Detecting Smooth Cluster Changes in Evolving Graph Structures

Sohei Okui, Kaho Osamura, and Akihiro Inokuchi

Abstract Graph mining is a set of techniques for finding useful patterns in various types of structured data. Many effective algorithms for mining static graphs have been proposed. However, graphs of human relationships and evolving genes change over time, and such evolving graphs require different algorithms for analysis. In this chapter, we explain a method called O2I for clustering in evolving graphs that can detect changes in clusters over time. O2I partitions the graph sequence into smooth clusters, even when the numbers of clusters and vertices vary. It first constructs a graph from the graph sequence, then uses spectral clustering and the RatioCut to apply k partitioning to this graph. O2I is compared in detail with the preserving clustering membership (PCM) algorithm, which is a conventional online graph-sequence clustering algorithm in which the numbers of clusters and vertices must remain constant. We further show that, in contrast to PCM, the performance of O2I is not dependent on the clustering of the initial graph in the graph sequence. Experiments on synthetic evolving graphs show that O2I is practical to calculate and addresses the main disadvantages of PCM. Further tests on real-world data show that O2I can obtain reasonable clusters. This method is hence a flexible clustering solution and will be useful on a wide range of graph-mining applications in which the connections, number of clusters, and number of vertices of the graphs evolve over time.

S. Okui
Graduate School of Science and Technology, Kwansei Gakuin University, Sanda, Japan

K. Osamura · A. Inokuchi (✉)
School of Science and Technology, Kwansei Gakuin University, Sanda, Japan
e-mail: osamura.kaho.oe5@is.naist.jp; inokuchi@kwansei.ac.jp

M. Sayed-Mouchaweh (ed.), *Learning from Data Streams in Evolving Environments*, Studies in Big Data 41, https://doi.org/10.1007/978-3-319-89803-2_10

1 Introduction

Studies on graph mining have established many approaches for finding useful patterns in various types of structured data. Although the major algorithms for graph mining are quite effective in practice, most of them focus on *static* graphs, whose structures do not change over time. However, evolving graphs are used to model many real-world applications [12]. For example, a human network can be represented as a graph in which each human and each relationship between two humans correspond to a vertex and an edge, respectively. If a human joins (or leaves) a community in the human network, the numbers of vertices and edges in the graph can change. Similarly, the evolution of a gene network, which consists of genes and their interactions, produces a graph sequence when genes are added, deleted, or mutated. Recently, much attention has been given to graph mining from evolving graphs [6, 19]. Figure 1 shows an example of an evolving graph with four steps and ten unique IDs, indicated by the numbers attached to the vertices. In addition, edge weights are represented by line thickness. For example, humans in a human network correspond to vertices, each of the humans has a unique ID. The strength of the friendship between two humans is represented as an edge weight between two vertices. The current human network is represented as a weighted graph, and the network evolves over time. To represent the evolving network, we use a graph sequence consisting of a series of graphs.

In this chapter, we tackle the problem of clustering in evolving graphs to detect changes in the clusters. In an evolving graph, the number of clusters increases when a cluster divides or decreases when two clusters merge. Although most conventional clustering algorithms focus on partitioning a set of points in a vector space into k clusters, von Luxburg [18] notes that the k partition problem in a vector space can be reduced to the k partition problem in a graph, where each vertex corresponds to a point in the set to be partitioned and an edge indicates the similarity between two vertices. Therefore, the methods in this chapter are applicable to both evolving graphs and points arriving over time.

In [19], problems involving clustering points arriving over time are categorized into four types. Let D_t be a set of points at time t in the vector space, and let $D = \langle D_1, D_2, \ldots, D_T \rangle$ be a series of sets of points. The first type of clustering

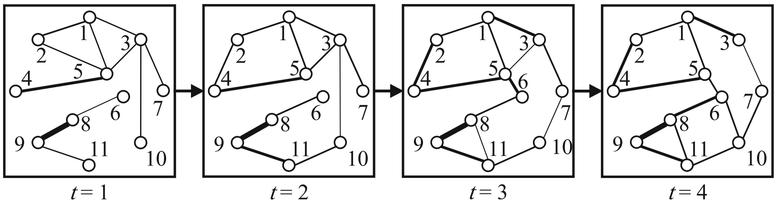


Fig. 1 Example of a weighted graph sequence with four steps (numbers attached to vertices represent vertex IDs)

problem is where only one point arrives at each time t [1–3, 7, 9, 11, 15]. This type of problem focuses on online data processing. The second type of problem is clustering n sequences into k clusters and is applicable to clustering DNA sequences or protein sequences in bioinformatics [4, 14, 17]. This differs from the first type of problem as it does not require online data processing and D_t consists of n points. The third type of problem is to cluster n data streams into k clusters [5, 8]. Although this is the same as the first type in terms of the online analysis of data, each set D_t contains n points, unlike the first type. Preserving cluster membership in Sect. 2.2 tackles this type of problem.

The fourth type of problem, which is addressed in this chapter, analyzes a series D of sets D_t, each of which contains at most n points and is given before the analysis [19]. Although each point clustered in the second type of problem is a sequence, each point clustered in the fourth type of problem is a point in D_t. In addition, while a set of k clusters is returned in the second type of problem, T sets of k clusters are returned in the fourth type. While the predecessors D_τ $(\tau < t)$ of D_t are used to cluster D_t in the third type of problem, its successors D_τ $(\tau \geq t)$ are also used to cluster D_t in the fourth type.

In this chapter, we explain an algorithm called O2I that partitions the vertices of a graph sequence into smooth clusters, even when the number of vertices is allowed to vary over time [16]. O2I uses spectral clustering and relies on applying the k partition problem to a graph constructed from a graph sequence. Several experiments demonstrate the performance of O2I and its advantages over existing methods.

The remainder of this chapter is organized as follows: In Sect. 2, we formalize the graph sequence clustering problem that we consider in this chapter and explain the conventional method called PCM and its some drawbacks. In Sect. 3, we explain a method called O2I that overcomes the drawbacks of PCM and discuss the relationship between the performance of OI2 and connectivities of graphs in a graph sequence. In Sect. 4, we compare O2I with PCM in terms of clustering accuracy using artificially generated datasets, and verify the practicality of O2I on a real-world dataset. Finally, we conclude the chapter in Sect. 5.

2 Clustering a Graph Sequence

2.1 *Problem Definition*

In this chapter, to model an evolving graph, we use a weighted graph sequence. A weighted graph at time t is represented by $G^{(t)} = \left(V^{(t)}, E^{(t)}, w^{(t)}\right)$, where $V^{(t)}$ is a set of vertices, each of which has a unique ID, $E^{(t)} = V^{(t)} \times V^{(t)}$ is the set of all edges, and $w^{(t)}$ is a function that assigns nonnegative real values to the edges at time t. A series of T graphs is called a weighted graph sequence with T steps and is denoted by $\left\langle G^{(1)}, G^{(2)}, \ldots, G^{(T)}\right\rangle$. Although we assume that the value of

$|V^{(t)}| = n$ is unchanged in the graph sequence in this section, the value of $|V^{(t)}|$ in O2I explained in the next section changes over time.

Figure 1 shows an example of a graph sequence with four steps. In the figure, edge weights are represented by line thickness. For the sake of simplicity, we do not show edges with weight 0 in the figures in this chapter.[1]

The vertices $V^{(t)}$ in a graph at time t are partitioned into k disjoint subsets $P^{(t)} = \left\{C_1^{(t)}, C_2^{(t)}, \ldots, C_k^{(t)}\right\}$, where $\bigcup_{j=1}^{k} C_j^{(t)} = V^{(t)}$. Using this notation, a cluster sequence can be written as $\left\langle C_j^{(1)}, C_j^{(2)}, \ldots, C_j^{(T)}\right\rangle$ for $1 \le j \le k$.

Given a graph sequence $\left\langle G^{(1)}, G^{(2)}, \ldots, G^{(T)}\right\rangle$ and the number of clusters k as inputs, the problem addressed in this chapter is how to determine cluster sequences $\left\{\left\langle C_j^{(1)}, C_j^{(2)}, \ldots, C_j^{(T)}\right\rangle \mid 1 \le j \le k)\right\}$ that satisfy the following two requirements:

1. Vertices connected by high-weight edges in a graph at time t should appear in the same cluster for each graph in the sequence.

