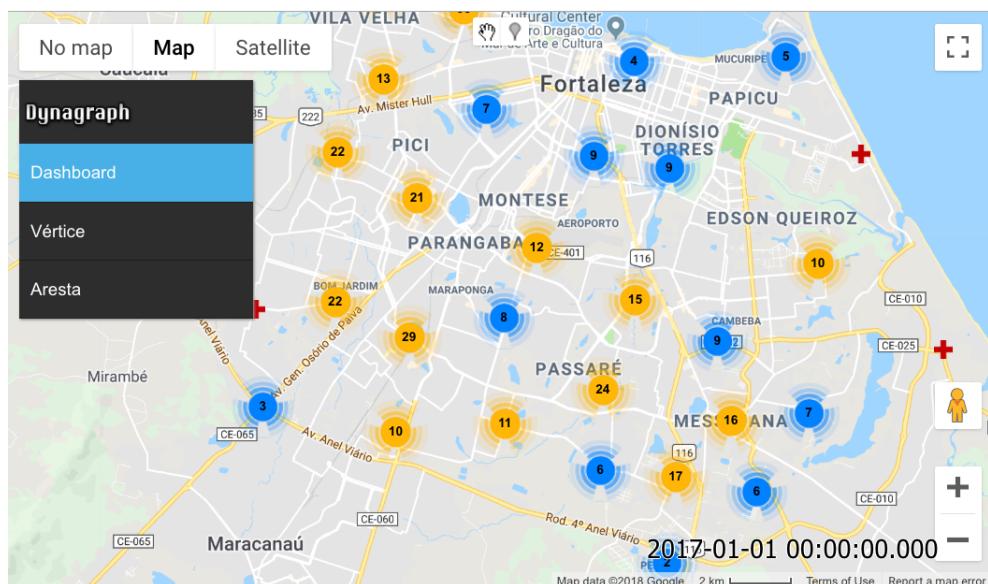
Foram usados dois algoritmos para permitir a visualização dos grupos formados em um dado tempo. O primeiro é uma biblioteca da API do Google chamada MarkerClusterer (MAPS, 2018) que cria e gerencia grupos de acordo com o nível de zoom para grandes quantidades de pontos. Essa biblioteca é combinada com a API Javascript do Google Maps para agrupar os pontos por proximidade em grupos e simplificar a exibição dos pontos no mapa. De acordo com o nível de zoom os grupos são formados com cores e tamanhos diferentes, como mostra as figuras 13 e 14.

**Figura 13 – Grupos - API Google Maps parte 1**



Fonte: Elaborado pelo autor

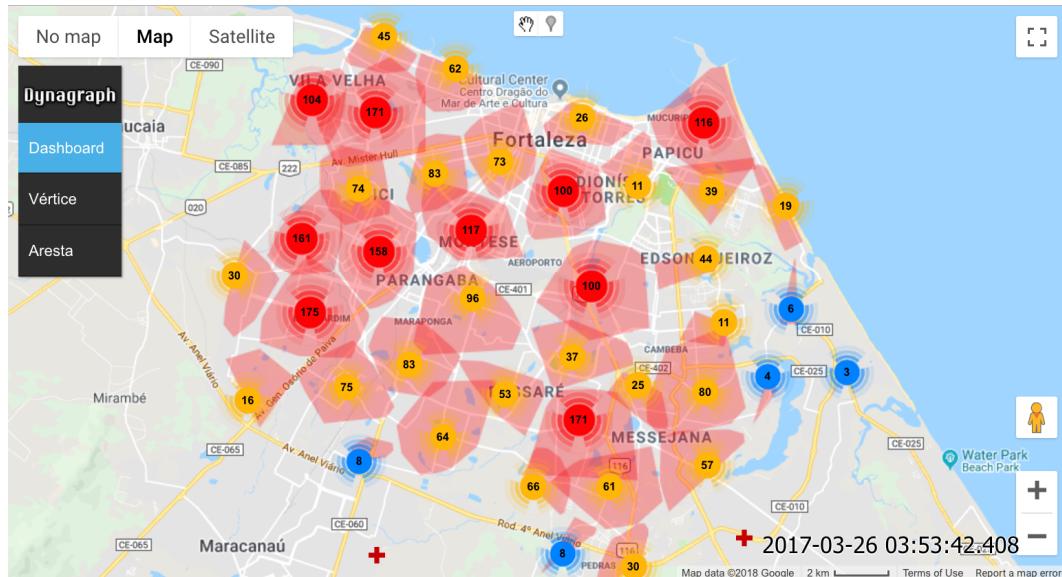
**Figura 14 – Grupos - API Google Maps parte 2**



