Proposta de Algoritmo por Crescimento Gradativo de Diâmetro para Detecção de k-*Flocks* em Dados de Trajetórias

Luan V. de Almeida, Vinicius E. C. Verdade, Daniel S. Kaster

[1]Departamento de Computação – Universidade Estadual de Londrina (UEL)
Caixa Postal 10.011 – CEP 86.057-970 – Londrina – PR – Brasil

luan.victor.almeida@uel.br,vinicius-verdade@hotmail.com,dskaster@uel.br

Abstract. *There is a growing demand for devices to collect and analyze information related to geolocation, including moving object comovement patterns' discovery. Among the most relevant comovement patterns is the Flock Pattern. A flock is a set of objects moving together defined by a moving disk of fixed diameter for a consecutive amount of time instants. Recently, a variation of this pattern was proposed, called the k_ϵ-Flocks pattern, which eliminates the need to define the distance parameter. The existing algorithm to detect k_ϵ-Flocks follows a top-down approach starting from a single flock candidate and performing successive subdivisions until finding k flocks with the smallest diameter possible. This work proposes a new bottom-up algorithm for the k_ϵ-Flocks problem based on iteratively clustering elements close in the space until reaching the k_ϵ-Flocks. The work describes the foundations of the solution, discusses its correctness, and presents the proposed algorithm. We expect the proposed algorithm outperforms the top-down one for many typical cases.*

Keywords: Comovement Patterns; Moving Object Trajectories; Flock Pattern.

Resumo. *Há uma demanda crescente por dispositivos para coletar e analisar informações relacionadas à geolocalização, incluindo a descoberta de padrões de comovimento de objetos móveis. Entre os padrões de comovimento mais relevantes está o padrão Flock. Um flock é um conjunto de objetos que se movem juntos, definidos por um disco móvel de diâmetro fixo, por um período consecutivo de instantes de tempo. Recentemente, foi proposta uma variação desse padrão, denominada padrão k_ϵ-Flocks, que dispensa a definição do parâmetro distância. O algoritmo existente para detectar k_ϵ-Flocks segue uma abordagem top-down, começando por um único flock candidato e realizando subdivisões sucessivas até encontrar k flocks com o menor diâmetro possível. Este trabalho propõe um novo algoritmo bottom-up para o problema k_ϵ-Flocks, baseado no agrupamento iterativo de elementos próximos no espaço até detectar os k_ϵ-Flocks. O trabalho descreve os fundamentos da solução, discute sua correção e apresenta o algoritmo proposto. Espera-se que o algoritmo proposto seja mais eficiente que o algoritmo top-down em muitos casos típicos.*

Palavras-chave: Padrões de comovimento; Trajetórias de objetos móveis; Padrão *Flock*.

1. Introdução

A presença crescente de sensores de posicionamento, notadamente GPS (*Global Positioning System*), tem potencializado a coleta e análise de dados de trajetórias de objetos móveis (*Moving Objects* – MOs). Uma categoria importante de análises concentra-se em objetos móveis que se movimentam próximos uns dos outros por instantes consecutivos de tempo, caracterizados como objetos móveis em *comovimento*. Os MOs podem ser pessoas, automóveis, animais ou qualquer outro dispositivo que tenha algum sensor de posicionamento instalado. Situações de comovimento incluem, por exemplo, uma carreata, animais migrando em conjunto, multidões em um protesto, etc., cujas suas detecções em um banco de dados de trajetórias podem ser de interesse para várias aplicações.

Os padrões de comovimento mais conhecidos na literatura são baseados em disco, de diâmetro fixo que engloba os objetos móveis [Vieira et al. 2009, Cao et al. 2016], ou em agrupamento por densidade [Wang et al. 2006]. Dentre os que utilizam discos para a análise dos agrupamentos, o padrão *flock* é um dos mais relevantes. O padrão *flock* [Benkert et al. 2008, Vieira et al. 2009, Cao et al. 2016, Sanches et al. 2018] é dado por um grupo de, pelo menos, μ MOs se movendo juntos, cuja distância máxima está limitada por um disco fixo de diâmetro ϵ, durante um intervalo de tempo de, no mínimo, δ instantes de tempo consecutivos. Todavia, o parâmetro que define um disco fixo é muito sensível e dependente do conjunto de dados, cuja definição imprecisa provoca mudanças impactantes para a análise do problema. Isto torna complicado para o usuário explorar um conjunto de dados utilizando este tipo de padrão sem que tenha pleno conhecimento da distribuição de distâncias dos pontos das trajetórias. Por isso, alguns autores propuseram novas visões acerca da mesma abordagem de disco, com intuito de deixar o parâmetro de distância menos rígido.

