Dynamic clustering of evolving networks: some results on the line

Cristina G. Fernandes, Marcio T.I. Oshiro, Nicolas Schabanel

Dynamic clustering of evolving networks: some results on the line[†]

Cristina G. Fernandes[1], Marcio T.I. Oshiro[1], and Nicolas Schabanel[2]

[1] *Department of Computer Science, University of São Paulo, Rua do Matão, 1010, 05508-090 São Paulo, Brazil.* `http://www.ime.usp.br/~cris/` *and* `http://www.ime.usp.br/~oshiro/`

[2]*CNRS, Université Paris Diderot, LIAFA, Case 7014, 75205 Paris Cedex 13, France; and IXXI, Université de Lyon, 46 allée d'Italie, 69364 Lyon, France.* `http://www.liafa.univ-paris-diderot.fr/~nschaban`

Understanding the dynamics of evolving social/infrastructure networks is a central challenge in many applied areas such as epidemiology, viral marketing, city planification, etc. During the last decade, a massive amount of data has been collected on such networks that still resist to analysis. In this article, we propose to use the data on the dynamics to find better partitions of the network into groups by requiring the groups to be stable over time. For that purpose, we introduce a dynamic version of the k-clustering problem which includes a cost for every point that moves from one cluster to another. We show that this yields in many realistic situations better fitting solutions than optimizing independently various snapshots of the network. We present a first non-trivial exact algorithm for this problem when the points move along a line ; this algorithm runs in polynomial time when k and the time horizon are bounded by a constant. We conclude with a series of surprising results on the complexity of the structure of optimal solutions for the line case.

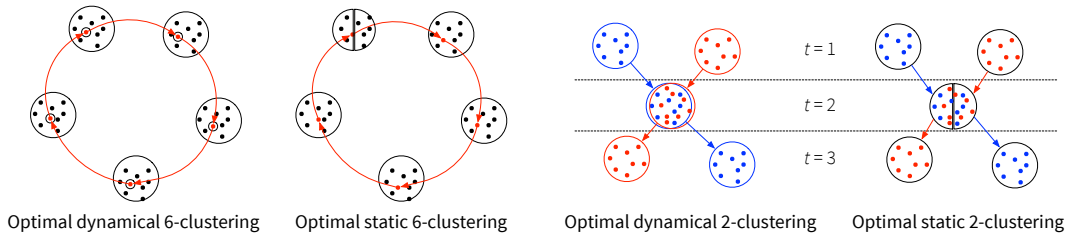
1 Introduction

During the last decade, a massive amount of data has been collected on diverse networks such as web links, nation- or world-wide social networks, online social networks (Facebook or Twitter for example), social encounters in hospitals, schools, companies, or conferences (e.g. [5, 7]), and other real-life networks. Those networks evolve with time, and their dynamics have a considerable impact on their structure and effectiveness (e.g. [6, 4]). Understanding the dynamics of evolving networks is a central question in many applied areas such as epidemiology, vaccination planning, anti-virus design, management of human resources, viral marketing, "facebooking", etc. Algorithmic approaches have for instance been successful in yielding useful insights on several real networks such as zebras social interaction networks [8].

But the dynamics of real-life evolving networks are not yet well understood, partly because it is difficult to observe and analyze such large networks sparsely connected over time. Some basic facts have been observed (such as the preferential attachment or copy-paste mechanisms) but more specific structures remain to be discovered. In this article, we propose to adapt the problem of k-clustering to these evolving networks. We show that requiring the solution to be stable over time yields in many realistic situations better fitting solutions than optimizing independently various snapshots of the network.

More precisely, we focus on the k-clustering problem of points moving in a metric space : we look for the best partition of the points in k groups (called clusters) over time minimizing a tradeoff between two objectives. The first objective is the *span* of the clusters (the sum of their diameter), which ensures that each cluster should contain points which are close to each other. The second objective is the *instability* of the clusters over time, measured as the number of points changing from one cluster to another over time. We argue that incorporating this stability requirement in the objective function helps in many realistic situations to obtain better solutions (see Section 2).