Fonte: Elaborado pelo autor

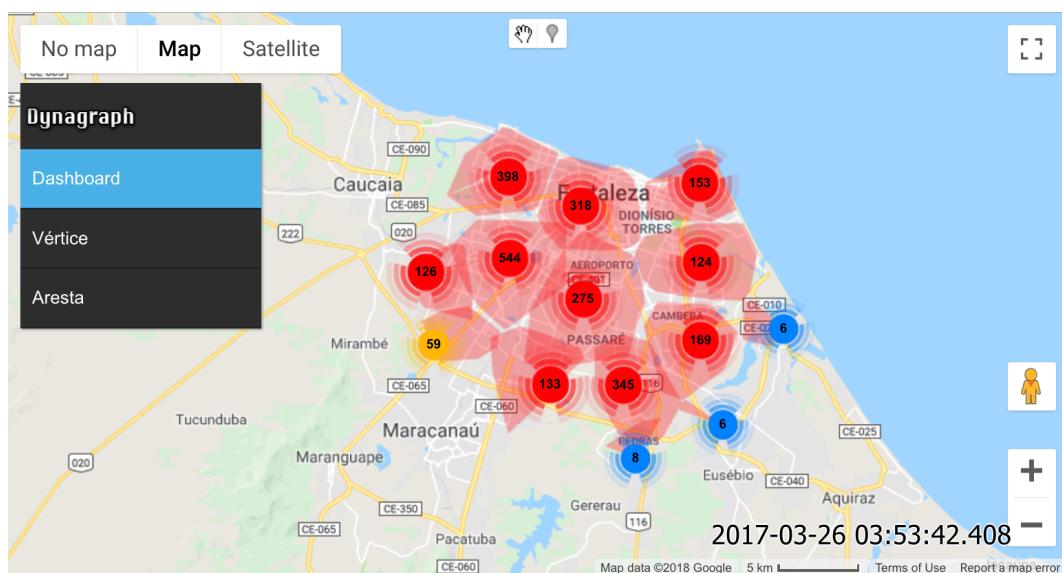
O segundo algoritmo é conhecido como Convex Hull (GRAHAM, 1972). Também conhecido como fecho convexo onde, dado um conjunto de pontos em um espaço, o problema consiste em encontrar o menor número de pontos que gerem um polígono convexo no qual abranja todos os outros pontos. As figuras 15 e 16 são exemplos do convex hull, onde cada polígono representa o grupo de casos de dengue naquela região delimitada. Dos vários pontos, somente os pontos mais externos e que formam o menor polígono que englobam todos os outros pontos.

**Figura 15 – Exemplo 1 de Convex Hull**



Fonte: Elaborado pelo autor

**Figura 16 – Exemplo 2 de Convex Hull**



Fonte: Elaborado pelo autor

## 2.5 TRABALHOS RELACIONADOS

Como há uma carência de estudos relacionando os assuntos abordados: agrupamento, previsão em dados dinâmicos espaço-temporais, grafos dinâmicos e sistemas web de forma integrada, foi necessário dividir o problema de agrupamentos e previsões dinâmicos em três etapas:

- Estrutura de dados em grafos dinâmicos
- Modelos de previsão espaço-temporais
- Algoritmos de agrupamentos dinâmicos

A pesquisa aborda estrutura de dados em grafos dinâmicos usando passos já descritos na literatura, principalmente o modelo Dynagraph (CALIXTO; NEGREIROS, 2013), que é baseado na primeira proposta em (CALIXTO; NEGREIROS, 2012), onde o Dynagraph usa sequências temporais para vértices, arestas, características modificáveis dos vértices e arestas e o relacionamento entre suas características. Com isso, é formado um grafo com as informações necessárias para qualquer instante no tempo. O Dynagraph é capaz de visualizar o comportamento do grafo ao longo de um período de tempo, e editá-lo.