Em um trabalho anterior [Sanches et al. 2018], introduzimos a ideia de *k-Patterns*, em que, em vez de exigir do usuário um valor para o parâmetro de distância, solicita-se um número k de padrões a serem detectados, deixando a cargo do padrão ajustar internamente o parâmetro de distância para retornar o número desejado de respostas. Em particular, o padrão proposto k_ϵ-*Flocks* retorna k *flocks* de diâmetro mínimo, onde k é o número de *flocks* desejado e o diâmetro do disco é um parâmetro livre, sendo adequado automaticamente ao objetivo buscado. Neste mesmo trabalho, foi apresentado um algoritmo *top-down* para detectar k_ϵ-*Flocks*, baseado na subdivisão iterativa de *flocks* candidatos até que se retorne os *top*-k mais relevantes em janela de tempo w, com δ instantes de tempo. Observando o algoritmo *top-down* proposto, percebe-se que a visão de possuir inicialmente um agrupamento que englobe todos os MOs, realizando divisões sucessivas no agrupamento, não é a opção mais eficiente para muitos casos, pelo fato de que o resultado procurado é baseado em discos de diâmetro mínimo.

Este artigo propõe um novo algoritmo para o Padrão k_ϵ-*Flock*, baseado em agrupamento sucessivo de objetos móveis próximos no espaço até encontrar os k_ϵ-*Flocks*. Ilustrando essa ideia, em [Markovic et al. 2019] são explicitadas aplicações interessantes para problemas de análise e agrupamento de MOs, onde é necessário a verificação dos dados estatísticos coletados sobre o contexto da movimentação de pessoas em grupo, incluindo a modelagem do comportamento dos indivíduos. Em grandes multidões, constantemente novos indivíduos ou até grupos inteiros são inseridos em grupos que já estão em movimento, sendo essa análise um exemplo de muita importância na classificação dos

caminhos escolhidos pelos indivíduos durante o seu trajeto.

Para manter o parâmetro de distância livre, a proposta constrói, gradativamente, um conjunto de grafos, um para cada instante de tempo, e identifica *flocks* candidatos à medida que é formado um subgrafo completo com μ vértices em todos os instante de tempo. Os fundamentos do algoritmo são a ordenação parcial de arestas com base na distância entre pares de pontos, construção incremental de subgrafos, inserindo iterativamente arestas de menor distância e identificando cliques por meio de varreduras locais nos subgrafos. O trabalho introduz os fundamentos da solução, discute sua corretude e apresenta o algoritmo proposto. Espera-se que este novo algoritmo seja consideravelmente mais eficiente que o algoritmo *top-down* para muitos casos típicos.

Este artigo está organizado como segue. A Seção 2 apresenta as definições iniciais para entendimento da proposta, juntamente com o contexto das discussões presentes na literatura dos padrões já abordados. A Seção 3 descreve o problema e os conceitos abordados no algoritmo proposto. A Seção 4 apresenta a conclusão e as considerações finais desde trabalho.

2. Fundamentação Teórica

2.1. Descoberta de Padrões em Trajetórias de Objetos Móveis

Uma trajetória, de acordo com [Spaccapietra et al. 2008], é definida como sendo segmentos de caminho de objetos móveis, que representam a evolução espaço-temporal dos MOs explicitados no espaço como pontos, desde sua partida até sua chegada. As trajetórias são coletadas por dispositivos que possuem tecnologias embutidas para captura dos movimentos dos MO com precisão, como o GPS (*Global Positioning System*). Durante um período de tempo, os dispositivos coletam a posição do MO enquanto ele se movimenta, coletando de forma consecutiva várias posições e descrevendo sua trajetória. Com isso, é possível encontrar muitas aplicações utilizando as análises de dados provenientes do de trajetórias, desde análise de dados de viagens [Markovic et al. 2019] até evolução de ciclones [Pérez et al. 2015].

