Dense Temporal Subgraphs in Protein-Protein Interaction Networks

Riccardo Dondi[1], Mohammad Mehdi Hosseinzadeh[1], and Italo Zoppis[2]

[1] Università degli Studi di Bergamo, Bergamo, Italy,
riccardo.dondi@unibg.it; m.hosseinzadeh@unibg.it
[2] Università degli Studi di Milano-Bicocca, Milano, Italy,
zoppis@disco.unimib.it

Abstract. Temporal networks have been successfully applied to represent the dynamics of protein-protein interactions. In this paper we focus on the identification of dense subgraphs in temporal protein-protein interaction networks, a relevant problem to find group of proteins related to a given functionality. We consider a drawback of an existing approach for this problem that produce large time intervals over which temporal subgraphs are defined. We propose a problem to deal with this issue and we design (1) an exact algorithm based on dynamic programming which solves the problem in polynomial time and (2) a heuristic, based on a segmentation of the time domain and the computation of a refinement. The experimental results we present on seven protein-protein interaction networks show that in many cases our heuristic is able to reduce the time intervals with respect to those computed by the existing methods.

Keywords: Network mining · Temporal Graphs · Protein Protein Interaction Networks · Densest Subgraphs · Algorithms

1 Introduction

The interactions between biological elements (genes or proteins, for example) are usually represented and analysed with a network/graph. Recently, the research focus has shifted to the evolution over time of the interactions. A model introduced to represent this evolution is that of temporal network (or temporal graph) [11, 14]. This model enriches the classical graph one by defining when edges are active over a discrete sequence of timestamps. The analysis of temporal networks has provided valuable insights about their structure and properties, notable examples being community detection [16, 4, 8] and frequent subgraph discovery [19, 15].

Protein-Protein Interaction (PPI) networks have been deeply studied in bioinformatics. In a PPI network, vertices represent gene coding proteins, edges interaction among proteins (for example co-expressed protein pairs). Since interactions among proteins change over time, their dynamic evolution has been analyzed considering temporal networks [9].

One of the most relevant problem in (static or temporal) networks is the identification of cohesive subgraphs. In PPI networks, cohesive subgraphs represent

ICCS Camera Ready Version 2022
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-08754-7_53

protein complexes involved in a biological process. Several models of cohesive subgraphs have been considered in literature, like cliques, s-plexes or s-clubs. A central model of cohesive subgraph applied in literature is that of densest subgraph, that defines a cohesive subgraph as one that maximizes the density (ratio between the number of edges and the number of vertices). The densest subgraph problem can be solved in polynomial-time with Goldberg's algorithm [10] and it can also be approximated within factor $\frac{1}{2}$ in linear-time [1, 3].

In many network mining applications, finding a single dense subgraph doesn't provide enough information on the network structure. Therefore, other approaches have been proposed, a notable example being Top-k-Overlapping Densest Subgraphs, introduced in [17, 7]. This problem asks for a set of k densest subgraphs, with $k \geq 1$, that may share vertices. The problem aims at maximizing an objective function that contains the sum of densities of the k subgraphs and a distance between them. The Top-k-Overlapping Densest Subgraphs problem has been recently applied to biological networks [12] and to dual networks [6].

The identification of densest subgraphs in temporal networks has been considered to analyze social networks in [17, 18] has a key to identify interesting episodes, like groups of users of a platform highly interacting in a temporal interval. In [17, 18] it is introduced the k-Densest-Temporal-Subgraphs problem whose goal is the identification of $k \geq 2$ densest temporal subgraphs in disjoint time intervals. In [17, 18], it is shown that the problem can be solved in polynomial time via dynamic programming, and Goldberg's algorithm. However, due of its time complexity, the algorithm is not applicable even for medium-size temporal networks [17], hence heuristics and approximation algorithms have been considered [17, 5, 2].

One of the drawbacks of the k-Densest-Temporal-Subgraphs approach propsoed in [17] is that optimal solutions for the problem usually induce a segmentation, that is each timestamp of the time domain belongs to an interval of the solution. It follows that a solution of k-Densest-Temporal-Subgraphs: (1) May contain temporal subgraphs that are defined over large intervals; (2) Some vertices or edges may incorrectly be included in a subgraph of the solution.