[†]This work was partially supported by the ANR-2010-BLAN-0204 Magnum and ANR-12-BS02-005 RDAM grants, and by CNPq 308523/2012-1 and Proj. MaCLinC of NUMEC/USP.



(a) The classroom. (b) Two groups crossing.

FIGURE 1: Dynamic versus static clustering (some of these examples are from [3]).

Our approach differs from the few existing traditional algorithmic approaches to dynamic settings because they ignore the stability of the solution. We show that offline static algorithms that construct an independent optimal solution for each snapshot of the network yield results that, in a large variety of realistic situations, are not only unstable (and thus arbitrarily bad for our objective), but also undesirable with respect to network dynamics analysis. Online greedy-type solutions such as [1] are also excessively pessimistic : we have access to the whole evolution of the network over time (as given by experiments such as [7]) and we can thus anticipate future changes.

Our results. After introducing the model, we present a series of examples demonstrating that this problem produces solutions which are fitting better to a dynamic network than previous clustering problems. Then, we focus on the case of points moving along the line and exhibit a first non-trivial exact algorithm, which is polynomial when the number of clusters and the time horizon are bounded by a constant. We then present a series of surprising results on the complexity of the structure of optimal solutions in even very restricted cases of the line case.

2 Dynamic k-clustering problem

2.1 Definition

We present a dynamic extension of the problem studied in [2]. Let (X,d) be a metric space. For a given n, an *n-configuration* in X is a sequence of n not necessarily distinct points in X. A *k-clustering* is a sorted partition of $[n] = \{1,\dots,n\}$ into k sets, that is, it is a sequence of disjoint subsets of $[n]$ whose union is $[n]$. Each subset C in a k-clustering is called a *cluster*, and the *diameter* of C for an n-configuration $(p_1,\dots,p_n)$ is the maximum distance between two points in C, that is, $\max\{d(p_i,p_j) : i,j \in C\}$. If C is empty, we say its diameter is zero. The *cost* of a k-clustering for an n-configuration is the sum of the diameters of its clusters.

Given a positive integer T, for each $t = 1,\dots,T$, let $(p_1^t,\dots,p_n^t)$ be an n-configuration in X. Such a sequence of n-configurations is called a *dynamic setting* (of n points) and represents the movement of n points in the metric space (X,d). It is denoted shortly by $P = (p_i^t)_{1\leqslant i\leqslant n, 1\leqslant t\leqslant T}$. A *dynamic k-clustering* $\mathcal{C} = (C_l^t)_{1\leqslant l\leqslant k, 1\leqslant t\leqslant T}$ of the dynamic setting P consists of a sequence of k-clusterings of $[n]$, one for each of the n-configurations $(p_1^t,\dots,p_n^t)$, for $t = 1,\dots,T$.

The cost of a dynamic clustering takes into account two objectives : the *total span* of its clusters and their *instability* over time. The instability of a dynamic clustering can be measured in several ways. In this paper, we adopt the number of times a point changes from a cluster to another over time, that is, the *instability* of a dynamic k-clustering $\mathcal{C} = (C_l^t)_{1\leqslant l\leqslant k, 1\leqslant t\leqslant T}$ is the number of pairs (i,t), for i in $[n]$ and $1 \leqslant t < T$, such that $i \in C_j^t$ but $i \notin C_j^{t+1}$. For a constant $c > 0$, we define the *cost* of $\mathcal{C}$ as c times the instability of $\mathcal{C}$ plus the sum of the cost of each of its clusterings (which happens to be the sum of the diameter of each C_l^t). The constant c allows one to put more or less weight on the instability cost with respect to the clustering cost. A similar formulation of the problem was also introduced in [3].