Dentro do estudo de trajetórias, existem áreas que estudam os padrões de movimento dos objetos móveis, principalmente quando estes se movimentam em conjunto, caracterizados como estando em comovimento. Para mineração dos padrões de comovimento, duas técnicas de agrupamento se destacam na literatura: o agrupamento baseado em disco [Cao et al. 2016, Vieira et al. 2009] e o agrupamento baseado em densidade [Wang et al. 2006, Ester et al. 1996]. O agrupamento em disco é caracterizado por criar um disco que englobe os MOs, mas possui a restrição espacial do disco e a dificuldade de escolher o seu diâmetro. Já agrupamento baseado em densidade propõe uma solução para a limitação do disco, agrupando os pontos baseando-se em *clustering* por densidade, tipicamente o DBSCAN (*Density-Based Spatial Clustering of Applications with Noise*) [Ester et al. 1996]. Porém, o DBSCAN ainda requer a definição de um raio de conectividade entre os pontos para classificar a região de vizinhança, além de ter tipicametne um alto custo computacional. O foco deste artigo é em padrões com agrupamento por disco. Nesta categoria, o padrão mais conhecido é o Padrão *Flock*, definido como um conjunto de, pelo menos, μ MOs se locomovendo em grupo por um número mínimo de δ instantes de tempo consecutivos, onde a

distância que os caracteriza como em comovimento é delimitada por um disco de diâmetro ϵ [Gudmundsson and van Kreveld 2006, Vieira et al. 2009].

Como já foi mencionado, o disco que engloba os MOs observados na definição de *flocks* precisa de um valor fixo de diâmetro ϵ, que é difícil de ser definido de forma precisa. A área que o disco engloba implica uma limitação espacial na obteção dos pontos, sabendo que os pontos precisam pertencer ao disco durante todos os instantes de tempo da janela de análise. Tal fator, dificulta o encontro dos padrões importantes, pois a noção que caracteriza os pontos como "estando juntos" pode ser inespecífica, caso não se conheça a fundo a distribuição de distâncias dos pontos das trajetórias. Por exemplo, para cada base de dados utilizada para testes, seriam necessários "chutes" sucessivos de diâmetro, que muitas vezes retornarão mais agrupamentos que o esperado, ou até mesmo nenhum, até se chegar em um valor adequado. Até mesmo em situações que a distribuição das distâncias varia muito, ou seja, quando o diâmetro do disco é muito alterado entre as medições dos MO, como por exemplo carros em uma estrada, variando constantemente sua valocidade e distância, dificultado a escolha do diâmetro ideal para o disco.

2.2. O Padrão k_ϵ-*Flocks*

Com intuito de facilitar a obtenção de *flocks* e tornar o parâmetro de distância menos rígido, vários padrões que flexibilizam o parâmetro de distância já foram propostos na literatura. Entre eles, o padrão *flock* com Grau de Liberdade, explicitado em [Cao et al. 2016], como extensão do padrão *flock*, mas com foco em agrupamentos em que as trajetórias são menos comportadas. Este padrão permite que as entidades possuam um grau de liberdade definido matematicamente e possam desconectar-se do grupo por breves instantes de tempo. De forma semelhante, o padrão Enxame [Li et al. 2010] propõe que o grupo pode se dispersar no decorrer dos instantes de tempo, desde que convirjam novamente em instantes de tempo posteriores. Já Evangelista et al. [Sanches et al. 2018] propôs o padrão k_ϵ-*Flock*, uma variação do padrão *Flock* que elimina a necessidade de se fornecer o parâmetro de distância. O padrão k_ϵ-*Flock* retorna os k *flocks* de diâmetro mínimo em cada janela temporal w de uma *stream* de dados de trajetórias, onde o diâmetro mínimo é dado pela Definição 1.

Definição 1 (Diâmetro Mínimo de um *Flock*) *O diâmetro mínimo de um flock $f_w(\mu, \epsilon)$, em uma janela w, é o menor valor $\epsilon' \in \mathbb{R}$ tal que $f_w(\mu, \epsilon') = f_w(\mu, \epsilon)$.*

O padrão k_ϵ-*Flock* não usa o diâmetro fixo, mas identifica *flocks* com o menor diâmetro possível que englobe a quantidade mínima de pontos μ para formar um *flock*. Assim, em vez do parâmetro de distância, o algoritmo requer um valor k, que é o número de *flocks* no conjunto resposta ao fim da busca. Formalmente, k_ϵ-*Flocks* são dados pela Definição 2.

