Report from Dagstuhl Seminar 21171

Temporal Graphs: Structure, Algorithms, Applications

Edited by

Arnaud Casteigts[1], Kitty Meeks[2], George B. Mertzios[3], and Rolf Niedermeier[4]

1 **University of Bordeaux, FR,** arnaud.casteigts@labri.fr
2 **University of Glasgow, GB,** kitty.meeks@glasgow.ac.uk
3 **Durham University, GB,** george.mertzios@durham.ac.uk
4 **TU Berlin, DE,** rolf.niedermeier@tu-berlin.de

Abstract

This report documents the program and the outcomes of Dagstuhl Seminar 121171 "Temporal Graphs: Structure, Algorithms, Applications". The seminar was organized around four core areas: models, concepts, classes; concrete algorithmic problems; distributed aspects; applications. Because of the ongoing pandemic crisis, the seminar had to be held fully online, with talk and open problems sessions focussing on afternoons. Besides 19 contributed talks and small-group discussions, there were lively open-problem sessions, and some of the problems and research directions proposed there are part of this document. Despite strongly missing the usual Dagstuhl atmosphere and personal interaction possibilities, the seminar helped to establish new contacts and to identify new research directions in a thriving research area between (algorithmic) graph theory and network science.

Seminar April 25–30, 2021 – http://www.dagstuhl.de/21171
2012 ACM Subject Classification Mathematics of computing → Graph algorithms; Theory of computation → Parameterized complexity and exact algorithms
Keywords and phrases algorithm engineering, complex network analysis, distributed computing, models and classes, parameterized complexity analysis
Digital Object Identifier 10.4230/DagRep.11.3.16
Edited in cooperation with William Pettersson, John Sylvester

1 Executive Summary

Arnaud Casteigts
Kitty Meeks
George B. Mertzios
Rolf Niedermeier

Traditionally, graphs (composed of vertices and edges) are used to abstractly model diverse real-world systems, where vertices and edges represent elementary system units and some kind of interactions between them, respectively. However, in modern systems this modeling paradigm using static graphs may be too restrictive or oversimplifying, as interactions often change over time in a highly dynamic manner. The common characteristic in all these application areas is that the system structure, i.e., graph topology, is subject to *discrete changes over time*. In such dynamically changing graphs the notion of *vertex adjacency* needs to be revisited and various graph concepts, e.g., reachability and connectivity, now crucially

depend on the exact temporal ordering of the edges' presence. Furthermore, the rate and/or degree of the changes is generally too high to be reasonably modeled in terms of network faults or failures: in these systems changes are not anomalies but rather an integral part of the nature of the system.

A *temporal graph* is a graph which changes over time. Although many different variations for the model of temporal graphs exist, the most common one concerns graphs whose vertex set is fixed while the edge set changes over time. According to this model, given a static underlying graph G, a temporal graph is obtained by assigning to every edge of G a set of nonnegative integer time-labels, indicating the discrete time steps in which this edge is active. Many notions and algorithms on static (i.e., non-temporal) graphs can be transferred in a natural way to their temporal counterpart, while in other cases new approaches are needed to define the appropriate temporal notions. In most cases, the existence of the time dimension adds some degree of freedom in defining a specific temporal variant of a static problem.

The purpose of this one-week seminar was to present and discuss recent advances in the area of temporal and dynamically changing graphs, and to identify and highlight some of the current key challenges to better understand the multiple facets of the computational complexity of temporal graph problems. The seminar was mostly rooted in the rich and mature experience with algorithmic problems on static (i.e., traditional) graphs and in the general quest to understand the computational complexity of temporal versions of fundamental graph problems and algorithms.

Four key areas had been identified, which constituted the backbone of the seminar:

- **Models, Concepts, Classes.** Transferring problems from static to temporal graphs in a meaningful way is a challenging task, with potentially several well-motivated models in the temporal setting corresponding to the same static graph problem. Since temporal graph problems are notoriously hard, a natural way to approach this issue is to restrict input instances. One of the most immediate ways to restrict inputs (which originates in the static algorithmic graph theory) is to restrict the family of underlying graphs G. However, in temporal graphs, the temporal dimension offers so much structure that one can encode hard problems without the need of a sophisticated structure in the underlying graph or in any single layer. This observation suggests that restrictions of the "temporal pattern", i.e., of the way in which the time-labels appear (either alone or in combination with the above mentioned restrictions) can often help to identify tractable special cases of temporal graph problems.
- **Concrete Algorithmic Problems.** There are many examples of canonical generalizations from classic optimization problems on static graphs to temporal graphs. The most meaningful way in which a temporal version of a classic optimization problem may be defined crucially depends on the underlying application domain. Generally, temporal variants tend to become computationally harder than the corresponding classic optimization problem. Canonical ways to tackle the computational hardness are to (i) restrict the input instances to certain graph classes that allow for efficient (exact) algorithms or find parameters that allow for fixed-parameter algorithms, (ii) aiming for polynomial-time approximation algorithms where a certain level of solution quality is guaranteed, or (iii) combining (i) and (ii).
- **Distributed Aspects.** A common approach to analyzing distributed algorithms is the characterization of necessary and sufficient conditions to their success. These conditions commonly refer to the communication model, synchronicity, or structural properties of the network (e.g., is the topology a tree, a grid, a ring, etc.) In a highly-dynamic network, the topology changes *during* the computation, and this may have a dramatic effect on

computation. The study of this impact has led the distributed algorithms community to define a number of classes of temporal graphs that capture various levels of regularities a graph may satisfy over time, regardless of its structure at any single instant. It turns out that some of these properties – in particular the ones pertaining to finite time – are also relevant in a (centralized) algorithmic setting and have an impact on the computational complexity (or feasibility) of classic problems. In particular, properties like periodicity, temporal connectivity, and bounded temporal diameters have been considered both in the distributed and in the non-distributed settings. The seminar played here a crucial role in allowing the different communities to meet and share their complementary experience of temporal properties, as well as to converge on terminology and modeling aspects.

- **Applications.** Whenever there is a situation with pairwise interactions and information about the time point when these interactions happen, the framework of temporal graphs offers a natural mathematical model. This is especially the case in applications where the time information is critical. Examples include traffic and transportation networks, social network analysis (especially when analyzing disease or rumor spreading phenomena), biological networks, mobile sensor networks, and neural networks. All these application areas have their own problem settings and problem instances with specific properties and, in order to apply the fast-developing theory of algorithms for temporal graphs, it is essential to identify and formalise the properties of temporal graphs derived from data sets in these application areas, with the goal of obtaining application-driven restrictions of the input instances that allow for efficient algorithms. Among other application-oriented talks, the seminar included two highly topical talks relating to the role of temporal graphs in pandemic modelling.

The Dagstuhl Seminar 21171 "Temporal Graphs: Structure, Algorithms, Applications" brought together 53 participants from 13 different countries in Europe, USA, Japan, and Israel. The list of participants and speakers contained international experts in algorithms and complexity, social networks and computational social choice, complex systems, distributed algorithms, and parameterized algorithmics. We had in total 21 talks, each of approximately 30 minutes duration, and two sessions where open problems were proposed and discussed.

As this Dagstuhl Seminar was held entirely online (which would be unheard of a few years ago), we provided access to all participants to the online platform GatherTown which enabled us to virtually interact in a way which simulated (very satisfactory) physical meetings and interaction (e.g., also having virtual boards at our disposal). We had the GatherTown platform open every day all day (from the morning until the night), while we specified slots of 2-hours daily where everybody was expected to come to GatherTown. The rest of the day, GatherTown was there to facilitate all people who wanted to have extended physical-looking scientific meetings. During the Seminar, collaborative work was encouraged over all formal and informal scientific discussions. By building on the pre-existing synergies between the participating researchers, new collaborations have been initiated and old ones have been reinforced, and this across all all the communities which were represented in the Seminar.

To conclude, this Seminar has been successful in bringing together scientists from different backgrounds and initiating or strengthening research collaborations on the wide topic of temporal and dynamically changing graphs. Last, but not least, these collaborations and scientific discussions presented a fruitful mix between young researchers, such as PhD students, with older and established researchers in the general field of temporal graphs.

We would like to express our special gratitude to the team of Schloss Dagstuhl who were extremely supportive and also flexible on how to organize a virtual Dagstuhl Seminar during these unprecedented times of the pandemic, without compromising the scientific quality and the overall participation experience of the Seminar.

Acknowledgments

The organizers are very grateful to Hendrik Molter amd Malte Renken (TU Berlin) for helping in organizing the seminar (program compilation, technical support during the seminar, etc.) and to William Petterson and John Sylvester (University of Glasgow) for their help in collecting the material and compiling this report.

2 Table of Contents

3 Overview of Talks

3.1 k-Edge-Connectivity Models in Temporal Graphs

Prithwish Basu (BBN Technologies – Cambridge, US)

Joint work of Prithwish Basu, Amotz Bar-Noy, Feng Yu, Ram Ramanathan, Dror Rawitz

We extend the notion of k-edge-connectivity from static graphs to temporal graphs. A connected static graph is k-edge-connected if it remains connected after the removal of any subset of fewer than k edges; it also means that k edge-disjoint paths exist between each pair of vertices. Temporal graphs can be represented as a temporally ordered sequence of static snapshots of graphs (or graphlets) containing edges in the spatial dimension, and each vertex in time slot j is connected by a temporal (directed) edge to its future counterpart in time slot $j+1$. In such graphs, we give multiple interpretations of k-edge-connectivity, which we refer to as T-k-edge-connectivity, depending on the different degrees of overlap experienced by multiple source-to-target or (s,t) paths.

