

# Brief Announcement: Distributed Maximum Flow in Planar Graphs

Yaseen Abd-Elhaleem  

Department of Computer Science, University of Haifa, Israel

Michal Dory  

Department of Computer Science, University of Haifa, Israel

Merav Parter  

Department of Computer Science and Applied Mathematics,  
Weizmann Institute of Science, Rehovot, Israel

Oren Weimann  

Department of Computer Science, University of Haifa, Israel

---

## Abstract

---

The dual of a planar graph  $G$  is a planar graph  $G^*$  that has a vertex for each face of  $G$  and an edge for each pair of adjacent faces of  $G$ . The profound relationship between a planar graph and its dual has been the algorithmic basis for solving numerous (centralized) classical problems on planar graphs involving distances, flows, and cuts. In the distributed setting however, the only use of planar duality is for finding a recursive decomposition of  $G$  [DISC 2017, STOC 2019].

In this paper, we extend the distributed algorithmic toolkit (such as recursive decompositions and minor-aggregations) to work on the dual graph  $G^*$ . These tools can then facilitate various algorithms on  $G$  by solving a suitable dual problem on  $G^*$ . Given a directed planar graph  $G$  with hop-diameter  $D$ , our key result is an  $\tilde{O}(D^2)$ -round algorithm<sup>1</sup> for Single Source Shortest Paths on  $G^*$ , which then implies an  $\tilde{O}(D^2)$ -round algorithm for Maximum  $st$ -Flow on  $G$ . Prior to our work, no  $\tilde{O}(\text{Poly}(D))$ -round algorithm was known for Maximum  $st$ -Flow. We further obtain a  $D \cdot n^{o(1)}$ -rounds  $(1 + \epsilon)$ -approximation algorithm for Maximum  $st$ -Flow on  $G$  when  $G$  is undirected and  $s$  and  $t$  lie on the same face. Finally, we give a near optimal  $\tilde{O}(D)$ -round algorithm for computing the weighted girth of  $G$ .

The main challenges in our work are that  $G^*$  is not the communication graph (e.g., a vertex of  $G$  is mapped to multiple vertices of  $G^*$ ), and that the diameter of  $G^*$  can be much larger than  $D$  (i.e., possibly by a linear factor). We overcome these challenges by carefully defining and maintaining subgraphs of the dual graph  $G^*$  while applying the recursive decomposition on the primal graph  $G$ . The main technical difficulty, is that along the recursive decomposition, a face of  $G$  gets shattered into (disconnected) components yet we still need to treat it as a dual node.

**2012 ACM Subject Classification** Theory of computation → Distributed algorithms

**Keywords and phrases** Maximum flow, shortest paths, planar graphs, distributed computing

**Digital Object Identifier** 10.4230/LIPIcs.DISC.2024.40

**Funding** O. Weimann was supported in part by Israel Science Foundation grant 810/21.

## 1 Introduction

Distributed algorithms for network optimization problems have a long and rich history. These problems are commonly studied under the CONGEST model [18] where the network is abstracted as an  $n$ -vertex graph  $G = (V, E)$  with hop-diameter  $D$ ; communications occur in synchronous rounds, and per round,  $O(\log n)$  bits can be sent along each edge. A sequence of

---

<sup>1</sup> The  $\tilde{O}(\cdot)$  notation is used to omit poly log  $n$  factors.



breakthrough results provided  $\tilde{O}(D + \sqrt{n})$ -round algorithms for fundamental graph problems, such as minimum spanning tree (MST) [6], approximate shortest-paths [16], minimum cuts [3], and approximate flow [9]. For general  $n$ -vertex graphs  $\tilde{O}(D + \sqrt{n})$  rounds is known to be near optimal, existentially [20].

