Local Density and Its Distributed Approximation

Aleksander Bjørn Christiansen
Technical University of Denmark, Lyngby, Denmark

Ivor van der Hoog
Technical University of Denmark, Lyngby, Denmark

Eva Rotenberg
Technical University of Denmark, Lyngby, Denmark

Abstract

The densest subgraph problem is a classic problem in combinatorial optimisation. Graphs with low maximum subgraph density are often called "uniformly sparse", leading to algorithms parameterised by this density. However, in reality, the sparsity of a graph is not necessarily uniform. This calls for a formally well-defined, fine-grained notion of density.

Danisch, Chan, and Sozio propose a definition for *local density* that assigns to each vertex v a value $\rho^*(v)$. This local density is a generalisation of the maximum subgraph density of a graph. I.e., if $\rho(G)$ is the subgraph density of a finite graph G, then $\rho(G)$ equals the maximum local density $\rho^*(v)$ over vertices v in G. They present a Frank-Wolfe-based algorithm to approximate the local density of each vertex with no theoretical (asymptotic) guarantees.

We provide an extensive study of this local density measure. Just as with (global) maximum subgraph density, we show that there is a dual relation between the local out-degrees and the minimum out-degree orientations of the graph. We introduce the definition of the local out-degree $g^*(v)$ of a vertex v, and show it to be equal to the local density $\rho^*(v)$. We consider the local out-degree to be conceptually simpler, shorter to define, and easier to compute.

Using the local out-degree we show a previously unknown fact: that existing algorithms already dynamically approximate the local density for each vertex with polylogarithmic update time. Next, we provide the first distributed algorithms that compute the local density with provable guarantees: given any ε such that $\varepsilon^{-1} \in O(\text{poly}\, n)$, we show a deterministic distributed algorithm in the LOCAL model where, after $O(\varepsilon^{-2} \log^2 n)$ rounds, every vertex v outputs a $(1+\varepsilon)$-approximation of their local density $\rho^*(v)$. In CONGEST, we show a deterministic distributed algorithm that requires $\text{poly}(\log n, \varepsilon^{-1}) \cdot 2^{O(\sqrt{\log n})}$ rounds, which is sublinear in n.

As a corollary, we obtain the first deterministic algorithm running in a sublinear number of rounds for $(1+\varepsilon)$-approximate densest subgraph detection in the CONGEST model.

2012 ACM Subject Classification Theory of computation → Graph algorithms analysis; Theory of computation → Distributed algorithms

Keywords and phrases Distributed graph algorithms, graph density computation, graph density approximation, network analysis theory

Digital Object Identifier 10.4230/LIPIcs.STACS.2025.25

Related Version *Full Version*: https://arxiv.org/abs/2411.12694

Funding This work was supported by the the VILLUM Foundation grant (VIL37507) "Efficient Recomputations for Changeful Problems", the Independent Research Fund Denmark grant 2020-2023 (9131-00044B) "Dynamic Network Analysis", and the European Union's Horizon 2020 research and innovation programme under the Marie Skłodowska-Curie grant agreement No 899987.

42nd International Symposium on Theoretical Aspects of Computer Science (STACS 2025).
Editors: Olaf Beyersdorff, Michał Pilipczuk, Elaine Pimentel, and Nguyễn Kim Thắng;
Article No. 25; pp. 25:1–25:20
Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

1 Introduction

Density or sparsity measures of graphs are widely studied and have many applications. Examples include the arboricity, the degeneracy, and the maximum subgraph density, all of which are asymptotically related within a factor of 2. Given a graph or subgraph H, its density, $\rho(H)$, is the average number of edges per vertex in H. The *maximum subgraph density* $\rho^{\max}(G)$ of a graph G is the maximum density $\rho(H)$ amongst all subgraphs $H \subseteq G$.

Computing maximum subgraph density has been studied both in the dynamic [10, 27], streaming [1, 4] and distributed [15, 28] setting. Often these measures are used to parameterise the sparsity of "uniformly sparse graphs" [2, 8, 25]. These measures are global measures in the sense that they measure the sparsity of the most dense part of the graph. In many cases the graph is not equally sparse (or dense) everywhere. Consider for example a lollipop graph: a large clique joined to a long path. The clique is a subgraph of high density, yet the vertices along the path sit in a part of the graph that is significantly less dense. Often, solutions for graph density related problems provide guarantees based on the most dense part of the graph. In some areas of computation more "local" solutions are desirable. Prior works of a global nature often completely disregard certain parts of the graphs, meaning that the output in sparser parts holds little to no information. We give three examples:

1) Many dynamic algorithms for estimating the subgraph density rely on modifying the solution locally. Algorithmic performance is expressed in terms of (global) graph sparsity, and thus fails to exploit the more fine-grained guarantee that local sparse areas yield.

2) In network analysis, one is often interested in determining dense subgraphs as these subgraphs can be interpreted, for instance, as communities within a social network. However, since many classical algorithms are tuned towards only detecting the densest subgraphs, these algorithms might fail to detect communities in sparser parts of the network [14, 26, 29].

3) Computing the maximum subgraph density is not very local, nor distributed. We consider the models LOCAL and CONGEST, and the lollipop graph. Here, almost instantly, the vertices of the clique realise they are part of a (very dense) clique. The vertices on the path may have to wait for diameter-many rounds before realising the maximum subgraph density of the graph. Distributed algorithms that wish to compute the *value* of the subgraph density are thus posed with a choice: either use $\Omega(D)$ rounds (where D is the diameter of the graph), or let every vertex output a value that is at most the maximum subgraph density.

1.1 Local density and results

We consider the definition of *local density* $\rho^*(v)$ by Danisch, Chan, and Sozio [14], defined at each node v of the graph (Definition 7). Our contributions can be split into four categories, which we present in four sections with corresponding titles.

A: Conceptual results for local density. Our primary contribution is an extensive overview of the theoretical properties of this local density measure. We show that, just as in the maximum-subgraph density problem, computing local density has a natural dual problem as we define the *local out-degree*. Consider a (fractional) orientation of the graph that is *locally fair*. i.e., for each directed edge (u, v), the out-degree $g(u)$ is at most $g(v)$. We prove that for each vertex v, the out-degree of v has the same value over all locally fair orientations. We define this value $g^*(v)$ as the local out-degree of v. We prove that the local density of each vertex is the dual of its local out-degree and thereby $g^*(v) = \rho^*(v)$. This new definition for local out-degree is considerably shorter than the definition for local density. It allows us to show some previously unknown interesting properties of local density:

B: Results for dynamic algorithms. We prove that in an *approximately fair* orientation (a definition by Chekuri et al. [10]) the out-degree $g(v)$ of each vertex is a $(1+\varepsilon)$-approximation of $g^*(v) = \rho^*(v)$ (Theorem 12). This implies a previously unknown fact: that there exist dynamic polylogarithmic algorithms [10, 13, 12] where each vertex v maintains a $(1+\varepsilon)$-approximation of its local density $\rho^*(v)$ as by Danisch, Chan, and Sozio [14].

C: Results in LOCAL. We show that each node v can obtain a $(1+\varepsilon)$-approximation of $\rho^*(v)$ by surveying its $O(\varepsilon^{-2}\log^2 n)$-hop neighbourhood. This induces a LOCAL algorithm where each vertex $v \in V$ computes a $(1+\varepsilon)$-approximation of $\rho^*(v)$ in $O(\varepsilon^{-2}\log^2 n)$ rounds.

Commentary on runtime: We observe that $\rho^{\max}(G)$ can be computed in LOCAL in $O(\varepsilon^{-1}\log n)$ rounds. In contrast, the stricter local density is computed in $O(\varepsilon^{-2}\log^2 n)$ rounds. This gap may be explained by considering the local density $\rho^*(v)$ for low-local-density vertices v in a graph that has high global density. The local density of v can be affected by a dense subgraph within a hop distance of $\Theta(\varepsilon^{-2}\log^2 n)$ (although it unclear if it can be affected *enough* to prohibit a $(1+\varepsilon)$-approximation of $\rho^*(v)$ in $o(\varepsilon^{-2}\log^2 n)$ time). The potential for a barrier of $O(\varepsilon^{-2}\log^2 n)$ rounds is also illustrated by existing dynamic algorithms [10, 28] that maintain η-fair orientations. In this scenario, these algorithms have a worst-case recourse of $\Omega(\varepsilon^{-2}\log^2 n)$. We consider it an interesting open problem to either improve our running time in LOCAL, or, show that $O(\varepsilon^{-2}\log^2 n)$ is tight.

D: Results in CONGEST. We show a significantly more involved algorithm in CONGEST, where after $O(\text{poly}\{\varepsilon^{-1}, \log n\} \cdot 2^{O(\sqrt{\log n})})$ rounds, each vertex v outputs a $(1+\varepsilon)$-approximation of $\rho^*(v)$. Since $\max_{v\in V} \rho^*(v) = \rho^{\max}(G)$, this is the first deterministic algorithm for $(1+\varepsilon)$-approximating of the global subgraph density $\rho^{\max}(G)$ in CONGEST, that runs in a number of rounds that is sublinear in the diameter of the graph.

In the main body, we focus on the value variant where we want each vertex v to output an approximation of $\rho^*(v)$. In the full version, we extend our analysis so that each vertex v can output a subgraph H with $v \in H$ where $\rho(H)$ approximates $\rho^*(v)$. See also Table 1.

Table 1 Results in LOCAL (L) or CONGEST (C) where prior work for computing the global subgraph density is compared to our running time for the local subgraph density. D denotes the diameter. Orange running times are not deterministic and occur with high probability.