2.2 Examples

The two examples in Fig. 1 show that dynamic k-clustering differs significantly from static k-clustering of every snapshot and furthermore yields more desirable partitions of the network. Example 1(a) shows the

case of a classroom where the students are split into five groups and the teacher move from one group to the other cyclically. When the number of students is large, an optimal static 6-clustering will isolate four groups and split one in two halves in every snapshots ; whereas the optimal dynamic 6-clustering will isolate every group of students and put the teacher in a sixth cluster, shedding more light on the dynamics of this network. Example 1(b) shows the case of two large groups of people crossing each other (in a street for instance) : an optimal static 2-clustering would first output the two groups, then split the union of the two groups in two halves regardless of the original groups, then split again the two groups ; whereas the dynamic 2-clustering would keep the same groups for the whole time period. Again, the dynamic 2-clustering yields a better understanding of the situation. The following fact generalizes this example to show that the sequence of T optimal static k-clusterings may yield an arbitrarily bad solution to the dynamic problem.

Fact 1 *The ratio of the cost of an optimal dynamic k-clustering of n points and a sequence of optimal static k-clusterings can be as large as* $\Omega(n)$.

3 Exact algorithms for the line

3.1 Dynamic k-clustering on the line

We consider the case where the metric space is the line. That is, the dynamic setting consists of points that move on the line : $p_i^t \in \mathbb{R}$ for all t and i. For all t, let π^t be the permutation of $[n]$ that stably sorts $p_1^t, \ldots, p_n^t$, i.e. such that $p^t_{\pi_1^t} \leqslant \cdots \leqslant p^t_{\pi_n^t}$ and, additionally, if $p^t_{\pi_i^t} = p^t_{\pi_j^t}$ and $i < j$ then $\pi_i^t < \pi_j^t$. (In other words, π_i^t is the index of the i-th point from the left at time t.)

Lemma 2 *The cost of any optimal dynamic k-clustering for a dynamic setting of n points on the line of time-length T is fully characterized by the indices of the two extreme points of each cluster. Moreover, a corresponding optimal dynamic k-clustering can be recovered in* $O(nk^2T)$ *time by dynamic programming.*

We are thus left with enumerating all the possible extremities for the k clusters. There are less than $\binom{n}{k}^2 \leqslant n^{2k}/k!^2$ choices for $t = 1$ and less than $k!\binom{n}{k}^2 \leqslant n^{2k}/k!$ choices for $t \geqslant 2$ (since the labels of the cluster matter as soon as $t \geqslant 2$ as the following facts will demonstrate). It follows that :

Theorem 3 *There is an* $O(n^{2kT+1}k^2T/k!^{T+1})$*-time algorithm that solves exactly the dynamic k-clustering problem for a dynamic setting of n points on the line for T units of time. This running time is polynomial in n if k and T are bounded by a constant.*

Note that this algorithm always outperforms the brute-force enumeration of all the $k^{nT}/k!$ possible k-clustering sequences (running in $O(k^{nT}kT)$ time).

3.2 Surprising facts on optimal dynamic clusterings on the line

For each t, an *interval* (k,n)*-cover* of n points at time t is a sequence of k pairs $(a_l^t, b_l^t)_{l=1..k}$ such that a_l^t, $b_l^t \in [n]$, the a_l^t are pairwise distinct for $l \in [k]$, the b_l^t are pairwise distinct for $l \in [k]$, $a_l^t \leqslant b_l^t$, and $\cup_{l=1}^k [a_l^t, b_l^t] \supseteq [n]$. We say $(a_l^t, b_l^t)_{l=1..k,t=1..T}$ is an *interval* (k,n)*-cover sequence* if, for $t = 1, \ldots, T$, each $(a_l^t, b_l^t)_{l=1..k}$ is an interval (k,n)-cover of the n points at time t. By Lemma 2, a dynamic k-clustering on $[n]$ can be represented by an interval (k,n)-cover.

The following lemma shows that almost every interval (k,n)-cover sequence can correspond to an optimal dynamic $(k+1)$-clustering of a dynamic setting of n points on the line (including sequences where all intervals are concentric !).

Lemma 4 *For every* $n \geqslant 2k$ *and every interval* (k,n)*-covers sequence* $(a_l^t, b_l^t)_{l=1..k,t=1..T}$*, there exists a dynamic setting of n points on the line of time-length* $T+1$ *for which, for* $t = 1..T$*,* a_l^t *and* b_l^t *for* $l = 1..k$ *are the extremities of the k first clusters of the only optimal dynamic* $(k+1)$*-clustering (of the optimal dynamic k-clustering when* $n = 2k$*).*

We restrict attention to the particular case in which the trajectories of the n points in the line do not cross each other. That is, the permutation π^t is the identity at every time t.