A major and concentrated effort has been invested in designing improved solutions for special graph families that escape the topology of the worst-case lower bound graphs of [20]. The lower bound graph is sparse, and of arboricity two, so it belongs to many graph families. Arguably, one of the most interesting non-trivial families that escapes it, is the family of planar graphs. Thus, a significant focus has been given to the family of planar graphs, due to their frequent appearance in practice and because of their rich structural properties. In their seminal work, Ghaffari and Haeupler [7, 8] initiated the line of distributed planar graph algorithms based on the notion of *low-congestion shortcuts*. The latter serves the communication backbone for obtaining  $\tilde{O}(D)$ -round algorithms for MST [8], minimum cut [8, 11] and approximate shortest paths [21, 22] in planar graphs.

An additional key tool in working with planar graphs, starting with the seminal work of Lipton and Tarjan [15], is that of a planar *separator path*: a path whose removal from the graph leaves connected components that are a constant factor smaller. Ghaffari and Parter [10] presented a  $\tilde{O}(D)$ -round randomized algorithm for computing a cycle separator of size  $O(D)$  which consists of a separator path plus one additional edge (that is possibly a *virtual* edge that is not in  $G$ ). By now, planar separators are a key ingredient in a collection of  $\tilde{O}(\text{poly}(D))$ -round solutions for problems such as DFS [10], distance computation [14], and reachability [17]. An important aspect of the planar separator algorithm of [10] is that it employs a computation on the dual graph, by communicating over the primal graph.

**Primal maximum flow via dual SSSP.** Our goal in this paper is to expand the algorithmic toolkit for performing computation on the dual graph. This allows us to exploit the profound algorithmic duality in planar graphs, in which solving a problem  $A$  in the dual graph provides a solution for problem  $B$  in the primal graph. Within this context, our focus is on the *Maximum  $st$ -Flow* problem (in directed planar graphs with edge capacities) which asks to compute the maximum amount of flow that can be sent from a source vertex  $s$  to a target vertex  $t$  while respecting edge capacities. The Maximum  $st$ -flow problem is arguably one of the most classical problems in theoretical computer science, extensively studied since the 50's, and still admitting breakthrough results in the sequential setting, such as the recent almost linear time algorithm by Chen, Kyng, Liu, Peng, Gutenberg and Sachdeva [1]. Despite persistent attempts over the years, our understanding of the distributed complexity of the Maximum  $st$ -flow problem is still quite lacking. For general *undirected*  $n$ -vertex graphs, there is a  $(1 + o(1))$ -approximation algorithm that runs in  $(\sqrt{n} + D)n^{o(1)}$  rounds, by Ghaffari, Karrenbauer, Kuhn, Lenzen and Patt-Shamir [9]. For directed  $n$ -vertex *planar* graphs, a  $D \cdot n^{1/2+o(1)}$ -round exact algorithm has been given by de Vos [2]. No better tradeoffs are known for undirected planar graphs. In lack of any  $\tilde{O}(\text{poly}(D))$ -round maximum  $st$ -flow algorithm for directed planar graphs (not even when allowing approximation) we ask:

► **Question 1.1.** *Is it possible to compute the maximum  $st$ -flow in directed planar graphs within  $\tilde{O}(\text{poly}(D))$  rounds?*

In directed planar graphs with integral edge-capacities, it is known from the 80's [23] that the maximum  $st$ -flow can be found by solving at most  $\log \lambda$  instances of *Single Source Shortest Paths* (SSSP) with positive and negative edge-lengths on the *dual* graph  $G^*$ , where  $\lambda$  is the maximum  $st$ -flow value. Their algorithm exploits the fact that any capacity-respecting

flow in  $G$  can be decomposed into (1) a not necessarily capacity-respecting  $st$ -flow of the same value, and (2) a feasible *circulation*. Since  $G$  is planar, a feasible circulation can be obtained by a *feasible potential* over its faces (i.e., nodes of  $G^*$ ); It is known that distances in  $G^*$  from any source constitute a feasible potential over its nodes. Hence, dual SSSP immediately implies primal maximum  $st$ -flow. We answer Question 1.1 in the affirmative by designing a  $\tilde{O}(D^2)$ -round SSSP algorithm on the dual graph  $G^*$ . Our algorithm works in the most general setting (i.e. when  $G^*$  is directed and has positive and negative integral edge-lengths) and matches the fastest known exact SSSP algorithm in the primal graph. We show:

► **Theorem 1.2** (Exact Maximum  $st$ -Flow in Directed Planar Graph). *There is a randomized distributed algorithm that given an  $n$ -vertex directed planar communication network  $G$  with hop-diameter  $D$  and integral edge-capacities, and two vertices  $s, t$ , computes the maximum  $st$ -flow value and assignment in  $\tilde{O}(D^2)$  rounds.*

No prior  $\tilde{O}(\text{poly}(D))$  algorithm has been known for this problem, not even when allowing a constant approximation. We further improve the running time to  $D \cdot n^{o(1)}$  rounds for the case of a  $(1 + \epsilon)$ -approximation, provided that  $G$  is undirected and that  $s$  and  $t$  both lie on the same face:

► **Theorem 1.3** (Approximate Maximum  $st$ -Flow in Undirected  $st$ -Planar Graphs). *There is a randomized distributed algorithm that given an  $n$ -vertex undirected planar communication network  $G$  with hop-diameter  $D$  and integral edge-capacities, and two vertices  $s, t$  lying on the same face, computes a  $(1 + \epsilon)$ -approximation of the maximum  $st$ -flow value and a matching assignment in  $D \cdot n^{o(1)}$  rounds.*

This latter result is also obtained by exploiting the duality between flows and distances. Our algorithm is based on an approximate SSSP algorithm that runs in  $D \cdot n^{o(1)}$  rounds in planar graphs [22]. Our implementation of the algorithm on the dual graph matches its round complexity in the primal graph. Our almost-optimal round complexity improves significantly over the current algorithm for general graphs that runs in  $(\sqrt{n} + D)n^{o(1)}$  rounds [9].

**Primal weighted girth via dual cuts.** A distance parameter of considerable interest is the network *girth*. For unweighted graphs, the girth is the length of the smallest cycle in the graph. For weighted graphs, the girth is the cycle of minimal total edge weight. Distributed girth computation has been studied over the years mainly for general  $n$ -vertex unweighted graphs. Frischknecht, Holzer and Wattenhofer [5] provided an  $\Omega(\sqrt{n})$ -round lower bound for computing a  $(2 - \epsilon)$  approximation of the unweighted girth. The state-of-the-art upper bound for the unweighted girth problem is a  $(2 - \epsilon)$  approximation in  $\tilde{O}(n^{2/3} + D)$  rounds, obtained by combining the works of Peleg, Roditty and Tal [19] and Holzer and Wattenhofer [12]. The weighted girth problem has been shown to admit a near-optimal lower bound of  $\tilde{\Omega}(n)$  rounds in general graphs by Hua, Qian, Yu, Shi and Jin [13]. Turning to planar graphs, Parter [17] devised a  $\tilde{O}(D^2)$  round algorithm for computing the weighted girth in directed planar graphs via SSSP computations. For undirected and unweighted planar graphs, the (unweighted) girth can be computed in  $\tilde{O}(D)$  rounds by replacing the  $\tilde{O}(D^2)$ -round SSSP algorithm by a  $O(D)$ -round BFS algorithm. In light of this gap, we ask:

► **Question 1.4.** *Is it possible to compute the weighted girth of an undirected weighted planar graph within (near-optimal)  $\tilde{O}(D)$  rounds?*

We answer this question in the affirmative by taking a different, non distance-related, approach than that taken in prior work. Our  $\tilde{O}(D)$  round algorithm exploits the useful duality between cuts and cycles. We present a dual framework of the *minor-aggregation*

model. Using it, we can simulate the primal exact minimum cut algorithm of Ghaffari and Zuzic [11] on the dual graph. This dual simulation matches the primal round complexity. The solution to the dual cut problem immediately yields a solution to the primal weighted girth problem. We show:

► **Theorem 1.5** (Planar Weighted Girth). *There is a randomized distributed algorithm that given an  $n$ -vertex undirected weighted planar communication network  $G$  with hop-diameter  $D$ , computes the weighted girth (and finds a corresponding cycle) in  $\tilde{O}(D)$  rounds.*

As the algorithmic power of the minor-aggregation model is currently limited to undirected graphs, it will be interesting to devise improved girth algorithms for directed planar graphs as well.

