New Algorithms and Lower Bounds for Streaming Tournaments

Prantar Ghosh
Department of Computer Science, Georgetown University, Washington, DC, USA

Sahil Kuchlous
Harvard University, Cambridge, MA, USA

Abstract

We study fundamental directed graph (digraph) problems in the streaming model. An initial investigation by Chakrabarti, Ghosh, McGregor, and Vorotnikova [SODA'20] on streaming digraphs showed that while most of these problems are provably hard in general, some of them become tractable when restricted to the well-studied class of tournament graphs where every pair of nodes shares exactly one directed edge. Thus, we focus on tournaments and improve the state of the art for multiple problems in terms of both upper and lower bounds.

Our primary upper bound is a deterministic single-pass semi-streaming algorithm (using $\widetilde{O}(n)$ space for n-node graphs, where $\widetilde{O}(.)$ hides polylog(n) factors) for decomposing a tournament into *strongly connected components* (SCC). It improves upon the previously best-known algorithm by Baweja, Jia, and Woodruff [ITCS'22] in terms of both space and passes: for $p \geqslant 1$, they used $(p+1)$ passes and $\widetilde{O}(n^{1+1/p})$ space. We further extend our algorithm to digraphs that are close to tournaments and establish tight bounds demonstrating that the problem's complexity grows smoothly with the "distance" from tournaments. Applying our SCC-decomposition framework, we obtain improved – and in some cases, optimal – tournament algorithms for s,t*-reachability*, *strong connectivity*, *Hamiltonian paths and cycles*, and *feedback arc set*.

On the other hand, we prove lower bounds exhibiting that some well-studied problems – such as (exact) *feedback arc set* and s,t*-distance* – remain hard (require $\Omega(n^2)$ space) on tournaments. Moreover, we generalize the former problem's lower bound to establish space-approximation tradeoffs: any single-pass $(1 \pm \varepsilon)$-approximation algorithm requires $\Omega(n/\sqrt{\varepsilon})$ space. Finally, we settle the streaming complexities of two basic digraph problems studied by prior work: *acyclicity testing* of tournaments and *sink finding* in DAGs. As a whole, our collection of results contributes significantly to the growing literature on streaming digraphs.

2012 ACM Subject Classification Theory of computation → Streaming, sublinear and near linear time algorithms; Theory of computation → Graph algorithms analysis

Keywords and phrases tournaments, streaming algorithms, graph algorithms, communication complexity, strongly connected components, reachability, feedback arc set

Digital Object Identifier 10.4230/LIPIcs.ESA.2024.60

Related Version *Full Version*: `https://arxiv.org/abs/2405.05952` [25]

Funding *Prantar Ghosh*: Supported in part by NSF under award 1918989. Part of this work was done while the author was at DIMACS, Rutgers University, supported in part by a grant (820931) to DIMACS from the Simons Foundation.
Sahil Kuchlous: Research done in part as a participant in the DIMACS REU program 2023 at Rutgers University, supported by NSF grant CCF-2150186.

Acknowledgements We are extremely grateful to Srijon Mukherjee for letting us know about the IOITC question on determining strong connectivity of a tournament from its set of indegrees, solving and building on which we obtained our results for SCC decomposition. Later we came to know that the question (and its solution) was designed by Sreejata Kishor Bhattacharya and Archit Karandikar at IOITC 2017. We also thank Michael Saks for the observation of simpler analysis for tournaments (mentioned in Remark 12) and Sepehr Assadi, Amit Chakrabarti, and Madhu Sudan for helpful discussions. Finally, we thank the organizers of the DIMACS REU program for making this collaboration possible.

1 Introduction

A graph streaming algorithm reads an input graph by making one or a few passes over a sequence of its edges, while maintaining a small summary that it uses to solve an underlying problem. While any memory sublinear in the number of edges is interesting, we typically aim for *semi-streaming* space, i.e., $\widetilde{O}(n)$ space[1] for n-node graphs: Feigenbaum et al. [22] observed that many graph problems become tractable at this bound. The study of such algorithms is motivated by modern large graphs such as web graphs given by hyperlinks, social networks given by "follows," citation networks, and biological networks. Chakrabarti, Ghosh, McGregor, and Vorotnikova [14] observed that while many of these graphs are *directed*, the graph streaming literature spanning over two decades had hitherto focused almost exclusively on undirected graphs (see [35] for a survey), with very few exceptions. In light of this, they conducted a thorough investigation on streaming directed graphs (digraphs) and laid a foundation for their study in this model.

Their findings, however, showed that most fundamental digraph problems are, in general, provably hard in streaming, perhaps justifying the lack of focus on these problems in prior work. At the same time, they found that restriction to *tournament* graphs – where every pair of vertices shares exactly one directed edge – makes many of these problems tractable yet non-trivial in streaming. Observe that tournaments, by definition, are dense graphs with $\Theta(n^2)$ edges, and hence, one of the prime examples of graphs that motivate memory-efficient streaming algorithms. Furthermore, many real-world graphs, such as ones representing *comparison matrices* [23] that record pairwise preferences among a large set of items, are tournaments and seek succinct sketches in the modern world of big data. Thus, the study of tournaments is well-motivated in the streaming model and in this work, we significantly contribute to this study.

1.1 Background and Context

Chakrabarti et al. [14] studied classical digraph problems including s,t*-reachability*, *topological sorting*, *acyclicity testing*, and (approximate) *feedback arc set*. These problems turn out to be hard for general digraphs: they need $\Omega(n^2)$ space in a single pass [22, 14] (i.e., we need to store almost the entire graph) and roughly $n^{1+\Omega(1/p)}$ space for p passes [27, 14]. In contrast, [14] showed that the last three problems in the above list admit single-pass semi-streaming algorithms on tournaments. However, two important questions remained: (i) "*what is the streaming complexity of* s,t*-reachability on tournaments?*" and (ii) "*which problems are hard on streaming tournaments?*"

Reachability on general digraphs, one of the most fundamental graph algorithmic problems, has received considerable attention in the streaming literature. It has strong lower bounds establishing $\Omega(n^{2-o(1)})$ space to be necessary even for $O(\sqrt{\log n})$ passes [17] (also see [6] who proved it for 2 passes), while almost $\log n$ passes are needed for semi-streaming space [27]. On the upper-bound side, Assadi et al. [4] designed an $O(\sqrt{n})$-pass semi-streaming algorithm. Closing this gap between the number of passes needed for semi-streaming space seems difficult and it is one of the major open questions in multipass streaming (see the very recent survey by Assadi [3]). Thus, resolving the complexity of the problem for the special class of tournaments seems to be a natural basic step before we try it for the general case.

[1] Throughout the paper, we use $\widetilde{O}(f)$ to hide polylog(f) factors.