A ideia central de (KIM; ANDERSON, 2012) é modelar uma rede dinâmica como digrafos orientados ao tempo (*time-ordered graph*), que é gerada através da ligação de instantes temporais com arestas direcionadas que unem cada nó ao seu sucessor no tempo. Com isso, transformar uma rede dinâmica em um grafo maior, mas facilmente analisável. Isto permite não só a utilização dos algoritmos desenvolvidos para grafos estáticos, mas também para melhor definir métricas para grafos dinâmicos. Segundo (KIM; ANDERSON, 2012) um sistema de grafos dinâmicos é um objeto de representação visual que pode descrever melhor o comportamento dinâmico de objetos relacionados a eventos dinâmicos e introduzir novas formas de enxergar ou descrever a evolução de eventos dinâmicos na natureza.

(KOSTAKOS, 2009) considera a estrutura de grafos temporais como grafos estáticos, no entanto avança sobre as métricas introduzindo conceitos como disponibilidade temporal, proximidade temporal e geodésica, e estuda os seus grafos sobre redes reais.

Segundo (ESTER *et al.*, 1996), o algoritmo DBScan (*Density-Based Spatial Clustering of Applications With Noise*) calcula a densidade de uma região contando quantos pontos existem em uma determinada área seguindo uma determinada métrica. Ele permite a redução de pontos não pertencentes a nenhum padrão, assim como possibilita a formação de grupos de diferentes formas. Seu objetivo principal é dividir os pontos em grupos através da densidade de

cada região.

(LAHIRI; BERGER-WOLF, 2007) apresentam um algoritmo de predição em redes temporais, e que usa a ideia de que certas interações sinalizam a ocorrência de outros em algum momento no futuro. Através de análises estatísticas o algoritmo mede o atraso entre as interações, e com isso pode-se prever quando certas interações vão ocorrer com base em observações passadas e atuais. Propõe-se a utilização de subgrafos frequentes e discute como identificar subgrafos que são persistidos em redes temporais. (LAHIRI; BERGER-WOLF, 2008) em seguida propõe um novo problema de mineração de dados para redes dinâmicas: detecção de todos os padrões de interação que ocorrem em intervalos de tempo regulares.

### **3 MODELO DE AGRUPAMENTO E PREVISÃO EM REDES DINÂMICAS**

#### **4 AVALIAÇÃO DO MÉTODO PROPOSTO**

## 5 CONCLUSÕES E TRABALHOS FUTUROS

### 5.1 CONSIDERAÇÕES FINAIS

Criação de um método para resolver o Problema de Agrupamento em Grafos Dinâmicos e previsão de evolução destes agrupamentos. A expectativa é de que ao final desta pesquisa, tenha-se uma extensão do Dynagraph para visualização de agrupamentos dinâmicos e previsão dinâmica. Utilizar a ferramenta para outros tipos de doenças como: Chikungunya e Zika Vírus. Finalmente, espera-se que o produto final e os resultados obtidos possibilitem a previsão e prevenção de novos casos de dengue para um combate efetivo à doença.

### 5.2 LIMITAÇÕES

Dentre as dificuldades que podem interferir na execução deste projeto de pesquisa, as seguintes podem ser citadas: 1. A escassez de estudos relacionando os assuntos abordados: agrupamento, previsão em dados dinâmicos espaço-temporais, grafos dinâmicos e sistemas web de forma integrada; 2. Obtenção das informações dos focos e casos de dengue geolocalizadas e tempos das ocorrências. A extração dos dados semanais requer um processo manual em (SIMDA, 2018), pois é necessário o usuário selecionar o ano e a semana correspondente; 3. Visualização dos agrupamentos dinâmicos. Para contornar as dificuldades apresentadas pretende-se: 1. Automatizar a forma de obtenção dos dados; 2. Utilizar o software Dynagraph como ambiente de suporte à validação e interação com os resultados dos métodos de agrupamento espaço-temporal utilizados.

### 5.3 TRABALHOS FUTUROS

Em estudos futuros, pretende-se executar o algoritmo em paralelo, a fim de melhorar o desempenho, pois grandes bancos de dados necessitam de alto poder computacional. Além disso, heurísticas mais performáticas podem ser encontradas para determinar os parâmetros de entrada Eps e MinPts.