Model	Problem	Each v outputs ρ_v with	Rounds		Source
L	2.1	$\rho_v \in [(1+\varepsilon)^{-1}\rho^{\max}(G), (1+\varepsilon)\rho^{\max}(G)]$	$\Theta(D)$		[28]
	2.2	$\max_v \rho_v \in [(1+\varepsilon)^{-1}\rho^{\max}(G), (1+\varepsilon)\rho^{\max}(G)]$	$O(\varepsilon^{-1}\log n)$		[28]
	3	$\boldsymbol{\rho_v \in [(1+\varepsilon)^{-1}\rho^*(v), (1+\varepsilon)\rho^*(v)]}$	$\boldsymbol{O(\varepsilon^{-2}\log^2 n)}$		**Cor. 15**
C	2.1	$\rho_v \in [(2+\varepsilon)^{-1}\rho^{\max}(G), (2+\varepsilon)\rho^{\max}(G)]$	$O(D \cdot \varepsilon^{-1}\log n)$		[15]
	2.1	$\rho_v \in [(1+\varepsilon)^{-1}\rho^{\max}(G), (1+\varepsilon)\rho^{\max}(G)]$	$O(\varepsilon^{-4}\log^4 n + D)$	whp.	[28]
	2.2	$\max_v \rho_v \in [(1+\varepsilon)^{-1}\rho^{\max}(G), (1+\varepsilon)\rho^{\max}(G)]$	$O(\varepsilon^{-4}\log^4 n)$	whp.	[28]
	2.2	$\boldsymbol{\rho_v \in [(1+\varepsilon)^{-1}\rho^*(v), (1+\varepsilon)\rho^*(v)]}$	$\boldsymbol{O(\text{poly}\{\log n, \varepsilon^{-1}\}) \cdot 2^{O(\sqrt{\log n})}}$		**Thm. 16**
	3	$\boldsymbol{\rho_v \in [(1+\varepsilon)^{-1}\rho^*(v), (1+\varepsilon)\rho^*(v)]}$	$\boldsymbol{O(\text{poly}\{\log n, \varepsilon^{-1}\}) \cdot 2^{O(\sqrt{\log n})}}$		**Thm. 16**
	3	$\boldsymbol{\rho_v \in [(1+\varepsilon)^{-1}\rho^*(v), (1+\varepsilon)\rho^*(v)]}$	$\boldsymbol{O(\text{poly}\{\log n, \varepsilon^{-1}\})}$	whp.	**Thm. 16**

2 Preliminaries and related work

Let $G = (V, E)$ be an undirected weighted graph with n vertices and m edges. For any $v \in V$ and any integer k, we denote by $H_k(v)$ the k-hop neighborhood of v. For each edge $e \in E$ we denote by $g(e)$ the *weight* of e. Any edge with endpoints u, v may be denoted as $\overline{uv}$.

We can augment any weighted graph G with a *(fractional) orientation*. An orientation $\overrightarrow{G}$ assigns to each edge $\overline{uv}$ two positive real values: $g(u \to v)$ and $g(v \to u)$ such that $g(\overline{uv}) = g(u \to v) + g(v \to u)$. These values may be interpreted as pointing a fraction of the edge $\overline{uv}$ from u to v, and the other fraction from v to u. Given an orientation $\overrightarrow{G}$, we denote by $g(u) = \sum_{v \in V} g(u \to v)$ the *out-degree* of u (i.e., how much fractional edges point outwards from u in $\overrightarrow{G}$). Given these definitions, we can consider two graph measures of G: the maximum subgraph density and the minimum orientation of G.

Global graph measures. For any subgraph $H \subseteq G$, its *density* $\rho(H)$ is defined as $\rho(H) = \frac{1}{|V(H)|} \sum_{e \in E(H)} g(e)$. The *maximum subgraph density* $\rho^{\max}(G)$ is then the maximum over all $H \subseteq G$ of $\rho(H)$. A subgraph $H \subseteq G$ is *densest* whenever $\rho(H) = \rho^{\max}(G)$. For any orientation $\overrightarrow{G}$ of G, its *maximum out-degree* $\Delta(\overrightarrow{G})$ is the maximum over all u of the out-degree $g(u)$. The *optimal out-degree* of G, denoted by $\Delta^{\min}(G)$, is subsequently the minimum over all $\overrightarrow{G}$ of $\Delta(\overrightarrow{G})$. An orientation $\overrightarrow{G}$ itself is *minimum* whenever $\Delta(\overrightarrow{G}) = \Delta^{\min}(G)$.

The density of G and the optimal out-degree are closely related. One way to illustrate this is through the following dual linear programs:

DS (Densest Subgraph)		‖	**FO (Fractional Orientation)**	
$\max \sum_{\overline{uv} \in E} g(\overline{uv}) \cdot y_{u,v}$	s.t.	‖	$\min \rho$	s.t.
$x_u, x_v \geq y_{u,v}$	$\forall \overline{uv} \in E$	‖	$g(u \to v) + g(v \to u) \geq g(\overline{uv})$	$\forall \overline{uv} \in E$
$\sum_{v \in V} x_v \leq 1$		‖	$\rho \geq \sum_{v \in V} g(u \to v)$	$\forall u \in V$
$x_v, y_{u,v} \geq 0$	$\forall u, v \in V$	‖	$g(u \to v), g(v \to u) \geq 0$	$\forall u, v \in V$

Denote by R the optimal value of DS and by Δ the optimal value of FO. By duality, $R = \Delta$. Moreover, Charikar [9] relates these two linear programs to the densest subgraph problem:

▶ **Theorem 1** (Theorem 1 in [9]). *Let G be a unit weight graph. Denote by R the optimal solution of DS and by D the optimal solution of FO. Then $\rho^{\max}(G) = R = \Delta = \Delta^{\min}(G)$.*

We show that this can be generalised to when G is a weighted graph:

▶ **Lemma 2** (See the full version). *Let G be any weighted graph. Denote by R the optimal solution of DS and by D the optimal solution of FO. Then $\rho^{\max}(G) = R = D = \Delta^{\min}(G)$.*

2.1 Densest subgraph in dynamic algorithms

In a classical, non-distributed model of computation we can immediately formalise both the value variant of the (approximate) densest subgraph:

▶ **Problem 1.** *Given a graph G and an $\varepsilon > 0$, output $\rho' \in [(1+\varepsilon)^{-1}\rho^{\max}(G), (1+\varepsilon)\rho^{\max}(G)]$.*

Alternatively, in the Fractional Orientation (FO) problem the goal is to output a $(1 + \varepsilon)$-approximation of $\Delta^{\min}(G)$. It turns out that FO is a more accessible problem to study. The LP formulations allow for a straightforward way to compute $\Delta^{\min}(G)$ and/or $\rho^{\max}(G)$. However, solving the LP requires information about the entire graph, and this information is expensive to collect. Sawlani and Wang [27] get around this difficulty by instead solving an approximate version of (FO). They work with a concept we call *local fairness*.

▶ **Definition 3.** *Let $\overrightarrow{G}$ be a fractional orientation of a graph G. We say that $\overrightarrow{G}$ is locally fair whenever $g(u \to v) > 0$ implies $g(u) \leq g(v)$.*

Chekuri et al. [10] extend this definition to η-fairness:

▶ **Definition 4.** *Let $\overrightarrow{G}$ be a fractional orientation of a graph G. We say that $\overrightarrow{G}$ is η-fair (for $\eta > 0$) whenever $g(u \to v) > 0$ implies that $g(u) \leq (1+\eta)g(v)$.*

Related work in dynamic algorithms. Chekuri et al. [10] continue to focus on computing a $(1+\varepsilon)$-approximation of the Densest Subgraph problem. They show that, if G is a unit weight graph, there exists a $(1+\varepsilon)$-approximate solution to FO that is η-fair (for some smartly chosen η). They subsequently prove that an η-fair orientation allows you to find a $(1+\varepsilon)$-approximate densest subgraph. This allows them to dynamically maintain the value of the densest subgraph of G in $O(\varepsilon^{-6} \log^3 n \log \rho^{\max}(G))$ time per insertion or deletion of edges in G. By leveraging the η-fairness of the orientation, they can report a $(1+\varepsilon)$-approximate densest subgraph in time proportional to the size of the subgraph.

2.2 Approximate densest subgraph in LOCAL and CONGEST

We focus on the *value* variant of the problem, where each vertex outputs a value (as opposed to the *reporting* variant in the full version, where the goal is to report a densest subgraph).

▶ **Problem 2.** *Given a graph G and an $\varepsilon > 0$, each vertex v outputs a value ρ_v and either:*
- ***Problem 2.1:*** *we require that $\forall v, \rho_v \in [(1+\varepsilon)^{-1}\rho^{\max}(G), (1+\varepsilon)\rho^{\max}(G)]$, or*
- ***Problem 2.2:*** *we require that $\max_v \rho_v \in [(1+\varepsilon)^{-1}\rho^{\max}(G), (1+\varepsilon)\rho^{\max}(G)]$.*

Related work. Problem 2.1 has a trivial $\Omega(D)$ lower bound, obtained by constructing a lollipop graph (where D denotes the diameter). In LOCAL, it is trivial to solve Problem 2.1 in $\Theta(D)$ time. Problem 2.2 was studied by Ghaffari and Su [20] who present a randomised $(1+\varepsilon)$-approximation in LOCAL that uses $O(\varepsilon^{-3} \log^4 n)$ rounds. Fischer et al. [16] present a deterministic $(1+\varepsilon)$-approximation in LOCAL that uses $2^{O(\log^2(\varepsilon^{-1} \log n))}$ rounds. Ghaffari et al. [19] improve this to $O(\varepsilon^{-9} \log^{15} n)$ rounds. The work by Harris [24] improves this to $\tilde{O}(\varepsilon^{-6} \log^6 n)$ rounds. Su and Vu [28] present the state-of-the-art in this area. They prove that for any graph G, there exists a vertex v such that for the k-hop neighbourhood $H_k(v)$ (with $k \in O(\varepsilon^{-1} \log n)$) the density $\rho^{\max}(H_k)$ is a $(1+\varepsilon)$-approximation of $\rho^{\max}(G)$. This immediately leads to a trivial LOCAL algorithm: each vertex u collects its k-hop neighbourhood $H_k(u)$ in $O(\varepsilon^{-1} \log n)$ rounds, solves the LP of Densest Subgraph in its own node, and reports the value $\rho^{\max}(H_k(u))$.

In CONGEST, the state-of-the-art deterministic algorithm for Problem 2.1 and 2.2 is by Das Sarma et al. [15] who present a $(2+\varepsilon)$-approximation in $O(D \cdot \varepsilon^{-1} \log n)$ rounds. The best randomised work is by Su and Vu [28] who present a randomised algorithm for Problem 2.2 that runs in $O(\varepsilon^{-4} \log^4 n)$ rounds w.h.p. See also Table 1 for an overview.