We introduce the following three models for measuring T-k-edge-connectivity between a given (s,t) pair of vertices:

1. Resource overlap model: no two (s,t)-paths can traverse a common spatial edge in any graphlet or a common temporal edge connecting consecutive graphlets, although they are allowed to traverse different instances of any spatial edge in multiple graphlets.
2. Resource and wait overlap model: this is the same as model 1 except that multiple (s,t)-paths are allowed to pass through a common temporal edge connecting two nodes in consecutive graphlets, which denotes waiting or buffering.
3. No overlap model: no two (s,t)-paths can traverse more than one instance of any spatial edge in multiple graphlets.

While models 1 and 2 capture the impact of congestion or failure of certain spatial edges or buffers at vertices, model 3 captures the scenario when a path may traverse a *single use* on-demand edge.

We show that in the resource overlap models, T-k-edge-connectivity can be computed in polynomial time by solving the Maximum Flow problem on a directed spatial-temporal graph with unit capacities on all edges (for model 1) and with infinite capacities on temporal edges (for model 2). In contrast, computing T-k-edge-connectivity for model 3 is NP-hard, which we show by reduction from the Maximum Independent Set problem. We simulate random temporal Erdos-Renyi graphs and random temporal graphs embedded on lattices and characterize how T-k-edge-connectivity varies with the number of vertices in each graphlet (N), the edge probability (p) and the number of graphlets (T). We also show that not surprisingly, the k-edge-connectivity of the union of T static graphs severely overestimates the T-k-edge-connectivity for most regimes of N, p, and T.

3.2 Interval-membership-width: dynamic programming on temporal graphs

Benjamin Bumpus (University of Glasgow, GB)

Joint work of Benjamin Bumpus, Kitty Meeks
Main reference Benjamin Merlin Bumpus, Kitty Meeks: "Edge exploration of temporal graphs", CoRR, Vol. abs/2103.05387, 2021.
URL https://arxiv.org/abs/2103.05387

We introduce a natural temporal analogue of Eulerian circuits and prove that, in contrast with the static case, it is NP-hard to determine whether a given temporal graph is temporally Eulerian even if strong restrictions are placed on the structure of the underlying graph and each edge is active at only three times. However, we do obtain an FPT-algorithm with respect to a new parameter called interval-membership-width which restricts the times assigned to different edges; we believe that this parameter will be of independent interest for other temporal graph problems. Our techniques also allow us to resolve two open question of Akrida, Mertzios and Spirakis [CIAC 2019] concerning a related problem of exploring temporal stars.

3.3 An overview on Dynamic community detection

Rémy Cazabet (University of Lyon, FR)

Joint work of Rémy Cazabet, Giulio Rossetti
Main reference Rémy Cazabet, Souâad Boudebza, Giulio Rossetti: "Evaluating community detection algorithms for progressively evolving graphs", J. Complex Networks, Vol. 8(6), 2021.
URL https://doi.org/10.1093/comnet/cnaa027

Community detection is one of the most famous problem of graph analysis. In recent years, many works have focused on the discovery of communities in dynamic/temporal networks. This problem raises new challenges, such as the tracking of the evolution of those communities, including community events (merge, split, etc.), temporal smoothing, ship of Theseus paradox, etc.

In this presentation, I summarize 3 recent publications [1, 2, 3] by Giulio Rossetti and myself on this topic, to give an overview of the state of the art and challenges to work on.

References

1 Rossetti, G., Cazabet, R. (2018). Community discovery in dynamic networks: a survey. ACM Computing Surveys (CSUR), 51(2), 1-37.
2 Cazabet, R., Rossetti, G. (2019). Challenges in community discovery on temporal networks. In Temporal Network Theory (pp. 181-197). Springer, Cham.
3 Cazabet, R., Boudebza, S., Rossetti, G. (2020). Evaluating community detection algorithms for progressively evolving graphs., Journal of Complex Networks, Volume 8, Issue 6, 1 December 2020, cnaa027, https://doi.org/10.1093/comnet/cnaa027

3.4 Reachability and Distances in Temporal Graphs

Pierluigi Crescenzi (Gran Sasso Science Institute, IT)

Joint work of Pierluigi Crescenzi, Marco Calamai, Clémence Magnien, Andrea Marino

In the last ten years, I get acquainted with some techniques which have been developed for computing reachability properties and distance-based measures in (static) graphs. In particular, these techniques are (1) the sketch techniques (one very popular algorithm based on sketches is the ANF algorithm for approximating the neighbourhood function [1]), (2) sampling (applied, for instance, for computing an approximation of the closeness centrality measure [2]), and (3) techniques based on iteratively update lower and upper bounds on specific graph measures (as in the case of the iFUB algorithm for computing the diameter [3]). Can we apply these techniques to temporal graphs? Indeed, we can, and this is the main motivation behind this presentation, which is based on the results contained in [4, 5, 6].

References

1 C.R. Palmer, P.B. Gibbons, C. Faloutsos: ANF: a fast and scalable tool for data mining in massive graphs. KDD 2002: 81-90.
2 D. Eppstein, J. Wang: Fast Approximation of Centrality. J. Graph Algorithms Appl. 8: 39-45 (2004).
3 P. Crescenzi, R. Grossi, M. Habib, L. Lanzi, A. Marino: On computing the diameter of real-world undirected graphs. Theor. Comput. Sci. 514: 84-95 (2013).
4 P. Crescenzi, C. Magnien, A. Marino: Approximating the Temporal Neighbourhood Function of Large Temporal Graphs. Algorithms 12(10): 211 (2019).
5 P. Crescenzi, C. Magnien, Andrea Marino: Finding Top-k Nodes for Temporal Closeness in Large Temporal Graphs. Algorithms 13(9): 211 (2020).
6 M. Calamai, P. Crescenzi, A. Marino: On Computing the Diameter of (Weighted) Link Streams. SEA 2021: to appear.

3.5 How can we help in rapid infectious disease modelling?

Jessica Enright (University of Glasgow, GB)

How can we help in rapid infectious disease modelling?

Network modelling of human and animal and movements have play a major role in modelling of infectious disease, including in modelling the ongoing Covid-19 pandemic. In many cases these models have incorporated information that could be viewed as a temporal graph. In this talk I outlined some of my experiences in modelling of this kind as well as examples from the work of others that I have seen. I focused on two challenges: first, how to best model temporal graphs that underlie models from data, and secondly the general question of interventions on temporal graphs to limit disease spread.

When discussing modelling of temporal graphs from data, I looked at volume-change plots of human mobility over the pandemic provided e.g. by national transport agencies or Google Community Mobility Reports, and gave examples from work on livestock movement networks e.g. [1].

When discussing interventions, I focussed in particular on the problem of assigning times to classes of edges so as to minimise the maximum reachability within a temporal graph, as in [2].

References

1 Ruget, AS., Rossi, G., Pepler, P.T. et al. Multi-species temporal network of livestock movements for disease spread. Applied Network Science 6, 15 (2021). https://doi.org/10.1007/s41109-021-00354-x

2 Enright, J. Meeks, K., Skerman, F. Assigning times to minimise reachability in temporal graphs. Journal of Computer and System Sciences, 115: 169-186 (2021). https://doi.org/10.1016/j.jcss.2020.08.001

3.6 Temporal Graph Exploration

Thomas Erlebach (University of Leicester, GB)

Joint work of Michael Hoffmann, Frank Kammer, Kelin Luo, Andrej Sajenko, Jakob Spooner
Main reference Thomas Erlebach, Frank Kammer, Kelin Luo, Andrej Sajenko, Jakob T. Spooner: "Two Moves per Time Step Make a Difference", in Proc. of the 46th International Colloquium on Automata, Languages, and Programming, ICALP 2019, July 9-12, 2019, Patras, Greece, LIPIcs, Vol. 132, pp. 141:1–141:14, Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2019.
URL https://doi.org/10.4230/LIPIcs.ICALP.2019.141

A temporal graph is a sequence of static graphs that all have the same vertex set, but the edge set in each step can be different. An agent can in each step either stay at its current vertex or move over an incident edge that is present in that step. The goal is to let the agent visit all vertices of the temporal graph as quickly as possible. While it is easy to visit all vertices of a static graph with n vertices in $O(n)$ steps (e.g., by using depth-first search), the exploration of temporal graphs may require a much larger number of steps: We show that even if the temporal graph is a connected graph in each step and the dynamics are fully known in advance, exploration may require a quadratic number of steps. We also present upper and lower bounds on the worst-case exploration time of temporal graphs for various restricted cases (e.g., restrictions on the underlying graph or on the maximum degree of the graph in each step), outlining the main techniques that have been used to obtain these bounds and highlighting the many cases with large gaps between the known upper and lower bounds.

References

1 Thomas Erlebach, Frank Kammer, Kelin Luo, Andrej Sajenko, and Jakob T. Spooner. Two Moves per Time Step Make a Difference. In *46th International Colloquium on Automata, Languages, and Programming (ICALP 2019)*, volume 132 of LIPIcs, Schloss Dagstuhl–Leibniz-Zentrum für Informatik, 2019, pages 141:1–141:14.

2 Thomas Erlebach, Michael Hoffmann, and Frank Kammer. On Temporal Graph Exploration. *Journal of Computer and System Sciences*, 119:1–18, 2021.

3 Thomas Erlebach and Jakob T. Spooner. Exploration of k-Edge-Deficient Temporal Graphs. In *17th International Symposium on Algorithms and Data Structures (WADS 2021)*, LNCS, Springer, 2021. To appear.

3.7 Approximation Algorithms for Multistage Problems

Bruno Escoffier (Sorbonne University – Paris, FR)

In the considered multistage setting, we are given a temporal graph $G_1, ..., G_T$, seen as a sequence of instances of a given optimization problem (matching, spanning tree,...). The goal is to compute a sequence $S_1, ..., S_T$ of feasible solutions of the problem (S_t begin feasible for G_t), while trying to minimize: - the (sum of) costs of individual solution - the (sum of) transition costs, which occurs when moving from solution S_t to solution S_{t+1}. In this talk we review some recent results, dealing with computational complexity and approximation algorithms, obtained on several graph multistage problems, such as matching, spanning tree, min cut, vertex cover,...

