# Brief Announcement: Local Distributed Algorithms in Highly Dynamic Networks

**Philipp Bamberger**
University of Freiburg, Georges-Köhler-Allee 106, 79110 Freiburg, Germany
philipp.bamberger@cs.uni-freiburg.de

**Fabian Kuhn**
University of Freiburg, Georges-Köhler-Allee 106, 79110 Freiburg, Germany
kuhn@cs.uni-freiburg.de

**Yannic Maus**
University of Freiburg, Georges-Köhler-Allee 106, 79110 Freiburg, Germany
yannic.maus@cs.uni-freiburg.de

**Abstract**

We define a generalization of local distributed graph problems to (synchronous round-based) dynamic networks and present a framework for developing algorithms for these problems. We require two properties from our algorithms: (1) They should satisfy non-trivial guarantees in every round. The guarantees should be stronger the more stable the graph has been during the last few rounds and they coincide with the definition of the static graph problem if no topological change appeared recently. (2) If a constant neighborhood around some part of the graph is stable during an interval, the algorithms quickly converge to a solution for this part of the graph that remains unchanged throughout the interval.

We demonstrate our generic framework with two classic distributed graph, namely *(degree+1)-vertex coloring* and *maximal independent set (MIS)*.

**2012 ACM Subject Classification** Networks → Network algorithms

**Keywords and phrases** dynamic networks, distributed graph algorithms, MIS, vertex coloring

**Digital Object Identifier** 10.4230/LIPIcs.DISC.2018.42

**Funding** Second and third author were supported by ERC Grant No. 336495 (ACDC).

## 1 Introduction

Many modern computer systems are built on top of large-scale networks such as the Internet, the world wide web, wireless ad hoc and sensor networks, or peer-to-peer networks. Often, the network topology of such systems is inherently dynamic: nodes can join or leave at any time and (e.g., in the context of overlay networks or mobile wireless networks) communication links might appear and disappear constantly. As a consequence, we aim to develop distributed algorithms that can cope with a potentially *highly dynamic network topology* and to understand what can and what cannot be computed in a dynamic network. In particular, we investigate techniques to develop distributed dynamic network solutions for *distributed graph problems* and more specifically for solving *local distributed graph problems* such as computing a graph coloring or a maximal independent set (MIS) of the network graph (see, e.g., [1, 5, 6]).

Most previous work on solving distributed graph problems in the dynamic setting is of the following flavor [3, 2]: After one or more topology changes, the algorithm has a *recovery period* to fix its output and the network does not undergo any changes during this recovery period. If further dynamic changes occur while recovering from a previous change such an


32nd International Symposium on Distributed Computing (DISC 2018).
Editors: Ulrich Schmid and Josef Widder; Article No. 42; pp. 42:1–42:4
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

algorithm loses its guarantees and it might even fail to provide any guarantees at all. We therefore follow a different approach. Our randomized algorithms constantly adapt to a changing environment. They always satisfy non-trivial guarantees, no matter how dynamic the network is. The guarantees become stronger if the network is less dynamic. In particular, if the network becomes static in a constant neighborhood around some part of the network, the solution of that part also converges to a static solution after a short time.

In the present paper, we develop a framework to build distributed dynamic network algorithms with the aforementioned properties and apply the framework to two of the classic distributed graph problems, namely, the problem of computing a maximal independent set (MIS) and the problem of computing a vertex coloring of the network graph. We however note that the general framework also applies to various additional graph problems. For example, it seems particularly suitable to convert classic covering or packing optimization problems to the dynamic setting. Examples for such problems are minimum dominating set, minimum vertex cover, or maximum matching. For the coloring problem, our algorithm guarantees that after two nodes are joined by an edge, they can only have the same color for a short time. Further, the total number of colors used is still essentially upper bounded by the maximum degree of the network as in the static version of the problem. In the context of dynamic networks, the degree of some node $v$ at a time $t$ is defined to be the number of distinct neighbors $v$ has had during the last few rounds.