2.3 Local density

Danisch, Chan, and Sozio [14] introduce a more local measure which they call the *local density*. Its lengthy definition assigns to each vertex v a value. We note for the reader that we almost immediately define our local out-degree (Definition 8), and only use local out-degree in proofs. Hence, the reader is not required to have a thorough understanding of the following:

▶ **Definition 5** (Definition 2.2 in [14]). *Let $G = (V, E)$ be a weighted graph where an edge e has weight $g(e)$. Let $B \subseteq V$. For any $X \subseteq V - B$, we define the* quotient edges $\hat{E}_B(X)$ *as all edges in G with one endpoint in X, and the other endpoint in X or B. We define:*

- *for $X \subseteq V - B$, the* quotient subgraph density $\hat{\rho}_B(X) := \frac{1}{|X|} \sum_{e \in \hat{E}_B(X)} g(e)$.
- *the* maximum quotient density $\hat{\rho}_B(G) := \max_{X \subseteq V - B} \hat{\rho}_B(X)$.

▶ **Definition 6** (Definition 2.3 in [14]). *Given a weighted undirected graph $G = (V, E)$, we define the diminishing-dense decomposition $\mathcal{B}$ of G as the sequence $B_0 \subset B_1 \ldots \subset B_\ell = V$:*
We define $B_0 = \emptyset$. For $i \geq 1$ if $B_{i-1} = V$ then $\ell := i$. Otherwise:

$$S_i := arg \max_{X \subseteq V - B_{i-1}} \hat{\rho}_{B_{i-1}}(X), \text{ and } B_i := B_{i-1} \cup S_i.$$

▶ **Definition 7** (Definition 2.3 in [14]). *Given a weighted undirected graph $G = (V, E)$ and a diminishing-dense decomposition $\mathcal{B}$, each vertex $v \in V$ has one integer i where $v \in S_i$. We define the* local density $\rho^*(v) := \hat{\rho}_{B_{i-1}}(S_i)$.

The benefit of local measures. Problem 2's variants have drawbacks in a distributed model of computation. Problem 2.1 has an $\Omega(d)$ lower bound (making it trivial in LOCAL). Problem 2.2 allows some vertices to output nonsense. The definition of local density alleviates these issues, as we may define an algorithmic problem which we consider to be more natural:

▶ **Problem 3.** *Given (G, ε), each vertex v outputs $\rho_v \in [(1+\varepsilon)^{-1}\rho^*(v), (1+\varepsilon)\rho^*(v)]$.*

Related work. Danisch, Chan, and Sozio [14] introduce the local density (Definition 7). Intuitively, they define a partition $V_1 \ldots V_k$ of V. For each i, they consider the set $X_i = \bigcup_{j=1}^{i} V_j$. They then define the graph H_i as the vertex-induced subgraph of X_i where for each edge in G between a vertex in X_i and a vertex not in X_i, they add a self loop. They define the *quotient density* of V_i as the density of H_i. The local density $\rho^*(v)$ of a vertex v is the quotient density of V_i for $v \in V_i$. Danisch, Chan, and Sozio [14] then define a quadratic program $\mathbf{FO}^2$. The domain of this program is the space of all orientations of the graph G. The cost function is the sum over all vertices u, of the out-degree squared. Consider an orientation $\overrightarrow{G}$ of G that optimises the cost function. Danisch, Chan, and Sozio show that for each vertex v it out-degree $g(v)$ equals the local density $\rho^*(v)$. As a consequence, over all optimal solutions to $\mathbf{FO}^2$, the out-degree of each vertex is unique. They provide a Frank-Wolfe based algorithm to solve the quadratic program with no theoretical guarantees.

Chekuri, Harb, and Quanrud [22] study computing the local density by solving the quadratic program. They show [22, Theorem 3.4] some interesting properties of $\mathbf{FO}^2$. Specifically, they show that the uniqueness property by Danish, Chan and Sozio is a special case of Fujishige's result [18] from 1980. They additionally show that the algorithm GREEDY++ by Boob, Gao, Peng, Sawlani, Tsourakakis, Wang, and Wang [5] (when applied to this quadratic program) converges to an orientation where the out-degree of each vertex v is a $(1+\varepsilon)$-approximation of $\rho^*(v)$. Chekuri, Harb, and Quanrud [23] subsequently show that the more general SUPER-GREEDY++ algorithm by Chekuri, Quanrud and Torres [11] also converges to $(1+\varepsilon)$-approximation of each $\rho^*(v)$.

Borradaile, Migler, and Wilfong [7] observe that there is a correlation between the fairness of an orientation and local density. An *integral orientation* is any orientation of the graph where for each edge (u, v), $g(u \to v) \in \{0, 1\}$. They consider an *egalitarian orientation* which is defined as an integral orientation where "the total available out-degree is shared among the vertices as equally as allowed by the topology of the graph". An egalitarian orientation, and other equivalent notions of "fair integral orientations", are formally defined through an integer flow problem on the graph [6, 17, 30]. From section 2 in [6] it follows that an integral orientation is egalitarian if and only if $g(u \to v) = 1$ implies that $g(u) \leq g(v) + 1$.

Using an egalitarian orientation, they create a decomposition of the graph that they call a *density decomposition*. The density decomposition by Borradaile, Migler, and Wilfong [7] is not equal to the local density. The analysis of Kopelowitz, Krauthgamer, Porat, and Solomon shows that if $\overrightarrow{G}$ is an egalitarian orientation then $\max_v g(v) \leq \rho(G) + O(\log n)$. Their decomposition, compared to the one used for the local density, can thus be viewed as one that is coarser but similar in spirit. It can thereby be argued that Borradaile, Migler, and Wilfong [7] are the first to show a connection between the local density and fair orientations of a graph. However, we note that the integrality of their orientation prevents exact (or $(1 + \varepsilon)$-approximate) computations of the local density.

3 Results and organisation

Now we are ready to formally state our contributions. Our primary contribution is that we show a dual definition to local density, which we call the local out-degree:

▶ **Definition 8.** *Given a graph $G = (V, E)$, we define the* local out-degree *as:*

$g^*(u) :=$ the out-degree $g(u)$ in any locally fair fractional orientation of G.

It is not immediately clear that the local out-degree is well-defined. We prove (Theorem 9) that each vertex in G has the same out-degree across all locally fair orientations of G (and thus, the set of all locally fair orientations of G assigns to each vertex a real value). We believe that the local out-degree is conceptually simpler that the local density. Through this definition, we are able to show various algorithms to approximate the local density.

3.A Conceptual results for local density

We prove in Section 4 that these local definitions generalise the global definition of subgraph density and out-degree, as they exhibit the same dual behaviour. We show several previously unknown properties of the local density, which we consider to be of independent interest:

▶ **Theorem 9.** *For any weighted graph G, $\forall v \in V$, $g^*(v)$ is well-defined and equals $\rho^*(v)$.*

▶ **Corollary 10.** *Given a weighted graph G, $\rho^{\max}(G) = \Delta^{\min}(G) = \max_v g^*(v)$.*

▶ **Corollary 11.** *For any graph G, there exists a fractional orientation $\overrightarrow{G}$ that is locally fair.*

3.B Results for dynamic algorithms

We show in Section 5 that η-fair orientations imply approximations for our local measures:

▶ **Theorem 12.** *Let G be a weighted graph and $\overrightarrow{G}$ be an η-fair fractional orientation for $\eta \leq \frac{\varepsilon^2}{128 \cdot \log n}$. Then $\forall v \in V$: $(1 + \varepsilon)^{-1} \rho^*(v) \leq g(v) \leq (1 + \varepsilon)\rho^*(v)$.*

This immediately implies the following Corollary by applying [10]:

▶ **Corollary 13.** *There exists an algorithm [10] that can fractionally orient a dynamic unit-weight graph G with n vertices subject to edge insertions and deletions with deterministic worst-case $O(\varepsilon^{-6} \log^4 n))$ update time such that for all $v \in V$:*

$g(v) \in [(1 + \varepsilon)^{-1} \rho^*(v), (1 + \varepsilon)\rho^*(v)]$.

3.C Results in LOCAL

The local density as a measure is not entirely local. However, we prove in Section 6 that far-away subgraphs affect the local density of a vertex v only marginally:

▶ **Theorem 14.** *Let G be a unit-weight graph. For any $\varepsilon > 0$ and vertex v, denote by $\rho^*(v)$ its local density and by $\rho^*_k(v)$ its local density in $H_k(v)$. Then $\rho^*_k(v) \in [(1+\varepsilon)^{-1}\rho^*(v), (1+\varepsilon)\rho^*(v)]$ for $k \in \Theta(\varepsilon^{-2}\log^2 n)$.*

This immediately implies a trivial algorithm for problem 2 in LOCAL (where each vertex v collects its k-hop neighbourhood $H_k(v)$ for $k \in \Theta(\varepsilon^{-2}\log^2 n)$ and then solves FO on $H_k(v)$):

▶ **Corollary 15.** *There exists an algorithm in LOCAL that given a unit graph G and $\varepsilon > 0$ computes in $O(\varepsilon^{-2}\log^2 n)$ rounds for all $v \in V$ a value $\rho_v \in [(1+\varepsilon)^{-1}\rho^*(v), (1+\varepsilon)\rho^*(v)]$.*

3.D Results in CONGEST

Finally in Section 7, we solve Problem 3 in CONGEST by computing an η-fair orientation. We use as a subroutine algorithm to compute *blocking flows* in an h-layered DAG [21]:

▶ **Theorem 16.** *Suppose one can compute a blocking flow in an n-node h-layered DAG in $\mathrm{Blocking}(h, n)$ rounds. There exists an algorithm in CONGEST that given a unit-weight graph G and $\varepsilon > 0$ computes in $O(\varepsilon^{-3}\log^4 n \cdot (\varepsilon^{-2}\log^2 n + \mathrm{Blocking}(\varepsilon^{-2}\log^2 n, n)))$ rounds an orientation $\overrightarrow{G}$ such that for all $v \in V$: $\mathrm{g}(v) \in [(1+\varepsilon)^{-1}\rho^*(v)), (1+\varepsilon)\rho^*(v)\rho^*(v)]$.*

As a corollary, we obtain the first deterministic algorithm running in a sublinear number of rounds for $(1+\varepsilon)$-approximate dense subgraph detection in the CONGEST model (Table 1).

4 Conceptual results for local density

Our primary contribution is the definition of local out-degree as a dual to local density.