3.8 Temporal Graph Problems From the Multistage Model

Till Fluschnik (TU Berlin, DE)

Joint work of Till Fluschnik, Rolf Niedermeier, Valentin Rohm, Carsten Schubert, Philipp Zschoche
Main reference Till Fluschnik, Rolf Niedermeier, Valentin Rohm, Philipp Zschoche: "Multistage Vertex Cover", in Proc. of the 14th International Symposium on Parameterized and Exact Computation, IPEC 2019, September 11-13, 2019, Munich, Germany, LIPIcs, Vol. 148, pp. 14:1–14:14, Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2019.
URL https://doi.org/10.4230/LIPIcs.IPEC.2019.14

In the multistage model, roughly speaking, we are given a sequence of τ instances (stages) of the same problem, and the task is to find good solutions for each instance such that any two solutions to consecutive instances differ little. As a concrete example, in the Multistage Vertex Cover problem, given τ instances of Vertex Cover – over the same vertex set and each with the same integer k – and an integer ℓ, the question is whether there is a sequence of k-sized solutions to the instances such that the symmetric difference of any two solutions to consecutive instances is of size at most ℓ. This is a temporal graph problem: Given a temporal graph with lifetime τ, integers k and ℓ, is there a k-sized vertex cover for each layer such that the symmetric difference of any two vertex covers to consecutive layers is of size at most ℓ.

In this talk, we consider temporal graph problems derived from the multistage model. We give a brief introduction for multistage problems, sketch their history, and outline their relation to temporal graphs. We then present two multistage graph problems: Multistage Vertex Cover and Multistage s-t Path (finding a short s-t path in each layer). We present results for the two problems and given some algorithmic detail; In particular, we demonstrate the usability of concepts regarding static graphs (problems) when lifted to temporal graphs (problems). Finally, we reflect and discuss further research directions, models, and variations regarding multistage problems.

3.9 Parameterized complexity of temporal domination and related problems

Hans L. Bodlaender

Joint work of Hans Bodlaender, Carla Groenland, Jesper Nederlof, Celine Swennenhuis
Main reference Hans L. Bodlaender, Carla Groenland, Jesper Nederlof, Céline M. F. Swennenhuis: "Parameterized Problems Complete for Nondeterministic FPT time and Logarithmic Space", CoRR, Vol. abs/2105.14882, 2021.
URL https://arxiv.org/abs/2105.14882

Several parameterized problems are known to belong in XP and hard for a class in the W-hierarchy, e.g., hard for W[t], for all positive integers t. In this talk, we discuss for several such problems completeness for a relatively little studied class: the class of problems that can be solved with a non-deterministic algorithm in $f(k)poly(n)$ time and $f(k)\log n$ space, with f a computable function, k the parameter, n the input size, and $poly$ a polynomial. An example is the maintenance of dominating sets of size k on temporal graphs, where at each step in time, we want a dominating set of size at most k, and want to minimize the number of times we change a vertex from the dominating set.

3.10 The Computational Complexity of Finding Temporal Paths under Waiting Time Constraints

Hendrik Molter (Ben-Gurion University, IL)

Joint work of Hendrik Molter, Arnaud Casteigts, Anne-Sophie Himmel, Philipp Zschoche

Computing a (short) path between two vertices is one of the most fundamental primitives in graph algorithmics. In recent years, the study of paths in temporal graphs, that is, graphs where the vertex set is fixed but the edge set changes over time, gained more and more attention. A path is time-respecting, or temporal, if it uses edges with non-decreasing time stamps. We investigate a basic constraint for temporal paths, where the time spent at each vertex must not exceed a given duration Δ, referred to as Δ-restless temporal paths. This constraint arises naturally in the modeling of real-world processes like packet routing in communication networks and infection transmission routes of diseases where recovery confers lasting resistance. While finding temporal paths without waiting time restrictions is known to be doable in polynomial time, we show that the "restless variant" of this problem becomes computationally hard even in very restrictive settings. For example, it is W[1]-hard when parameterized by the distance to disjoint path of the underlying graph, which implies W[1]-hardness for many other parameters like feedback vertex number and pathwidth. A natural question is thus whether the problem becomes tractable in some natural settings. We explore several natural parameterizations, presenting FPT algorithms for three kinds of parameters: (1) output-related parameters (here, the maximum length of the path), (2) classical parameters applied to the underlying graph (e.g., feedback edge number), and (3) a new parameter called timed feedback vertex number, which captures finer-grained temporal features of the input temporal graph, and which may be of interest beyond this work.

3.11 Multistage Graph Problems on a Global Budget

Frank Kammer (THM – Gießen, DE)

Joint work of Klaus Heeger, Anne-Sophie Himmel, Frank Kammer, Rolf Niedermeier, Malte Renken, Andrej Sajenko
Main reference Klaus Heeger, Anne-Sophie Himmel, Frank Kammer, Rolf Niedermeier, Malte Renken, Andrej Sajenko: "Multistage graph problems on a global budget", Theor. Comput. Sci., Vol. 868, pp. 46–64, 2021.
URL https://doi.org/10.1016/j.tcs.2021.04.002

Time-evolving or temporal graphs gain more and more popularity when exploring complex networks. In this context, the multistage view on computational problems is among the most natural frameworks. Roughly speaking, herein one studies the different (time) layers of a temporal graph (effectively meaning that the edge set may change over time, but the vertex set remains unchanged), and one searches for a solution of a given graph problem for each layer. The twist in the multistage setting is that the solutions found must not differ too much between subsequent layers. We relax on this already established notion by introducing a global instead of the local budget view studied so far. More specifically, we allow for few disruptive changes between subsequent layers but request that overall, that is, summing over all layers, the degree of change is moderate. Studying several classical graph problems (both NP-hard and polynomial-time solvable ones) from a parameterized complexity angle, we encounter both fixed-parameter tractability and parameterized hardness results. Surprisingly, we find that sometimes the global multistage versions of NP-hard problems such as Vertex Cover turn out to be computationally more tractable than the ones of polynomial-time solvable problems such as Matching. In addition to time complexity, we also analyze the space efficiency of our algorithms.

3.12 Efficient limited time reachability estimation in temporal networks

Mikko Kivelä (Aalto University, FI)

Joint work of Arash Badie Modiri, Márton Karsai, Mikko Kivelä
Main reference Arash Badie-Modiri, Márton Karsai, Mikko Kivelä: "Efficient limited-time reachability estimation in temporal networks", Phys. Rev. E, Vol. 101, p. 052303, American Physical Society, 2020.
URL https://doi.org/10.1103/PhysRevE.101.052303

Time-limited states characterise several dynamical processes evolving on the top of networks. During epidemic spreading infected agents may recover after some times, in case of information diffusion people may forget news or consider it out-dated, or in travel routing systems passengers may not wait forever for a connection. These systems can be described as limited waiting-time processes, which can evolve along possible network paths strongly determined by the time-limited states of the interacting nodes. This is particularly important on temporal networks where the time-scales of interactions are heterogeneous and correlated in various ways. The structure of the temporal paths has previously been studied by finding the reachability from a sampled set of sources or by simulating spreading processes. Recently temporal event graphs were proposed as an efficient representation of temporal networks mapping all time-respecting paths at once so that one could study how they form connected structures in the temporal network fabric. However, their analysis has been limited to their weakly connected components, which only give an upper bound for their physically important

in- and out-components determining the downstream outcome of any dynamical processes. Here we propose a probabilistic counting algorithm, which gives simultaneous and precise estimates of the in- and out-reachability (with any chosen waiting-time limit) for every starting event in a temporal network. Our method is scalable allowing measurements for temporal networks with hundreds of millions of events. This opens up the possibility to analyse reachability, spreading processes, and other dynamics in large temporal networks in completely new ways; to compute centralities based on global reachability for all events; or to find with high probability the exact node and time, which could lead to the largest epidemic outbreak.

3.13 Nearly optimal spanners in quite sparse random simple temporal graphs

Michael Raskin (TU München, DE)

Joint work of Michael Raskin, Arnaud Casteigts, Malte Renken, Victor Zamaraev

A graph whose edges only appear at certain points in time is called a temporal graph. The addition of this temporal aspect fundamentally changes the notion of connectivity, as paths in temporal graphs can traverse edges only in chronological order. Whereas every static connected graph contains a spanning tree with linear number of edges, analogous statement about temporal spanners, i.e., temporal subgraphs providing universal connectivity, have turned out to be false [Axiotis, Fotakis 2016]. Nevertheless, by taking a probabilistic point of view, we are now able to prove that any temporally connected temporal graph asymptotically almost surely contains a temporal spanner with $2n + o(n)$ edges, which closely matches a long-known lower bound of $2n - 4$.

3.14 Connectivity Thresholds in Random Temporal Graphs

Malte Renken (TU Berlin, DE)

Joint work of [Malte Renken, Arnaud Casteigts, Michael Raskin, Victor Zamaraev
Main reference Arnaud Casteigts, Michael Raskin, Malte Renken, Viktor Zamaraev: "Sharp Thresholds in Random Simple Temporal Graphs", CoRR, Vol. abs/2011.03738, 2020.
URL https://arxiv.org/abs/2011.03738

A graph whose edges only appear at certain points in time is called a temporal graph. The addition of this temporal aspect fundamentally changes the notion of connectivity, as paths in temporal graphs can traverse edges only in chronological order. We consider a simple model of a random temporal graph, obtained by assigning to every edge of an Erdős–Rényi random graph $G_{n,p}$ a uniformly random presence time in the real interval $[0, 1]$. It turns out that this model exhibits a surprisingly regular sequence of thresholds related to temporal reachability. In particular, we show that at p = log n/n any fixed pair of vertices can asymptotically almost sureley (a.a.s.) reach each other, at 2 log n/n at least one vertex (and in fact, any fixed node) can a.a.s. reach all others, and at 3 log n/n all the vertices can a.a.s. reach each other (i.e., the graph is temporally connected). All these thresholds are sharp and also hold in a random temporal graph model where edges appear according to independent Poisson processes.