Baweja, Jia, and Woodruff [8] gave the first algorithm for reachability on tournaments: although they didn't explicitly mention it, a $(p+1)$-pass $\widetilde{O}(n^{1+1/p})$-space algorithm follows from their SCC (strongly connected components) decomposition algorithm for tournaments that uses the same space-pass tradeoff. But this bound was not proven to be tight for either reachability or SCC decomposition,[2] and hence the former's streaming complexity on tournaments still remained unresolved. Additionally, one can ask more generally, "*what is the streaming complexity of SCC decomposition on tournaments?*"

In this work, we answer this general question by designing a deterministic single-pass semi-streaming algorithm for SCC decomposition on tournaments. In fact, our algorithm works for a broader class of digraphs: those with *at least* one directed edge between each pair of vertices. Our algorithm is very simple: during the stream we only need to store the degrees of each vertex! The post-processing phase and its analysis are more elaborate, and we demonstrate how we can derive the *SCC-graph* (obtained by contracting each SCC into a supernode) from just the degree information.[3] Consequently, we get single-pass semi-streaming algorithms for reachability and strong connectivity on tournaments. We prove matching lower bounds for each of these problems, also settling their complexities.

We extend our algorithms to handle graphs that are "almost" tournaments. Precisely, for digraphs that need to add or delete a total of k edges to form a tournament (i.e., are *k-close* to tournament), we obtain $\widetilde{O}(n+k)$-space solutions by using our tournament algorithms as subroutines. Additionally, our matching $\Omega(n+k)$-space lower bounds for these graphs show that this is as good as generalizations can get: it exhibits how the complexity of the problem increases smoothly with the input's "distance" from the tournament property and highlights the importance of the property in attaining sublinear-space solutions. We show further applications of our SCC decomposition framework for tournaments: it yields efficient streaming algorithms for problems like *Hamiltonian cycle*, *Hamiltonian path*, and *feedback arc set* on tournaments (see Section 1.2 for details).

As we continue to find efficient tournament algorithms for well-studied problems, it is natural to revisit the second important question mentioned above: which streaming problems remain hard on this class of graphs? Prior work did not show any strong lower bound on these graphs; in particular, we did not know any tournament problem with a single-pass $\Omega(n^2)$-space lower bound. We prove the first such lower bound for two well-studied problems, namely *s,t-distance* and (exact) *feedback arc set* on tournaments (FAS-T), and extend the latter result to a generalized lower bound for space-approximation tradeoffs. We prove this general result using communication complexity, as is standard for streaming lower bounds, but the underlying communication lower bound uses a stronger version of the *direct-sum* argument [15] that might be of independent interest.

Finally, we settle the streaming complexities of two basic problems studied by prior work: acyclicity testing of tournaments and sink finding in DAGs. Collectively, our results significantly advance the state of the art of streaming digraph problems.

---

[2] For general digraphs, [8] proved an $n^{1+\Omega(1/p)}$ space lower bound for p-pass algorithms solving SCC decomposition. This, however, doesn't extend to tournaments, and hence, doesn't complement the said upper bound.

[3] We were informed via personal communication that it was known how to test strong connectivity using the degree information as it appeared as an IOITC (International Olympiad in Informatics Training Camp) problem, but we did not find any published result or documented version of it. Nevertheless, finding the SCC-graph requires considerably more work than just testing strong connectivity.

1.2 Our Contributions and Comparison to Prior Work

Table 1 Summary of our main upper and lower bounds, along with the state-of-the-art results for comparison. The problems are formally defined in Section 1.2. Here, "k-close input" means that the input digraph is at most k edges away from being a tournament (see Section 3.1.2 for formal definition). The parameter p is any integer $\geqslant 1$ and ε is any positive real.

Problem	Passes	Space	Reference
SCC-DEC-T	$p+1$	$\widetilde{O}(n^{1+1/p})$	[8]
SCC-DEC-T	1	$\Theta(n \log n)$	Corollary 13
SCC-DEC (k-close input)	1	$\widetilde{\Theta}(n+k)$	Corollaries 16 and 17
REACH-T, STR-CONN-T	$p+1$	$\widetilde{O}(n^{1+1/p})$	[8] (REACH-T result implicit)
REACH-T, STR-CONN-T	1	$\widetilde{O}(n)$	Corollaries 18 and 19
REACH-T, STR-CONN-T	p	$\Omega(n/p)$	Theorems 20 and 21
REACH, STR-CONN (k-close input)	1	$\widetilde{\Theta}(n+k)$	Theorems 22 and 23
HAM-CYCLE-T	$O(\log n)$	$\widetilde{O}(n)$	Theorem 25
FAS-SIZE-T	1	$\Omega(n^2)$	Theorem 29
FAS-T	1	$\Omega(n^2)$	Lemma 27
$(1+\varepsilon)$-approx. FAS-T	1	$\Omega\left(\min\{n^2, n/\sqrt{\varepsilon}\}\right)$	Theorem 28
$(1+\varepsilon)$-approx. FAS-T	1	$\widetilde{O}\left(\min\{n^2, n/\varepsilon^2\}\right)$	[14]
STDIST-T	1	$\Omega(n^2)$	Theorem 30
ACYC-T	p	$\Omega(n/p)$	[14], also Theorem 31
ACYC-T	p	$\widetilde{O}(n/p)$	Theorem 32
SINK-DAG	p	$\Theta(n/p)$	Theorem 33 (upper bound trivial)

Table 1 summarizes our main results. Below, we define each problem formally and discuss the context and comparisons to prior work. For each problem, the suffix "-T" indicates the version of the problem where the input is a tournament.

SCC decomposition, strong connectivity, and reachability. Recall that a graph is called strongly connected if and only if each vertex is reachable from every other vertex. A strongly connected component (SCC) of a graph is a maximal subgraph that is strongly connected. In the streaming model, by SCC decomposition, we mean partitioning the vertex set into subsets where each subset induces an SCC (we don't need the edges contained inside the SCCs). Formally, the problem is defined as follows.

SCC-DEC: Given a digraph $G = (V, E)$, output a partition $(V_1, \ldots, V_\ell)$ of V such that each V_i induces a strongly connected component of G.

We design a single-pass semi-streaming algorithm for SCC-DEC-T (Corollary 13). Note that our algorithm not only outputs the partition, but the SCC-graph which is a DAG obtained by contracting each SCC into a (super)node. Observe that since the SCC-graph of a tournament is an acyclic tournament, it can be represented simply as an ordering of its SCC's, where the edges between the components are implicit: all of them go from left to right. Our algorithm outputs this ordering.

As a consequence, we get single-pass semi-streaming algorithms for checking reachability (Corollary 18) and strong connectivity (Corollary 19) on tournaments. The general problems are defined below.

STR-CONN: Given a digraph $G = (V, E)$, output whether it is strongly connected.

REACH: Given a digraph $G = (V, E)$ and nodes $s, t \in V$, output whether t has a directed path from (i.e., is *reachable* from) s.

Baweja, Jia, and Woodruff [8] gave $\widetilde{O}(n^{1+1/p})$-space algorithms for SCC-DEC-T and STR-CONN-T using $p+1$ passes for any $p \geqslant 1$. An algorithm for REACH-T using same space and passes is implicit, since it can be determined from the SCC-graph (which their algorithm also outputs). Observe that their algorithm needs almost $\log n$ passes to achieve semi-streaming space. Further, it needs at least 3 passes to even attain sublinear ($o(n^2)$) space. In contrast, we achieve semi-streaming space in just a single pass.