## 2 Technical Overview

Our results are based on two main (primal) tools that we extend to work on the dual graph: Minor Aggregation and Bounded Diameter Decomposition. We highlight the key ideas of these techniques and the challenges encountered in their dual implementation. For all the algorithms that we implement in the dual graph, we match the primal round complexity.

### 2.1 Minor-Aggregations in the Dual

An important recent development in the field of distributed computing was a new model of computation, called the *minor-aggregation model* introduced by Zuzic <sup>Ⓘ</sup> Goranci <sup>Ⓘ</sup> Ye <sup>Ⓘ</sup> Haeupler <sup>Ⓘ</sup> Sun [22], then extended by Ghaffari and Zuzic [11] to support working with *virtual nodes* added to the input graph. Recent state-of-art algorithms for various classical problems can be formulated in the minor-aggregation model (e.g., the exact min-cut algorithm of [11], and the undirected shortest paths approximation algorithms of [21, 22]). Motivated by the algorithmic power of this model, we provide an implementation of the minor aggregation model in the dual graph. The round complexity of our implementation matches its primal complexity. As noted by [22], minor aggregations can be implemented by solving the (simpler) part-wise aggregation task, where one needs to compute an aggregate function in a collection of vertex-disjoint connected parts of the graph. The planar separator algorithm of [10] implicitly implements a part-wise aggregation algorithm in the dual graph. Our contribution is in providing an explicit and generalized implementation of the dual part-wise aggregation problem and using it to implement the minor-aggregation model in the dual graph. We then use this algorithm for computing the exact minimum weighted cut in the dual graph, which by duality provides a solution to the weighted girth problem in the primal graph. We also use it to simulate the recent approximate SSSP by [22] in the dual graph, leading to our approximate max  $st$ -flow algorithm. Since currently there are fast SSSP minor-aggregation algorithms only for undirected graphs with positive weights, this approach leads to an approximate max  $st$ -flow algorithm in undirected planar graphs when  $s$  and  $t$  are on the same face. To solve the more general version of the max flow problem, we need additional tools described next.

---

<sup>2</sup> <sup>Ⓘ</sup> is used to denote that the authors' ordering is randomized, as the authors ask to cite their work this way.

## 2.2 SSSP in the Dual

**Bounded diameter decompositions.** The Bounded Diameter Decomposition (BDD), introduced by Li and Parter [14] is an algorithmic tool for solving graph problems in a divide-and-conquer manner, in the CONGEST model. Intuitively, the BDD plays an analogous role to planar *separator decomposition* in the centralized setting, in the following sense. The centralized divide-and-conquer approach repetitively removes the separator vertices from the graph and recurses on the remaining subgraphs that are (a constant factor) smaller in size. For the algorithmic applications it is only important that the size of the separator and the remaining subgraphs are small. In the distributed setting, it is desired to obtain a separator of  $O(D)$  size in all recursive subgraphs, allowing a fast ( $\tilde{O}(\text{poly}(D))$ -round) broadcast of separator related information (e.g. pairwise distances), which is in particular useful for a divide-and-conquer approach. While this can be obtained in the first recursion level, once we remove the first separator, the remaining subgraphs are smaller in size, but they may have considerably larger diameter, even up to  $\Theta(n)$ . Allowing the algorithm to use the other subgraphs, to provide shortcut paths, creates the possibility of *congestion* as now many subproblems may need to use the same edge. These two opposing dilation and congestion forces are settled by the BDD algorithm, in a near optimal manner. The BDD provides a hierarchical graph decomposition of  $O(\log n)$  layers. The subgraphs (called bags) obtained in each recursive level are nearly edge-disjoint (sharing only the edges of the separator) and of diameter  $\tilde{O}(D)$ . I.e., allowing one to apply an algorithm on all bags of the same level simultaneously without incurring more than a poly  $\log n$  factor overhead in the round complexity of running the same algorithm on the original network of communication (which has a small diameter of  $D$ ). There might be as many as  $\tilde{O}(D)$  children of a bag (all a constant factor smaller than their parent bag). However, the number of child bags has no importance, as we can work on all of them in parallel. BDDs have proven to be useful for divide-and-conquer CONGEST algorithms on the (primal) graph  $G$  (e.g. distance labeling, diameter approximation, routing schemes and reachability [4, 14, 17]).