3.15 Exploration of k-edge-deficient temporal graphs

Jakob Spooner (University of Leicester, GB)

Joint work of Thomas Erlebach, Jakob Spooner

An *always-connected temporal graph* $\mathcal{G} = \langle G_1, ..., G_L \rangle$ with underlying graph $G = (V, E)$ is a sequence of graphs $G_t \subseteq G$ such that $V(G_t) = V$ and G_t is connected for all t. This paper considers the property of *k-edge-deficiency* for temporal graphs; such graphs satisfy $G_t = (V, E - -X_t)$ for all t, where $X_t \subseteq E$ and $|X_t| \leq k$. In this talk I'll discuss the the Temporal Exploration problem (compute a temporal walk that visits all vertices $v \in V$ at least once and finishes as early as possible) restricted to always-connected, k-edge-deficient temporal graphs. I'll sketch constructive proofs that show that k-edge-deficient and 1-edge-deficient temporal graphs can be explored in $O(kn \log n)$ and $O(n)$ timesteps, respectively. In the paper, a lower-bound construction of an infinite family of always-connected k-edge-deficient temporal graphs for which any exploration schedule requires at least $\Omega(n \log k)$ timesteps is also given.

3.16 Alternative Routes in Time-Dependent Networks

Christos Zaroliagis

Joint work of Christos Zaroliagi, Spyros Kontogiannis, Andreas Paraskevopoulos
Main reference Spyros C. Kontogiannis, Andreas Paraskevopoulos, Christos D. Zaroliagis: "Time-Dependent Alternative Route Planning", in Proc. of the 20th Symposium on Algorithmic Approaches for Transportation Modelling, Optimization, and Systems, ATMOS 2020, September 7-8, 2020, Pisa, Italy (Virtual Conference), OASIcs, Vol. 85, pp. 8:1–8:14, Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2020.
URL https://doi.org/10.4230/OASIcs.ATMOS.2020.8

We present a new method for computing a set of alternative origin-to-destination routes in road networks with an underlying time-dependent metric. The resulting set is aggregated in the form of a time-dependent alternative graph and is characterized by minimum route overlap, small stretch factor, small size and low complexity. To our knowledge, this is the first work that deals with the time-dependent setting in the framework of alternative routes. Based on preprocessed minimum travel-time information between a small set of nodes and all other nodes in the graph, our algorithm carries out a collection phase for candidate alternative routes, followed by a pruning phase that cautiously discards uninteresting or low-quality routes from the candidate set. Our experimental evaluation on real time-dependent road networks demonstrates that the new algorithm performs much better (by one or two orders of magnitude) than existing baseline approaches. In particular, the entire alternative graph can be computed in less than 0.384sec for the road network of Germany, and in less than 1.24sec for that of Europe. Our approach provides also "quick-and-dirty" results of decent quality, in about 1/300 of the above mentioned query times for continental-size instances.

3.17 Flooding and Self-Healing for Temporal Networks?

Amitabh Trehan (Durham University, GB)

Main reference Amitabh Trehan: "Algorithms for Self-Healing Networks", CoRR, Vol. abs/1305.4675, 2013.
URL http://arxiv.org/abs/1305.4675

We explore the connections between distributed algorithms and temporal graph theory. We look at some possible extensions and interesting questions with respect to temporal graph theory.

In particular, Amnesiac Flooding [1] is among the simplest algorithms for information dissemination on a network. The basic algorithm is to forward a message to every node except the ones you just received the message from. This algorithm is 'stateless' and uses no additional memory. However, with such loss of history, it is possible that the messages could be regenerated ad-infinitum and circulated forever on the network. In practice, "memory" flags are used explicitly to make sure a message is not circulated again. However, surprisingly and despite its simplicity, it turns out that amnesiac flooding terminates on every graph i.e. message circulation stops in a finite number of rounds. How would this behave for temporal graphs i.e. graphs whose edges may change over time?

Another interesting question is "self-healing" – i.e. fault-tolerance by quick local repair of a system under adversarial attack to another good but possibly degraded state [2]. One version that can be imagined is playing a round-based game on a graph where one player (adversary) can remove or add one node per round with the other player (healer) adding or removing edges in the locality of the attack with the aim of preserving certain invariants throughout the history of the attacks and repairs. Can this inspire extensions of temporal graph theory in presence of churn (node additions/deletions)? How does one even state results when node additions are permitted?

References

1 Walter Hussak and Amitabh Trehan. *On the Termination of Flooding*. 37th International Symposium on Theoretical Aspects of Computer Science STACS 2020, March 10-13, 2020, Montpellier, France. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. 2020.

2 Amitabh Trehan. *Algorithms for self-healing networks*. Dissertation, University of New Mexico, 2010. URL: `http://proquest.umi.com/pqdlink?did=2085415901&Fmt=2&clientId=11910&RQT=309&VName=PQD`.

3.18 Persistent connected components on dynamic graphs

Mathilde Vernet (University of Le Havre, FR)

Joint work of Mathilde Vernet, Yoann Pigné, Eric Sanlaville

This work focuses on connectivity in a dynamic graph. An undirected graph is defined on a finite and discrete time interval. Edges can appear and disappear over time. The first objective of this work is to extend the notion of connected component to dynamic graphs in a new way. Persistent connected components are defined by their size, corresponding to the number of vertices, and their length, corresponding to the number of consecutive time steps they are present on. The second objective of this work is to develop an algorithm computing

the largest, in terms of size and length, persistent connected components in a dynamic graph. PICCNIC algorithm (PersIstent Connected CompoNent InCremental Algorithm) is a polynomial time algorithm of minimal complexity. Another advantage of this algorithm is that it works online: knowing the evolution of the dynamic graph is not necessary to execute it. PICCNIC algorithm is implemented and experimented in order to carefully study the outcome of the algorithm according to different input graph types, as well as real data networks, to verify the theoretical complexity, and to confirm its feasibility for graphs of large size.

3.19 Modelling of interactions over time and the stream isomorphism problem

Tiphaine Viard (Telecom Paris, FR)

Joint work of Tiphaine Viard, Matthieu Latapy, Clémence Magnien, Florian Yger
Main reference Matthieu Latapy, Tiphaine Viard, Clémence Magnien: "Stream Graphs and Link Streams for the Modeling of Interactions over Time", CoRR, Vol. abs/1710.04073, 2017.
URL http://arxiv.org/abs/1710.04073

Interactions are everywhere. Be it people meeting in a room, individuals phoning each other or sending emails, machines exchanging IP packets over the network, we are faced with a massive intake of data that can be modelled as two entities u and v interacting at a time t. Being able to model and formally describe these real-world objects is of crucial importance from both a theoretical and applicative standpoint: they raise interesting formal and algorithmic questions, and can be applied for mining patterns, anomaly detection, community detection, among others.

Stream graphs, along with temporal networks and time varying graphs, are a formal model for these streams of interactions. In our talk, we present the core elements of the stream graph formalism, stating with streams, substreams, neighbourhoods, degrees and density and building gradually towards more complex notions, such as paths and reachability, notions of centrality or the clique enumeration problem. We show how these notions are both rooted in real-world data analysis and a generalization of the existing graph analysis concepts. We also show that what is picked up by these notions on real-world datasets is typically different than what can be seen through the graph, shedding light on the data in a complementary way.

We then spend some time introducing and proposing solutions to the stream isomorphism problem. We show that, for a stream graph with n nodes and m interactions, in somes cases, it is possible to reduce the problem to the graph isomorphism problem, by building a time-directed graph that has $2m$ nodes and $3m$ edges. We also discuss the fact that the problem is significantly more complex if one cannot reduce to the graph case, for example when nodes appear and disappear in time, or when interactions have durations. We outline some potential solutions in that case, and briefly discuss possible applications.

4 Open problems

4.1 Temporal matching in $O^*(2^n)$ FPT time?

Binh-Minh Bui-Xuan, (LIP6,CNRS – Sorbonne Université)

A temporal graph is a sequence of graphs over the same vertex set: $\mathcal{G} = (G_i)_{i \in T}$ with $T \subseteq \mathbb{N}$ a finite interval and $G_i = (V, E_i)$ a graph for every $i \in T$. Given an integer $\Delta \in \mathbb{N}$, we address the problem of finding a maximum set of independent sequences, each composed of exactly Δ edges: find $\mathcal{M} = \{(t, uv) : uv \in \bigcap_{t \leq i < t+\Delta} E_i\}$ such that $|\mathcal{M}|$ is maximum and the following hold: $(t, uv) \in \mathcal{M} \wedge (t', uw) \in \mathcal{M} \wedge |t - t'| < \Delta \Rightarrow v = w$.

This is an NP-hard problem for $\Delta > 1$, where any greedy algorithm is also a 2-approximation [1]. The problem stays NP-hard even on very restricted instances of temporal graphs [3]. On the positive side, the greedy approach performs very well on temporal graphs collected under geometric constraints, where the approximation ratio in these datasets averages at 1.02, not 2 [2, 4].

Taking $n = |V|$ as an FPT parameter, there exists a dynamic programming solving optimally the problem in time $O^*((\Delta + 1)^n)$ [4]. The dynamic programming exploits the total order of the time dimension $T \subseteq \mathbb{N}$: here, a sliding time window of size $\Delta + 1$ would record all possibilities of "boundaries" of a current candidate for being a future optimal solution.