## REFERÊNCIAS

- ANGULARJS. **What Is AngularJS?** 2010. Acessado em Outubro. 22, 2018. Disponível em: <<https://docs.angularjs.org/guide/introduction>>.
- BERKHIN, P. **Survey Of Clustering Data Mining Techniques.** San Jose, CA, 2002. Disponível em: <<http://citeseer.ist.psu.edu/berkhin02survey.html>>.
- BIRANT, D.; KUT, A. St-dbscan: An algorithm for clustering spatial-temporal data. **Data Knowl. Eng.**, v. 60, p. 208–221, 2007.
- CALIXTO, A.; NEGREIROS, M. DYNAGRAPH: Um Modelo de Edição e Representação de Grafos Dinâmicos. 1 ed. CLAIO/SBPO, p. 8, 2012.
- CALIXTO, A.; NEGREIROS, M. **DYNAGRAPH: Um Modelo de Edição e Representação de Grafos Dinâmicos.** Dissertação (Mestrado) — Mestrado Profissional em Computação Aplicada (MPCOMP), Universidade Estadual do Ceará, 2013.
- ECMA-262. **ECMAScript® 2015 Language Specification.** 2015. Acessado em Outubro. 20, 2018. Disponível em: <<https://www.ecma-international.org/ecma-262/6.0/>>.
- ECMA-404. **The JSON Data Interchange Standard.** 2018. Acessado em Outubro. 20, 2018. Disponível em: <<https://www.json.org/>>.
- ESTER, M.; KRIEGEL, H.-P.; SANDER, J.; XU, X. A density-based algorithm for discovering clusters in large spatial databases with noise. **KDD**, v. 96, n. 34, p. 226–231, 1996.
- ESTER, M.; KRIEGEL, H.-P.; SANDER, J.; WIMMER, M.; XU, X. Incremental clustering for mining in a data warehousing environment. In: **Proceedings of the 24rd International Conference on Very Large Data Bases.** San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1998. (VLDB '98), p. 323–333. ISBN 1-55860-566-5. Disponível em: <<http://dl.acm.org/citation.cfm?id=645924.671201>>.
- FACHIN, O. **Fundamentos de metodologia.** [S.l.]: 3a Edição, Editora Saraiva, 2001.
- FAGHMOUS, J. H.; KUMAR, V. **Spatio-Temporal Data Mining for Climate Data: Advances, Challenges, and Opportunities.** [S.l.: s.n.], 2013. 83-116 p.
- GRAHAM, R. L. An efficient algorithm for determining the convex hull of a finite planar set. **Inf. Process. Lett.**, v. 1, n. 4, p. 132–133, 1972.
- GUHA, S.; RASTOGI, R.; SHIM, K. Cure: An efficient clustering algorithm for large databases. **SIGMOD Rec.**, ACM, New York, NY, USA, v. 27, n. 2, p. 73–84, jun. 1998. ISSN 0163-5808. Disponível em: <<http://doi.acm.org/10.1145/276305.276312>>.
- HOLME, P. Modern temporal network theory: a colloquium. **The European Physical Journal B**, v. 88, n. 9, p. 1, 2015.
- KAUFMANN, L.; ROUSSEEUW, P. J. **Finding Groups in Data: An Introduction to Cluster Analysis.** [S.l.]: John Wiley and Sons, 1990.
- KIM, H.; ANDERSON, R. Temporal node centrality in complex networks. 1 ed. **PHYSICAL REVIEW**, p. 8, 2012.
- KOSTAKOS, V. Temporal graphs. **Physica A**, p. 1007–1023, 2009.