Further, [8] showed an $n^{1+\Omega(1/p)}$ space lower bound for p-pass algorithms solving SCC-DEC or STR-CONN, while [27] gave a similar lower bound for REACH. Our results rule out the possibility of extending these lower bounds to tournaments, and show a large gap between the complexity of each problem and its tournament version.

While semi-streaming space is clearly optimal for SCC-DEC-T since it is the space needed to simply present the output, it is not clear that the same holds for STR-CONN-T and REACH-T which have single-bit outputs. We prove matching $\Omega(n)$-space single-pass lower bounds for these problems (Theorems 20 and 21). In fact, we prove more general lower bounds that show that semi-streaming space is optimal for them (up to polylogarithmic factors) even for polylog(n) passes. It is important to note that while our upper bounds are deterministic, our lower bounds hold even for randomized algorithms, thus completely settling the single-pass complexity of these problems (up to logarithmic factors).

Extension to almost-tournaments and tight bounds. A possible concern about our results is that they are restricted to tournaments, a rather special class of digraphs. Indeed, our results are provably not generalizable to any arbitrary digraph since there exist graphs that don't admit sublinear-space solutions for these problems [22, 8]. Can we still cover a broader class of graphs? To address this, we first show that our algorithms work as is for any digraph without non-edges (hence allowing bidirected edges between arbitrary number of vertex pairs). Secondly, we generalize our algorithm to work for digraphs that are "close to" tournaments (but can have non-edges). A standard measure of closeness to a graph property $\mathcal{P}$ (widely used in areas such as Property Testing) is the number of edges that need to be added or deleted such that the graph satisfies $\mathcal{P}$. In similar vein, we define a digraph G to be *k-close* to a tournament if a total of at most k edge additions and deletions to G results in a tournament. We design an $\widetilde{O}(n+k)$-space algorithm for any such digraph. Thirdly, and perhaps more importantly, we prove that this is tight: there exist digraphs k-close to tournaments where $\Omega(n+k)$ space is necessary. This exhibits a smooth transition in complexity based on the "distance" from the tournament property. We see this result as an important conceptual contribution of our work: "tournament-like" properties are indeed necessary to obtain efficient streaming algorithms for these problems, justifying the almost-exclusive focus of prior work [14, 8] as well as ours on the specific class of tournaments when designing digraph streaming algorithms.

Our algorithms for almost-tournaments use our scheme for (exact) tournaments as a subroutine. We call it exponentially many times in the worst case, and we can do so without any error by leveraging the fact that our algorithm is deterministic. Although the streaming model doesn't focus on time complexity, the exponential runtime might be unsatisfactory. But we show that it is somewhat necessary, at least for algorithms that use a tournament algorithm as a blackbox (see Section 3.2.3 and the corresponding section in the full version [25] for details). This is similar in spirit to a result of [14] that justified the exponential runtime of their algorithm for approximate FAS-T.

Hamiltonian cycle and Hamiltonian path. We exhibit applications of our SCC-DEC-T algorithm to obtain efficient algorithms for the problems of finding Hamiltonian cycles and Hamiltonian paths.

HAM-CYCLE: Given a digraph $G = (V, E)$, find a directed cycle that contains all nodes in V if one exists or output NONE otherwise.

HAM-PATH: Given a digraph $G = (V, E)$, find a directed path that contains all nodes in V if one exists or output NONE otherwise.

Note that although these problems are NP-hard in general [31], they admit polynomial time solutions for tournaments [10]. Further, it is known that every tournament contains a Hamiltonian path, but it may or may not contain a Hamiltonian cycle. A tournament is Hamiltonian if and only if it is strongly connected [12], and so we can use our STR-CONN-T algorithm to determine the existence of a Hamiltonian cycle in a single-pass and semi-streaming space. Thus, the problem boils down to finding such a cycle if it exists.

We give an $O(\log n)$-pass semi-streaming algorithm for HAM-CYCLE-T (Theorem 25). Although [8] gave such an algorithm for HAM-PATH-T, to our knowledge, no streaming algorithm for HAM-CYCLE-T was known. We design our algorithm by using SCC-DEC-T as a primitive and devising a streaming simulation of a parallel algorithm by Soroker [41]. Additionally, we prove that if one instead used [8]'s SCC-DEC-T algorithm as the subroutine, it would lead to an $O(\log^2 n)$-pass algorithm in the worst case.

As another application of our SCC framework, we obtain improved algorithms for HAM-PATH-T when the input tournament is known to have somewhat small connected components. Observe that in the extreme case when the maximum size s of an SCC is 1, the tournament is a DAG, and a trivial algorithm that sorts the nodes by their indegrees finds the Hamiltonian path in a single semi-streaming pass. This is significantly better than the general p-pass $\widetilde{O}(n^{1+1/p})$-space algorithm of [8]. We demonstrate that even when s is larger but still "reasonably small", specifically, whenever $s < n^{(p-1)/p}$ for some $p \geqslant 1$, we get better p-pass algorithms than [8] and the space complexity grows smoothly with s. A similar algorithm for approximate FAS-T is also obtained. See Section 3.3.1 and the corresponding section in the full version [25] for details on these results.

Feedback arc set. The feedback arc set (FAS) in a digraph is a set of arcs (directed edges) whose deletion removes all directed cycles. Indeed, we are interested in finding the *minimum* FAS. One version asks for only the *size* of a minimum FAS.

FAS-SIZE: Given a digraph $G = (V, E)$, output the size of a minimum FAS.

For the version asking for the actual set, since the output can have size $\Theta(n^2)$ in the worst case, the streaming model considers an equivalent version where the output-size is always $\widetilde{O}(n)$ [14, 8]. We define it below.

FAS: Given a digraph $G = (V, E)$, find an ordering of vertices in V such that the number of back-edges (edges going from right to left in the ordering) is minimized.

The FAS and FAS-SIZE problems, even on tournaments (FAS-T and FAS-SIZE-T), are NP-hard [16]. However, each of FAS-T and FAS-SIZE-T admits a PTAS [32]. The FAS problem has a variety of applications including in rank aggregation [32], machine learning [7], social network analysis [39], and ranking algorithms [24]. In the streaming model, [14] and [8] gave multipass polynomial-time approximation algorithms for these problems (see Section 1.3 for details). We, however, focus on the single-pass setting. The best-known single-pass algorithm achieves $(1 + \varepsilon)$-approximation in only $\widetilde{O}(n/\varepsilon^2)$ space, albeit in exponential time [14]. However, there was no complementary space lower bound for FAS-T or FAS-SIZE-T. The said paper gave

lower bounds only for the harder versions of the problems on general digraphs; these do not extend to tournaments. In fact, there was no lower bound even for *exact* FAS-T. Note that although this is an NP-hard problem, an exponential-time but sublinear-space streaming algorithm is not ruled out. In fact, such algorithms for NP-hard problems do exist in the literature (see for instance, a deterministic $(1+\varepsilon)$-approximation to correlation clustering (which is APX-hard) [2, 9], or algorithms for max coverage [36]).