Open question: *Is there an $O^*(2^n)$ dynamic programming finding such a maximum independent Δ-edge set?*

References

1 J. Baste and B.M. Bui-Xuan. Temporal matching in link stream: kernel and approximation. In *16th Cologne-Twente Workshop on Graphs and Combinatorial Optimization*, 2018.
2 J. Baste, B.M. Bui-Xuan, and A. Roux. Temporal matching. *Theoretical Computer Science*, 806:184–196, 2020.
3 G.B. Mertzios, H. Molter, R. Niedermeier, V. Zamaraev, and P. Zschoche. Computing maximum matchings in temporal graphs. In *37th International Symposium on Theoretical Aspects of Computer Science*, volume 154 of *LIPIcs*, pages 27:1–27:14, 2020.
4 T. Picavet, N.T. Nguyen, and B.M. Bui-Xuan. Temporal matching on geometric graph data. In *12th International Conference on Algorithms and Complexity*, volume 12701 of *LNCS*, pages 394–408, 2021.

4.2 Degenerate and slowly evolving dynamic networks

Rémy Cazabet (Université de Lyon, FR)

Many formalisms exist to represent dynamic/temporal networks. Among the most famous ones, we can cite link streams, interval graphs, snapshot sequences, time varying graphs, etc.

However, in recent studies, I've noted that there exist at least two very different types of *dynamic networks*, that I'll call *degenerate* and *slowly evolving* dynamic networks.

The difference between the two can be expressed intuitively as follows: a dynamic network is *degenerate* if we consider that it is not meaningful to observe the network at an instantaneous point in time t. On the contrary, in a slowly evolving network, the network is considered a meaningful graph at any t of its period of evolution.

Typical examples of a degenerate graph would be a collection of collected interactions, such as instantaneous messages sent in a social network platform. On the contrary, the relations of friends/followers in the same network would form a slowly evolving network.

My observation is that, although most authors are aware of such a distinction, it is barely ever discussed in the literature, so that when authors present a new algorithm or a new method, they do not specify if its adapted to one or the other of these networks.

My question relates to the fact that, although both categories of networks seem very different at first sight, many real networks do actually fall somewhere in-between the two. I gave as example the data from the Sociopatterns project, in which face-to-face interactions between individuals were automatically collected using wireless sensors every 20s. Although one might think at first that this data belongs to the slowly evolving network, when studying the data, it is clear that it rather belongs to the degenerate category. I subsequently argued that the problem also arises as soon as we aggregate interaction data: at which aggregation level do the dynamic networks shift from one category to the other, and thus methods designed to work with one type can be applied or not? What I wanted to discuss were therefore the existing methods, algorithms, discussion papers or definitions that could help researchers to identify, formally or intuitively, the type of dynamic networks they are confronted with.

4.3 Distances in temporal graphs

Pierluigi Crescenzi (Gran Sasso Science Institute, IT)

Two problems were posed by Pierluigi Crescenzi regarding the time and space complexity of determining distances in temporal graphs under various assumptions on the ordering of the input.

These problems were later solved as the result of a discussion with Michael Raskin.

4.4 Minimal complexity of MaxFlow on dynamic graphs

Eric Sanlaville (University of Le Havre, FR)

Dynamic Model.

- Discrete model: T time steps
- Dynamic Graph $\mathcal{G} = (V, \mathcal{E})$ composed of a sequence of t-graphs (or snapshot graphs) $G_1, \ldots, G_T$
- Each t-graph $G_\theta = (V, E_\theta)$
- Each time-arc $e = uv_\theta \in E_\theta$ has a capacity $c_\theta(uv)$
- There are two special nodes s the source and t the sink

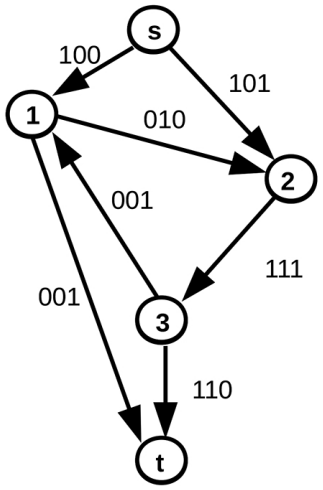

Figure 1 An example of a compact representation. Capacities are represented as time vectors along the arcs. In this simple example capacities are unitary.

Max Flow problem on the dynamic graph (simplest version).

- Send the maximum amount of flow from s to t during these T time steps.
- There is no travel time for the arcs
- Infinite storage is allowed on the nodes.

Complexity of classic static algorithms

As a short reminder, efficient algorithms are based on the following two ideas:

1. SSP: compute a Sequence of augmenting paths in the residual graph (choice of Shortest Paths).
2. PF: compute Pre-Flows, try to remove excess by local operations

We now give an incomplete summary of results for the static case (see [1]) with a focus on complexity independent of capacities.

algorithm	*idea*	*complexity (static)*
SSP	$Edmonds\ Karp$	n^2m
$Generic\ pre\ flow$	PF	n^2m
$FIFO\ Preflow$	$PF, FIFO\ order$	n^3
$HL\ Preflow$	$PF, Highest\ distance\ order$	$n^2\sqrt{m}$

Expanded graphs

Building the extended graph.

- A super-source and a super-sink are added
- Storage arcs are added as horizontal links, with infinite capacity

Size of the expanded graph.

- Number of nodes: $nT + 2 = \Theta(nT)$
- Number of arcs: $mT + (n-2)T + 2T$
  $= \Theta(mT)$ if $n = O(m)$.

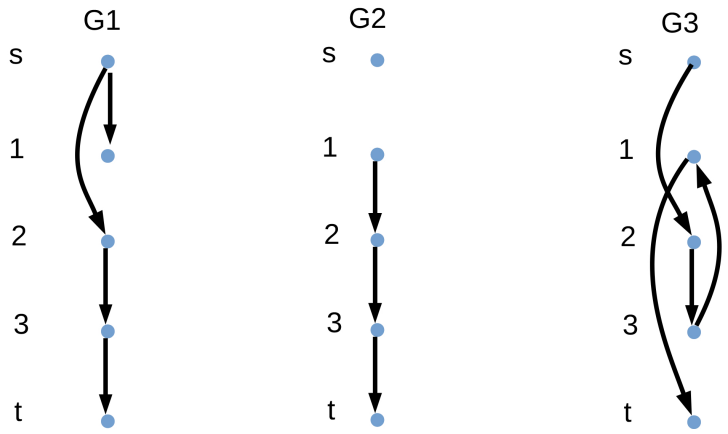

Figure 2 An example of the sequence of snapshots representation where arcs without capacity are omitted.

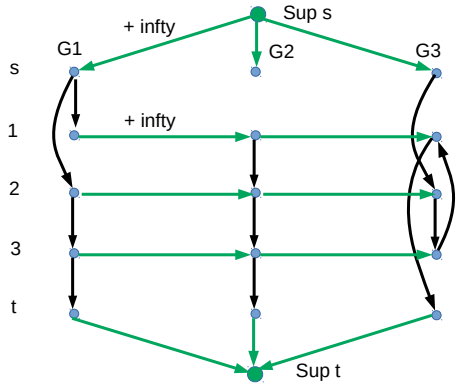

Figure 3 Example of the expanded graph of the temporal graph from Figure 2.

Complexity for the dynamic Max Flow using expanded graph

The following table shows direct use of classical algorithms on expanded graph

algorithm	*idea*	*complexity* (*static*)	*complexity* (*expanded*)
SSP	$Edmonds\ Karp$	n^2m	n^2mT^3
$Generic\ pre\ flow$	PF	n^2m	n^2mT^3
$FIFO\ Preflow$	$PF, FIFOorder$	n^3	n^3T^3
$HL\ Preflow$	$PF, Highest\ distance\ order$	$n^2\sqrt{m}$	$n^2\sqrt{m}T^2\sqrt{T}$

All algorithm complexity have a T^3 factor, except for the Highest Label (distance to the sink) first implementation of Pre-Flow : $T^2\sqrt{T}$.

Question. Can we do better than $T^{5/2}$?

Hopes.

- The expanded graph is not any directed graph
- Compact representation with capacity vectors allows for other algorithms?

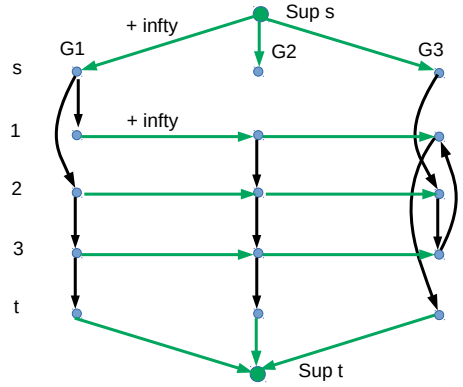

Figure 4 Example of successive shortest paths. In the image above there are three successive shortest paths:

- path $P_1 = s, 2, 3, t$
- path $P_2 = s, 1 \to 1 \to 1, t$
- path $P_3 = s, 2, 3, 1 \leftarrow 1, 2, 3, t$.

Wrong tracks.

- No direct result with structural consideration on expanded graph
- Examples can be built, for which augmenting paths are arbitrary long
- No optimum algorithm (so far) keeping compact representation (upper bounds using aggregated capacities)
- Maybe the answer is NO ? And that's why there is no work on this in the literature ?

References

1 Ravindra K. Ahuja, Thomas L. Magnanti, and James B. Orlin. *Network flows – theory, algorithms and applications*. Prentice Hall, 1993.

4.5 Three open problems pertaining to self-preserving communities, meta-theorems for multistage problems, and influence of burstiness on the computational complexity

Manuel Sorge (TU Wien, AT)

Self-preserving communities

In their recent work [8] put forward a definition of what a biological individual is, based on information-theoretic measures. Essentially, they propose that individuals are subsets of the environment that represent local maxima in the so-called mutual information between consecutive time steps. In plain words, the mutual information of some subset between two time steps is large, if there are many possible states of this subset and the states are strongly correlated between time steps. Even simpler, an individual is lively and it preserves itself.