Definição 2 (k_ϵ-*Flocks*) *k_ϵ-Flocks em respeito a uma janela temporal w e um número mínimo de trajetórias $\mu > 1$ ($\mu \in \mathbb{N}$), representado como k_ϵ-Flocks(μ, w), é o conjunto $\mathcal{F}_w^k$ contendo k ($k \in \mathbb{N}$) flocks tal que para cada flock $f_w(\mu, \epsilon) \notin \mathcal{F}_w^k$ temos que $\epsilon \geq \epsilon_k$, onde ϵ_k é o menor diâmetro do flock mais extenso em $\mathcal{F}_w^k$. Caso não exista flocks suficientes na janela, menos de k flocks são reportados.*

2.3. Algoritmo Top-Down para detecção de K_ϵ-*Flocks*

Foi proposto por Sanches et al. [Sanches et al. 2018] um algoritmo *top-down* para detecção de k_ϵ-*Flocks*. A ideia do algoritmo é identificar k_ϵ-*Flocks* a partir de subdi-

visões consecutivas de um *flock* maior que o de tamanho mínimo procurado, até que se encontre o *flock* minimal em cada instante de tempo da janela.

Inicialmente, tem-se um único *flock* que engloba todas as entidades daquele instante de tempo, que é subdividindo em pelo menos dois subflocks menores que o original, com um MO diferente entre eles. O *flock* poderá ser dividido em subflocks menores, dependendo da quantidade de pontos de borda presentes no disco e do μ escolhido para o teste. A Figura 1 ilustra em determinado instante de tempo, o *flock* mais extenso formado por um disco $\overline{f_w}$. O disco é subdividido em dois discos menores (f_w^1 e f_w^2), devido à presença de dois pontos de borda, formados pelas mesmas entidades do *flock* original, com exceção do ponto de borda utilizado na divisão. Na segunda etapa a direita, o mesmo processo é realizado, com a diferença de que o antigo disco f_w^1 agora é o *flock* mais extenso $\overline{f_w}$ em questão, contendo três pontos de borda e formando, por consequência, três novos *flocks* menores. O algoritmo então realiza todos os passos de divisão dos agrupamentos buscando retornar o conjunto resposta com os *flocks* de tamanho mínimo.

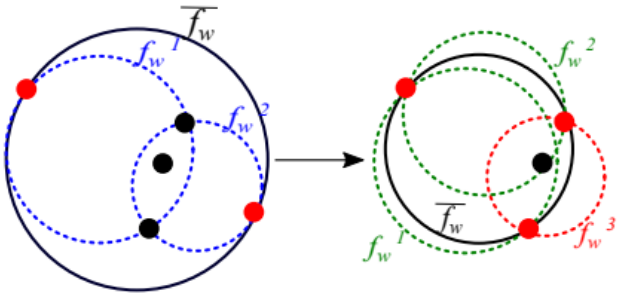

Figura 1. Exemplo da divisão em subflocks de dois discos $\overline{f_w}$ em duas iterações do algoritmo diferentes.(retirado de [Sanches et al. 2018])

Porém, mesmo funcional, o algoritmo pode ser bastante ineficiente, dependendo das características do dataset. Para instantes de tempo com poucas entidades e o μ desejado estando próximo a quantidade total de entidades, o *top-down* pode ser considerado eficiente. Porém, quanto menor o parâmetro μ e maior o número de entidades de cada instante de tempo, mais prejudicada fica o seu desempenho. Como casos em que o μ próximo ao valor total de entidades do instante de tempo são pouco comuns, uma nova abordagem do problema foi proposta desse trabalho.

3. Proposta de um algoritmo por agrupamento de pontos para detecção de k_ϵ-*Flocks*

Este trabalho propõe uma abordagem para a detecção de k_ϵ-*Flocks* em uma janela temporal, baseada em sucessivas aglutinações de trajetórias para formação de *flocks*. A ideia fundamental do algoritmo é baseada na observação de situações típicas, onde o número mínimo de MOs fornecido para indicar um *flock* (μ) é pequeno, se comparado ao total de MOs presentes na janela de tempo em análise. Nestes casos, o algoritmo *top-down* demanda um número grande de iterações de subdivisão de *flocks* candidatos até chegar a *subflocks* com o número desejado de entidades. A abordagem proposta, ao contrário, parte de MOs isolados e vai formando e aumentando grupos de MOs próximos no espaço até que se chegue em candidatos com o número mínimo de entidades para configurar um

flock. Sendo μ pequeno em relação ao número de MOs na janela, o número de iterações desta abordagem tende a ser muito menor que da abordagem *top-down*, possibilitando reduzir consideravelmente o tempo de execução do algoritmo.