## 2 Model, Contribution & Techniques

We model a dynamic network as a synchronous system over a set $V$ of $n$ potential nodes. Time is divided into rounds and in each round $r = 0, 1, 2, \ldots$, there is a communication graph $G_r = (V_r, E_r)$. We generally assume that nodes can wake up gradually, however for the purpose of this summary, we assume that all nodes wake up initially and we thus have $V_r = V$ for all $r \geq 1$. We consider graph problems that can be decomposed into two parts that are given by a *packing* and a *covering* graph property. Essentially, a packing property is a graph property that remains true when removing edges and a covering property is a graph property that remains true when adding edges. In addition, we assume that the validity of a solution can be checked locally, i.e., by evaluating it in the constant neighborhood of every node [4]. For example, the problem of finding an MIS on a graph $G$ can be decomposed into the problem of finding a subset $S$ of the nodes such that no two neighbors are in $S$ (packing property) and $S$ is a dominating set of $G$ (covering property). For the (degree+1)-coloring problem, the requirement that the vertex coloring is proper is a packing property and the requirement that the color of a node $v$ is from $\{1, \ldots, \deg(v) + 1\}$ is a covering property. For a given graph problem and an integer parameter $T \geq 1$, we say that a given solution is a *$T$-dynamic solution* at time $r$ if a) the solution satisfies the packing property for the *intersection graph* $G_r^{T\cap} = G_{r-T+1} \cap G_{r-T+1} \cap \ldots \cap G_r$ (i.e., the graph that contains all edges that have been present throughout the last $T$ rounds), and b) the solution satisfies the covering property for the *union graph* $G_r^{T\cup} = G_{r-T+1} \cup G_{r-T+1} \cup \ldots \cup G_r$ (i.e., the graph that contains all edges that have been present at least once in the last $T$ rounds). We believe that this provides a natural generalization of a static graph problem to the dynamic context. Note that the dynamic guarantees are stronger the less dynamic the graph is and if the graph has been static during rounds $r - T + 1, \ldots, r$, a $T$-dynamic solution at time $r$ is the same as a static solution for the given graph problem for the graph $G_r$ in round $r$.

When designing a distributed algorithm for a given dynamic graph problem, we require that for some $T \geq 1$, the algorithm outputs a $T$-dynamic solution after each round $r$. Assume that we can construct an algorithm $\mathcal{A}$ such that if all nodes start $\mathcal{A}$ in round 1, after round $T$, $\mathcal{A}$ outputs a $T$-dynamic solution w.r.t. to the first $T$ graphs (i.e., a solution that satisfies the packing property for $G_T^{T\cap}$ and the covering property for $G_T^{T\cup}$). Given such an algorithm $\mathcal{A}$, we can in principle design an algorithm that always outputs a $T$-dynamic solution by just starting a new instance of $\mathcal{A}$ in every round and outputting the solution of an instance started in round $r+1$ after round $r+T$. However, such a solution would not be satisfactory because especially if $\mathcal{A}$ is randomized, the output might change completely from round to round even if the graph is only mildly dynamic or even static. Thus we refine this approach and define two *abstract* types of algorithms to deal with dynamic graph problems.