**Our approach: recurse on primal, solve on dual.** Due to the wide applicability of BDDs for solving graph problems in planar graphs, we would like to exploit them also for solving problems on the dual graph. A natural approach could be to simulate a BDD algorithm on the dual network. However, there are several barriers. First, it is unclear how to simulate a general algorithm on the dual network, as this is not our communication network. Second, the diameter of the dual graph can be large (possibly linear) and the running time of the algorithm depends on the graph diameter. To overcome it we take a different approach. We apply a divide-and-conquer approach on the dual graph  $G^*$  by using the BDD computation on the primal graph  $G$ . Taking a dual lens on the primal BDD introduces several challenges that arise when one needs to define the dual bags from the given primal bags. This primal to dual translation is rather non-trivial due to critical gaps that arise when one needs to maintain information w.r.t faces of  $G$ , rather than vertices of  $G$ , over the recursive BDD procedure, as we elaborate next.

**Challenge I: shattered faces.** Throughout, we refer to faces of the primal graph  $G$  as *nodes* (rather than vertices) of the dual graph  $G^*$ . In the primal graph, a vertex is an atomic unit, which keeps its identity throughout the computation. The situation in the dual graph is considerably more involved. Consider a constant diameter (primal) graph with a face  $f$  with  $\Theta(n)$  edges. Throughout the recursive BDD, the vertices of the face  $f$  are split among multiple faces, and eventually  $f$  is shattered among possibly a linear number of leaf bags.

This means that a node in a dual bag does no longer correspond to a face  $f$  of the primal bag, but rather to a subset of edges of  $f$ . This creates a challenge in the divide-and-conquer computation, where one needs to assemble fragments of information from multiple bags.

**Challenge II: virtual edges.** For the BDD implementation, it is crucial for the separator to be a *simple cycle*. This was obtained in [10] by adding a single artificial (virtual) edge. The virtual edges are embedded in a way that preserve planarity, but they require special treatment since they are not part of the communication graph. In the primal BDD, the role of the virtual edge is limited to defining the child bags, and can be discarded afterwards. This use-and-forget mindset can no longer be applied in our setting. We elaborate. In a primal divide-and-conquer algorithm, the separator is thought of as a subset of vertices where each path in  $G$  from one side of the separator to the other side must intersect the separator at a *vertex*. In our case, since we are working with the dual graph, we have that paths in  $G^*$  from one side of separator to the other side must intersect the separator at an *edge*. That is, we view the separator as an edge-separator (i.e., a cut in the dual graph) not a vertex-separator. This is challenging because now we need to take into account the virtual edge that is not a real edge of  $G$  but is an edge of the separator.