As [8] briefly alluded to, this notion also applies in other contexts, such as communities in social networks. The advantage of looking at communities in social networks is that there is ample data to test [8] hypothesis modified to this context. However, we lack the algorithmic research to do this. What is a concrete formulation as a computational problem to test [8]'s hypothesis and what is the computational complexity of solving this problem?

Meta-theorems for multistage problems

The general aim in multistage problems is to find solutions, e.g. independent sets, for each snapshot of a given temporal graph such that the solutions satisfy some quality measure, e.g. an upper bound on the independent set size, and such that the solutions have some temporal relationship, such as differing in few vertices between consecutive snapshots. Several authors studied the computational complexity of concrete multistage problems, see e.g. [1, 3] and references therein. Perhaps time is rife to study the complexity of common problem formulations more generally. A somewhat informal common problem formulation is the following. Given a temporal graph $(G_i)_{i\in[\tau]}$ on vertex set V and integers $k, \ell \in \mathbb{N}$, we want to decide whether the following formula holds:

$$\exists (x_{i,j})_{i\in[k],j\in[\tau]} \in V^{k+\tau} \colon \forall j \in [\tau - -1] \colon \mathsf{similar}_j((x_{i,j}, x_{i,j+1})_{i\in[k]}, \ell) \wedge \forall j \in [\tau] \colon \mathsf{property}_j((x_{i,j})_{i\in[k]}).$$

E.g. to solve a recently studied multistage path problem [3], set $\mathsf{property}_j$ to a function that checks whether the sequence of k variables given as an argument is the sequence of vertices of a path in G_j and set $\mathsf{similar}_j$ to a formula that checks whether the two sequences of vertices given as an argument differ by at most ℓ vertices when interpreted as a set. An example of the type of results that this formulation affords is the following. We may consider a temporal graph as a structure from formal logic, whose domain contains the vertex set and whose signature contains equality and the adjacency relation for each snapshot G_i. If the maximum degree in each snapshot is bounded by Δ, and if $\mathsf{property}_j$ and $\mathsf{similar}_j$ are expressible in first-order logic, then from Seese's theorem [9] it follows that that the above problem is solvable in $f(\tau, \Delta, \zeta)n$ where ζ is the quantifier depth of $\mathsf{property}_j$ and $\mathsf{similar}_j$ (trivially upper bounded by their length), and f is an exponential function. The (exponential) dependency on τ is not nice and there are nontrivial examples for which it can be avoided, such as the above mentioned multistage path problem [3]. An open question is for example to explore and characterize the formulas for $\mathsf{property}_j$ and $\mathsf{similar}_j$ such that dropping exponential dependency on τ is possible. Another direction is to consider generalizations of Seese's theorem for structures of bounded local treewidth or other structural sparsity measures [4] and properly adapt them to the multistage setting.

Burstiness

Burstiness refers to the temporal clustering of activity, such as in sending emails, where it can be observed that users have short periods of activity followed by long periods of inactivity. In particular, such clustering results in distributions of inter-interaction times that roughly follow a power law [10]. It is not far-fetched to assume that such structure can be used algorithmically. For example, a simple measure of the degree distribution of a graph is the h-index. The h-index is the largest integer h such that there are at least h vertices of degree at least h. This parameter is a small constant in real world networks [2], which often have degree distributions similar to power laws. It is not hard to show then, that the in general NP-hard Independent Set problem is solvable in $O(2^h \cdot (n + m))$ time. Similar results hold for other problems including various dense-subgraph problems [7, 6]. The only research

in the direction of measuring and exploiting burstiness algorithmically that I am aware of is when we think of burstiness as a kind of temporal sparsity. [5] looked at unions of intervals of snapshot graphs and computed their so-called degeneracy, a measure of sparsity, and showed that it is a small constant in real-world temporal graphs, much smaller than the degeneracy of the union of all snapshots. However, this is not a first-principles approach and with this method we do not learn or exploit, for example, whether (and how many) bursts temporally overlap. What is a good combinatorial measure of burstiness? Is there one that can be exploited algorithmically, for example, in algorithms enumerating dense temporal subgraphs?

References

1. Jiehua Chen, Hendrik Molter, Manuel Sorge, and Ondrej Suchý. Cluster editing in multi-layer and temporal graphs. In Wen-Lian Hsu, Der-Tsai Lee, and Chung-Shou Liao, editors, *29th International Symposium on Algorithms and Computation (ISAAC 2018)*, volume 123 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 24:1–24:13, Dagstuhl, Germany, 2018. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik.
2. David Eppstein and Emma S. Spiro. The h-index of a graph and its application to dynamic subgraph statistics. *Journal of Graph Algorithms and Applications*, 16(2):543–567, 2012.
3. Till Fluschnik, Rolf Niedermeier, Carsten Schubert, and Philipp Zschoche. Multistage s-t path: Confronting similarity with dissimilarity in temporal graphs. In Yixin Cao, Siu-Wing Cheng, and Minming Li, editors, *31st International Symposium on Algorithms and Computation (ISAAC 2020)*, volume 181 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 43:1–43:16, Dagstuhl, Germany, 2020. Schloss Dagstuhl–Leibniz-Zentrum für Informatik.
4. Martin Grohe, Stephan Kreutzer, and Sebastian Siebertz. Deciding first-order properties of nowhere dense graphs. *J. ACM*, 64(3):17:1–17:32, 2017.
5. Anne-Sophie Himmel, Hendrik Molter, Rolf Niedermeier, and Manuel Sorge. Adapting the bron–kerbosch algorithm for enumerating maximal cliques in temporal graphs. *Social Network Analysis and Mining*, 7(1):35, 2017.
6. Falk Hüffner, Christian Komusiewicz, and Manuel Sorge. Finding highly connected subgraphs. In *SOFSEM 2015: Theory and Practice of Computer Science*, Lecture Notes in Computer Science, pages 254–265. Springer, Berlin, Heidelberg, 2015.
7. Christian Komusiewicz and Manuel Sorge. An algorithmic framework for fixed-cardinality optimization in sparse graphs applied to dense subgraph problems. *Discrete Applied Mathematics*, 193:145–161, 2015.
8. David Krakauer, Nils Bertschinger, Eckehard Olbrich, Jessica C. Flack, and Nihat Ay. The information theory of individuality. *Theory in Biosciences*, 139(2):209–223, 2020.
9. Detlef Seese. Linear time computable problems and first-order descriptions. *Mathematical Structures in Computer Science*, 6(6):505–526, 1996.
10. Alexei Vazquez, Balázs Rácz, András Lukács, and Albert-László Barabási. Impact of non-poissonian activity patterns on spreading processes. *Physical Review Letters*, 98(15):158702, 2007.

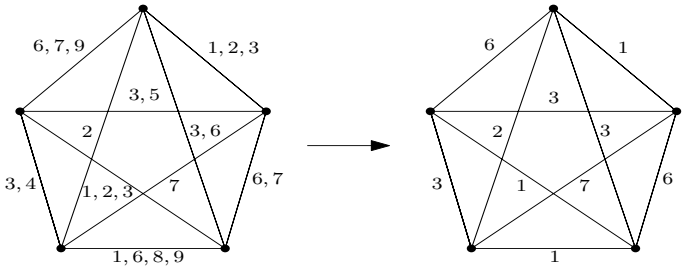

Figure 5 Preliminary step from [1] on an arbitrary temporal clique of size 5. One label per edge is sufficient to maintain temporal connectivity. (Details concerning strict or non-strict journeys are omitted in this document.)

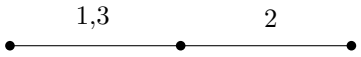

Figure 6 Trivial counterexample showing that one label per edge is not sufficient to maintain temporal connectivity in general temporal graphs.

4.6 The number of labels per edge maintaining temporal connectivity

Jason Schoeters (University of Bordeaux, FR)

Introduction

In [1], the authors are interested in removing labels from temporal cliques, without breaking temporal connectivity. The remaining structure is called a spanner. They succeed in removing a significant amount of labels from any given initial temporal clique through their proposed algorithm, obtaining a sparse spanner of size $O(n \log n)$, with n the number of vertices of the temporal graph. As a preliminary step (see Figure 5, the authors reduce any given temporal clique to one with only one label per edge. In other words, independent from the labelling, it is sufficient to keep only one label per edge of a temporal clique to maintain temporal connectivity.

As a quick motivation for this document, if the temporal graph represents a network in which messages are sent, one may be interested in knowing how many messages may need to pass through any given link so as to keep the network up to date (whereas in [1], the authors would be interested in knowing how many messages **in total** are needed to stay up to date).

These notes, and the presentation during the open problem session, are an attempt to extend this preliminary step to other classes of temporal graphs. We present preliminary results and open problems surrounding this topic, as well as mention a result obtained during the Dagstuhl.

Preliminary results

In general temporal graphs, it is quickly clear that one label per edge may not be sufficient to maintain temporal connectivity (see Figure 6).

So maybe two labels would suffice? The idea being that one label allows messages going through in one way, and the other in the other way. Indeed, for some specific edges (and graphs) this idea holds.