O algoritmo proposto depende de duas coisas. A primeira é uma forma de agrupar gradativamente elementos mais próximos entre si. Isto pode ser resolvido por meio de um acesso a pares de elementos ordenado pela distância entre os elementos. A alternativa básica é calcular as distâncias entre todos pares de elementos em cada instante de tempo e estabelecer uma ordenação global compreendendo todos os instantes de tempo da janela. Mas, há outras possibilidades que permitem reduzir o custo desta tarefa. A segunda coisa é definir uma condição de parada que garanta a corretude do algoritmo. As seções a seguir discutem duas possibilidades, uma que não atende à necessidade do problema e uma segunda, que foi a escolhida neste trabalho.

3.1. Definição da condição de parada

A abordagem proposta tem semelhança com a ideia do algoritmo de agrupamento por densidade HDBSCAN. Assim, uma das possibilidades de condição de parada para a abordagem proposta seria utilizar a estratégia adotada pelo algoritmo de *clustering* HDBSCAN (*Hierarquical Density-Based Spatial Clustering of Applications with Noise*) [Vieira et al. 2009], que é baseada na formação de árvores espalhadas mínimas (*Minimal Spanning Tree* – MST) e na estabilidade de *clusters*. Entretanto, esta abordagem não se aplica ao problema, pois pode acontecer do valor de ε ser muito incrementado, aglutinando, de um nível para outro do dendograma, agrupamentos com muito mais que μ elementos. Portanto, seria necessário fazer um pós-processamento, utilizando a estratégia *top-down* para subdividir os *flocks* até chegar-se à resposta correta.

Contornando os problemas da alternativa anterior, a proposta desse trabalho é buscar k agrupamentos a partir da identificação de cliques de tamanho μ em cada instante de tempo da janela ω. Um clique é um subgrafo completo, formado por vértices conectados por arestas não direcionadas, onde cada vértice é conectado com todos os seus adjacentes. O problema de encontro de um subgrafo completo é conhecido na literatura como o Problema do Clique e tem algumas variações. As variações incluem encontrar o clique com maior número de vértices em dado instante de tempo, o clique de maior valor em um grafo com pesos ou, dado um valor μ, encontrar os subgrafos completos contendo μ vértices, como é utilizado neste trabalho. Um clique é explicitado pela Definição 3, como segue.

Definição 3 (Clique) *Dado um grafo G não direcionado, C é clique, tal que C seja um subconjunto de G e $G(C)$ seja completo, ou seja, todo vértice de C é adjacente a todos os outros vértices de C.*

Para inserir cliques na detecção de k_ϵ-*Flocks*, este trabalho propõe um algoritmo que encontre os cliques em subgrafos construídos gradativamente para cada instante de tempo, inserindo-se consecutivamente as arestas de menor peso global. Isto é, cada instante de tempo é formado inicialmente por um grafo vazio. A cada iteração, identifica-se a aresta de menor peso atual, que corresponde ao par de pontos mais próximos entre si, em qualquer instante de tempo, que ainda não foi conectado por uma aresta, e insere-se estes vértices e esta aresta no grafo do instante de tempo respectivo. Como as arestas são adicionadas de forma crescente, sempre selecionando a menor aresta atual, é garantido

que o primeiro agrupamento definido por um clique de tamanho μ corresponde ao *flock* de diâmetro mínimo com μ entidades na janela, o segundo clique é o segundo *flock* e assim sucessivamente. Esta estratégia permite definir uma condição de parada correta para o problema, interrompendo o algoritmo quando os mesmos k cliques forem encontrados em todos os grafos dos instantes de tempo da janela ω.

O principal desafio dessa abordagem é que, em sua versão clássica, o Problema do Clique é NP-Completo. Para contornar esse problema, a proposta consiste no encontro de cliques de forma incremental, à medida que novas arestas vão sendo adicionadas nos grafos, amortizando consideravelmente o custo da detecção de cliques. A próxima seção detalha o algoritmo proposto.