▶ **Lemma 17.** *The local density is well-defined. That is, for any two locally fair orientations $\overrightarrow{G}$ or $\overrightarrow{G}'$ where a vertex u has out-degree $\mathrm{g}(u)$ or $\mathrm{g}'(u)$ respectively, $\mathrm{g}(u) = \mathrm{g}'(u)$.*

Proof. Suppose for the sake of contradiction that there exists two locally fair orientations $(\overrightarrow{G}, \overrightarrow{G}')$ and a vertex $u \in V$ where $\mathrm{g}(u) > \mathrm{g}'(u)$. We define their *symmetric difference* graph S as a digraph where the vertices are V and there exists an edge $\overrightarrow{ab}$ whenever $\mathrm{g}(a \to b) > \mathrm{g}'(a \to b)$. We may assume that S contains no directed cycles:

Indeed if S contains any directed cycle π we change $\overrightarrow{G'}$, where for all $\overrightarrow{ab} \in \pi$ we slightly increase $\mathrm{g}'(a \to b)$ until S loses an edge. This operation does not change the out-degree of any vertex in $\overrightarrow{G}'$. So, we still have two locally fair orientations $(\overrightarrow{G}, \overrightarrow{G}')$ with $\mathrm{g}(u) > \mathrm{g}'(u)$.

Since $\mathrm{g}(u) > \mathrm{g}'(u)$, the vertex u must have at least one out-edge in S and since S has no cycles, it follows that u must have some directed path π_v to a sink v in S. Since v is a sink in the symmetric difference graph it follows that $\mathrm{g}(v) < \mathrm{g}'(v)$.
However we now observe the following property of the path π_v:

- $\forall \overrightarrow{ab} \in \pi_v$, $\mathrm{g}(a \to b) > \mathrm{g}'(a \to b)$. Thus, $\mathrm{g}(a \to b) > 0$ and so there exists a directed path from u to v in $\overrightarrow{G}$. Since $\overrightarrow{G}$ is locally fair this implies that $\mathrm{g}(u) \leq \mathrm{g}(v)$.
- $\forall \overrightarrow{ab} \in \pi_v$, $\mathrm{g}(a \to b) > \mathrm{g}'(a \to b)$. Thus, $\mathrm{g}'(a \to b) < \mathrm{g}(ab)$ and so $\mathrm{g}'(b \to a) > 0$. It follows that there exists a directed path from v to u in $\overrightarrow{G}'$. Local fairness implies $\mathrm{g}'(v) \leq \mathrm{g}'(u)$.

The 4 equations: $\mathrm{g}(u) > \mathrm{g}'(u)$, $\mathrm{g}(v) < \mathrm{g}'(v)$, $\mathrm{g}(u) \leq \mathrm{g}(v)$, and $\mathrm{g}'(u) \geq \mathrm{g}'(v)$ give a contradiction. ◀

Lemma 17 would imply that the local out-degree is well-defined, *if* the set of locally fair orientations is non-empty. Bera, Bhattacharya, Choudhari and Ghosh already aim to prove this in Section 4.1 of [3] (right above Equation 9). They claim that a locally fair orientation always exist by the following argument: They consider an arbitrary orientation $\overrightarrow{G}$ that is not locally fair. They claim that for any pair (u, v) where $g(u) > g(v)$ and $g(u \to v) > 0$ it is possible to transfer some out-degree from $g(u)$ to $g(v)$. The existence of a locally fair orientation would follow, if it can be shown that this procedure converges to a locally fair orientation. Indeed, since the space of all orientations is a compact polytope, the limit of a converging sequence over this domain must lie within the space.

It is intuitive but not clear that this procedure indeed converges. Indeed, decreasing $g(u)$ and increasing $g(v)$ may cause some other edge (w, u) or (v, w) to start violating local fairness. One way to show convergence is to define a potential function and to show that such a transfer always decreases the potential. We define the potential function $\sum_{v \in V} g^2(v)$, thereby creating a quadratic program where the domain is the space of all orientations of G. We prove that any optimal solution to this quadratic program must be a locally fair orientation. Any quadratic function over a compact domain has an optimum and so the existence of a locally fair orientation follows.

▶ **Theorem 9.** *For any weighted graph G, $\forall v \in V$, $g^*(v)$ is well-defined and equals $\rho^*(v)$.*

Proof. We consider the following quadratic program $\mathbf{FO}^2$ from [14, Section 4] where we compute a fractional orientation of the graph G subject to a quadratic cost function:

$$\begin{aligned} \min \sum g(u)^2 \quad & \text{s.t.} \\ g(u \to v) + g(v \to u) \geq g(\overline{uv}) \quad & \forall \overline{uv} \in E \\ g(u) \geq \sum_{v \in V} g(u \to v) \quad & \forall u \in V \\ g(u \to v), g(v \to u) \geq 0 \quad & \forall u, v \in V \end{aligned}$$

Consider any optimal solution to the quadratic program. It must be that $g(u) = \sum_{v \in V} g(u \to v)$. Danisch, Chan, and Sozio [14, Corollary 4.4] prove that for any vertex u, the local density $\rho^*(u) = g(u)$.

We first note that any solution to the quadratic program is an orientation. Indeed, suppose for the sake of contradiction that there exists an edge $\overline{uv} \in E$ where $g(u \to v) + g(v \to u) > g(\overline{uv})$. We may now decrease either $g(u \to v)$ or $g(v \to u)$ to obtain another viable solution to the program. Consider decreasing $g(u \to v)$, then we may decrease $g(u)$ and maintain a viable and better solution to the program – a contradiction.

Secondly, we claim that the optimal solution to the quadratic program is a locally fair orientation. Suppose for the sake of contradiction that there exist u, v with $g(u) = g(v) + \delta'$ and $g(u \to v) = \delta$ for $\delta, \delta' > 0$. We can decrease $g(u \to v)$ to zero by increasing $g(v \to u)$ by $\Delta = \min\{\delta, \delta'\}$ and still maintain a solution to the program. This reduces the solution's value by $(g(u) - \Delta)^2 - (g(v) + \Delta)^2$. However, we now found a solution to the quadratic program with a lower value than the optimal solution – a contradiction.

Thus, the solution to $\mathbf{FO}^2$ gives a locally fair orientation $\overrightarrow{G}$ where each vertex u has out-degree $g(u)$. The local density $g^*(u) = g(u)$ is by Lemma 17 well-defined. Danisch, Chan and Sozio [14, Corollary 4.4] show that $\rho^*(u) = g(u)$, which proves the theorem. ◀

Since the local density equals the local out-degree, we conclude from [14] that:

▶ **Corollary 10.** *Given a weighted graph G, $\rho^{\max}(G) = \Delta^{\min}(G) = \max_v g^*(v)$.*

Since a quadratic program over a convex domain always has a solution, we may also note the following interesting fact:

▶ **Corollary 11.** *For any graph G, there exists a fractional orientation $\overrightarrow{G}$ that is locally fair.*

5 Results for dynamic algorithms

We use our definition of local out-degree to show that there already exist dynamic algorithms that approximate the local density of each vertex. Recall that an orientation $\overrightarrow{G}$ is η-fair whenever for all $\overline{uv} \in E(\overrightarrow{G})$, $g(u \to v) > 0$ implies that $g(u) \leq (1+\eta)g(v)$. We show that if we choose $\eta \leq \frac{\varepsilon^2}{128 \cdot \log n}$, then for any η-fair orientation, for all v, the out-degree $g(v)$ is a $(1+\varepsilon)$ approximation of $g^*(v) = \rho^*(v)$. Moreover, we prove that the maximal local out-degree (i.e. $\max_{u \in V} g(u)$) is a $(1+\varepsilon)$ approximation of $\Delta^{\min}(G) = \rho^{\max}(G)$; illustrating that approximating the local measures is a strictly more general problem. To this end, we prove the following helper lemma:

▶ **Lemma 18.** *Let $\eta \leq \frac{\varepsilon^2}{128 \cdot \log n}$ and $k \leq \log_{(1+\frac{1}{16}\varepsilon)} n$. Then $(1+\eta)^{-k} \geq (1+0.5\varepsilon)^{-1}$.*

Proof. Using $\log(1+x) \geq x/2$ whenever $x < 1$, we obtain:

$$-\log_{1+\frac{\varepsilon}{16}}(n) = -\frac{\log(n)}{\log 1 + \frac{\varepsilon}{16}} \geq -\frac{\log(n)}{\frac{\varepsilon}{32}} \geq -\frac{\varepsilon}{4} \cdot \frac{128\log(n)}{\varepsilon^2}$$

$$(1+\eta)^{-k} \geq (1 + \frac{\varepsilon^2}{128 \cdot c \cdot \log n})^{-c \cdot \log_{1+\frac{\varepsilon}{16}}(n)} \geq (1 + \frac{\varepsilon^2}{128 \cdot c \log n})^{-\frac{\varepsilon}{4} \cdot \frac{128 \cdot c \cdot \log(n)}{\varepsilon^2}}$$

$$(1+\eta)^{-k} \geq EXP\left[-\frac{\varepsilon}{4}\right] \geq EXP\left[-\log(1+\frac{\varepsilon}{2})\right] \geq (1+\frac{\varepsilon}{2})^{-1}$$ ◀

▶ **Theorem 12.** *Let G be a weighted graph and $\overrightarrow{G}$ be an η-fair fractional orientation for $\eta \leq \frac{\varepsilon^2}{128 \cdot \log n}$. Then $\forall v \in V$: $(1+\varepsilon)^{-1}\rho^*(v) \leq g(v) \leq (1+\varepsilon)\rho^*(v)$.*

Proof. **First, we show that for all vertices v, $g(v) \leq (1+\varepsilon)\rho^*(v)$.**

Suppose for the sake of contradiction that there exists a vertex u with $g(u) > (1+\varepsilon)\rho^*(u)$. We fix $\rho^*(u) = g^*(u) = \gamma$ and work with γ throughout the remainder of this proof to show a contradiction. By Corollary 11, there exists at least one locally fair fractional orientation $\overrightarrow{G}_x$. By Corollary 10, every vertex v in this orientation has out-degree $g_x(v) = g^*(v) = \rho^*(v)$. And thus, the fractional orientation $\overrightarrow{G}_x$ is not equal to $\overrightarrow{G}$.