For two positive integers $T$ and $\alpha$, we say that an algorithm $\mathcal{A}_1$ is a $(T,\alpha)$*-network-static algorithm* for a given dynamic graph problem if it satisfies the following properties. At the end of each round $r \geq 1$, the algorithm outputs a valid *partial* solution for the graph $G_r$ (In a partial solution, nodes are allowed to output $\perp$ and for each node $v$ that outputs a value $\neq \perp$, there is an extension of the partial solution such that the packing property for $v$ is satisfied and the covering property for $v$ is satisfied for all extensions of the partial solution). In addition, if the $\alpha$-neighborhood of some node $v$ remains static in some interval $[r, r_2]$, $v$ must output a fixed value $\neq \perp$ throughout the interval $[r+T, r_2]$. Further, for a positive integer $T$, we say that an algorithm $\mathcal{A}_2$ is a $T$*-dynamic algorithm* for a given dynamic graph problem if it satisfies the following property. Let $r \geq 1$ be some round and assume that we are given a valid partial solution for $G_r$. If $\mathcal{A}_2$ is started in round $r+1$, at the end of round $r+T-1$, it outputs a $T$-dynamic solution that extends the given partial solution for $G_r$. The following theorem shows that a $T_1$-dynamic algorithm and a $(T_2,\alpha)$-network-static algorithm can be combined to obtain a distributed algorithm that always outputs a $T_1$-dynamic solution while behaving well if the graph is locally static for sufficiently long. Our framework thus allows to separate the two tasks of (1) always outputting a $T$-dynamic solution and (2) providing a locally stable output if the network is locally static.

▶ **Theorem 1.** *Let $T_1$ and $T_2$ be positive integers, $\mathcal{P}$ a packing, and $\mathcal{C}$ a covering problem. Given a $T_1$-dynamic algorithm and a $(T_2,\alpha)$-network-static algorithm for $(\mathcal{P},\mathcal{C})$, one can combine both algorithms to an algorithm such that:*

1. *(dynamic solution) Its output in round $r$ is a $T_1$-dynamic solution for $(\mathcal{P},\mathcal{C})$ .*
2. *(locally static) If the graph is static in the $\alpha$-neighborhood of a node $v \in V_r$ in all rounds in an interval $[r, r_2]$, then the output of $v$ does not change for all rounds in $[r+T_1+T_2, r_2]$.*

## 3 Two Sample Problems: MIS & Vertex-Coloring

We show how to apply the above framework to two of the classic local symmetry breaking problems: computing a vertex coloring and computing an MIS of the network graph. In both cases, we only slightly adapt existing randomized algorithms (e.g., [5, 1, 6]) to obtain the results. We see the relatively simple adaptation – compared to a huge and heavy machinery – of existing static algorithms to the dynamic case as a strength of the framework in terms of practicability. Of course, some of the existing proofs need additional care and some algorithms, e.g., the MIS algorithm by Ghaffari [5], need some (crucial) modifications to assure termination in the dynamic setting.

▶ **Corollary 2.** *There is a $T = O(\log n)$ and an algorithm that, w.h.p., outputs a $T$-dynamic solution for (degree+1)-coloring (for MIS) in every round and the output of any node $v$ is static in all rounds in the interval $[r+2T, r_2]$ if the 2-neighborhood of $v$ is static in all rounds in the interval $[r, r_2]$.*

## References

1 N. Alon, L. Babai, and A. Itai. A fast and simple randomized parallel algorithm for the maximal independent set problem. *J. of Algorithms*, 7(4):567–583, 1986.

2 S. Assadi, K. Onak, B. Schieber, and S. Solomon. Fully dynamic maximal independent set with sublinear update time. In *Proc. 50th ACM Symp. on Theory of Comp. (STOC)*, 2018.

3 K. Censor-Hillel, E. Haramaty, and Z.S. Karnin. Optimal dynamic distributed MIS. In *Proc. 35th ACM Symp. on Principles of Distr. Computing (PODC)*, pages 217–226, 2016.

4 P. Fraigniaud, A. Korman, and D. Peleg. Local distributed decision. In *Proc. 52nd Symp. on Foundations of Computer Sc. (FOCS)*, pages 708–717. IEEE Computer Society, 2011.

5 M. Ghaffari. An improved distributed algorithm for maximal independent set. In *Proc. 27th ACM-SIAM Symp. on Discrete Algorithms (SODA)*, pages 270–277, 2016.

6 M. Luby. A simple parallel algorithm for the maximal independent set problem. *SIAM J. Comp.*, 15:1036–1053, 1986.