Here we consider an approach (called k-Densest-Temporal-Refinement) that aims at correcting this drawback of the k-Densest-Temporal-Subgraphs model and that aims at identifying whether there exist temporal graphs of high density and defined over smaller intervals with respect to the ones of a solution of k-Densest-Temporal-Subgraphs. The idea is that if a dense temporal subgraph contains a temporal subgraph active in a smaller interval and with a close density, then this latter better represents the real cohesive group of elements.

We first present a dynamic programming algorithm that is able to solve the problem in polynomial time. However, the time complexity of this algorithm makes it not applicable even on small size graphs. Thus we design a heuristic for the problem and we present an experimental evaluation on a set of seven PPI networks. The experimental results show that in many cases our approach is able to reduce the length of the intervals, while maintaining the density significantly high.

ICCS Camera Ready Version 2022
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-08754-7_53

The paper is organized as follows. First in Section 2 we give the definitions and we introduce the problems we are interested into. Then, in Section 3 we present a dynamic programming algorithm for the k-Densest-Temporal-Refinement problem. In Section 4 we present an efficient heuristic for the problem, while in Section 5 we present an experimental evaluation on seven PPI temporal networks.

2 Definitions

A time domain $\mathcal{T} = [t_1, t_2, \ldots, t_z]$, where $t_1 < t_2 < \cdots < t_z$, is a sequence of positive integer, each one called timestamp. An interval $[t_i, t_j]$ in $\mathcal{T} = [t_1, t_2, \ldots, t_z]$ is the sequence of timestamps between t_i, t_j, with $t_i \leq t_j$. $T' = [t_a, t_b]$ is a subinterval of $T = [t_c, t_d]$, denoted by $T' \subseteq T$, if $t_c \leq t_a \leq t_b \leq t_d$ and $t_c < t_a$ or $t_b < t_d$. Now, we can introduce the definition of temporal graphs.

Definition 1. *$G = (V, \mathcal{T}, E)$ is a temporal graph, where V is a set of vertices, $\mathcal{T}$ is a time domain and E is a set of triple $\{u, v, t\}$, where $u, v \in V$ and $t \in \mathcal{T}$.*

In the following we denote by z the number of timestamps in $\mathcal{T}$ (that is $z = |\mathcal{T}|$), by n the number of vertices in V (that is $n = |V|$) and by $|m|$ the number of edges in E (that is $m = |E|$). We give now the definition of induced temporal subgraph.

Definition 2. *Given a temporal graph $G = (V, \mathcal{T}, E)$, an induced temporal subgraph of G is defined as a pair $G[T, V']$, where*

- *T is an interval in $\mathcal{T}$ over which the induced temporal subgraph is defined*
- *$V' \subseteq V$ is the set of vertices of the induced temporal graph.*

Notice that the edge set of $G[T, V']$, denoted by $E(G[T, V'])$, is defined as follows:

$$E(G[T, V']) = \{\{u, v, t\} : u, v \in V', t \in T\}.$$

Given an induced temporal subgraph $G[T, V']$ and a timestamp t that belongs to T, we say that $G[T, V']$ covers t. The density of an induced subgraph $G[T, V']$, denoted by $dens(G[T, V'])$, is defined as follows:

$$dens(G[T, V']) = \frac{|E(G[T, V'])|}{|V'|}.$$

Given an interval I in $\mathcal{T}$, we denote by $MaxD(I)$ the density of a densest subgraph of G defined in interval I.

The first problem we consider, called k-Densest-Temporal-Subgraphs, introduced in [17], is defined as follows.

ICCS Camera Ready Version 2022
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-08754-7_53

Problem 1. k-Densest-Temporal-Subgraphs
Input: A temporal graph $G = (V, \mathcal{T}, E)$, a positive integer $k \geq 2$.
Output: A set S of k temporal subgraphs, $S = \{G[T_1, V_1], \ldots, G[T_k, V_k]\}$, such that $T_1, \ldots, T_k$ are disjoint intervals and

$$\sum_{i=1}^{k} dens(G[T_i, V_i])$$

is maximized.