- LAHIRI, M.; BERGER-WOLF, T. Structure prediction in temporal networks using frequent subgraphs. IEEE Symposium on Computational Intelligence and Data Mining, p. 35–42, 2007.
- LAHIRI, M.; BERGER-WOLF, T. Mining periodic behavior in dynamic social networks. Eighth IEEE International Conference on Data Mining, 2008.
- LIN, J.; VLACHOS, M.; KEOGH, E.; GUNOPULOS, D. Iterative incremental clustering of time series. In: BERTINO, E.; CHRISTODOULAKIS, S.; PLEXOUSAKIS, D.; CHRISTOPHIDES, V.; KOUBARAKIS, M.; BÖHM, K.; FERRARI, E. (Ed.). **Advances in Database Technology - EDBT 2004**. Berlin, Heidelberg: Springer Berlin Heidelberg, 2004. p. 106–122.
- MACQUEEN, J. Some methods for classification and analysis of multivariate observations. In: **In 5-th Berkeley Symposium on Mathematical Statistics and Probability**. [S.l.: s.n.], 1967. p. 281–297.
- MAPS, G. **Marker Clustering**. 2018. Acessado em Outubro. 14, 2018. Disponível em: <<https://developers.google.com/maps/documentation/javascript/marker-clustering>>.
- MITSA, T. **Temporal Data Mining**. 1st. ed. [S.l.]: Chapman & Hall/CRC, 2010. ISBN 1420089765, 9781420089769.
- SHEIKHOLESLAMI, G.; CHATTERJEE, S.; ZHANG, A. Wavecluster: A multi-resolution clustering approach for very large spatial databases. In: **Proceedings of the 24rd International Conference on Very Large Data Bases**. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1998. (VLDB '98), p. 428–439. ISBN 1-55860-566-5. Disponível em: <<http://dl.acm.org/citation.cfm?id=645924.671342>>.
- SIMDA. **Sistema de Monitoramento Diário de Agravos**. 2018. Acessado em Julho. 24, 2018. Disponível em: <<http://tc1.sms.fortaleza.ce.gov.br/simda/index/>>.
- W3SCHOOLS. **CSS Introduction**. 2018. Acessado em Outubro. 20, 2018. Disponível em: <[https://www.w3schools.com/css/css\\_intro.asp](https://www.w3schools.com/css/css_intro.asp)>.
- W3SCHOOLS. **HTML Introduction**. 2018. Acessado em Outubro. 20, 2018. Disponível em: <[https://www.w3schools.com/html/html\\_intro.asp](https://www.w3schools.com/html/html_intro.asp)>.
- WANG, W.; YANG, J.; MUNTZ, R. R. Sting: A statistical information grid approach to spatial data mining. In: **Proceedings of the 23rd International Conference on Very Large Data Bases**. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1997. (VLDB '97), p. 186–195. ISBN 1-55860-470-7. Disponível em: <<http://dl.acm.org/citation.cfm?id=645923.758369>>.
- WU, X.; KUMAR, V.; QUINLAN, J. R.; GHOSH, J.; YANG, Q.; MOTODA, H.; MCLACHLAN, G. J.; NG, A.; LIU, B.; YU, P. S.; ZHOU, Z.-H.; STEINBACH, M.; HAND, D. J.; STEINBERG, D. Top 10 algorithms in data mining. **Knowledge and Information Systems**, v. 14, n. 1, p. 1–37, Jan 2008.
- YIP, A. M.; DING, C.; CHAN, T. F. Dynamic cluster formation using level set methods. In: **Proceedings of the 9th Pacific-Asia Conference on Advances in Knowledge Discovery and Data Mining**. Berlin, Heidelberg: Springer-Verlag, 2005. (PAKDD'05), p. 388–398. ISBN 3-540-26076-5, 978-3-540-26076-9. Disponível em: <[http://dx.doi.org/10.1007/11430919\\_46](http://dx.doi.org/10.1007/11430919_46)>.
- ZHANG, T.; RAMAKRISHNAN, R.; LIVNY, M. Birch: An efficient data clustering method for very large databases. **SIGMOD Rec.**, ACM, New York, NY, USA, v. 25, n. 2, p. 103–114, jun. 1996. ISSN 0163-5808. Disponível em: <<http://doi.acm.org/10.1145/235968.233324>>.

## **APÊNDICES**

## APÊNDICE A – Tecnologias

A arquitetura do software foi desenvolvida em três partes: servidor, banco de dados e cliente. Tanto o lado cliente (*frontend*) como o lado servidor(*backend*) utilizam a linguagem Javascript. Na parte do servidor, ou *server-side*, foram utilizados NodeJS, Express e Mongoose. No lado do cliente foram utilizadas as seguintes tecnologias: HTML, CSS, Javascript (EcmaScript 6), AngularJS e o Dynagraph. O banco de dados MongoDB foi utilizado como suporte a leitura inicial dos dados.

### A.1 SERVIDOR

O *backend* foi utilizado para manipular o banco de dados e transformar as informações na estrutura de dados do Dynagraph para permitir sua análise no *frontend*. O Node.js é uma plataforma de desenvolvimento para executar código Javascript no *server side* sobre o motor JavaScript do Google Chrome para facilmente construir aplicações de rede rápidas e escaláveis. Uma das maiores vantagens do Node.js é que ele usa um modelo de I/O direcionada a evento não bloqueante que o torna leve e eficiente, ideal para aplicações em tempo real com troca intensa de dados através de dispositivos distribuídos.

O Express.js é um framework JavaScript que facilita a criação de aplicativos web utilizando o Node.js e fornece um conjunto robusto de recursos para aplicativos web e móvel. Com vários métodos utilitários HTTP e *middleware* disponíveis para criar uma API robusta, rápida e fácil.