2. Clusters $C_j^{(t)}$ and $C_j^{(t+1)}$ should be almost the same. This requirement is called cluster smoothness.

Figures 2 and 3 show cluster sequences obtained from the graph sequence in Fig. 1. When we do not take requirement (2) into account, vertices 5 and 6 appear in the same cluster because the edge (5, 6) at time 3 has a high weight, as shown in Fig. 2. In contrast, when we take requirement (2) into account, vertex 6 is assigned to $C_2^{(t)}$ before and after time 3, and hence both clusters $C_2^{(2)}$ and $C_2^{(3)}$ are the same.

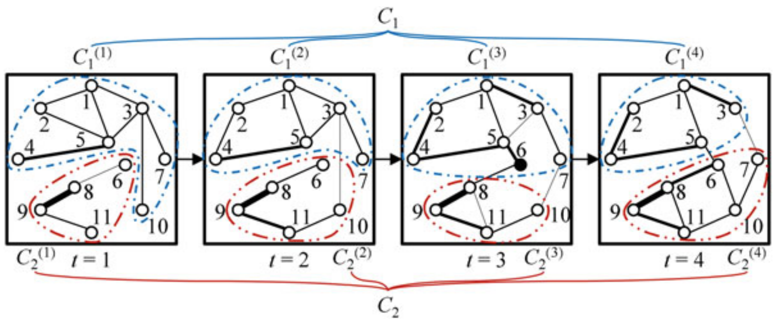


Fig. 2 Cluster sequences (1) obtained from the graph sequence in Fig. 1

[1]The weight 0 means that there is no connection between two vertices. We need these zeros to create Laplacian matrices in Sect. 2.2.

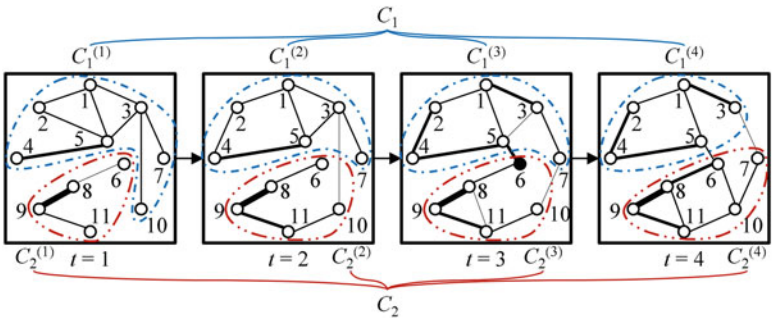


Fig. 3 Cluster sequences (2) obtained from the graph sequence in Fig. 1

2.2 *Preserving Cluster Membership*

The k partition problem for a graph $G = (V, E, w)$ is defined as the problem of finding non-empty sets $C_1, C_2, \ldots, C_k$ that partition V and that minimize

$$\sum_{j=1}^{k} \frac{1}{|C_j|} \sum_{e \in E(C_j, V \setminus C_j)} w(e),$$

where $E(S, V \setminus S)$ is the set of edges (v, u) with $v \in S$ and $u \in V \setminus S$. This optimization problem minimizes the function called RatioCut. According to [18], the above minimization problem is equivalent to

$$\min_{X \in \mathscr{R}^{n \times k}} tr(X^T L X) \qquad s.t. \;\; X^T X = I, [2] \tag{1}$$

where the n-by-k matrix X indicates the cluster to which each vertex belongs, with element x_{ij} of the matrix given by

$$x_{ij} = \begin{cases} \frac{1}{\sqrt{|C_j|}} & if \;\; v_i \in C_j, \\ 0 & otherwise, \end{cases}$$

where $X^T X = I$ indicates that each vertex in graph G belongs to one cluster, where I is the identity matrix of size n. In addition, L is the Laplacian matrix of G, defined as follows. Let A be an adjacency matrix for G, where the (i, j)th element a_{ij} is weight $w((i, j))$ if an edge exists between v_i and v_j in G. Otherwise, a_{ij} is 0. Setting $D = diag(\sum_{i=1}^{n} a_{i1}, \sum_{i=1}^{n} a_{i2}, \ldots, \sum_{i=1}^{n} a_{in})$, the Laplacian matrix is $L = D - A$. Equation (1) is called spectral clustering.

[2]Because of space limitations, we omit $X^T X = I$ henceforth.

One online algorithm for clustering a graph sequence is called preserving cluster membership (PCM) [10]. In this algorithm, the matrix X_{t-1} corresponding to $G^{(t-1)}$ is known and we are given graph $G^{(t)}$. The algorithm obtains cluster sequences by iteratively optimizing

$$\min_{X_t \in \mathscr{R}^{n\times k}} tr(X_t^T L_t X_t) + \alpha ||X_t X_t^T - X_{t-1} X_{t-1}^T||^2, \tag{2}$$

where $\alpha \geq 0$ and L_t is the Laplacian matrix of $G^{(t)}$. If the ith and jth vertices belong to the same cluster at time t, then the (i, j)th element of $X_t X_t^T$ is a positive real number. Otherwise, the (i, j)th element is 0. Minimizing the second term of the objective function in Eq. (2) under the Frobenius norm, where $||W||^2 = tr(W^T W)$, satisfies requirement (2). The objective function is transformed as follows:

$$\begin{aligned} & tr(X_t^T L_t X_t) + \alpha ||X_t X_t^T - X_{t-1} X_{t-1}^T||^2 \\ &= tr(X_t^T L_t X_t) + \alpha\, tr(X_t X_t^T - X_{t-1} X_{t-1}^T)^T (X_t X_t^T - X_{t-1} X_{t-1}^T) \\ &= tr(X_t^T L_t X_t) + \alpha\, tr(X_t X_t^T X_t X_t^T - 2 X_t X_t^T X_{t-1} X_{t-1}^T + X_{t-1} X_{t-1}^T X_{t-1} X_{t-1}^T) \\ &= tr(X_t^T L_t X_t) + 2\alpha k - 2\alpha\, tr(X_t^T X_{t-1} X_{t-1}^T X_t) \\ &= 2\alpha k + tr(X_t^T L_t X_t - 2\alpha X_t^T X_{t-1} X_{t-1}^T X_t) \\ &= 2\alpha k + tr[X_t^T (L_t - 2\alpha X_{t-1} X_{t-1}^T) X_t]. \end{aligned}$$

Therefore, Eq. (2) is equivalent to

$$\min_{X_t \in \mathscr{R}^{n\times k}} tr\left[X_t^T \left(L_t - 2\alpha X_{t-1} X_{t-1}^T\right) X_t\right]. \tag{3}$$

In [10], an offline algorithm was also proposed as an extension to PCM. To demonstrate the offline algorithm, the authors introduced an optimization problem for clustering $G^{(t)}$ using known X_{t-1} and X_{t+1} corresponding to clusters $P^{(t-1)}$ and $P^{(t+1)}$, respectively:

$$\min_{X_t \in \mathscr{R}^{n\times k}} tr\left[X_t^T \left(L_t - \alpha X_{t-1} X_{t-1}^T - \alpha X_{t+1} X_{t+1}^T\right) X_t\right]. \tag{4}$$

We define the functions $func_1(L)$, $func_2(L_t, X_{t-1}, \alpha)$, and $func_3(L_t, X_{t-1}, X_{t+1}, \alpha)$ to be the minimum values of Eqs. (2), (3), and (4), respectively. Using these functions, the PCM offline algorithm is shown in Algorithm 1. First, the PCM offline algorithm clusters $G^{(1)}$, and then it clusters $G^{(2)}$ using the results from time 1. This process is repeated for the series of graphs. Next, it clusters $G^{(1)}$ using the results from time 2. Then, it clusters $G^{(2)}$ using the results from times 1 and 3. The process repeats until convergence.

Algorithm 1: PCM_offline

```
   Data: $\langle G^{(1)}, G^{(2)}, \ldots, G^{(T)} \rangle, k$
   Result: $X_1, X_2, \ldots, X_T$
1  for $t \in [1, T]$ do
2      if $t = 1$ then
3          $X_1 = func_1(L_1)$;
4      else
           $X_t = func_2(L_t, X_{t-1}, \alpha)$;
5  repeat
6      for $t \in [1, T]$ do
7          if $t = 1$ then
8              $X_1 = func_2(L_1, X_2, \alpha)$;
9          else
               if $t = T$ then
10                 $X_T = func_2(L_T, X_{T-1}, \alpha)$;
11             else
                   $X_t = func_3(L_t, X_{t-1}, X_{t+1}, \alpha)$;
   until $X_1, X_2, \ldots, X_T$ converge;
12 return $X_1, X_2, \ldots, X_T$;
```

When α is decreased, each graph in a graph sequence is clustered independently because the first term in Eq. (2), which relates to requirement (1), is emphasized over requirement (2). This results in the cluster sequences in Fig. 2. On the one hand, when α is increased, the smooth cluster sequences in Fig. 3 are obtained because the second term in Eq. (2), which relates to requirement (2), is emphasized. In concrete terms, vertex 6 at time 3 belongs to cluster sequence C_1 in Fig. 2, while it belongs to the other cluster sequence C_2 at times 2 and 4. On the other hand, placing vertex 6 in C_1 in Fig. 3 decreases the second term of Eq. (2), and hence vertex 6 belongs to C_1 at all times.