3.2. Descrição do algoritmo

O Algoritmo 1 apresenta o pseudocódigo da solução proposta. A partir de uma janela ω de um conjunto de trajetórias contendo as informações de MOs, é construída uma fila de prioridade Q, com todos os pares de pontos de cada instante de tempo, organizada pela distância entre eles. Sem otimizações que reduzam o número de cálculos de distância entre pontos, esta fila contém todos os pares que simulam as arestas de um grafo completo para cada instante de tempo. Também são inicializados o conjunto resposta F_k e os conjuntos G e C, para armazenar, respectivamente, os subgrafos e os conjuntos de cliques para os instantes de tempo da janela.

Em seguida, a partir de iterações consecutivas, é obtido o par de elementos mais próximos entre si, dentre os pares em Q. Os vértices incidentes nesta aresta são adicionados ao grafo $G[t_i]$, caso ainda não façam parte do grafo, bem como a aresta, onde t_i é o instante de tempo correspondente ao par. A Figura 2 mostra o estado dos subgrafos após uma sequência de quatro iterações.

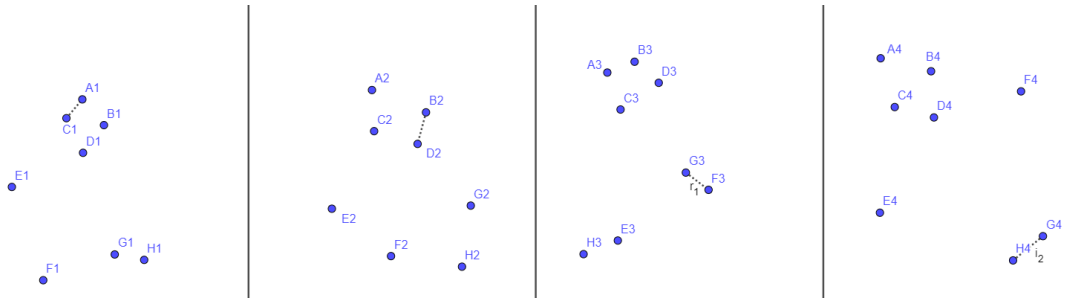

Figura 2. Representação da escolha das menor aresta (presente no instante de tempo 1) de uma base de dados de 4 instantes de tempo com 7 entidades cada. Menor aresta de cada instante de tempo representada por uma linha pontilhada.

Para cada aresta adicionada, é feita uma verificação se um ou mais novos cliques foram formados com a adição da aresta no grafo $G[t_i]$. Os cliques que forem sendo encontrados em determinado instante de tempo são lembrados, para que não haja recálculos desnecessários em iterações posteriores. Caso um ou mais novos cliques forem formados, verifica-se se o(s) mesmo(s) clique(s) já foi/foram encontrado(s) nos demais instantes de tempo, por meio de uma varredura localizada a partir dos vértices incidentes na aresta. Caso encontre em todos os instantes de tempo da janela grafos completos com esses

Algoritmo 1: Algoritmo Bottom-Up para encontro de agrupamentos de menor diâmetro mínimo