**Our approach.** To deal with the above challenges, we work as follows. First, we analyze the way faces are partitioned during the BDD algorithm. We prove that in each bag  $X$  of the BDD there is at most one face of  $G$  that can be partitioned between the different child bags of  $X$  and was not partitioned in previous levels, this is exactly the face  $f$  that contains the virtual edge of the bag. We call the different parts of  $f$  that appear in different child bags *face-parts*. Since the decomposition has  $O(\log n)$  levels, overall we have at most  $O(\log n)$  face-parts in each bag. For a bag  $X$ , we define a dual bag  $X^*$  as follows. The nodes of  $X^*$  are the faces and face-parts of  $G$  that appear in  $X$ , where two nodes are connected by a dual edge if they share a primal edge in  $X$ . If  $X = G$ , this is exactly the dual graph  $G^*$ . The nodes  $g$  in  $X^*$  will be simulated by the vertices of the corresponding face or face-part, and each dual edge adjacent to  $g$  will be known by one of these vertices.

Our next goal is to use the decomposition in order to compute distances in the dual graph. More concretely, we compute distance labels. Each node in a bag  $X^*$  gets a short label of size  $\tilde{O}(D)$ , such that given the labels of two nodes in  $X^*$  we can deduce their distance. We take a recursive approach. We first compute distance labels in the child bags of a dual bag  $X^*$  and then combine them to compute distances in  $X^*$ . To do so, we identify a set of  $\tilde{O}(D)$  special nodes  $F_x$  in  $X^*$ , that contain nodes adjacent to the separator, as well as nodes corresponding to faces or face-parts that are partitioned between child bags of  $X^*$ . We prove that any shortest path in  $X^*$  is either entirely contained in one of the child bags (and hence we already computed the distances recursively), or has a special node in  $F_x$ . Hence, it is enough to store in the label of a node  $g$  its distances from nodes in  $F_x$  and its label in the child bag of  $X^*$  that contains  $g$  (if  $g$  is partitioned to several child bags,  $g$  is in  $F_x$ , and in this case we just store the distances to nodes in  $F_x$  without a recursive label). Finally, we broadcast  $\tilde{O}(D^2)$  information that includes labels of nodes in  $F_x$  (or corresponding face-parts) in the child bags, and the edges of the separator, and prove that based on this information nodes can deduce locally their distance label in  $X^*$ . This follows as each shortest path is either entirely contained in a child bag, or can be broken up to subpaths whose endpoints are in  $F_x$ , and are either entirely contained in a child bag (and hence their distance can be deduced from the labels we broadcast), or use a separator edge between different child bags (we broadcast all these edges), or use face-parts of the same face that are contained in different child bags (in this case, we connect the corresponding face-parts with a zero weight edge).