2.3 Drawbacks of PCM

We point out three drawbacks of the PCM offline algorithm. First, the performance of PCM is dependent on $G^{(1)}$. If the vertices in each latent cluster of $G^{(1)}$ are strongly connected, then the problem of obtaining cluster sequences from a graph sequence is relatively easy because the algorithm uses X_1 to cluster $G^{(2)}$ and then uses X_t to cluster $G^{(t+1)}$ for $t > 1$. However, if the clusters for $G^{(1)}$ are not suitable, then this unsuitability propagates to clusters in $P^{(t)}$ for $t > 1$ because of the second term in Eq. (2).

The second drawback comes from having k clusters at all times. For this reason, PCM cannot determine suitable cluster sequences from a graph sequence when the number of clusters increases after one cluster divides or when the number of clusters decreases after two clusters merge. To detect suitable cluster sequences from a graph sequence, we should allow the number of clusters to vary over time.

The third drawback of PCM is related to the second drawback. The number of vertices in the graph is the same at all times in PCM. However, the members of a social network are not constant, but change over time. Therefore, we should allow members to join and leave the network and develop a clustering algorithm for data in which the number of vertices is not constant.

3 Detecting Smooth Cluster Changes in a Graph Sequence

3.1 Clustering a Graph Sequence Using Smoothness Between Two Successive Graphs

Okui et al. have proposed a method called O2I that overcomes the first and second drawbacks of PCM that are explained in the previous section [16]. To explain the method, we discuss the problem of obtaining the $X_1, X_2, \ldots, X_T \in \mathscr{R}^{n\times k}$ that minimize

$$\sum_{t=1}^{T} tr(X_t^T L_t X_t) + \alpha' \sum_{t=1}^{T-1} ||X_t - X_{t+1}||^2, \tag{5}$$

where $\alpha' > 0$. Minimizing the first term in Eq. (5) corresponds to clustering each graph $G^{(t)}$ in a graph sequence according to requirement (1). To show that minimizing the second term in Eq. (5) corresponds to requirement (2), we consider a graph sequence consisting of only two graphs. The objective function for the sequence is given by

$$tr(X_1^T L_1 X_1) + tr(X_2^T L_2 X_2) + \alpha' ||X_1 - X_2||^2. \tag{6}$$

From Eq. (6), we derive the equation in Fig. 4. Similarly, the equation shown in Fig. 5 is derived from Eq. (5). When D' is the underlined matrix in Fig. 5 and W' is the double underlined matrix in Fig. 5, matrix $L' = D' - W'$ is the Laplacian matrix for a graph G' that satisfies the following:

- The number of vertices in G' is $n \times T$. Henceforth, the ith vertex of G' at time t is represented by $v_{t,i}$.
- If $G^{(t)}$ contains an edge (i, j) of weight $w((i, j))$, then G' also contains an edge $(v_{t,i}, v_{t,j})$ of $w((i, j))$.
- Graph G' contains an edge $(v_{t,i}, v_{t+1,i})$ of weight α' for $1 \le t \le T - 1$ and $1 \le i \le n$.

Therefore, the problem of minimizing Eq. (5) is reduced to the k partition problem for G'. The edges between vertices $v_{t,i}$ and $v_{t+1,i}$ have weight α'. Cutting some

$$
\begin{aligned}
& tr\left[X_1^T L_1 X_1\right] + tr\left[X_2^T L_2 X_2\right] + \alpha' \, ||X_1 - X_2||^2 \\
&= tr\left[X_1^T L_1 X_1 + X_2^T L_2 X_2\right] + \alpha' \, tr[X_1^T X_1 + X_2^T X_2 - X_2^T X_1 - X_1^T X_2] \\
&= tr\left[\begin{pmatrix} X_1 \\ X_2 \end{pmatrix}^T \begin{pmatrix} L_1 & 0 \\ 0 & L_2 \end{pmatrix} \begin{pmatrix} X_1 \\ X_2 \end{pmatrix}\right] + \alpha' \, tr\left[\begin{pmatrix} X_1 \\ X_2 \end{pmatrix}^T \begin{pmatrix} I & -I \\ -I & I \end{pmatrix} \begin{pmatrix} X_1 \\ X_2 \end{pmatrix}\right] \\
&= tr\left[\begin{pmatrix} X_1 \\ X_2 \end{pmatrix}^T \begin{pmatrix} L_1 + \alpha' I & -\alpha' I \\ -\alpha' I & L_2 + \alpha' I \end{pmatrix} \begin{pmatrix} X_1 \\ X_2 \end{pmatrix}\right] \\
&= tr\left[\begin{pmatrix} X_1 \\ X_2 \end{pmatrix}^T \left\{ \begin{pmatrix} D_1 + \alpha' I & 0 \\ 0 & D_2 + \alpha' I \end{pmatrix} - \begin{pmatrix} W_1 & \alpha' I \\ \alpha' I & W_2 \end{pmatrix} \right\} \begin{pmatrix} X_1 \\ X_2 \end{pmatrix}\right]
\end{aligned}
$$

Fig. 4 Objective function for a graph sequence with two steps

$$
tr\left[\begin{pmatrix} X_1 \\ X_2 \\ \vdots \\ \vdots \\ X_T \end{pmatrix}^T \left\{ \begin{pmatrix} D_1 + \alpha' I & 0 & \cdots & \cdots & 0 \\ 0 & D_2 + 2\alpha' I & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & 0 \\ \vdots & \ddots & \ddots & D_{T-1} + 2\alpha' I & 0 \\ 0 & \cdots & \cdots & 0 & D_T + \alpha' I \end{pmatrix} - \begin{pmatrix} W_1 & \alpha' I & 0 & \cdots & 0 \\ \alpha' I & W_2 & \ddots & \ddots & \vdots \\ 0 & \ddots & \ddots & \ddots & 0 \\ \vdots & \ddots & \ddots & W_{T-1} & \alpha' I \\ 0 & \cdots & 0 & \alpha' I & W_T \end{pmatrix} \right\} \begin{pmatrix} X_1 \\ X_2 \\ \vdots \\ \vdots \\ X_T \end{pmatrix}\right]
$$

Fig. 5 Objective function for a graph sequence with T steps

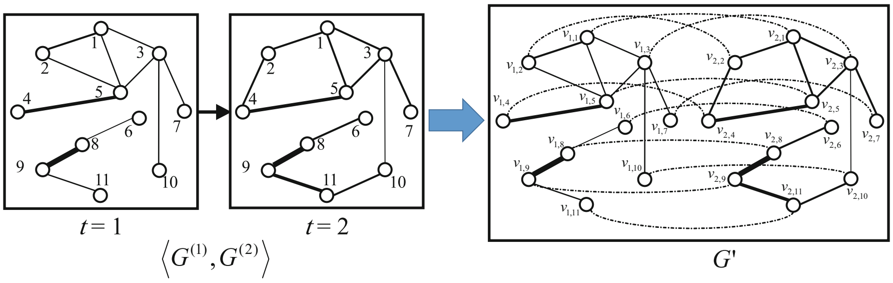


Fig. 6 Conversion of a graph sequence with two steps into a graph G'

of these edges increases the value of the objective function in Eq. (5) when G' is partitioned to k subgraphs. For this reason, $v_{t,i}$ and $v_{t+1,i}$ are likely to appear in the same cluster, so these edges may not be cut. Therefore, minimizing Eq. (5) satisfies requirement (2). Figure 6 shows an example of transforming a graph sequence with two steps $\langle G^{(1)}, G^{(2)} \rangle$ to a graph G'. In this figure, broken lines represent edges of weights α'.