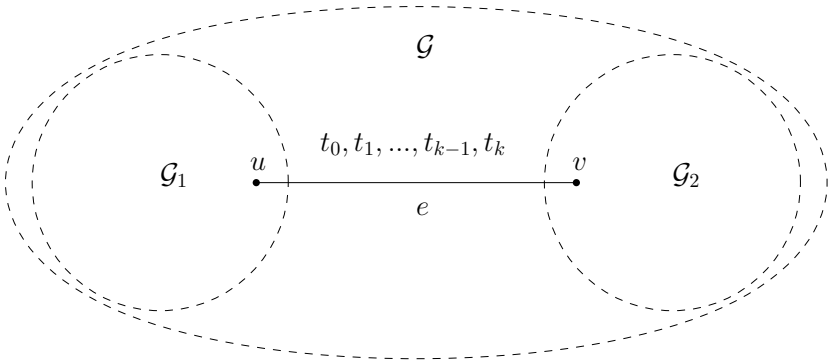

Figure 7 Sketch of a temporal graph with a bridge edge.

▶ **Lemma 1.** *Two labels are sufficient for bridge edges to maintain temporal connectivity.*

Proof. Consider Figure 7, a sketch of a temporal graph $\mathcal{G}$ with bridge edge e having k labels $t_0, t_1, ..., t_k$, separating $\mathcal{G}$ into temporal graph $\mathcal{G}_1$ and $\mathcal{G}_2$. Define t_u^- the earliest time at which all vertices in $\mathcal{G}_1$ are able to reach u. Similarly, define t_v^+ to be the latest time at which all vertices in $\mathcal{G}_2$ can be reached by v. Since $\mathcal{G}$ is temporally connected, there exists some label t_i of e such that $t_u^- < t_i < t_v^+$. Keeping t_i is thus sufficient for maintaining temporal connectivity from $\mathcal{G}_1$ to $\mathcal{G}_2$. A symmetrical argument can be used to find t_j for maintaining temporal connectivity from $\mathcal{G}_2$ to $\mathcal{G}_1$. Together, t_i and t_j maintain temporal connectivity in $\mathcal{G}$. ◀

Since trees contain only bridge edges, we have the following corollary.

▶ **Corollary 2.** *Two labels per edge are sufficient for maintaining temporal connectivity in temporal trees.*

Now one may ask the question whether two labels per edge could be sufficient for maintaining temporal connectivity in general temporal graphs. Indeed, maybe these last results can be generalized in some manner? Unfortunately, this is not the case.

▶ **Lemma 3.** *Two labels per edge are not sufficient for maintaining temporal connectivity in some temporal graphs.*

Proof idea. Consider the temporal graph given in Figure 8. The idea is to force the need of the three labels on the top edge, so as to maintain temporal connectivity from all u_i towards v_i. With this in mind, we complete the temporal graph so as to make it temporally connected (see Figure 9). Now, removing any of the three labels breaks temporal connectivity (from the corresponding u_i to v_i). ◀

In fact, the construction from Figures 8 & 9 can be easily extended, adding more rows of u_i and v_i couples with arcs such as the other u_i and v_i, so as to force even more labels on the top edge.

▶ **Corollary 4.** *Any constant number of labels per edge are not sufficient for maintaining temporal connectivity in some temporal graphs.*

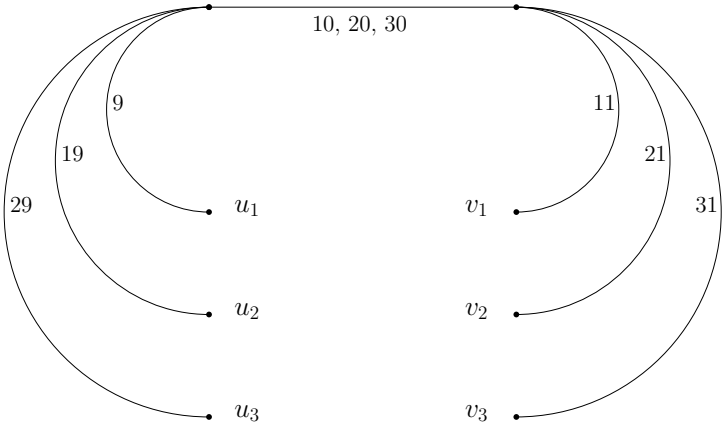

Figure 8 Construction of a temporal graph needing three labels on an edge to maintain temporal connectivity (Incomplete version).

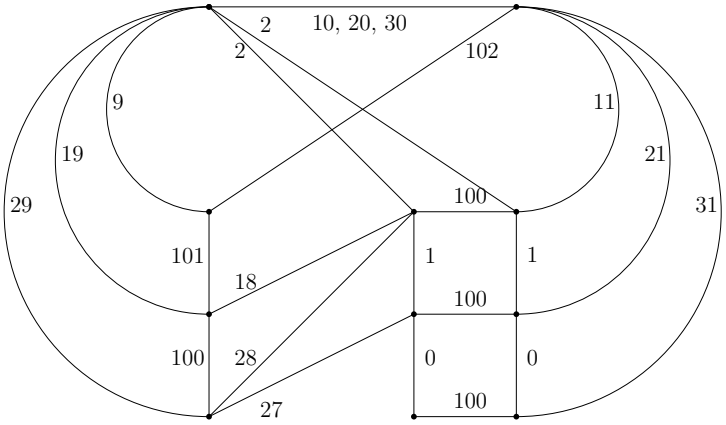

Figure 9 Construction of a temporal graph needing three labels on an edge to maintain temporal connectivity (Temporally connected version).

Open questions

- Does there exist other classes of graphs in which the number of labels per edge can be bounded?
- Is computing this number NP-hard?
- Can one force more labels on an edge than in Figures 8 & 9 (approx. $\frac{n}{3}$)?

The first of these open questions was partially treated during the Dagstuhl. Considering cycles graphs, a nice example needing $\frac{n}{4}$ labels per edge to maintain temporal connectivity was given by Malte Renken (see Figure 10). A bound on the number of labels per edge for cycles is thus at least $\frac{n}{4}$, establishing a large gap between tree graphs and cycles graphs.

References

1 Arnaud Casteigts, Joseph G Peters, and Jason Schoeters. Temporal cliques admit sparse spanners. *Journal of Computer and System Sciences*, 121:1–17, 2021.

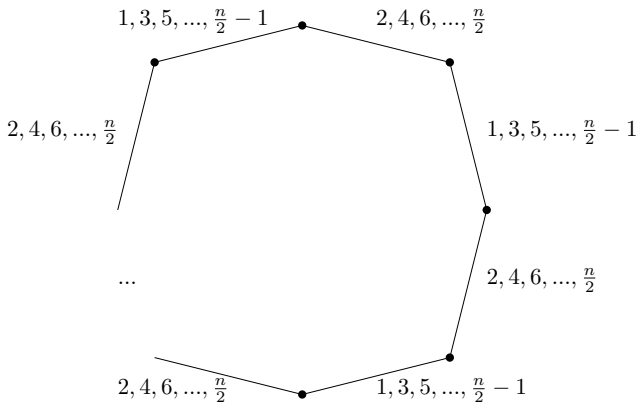

Figure 10 Temporal cycle needing $\frac{n}{4}$ labels per edge to maintain temporal connectivity.

4.7 Obtaining tighter bounds on the arrival time of non-strict exploration schedules in temporal graphs of temporal diameter 2

Jakob T. Spooner (University of Leicester, GB)

We present two open problems (originally posed in [2]) concerned with obtaining improved asymptotic bounds on the *arrival time of non-strict exploration schedules* for *non-strict temporal graphs with temporal diameter* 2. The overall goal is to close, as much as possible, the gap between an existing $\Omega(\sqrt{n})$ lower bound and $O(\sqrt{n}\log n)$ upper-bound. Non-strict temporal walks can be viewed as temporal walks that are allowed to make an unbounded but finite number of edge traversals in any given timestep, and have been studied within a variety of settings, see [1, 4, 3, 2] for examples. Temporal diameter as a temporal graph parameter is introduced by Definition 3, although we mention now that it is a relatively straightforward adaptation of diameter in static graphs.

Throughout the following we denote by $[n]$ the set of integers $\{1, 2, ..., n\}$ and by $[x, n]$ $(x \leq n)$ the set of integers $[n] - \{1, 2, ..., x-1\}$. We begin with a formal definition of a *non-strict temporal graph*:

▶ **Definition 1** (Non-strict temporal graph, $\mathcal{G}$)**.** A non-strict temporal graph $\mathcal{G} = \langle G_1, ..., G_L \rangle$ with vertex set $V := V(\mathcal{G})$ and lifetime L is an indexed sequence of partitions $G_i = \{C_{i,1}, ..., C_{i,s_i}\}$ of V, with $i \in [L]$. For all $i \in [L]$, every $v \in V$ satisfies $v \in C_{i,j_i}$ for a unique $j_i \in [s_i]$. We denote by $|G_i|$ the number of components in layer G_i.

▶ **Definition 2** (Non-strict temporal walk, W)**.** A non-strict temporal walk W with *duration* $k \in [L]$ through a non-strict temporal graph $\mathcal{G} = \langle G_1, ..., G_L \rangle$ is a sequence $W = C_{t,j_1}, C_{t+1,j_2}, ..., C_{t+k-1,j_k}$ (with $t \in [L-k+1]$) of components C_{i,j_i} such that:
(1) $C_{t+i-1,j_i} \in G_{t+i-1}$ and $j_i \in [s_{t+i-1}]$ $(i \in [1, k])$
(2) $C_{t+i-1,j_i} \cap C_{t+i,j_{i+1}} \neq \emptyset$ $(i \in [1, k-1])$
We say W has duration k, and that it *starts* in timestep t and *finishes* in timestep $t+k-1$. Furthermore, W *visits* the set of vertices $\bigcup_{i \in [0,k-1]} C_{t+i,j_{i+1}}$, and is called a *non-strict exploration schedule* of $\mathcal{G}$ if $\bigcup_{i \in [0,k-1]} C_{t+i,j_{i+1}} = V(\mathcal{G})$. The timestep $t' \in [t, t+k-1]$, during which W reaches the component that contains the final vertex $v \in V(\mathcal{G})$ such that v is contained in no component previously visited by W, is known as the *arrival time* of W.