Given $\overrightarrow{G}$ and a locally fair fractional orientation $\overrightarrow{G}_x$, we do three steps:

- We partition the vertices of G to create two graphs G_1 and G_2. The partition is based on the orientation $\overrightarrow{G}_x$ as we set: $G^1 = G[v \in V \mid g_x(v) \leq \gamma]$ and by $G^2 = G[v \in V \mid g_x(v) > \gamma]$. For ease of notation, we write any edge with one endpoint $a \in V(G^1)$ and one endpoint $b \in V(G^2)$ as (a, b) and never as (b, a).
- From G^1, we create a family of nested subgraphs using $\overrightarrow{G}$. We define graphs $G^1_i := G^1[v \in V(G^1) \mid g(v) \geq \frac{g(u)}{(1+\eta)^i}]$. We denote by k the lowest integer such that $|V(G^1_{k+1})| < (1+\frac{\varepsilon}{16})|V(G^1_k)|$. We apply Lemma 18 to observe that $(1+\eta)^{-k} \geq (1+\varepsilon/2)^{-1}$.
- Finally, we use both orientations to create three claims that contradict one another.

The first claim. We denote by $E_\uparrow$ the set of all edges $e = (a, b)$ with $a \in V(G^1_{k+1})$ and $b \in V(G^2)$ and claim that:

$$\sum_{e \in E(G^1_{k+1})} g(e) + \sum_{e \in E_\uparrow} g(e) \leq \sum_{v \in V(G^1_{k+1})} g_x(v). \tag{1}$$

Indeed for $\overline{ab} \in E(G^1_{k+1})$, both endpoints are in $V(G_{k+1})$. Because $\overrightarrow{G}_x$ is locally fair, this implies that $g_x(a \to b) + g_x(b \to a) = g(\overline{ab})$. For all $\overline{ab} \in E_\uparrow$, $g_x(b) > \gamma \geq g_x(a)$ per definition of G^1 and G^2. By local fairness, $g_x(a \to b) = g(\overline{ab})$ and the inequality follows.
The second claim. We secondly claim that:

$$\sum_{v \in V(G^1_k)} g(v) > \sum_{v \in V(G^1_{k+1})} g_x(v). \tag{2}$$

This is because we can lower bound $\sum_{v \in V(G^1_k)} g(v)$ as follows:

$$\sum_{v \in V(G^1_k)} g(v) \geq (1+\eta)^{-k} \cdot g(u) \cdot |V(G^1_k)| > (1+\eta)^{-k} \cdot g(u) \cdot |V(G^1_{k+1})| \cdot (1+\frac{\varepsilon}{16})^{-1}$$
$$> (1+\frac{\varepsilon}{2})^{-1} \cdot (1+\varepsilon)\gamma \cdot |V(G^1_{k+1})| \cdot (1+\frac{\varepsilon}{16})^{-1} \geq \gamma \cdot |V(G^1_{k+1})|$$

The claim follows by noting that per definition of G^1, for all $v \in V(G^1_{k+1})$, $g_x(v) \leq \gamma$.
The third claim. Lastly, we claim that:

$$\sum_{v \in V(G^1_k)} g(v) \leq \sum_{e \in E(G^1_{k+1})} g(e) + \sum_{e \in E_\uparrow} g(e) \tag{3}$$

Consider any $v \in V(G^1_k)$ and any vertex a with $g(v \to a) > 0$. Recall that $\overrightarrow{G}$ is an η-fair orientation. Thus, if $a \in G^1$, then $\overline{va} \in E(G^1_{k+1})$. If $a \in G^2$, then per definition $\overline{va} \in E_\uparrow$. Per definition of a fractional orientation $g(v \to a) \leq g(\overline{va})$ and so the claim follows.
A contradiction. Equation 1, 2 and 3 contradict each other. Thus, we have proven that for all vertices v, $g(v) \leq (1+\varepsilon)\rho^*(v)$.

The full version finishes the proof by showing the other direction. ◀

If G is a unit-weight graph, Chekuri et al. [10] present a dynamic algorithm to maintain an η-fair orientation in a unit-weight graph with $\eta \in O(\varepsilon^{-2} \log n)$. Thus, by Theorem 12, we may now conclude that they approximate the local density (and/or the local out-degree):

▶ **Corollary 13.** *There exists an algorithm [10] that can fractionally orient a dynamic unit-weight graph G with n vertices subject to edge insertions and deletions with deterministic worst-case $O(\varepsilon^{-6} \log^4 n))$ update time such that for all $v \in V$:*

$$g(v) \in [(1+\varepsilon)^{-1}\rho^*(v), (1+\varepsilon)\rho^*(v)].$$

6 Results in LOCAL

We prove that the local out-degree of each $v \in V$ is (largely) determined by its local neighbourhood. As a result, we immediately get an algorithm to solve Problem 3 in LOCAL.

▶ **Theorem 14.** *Let G be a unit-weight graph. For any $\varepsilon > 0$ and vertex v, denote by $\rho^*(v)$ its local density and by $\rho^*_k(v)$ its local density in $H_k(v)$. Then $\rho^*_k(v) \in [(1+\varepsilon)^{-1}\rho^*(v), (1+\varepsilon)\rho^*(v)]$ for $k \in \Theta(\varepsilon^{-2} \log^2 n)$.*

Proof. To prove the theorem, we design a simple deletion-only algorithm to maintain an η-fair orientation. For $\eta = \frac{\varepsilon^2}{128 \log n}$, this algorithm has a recursive depth of $O(\varepsilon^{-2} \log^2 n)$.

Algorithm 1 DECREASE($g(u \to v)$ by Δ – assuming that $\Delta \leq g(u \to v)$).

```
w* ← arg max_{w∈V} {g(w) | g(w→u) > 0}
while Δ > 0 and g(w*) > (1+η)(g(u) − Δ) do
  if Δ > g(w*→u) then
    Decrease(g(w*→v) by g(w*→v))
    Δ = Δ − g(w*→u)
  else
    Decrease(g(w*→v) by Δ)
  w* ← arg max_{w∈V} {g(w) | g(w→u) > 0}
```

Specifically, we say that a directed edge $\overline{uv}$ is bad whenever $g(u \to v) > 0$ and $g(u) > (1+\eta)g(v)$. In an η-fair orientation no edge is bad. Whenever we delete an edge $\overline{x_1 x_0}$, the out-degree $g(x_1)$ decreases by $g(x_1 \to x_0)$. For vertices x_2 with $g(x_2 \to x_1) > 0$, it may now be that $g(x_2) > (1+\eta)g(x_1)$ (and thus the edge $\overline{x_2 x_1}$ becomes bad). Note if there exists such a vertex x_2, then it must hold for the vertex $x_2^* := \arg\max_{x_2 \in V}\{g(x_2) \mid g(x_2 \to x_1) > 0\}$. This leads to a recursive algorithm to decrease the out-degree of a vertex (Algorithm 1).

We claim that this algorithm as a recursive depth of $O(\log_{1+\eta} n)$. Indeed any sequence of recursive calls is a path in G. Denote the path belonging to the longest sequence of recursive calls by $x_0, x_1, \dots x_\ell$. Since Δ is always at most 1, it must hold for all i that: $g(x_i) > (1+\eta)(g(x_{i-1}) - 1))$. Since a graph may have at most n^2 edges, $g(x_\ell) \leq n$ and it follows that the recursive depth is at most $\ell \in O(\log_{1+\eta} n)$. We now apply $\log(1+x) \geq x/2$ and note that: $\ell \leq \frac{\log n}{\log(1+\eta)} \leq \frac{\log n}{\eta/2} \subseteq O(\frac{\log n}{64\varepsilon^2/\log n}) \subseteq O(\frac{\log^2 n}{\varepsilon^2})$.

Given this theoretical algorithm, we prove the lemma. Consider any fair orientation $\overrightarrow{G}$. Then by Theorem 9 for any vertex v, $g(v) = \rho^*(v)$. Choose some $k \in \Theta(\varepsilon^{-2}\log^2 n)$ sufficiently large and let $H_k(v)$ be the k-hop neighbourhood of v and E_k be all the edges in $H_{k+1}(v)$ that are not in H_k.

Choose $\eta = \frac{\varepsilon^2}{128 \log n}$. The orientation $\overrightarrow{G}$ is per definition an η-fair orientation. We run on $\overrightarrow{G}$ our deletion-only algorithm, deleting all edges in E_k. Since our algorithm has a recursive depth of $\ell < k$, we end up with an η-fair orientation of $H_k(v)$ where $g(v) = \rho^*(v)$. We apply Theorem 12 to conclude that $\rho^*(v) = g(v) \in [(1+\varepsilon)^{-1}\rho_k^*(v), (1+\varepsilon)^{-1}\rho_k^*(v)]$ which concludes the theorem. ◀

▶ **Corollary 15.** *There exists an algorithm in LOCAL that given a unit graph G and $\varepsilon > 0$ computes in $O(\varepsilon^{-2}\log^2 n)$ rounds for all $v \in V$ a value $\rho_v \in [(1+\varepsilon)^{-1}\rho^*(v), (1+\varepsilon)\rho^*(v)]$.*

7 Results in CONGEST

We now describe an algorithm in CONGEST that for any unit-weight graph G, creates an η-fair orientation (with $\eta = \frac{\varepsilon^2}{128 \cdot \log n}$). Our algorithm uses as a subroutine a distributed algorithm to compute a *blocking flow* in an h-layered DAG in $O(\text{Blocking}(h, n))$ rounds.

▶ **Definition 19.** *An edge-capacitated DAG G is h-layered if the vertices can be embedded on a grid of height h, such that every directed edge $\overline{uv}$ points downwards.*

▶ **Definition 20.** *For an edge-capacitated DAG G with sources S and terminals T, a flow f from S to T is* blocking *if every augmenting path of f contains at least one saturated edge.*

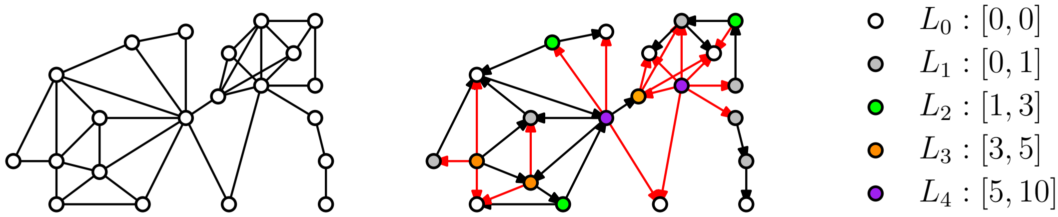

Figure 1 Given a graph G, we arbitrarily orient G. This allows us to partition the vertices of G into *levels* $L_1, \ldots L_6$ based on their current out-degree. We say that an edge (u, v) is *violating* whenever $g(u \to v) > 0$, $u \in L_i$, $v \in L_j$ and $i > j + 1$. We show violating edges in red. Our algorithm iterates over an integer h from high to low, and tries to flip all violating edges from level L_h.