In light of this, and since the state-of-the-art $(1+\varepsilon)$-approximation algorithm already uses exponential time, it is natural to ask whether it can be improved to solve exact FAS-T. We rule this out by proving a single-pass lower bound of $\Omega(n^2)$ space (Lemma 27). No such lower bound was known for *any* tournament problem prior to this.[4] Additionally, we prove the same lower bound even for FAS-SIZE-T (Theorem 29) whose output is only a real value (rather than a vertex ordering like FAS-T).

We further extend our FAS-T lower bound to establish a smooth space-approximation tradeoff: given any $\varepsilon > 0$, a $(1+\varepsilon)$ approximation to FAS-T requires $\Omega(\min\{n^2, n/\sqrt{\varepsilon}\})$ space (Theorem 28). Note that for $\varepsilon \leq 1/n^2$, the lower bound follows directly from Lemma 27 since it is equivalent to the exact version. However, it does not follow that the bound holds for larger values of ε. We prove that it is indeed the case for *any* $\varepsilon > 0$.

Distance, acylicity testing, and sink-finding. It is natural to ask what other problems are hard on tournaments. In particular, is there any polynomial-time solvable well-studied problem that requires $\Omega(n^2)$ space in a single pass? We answer this in the affirmative by exhibiting the s,t-distance problem as an example (Theorem 30).

STDIST: Given a digraph $G = (V, E)$ and vertices $s, t \in V$, find the distance, i.e., the length of the shortest directed path, between s and t.

Our results also exhibits a contrast between STDIST-T and REACH-T in streaming: while checking whether s has a path to t is easy, finding the exact distance is hard.

Next, we revisit the basic problem of testing acylicity of a digraph studied by [14, 8].

ACYC: Given a digraph $G = (V, E)$, is there a directed cycle in G?

We design a p-pass $\widetilde{O}(n/p)$-space algorithm for ACYC-T for any $p \geqslant 1$ (Theorem 32). This matches a p-pass $\Omega(n/p)$-space lower bound (up to logarithmic factors) for the problem, proven by [14]. We also provide a simpler proof of the lower bound (Theorem 31). Again, note that since our algorithm is deterministic and the lower bound holds for randomized algorithms, we fully settle the streaming complexity of the problem, even for multiple passes.

Finally, consider the sink (or equivalently, source) finding problem in DAGs.

SINK-DAG: Given a DAG $G = (V, E)$, find a sink node, i.e., a node with outdegree 0, in G.

We fully resolve the complexity of this problem by proving an $\Omega(n/p)$-space lower bound for any randomized p-pass algorithm (Theorem 33). This means that the trivial p-pass $O(n/p)$-space algorithm which, in the ith pass, considers the ith set of n/p nodes and checks whether any of them has outdegree zero, is optimal. Note that Henzinger et al. [29] gave an $\Omega(n/p)$ space lower bound for sink finding in general digraphs that may not be DAGs. The lower bound is significantly more challenging to prove when the input is promised to be a DAG (see Section 5.2 in the full version [25] for details). Further, our result establishes a gap between the complexities of SINK-DAG under general digraphs and tournaments: for SINK-DAG-T, [14] showed that $\widetilde{O}(n^{\Theta(1/p)})$-space is sufficient for p passes.

[4] It is, however, not hard to formulate a problem like T-EDGE (see Section 2), for which such a lower bound is intuitive and easy to prove. Here, we mean that no "textbook" or well-studied digraph problem was known to have such a lower bound.

Communication complexity and combinatorial graph theory. As a byproduct of our results, we resolve the communication complexity of multiple tournament problems (for their standard two-party communication versions) including REACH-T, STR-CONN-T, and SCC-DEC-T (see Section 6 in the full version [25] for details), which might be of independent interest. Very recently, Mande et al. [34] studied the communication complexity of several tournament problems; our results contribute to this study.

As further byproduct of our results, we make interesting observations on graph theoretic properties of tournaments that might be significant on their own. For instance, our results show that the indegree sequence of a tournament completely determines its SCCs. Again, for any two nodes u and v, indegree of u being smaller than that of v implies a directed path from u to v. See Section 6 in the full version [25] for a compilation of such facts that, to the best of our knowledge, have not been documented in the literature. They should find applications in future combinatorial analysis or algorithm design on tournaments.

1.3 Other Related Work

Above we discussed prior works that are most relevant to ours. Here, we give an account of other related results, going problem by problem.

For FAS-T, Coppersmith et al. [19] showed that simply sorting by indegree achieves a 5-approximation. Hence, this implies a deterministic single-pass semi-streaming algorithm for the same. Chakrabarti et al. [14] gave a one-pass $\widetilde{O}(n/\varepsilon^2)$-space algorithm for $(1+\varepsilon)$-approximate FAS-T in exponential time and a p-pass $\widetilde{O}(n^{1+1/p})$-space algorithm for 3-approximation in polynomial-time. Baweja et al. [8] improved the latter to a $(1+\varepsilon)$-approximation under the same space-pass tradeoff and polynomial time. They also established a space-time tradeoff for single-pass FAS-T algorithms. Guruswami, Velingker, and Velusamy [28] studied the dual of the FAS problem, namely the *maximum acyclic subgraph* (MAS) problem, in the streaming model and showed that an algorithm that obtains a better-than-(7/8) approximation for MAS-size requires $\Omega(\sqrt{n})$ space in a single pass. Velusamy, Singer, and Sudan [40] improved this result to show that better-than-1/2 approximation for MAS-size needs $\Omega(n)$ space. Assadi et al. [5] extended the lower bound to multiple passes, proving that p-pass $(1-\varepsilon)$-approximation algorithms for MAS-size requires $\Omega(n^{1-\varepsilon^{c/p}})$ space for some constant c. However, we do not know of any prior work that considered MAS specifically for streaming tournaments.

The ACYC problem was considered by [8] who gave a tight single-pass $\Omega(m \log(n^2/m))$ space lower bound for the problem, where m is the number of edges. They showed a similar lower bound for testing reachability from a single node to all other nodes in a general digraph. Note that the $\Omega(n^{2-o(1)})$-space lower bound for REACH that Chen at al. [17] proved for $O(\sqrt{\log n})$ passes, and Assadi and Raz [6] previously proved for 2 passes, also apply to ACYC, as well as to the FAS (any multiplicative approximation) and topological sorting problems. For topological sorting of random graphs drawn from certain natural distributions, [14] gave efficient random-order as well as adversarial-order streaming algorithms.

SINK-DAG-T was considered by Chakrabarti et al. [14] on random-order streams. They showed that the problem admits an exponential separation between such streams and adversarial-order streams: the former allows a polylog(n)-space algorithm in just a single pass, but the latter necessitates $\Omega(n^{1/p}/p^2)$ space for p passes. In a recent independent and parallel work, Mande, Paraashar, Sanyal, and Saurabh [34] studied sink finding[5] in (not

[5] They term it as *source* finding, which is equivalent to sink finding.

necessarily acyclic) tournaments in the two-party communication model and proved a tight bound of $\widetilde{\Theta}(\log^2 n)$ on its (unrestricted round) communication complexity. In contrast, our lower bound proof for SINK-DAG shows its communication complexity to be $\Omega(n)$. Thus, this demonstrates an exponential gap between the communication complexity of sink-finding under tournaments and general digraphs (even if they are promised to be DAGs).