As discussed in [17], the density of an interval is a non decreasing function with respect to the length of an interval. Thus there exists an optimal solution of k-Densest-Temporal-Subgraphs that induces a segmentation of the time domain, that is each timestamp of the time domain $\mathcal{T}$ is covered by exactly one temporal subgraph of the solution. The methods proposed in literature [5, 17] compute indeed solutions that induce a segmentation, and they may contain subintervals that gives a small contribution to the density.

Here we propose an approach to possibly shrink interval length. We start by introducing the definition of ε-refinement.

Definition 3. *Given an induced temporal subgraph $G[T, U]$ of a temporal graph $G = (V, \mathcal{T}, E)$, $G[T', U']$ is an ε-refinement of $G[T, U]$ if*

1. *T' is a subinterval of T*
2. *$dens(G[T', U']) \geq (1 - \varepsilon)\ dens(G[T, U])$*
3. *There is no ε-refinement of $G[T, U]$ in subintervals of T'*

Notice that by Point 3 of Definition 3, if $G[T', U']$ is an ε-refinement of $G[T, U]$ it is not possible to further reduce the interval T' in order to obtain an ε-refinement of $G[T, U]$.

The definition of ε-refinement can be extended to ε-complete refinement of a solution of k-Densest-Temporal-Subgraphs.

Definition 4. *Given a temporal graph $G = (V, \mathcal{T}, E)$, an ε-complete refinement $RF = \{R_1, \ldots, R_q\}$, where each R_i, $1 \leq i \leq q$, is a quadruple (V_i, T_i, V'_i, T'_i) such that $T_1, \ldots, T_q$ is a segmentation of $\mathcal{T}$, $G[V_i, T_i]$ is a densest subgraph in interval T_i and V'_i, T'_i are defined as follows:*

1. *If there does not exist a ε-refinement of $G[V_i, T_i]$, then $V'_i = V_i$ and $T'_i = T_i$ and the profit of R_i, denoted by $p(R_i)$, is defined as $p(R_i) = (1-\varepsilon)\ dens(G[T_i, V_i])$*
2. *If there exists an ε-refinement of $G[T_i, V_i]$, then $G[T'_i, V'_i]$ is an ε-refinement of $G[T_i, V_i]$ of maximum density; the profit of R_i, denoted by $p(R_i)$, is defined as $p(R_i) = dens(G[T'_i, V'_i])$*

The profit of a ε-complete refinement of $R_1, \ldots, R_q$ of $G_1, \ldots, G_q$ is:

$$\sum_{i}^{k} p(R_i).$$

ICCS Camera Ready Version 2022
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-08754-7_53

Problem 2. k-Densest-Temporal-Refinement
Input: A temporal graph $G = (V, \mathcal{T}, E)$, a positive integer $k \geq 2$.
Output: An ε-complete refinement RF having maximum density.

Algorithms for Computing a Densest Subgraph

A densest subgraph in a given static graph can be computed in $O(mn \log n)$ time by reducing the problem to min-cut with Goldberg's algorithm [10]. When $mn \leq n^3$, the time complexity of Goldberg's algorithm for unweighted graphs has been improved to $O(n^3)$ [13]. Finding a densest subgraph can be approximated within factor $\frac{1}{2}$ in linear time with a greedy algorithm (here called Charikar's algorithm) [3].

3 An Exact Algorithm for k-Densest-Temporal-Refinement

In this section, we present a dynamic programming algorithm for the k-Densest-Temporal-Refinement problem. First, define $P(j,h)$, $1 \leq j \leq z$ and $1 \leq h \leq k$, as a function that is equal to the maximum profit of an ε-complete refinement consisting of h subgraphs until timestamp $t_j \leq t_z$.