The problem of minimizing Eq. (5) has hence been reduced to the k partition problem for G'. The cluster sequences obtained by O2I is not dependent on clustering $G^{(1)}$, unlike in PCM, which first partitions $G^{(1)}$ and then iteratively partitions the other graphs. Thus, the first drawback of PCM is overcome. In

Algorithm 2: O2I

```
    Data: $\langle G^{(1)}, G^{(2)}, \dots, G^{(T)} \rangle, k$
    Result: $X_1, X_2, \dots, X_T$
 1  Construct $G'$ from $\langle G^{(1)}, G^{(2)}, \dots, G^{(T)} \rangle$;
 2  $\ell = n \times T$;
 3  Compute the Laplacian matrix $L'$ of $G'$;
 4  Compute the first $k$ eigenvectors $\mathbf{u}_1, \mathbf{u}_2, \dots, \mathbf{u}_k$ of $L'$;
 5  Let $U \in \mathscr{R}^{\ell \times k}$ be the matrix that has $\mathbf{u}_q$ as its $q$th column;
 6  For $i = 1, \dots, \ell$, let $\mathbf{y}_i \in \mathscr{R}^k$ be the vector corresponding to the $i$th row of $\Gamma$;
 7  Use the $k$-means algorithm to cluster the points $\{\mathbf{y}_1, \mathbf{y}_2, \dots, \mathbf{y}_\ell\}$ in $\mathscr{R}^k$ into clusters
    $P_1, P_2, \dots, P_k$;
 8  for $t \in [1, T]$ do
 9      $X_t = 0$;
10  for $j \in [1, k]$ do
11      for $v_{t,i} \in P_j$ do
12          $x_{i,j}^{(t)} = \frac{1}{\sqrt{|\{v_{t',i'} \in P_j | t = t'\}|}}$;
13  return $X_1, X_2, \dots, X_T$;
```

addition, some clusters obtained using O2I may not contain any vertex from time t. Therefore, O2I does not guarantee a partition of each graph $G^{(t)}$ into exactly k clusters, but instead partitions the graph into k or fewer clusters. Hence, O2I overcomes the second drawback.

Note that O2I requires $\alpha' > 0$ in Eq. (5). If $\alpha' = 0$, then there are no edges between $G^{(t)}$ and $G^{(t+1)}$ in G', so when G' is partitioned into T clusters, each graph $G^{(t)}$ becomes a cluster. Therefore, α' should be a positive real number.

The objective function for O2I is similar to the objective function for PCM. However, it is impossible to derive an equation in the form of Fig. 4 from

$$tr(X_1^T L_1 X_1) + tr(X_2^T L_2 X_2) + \alpha ||X_1^T X_1 - X_2^T X_2||^2.$$

Thus, it is impossible to reduce the objective function for PCM to the k partition problem for a graph.

Algorithm 2 shows the pseudo code for O2I. The method uses the spectral clustering algorithm [18] in lines 3–6. It then initializes X_t to the zero matrix in lines 7–8. In lines 9–11, if the jth cluster P_j contains $v_{t,i}$, then $\frac{1}{\sqrt{|\{v_{t',i'} \in P_j | t = t'\}|}}$ replaces $x_{i,j}^{(t)}$ in X_t.

Using Algorithm 2, the number of vertices does not have to be n at all times. The third drawback is hence resolved by replacing $\ell = n \times T$ in line 2 with $\ell = \sum_{t=1}^{T} |V^{(t)}|$.

3.2 Clustering Using the Forgetting Rate

We considered the smoothness of clusters between two consecutive timesteps in the previous subsection. In this section, we extend O2I to consider cluster smoothness between timesteps separated by distance τ. This is formulated in the following equation:

$$\sum_{t=1}^{T} tr(X_t^T L_t X_t) + \alpha' \sum_{\tau=1}^{T-1} \gamma^{\tau-1} \sum_{t=1}^{T-\tau} ||X_t - X_{t+\tau}||^2, \tag{7}$$

where γ is called the forgetting rate and $0 \leq \gamma \leq 1$. Equation (7) is a generalization of Eq. (5), as they are equivalent when $\gamma = 0$. The equation shown in Fig. 7 is derived in a manner similar to that in the previous section, where $B_t = \alpha' \sum_{\tau=1,\tau\neq t}^{T} \gamma^{|t-\tau|-1} I$ is a diagonal matrix. The underlined part in Fig. 7 is the Laplacian matrix L'' of graph G'' that satisfies the following:

- The number of vertices in G'' is $n \times T$.
- If $G^{(t)}$ contains an edge (i, j) of weight $w((i, j))$, then G' also contains an edge $(v_{t,i}, v_{t,j})$ of $w((i, j))$.
- Graph G'' contains an edge of weight $\alpha'\gamma^{(t'-t-1)}$ between $v_{t,i}$ and $v_{t',i}$ for $1 \leq t < t' \leq T$ and $1 \leq i \leq n$.

Graph G' is a subgraph of G''. When $\gamma = 0$, G' is isomorphic to G''. Algorithm 2 is applicable to G'' by replacing G' and L' with G'' and L'', respectively. In the previous section, we explained that O2I overcomes the third drawback of PCM. However, when v_i is contained in $G^{(t)}$ but not in both $G^{(t-1)}$ and $G^{(t+1)}$, we cannot consider the smoothness of clusters for this vertex. In contrast, O2I using the forgetting rate further overcomes the third drawback by introducing edges with weights that decrease exponentially with the distance between graphs.

$$tr\left[\begin{pmatrix} X_1 \\ X_2 \\ \vdots \\ \vdots \\ X_T \end{pmatrix}^T \left\{ \underline{\begin{pmatrix} D_1+B_1 I & 0 & \cdots\cdots & 0 \\ 0 & D_2+B_2 & \ddots\ \ddots & \vdots \\ \vdots & \ddots & \ddots\ \ddots & 0 \\ \vdots & \ddots & \ddots\ \ddots & 0 \\ 0 & \cdots & \cdots\ 0 & D_T+B_T \end{pmatrix} - \begin{pmatrix} W_1 & \alpha' I & \alpha'\gamma I & \cdots & \alpha'\gamma^{T-2} I \\ \alpha' I & W_2 & \ddots & \ddots & \vdots \\ \alpha'\gamma I & \ddots & \ddots & \ddots & \alpha'\gamma I \\ \vdots & \ddots & \ddots & W_{T-1} & \alpha' I \\ \alpha'\gamma^{T-2} I & \cdots & \alpha'\gamma I & \alpha' I & W_T \end{pmatrix}} \right\} \begin{pmatrix} X_1 \\ X_2 \\ \vdots \\ \vdots \\ X_T \end{pmatrix}\right]$$

Fig. 7 Objective function with forgetting rate for a graph sequence with T steps

3.3 Connectivities of Graphs

In this section, we discuss the effect of the connectivity of each graph in a graph sequence on the clustering result. For the sake of simplicity, we consider the simple example of a sequence of sets of points, as shown in Fig. 8. Each timestep consists of three points whose coordinates are given in the figure. We convert each of the sets of the points into a graph where the vertices and edge weights are the points and $\exp(-\frac{d^2}{2})$, respectively, where d is the Euclidean distance between two points. We then obtain a graph sequence with two steps. We assume that $\langle\{v_1\}, \{v_1, v_2\}\rangle$ and $\langle\{v_2, v_3\}, \{v_3\}\rangle$ are desirable cluster sequences obtained from the graph sequence for $k = 2$.

Figure 9 shows the graph $G'_1 = (V'_1, E'_1, w'_1)$ created from the graph sequence with two steps. Because G'_1 is partitioned by RatioCut, we obtain the following solutions depending on the value of α'.

$$\min \sum_{j=1}^{2} \frac{1}{|C_j|} \sum_{e \in E'_1(C_j, V'_1 \setminus C_j)} w'_1(e)$$
$$= \begin{cases} 2\alpha' & \text{if } 0 < \alpha' \leq 0.56 \; (C_1 = \{v_{1,1}, v_{1,2}, v_{1,3}\}), \text{ and} \\ 1.11 & \text{otherwise} \qquad (C_1 = \{v_{1,1}, v_{2,1}\}). \end{cases} \tag{8}$$