▶ **Lemma 21** (Lemma 7.2 and 9.1 in [21]). *There exists an algorithm which, given an n-node h-layer edge-capacitated DAG D with sources S and terminals T computes a blocking ST-flow in CONGEST in:*

- $\text{Blocking}(h, n) = \tilde{O}(h^4)$ *rounds with high probability,*
- $\text{Blocking}(h, n) = \tilde{O}(h^6 \cdot 2^{c\sqrt{\log n}})$ *deterministic rounds for some constant c.*

We compute an η-fair orientation by repeatedly constructing a DAG with $h \in O(\varepsilon^{-2} \log^2 n)$ and computing blocking flows. Theorem 12 implies in an η-fair orientation (for our choice of η) the out-degree of each vertex v is a $(1 + \varepsilon)$-approximation $\rho^*(v)$.

The initialising step. Before we start our algorithm, we create a starting orientation where we set for every edge $e = \overline{uv}$, $g(u \to v) = \frac{1}{2} g(e)$ and $g(v \to u) = \frac{1}{2} g(e)$. This gives each vertex u *some* out-degree $g(u)$ which we partition:

▶ **Definition 22.** *Let each vertex u have out-degree $g(u)$. We define* level i *as:*

$$L_i := \left\{ u \in V \mid g(u) \in \left[(1 + \frac{\eta}{2})^i, (1 + \frac{\eta}{2})^{i+1} \right] \right\}.$$

A vertex $u \in L_i$ is at level i *and ℓ' denotes the highest level that is not empty.*
Whenever $g(u) = (1 + \frac{\eta}{2})^i$, u may decide whether $u \in L_i$ or $u \in L_{i-1}$; whenever our algorithm increases $g(u)$ the vertex u defaults to the lowest possible level and vice versa.

▶ **Definition 23.** *Consider an edge $\overline{uv}$ with $u \in L_i$ and $v \in L_j$. We say that:*

- (u, v) *is an* out-edge *from u and an* in-edge *into v whenever $i > j$ and $g(u \to v) > 0$, and*
- (u, v) *is* violating *whenever $i > j + 1$ and $g(u \to v) > 0$*

Note that the orientation is η-fair whenever there exist no violating edges.

▶ **Lemma 24.** *Let $\eta \leq \frac{\varepsilon^2}{128 \cdot \log n}$. For our orientation, let ℓ' be the highest level such that $L_{\ell'}$ is not empty then $\ell' \leq \varepsilon^{-2} \log^2 n$.*

Proof. The maximal out-degree of a vertex is n. Thus, $\ell' \leq \frac{\log n}{\log(1 + \frac{\eta}{2})}$. We now apply $\log(1 + x) \geq x/2$ and note that: $\ell' \leq \frac{\log n}{\log(1 + \frac{\eta}{2})} \leq \frac{\log n}{\eta / 4} = \frac{\log n}{32 \varepsilon^2 / \log n} \leq \frac{\log^2 n}{\varepsilon^2}$. ◀

7.1 Algorithm overview

Denote $\ell = \lceil \varepsilon^{-2} \log^2 n \rceil$. Our algorithm runs on a "clock" denoted by $(h : m : s)$ where:

- Each HOUR h lasts $2\lceil \eta^{-1} \rceil + 2$ MINUTES,
- Each even MINUTE m lasts $4\lceil \log_{8/7} n \rceil + 2$ rounds,

- Each odd MINUTE m in HOUR h lasts $\ell - h + 1$ SECONDS,
- Each SECOND s lasts $\ell + \text{Blocking}(\ell, n)$ rounds.

Each vertex tracks the clock to know which actions of our algorithm it should execute (if any). Our clock is special, in the sense that hours tick downwards. Minutes and seconds tick upwards, starting from zero. Each vertex v in our graph keeps track of the current time, measured in the current HOUR h, MINUTE m and SECOND s. We maintain the following:

▶ **Invariant 1.** *During HOUR h, there are no violating out-edges from level k for all $k > h+1$. At the start of each even MINUTE in HOUR h, there are no violating out-edges from level k for all $k > h$.*

This invariant implies that we compute an η-fair orientation when the clock reaches $(0:0:0)$. Going from HOUR h to HOUR $h-1$ we maintain this invariant by flipping directed paths:

▶ **Definition 25.** *For any edge $\overline{uv}$ with $g(u \to v) > 0$, we say that our algorithm is* flipping *$\overline{uv}$ whenever it decreases $g(u \to v)$ (increasing $g(v \to u)$). Moreover, we say that $\overline{uv}$ is flipped whenever our algorithm has decreased $g(u \to v)$ such that $g(u \to v) = 0$.*

Algorithm (see also Figure 2 and Algorithms 2 and 3 and 4 and 5). Each time frame has a purpose:

- Each HOUR h, the goal is to identify and "fix" all violating out-edges from vertices in L_h; without introducing violating out-edges from vertices in a level L_k with $k > h$. We do this iteratively, using two different steps:
  - Each even MINUTE, we fix all violating out-edges from vertices in L_h, possibly creating violating out-edges from vertices in L_{h+1} to vetices in L_{h-1}.
  - Each odd MINUTE, we fix all violating out-edges from vertices in L_{h+1} to vertices in L_{h-1}. We do this in such a manner, that we create no new violating out-edges from vertices in L_k for $k > h$. However, we may create violating out-edges from level L_h.
- Each even MINUTE m', we define the set $V_{m'} \subset L_h$ of vertices that have at least one violating out-edge. Over $4\lceil \log_{8/7} n \rceil + 2$ rounds, each $u \in V_{m'}$ flips violating out-edges $\overline{uv}$. Moreover:
  - The out-degree $g(u)$ is decreased such that u drops at most one level, and
  - The out-degree $g(v)$ such that v increases its level up to at most $h - 1$.

  We prove that at the end of this MINUTE m', there exist no more $u \in V_{m'}$ with a violating out-edge. Thus, for each vertex $u \in V_{m'}$, either u decreased one level, or all $\overline{uv}$ that were violating now have that $g(u \to v) = 0$, or $v \in L_{h-1}$.
- In each odd MINUTE m, we inspect the vertices $T_m := \{u \in V_{m-1} \text{ that dropped a level}\}$. For each $u \in T_m$, for each edge $\overline{wu}$ with $w \in L_{h+1}$ and $g(w \to u) > 0$ the edge (w, u) has become a violating in the previous MINUTE. We want to fix these edges, whilst guaranteeing that we create no violating in-edges from vertices in level $h + 2$ and above. We obtain this, by recording at the start of the MINUTE for every vertex u its level $l_m(u)$. We then invoke $\ell - h + 1$ SECONDS. Each SECOND s, we create a DAG D_s where the sinks are T_m. We increase for all $u \in T_m$ the out-degree $g(u)$ by flipping a directed path from a source in D_s to T_m. We construct our DAG in such a manner that this procedure does not create new violating in-edges, and that for all SECONDS s, D_{s+1} is a subgraph of D_s.
- At the start of SECOND s of each MINUTE m, we fix for every edge $\overline{uv}$ the values $g_s(u \to v)$, and the levels $l_s(u)$ and $l_s(v)$ of u and v. We then define a graph $D_s = (V_s, E_s)$ as follows:
  - The edges E_s are all violating in-edges to vertices in T_m plus all $\overline{uv}$ with: $l_s(u) = l_m(u) > h + 1$, $l_s(u) > l_s(v)$, and $g_s(u \to v) > 0$.
  - The vertices V_s include T_m plus all vertices in G with a directed path to T_m in E_s.

- The vertices $S_s \subset V_s$ are all sources in D_s (these are not in T_m).

For each $v \in T_m$ we increase $g(v)$ by flipping a directed path from a vertex in S_s to v. We continue this until either $g(v) = (1 + \frac{\eta}{2})^{h+1}$, or, there exist no more edges $(u, v) \in E_s$ with $g(u \to v) > 0$. In both cases, v has no more violating in-edges. To find these directed paths, we create a flow problem on a graph D_s^* where the maximal path has length $\ell + 2$:

- For each $u \in S_s$, we define $\sigma(u) = g(u) - (1 + \frac{\eta}{2})^{l_m(u)-1}$ (the maximal amount we can decrease $g(u)$ such that is does not arrive in level $l_m(u) - 2$). We connect every $u \in S_s$ to a unique source s_u where the edge $\overline{s_u u}$ has capacity $\sigma(u)$.
- For each $v \in T_m$, we define $\delta(v) = (1 + \frac{\eta}{2})^{h+1} - g(v)$ (the maximal amount we can increase $g(v)$ such that it does not arrive in level $h + 1$). We connect every $v \in T_m$ to a unique sink t_v where the edge $\overline{vt_v}$ has capacity $\delta(v)$.
- Each other edge $\overline{uv} \in E_s$ has capacity $g(u \to v)$.