Given two timestamps t_i and t_j, with $1 \leq t_i \leq t_j \leq t_z$, we denote by $Pr([i,j])$ the maximum profit of a temporal subgraph of G defined in interval $[t_i, t_j]$. Now, $P(j,h)$ can be computed as follows:

$$P(j,h) = \begin{cases} \max_{1 \leq i < j} P(i, h-1) + Pr([i+1, j]) & \text{if } 2 \leq h \leq z \\ Pr([1,j]) & \text{if } h = 1 \end{cases} \quad (1)$$

The computation of $P(j,h)$ with Recurrence 1 requires the computation of $Pr([i,j])$, for each i and j, with $1 \leq i \leq j \leq z$. $Pr([i,j])$ can be computed as follows:

$$Pr([i,j]) = \max \begin{cases} (1-\varepsilon) MaxD([i,j]) \\ \max_{[a,b] \subseteq [i,j]} MaxD([a,b]) \end{cases} \quad (2)$$

Lemma 1. *$P(j,h) = q$ if and only if there exists an ε-complete refinement of G on interval $[1,j]$ consisting of h subgraphs of profit q.*

Proof. We prove the lemma by induction on h. In the base case, when $h = 1$, then the profit of a single temporal subgraph in the interval $[t_1, t_j]$ is equal to $Pr[1,j]$.

Assume that the lemma holds for each $1 \leq i < j$, we prove that it holds for j. Assume that $P(j,h) = q$, then by Recurrence 1 there must exists a value $i < j$ such that (1) $P(i, h-1) = q_1$ (2) $Pr([i+1, j]) = q_2$, where $q_1 + q_2$. By induction hypothesis there exists an ε-complete refinement consisting of $h-1$ subgraphs in interval $[1,i]$ of profit q_1 and by definition of Pr, there exists a subgraph in $[i+1, j]$ of profit q_2.

ICCS Camera Ready Version 2022
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-08754-7_53

Assume that there exists an ε-complete refinement S consisting of h subgraphs in interval $[1, j]$ of profit q. Assume that among the subgraphs in S, the subgraph defined in the leftmost interval $[i+1, j]$ has profit $Pr([i+1, j]) = q_2$. Then there exists an ε-complete refinement consisting of $h-1$ subgraphs in interval $[1, i]$ of profit q_1, with $q_1 + q_2 = q$. By induction hypothesis, it holds that $P(i, h-1) = q_1$ and, by the first case of Recurrence 1, it holds that $P(j, h) = q$. □

Theorem 1. k-Densest-Temporal-Refinement *can be solved in* $O(z^4 mn \log(n))$ *time.*

Proof. By Lemma 1 it follows that an optimal solution of k-Densest-Temporal-Refinement on instance $G = (V, E, \mathcal{T})$ is equal to $P(z, k)$. Now, $P(j, h)$, with $1 \leq j \leq z$ and $1 \leq h \leq k$, consists of zk entries. Each entry $P(j, h)$ can be computed in time $O(z)$, assuming the values $Pr([i+1, j])$ are known. Thus, since $k \leq z$, the entries $P(j, h)$, with $1 \leq j \leq z$ and $1 \leq h \leq k$, can be computed in $O(z^3)$ time, assuming the values $Pr([i+1, j])$ are known.

The values $Pr([i, j])$ are $O(z^2)$ and each value can be computed by applying the Goldberg's algorithm on the graph active in interval in $[a, b]$ with $[a, b]$ a subinterval of $[i, j]$. This requires, for each i, j, the computation of $O(z^2)$ densest subgraph via Goldberg's algorithm, so requiring time $O(z^2 mn \log n)$. The overall time complexity to compute an ε-complete refinement is then $O(z^4 mn \log n)$. □

4 A Heuristic of k-Densest-Temporal-Refinement

While k-Densest-Temporal-Refinement is solvable in polynomial time with the dynamic programming algorithm given in Section 3, the computational complexity of this algorithm makes it not practical even for small size temporal networks. Indeed, we applied it to the temporal PPI networks described in Section 5 and it was not able to produce any result within 10 hours.

Thus, we have designed an efficient heuristic, called Reduce, for the k-Densest-Temporal-Refinement problem. Reduce first computes a solution $S = \{G[T_1, V_1], G[T_2, V_2], \ldots, G[T_k, V_k]\}$ of the k-Densest-Temporal-Subgraphs problem, by applying the method described in [5]. Notice that the temporal graphs in S induce a segmentation, of the time domain and that the solution S may not be an optimal solution of k-Densest-Temporal-Subgraphs.