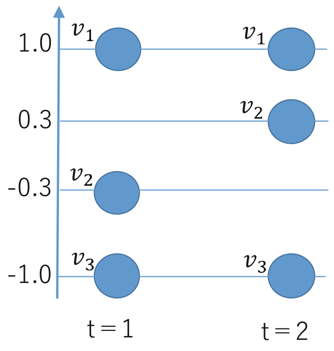


Fig. 8 Example of a sequence of sets of points

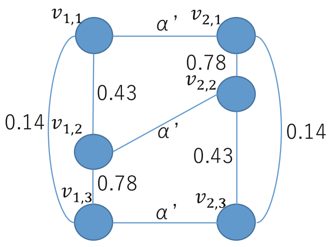


Fig. 9 Graph G'_1 created from the sequence of points in Fig. 8

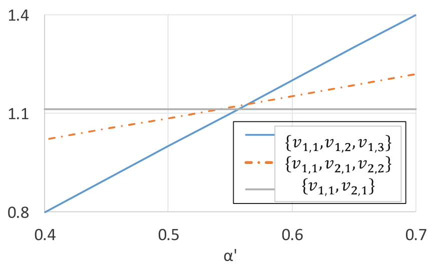


Fig. 10 Solutions for G'_1 and various α'

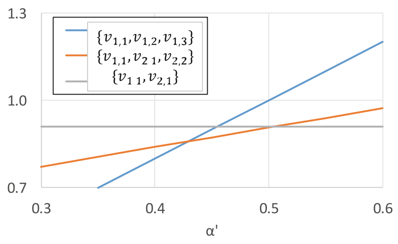


Fig. 11 Solutions for G'_2 and various α'

Equation (8) is represented by Fig. 10 for various α'. When α' is less than 0.56, G'_1 is partitioned into $\{v_{1,1}, v_{1,2}, v_{1,3}\}$ and $\{v_{2,1}, v_{2,2}, v_{2,3}\}$, which does not satisfy cluster smoothness. In contrast, when α' is greater than 0.56, G'_1 is partitioned into $\{v_{1,1}, v_{2,1}\}$ and $\{v_{1,2}, v_{1,3}, v_{2,2}, v_{2,3}\}$, which means that O2I cannot detect any changes in the clusters because requirement (2) is oversatisfied. Thus, O2I has no chance to obtain the desirable cluster sequences from G'_1 shown in Fig. 9, even if α' is tuned to the optimal value.

In the previous example, we created the complete graph from each set of points in a timestep. In [18], the ϵ-neighborhood graph and κ-nearest neighbor graph are introduced as an alternative to a complete graph created from a set of points. In the ϵ-neighborhood graph, two points are connected by an edge if a distance d between the points is less than ϵ. In the κ-nearest neighbor graph, two points are connected if one of them is among the κ-nearest neighbors of the other. When ϵ-neighborhood graph $G'_2 = (V'_2, E'_2, w'_2)$ of ϵ 1 is created from a sequence of points, we obtain the following solutions, depending on the value of α' after partitioning G'_2 using RatioCut.

$$\min \sum_{j=1}^{2} \frac{1}{|C_j|} \sum_{e \in E'_2(C_j, V'_2 \setminus C_j)} w'_2(e)$$

$$= \begin{cases} 2\alpha' & \text{if } 0 < \alpha' \leq 0.43 \quad (C_1 = \{v_{1,1}, v_{1,2}, v_{1,3}\}) \\ 0.67\alpha' + 0.57 & \text{if } 0.43 < \alpha' \leq 0.50 \ (C_1 = \{v_{1,1}, v_{2,1}, v_{2,2}\}) \\ 1.11 & \text{otherwise} \quad (C_1 = \{v_{1,1}, v_{2,1}\}). \end{cases} \tag{9}$$

Similarly to Eq. (8) and Fig. 10, Eq. (9) is illustrated by Fig. 11 for various α'. In contrast to the case for G'_1, Fig. 11 indicates that O2I can obtain the desirable cluster sequence when $0.43 < \alpha' \leq 0.50$.

When G'_1 consists of two complete graphs, each of which is created from a set of points, vertices coming from the same timestep are connected densely with each other, while every pair of vertices coming from different timesteps is rarely connected. In this case, $\{v_{1,1}, v_{1,2}, v_{1,3}\}$ coming from the same timestep minimizes RatioCut rather than $\{v_{1,1}, v_{2,1}, v_{2,2}\}$. In contrast, when a sparse graph is created from a set of points instead of a dense graph, the connectivities among vertices coming from the same timestep and among vertices coming from the different timesteps are balanced, and O2I can select $\{v_{1,1}, v_{2,1}, v_{2,2}\}$ to minimize RatioCut for G' by tuning α'.

4 Experimental Evaluation

4.1 Experimental Setup

In this section, we compare O2I with the PCM offline algorithm using the adjusted Rand index (ARI). The ARI measures the similarity of two sets of clusters, and is calculated using the number of vertices common to each pair of clusters. We assume that a set of n vertices is partitioned both into r disjoint subsets $\mathscr{U} = \{U_1, U_2, \ldots, U_r\}$ and into c disjoint subsets $\mathscr{V} = \{V_1, V_2, \ldots, V_c\}$, so that $\sum_{i=1}^{r} |U_i| = \sum_{j=1}^{c} |V_j| = n$. The number of vertices common to U_i and V_j is denoted by n_{ij}, as shown in Table 1 , where $n_{i.}$ and $n_{.j}$ are the numbers of vertices in clusters U_i and V_j, respectively. The number of pairs of vertices commonly contained in U_i and V_j is calculated by $\binom{n_{ij}}{2}$. The ARI is hence calculated as

$$\frac{\sum_{i,j}\binom{n_{ij}}{2} - \left[\sum_i \binom{n_{i.}}{2} \sum_j \binom{n_{.j}}{2}\right] / \binom{n}{2}}{\frac{1}{2}\left[\sum_i \binom{n_{i.}}{2} + \sum_j \binom{n_{.j}}{2}\right] - \left[\sum_i \binom{n_{i.}}{2} \sum_j \binom{n_{.j}}{2}\right] / \binom{n}{2}}.$$

The ARI takes a value between 0 and 1. Larger ARI values indicate that the obtained clusters are more suitable because $\mathscr{U}$ and $\mathscr{V}$ correspond to the original partition of

Table 1 Contingency table comparing partitions $\mathscr{U}$ and $\mathscr{V}$

	$\mathscr{V}$				
$\mathscr{U}$	V_1	V_2	$\ldots$	V_c	Total
U_1	n_{11}	n_{12}	$\ldots$	n_{1c}	$n_{1.}$
U_2	n_{21}	n_{22}	$\ldots$	n_{2c}	$n_{2.}$
$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$	$\vdots$
U_r	n_{r1}	n_{r2}	$\ldots$	n_{rc}	$n_{r.}$
Total	$n_{.1}$	$n_{.2}$	$\ldots$	$n_{.c}$	$n_{..} = n$

the data and the partition obtained by the algorithm, respectively. The ARI values given in this chapter are averages for 20 trials.

4.2 Results

4.2.1 Dependence on the Initial Graph of the Graph Sequence

To compare O2I with PCM, we generated artificial datasets using the following procedures with the parameters shown in Table 2. The means of the k Gaussian distributions were placed at equal intervals on a circle of radius r whose center is the origin. A set of points was generated under a Gaussian distribution for each of the means. In this experiment, the sizes of the three sets of points were set to 600, 300, and 200. The set of generated points corresponds to a latent cluster of vertices. The sets of points move toward and away from the origin as time advances, as shown in Fig. 12. The means of the Gaussian distributions are on a sine curve whose amplitude, angular frequency, and initial phase are denoted by A, ω, and φ, respectively. Therefore, the mean of the Gaussian distribution corresponding to the jth cluster at time t is given by

Table 2 Parameters of the artificial data

Parameters	Default values
Number of cluster sequences	k =3
Radius	r =3
Variance of each Gaussian distribution	var =1.0
Number of vertices	n =1100 (=600+300+200)
Amplitude	A =1.0
Angular frequency	$\omega = \frac{\pi}{4}$
Initial phase	φ =0
Steps	T =10
Number of moving vertices	m =5
Connectivity	κ =10

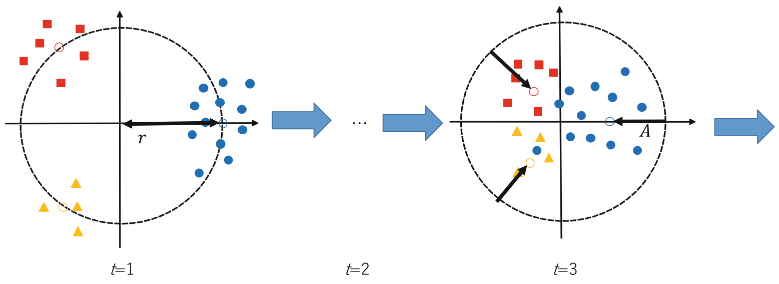


Fig. 12 Generation of the artificial datasets

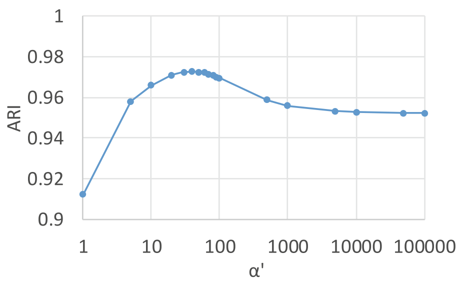


Fig. 13 ARIs for O2I for various values of α'