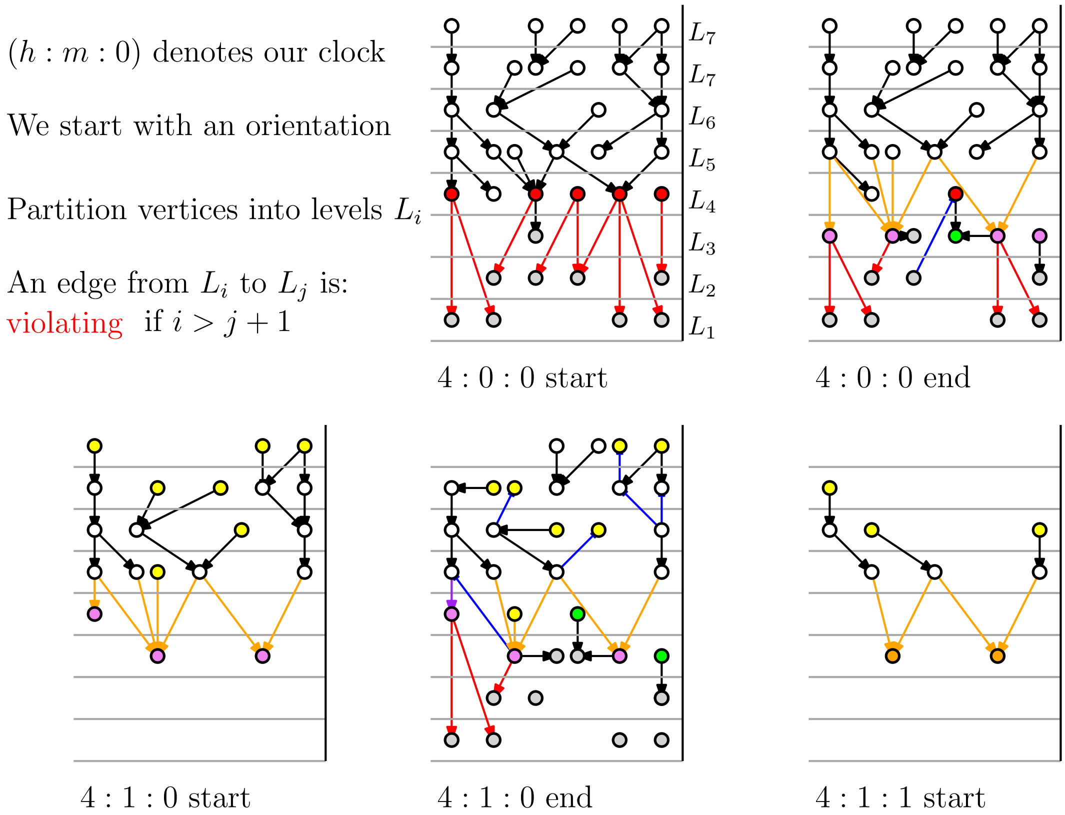

Figure 2 (4 : 0 : 0) – at the first MINUTE start in HOUR 4, we consider all violating out-edges $\overline{uv}$ from level 4 (red). Per definition, these edges point to level 2 or lower.
(4 : 0 : 0) – at the first MINUTE end, either u has dropped a level (pink), v increased their level to $h - 1$ (green) or the edge $\overline{uv}$ is flipped (blue). We consider violating in-edges to pink vertices (orange)
(4 : 1 : 0) – at the first SECOND start, we construct a DAG D_0 where the edges are the orange edges plus black edges. The vertex set are all u with a directed path to a pink vertex ($v \in T_1$). The yellow vertices are S_0.
(4 : 1 : 0) – at the first SECOND end, edges in the DAG may have flipped (blue), vertices in S_0 may have dropped a level or vertices in T_1 may have increased a level (making some edges no longer violating – purple).
(4 : 1 : 1) – at the second SECOND start, we construct a DAG D_1. Note that D_1 is a subgraph of D_0.

7.2 Formal algorithm definition

We now formalise our algorithm top-down, starting with defining variables.

▶ **Definition 26.** *At the start of* $(h, m', 0)$*, where* m' *is even, we fix the set* $V_{m'} := \{v \in L_h \mid v \text{ has at least one violating out-edge}\}$. *At the start of* $(h, m, 0)$ *where* m *is odd, we fix:*

$$T_m := \{v \in V_{m-1} \mid \begin{array}{l} v \text{ decreased by one level in the previous MINUTE and} \\ v \text{ has at least one violating in-edge}\}. \end{array}$$

Finally, we denote for any vertex u *by* $l_m(u)$ *its level* at the start of the MINUTE m.

▶ **Definition 27.** *At the start of* $(h : m : s)$*, where* m *is odd, we fix the following:*

- *For any vertex* u*, we denote by* $l_s(u)$ *its level at the start of the* SECOND.
- *For any edge* $\overline{uv}$ *denote by* $g_s(u \to v)$ *the out-degree from* u *to* v *at the start of the* SECOND.
- *We define the edge set* E_s *as all violating in-edges to vertices in* T_m *plus all* $\overline{uv}$ *with:* $l_s(u) = l_m(u) > h + 1$*,* $l_s(u) > l_s(v)$*, and* $g_s(u \to v) > 0$.
- V_s *are all vertices with a directed path to a vertex in* T_m.
- $D_s = (V_s, E_s)$ *is a DAG where* S_s *are all the sources (per definition* $S_s \cap T_m = \emptyset$*).*

▶ **Definition 28.** *At the start of* (h, m, s) *where* m *is odd, given* T_m*,* S_s *and* D_s*, we define the DAG* D_s^* *by connecting all* $u \in S_s$ *to a unique sink* s_u *and all* $v \in T_m$ *to a unique sink* t_v.

- *For* $u \in S_s$*, the edge* $\overline{s_u u}$ *has capacity* $\sigma(u) = g_s(u) - (1 + \frac{\eta}{2})^{l_m(u)-1}$.
- *For* $v \in T_m$*, the edge* $\overline{v t_v}$ *has capacity* $\delta(v) = (1 + \frac{\eta}{2})^{h+1} - g_s(v)$.
- *Each other edge* $\overline{uv} \in E_s$ *has capacity* $g_s(u \to v)$.

▶ **Observation 29.** *At the start of* $(h : m : s)$ *with* m *odd, if every vertex* u *is given* $l_m(u)$*, whether* $u \in T_m$*, and* h*, then we may compute all elements of Definitions 27 and 28 in* ℓ *rounds.*

Algorithm 2 HOUR(h).

```
for m = 0 to 2⌈η^{-1}⌉ + 2 do
  if  m is even then
    EVENMINUTE(h, m)
  else
    ODDMINUTE(h, m)
if h > 0 then
  HOUR(h − 1)
```

Algorithm 3 EVENMINUTE(int h, int m).

```
V_m := {v ∈ L_h | v has at least one violating out-edge }
for  t = 0 to 2⌈log_{8/7} n⌉ + 1 do
  Let A_t ⊂ V_m be the set of vertices at level h with at least one violating out-edge
  Each a ∈ A_t determines the set E_t(a) of violating out-edges.
  Each a ∈ A_t computes δ_t(a) = g(a) − (1 + η)^{h−1} and reports δ_t(a)/|E_t(a)| across E_t(a).
  /* next round:                                                                  */
  Let B_t be the set of vertices that receive at least one violating in-edge from A_t.
  Each b ∈ B_t determines the set I_t(b) of vertices that reported a value to v.
  Each b ∈ B_t sorts the a ∈ I_t(b) by δ(a).
  Each b ∈ B_t greedily decreases g(a→b) by at most δ_t(a) for a ∈ I_t(b); until g(b) = (1 + η/2)^h.
```

Algorithm 4 ODDMINUTE(int h, int m).

$T_m := \{v \in V_{m-1} \mid v$ has at least one violating in-edge $\}$
for $s = \ell$ down to h **do**
  SECOND(h, m, s)

Algorithm 5 SECOND(int h, int m, int s).

Compute the graph D_s in ℓ rounds.
Compute the graph D_s^* by adding a shared source to S_s and a shared sink to T_m.
f = COMPUTEBLOCKINGFLOW(D_s^*)
Flip all edges in D_s with the corresponding flow in f.

7.3 Proving our algorithm's correctness

Per definition, our algorithm runs in $O(\eta^{-1} \log n \cdot \ell \cdot (\ell + \text{Blocking}(\ell, n)) = O(\varepsilon^{-3} \log^4 n \cdot (\varepsilon^{-2} \log^2 n + \text{Blocking}(\varepsilon^{-2} \log^2 n, n)))$ rounds. What remains is to show that we maintain Invariant 1, which implies that we compute an η-fair orientation.
We note that by the choice of our algorithm's variables, we have the following property:

▶ **Observation 30.** *For all times $(h : m : s)$ with m odd:*
- *Vertices at level $k > h$ only decrease their level and by at most one (because afterwards, $l_s(u) < l_m(u)$ and thus u has no out-edges in E_s),*
- *vertices at level $h - 1$ only increase their level (by at most 1), and*
- *vertices at level $k' < h - 1$ and level h do not change their level.*

We use this observation in the full version to show that we maintain Invariant 1 by induction. Trivially, Invariant 1 holds at the start of $(\ell : 0 : 0)$. We now assume that the invariant holds at $(h : 0 : 0)$, i.e. that there are no violating out-edges from level L_k with $k > h$. We prove that our algorithm ensures that at the start of $(h - 1 : 0 : 0)$ there are no violating out-edges from level L_k with $k \geq h$.

Moreover, we maintain the invariant that throughout HOUR $(h - 1)$ there are no violating out-edges from level $L_{k'}$ with $k' > h + 1$. We prove this in the following way:

- During $(h : m : s)$ for m even, our algorithm eliminates all violating edges going from L_h. We show that in each iteration, the number of violating edges from L_h drops by a factor 7/8. The graph has at most n^2 edges. This implies that after the MINUTE, there are no more violating edges going from L_h. However, there may now be violating out-edges from vertices in L_{h+1} to vertices in $T_{m+1} \subseteq L_{h-1}$.
- During $(h : m : s)$ for m odd, our algorithm eliminates all violating edges going from L_{h+1} to T_m. We show that for all s, the DAG D_s is a subgraph of D_{s-1} where the height is one fewer. Since the height of D_s is at most $\ell - h + 1$, this implies that after the MINUTE m, the graph D_s is empty and there are no more violating in-edges to T_m.
- Each MINUTE, our algorithm alternates between having violating edges from L_h and L_{h+1}. We show that our algorithm can alternate at most η times before there are no violating edges from both L_h and L_{h+1}, which implies Invariant 1 at the start of $(h - 1 : 0 : 0)$.