For each temporal subgraph $G[T_i, V_i]$, $1 \leq i \leq k$, of S, with $dens(G[T_i, V_i]) = d$, Reduce looks for an ε-refinement by computing the following three temporal subgraphs:

1. $G[T_{i,1}, V_{i,1}]$ is computed by applying Charikar's algorithm on the temporal subgraph of $G[T_i, V_i]$ obtained by removing the first and the last timestamp of T_i
2. $G[T_{i,2}, V_{i,2}]$ is computed by applying Charikar's algorithm on the temporal subgraph of $G[T_i, V_i]$ obtained by removing the first timestamp of T_i

ICCS Camera Ready Version 2022
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-08754-7_53

3. $G[T_{i,3}, V_{i,3}]$ is computed by applying Charikar's algorithm on the temporal subgraph of $G[T_i, V_i]$ obtained by removing the last timestamp of T_i

Now, if $G[T_{i,1}, V_{i,1}]$ has density at least $(1-\varepsilon)d$ (hence is an ε-refinement of $G[T_i, V_i]$), the three previous steps are applied on $G[T_{i,1}, V_{i,1}]$. If $G[T_{i,1}, V_{i,1}]$ has density smaller then $(1-\varepsilon)d$, and one of $G[T_{i,2}, V_{i,2}]$ $G[T_{i,3}, V_{i,3}]$ has density at least $(1-\varepsilon)d$ (thus there exists an ε-refinement of $G[T_i, V_i]$), the subgraphs having largest density between $G[T_{i,2}, V_{i,2}]$ $G[T_{i,3}, V_{i,3}]$ is selected and the three steps are applied on it.

If none of $G[T_{i,1}, V_{i,1}]$, $G[T_{i,2}, V_{i,2}]$, $G[T_{i,3}, V_{i,3}]$ have density at least $(1-\varepsilon)d$, the algorithm defines V_i', T_i' in $R_i = (V_i, T_i, V_i', T_i')$ as follows: $V_i' = V_i$ and $T_i' = T_i$.

5 Experimental Analysis

In this section, we present the experimental results for our heuristics (Reduce) on real-world datasets. Since the real-world datasets we consider are over a limited number of timestamps (i.e. 36) we have defined small values of k, that is $k = 2$ and $k = 3$, in the k-Densest-Temporal-Refinement problem. We implemented Reduce in Python on MacBook-Pro (OS version 12.0.1) with processor 2.9 GHzIntel Core i5 and 8GB 2133 MHz LPDDR3 of RAM, Intel Iris Graphics 550 1536 MB.

Table 1. PPI temporal networks informations

PPINs Temporal Networks	#Verticess	#Edges	#Timestamps
DPPIN-Babu	5,003	111,466	36
DPPIN-Gavin	2,541	140,040	36
DPPIN-Hazbun	143	1,959	36
DPPIN-Ho	1,548	42,220	36
DPPIN-Ito	2,856	8,638	36
DPPIN-Krogan(LCMS)	2,211	85,133	36
DPPIN-Lambert	697	6,654	36

For testing the performances of Reduce, we have considered seven PPI temporal networks, taken from [9]. The characteristics of the networks are summarized in Table 1. Notice that all the temporal networks we considered are defined over a time domain consisting of 36 timestamps, while the number of vertices and the number of edges vary significantly (the number of vertices ranges from 143 to 5003 and the number of edges ranges from 1959 to 140040).

In Table 2 we report the densities and the intervals of the solutions of k-Densest-Temporal-Subgraphs computed by Reduce (called **first phase**) and the refinement solutions, where $\varepsilon = 0.05$, computed by Reduce (called **second**

ICCS Camera Ready Version 2022
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-08754-7_53

phase). We report also the Jaccard's index similarity[3] of the solutions returned by the two phases of Reduce in order to measure the fraction of shared vertices.