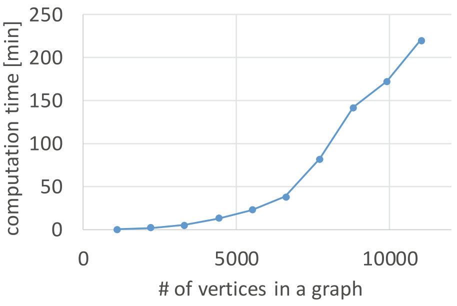


Fig. 14 Computation time for various values of n

$$\begin{pmatrix} x \\ y \end{pmatrix} = R\left(\frac{2\pi(j-1)}{k}\right)\left[A\begin{pmatrix}\sin(\omega(t-1)+\varphi) \\ 0\end{pmatrix} + \begin{pmatrix} r \\ 0 \end{pmatrix}\right], \tag{10}$$

where $R(\theta)$ is a rotation matrix for angle θ, with $1 \le j \le k$ and $1 \le t \le T$. In addition, m points in the largest cluster move to the second-largest cluster at each step. The sets of points at time t are converted into the κ-nearest neighbor graph $G^{(t)}$ with edges with weights $\exp\left(-\frac{d^2}{2}\right)$, where d is the Euclidean distance between two of the points.

Figure 13 shows the ARIs for O2I as α' increases from 1 to 100,000. When $\alpha' = 40$, the two requirements are balanced. When $\alpha' > 40$, the ARI decreases because requirement (2) is oversatisfied because of the influence of the second term in Eq. (5). In contrast, when $\alpha' < 40$, the ARI decreases because requirement (1) is oversatisfied because of the influence of the first term in Eq. (5). Moreover, when α' is decreased to less than 1, the ARI decreases drastically because the weights of the edges between two successive graphs in the graph sequence are less than the weights of the edges in each graph $G^{(t)}$ and O2I partitions G' into clusters by cutting the edges between successive graphs. Therefore, these results confirm that O2I obtains suitable cluster sequences satisfying the requirements by tuning α'. Henceforth, α' is set to 40. Because a similar result was obtained for PCM, α was set to 4.

Figure 14 shows the computation time for O2I when the number of vertices in each graph in a graph sequence increases. The computation time is proportional to the cube of the number of the vertices because the most time-consuming procedure in O2I is the calculation of the eigenvectors of the Laplacian matrix of G'. Figure 14 shows that O2I is practical for graphs G' with more than 10,000 $\times$ 10 vertices.

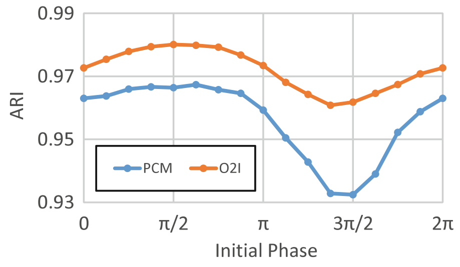


Fig. 15 ARIs for various values of φ

Figure 15 shows the ARIs for O2I and PCM when φ increases from 0 to 2π. The ARI for PCM substantially decreases around $\varphi = \frac{3}{2}\pi$. When $\varphi = \frac{3}{2}\pi$, the distributions of the points significantly overlap at $t = 1$, 5, and 9, because the means of the distributions approach one another. In particular, when the clusters for $G^{(1)}$ are not suitable, the ARI value decreases substantially because the unsuitability propagates through the clusters for $G^{(t)}$ with $t > 1$. In contrast, the ARI for O2I is better than the ARI for PCM for all φ, although the ARI decreases slightly around $\varphi = \frac{3}{2}\pi$. Hence, the results confirm that O2I overcomes the first drawback of PCM.

4.2.2 Varying Cluster Numbers

In this experiment, the means of the two small latent clusters out of the three clusters are located at

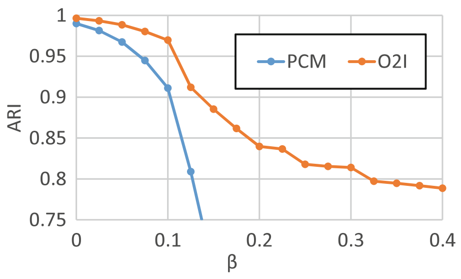
$$\begin{pmatrix} x \\ y \end{pmatrix} = R\left(\frac{2\pi(j-1)}{k}\right) r \begin{pmatrix} \exp\left[\beta(t-T)\right] \\ 0 \end{pmatrix}, \tag{11}$$

rather than the points in Eq. (10). Although the means of the two clusters are close to each other at time 1, they diverge exponentially and move toward a circle of radius r whose center is the origin at time T. By assuming that the points generated from distributions whose means are closer than $2var$ belong to the same cluster, we generate an artificial graph sequence where one of the latent clusters divides into two latent clusters. In this experiment, m is set to 0.

Figure 16 shows the ARIs for O2I and PCM as β increases from 0 to 0.4. When β is low, the ARI for PCM is high because the three latent clusters are separate from one another in $G^{(1)}$ and remain separate until time T. However, when β is high, the ARI value decreases because PCM partitions each graph $G^{(t)}$ into three clusters for small t even though the number of latent clusters is 2. In contrast, O2I partitions each graph into k or fewer clusters because it first converts the graph sequence into G' and then solves the k partition problem for G'. Thus, O2I partitions each graph $G^{(t)}$ into two clusters for small t and into three clusters for large t. Hence, it detects suitable clusters for various values of β.

Figure 17 shows three sets of points: One of the sets is derived from Eq. (10) and the others are derived from Eq. (11). Each point is colored according to cluster

Fig. 16 ARIs for various values of β



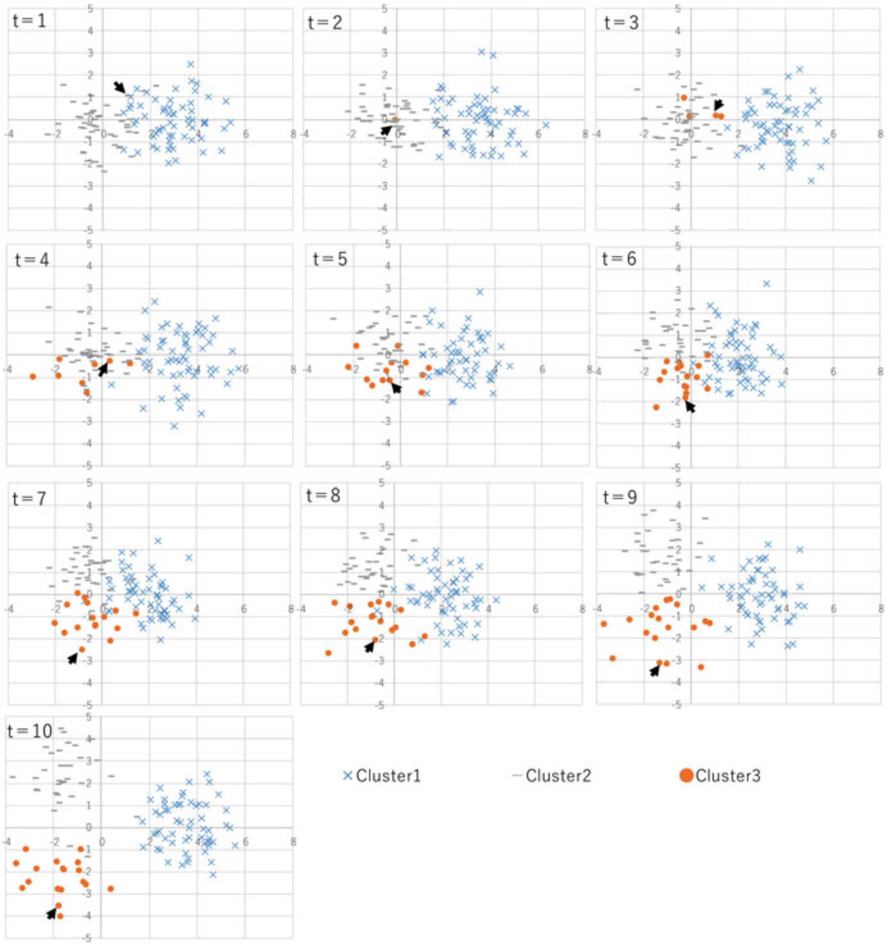


Fig. 17 Distribution of vertices in detected cluster sequences

sequences detected by O2I. This result was obtained for $n = 110$ and $\beta = 0.35$. The points with arrows are points with the same ID. Because cluster 1 vibrates in a sinusoid, the number of points is almost the same over time. In contrast, although the number of points in detected cluster 3 is 0 at time 1, the number of vertices increases and becomes 18 at the last timestep.[3] It is a difficult task to detect the three different clusters at time 2 by conventional methods because the point detected as a point in cluster 3 at time 2 exists near the centroid of cluster 2. However, the second term of Eq. (5) enables O2I to detect the three clusters. This figure indicates that O2I detects cluster sequences in which one cluster divides into two clusters.