Entrada: ω: janela temporal contendo o conjunto de trajetórias $\mathcal{T}$
μ: número mínimo de entidades para formação de um *flock*
δ: número de instantes de tempo consecutivos para formar um *flock*
k: quantidade de flocks de retorno
Saída: F_k: Conjunto de k *flocks* de tamanho mínimo
início
  $Q \leftarrow buildPQueue(\omega)$
  $F_k \leftarrow \emptyset$
  para cada $t_i \in \omega$ **faça**
    $G[t_i] \leftarrow empty\_graph$
    $C[t_i] \leftarrow \emptyset$
  fim
  enquanto $|F_k| < k$ ***e*** $Q \neq \emptyset$ **faça**
    /* Escolhe a menor aresta entre todos os instantes de tempo da janela */
    $(u, v, t_i) \leftarrow Q.removeMin()$
    /* Insere os vértices e a aresta no grafo local do respectivo instante de tempo */
    $G[t_i].addVertex(u)$
    $G[t_i].addVertex(v)$
    $G[t_i].addEdge(u, v)$
    /* Verifica se a aresta gera novos cliques de tamanho μ em $G[t_i]$ */
    $C_{new} \leftarrow findNewCliques(G[t_i], (u, v), \mu)$
    para cada $c \in C_{new}$ **faça**
      /* Verifica se o novo clique já aparece nos demais instantes */
      $candidate \leftarrow true$
      para cada $t_j \in \{\omega - t_i\}$ **faça**
        se $c \notin C[t_j]$ **então** $candidate \leftarrow false$;
      fim
      /* Se aparece em todos os instantes, é um candidato a k_ϵ-*Flock* */
      se $candidate$ **então**
        /* Computa discos e o diâmetro do *flock* candidato */
        $D \leftarrow \emptyset$
        $\epsilon \leftarrow \infty$
        para cada $t_j \in \omega$ **faça**
          $D[t_j] \leftarrow computeMinDisk(c)$
          se $\epsilon > diameter(D[t_j])$ **então** $\epsilon \leftarrow diameter(D[t_j])$;
        fim
        /* Verifica se existem outros elementos que não são parte do clique, mas estão no diâmetro do *flock* em todos os instantes, para torná-lo maximal */
        $c' \leftarrow c$
        para cada $t_j \in \omega$ **faça**
          $c' \leftarrow c' \cap rangeQuery(w[t_j], disk[t_j].center, \epsilon/2)$
        fim
        /* Insere o candidato no conjunto resposta */
        $f_\omega \leftarrow buildFlock(c', disk)$
        $F_k \leftarrow F_k \cup f_\omega$
      fim
    fim
  fim
  /* Se o conjunto resposta ficou com mais de k respostas, remove os k'-ésimo *flocks* tais que $k' > k$ */
  se $|F_k| > k$ **então**
    $F_k.removeLargest(k)$
  fim
fim

vértices em comum, um candidato a k_ϵ-*Flock* foi detectado, como ilustra a Figura 3. Em seguida, para cada instante de tempo, o algoritmo computa os discos envolventes de tamanho mínimo que cobrem os pontos no clique. O diâmetro ϵ do k_ϵ-*Flock* candidato é o maior diâmetro dentre esses discos. Esta operação identifica o *Smallest Enclosing Circle*[1] que engloba todos os pontos do grafo completo para cada instante de tempo da janela.

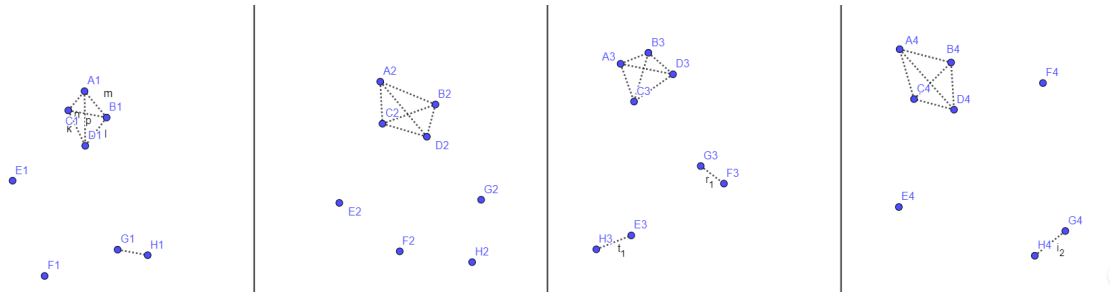

Figura 3. Exemplo de clique encontrado nos elementos {A, B, C, D} com μ = 4, k = 1, δ = 4.

Podem existir outros pontos que não estejam no clique, mas que também possam ser cobertos pelo disco definido pelo *flock* candidato em todos os instantes de tempo da janela. As entidades correspondentes a estes pontos devem ser inseridas no candidato, para garantir que seja um *flock* maximal. Para isto, o algoritmo procura em cada instante de tempo por pontos cuja distância do centro do disco envolvente mínimo do instante seja menor que metade do diâmetro ϵ do *flock*. Os pontos que estiverem na interseção dos pontos nos discos de todos os instantes de tempo definem as entidades que formam o *flock* f_ω, que é um dos k_ϵ-*Flocks*. Todo o ciclo é repetido até que se tenha na lista de retorno k *flocks* ou todas as arestas em Q tenha sido processadas, retornando os k_ϵ-*Flocks* na janela.

4. Conclusão

Através das análises realizadas em diversos trabalhos da área de trajetórias e objetos móveis, é evidente a necessidade do estudo de padrões de comovimento de entidades. Nos trabalhos citados, são encontrados métodos baseados no encontro de agrupamentos com parâmetro de distância livre, retirando a dificuldade de setar o parâmetro da distância e abrindo um novo leque de possibilidades para o desenvolvimento de algoritmos de detecção.