Mongoose é uma biblioteca do Nodejs que proporciona uma solução baseada em esquemas para modelar os dados da sua aplicação. Ele possui sistema de conversão de tipos, validação, criação de consultas e *hooks* para lógica de negócios. Mongoose fornece um mapeamento de objetos do MongoDB similar ao ORM (Object Relational Mapping), ou ODM (Object Data Mapping) no caso do Mongoose. Isso significa que o Mongoose traduz os dados do banco de dados para objetos JavaScript para que possam ser utilizados por sua aplicação.

### A.2 BANCO DE DADOS

MongoDB é um dos mais populares de banco de dados NoSQL. O mesmo é open-source e escrito em C++. MongoDB é um banco de dados orientado a documentos que armazena dados em documentos JSON com esquema dinâmico. Isso significa que você pode armazenar

seus registros sem se preocupar com a estrutura de dados, como o número de campos ou tipos de campos para armazenar valores. Os documentos do MongoDB são semelhantes aos objetos JSON.

### A.3 CLIENTE

O HTML é uma Linguagem de Marcação de Hipertexto (*Hypertext Markup Language*) utilizada para criar páginas web. Ele descreve a estrutura das páginas web usando marcadores (*markup*). Os seus elementos são construídos em blocos de páginas HTML e são representados por *tags*. Tags de HTML marcam partes de conteúdo, como "cabeçalho"(*heading*), "parágrafo"(*paragraph*), "tabela"(*table*) e assim por diante. Os navegadores não exibem as tags HTML, mas as usam para renderizar o conteúdo da página (W3SCHOOLS, 2018).

Junto ao HTML temos o CSS (*Cascading Style Sheets*), que é uma linguagem que descreve o estilo (*style*) de um documento HTML. Além disso, ele descreve como os elementos HTML devem ser exibidos. Uma das vantagens em utilizar o CSS é controlar o *layout* de múltiplas páginas web ao mesmo tempo. Ele é projetado para permitir a separação de apresentação e conteúdo, incluindo layout, cores e fontes. Essa separação pode melhorar a acessibilidade de conteúdo, fornecer mais flexibilidade e controle na especificação de características de apresentação e reduzir a complexidade e a repetição no conteúdo estrutural (W3SCHOOLS, 2018).

O JavaScript é uma linguagem de programação interpretada, frequentemente abreviado como JS, de alto nível, caracterizada como dinâmica, fracamente tipada, baseada em protótipos, de tipagem fraca. O ECMAScript é uma linguagem de programação baseada em scripts, padronizada pela Ecma International na especificação ECMA-262. A linguagem é bastante usada em tecnologias para Internet, sendo esta base para a criação do JavaScript. A versão utilizada do ECMAScript para esta pesquisa foi a ECMAScript 6 ou ECMAScript2015 (ECMA-262, 2015).

Outro recurso associado ao *frontend* é o JSON (*JavaScript Object Notation*), que é um formato “leve” para troca de informações. Ele é baseado em um subconjunto da linguagem de programação JavaScript (ECMA-404, 2018) e também em dois tipos de estruturas: Coleção de pares chave/valor e lista ordenada de valores. As coleções de chave e valor são equivalentes a record, struct, dictionary, hash e map nas linguagens. As listas são equivalentes a array, vector ou list. Um objeto é um conjunto não ordenado de pares nome/valor. Um objeto começa com (chave esquerda) e termina com (chave direita). Cada nome é seguido por : (dois pontos) e os

pares nome/valor são separados por , (vírgula).

O Editor de Características foi incorporado ao Dynagraph após a sua conclusão, e o seu desenvolvimento foi realizado utilizando principalmente o framework AngularJS (ANGULARJS, 2010). Este permite construir aplicações web dinâmicas e estender a sintaxe do HTML, e com isso simplifica o código e permite uma estrutura componentizada. A ligação de dados e a injeção de dependência do AngularJS eliminam grande parte do código que seria necessário escrever. Seu principal padrão arquitetural é o MVC (*Model View Controller*):