In this experiment, we generated artificial graph sequences where one of the latent clusters divides into two latent clusters. O2I does not depend on the direction of the temporal axis because it converts the graph sequence into graph G' and solves the k partition problem for G'. If Eq. (12) is used instead of Eq. (11), we can generate an artificial graph sequence in which two latent clusters merge.

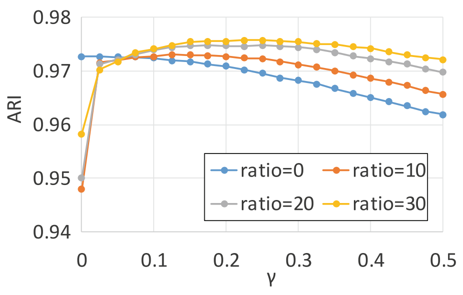
$$\begin{pmatrix} x \\ y \end{pmatrix} = R\left(\frac{2\pi(j-1)}{k}\right) r \begin{pmatrix} \exp\left[-\beta(t-1)\right] \\ 0 \end{pmatrix} \tag{12}$$

In this case, the same result is obtained for O2I as in Fig. 16. Therefore, we have confirmed that O2I overcomes the second drawback of PCM.

4.2.3 Varying Numbers of Vertices

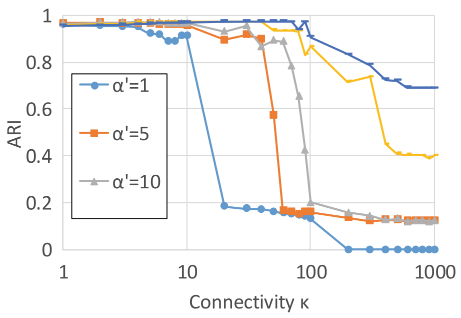
Figure 18 shows the ARIs for O2I for artificial data with $ratio\%$ of the vertices removed from a graph sequence generated using Eq. (10). In this experiment, we measure the ARIs for forgetting rates γ equal to 0, 0.1, 0.2, 0.3, 0.4, and 0.5, as $ratio$ increases from 0 to 30. The setting for $ratio = 0$ and $\gamma = 0$ is the same as in the experiments for Fig. 13. Figure 18 does not contain any results for PCM because PCM cannot be applied to this data. When γ is increased, then ARI increases except

Fig. 18 ARIs for various values of $ratio$ and γ



[3]The numbers of vertices in the third detected cluster sequence increases with time as $\langle 0, 1, 4, 8, 13, 16, 16, 18, 18, 18 \rangle$.

Fig. 19 AIRs for various values of α' and κ (1)



for $ratio = 0$ because vertices with the same ID that are Δ timesteps apart are connected by an edge weight $\alpha'\gamma^{\Delta-1}$ and requirement (2) is satisfied. In particular, increasing γ from 0 to 0.025 is the most effective. When γ is increased further, the ARI decreases because requirement (2) is oversatisfied. This result is consistent with Sect. 4.2.1 In contrast, because α' for $ratio = 0$ is sufficiently tuned in the experiment of Sect. 4.2.1. The ARI decreases when γ is increased. Hence, these results confirm that O2I overcomes the third drawback of PCM.

For $\gamma \geq 0.1$, the reason why ARI for large $ratio$ is more than for small $ratio$ is as follows. Because the graph $G^{(t)}$ for $ratio = 0$ and $\gamma = 0$ has about $\kappa|V^{(t)}|/2$ edges and it connects to $G^{(t+1)}$ with $|V^{(t)}|$ edges, the former connectivity is higher than the latter. Their connectivities are not balanced. In contrast, when $ratio$ or γ is increased, both connectivities are balanced and the ARI increases. In the next subsections, we further investigate effect of other parameters on connectivities of graphs in graph sequences.

4.2.4 Graph Connectivities

Figure 19 shows the result of ARIs when α' and κ are changed. When κ is increased for a certain α', the ARI decreases. This is because the connectivity among vertices coming from the same timestep becomes dense by increasing κ, and cutting edges with weights α' in G' minimizes the objective function Eq. (5) of O2I compared with cutting edges among the vertices coming from the same timestamp. Figure 20 shows the maximum ARIs and their corresponding α' obtained by setting α' to 1, 5, 10, 50, 100, 500, 1000, 5000, and 10,000 for each κ. Because vertices in the latent clusters in G' rarely connect with one another for small κ, the ARI increases when κ is increased. When κ is further increased, the ARI decreases drastically. When κ is greater than 100, the tuned α' is greater than 500. In this case, most of the clusters in the obtained cluster sequences satisfy $C_j^{(t)} = C_j^{(t+1)}$, which means that the cluster sequences do not change over time. Thus, although the effectiveness of O2I decreases when κ becomes too large, we can improve its effectiveness by making the graphs in the graph sequences sparse.

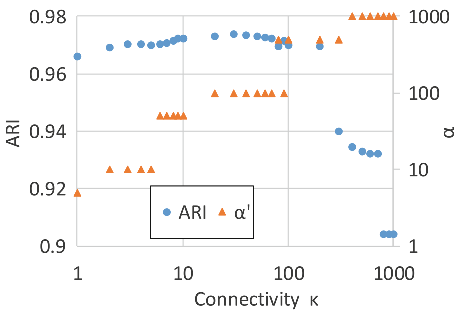


Fig. 20 AIRs for various values of α' and κ (2)

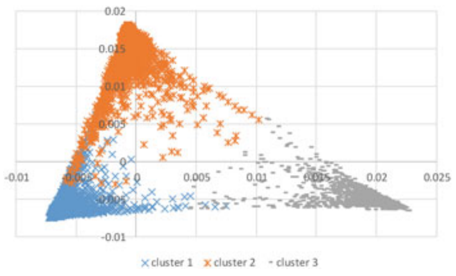


Fig. 21 Distribution of $\mathbf{y}_i$ for $\alpha' = 50$ and $\kappa = 10$

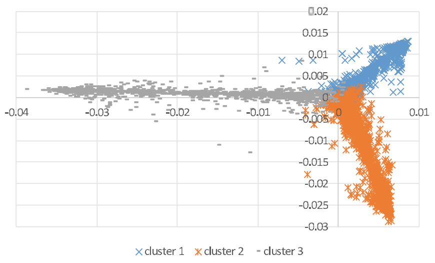


Fig. 22 Distribution of $\mathbf{y}_i$ for $\alpha' = 50$ and $\kappa = 1000$

As mentioned in Algorithm 2, O2I contains spectral clustering. In general, when the spectral clustering is applied to a certain graph G_1 consisting of k connected components, all vertices in the jth $(1 \leq j \leq k)$ connected component are mapped to the same point $\mathbf{p}_j$ in the k-dimensional space in Line 6 of Algorithm 2. In addition, when the spectral clustering is applied to another connected graph G_2 created from G_1 by adding some edges with small weights, all of the vertices that formed the jth $(1 \leq j \leq k)$ connected component in G_1 are mapped to points $\mathbf{y}_i$ around the point $\mathbf{p}_j$ in k-dimensional space [18].[4] Because k-means is applied to points $\mathbf{y}_i$ around this point $\mathbf{p}_j$, the spectral clustering adequately detects clusters in which the vertices are connected with large weights to one another in G_2. Figures 21 and 22 show the distributions of $\mathbf{y}_i$ derived from artificially generated graph sequences for $\kappa = 10$ and $\kappa = 1000$, respectively. Although $\mathbf{y}_i$ are 3-dimensional vectors, we use their second and third elements to plot the distributions because their first

[4]Vector $\mathbf{y}_i$ is the same symbol used in Algorithm 2.

elements are the same. Each point is colored according to the latent clusters to which the point belongs. In Fig. 21, because points in each cluster are distributed around either (0.00, 0.015), (−0.005, −0.005), or (−0.02, 0.005), k-means detects the latent clusters accurately. In this figure, the reason why the distributions for clusters 1 and 2 overlap is because m points at each timestep move from cluster 1 to cluster 2. In contrast, in Fig. 22, the distributions of the three clusters overlap around the origin. This is because vertices coming from the same timestamps connect to each other, vertices coming from the different timestamps connect with large weights α' and G' becomes a large connected component. In this case, k-means cannot partition the points around the origin accurately, and the ARI of O2I decreases.