Elkin [21] gave multipass streaming algorithms for the problem of computing exact shortest paths on general digraphs, which implies algorithms for STDIST. The SCC decomposition problem was studied by Laura and Santaroni [33] on arbitrary digraphs under W-streams (allowing streaming outputs), where they designed an algorithm using $O(n \log n)$ space and $O(n)$ passes in the worst case. Other notable works on streaming digraphs include algorithms for simulating random walks [38, 30, 18].

1.4 Presentation

Due to space restrictions, in this version we only present our main algorithm and elaborate on its analysis. For the remaining results, we mostly mention the theorem statements only. All detailed proofs and discussions appear in the full version [25].

2 Preliminaries

Notation and Terminology. All graphs in this paper are simple and directed. We typically denote a general digraph by $G = (V, E)$, where V is the set of vertices and $E \subseteq V \times V$ is the set of directed edges. We usually denote a tournament by $T = (V, E)$. Throughout the paper, n denotes the number of vertices of the graph in context. The notation $\widetilde{O}(f), \widetilde{\Omega}(f)$, and $\widetilde{\Theta}(f)$ hide factors polylogarithmic in f. The outdegree and indegree of a vertex v are denoted by $d_{\text{out}}(v)$ and $d_{\text{in}}(v)$ respectively. For a positive integer N, the set $[N] := \{1, \ldots, N\}$, and for integers $A \leq B$, the set $[A, B] := \{A, \ldots, B\}$. When we say $\{u, v\}$ is a *non-edge* in a digraph $G = (V, E)$, we mean that vertices $u, v \in V$ do not share any directed edge, i.e., $(u, v) \notin E$ and $(v, u) \notin E$. Again, we call a pair of edges $\{(u, v), (v, u)\}$ as *bidirected edges*.

The Graph Streaming Model. The input graph $G = (V, E)$ is presented as follows. The set of n nodes V is fixed in advance and known to the algorithm. The elements of E are inserted sequentially in a stream. The input graph and the stream order are chosen adversarially (worst case). We are allowed to make one or a few passes over the stream to obtain a solution. The goal is to optimize the space usage and the number of passes.

We use the following standard tool from the streaming literature.

▶ **Fact 1** (Sparse recovery [26, 20]). *There is a deterministic $O(k \cdot \text{polylog}(M, N))$-space algorithm that receives streaming updates to a vector $\mathbf{x} \in [-M, M]^N$ and, at the end of the stream, recovers $\mathbf{x}$ in polynomial time if $\mathbf{x}$ has at most k non-zero entries.*

The Communication Model. All our streaming lower bounds are proven via reductions from problems in communication complexity. In this paper, we use the two-player communication model of Yao [42]. Here, players Alice and Bob receive $\mathbf{x} \in \{0,1\}^N$ and $\mathbf{y} \in \{0,1\}^N$ respectively, and must send messages back and forth to compute $\mathcal{F}(\mathbf{x}, \mathbf{y})$ for some relation $\mathcal{F}$ defined on $\{0,1\}^N \times \{0,1\}^N$. We consider the randomized setting where the players can communicate based on private random strings. The *communication cost* of a randomized protocol for $\mathcal{F}$ is defined as the maximum number of bits exchanged by the players, over all possible inputs $(\mathbf{x}, \mathbf{y})$ and all possible random strings. The *randomized communication*

complexity of a relation $\mathcal{F}$, denoted by $\mathsf{R}(\mathcal{F})$, is defined as the minimum communication cost over all randomized protocols that, given *any* input $(\mathbf{x}, \mathbf{y})$, compute $\mathcal{F}(\mathbf{x}, \mathbf{y})$ correctly with probability at least $2/3$. Analogously, the *one-way randomized communication complexity* of $\mathcal{F}$, denoted by $\mathsf{R}^{\rightarrow}(\mathcal{F})$, is the minimum communication cost over all one-way communication protocols where Alice sends a single message to Bob, following which he must report the output.

We use classical communication problems such as Index (INDEX), Disjointness (DISJ), and Set Intersection (SET-INT) to prove our lower bounds. The problems are formally defined below.

INDEX$_N$: Alice holds a vector $\mathbf{x} \in \{0,1\}^N$, and Bob holds an index $i \in [N]$. The goal is to find $\mathbf{x}_i$, the ith bit of $\mathbf{x}$.

DISJ$_N$: Alice holds $\mathbf{x} \in \{0,1\}^N$ and Bob holds $\mathbf{y} \in \{0,1\}^N$. The goal is to decide whether $\mathbf{x}$ and $\mathbf{y}$ are *disjoint* as sets, i.e., output whether there exists an index $i \in [N]$ such that $\mathbf{x}_i = \mathbf{y}_i = 1$. We use the "promise version" (also known as *unique disjointness*) where we are promised that the sets are either disjoint or have a *unique* intersecting element i.

SET-INT$_N$: Alice holds $\mathbf{x} \in \{0,1\}^N$ and Bob holds $\mathbf{y} \in \{0,1\}^N$, where $\mathbf{x}$ and $\mathbf{y}$ intersect at exactly one index, i.e., there is a unique $i \in [N]$ such that $\mathbf{x}_i = \mathbf{y}_i = 1$. The goal is to output i.

We exploit the following known bounds on the randomized communication complexity of the above problems.

▶ **Fact 2** ([1]). $R^{\rightarrow}(\textit{INDEX}_N) = \Omega(N)$

▶ **Fact 3** ([37, 11]). $R(\textit{DISJ}_N) = \Omega(N) = R(\textit{SET-INT}_N)$

We also use the one-way randomized communication complexity of a variant of the INDEX problem called *Sparse Index* (SP-INDEX).

SP-INDEX$_{N,k}$: This is the INDEX$_N$ problem with the promise that Alice's vector $\mathbf{x}$ has at most k 1's, i.e., $|\{i : \mathbf{x}_i = 1\}| \leq k$.

▶ **Fact 4** ([13]). $R^{\rightarrow}(\textit{SP-INDEX}_{N,k}) = \Omega(k)$

For convenience, we define a new communication problem T-EDGE, which is basically a version of the Index problem.

T-EDGE$_n$: Alice holds a tournament $T = (V, E)$ with $|V| = n$, and Bob holds a pair of vertices $u, v \in V$. The goal is to find the orientation of the edge between u and v in T.

This problem is equivalent to INDEX$_N$ for $N = \binom{n}{2}$, and hence we get the following proposition.

▶ **Proposition 5.** $R^{\rightarrow}(\textit{T-EDGE}_n) = \Omega(n^2)$

3 Streaming Algorithms for SCC Decomposition and Applications

3.1 Finding the SCC-graph

First we design an algorithm for digraphs with zero non-edges and then show how to extend it to general digraphs.