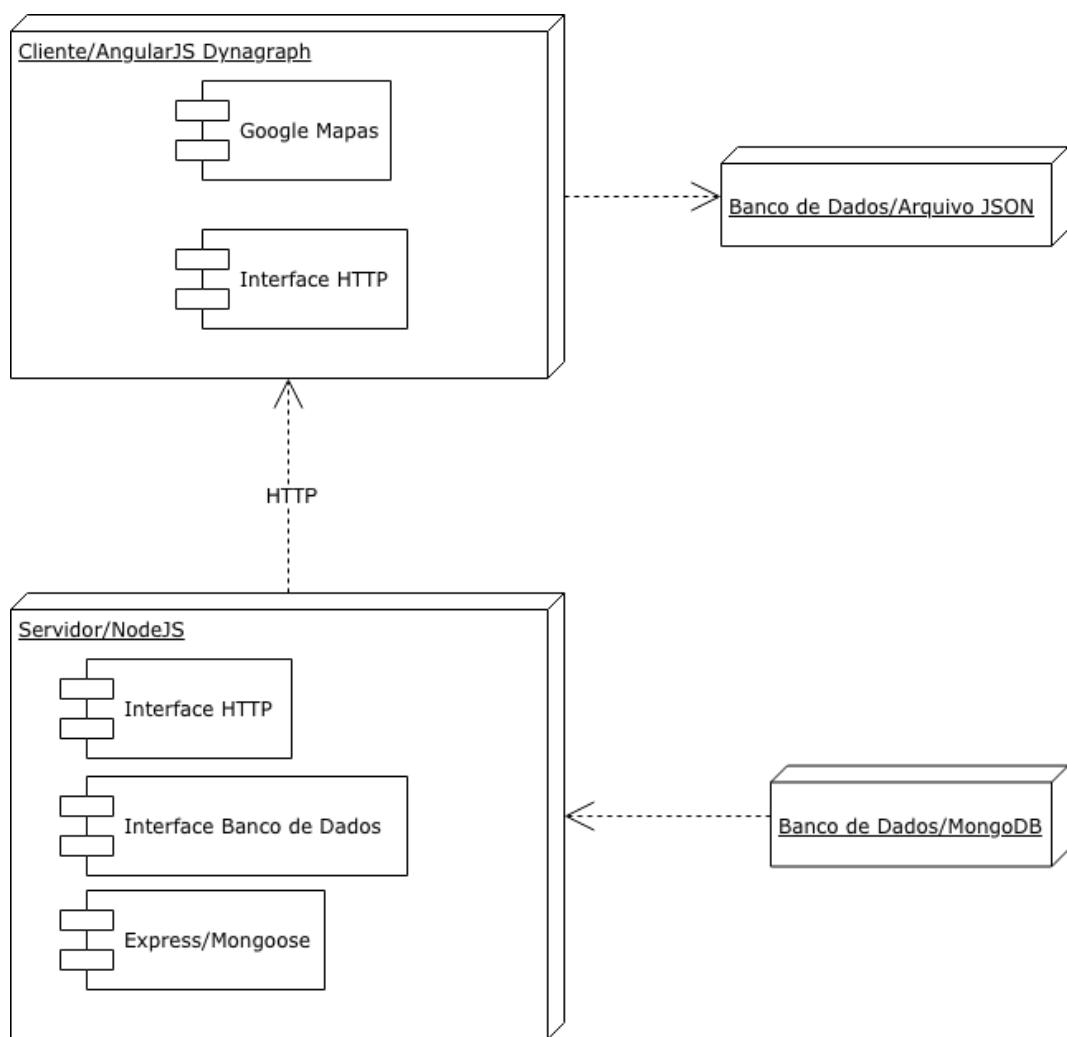
- Modelo (*Model*): são as propriedades de um escopo; escopos são anexados ao DOM onde as propriedades do escopo são acessadas por meio de ligações.
- Visão (*View*): O *template* (HTML com ligações de dados) que é renderizado na Visão.
- Controlador (*Controller*): é a classe que contém a lógica de negócios por trás do aplicativo.

#### Dynagraph (HTML 5)

#### A.4 ARQUITETURA DO SOFTWARE

A princípio o MongoDB foi utilizado para armazenar os dados salvos do (SIMDA, 2018), a respeito da dengue e chikungunya. O cliente requisita os dados do servidor, que por sua vez processa as informações, seguindo a estrutura de dados do Dynagraph. Em seguida, esses dados processados são passados para a aplicação no cliente. Com os dados exibidos no browser podemos editá-los, gerar agrupamentos e finalmente salvar as informações em um arquivo no formato JSON.

**Figura 17 – Arquitetura do Software**



Fonte: Elaborado pelo autor