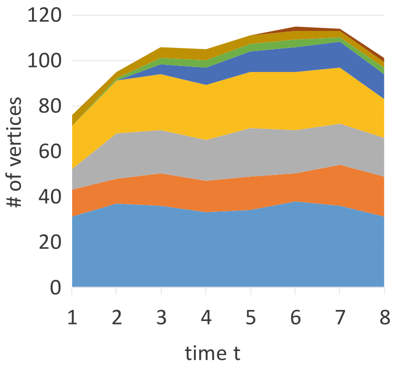
The above experiments confirm that κ is an important hyperparameter of O2I that enables the method to cluster graph sequences accurately. When a graph sequence consisting of dense graphs is given as input, we must select edges with large weights in the graphs to make the graphs sparse before line 1 of Algorithm 2. Making graphs sparse is easier than making graphs dense.

4.2.5 Real-World Data

To assess the practicality of O2I, we applied it to the Enron e-mail dataset [13]. We divided the dataset into T periods according to timestamps of e-mails, assigned a unique ID to the e-mail address for each person participating in the communication, and assigned an edge to a pair of individuals if they communicate via e-mail within each period, assigned weight $\log(c+1)$ to the edge between two vertices if c e-mails are sent between the corresponding individuals, and obtained graphs $G^{(t)}$ for each period t.

Figure 23 shows the number of vertices in the clusters detected by O2I at each timestep. This result was obtained from a graph sequence with eight steps in which the vertices correspond to about 150 senior managers. The hyperparameters for O2I were set as $k = 24$, $\alpha' = 12$, and $\gamma = 0.1$, and O2I took about 5 s to obtain this result. For the sake of visibility, we omitted any

Fig. 23 Number of vertices in each cluster at time t



detected outliers having only a few vertices from the figure. The reason why O2I detects the outliers is that there are many senior managers who each contacted a particular senior manager in the dataset. This figure shows that O2I is applicable to a graph sequence where the number of vertices varies over time, with $(|V^{(1)}|, |V^{(2)}|, \ldots, |V^{(8)}|) = (79, 99, 113, 113, 114, 123, 124, 109)$. In addition, the number of clusters detected by O2I also varies over time, with $(|P^{(1)}|, |P^{(2)}|, \ldots, |P^{(8)}|) = (8, 9, 12, 14, 9, 15, 17, 16)$. As shown in Fig. 23, the cluster represented by dark blue appears at time 3 and then gradually grows larger, although it does not exist at times 1 and 2.

We also applied O2I to the same graph sequence for the approximately 150 managers, with $\gamma = 0$ and the other hyperparameters set as before. In the detected cluster sequences, more than 90% of the vertices belong to a particular cluster sequence at each time and the rest of the vertices belong to $k-1$ outliers. Therefore, the forgetting rate γ in O2I is beneficial for obtaining suitable cluster sequences.

5 Conclusion

In this chapter, we explained O2I for clustering in evolving graphs that can detect changes in clusters over time. In O2I, the graph sequence is partitioned into smooth clusters, even when the numbers of clusters and vertices vary. The method first constructs a graph from the graph sequence, then uses spectral clustering and the RatioCut to apply k partitioning to this graph. The method approach was compared in detail with the preserving clustering membership (PCM) algorithm, which is a conventional online graph-sequence clustering algorithm in which the numbers of clusters and vertices must remain constant. We further showed that, in contrast to PCM, the performance of O2I is not dependent on the clustering of the initial graph in the graph sequence. Experiments on synthetic evolving graphs showed that O2I is practical to calculate and addresses the main disadvantages of PCM. Further tests on real-world data showed that O2I can obtain reasonable clusters. It is hence a flexible clustering solution and will be useful on a wide range of graph-mining applications in which the connections, number of clusters, and number of vertices of the graphs evolve over time.

References

1. Aggarwal, C.C., Han, J., Wang, J., Philip S.Y.: A framework for clustering evolving data streams. In: Proceedings of International Conference on Very Large Data Bases (VLDB), pp. 81–92 (2003)
2. Aggarwal, C.C., Han, J., Wang, J., Philip S.Y.: A framework for projected clustering of high dimensional data streams. In: Proc. of International Conference on Very Large Data Bases (VLDB), pp. 852–863 (2004)

3. Aggarwal, C.C., Han, J., Wang, J., Philip S.Y.: On demand classification of data streams. In: Proceedings of International Conference on Knowledge Discovery and Data Mining (KDD), pp. 503–508 (2004)
4. Bar-Joseph, Z., Gerber, G.K., Gifford, D.K., Jaakkola, T.S., Simon, I.: A new approach to analyzing gene expression time series data. In: Proceedings of International Conference on Computational Biology (RECOMB), pp. 39–48 (2002)
5. Beringer, J., Hüllermeier, E.: Online clustering of parallel data streams. Data Knowl. Eng. **58**(2), 180–204 (2006)
6. Berlingerio, M., Bonchi, F., Bringmann, B., Gionis, A.: Mining graph evolution rules. In: Proceedings of European Conference on Machine Learning and Knowledge Discovery in Databases (ECML/PKDD), pp. 115–130 (2009)
7. Cao, F., Ester, M., Qian, W., Zhou, A.: Density-based clustering over an evolving data stream with noise. In: Proceedings of SIAM International Conference on Data Mining (SDM), pp. 328–339 (2006)
8. Chakrabarti, D., Kumar, R., Tomkins, A.: Evolutionary clustering. In: Proceedings of International Conference on Knowledge Discovery and Data Mining (KDD), pp. 554–560 (2006)
9. Charikar, M., O'Callaghan, L., Panigrahy, R.: Better streaming algorithms for clustering problems. In: Proceedings of Annual ACM Symposium on Theory of Computing (STOC), pp. 30–39 (2003)
10. Chi, Y., Song, X., Zhou, D., Hino, K., Tseng, B.L.: On evolutionary spectral clustering. ACM Trans. Knowl. Discov. Data **3**(4), 17:1–17:30 (2009)
11. Domingos, P.M., Hulten, G.: A general method for scaling up machine learning algorithms and its application to clustering. In: Proceedings of International Conference on Machine Learning (ICML), pp. 106–113 (2001)
12. Inokuchi, A., Washio, I.: Mining frequent graph sequence patterns induced by vertices. In: Proceedings of SIAM International Conference on Data Mining (SDM), pp. 466–477 (2010)
13. Klimmt, B., Yang, Y.: Introducing the Enron corpus. In: CEAS Conference (2004)
14. Möller-Levet, C.S., Klawonn, F., Cho, K.-H., Yin, H., Wolkenhauer, O.: Clustering of unevenly sampled gene expression time-series data. Fuzzy Sets Syst. **152**(1), 49–66 (2005)
15. O'Callaghan, L., Meyerson, A., Motwani, R., Mishra, N., Guha, S.: Streaming-data algorithms for high-quality clustering. In: Proceedings of International Conference on Data Engineering (ICDE), pp. 685–694 (2002)
16. Okui, S., Osamura, K., Inokuchi, A.: Detecting smooth cluster changes in evolving graphs. In: Proceedings of International Conference on Machine Learning and Applications (ICMLA), pp. 369–374 (2016)
17. van Wijk, J.J., van Selow, E.R.: Cluster and calendar based visualization of time series data. In: Proceedings of IEEE Symposium on Information Visualization (INFOVIS), pp. 4–9 (1999)
18. von Luxburg, U.: A tutorial on spectral clustering. Stat. Comput. **17**(4), 395–416 (2007)
19. Wang, Y., Liu, S.-X., Feng, J., Zhou, L.: Mining naturally smooth evolution of clusters from dynamic data. In: Proceedings of SIAM International Conference on Data Mining (SDM), pp. 125–134 (2007)