3.1.1 Tournaments and Other Digraphs with No Non-Edges

We present a single-pass semi-streaming algorithm for SCC-DEC-T. In fact, our algorithm works for a broader class of digraphs: all digraphs without non-edges (i.e., every pair of vertices either shares exactly one directed edge or bidirected edges). Further, our algorithm solves the more general problem of finding the SCC-graph. Let G_{SCC} denote the SCC-graph of a digraph G. It is easy to see that the SCC-graph of any digraph is always a DAG. Hence, for a digraph G with no non-edge, its SCC-graph G_{SCC} is an acyclic *tournament*: between any two SCCs, there must exist an edge, but edges cannot exist in both directions. Since any acyclic tournament has a unique topological ordering (obtained by sorting its nodes by indegree), so does G_{SCC}. Thus, G_{SCC} can be simply and succinctly represented by a chain or permutation of G's SCCs. The edges of G_{SCC} are implicit: they go from left to right. Henceforth, we identify G_{SCC} with its topological ordering, and our algorithm outputs this permutation.

Formally, we prove the following theorem.

▶ **Theorem 6.** *Given an input digraph $G = (V, E)$ with no non-edge, there is a deterministic single-pass algorithm that uses $O(n \log n)$ bits of space and outputs a partition $\langle V_1, \ldots, V_\ell \rangle$ of V such that each V_i is an SCC of G, and for every pair $i, j \in [\ell]$ with $i < j$, all edges between V_i and V_j are directed from V_i to V_j.*

The algorithm is given in Algorithm 1. We prove its correctness using a series of lemmas and then prove the above theorem. For convenience, we define "SCC-cuts" that play an important role in the proof.

▶ **Definition 7** (SCC-cut)**.** *Given a digraph $G = (V, E)$ with no non-edge, for $U \subseteq V$, we call the cut $(U, V \setminus U)$ an* SCC-cut *if U is the union of a (non-empty) prefix of SCCs in G_{SCC}.*

The next three lemmas prove important properties of SCC-cuts.

▶ **Lemma 8.** *Given a digraph $G = (V, E)$ without any non-edge, a cut $(U, V \setminus U)$ is an SCC-cut if and only if every edge between U and $V \setminus U$ is oriented from U to $V \setminus U$ in G.*

▶ **Lemma 9.** *In a digraph $G = (V, E)$ with no non-edge, a cut $(U, V \setminus U)$ is an SCC-cut if and only if*

$$\sum_{u \in U} (d_{out}(u) - d_{in}(u)) = |U| \cdot |V \setminus U| \tag{1}$$

▶ **Lemma 10.** *Given a digraph $G = (V, E)$ without any non-edge and an SCC-cut $(U, V \setminus U)$ of G, we have*

$$\forall u \in U,\ v \in V \setminus U : \quad d_{in}(u) < d_{in}(v)$$

We are now ready to prove Theorem 6.

Proof of Theorem 6. First, note that Algorithm 1 stores only the indegree and the outdegree of each node, and hence the space usage is clearly $O(n \log n)$ bits. Thus, it only remains to prove the correctness, i.e., prove that the list it returns is indeed the (topological ordering of the) SCC-graph G_{SCC} of G.

We induct on the number of SCCs in G. Consider the base case when G has a single SCC. By Definition 7, G has only one SCC-cut $(V, \emptyset)$. Let L be the list obtained by sorting the nodes of G by indegree (as in line 9). The for-loop on line 10 goes over the vertices of G

Algorithm 1 A single-pass semi-streaming algorithm for finding SCC-graph of a digraph with no non-edge.

Input: Stream of edge insertions of an n-vertex digraph $G = (V, E)$ that has no non-edge

Initialize:

1: $d^-(v) \leftarrow 0$ for each $v \in V$ ▷ Indegree counters that count $d_{\text{in}}(v)$
2: $d^+(v) \leftarrow 0$ for each $v \in V$ ▷ Outdegree counters that count $d_{\text{out}}(v)$; not required if G is a tournament

Process (edge (u, v)):

3: $d^-(v) \leftarrow d^-(v) + 1$ ▷ Increase indegree of v
4: $d^+(u) \leftarrow d^+(u) + 1$ ▷ Increase outdegree of u; not required if G is a tournament

Post-processing:

5: $S \leftarrow \emptyset$ ▷ The current SCC
6: $G_{\text{SCC}} \leftarrow \emptyset$ ▷ The SCC-graph so far, stored as a list of SCCs in topological order
7: $n' \leftarrow n$ ▷ The size of the "remaining graph" $G \setminus G_{\text{SCC}}$
8: $c \leftarrow 0$ ▷ Initialize counter
9: $\langle v_1, \ldots, v_n \rangle \leftarrow$ vertices sorted such that $d^-(v_1) \leq \ldots \leq d^-(v_n)$ ▷ Ties broken arbitrarily
10: **for** $i = 1$ to n **do**:
11: Append v_i to S ▷ Add vertex to current SCC
12: $d^-(v_i) \leftarrow d^-(v_i) - (n - n')$ ▷ Calculate indegree of v_i in the remaining graph, removing edges from previous SCCs
13: $c \leftarrow c + d^+(v_i) - d^-(v_i)$ ▷ Update counter; if G is a tournament, $d^+(v_i) = n' - 1 - d^-(v_i)$
14: **if** $c = |S| \cdot (n' - |S|)$ **then**: ▷ Check if SCC is complete
15: Append S to G_{SCC} ▷ Add SCC to the SCC-graph
16: $n' \leftarrow n' - |S|$ ▷ Update size of remaining graph
17: $S \leftarrow \emptyset$; $c \leftarrow 0$ ▷ Empty S and reset counter
18: Output G_{SCC}

following the order in L. Consider any iteration $j < n$ of the for-loop. Assume that for all previous iterations, the if condition on line 14 hasn't been satisfied and n' has remained equal to n (which is true initially, i.e., for $j = 1$). Then, S now contains $\{v_1 \ldots, v_j\}$ due to line 11. Line 12 has had no effect on $d^-(v_i)$ for $i \in [j]$, and c now equals $\sum_{i=1}^{j} d_{\text{out}}(v_i) - d_{\text{in}}(v_i)$ due to line 13. Thus, the current count c is precisely $\sum_{u \in S}(d_{\text{out}}(u) - d_{\text{in}}(u))$ for the current set S. Since the set S contains a proper subset of V, the cut $(S, V \setminus S)$ cannot be an SCC-cut. Then, by Lemma 9, $c = \sum_{u \in S}(d_{\text{out}}(u) - d_{\text{in}}(u))$ cannot equal $|S| \cdot (n' - |S|)$ (note, $n' = n$). So the if condition on line 14 will not be satisfied. Hence, we will move on to the next iteration and the value of n' will remain n. Therefore, we will inductively add each node v_j to S and reach iteration n, where S contains all nodes, i.e., $S = V$. This means $(S, V \setminus S)$ is an SCC-cut, and by Lemma 9, c must equal $|S| \cdot (n' - |S|)$ now. Thus, the if condition on line 14 will be satisfied. Line 15 then adds $S = V$ to G_{SCC}, after which the algorithm terminates. Hence, it will correctly be the only set in the returned G_{SCC}. This proves the base case.