## References

- 1 Li Chen, Rasmus Kyng, Yang P. Liu, Richard Peng, Maximilian Probst Gutenberg, and Sushant Sachdeva. Maximum flow and minimum-cost flow in almost-linear time. In *63rd FOCS*, pages 612–623, 2022. doi:10.1109/FOCS54457.2022.00064.
- 2 Tijn de Vos. Minimum cost flow in the congest model. In *30th SIROCCO*, pages 406–426, 2023. doi:10.1007/978-3-031-32733-9\_18.
- 3 Michal Dory, Yuval Efron, Sagnik Mukhopadhyay, and Danupon Nanongkai. Distributed weighted min-cut in nearly-optimal time. In *53rd STOC*, pages 1144–1153, 2021. doi:10.1145/3406325.3451020.
- 4 Jinfeng Dou, Thorsten Götte, Henning Hillebrandt, Christian Scheideler, and Julian Werthmann. Brief announcement: Distributed construction of near-optimal compact routing schemes for planar graphs. In *41st PODC*, pages 67–70, 2023. doi:10.1145/3583668.3594561.
- 5 Silvio Frischknecht, Stephan Holzer, and Roger Wattenhofer. Networks cannot compute their diameter in sublinear time. In *23rd SODA*, pages 1150–1162, 2012. doi:10.1137/1.9781611973099.91.
- 6 Juan A. Garay, Shay Kutten, and David Peleg. A sublinear time distributed algorithm for minimum-weight spanning trees. *SIAM J. Comput.*, 27(1):302–316, 1998. doi:10.1137/S0097539794261118.
- 7 Mohsen Ghaffari and Bernhard Haeupler. Distributed algorithms for planar networks I: planar embedding. In *34th PODC*, pages 29–38, 2016. doi:10.1145/2933057.2933109.
- 8 Mohsen Ghaffari and Bernhard Haeupler. Distributed algorithms for planar networks II: low-congestion shortcuts, mst, and min-cut. In *27th SODA*, pages 202–219, 2016. doi:10.1137/1.9781611974331.CH16.
- 9 Mohsen Ghaffari, Andreas Karrenbauer, Fabian Kuhn, Christoph Lenzen, and Boaz Patt-Shamir. Near-optimal distributed maximum flow. In *33rd PODC*, pages 81–90, 2015.
- 10 Mohsen Ghaffari and Merav Parter. Near-Optimal Distributed DFS in Planar Graphs. In *31st DISC*, pages 21:1–21:16, 2017. doi:10.4230/LIPICS.DISC.2017.21.
- 11 Mohsen Ghaffari and Goran Zuzic. Universally-optimal distributed exact min-cut. In *40th PODC*, pages 281–291, 2022. doi:10.1145/3519270.3538429.
- 12 Stephan Holzer and Roger Wattenhofer. Optimal distributed all pairs shortest paths and applications. In *30th PODC*, pages 355–364, 2012. doi:10.1145/2332432.2332504.
- 13 Qiang-Sheng Hua, Lixiang Qian, Dongxiao Yu, Xuanhua Shi, and Hai Jin. A nearly optimal distributed algorithm for computing the weighted girth. *Sci. China Inf. Sci.*, 64(11), 2021. doi:10.1007/S11432-020-2931-X.
- 14 Jason Li and Merav Parter. Planar diameter via metric compression. In *51st STOC*, pages 152–163, 2019. doi:10.1145/3313276.3316358.
- 15 Richard J. Lipton and Robert Endre Tarjan. A separator theorem for planar graphs. *SIAM Journal on Applied Mathematics*, 36(2):177–189, 1979.
- 16 Danupon Nanongkai. Distributed approximation algorithms for weighted shortest paths. In *46th STOC*, pages 565–573, 2014. doi:10.1145/2591796.2591850.
- 17 Merav Parter. Distributed planar reachability in nearly optimal time. In *34th DISC*, volume 179, pages 38:1–38:17, 2020. doi:10.4230/LIPICS.DISC.2020.38.
- 18 David Peleg. *Distributed Computing: A Locality-Sensitive Approach*. Society for Industrial and Applied Mathematics, 2000.
- 19 David Peleg, Liam Roditty, and Elad Tal. Distributed algorithms for network diameter and girth. In *39th ICALP*, pages 660–672, 2012. doi:10.1007/978-3-642-31585-5\_58.
- 20 Atish Das Sarma, Stephan Holzer, Liah Kor, Amos Korman, Danupon Nanongkai, Gopal Pandurangan, David Peleg, and Roger Wattenhofer. Distributed verification and hardness of distributed approximation. In *43rd STOC*, pages 363–372, 2011. doi:10.1145/1993636.1993686.

## 40:8 Brief Announcement: Distributed Maximum Flow in Planar Graphs

- 21 Václav Rozhon  Christoph Grunau  Bernhard Haeupler  Goran Zuzic  Jason Li. Undirected  $(1+\epsilon)$ -shortest paths via minor-aggregates: near-optimal deterministic parallel and distributed algorithms. In *54th STOC*, pages 478–487, 2022.
- 22 Goran Zuzic  Gramoz Goranci  Mingquan Ye  Bernhard Haeupler  Xiaorui Sun. Universally-optimal distributed shortest paths and transshipment via graph-based  $l_1$ -oblivious routing. In *33rd SODA*, pages 2549–2579, 2022.
- 23 Shankar M. Venkatesan. *Algorithms for network flows*. Ph.D. thesis, Pennsylvania State University, 1983.