Invariant 1 implies that we compute an η-fair orientation at time $(0 : 0 : 0)$, thus:

▶ **Theorem 16.** *Suppose one can compute a blocking flow in an n-node h-layered DAG in* Blocking(h, n) *rounds. There exists an algorithm in CONGEST that given a unit-weight graph G and $\varepsilon > 0$ computes in $O(\varepsilon^{-3} \log^4 n \cdot (\varepsilon^{-2} \log^2 n + \text{Blocking}(\varepsilon^{-2} \log^2 n, n)))$ rounds an orientation $\overrightarrow{G}$ such that for all $v \in V$: $g(v) \in [(1 + \varepsilon)^{-1} \rho^*(v)), (1 + \varepsilon)\rho^*(v)\rho^*(v)]$.*

We plug in the runtime of Blocking(h, n) of Lemma 21 by Haeupler, Hershkowitz, and Saranurak [21] to obtain the following runtime:

▶ **Corollary 31.** *There is an algorithm in CONGEST that given a unit-weight graph G and an $\varepsilon > 0$ computes an orientation $\overrightarrow{G}$ such that $\forall v \in V$, the out-degree $g(v) \in [(1+\varepsilon)^{-1}\rho^*(v), (1+\varepsilon)\rho^*(v)]$ in:*
- $\tilde{O}(\varepsilon^{-11} \log^{12} n)$ *rounds with high probability, or*
- $\tilde{O}(\varepsilon^{-15} \log^{16} n \cdot 2^{O(\sqrt{\log n})})$ *deterministic rounds.*

Finally, we apply Corollary 10 which states that $\rho^{\max}(G) = \Delta^{\min}(G) = \max_v g^*(v)$ to conclude:

▶ **Corollary 32.** *There is a deterministic algorithm in CONGEST that given a unit-weight graph G and an $\varepsilon > 0$, that computes an orientation $\overrightarrow{G}$ such that:*

$$\max_{v \in V} g(v) \in [(1+\varepsilon)^{-1}\rho^{\max}(G), (1+\varepsilon)\rho^{\max}(G)],$$

in a number of rounds that is sublinear in n.

This is the first deterministic sublinear algorithm that solves Problem 2.2.

References

1 Bahman Bahmani, Ravi Kumar, and Sergei Vassilvitskii. Densest subgraph in streaming and mapreduce. *Proc. VLDB Endow.*, 5(5):454–465, 2012. doi:10.14778/2140436.2140442.

2 Soheil Behnezhad, Mahsa Derakhshan, Alireza Farhadi, MohammadTaghi Hajiaghayi, and Nima Reyhani. Stochastic matching on uniformly sparse graphs. In *Algorithmic Game Theory: 12th International Symposium, SAGT 2019, Athens, Greece, September 30 – October 3, 2019, Proceedings*, pages 357–373, Berlin, Heidelberg, 2019. Springer-Verlag. doi:10.1007/978-3-030-30473-7_24.

3 Suman K. Bera, Sayan Bhattacharya, Jayesh Choudhari, and Prantar Ghosh. A new dynamic algorithm for densest subhypergraphs. In Frédérique Laforest, Raphaël Troncy, Elena Simperl, Deepak Agarwal, Aristides Gionis, Ivan Herman, and Lionel Médini, editors, *WWW '22: The ACM Web Conference 2022, Virtual Event, Lyon, France, April 25 - 29, 2022*, pages 1093–1103. ACM, 2022. doi:10.1145/3485447.3512158.

4 Sayan Bhattacharya, Monika Henzinger, Danupon Nanongkai, and Charalampos E. Tsourakakis. Space- and time-efficient algorithm for maintaining dense subgraphs on one-pass dynamic streams. In Rocco A. Servedio and Ronitt Rubinfeld, editors, *Proceedings of the Forty-Seventh Annual ACM on Symposium on Theory of Computing, STOC 2015, Portland, OR, USA, June 14-17, 2015*, pages 173–182. ACM, 2015. doi:10.1145/2746539.2746592.

5 Digvijay Boob, Yu Gao, Richard Peng, Saurabh Sawlani, Charalampos Tsourakakis, Di Wang, and Junxing Wang. Flowless: Extracting densest subgraphs without flow computations. In *Proceedings of The Web Conference 2020*, 2020.

6 Glencora Borradaile, Jennifer Iglesias, Theresa Migler, Antonio Ochoa, Gordon Wilfong, and Lisa Zhang. Egalitarian graph orientations. *Journal of Graph Algorithms and Applications*, 2017.

7 Glencora Borradaile, Theresa Migler, and Gordon Wilfong. Density decompositions of networks. In *Conference on Complex Networks (CompleNet)*. Springer, 2018.

8 Gerth Stølting Brodal and Rolf Fagerberg. Dynamic representations of sparse graphs. In *In Proc. 6th International Workshop on Algorithms and Data Structures (WADS)*, pages 342–351. Springer-Verlag, 1999. doi:10.1007/3-540-48447-7_34.

9 Moses Charikar. Greedy approximation algorithms for finding dense components in a graph. In *Approximation Algorithms for Combinatorial Optimization (APPROX)*. Springer, 2003.

10 Chandra Chekuri, Aleksander Bjørn Christiansen, Jacob Holm, Ivor van der Hoog, Kent Quanrud, Eva Rotenberg, and Chris Schwiegelshohn. Adaptive out-orientations with applications. In *Proceedings of the 2024 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 3062–3088. SIAM, 2024.

11 Chandra Chekuri, Kent Quanrud, and Manuel R Torres. Densest subgraph: Supermodularity, iterative peeling, and flow. In *Proceedings of the 2022 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*. SIAM, 2022.

12 Aleksander Bjørn Grodt Christiansen, Krzysztof Nowicki, and Eva Rotenberg. Improved dynamic colouring of sparse graphs. In Barna Saha and Rocco A. Servedio, editors, *Proceedings of the 55th Annual ACM Symposium on Theory of Computing, STOC 2023, Orlando, FL, USA, June 20-23, 2023*, pages 1201–1214. ACM, 2023. doi:10.1145/3564246.3585111.

13 Aleksander Bjørn Grodt Christiansen and Eva Rotenberg. Fully-Dynamic $\alpha + 2$ Arboricity Decompositions and Implicit Colouring. In Mikołaj Bojańczyk, Emanuela Merelli, and David P. Woodruff, editors, *49th International Colloquium on Automata, Languages, and Programming (ICALP 2022)*, volume 229 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 42:1–42:20, Dagstuhl, Germany, 2022. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. doi:10.4230/LIPIcs.ICALP.2022.42.

14 Maximilien Danisch, T.-H. Hubert Chan, and Mauro Sozio. Large scale density-friendly graph decomposition via convex programming. In *Proceedings of the 26th International Conference on World Wide Web*, WWW '17, pages 233–242, Republic and Canton of Geneva, CHE, 2017. International World Wide Web Conferences Steering Committee. doi:10.1145/3038912.3052619.

15 Atish Das Sarma, Ashwin Lall, Danupon Nanongkai, and Amitabh Trehan. Dense subgraphs on dynamic networks. In Marcos K. Aguilera, editor, *Distributed Computing - 26th International Symposium, DISC 2012, Salvador, Brazil, October 16-18, 2012. Proceedings*, volume 7611 of *Lecture Notes in Computer Science*, pages 151–165. Springer, 2012. doi:10.1007/978-3-642-33651-5_11.

16 Manuela Fischer, Mohsen Ghaffari, and Fabian Kuhn. Deterministic distributed edge-coloring via hypergraph maximal matching. In *2017 IEEE 58th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 180–191. IEEE, 2017. doi:10.1109/FOCS.2017.25.

17 András Frank and Kazuo Murota. Fair integral network flows. *Mathematics of Operations Research*, 2023.

18 Satoru Fujishige. Lexicographically optimal base of a polymatroid with respect to a weight vector. *Mathematics of Operations Research*, 5(2):186–196, 1980. doi:10.1287/MOOR.5.2.186.

19 Mohsen Ghaffari, David G Harris, and Fabian Kuhn. On derandomizing local distributed algorithms. In *2018 IEEE 59th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 662–673. IEEE, 2018. doi:10.1109/FOCS.2018.00069.

20 Mohsen Ghaffari and Hsin-Hao Su. Distributed degree splitting, edge coloring, and orientations. In Philip N. Klein, editor, *Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2017, Barcelona, Spain, Hotel Porta Fira, January 16-19*, pages 2505–2523. SIAM, 2017. doi:10.1137/1.9781611974782.166.

21 Bernhard Haeupler, D Ellis Hershkowitz, and Thatchaphol Saranurak. Maximum length-constrained flows and disjoint paths: Distributed, deterministic, and fast. In *Proceedings of the 55th Annual ACM Symposium on Theory of Computing*, pages 1371–1383, 2023. doi:10.1145/3564246.3585202.

22 Elfarouk Harb, Kent Quanrud, and Chandra Chekuri. Faster and scalable algorithms for densest subgraph and decomposition. *Advances in Neural Information Processing Systems*, 35:26966–26979, 2022.

23 Elfarouk Harb, Kent Quanrud, and Chandra Chekuri. Convergence to lexicographically optimal base in a (contra) polymatroid and applications to densest subgraph and tree packing. In *European Symposium on Algorithms (ESA)*. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2023. doi:10.4230/LIPICS.ESA.2023.56.

24 David G Harris. Distributed local approximation algorithms for maximum matching in graphs and hypergraphs. In *2019 IEEE 60th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 700–724. IEEE, 2019. doi:10.1109/FOCS.2019.00048.

25 Krzysztof Onak, Baruch Schieber, Shay Solomon, and Nicole Wein. Fully dynamic mis in uniformly sparse graphs. *ACM Trans. Algorithms*, 16(2), March 2020. doi:10.1145/3378025.

26 Lu Qin, Rong-Hua Li, Lijun Chang, and Chengqi Zhang. Locally densest subgraph discovery. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '15, pages 965–974, New York, NY, USA, 2015. Association for Computing Machinery. doi:10.1145/2783258.2783299.

27 Saurabh Sawlani and Junxing Wang. Near-optimal fully dynamic densest subgraph. In *Proceedings of the 52nd Annual ACM SIGACT Symposium on Theory of Computing*, pages 181–193, 2020. doi:10.1145/3357713.3384327.

28 Hsin-Hao Su and Hoa T. Vu. Distributed dense subgraph detection and low outdegree orientation. In Hagit Attiya, editor, *34th International Symposium on Distributed Computing, DISC 2020, October 12-16, 2020, Virtual Conference*, volume 179 of *LIPIcs*, pages 15:1–15:18. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2020. doi:10.4230/LIPIcs.DISC.2020.15.

29 Nikolaj Tatti and Aristides Gionis. Density-friendly graph decomposition. In *Proceedings of the 24th International Conference on World Wide Web*, WWW '15, pages 1089–1099, Republic and Canton of Geneva, CHE, 2015. International World Wide Web Conferences Steering Committee. doi:10.1145/2736277.2741119.

30 Yalong Zhang, Rong-Hua Li, Qi Zhang, Hongchao Qin, and Guoren Wang. Efficient algorithms for density decomposition on large static and dynamic graphs. *Proceedings of the VLDB Endowment*, 2024.