We say that a pair of vertices $u, v \in V(\mathcal{G})$ are *connected* by a temporal walk $W = C_{t,j_1}, C_{t+1,j_2}, ..., C_{t+k-1,j_k}$ iff $u \in C_{t,j_1}$ and $v \in C_{t+k-1,j_k}$; u and v are *disconnected* if no such walk exists.

Let $\mathcal{G} = \langle G_1, ..., G_L \rangle$ be a non-strict temporal graph and let $sw(u, v, t)$ denote the length of a minimal duration non-strict temporal walk W in $\mathcal{G}$ that starts at timestep $t \in [L]$ and connects $u, v \in V(G)$. The following temporal graph parameter is a straightforward temporal analogue of the diameter of a static graph, and is central to our open problems of interest:

▶ **Definition 3** ((Non-strict) temporal diameter)**.** A non-strict temporal graph $\mathcal{G}$ has diameter bounded by d if

$$\max_{u,v \in V(\mathcal{G}), t \in [L-d+1]} sw(u, v, t) \leq d$$

Note the restriction on the range of t in the above maximisation function; this ensures that the temporal diameter condition holds for every length-d time interval throughout $\mathcal{G}$'s lifetime.

It was shown by the authors in [2] that there exists an infinite family of non-strict temporal graphs with diameter $d \geq 3$ and lifetime $\geq d(n-1)$, for which all exploration schedules for any graph in the family have arrival time $\Omega(n)$. This yields a tight asymptotic bound of $\Theta(n)$ on the amount of time required to explore arbitrary non-strict temporal graphs when $d \geq 3$, since any temporal graph with constant diameter $d \geq 3$ can be explored easily in $O(n)$ timesteps. On the other hand, for the case in which $d = 2$ it was shown (also in [2]) that exploration schedules with duration $O(\sqrt{n} \log n)$ are guaranteed to exist, and an infinite family of non-strict temporal graphs for which any exploration schedule has duration $\Omega(\sqrt{n})$ was also constructed. This leaves a gap of $\Theta(\log n)$ between the current best-known upper and lower bounds for the case when $d = 2$. We now state formally our open problems:

Open problems.

(1) Find a function $f(n) = \omega(\sqrt{n})$ such that there exists an infinite family $\mathcal{F}$ of non-strict temporal graphs with temporal diameter $d = 2$ and lifetime $L \geq 2(n-1)$ such that, for any graph $\mathcal{G} \in \mathcal{F}$, all exploration schedules of $\mathcal{G}$ have arrival time $\Omega(f(n))$.

(2) Find a function $f'(n) = o(\sqrt{n} \log n)$ such that any non-strict temporal graph $\mathcal{G}$ with diameter $d = 2$ and lifetime $L \geq 2(n-1)$ admits an exploration schedule with arrival time $O(f'(n))$.

Finding a function $g(n)$ that satisfies the requirements of both (1) and (2) would then provide an asymptotically tight bound on the arrival time of exploration schedules for arbitrary non-strict temporal graphs of diameter $d = 2$ and with lifetime $L \geq 2(n-1)$, and the problem would be resolved entirely.

Finally, we sketch the proofs of the known upper and lower bound. For the lower bound, let n be a square number and imagine the numbers from 1 to n arranged in a square grid with $\sqrt{n}$ rows and $\sqrt{n}$ columns. In odd time steps, the partition consists of the rows of the grid, and in even steps, it consists of the columns. This defines a temporal graph with temporal diameter 2. As each set in each partition has size $\sqrt{n}$, it is clear that $\Omega(\sqrt{n})$ steps are needed in an exploration schedule. For the upper bound, we observe that there cannot be two consecutive partitions that both have more than $\sqrt{n}$ sets: If the first of the two partitions has more than $\sqrt{n}$ sets, the smallest of those sets has size smaller than $\sqrt{n}$, and since that set must intersect all sets of the next partition, that next partition must contain less than $\sqrt{n}$ sets. In the partition that has at most $\sqrt{n}$ sets, we can visit the set that contains the most unvisited vertices, which must be at least a $1/\sqrt{n}$ fraction of the unvisited vertices. After $O(\sqrt{n} \log n)$ iterations of this operation, all vertices are visited.

References

1 Matthieu Barjon, Arnaud Casteigts, Serge Chaumette, Colette Johnen, and Yessin M. Neggaz. Testing Temporal Connectivity in Sparse Dynamic Graphs. *arXiv:1404.7634*, April 2014.

2 Thomas Erlebach and Jakob T. Spooner. Non-strict Temporal Exploration. In Andrea Werneck Richa and Christian Scheideler, editors, *27th International Colloquium on Structural Information and Communication Complexity (SIROCCO 2020)*, volume 12156 of *Lecture Notes in Computer Science*, pages 129–145. Springer, 2020.

3 Till Fluschnik, Hendrik Molter, Rolf Niedermeier, Malte Renken, and Philipp Zschoche. Temporal graph classes: A view through temporal separators. *Theoretical Computer Science*, 806:197–218, 2020.

4 David Kempe, Jon M. Kleinberg, and Amit Kumar. Connectivity and Inference Problems for Temporal Networks. *J. Comput. Syst. Sci.*, 64(4):820–842, 2002.

4.8 The cover time of random walks on temporal graphs

John Sylvester (University of Glasgow, GB)

Graph Model: Sequence of graphs $\mathcal{G} = G_1, G_2, \ldots$ on the same set of n vertices, such that each G_i is connected and has a loop at each vertex.

Simple Random Walk (SRW): Start at some vertex. Then at each time step choose a neighbour of the current vertex uniformly at random.

Problem: Can we get a tight bound on the expected time for a SRW to visit all vertices (cover time) of any $\mathcal{G}$? We note that $\mathcal{G}$ is fixed in advance; it cannot be adapted based on the trajectory of the walk.

Question: Can we close the gap for general case?

What is known?

- In [1] an example is given with cover time $2^{\Omega(n)}$.
- A simple upper bound of $n^{\mathcal{O}(n)} = 2^{\mathcal{O}(n \log n)}$ is given in [1].
- If if we take an "ultra lazy walk" [1], or a lazy random walk on a graph sequence where the stationary distribution π is static [2], then the cover time is $\tilde{\mathcal{O}}(n^3)$.

References

1 Chen Avin, Michal Koucký, and Zvi Lotker. Cover time and mixing time of random walks on dynamic graphs. *Random Struct. Algorithms*, 52(4):576–596, 2018.

2 Thomas Sauerwald and Luca Zanetti. Random walks on dynamic graphs: Mixing times, hitting times, and return probabilities. In Christel Baier, Ioannis Chatzigiannakis, Paola Flocchini, and Stefano Leonardi, editors, *46th International Colloquium on Automata, Languages, and Programming, ICALP 2019, July 9-12, 2019, Patras, Greece*, volume 132 of *LIPIcs*, pages 93:1–93:15. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2019.

Remote Participants

- Evripidis Bampis
Sorbonne University – Paris, FR
- Julien Baste
University of Lille, FR
- Prithwish Basu
BBN Technologies – Cambridge, US
- Cristiano Bocci
University of Siena, IT
- Hans L. Bodlaender
Utrecht University, NL
- Binh-Minh Bui-Xuan
UPMC – Paris, FR
- Benjamin Bumpus
University of Glasgow, GB
- Chiara Capresi
University of Siena, IT
- Arnaud Casteigts
University of Bordeaux, FR
- Rémy Cazabet
University of Lyon, FR
- Andrea Clementi
University of Rome "Tor Vergata", IT
- Timothée Corsini
University of Bordeaux, FR
- Pierluigi Crescenzi
Gran Sasso Science Institute, IT
- Jessica Enright
University of Glasgow, GB
- Thomas Erlebach
University of Leicester, GB
- Bruno Escoffier
Sorbonne University – Paris, FR
- Till Fluschnik
TU Berlin, DE
- Samuel Hand
University of Glasgow, GB
- Petter Holme
Tokyo Institute of Technology, JP
- Frank Kammer
THM – Gießen, DE
- Mikko Kivelä
Aalto University, FI
- Ralf Klasing
CNRS and University of Bordeaux, France
- Christian Komusiewicz
Universität Marburg, DE
- Michael Lampis
University Paris-Dauphine, FR
- Zvi Lotker
Bar-Ilan University, IL
- Andrea Marino
Univerisità degli Studi di Firenze, IT
- Kitty Meeks
University of Glasgow, GB
- Nicole Megow
Universität Bremen, DE
- George B. Mertzios
Durham University, GB
- Othon Michail
University of Liverpool, GB
- Hendrik Molter
TU Berlin, DE
- Vincenzo Nicosia
Queen Mary University of London, GB
- Rolf Niedermeier
TU Berlin, DE
- William Pettersson
University of Glasgow, GB
- Yoann Pigné
University of Le Havre, FR
- Michael Raskin
TU München, DE
- Malte Renken
TU Berlin, DE
- Eric Sanlaville
University of Le Havre, FR
- Jason Schoeters
University of Bordeaux, FR
- Fiona Skerman
Uppsala University, SE
- Martin Skutella
TU Berlin, DE
- Manuel Sorge
TU Wien, AT
- Paul Spirakis
University of Liverpool, GB
- Jakob Spooner
University of Leicester, GB
- Ondrej Suchý
Czech Technical University – Prague, CZ
- John Sylvester
University of Glasgow, GB
- Nikolaj Tatti
University of Helsinki, FI
- Amitabh Trehan
Durham University, GB
- Mathilde Vernet
University of Le Havre, FR
- Tiphaine Viard
Telecom Paris, FR
- Victor Zamaraev
University of Liverpool, GB
- Christos Zaroliagis
University of Patras, GR
- Philipp Zschoche
TU Berlin, DE