Next, assume by induction hypothesis that Algorithm 1 works correctly if the input digraph has ℓ SCC's for some $\ell \geqslant 1$. Now consider an input digraph G that has $\ell + 1$ SCCs, where $\langle V_1, \ldots, V_{\ell+1} \rangle$ represents G_{SCC}. By Lemma 10, we know that for all $u \in V_1$ and

$v \in V \setminus V_1$, $d_{\text{in}}(u) < d_{\text{in}}(v)$. Hence, the first $|V_1|$ vertices in the list L constitute V_1 and the for-loop on line 10 first goes over these vertices one by one. If we keep adding the nodes of V_1 to S, the cut $(S, V \setminus S)$ cannot be an SCC-cut until S equals the entire V_1. Hence, considering any iteration $j < |V_1|$ and applying analogous arguments as in the base case, it follows that the for-loop will inductively add the nodes of V_1 to S until we reach iteration $|V_1|$, when the condition on line 14 is satisfied. Then we add $S = V_1$ to G_{SCC} (line 15).

We now show that the remaining iterations ($i = |V| + 1$ to n) of the for-loop essentially recurses on the digraph $G \setminus V_1$. After the condition on line 14 is satisfied in iteration $|V_1|$, line 16 decrements n' by $|S| = |V_1|$, resulting in the correct size of the remaining graph $G \setminus V_1$. Line 17 resets S and c to their initial values. The remaining list $L \setminus V_1$ is in the correct order for $G \setminus V_1$ since each remaining vertex has its indegree dropped by the same value ($|V_1|$). The outdegrees $d^+(v)$ stay the same for each node v in the remaining graph since they had no outneighbor in V_1. Therefore, as we move to iteration $|V_1| + 1$, all the variables have their correct values for the recursion, except the variables $d^-(v)$. However, this is handled on the fly for each vertex v_i on line 12 when we decrement $d^-(v_i)$ by $(n - n')$. This is correct because when v_i is considered, its indegree has dropped from its initial value by exactly the total number of nodes in the "previous" SCC's removed from G (since each such node has exactly one edge to v_i). This number is $n - n'$ because the remaining subgraph has size n'.

Thus, the algorithm recursively runs on a graph with ℓ SCCs. By the induction hypothesis, it will correctly find $\langle V_2, \ldots, V_{\ell+1} \rangle$ as the SCC-graph of $G \setminus V_1$, which it will append after $\langle V_1 \rangle$. Thus, our output is $\langle V_1, \ldots, V_{\ell+1} \rangle$, which is indeed the SCC-graph of G. By induction, we conclude that given any digraph G without any non-edge, our algorithm correctly finds the SCC-graph G_{SCC}. ◀

▶ **Lemma 11.** *The post-processing time taken by Algorithm 1 is $O(n \log n)$.*

▶ Remark 12. Since tournaments form a subclass of digraphs with no non-edges, Theorem 6 applies to them in particular. However, multiple simplifications are possible for tournaments: (i) we do not even need to store both indegrees and outdegrees; just one of them, say the set of indegrees, is enough since we can derive the outdegree values from them. (ii) We can go over $i = 1$ to n and simply check whether $\sum_{j=1}^{i} d_{\text{in}}(v_j) = \binom{i}{2}$ to identify the first SCC (since the set of all edges incident on this SCC is precisely the set of edges contained in it) and then recurse as in Algorithm 1; this avoids the more involved analysis using SCC-cuts.

▶ **Corollary 13.** *There is a deterministic one-pass $O(n \log n)$-space algorithm for* SCC-DEC-T.

Observe that this space bound is optimal since $\Omega(n \log n)$ bits are needed to simply present the output of SCC-DEC-T.

3.1.2 Generalization to Arbitrary Digraphs

We extend our algorithm to get sublinear-space solutions for a broader class of graphs. These are *almost tournaments* which are $o(n^2)$ edges "away" from being a tournament. First, we formally define "k-closeness to tournaments".

▶ **Definition 14** (k-close to tournament)**.** *A digraph $G = (V, E)$ is called* k-close to tournament *if a total of at most k edges can be added to or deleted from G such that the resulting digraph is a tournament.*

In other words, a graph is k-close to tournament iff the total number of non-edges and bidirected edges it contains is at most k. We obtain the following result for such graphs.

▶ **Theorem 15.** *Given an input n-node digraph $G = (V, E)$ that is k-close to tournament, there is a deterministic single-pass $\widetilde{O}(n + k)$-space streaming algorithm that finds the SCC-graph G_{SCC}, represented by its topological ordering plus the set of all its non-edges.*

▶ **Corollary 16.** *Given an input n-node digraph $G = (V, E)$ that is k-close to tournament, there is a deterministic $\widetilde{O}(n + k)$-space streaming algorithm for* SCC-DEC.

A lower bound of $\Omega(n+k)$ space for SCC-DEC (even when we need just the vertex partition, and not necessarily the SCC-graph) follows immediately from the lower bound for STR-CONN (Theorem 22) that we prove in Section 3.2.

▶ **Corollary 17.** *Given any $k \leq \binom{n}{2}/2$, a single-pass streaming algorithm that solves* SCC-DEC *on any digraph that is k-close to tournament requires $\Omega(n + k)$ space.*

3.2 Reachability and Strong Connectivity

3.2.1 Tight Lower Bounds for Tournaments

Observe that Theorem 6 immediately implies single-pass semi-streaming algorithms for STR-CONN-T and REACH-T. Given the SCC-graph $\langle V_1, \ldots, V_\ell \rangle$ of T, a node t is reachable from another node s in T iff $i \leq j$, where V_i is the SCC containing s and V_j is the SCC containing t. Again, a tournament T is strongly connected iff the SCC-graph contains a single SCC. Thus, we get the following corollaries.

▶ **Corollary 18.** *There is a deterministic one-pass $O(n \log n)$-space algorithm for* REACH-T.

▶ **Corollary 19.** *There is a deterministic one-pass $O(n \log n)$-space algorithm for* STR-CONN-T.

We now prove the space bound for these problems to be tight (up to polylogarithmic factors) even for polylog(n) passes. Note that while our algorithms work for any digraph without non-edges, our lower bounds hold even for the easier versions where the inputs are promised to be tournaments.

▶ **Theorem 20.** *Any randomized p-pass algorithm solving* REACH-T *requires $\Omega(n/p)$ space.*

▶ **Theorem 21.** *Any randomized p-pass algorithm solving* STR-CONN-T *requires $\Omega(n/p)$ space.*

3.2.2 Tight Lower Bounds for Arbitrary Digraphs

Observe that Theorem 15 immediately implies an $\widetilde{O}(n + k)$-space algorithm for STR-CONN and REACH on digraphs that are k-close to tournaments. Here, we prove that the space bound is tight (up to polylogarithmic factors) for these problems.

▶ **Theorem 22.** *A single-pass streaming algorithm that solves* REACH *or* STR-CONN *on any digraph that is k-close to tournament requires $\Omega(n + k)$ space.*

Hence, the same lower bound applies to SCC-DEC, as we noted in Corollary 17. This demonstrates how the complexities of the SCC-DEC, STR-CONN, and REACH problems smoothly increase with the "distance" from the tournament property.