Table 2 shows that for $k = 2$, when Reduce finds an ε-refinement, the subgraph computed by the second phase of Reduce has a Jaccard index of at least 0.94. Similar results are obtained for $k = 3$. Thus the ε-refinement computed in the second phase are very similar to the subgraphs computed in the first phase. On the other hand, there is a significant reduction in the length of the intervals over which the subgraphs returned by the second phase are defined, as discussed next.

In Table 3 we present, for each value of k and for each temporal network considered, the ratio between the densities and the interval lengths of the solutions returned by Phase 2 and Phase 1 of Reduce (if Phase 2 produces an ε-refinement). The results show that the ε-refinements significantly reduce the interval length. For instance for $k = 2$, the ε-refinements computed are defined over intervals of length at most 75% and at least 31% of the interval length of first phase solutions. For $k = 3$, the interval length of ε-refinement solutions are at most 81% and at least 9% of the interval length of first phase solutions.

In Table 4, we report the profit of the solutions of the k-Densest-Temporal-Refinement problem. We report in bold the densities for the cases where Phase 2 is able to computed an ε-refinement, while for the other cases the profit is $(1-\varepsilon)$ of the density of the subgraph returned by Phase 1.

Finally, we evaluate, for each network considered, the minimum value of ε so that Phase 2 of Reduce is able to compute an ε-refinement. In Table 5 we report the ratios between the densities of first and second phases of Reduce by varying ε from 5% to 37% (in bold the ε-refinement when $\varepsilon = 0.05$). Notice that, while the ε-refinements have a ratio close to 1 (for 12 of the 17 0.05-refinements computed the ratio is at least 0.99), for the cases where Reduce is not able to computed an ε-refinement the ratio is always smaller than 0.9. Furthermore, for three networks (DPPIN-Hazbun, DPPIN-Ito, DPPIN-Lambert), the second phase of Reduce was never able to compute an ε-refinement, while for the other networks this happens in at most one case. The three networks DPPIN-Hazbun, DPPIN-Ito, DPPIN-Lambert are also (considered as static graphs) those having lower density (they have a density of at most 13.7, the other graphs a density of at least 22). This suggest that the existence of an ε-refinement may be related to the density of the input graph.

6 Conclusion

We have presented a problem for finding dense subgraphs in temporal networks, with an application to protein-protein interactions. We have designed an exact algorithm based on dynamic programming, that solves the problem in polynomial time, but it is not practical even for small size datasets. We have designed a heuristic, based on a segmentation of the time domain and the computation

[3] Jaccard's index measures the similarity between two sets by taking the ratio of intersection over union of the two sets.

ICCS Camera Ready Version 2022
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-08754-7_53

Table 2. Refinement solution for $\varepsilon = 0.05$, $k = 2$ and $k = 3$ of the first phase.

Datasets		k=2		k=3		
DPPIN-Babu	**first phase**					
	density	78.58	75.99	78.58	60.87	72.15
	interval	(0,15)	(16,35)	(0,8)	(9,27)	(28,35)
	second phase					
	density	78.58	72.16	78.58	–	72.13
	interval	(0,8)	(23,31)	(8,8)	–	(28,31)
	Jaccard index	1	0.98	1	–	0.99
DPPIN-Gavin	**first phase**					
	density	123.41	119.74	123.38	81.07	114.33
	interval	(0,11)	(12,35)	(0,10)	(11,24)	(25,35)
	second phase					
	density	123.38	114.45	123.38	–	114.29
	interval	(0,8)	(23,31)	(0,8)	–	(25,31)
	Jaccard index	1	0.94	0.99	–	1
DPPIN-Hazbun	**first phase**					
	density	19.52	19.36	18.75	15.05	16.33
	interval	(0,13)	(14,35)	(0,10)	(11,20)	(21,35)
	second phase					
	density	–	–	–	–	–
	interval	–	–	–	–	–
	Jaccard index	–	–	–	–	–
DPPIN-Ho	**first phase**					
	density	39.94	39.25	38.91	33.97	38.83
	interval	(0,19)	(20,35)	(0,9)	(10,24)	(25,35)
	second phase					
	density	38.86	38.80	38.91	33.77	38.74
	interval	(0,8)	(31,35)	(5,8)	(18,19)	(6,6)
	Jaccard index	0.94	0.98	1	0.97	1
DPPIN-Ito	**first phase**					
	density	5.01	4.48	4.27	4.48	4.13
	interval	(0,20)	(21,35)	(0,12)	(13,24)	(25,35)
	second phase					
	density	–	–	–	–	–
	interval	–	–	–	–	–
	Jaccard index	–	–	–	–	–
DPPIN-Krogan(LCMS)	**first phase**					
	density	96.82	97.48	96.78	64.98	92.24
	interval	(0,15)	(16,35)	(0,10)	(11,30)	(31,35)
	second phase					
	density	96.71	92.47	96.71	–	92.14
	interval	(0,8)	(22,31)	(0,8)	–	(31,31)
	Jaccard index	1	0.95	1	–	0.99
DPPIN-Lambert	**first phase**					
	density	19.23	20.0	17.89	16.61	15.47
	interval	(0, 15)	(16, 35)	(0, 11)	(12, 24)	(25, 35)
	second phase					
	density	–	–	–	–	–
	interval	–	–	–	–	–
	Jaccard index	–	–	–	–	–