Este artigo apresentou um novo algoritmo para detecção de k_ϵ-*Flocks* em uma janela temporal, baseado no agrupamento sucessivo de pontos próximos, encontrando agrupamentos correspondentes aos k_ϵ-*Flocks* ao mapear-se o espaço como grafos e detectar-se cliques incrementalmente. Trabalhos futuros incluem a implementação do código referente a proposta em plataforma equivalente aos trabalhos já publicados e a avaliação de diferentes conjuntos de dados de trajetórias para teste, além de propor otimizações para acelerar a construção das listas ordenadas de arestas e o encontro de cliques, para melhorar a eficiência do algoritmo.

Agradecimentos

Este trabalho teve apoio do CNPq e da Fundação Araucária.

[1] `https://www.nayuki.io/page/smallest-enclosing-circle`

Referências

Benkert, M., Gudmundsson, J., Hübner, F., and Wolle, T. (2008). Reporting flock patterns. *Comput. Geom.*, 41(3):111–125.

Cao, Y., Zhu, J., and Gao, F. (2016). An algorithm for mining moving flock patterns from pedestrian trajectories. In Morishima, A., Chang, L., Fu, T. Z. J., Liu, K., Yang, X., Zhu, J., Zhang, R., Zhang, W., and Zhang, Z., editors, *Web Technologies and Applications - APWeb 2016 Workshops, WDMA, GAP, and SDMA, Suzhou, China, September 23-25, 2016, Proceedings*, volume 9865 of *Lecture Notes in Computer Science*, pages 310–321.

Ester, M., Kriegel, H., Sander, J., and Xu, X. (1996). A density-based algorithm for discovering clusters in large spatial databases with noise. In Simoudis, E., Han, J., and Fayyad, U. M., editors, *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining (KDD-96), Portland, Oregon, USA*, pages 226–231. AAAI Press.

Gudmundsson, J. and van Kreveld, M. J. (2006). Computing longest duration flocks in trajectory data. In de By, R. A. and Nittel, S., editors, *14th ACM International Symposium on Geographic Information Systems, ACM-GIS 2006, November 10-11, 2006, Arlington, Virginia, USA, Proceedings*, pages 35–42. ACM.

Li, Z., Ding, B., Han, J., and Kays, R. (2010). Swarm: Mining relaxed temporal moving object clusters. *Proc. VLDB Endow.*, 3(1):723–734.

Markovic, N., Sekula, P., Laan, Z. V., Andrienko, G. L., and Andrienko, N. V. (2019). Applications of trajectory data from the perspective of a road transportation agency: Literature review and maryland case study. *IEEE Trans. Intell. Transp. Syst.*, 20(5):1858–1869.

Pérez, I. A., Artuso, F., Mahmud, M., Kulshrestha, U., Sánchez, M. L., and García, M. (2015). Applications of air mass trajectories. *Advances in Meteorology*, 2015.

Sanches, D. E., Alvares, L. O., Bogorny, V., Vieira, M. R., and Kaster, D. S. (2018). A top-down algorithm with free distance parameter for mining top-k flock patterns. In Mansourian, A., Pilesjö, P., Harrie, L., and van Lammeren, R., editors, *Geospatial Technologies for All - Selected Papers of the 21st AGILE Conference on Geographic Information Science, Lund, Sweden, 12-15 June 2018*, Lecture Notes in Geoinformation and Cartography, pages 233–249. Springer.

Spaccapietra, S., Parent, C., Damiani, M. L., de Macêdo, J. A. F., Porto, F., and Vangenot, C. (2008). A conceptual view on trajectories. *Data Knowl. Eng.*, 65(1):126–146.

Vieira, M. R., Bakalov, P., and Tsotras, V. J. (2009). On-line discovery of flock patterns in spatio-temporal data. In Agrawal, D., Aref, W. G., Lu, C., Mokbel, M. F., Scheuermann, P., Shahabi, C., and Wolfson, O., editors, *17th ACM SIGSPATIAL International Symposium on Advances in Geographic Information Systems, ACM-GIS 2009, November 4-6, 2009, Seattle, Washington, USA, Proceedings*, pages 286–295. ACM.

Wang, Y., Lim, E., and Hwang, S. (2006). Efficient mining of group patterns from user movement data. *Data Knowl. Eng.*, 57(3):240–282.