One may hope that, if the trajectories of the n points never cross, the k clusters may keep the same relative order. This is unfortunately not the case, as the next example shows.

Fact 5 *There exists a dynamic setting of five points on the line with non-crossing trajectories for which the order of the clusters in the only optimal dynamic* 2*-clustering changes. This can be generalized to a dynamic setting of n points on the line with* $n \geqslant 5$*,* $k \geqslant 2$*, and* $T \geqslant 2$*.*

So, even when the trajectories are non-crossing, it is not straightforward to design a dynamic programming algorithm that runs in time polynomial in n, k, and T. Non-crossing trajectories have however a simpler structure for $T = 2$ or $k = 2$, but, surprisingly, not as soon as $k \geqslant 4$ and $T \geqslant 3$! An interval (n,k)-cover is non-overlapping if its intervals are pairwise disjoint.

Fact 6 *Consider a dynamic setting of n points on the line having non-crossing trajectories. If* $T = 2$ *or* $k = 2$*, then there is an optimal dynamic k-clustering which consists in a sequence of non-overlapping interval* (n,k)*-covers.*

Fact 7 *There is an instance with* $n = 7$ *points,* $k = 4$ *clusters, and* $T = 3$*, with non-crossing trajectories, for which* all *optimal dynamical* 4*-clusterings have two clusters that overlap at some time t.*

When trajectories are non-crossing, the span of a dynamic k-clustering may have some *laminar structure*, i.e., at each time step the spans of every pair of clusters are either disjoint or included one into the other, and moreover with every point belonging to the cluster with smallest span that contains it. If this were true, then we would have a much better algorithm, running in polynomial time in n and *linear time* in T.

Theorem 8 *When trajectories are non-crossing, there is an algorithm that computes an optimal laminar dynamic k-clustering in* $O((k!)^2 n^{4k-4} T)$ *time and* $O(k!\, n^{2k-2} T)$ *space.*

4 Conclusion

We believe that this problem is a key to the understanding of real-life evolving network, such as social interactions. It appears to have very intriguing behavior even in the simpler case of the line. Although running in polynomial time for fixed k and T, our algorithms need still to be improved to be fitted for real-life data collections which are typically huge. One may thus consider aiming at faster algorithms approximating the optimum. In other ongoing work, [3] develops an approximation algorithm and suggests that there might be some strong inapproximability results related to this problem but the situation needs yet to be clarified.

Références

[1] M. Charikar, C. Chekuri, T. Feder, and R. Motwani. Incremental clustering and dynamic information retrieval. In *STOC'97*, pages 626–635, 1997.

[2] Moses Charikar and Rina Panigrahy. Clustering to minimize the sum of cluster diameters. In *STOC'01*, pages 1–10, 2001.

[3] David Eisenstat, Claire Mathieu, and Nicolas Schabanel. Ongoing work on dynamic clustering of evolving graph, 2012.

[4] Jon M. Kleinberg. The small-world phenomenon and decentralized search. *SIAM News*, 37(3), 2004.

[5] Mark E. J. Newman. The structure and function of complex networks. *SIAM Review*, 45(2) :167–256, 2003.

[6] Romualdo Pastor-Satorras and Alessandro Vespignani. Epidemic spreading in scale-free networks. *Physical Review Letters*, 86 :3200–3203, 2001.

[7] J. Stehlé, N. Voirin, A. Barrat, C. Cattuto, L. Isella, J.-F. Pinton, M. Quaggiotto, W. Van den Broeck, C. Régis, B. Lina, and P. Vanhems. High-resolution measurements of face-to-face contact patterns in a primary school. *PLoS ONE*, 6(8) :e23176, 2011.

[8] C. Tantipathananandh, T. Y. Berger-Wolf, and D. Kempe. A framework for community identification in dynamic social networks. In *KDD*, pages 717–726, 2007.