ICCS Camera Ready Version 2022
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-08754-7_53

Table 3. Ratio between the densities and the interval lengths of the solutions returned by Phase 1 and Phase 2 of Reduce for $\varepsilon = 0.05$

Datasets		k=2		k=3		
DPPIN-Babu	density	1	0.95	1	–	0.99
	interval	0.56	0.45	0.11	–	0.5
DPPIN-Gavin	density	0.99	0.96	1	–	0.99
	interval	0.75	0.33	0.81	–	0.64
DPPIN-Hazbun	density	–	–	–	–	–
	interval	–	–	–	–	–
DPPIN-Ho	density	0.97	0.99	1	0.99	0.97
	interval	0.45	0.31	0.4	0.13	0.09
DPPIN-Ito	density	–	–	–	–	–
	interval	–	–	–	–	–
DPPIN-Krogan(LCMS)	density	0.99	0.95	0.99	–	0.99
	interval	0.56	0.5	0.81	–	0.2
DPPIN-Lambert	density	–	–	–	–	–
	interval	–	–	–	–	–

Table 4. Profit of the solutions for $\varepsilon = 0.05$ of the k-Densest-Temporal-Refinement problem

Datasets	K=2		K=3		
DPPIN-Babu	**78.58**	**72.16**	**78.58**	57.83	**72.13**
DPPIN-Gavin	**123.38**	**114.45**	**1**	77.02	**114.29**
DPPIN-Hazbun	18.54	18.39	17.81	14.30	15.51
DPPIN-Ho	**38.86**	**38.80**	**38.91**	**33.77**	**38.74**
DPPIN-Ito	4.76	4.26	4.06	4.26	3.92
DPPIN-Krogan(LCMS)	**96.71**	**92.47**	**96.71**	61.73	**92.14**
DPPIN-Lambert	18.27	19	16.99	15.78	14.70

Table 5. Ratio between the solutions (densities) of the first and second phases - varying ε from from 5% to 37% for refinement

Datasets	K=2		K=3		
DPPIN-Babu	**1**	**0.95**	**1**	0.88	**0.99**
DPPIN-Gavin	**0.99**	**0.96**	**1**	0.88	**0.99**
DPPIN-Hazbun	0.83	0.75	0.86	0.87	0.89
DPPIN-Ho	**0.97**	**0.99**	**1**	**0.99**	**0.97**
DPPIN-Ito	0.78	0.83	0.81	0.81	0.81
DPPIN-Krogan(LCMS)	**0.99**	**0.95**	**0.99**	0.86	**0.99**
DPPIN-Lambert	0.73	0.65	0.77	0.63	0.72

ICCS Camera Ready Version 2022
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-08754-7_53

of a refinement, that allows us to produce dense subgraphs and to reduce the length of the intervals.

Future works include an extension of the experimental part to other PPI networks in order to verify if properties of the networks are related to the existence of an ε-refinement. It would be interesting to understand if the dense subgraphs identified are related to some functionality and, more generally, a biological analysis of the inferred temporal subgraphs. Finally, from an algorithmic point it would be interesting to design other heuristics for the problem.

References

1. Asahiro, Y., Iwama, K., Tamaki, H., Tokuyama, T.: Greedily finding a dense subgraph. J. Algorithms **34**(2), 203–221 (2000). https://doi.org/10.1006/jagm.1999.1062
2. Castelli, M., Dondi, R., Hosseinzadeh, M.M.: Genetic algorithms for finding episodes in temporal networks. Procedia Computer Science **176**, 215–224 (2020)
3. Charikar, M.: Greedy approximation algorithms for finding dense components in a graph. In: Approximation Algorithms for Combinatorial Optimization, Third International Workshop, APPROX 2000, Proceedings. pp. 84–95 (2000)
4. Coscia, M., Giannotti, F., Pedreschi, D.: A classification for community discovery methods in complex networks. Statistical Analysis and Data Mining: The ASA Data Science Journal **4**(5), 512–546 (2011)
5. Dondi, R., Hosseinzadeh, M.M.: Dense sub-networks discovery in temporal networks. SN Comput. Sci. **2**(3), 158 (2021). https://doi.org/10.1007/s42979-021-00593-w
6. Dondi, R., Hosseinzadeh, M.M., Guzzi, P.H.: A novel algorithm for finding top-k weighted overlapping densest connected subgraphs in dual networks. Applied Network Science **6**(1), 1–17 (2021)
7. Dondi, R., Hosseinzadeh, M.M., Mauri, G., Zoppis, I.: Top-k overlapping densest subgraphs: approximation algorithms and computational complexity. Journal of Combinatorial Optimization **41**(1), 80–104 (2021)
8. Fortunato, S.: Community detection in graphs. Physics reports **486**(3-5), 75–174 (2010)
9. Fu, D., He, J.: Dppin: A biological repository of dynamic protein-protein interaction network data. arXiv preprint arXiv:2107.02168 (2021)
10. Goldberg, A.V.: Finding a maximum density subgraph. Tech. rep., Berkeley, CA, USA (1984)
11. Holme, P.: Modern temporal network theory: a colloquium. The European Physical Journal B **88**(9), 234 (2015)
12. Hosseinzadeh, M.M.: A new heuristic to find overlapping dense subgraphs in biological networks. Proceeding of Current Trends in Theory and Practice of Computer Science p. to appear (2020)
13. Kawase, Y., Miyauchi, A.: The densest subgraph problem with a convex/concave size function. Algorithmica **80**(12), 3461–3480 (2018). https://doi.org/10.1007/s00453-017-0400-7
14. Kempe, D., Kleinberg, J., Kumar, A.: Connectivity and inference problems for temporal networks. Journal of Computer and System Sciences **64**(4), 820–842 (2002)

ICCS Camera Ready Version 2022
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-08754-7_53

15. Kovanen, L., Karsai, M., Kaski, K., Kertész, J., Saramäki, J.: Temporal motifs in time-dependent networks. Journal of Statistical Mechanics: Theory and Experiment **2011**(11), P11005 (2011)
16. Rossetti, G., Cazabet, R.: Community discovery in dynamic networks: a survey. ACM Computing Surveys (CSUR) **51**(2), 35 (2018)
17. Rozenshtein, P., Bonchi, F., Gionis, A., Sozio, M., Tatti, N.: Finding events in temporal networks: Segmentation meets densest subgraph discovery. Knowledge and Information Systems (2019)
18. Rozenshtein, P., Gionis, A.: Mining temporal networks. In: Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining. pp. 3225–3226. ACM (2019)
19. Wackersreuther, B., Wackersreuther, P., Oswald, A., Böhm, C., Borgwardt, K.M.: Frequent subgraph discovery in dynamic networks. In: Proceedings of the Eighth Workshop on Mining and Learning with Graphs. pp. 155–162. ACM (2010)

ICCS Camera Ready Version 2022
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-08754-7_53