3.2.3 Improved and "Optimal" Time Complexity

Recall that our SCC-DEC algorithm for digraphs k-close to tournaments (Theorem 15) takes $\widetilde{O}(2^n)$ time. We show that this runtime can be improved for small k when solving STR-CONN and REACH (rather than the general SCC-DEC problem): our modified algorithms use the same space but take $\widetilde{O}(2^k \cdot n)$ time if $k < n$. Precisely, we prove the following theorem.

▶ **Theorem 23.** *Given an input n-node digraph $G = (V, E)$ that is k-close to a tournament, there are deterministic single-pass streaming algorithms that solve REACH and STR-CONN on G using $\widetilde{O}(n + k)$ space and $\widetilde{O}(2^{\min(k,n)})$ time for each problem.*

Subsequently, we show that the runtime of this algorithm is optimal in some sense, at least for STR-CONN.

▶ **Theorem 24.** *Suppose we have an unknown digraph G that is k-close to a tournament. Assume that we are given access to F, the set of G's non-edges, and an oracle $\mathcal{O}$ that, when queried with an orientation of edges corresponding to F, returns whether the resultant completion of G is strongly connected or not. Then, detecting whether G is strongly connected requires $\Omega(2^{\min(k,n)})$ queries to $\mathcal{O}$.*

3.3 Other Applications of SCC Decomposition of Tournaments

3.3.1 Hamiltonian Cycle

Using our SCC framework, we obtain the first streaming algorithm for HAM-CYCLE-T (see Section 1.2 for formal definition). We build on the parallel algorithm for HAM-CYCLE-T in [41] and adopt it in the streaming setting, obtaining the following theorem.

▶ **Theorem 25.** *There is a deterministic $O(\log n)$-pass $\widetilde{O}(n)$-space algorithm for HAM-CYCLE-T.*

3.3.2 Hamiltonian Path and Feedback Arc Set

Now we show applications of SCC-DEC-T to design algorithms for HAM-PATH-T and FAS-T.

▶ **Theorem 26.** *Given an input n-node tournament T whose largest SCC has size s, for any $p \geqslant 1$, there are deterministic $(p+1)$-pass $\widetilde{O}(ns^{1/p})$ space streaming algorithms for HAM-PATH-T and $(1+\varepsilon)$-approximate FAS-T. In particular, there are $O(\log s)$-pass semi-streaming algorithm for each problem.*

4 Lower Bounds for Feedback Arc Set and Shortest Distance on Tournaments

In this section, we investigate problems that remain hard on tournaments. In particular, we show that exactly solving FAS-T, FAS-SIZE-T, or STDIST requires storage of almost the entire graph. We also give a generalized space-approximation tradeoff for FAS-T.

4.1 Warm Up: A Lower Bound for Exact FAS-T

First we prove a lower bound for the exact version.

▶ **Lemma 27.** *Any randomized single-pass algorithm that exactly solving FAS-T needs $\Omega(n^2)$ space.*

4.2 A Generalized Lower Bound for Approximate FAS-T

Next we establish the following generalization.

▶ **Theorem 28.** *Given any $\varepsilon > 0$, a single-pass $(1+\varepsilon)$-approximation algorithm for* FAS-T *needs $\Omega(\min\{n^2, n/\sqrt{\varepsilon}\})$ space.*

We also show that solving exact FAS-SIZE-T requires $\Omega(n^2)$ space in a single pass. Note that, given an FAS ordering, we can find the corresponding number of back-edges in one more pass, but a single-pass lower bound for FAS-SIZE-T does not imply the same lower bound for FAS.

▶ **Theorem 29.** *Solving* FAS-SIZE-T *exactly in a single pass requires $\Omega(n^2)$ space.*

4.3 Shortest Distance

We proved that NP-hard problems like FAS and FAS-T need $\Omega(n^2)$ space in a single pass. Now we prove that classical polynomial-time solvable problems such as STDIST also need $\Omega(n^2)$ space on tournaments.

▶ **Theorem 30.** *Any randomized single-pass algorithm that exactly computes the distance between two input nodes in a tournament requires $\Omega(n^2)$ space.*

5 Resolving the Streaming Complexities of Acyclicity Testing and Sink Finding

Acyclicity testing is closely related to FAS problems, since lower bounds on acyclicity testing imply corresponding lower bounds on FAS approximation algorithms. In [14], the authors show that ACYC-T requires $\Omega(n/p)$ space in p passes. We first present a simpler proof of this lower bound. Then we give an algorithm matching the bound.

▶ **Theorem 31** ([14])**.** *Any randomized p-pass algorithm for acyclicty testing on tournaments requires $\Omega(n/p)$ space.*

▶ **Theorem 32.** *Given an input tournament T, for any $p \geqslant 1$, there is a deterministic p-pass $\widetilde{O}(n/p)$-space algorithm for detecting whether T is acyclic or not.*

Our final result is a tight lower bound for SINK-DAG.

▶ **Theorem 33.** *Any randomized p-pass algorithm that finds a sink node in a DAG requires $\Omega(n/p)$ space.*

References

1 Farid Ablayev. Lower bounds for one-way probabilistic communication complexity and their application to space complexity. *Theoretical Computer Science*, 175(2):139–159, 1996.

2 KookJin Ahn, Graham Cormode, Sudipto Guha, Andrew McGregor, and Anthony Wirth. Correlation clustering in data streams. In *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pages 2237–2246. PMLR, 2015.

3 Sepehr Assadi. Recent advances in multi-pass graph streaming lower bounds. *ACM SIGACT News*, 54(3):48–75, September 2023. doi:10.1145/3623800.3623808.

4 Sepehr Assadi, Arun Jambulapati, Yujia Jin, Aaron Sidford, and Kevin Tian. Semi-streaming bipartite matching in fewer passes and optimal space. In *Proceedings of the 2022 ACM-SIAM Symposium on Discrete Algorithms, SODA 2022, Virtual Conference / Alexandria, VA, USA, January 9 - 12, 2022*, pages 627–669. SIAM, 2022. doi:10.1137/1.9781611977073.29.

5 Sepehr Assadi, Gillat Kol, Raghuvansh R. Saxena, and Huacheng Yu. Multi-pass graph streaming lower bounds for cycle counting, max-cut, matching size, and other problems. In *61st IEEE Annual Symposium on Foundations of Computer Science, FOCS 2020, Durham, NC, USA, November 16-19, 2020*, pages 354–364. IEEE, 2020. doi:10.1109/FOCS46700.2020.00041.

6 Sepehr Assadi and Ran Raz. Near-quadratic lower bounds for two-pass graph streaming algorithms. In *61st IEEE Annual Symposium on Foundations of Computer Science, FOCS 2020, Durham, NC, USA, November 16-19, 2020*, pages 342–353. IEEE, 2020. doi:10.1109/FOCS46700.2020.00040.

7 Reuven Bar-Yehuda, Dan Geiger, Joseph (Seffi) Naor, and Ron M. Roth. Approximation algorithms for the feedback vertex set problem with applications to constraint satisfaction and bayesian inference. *SIAM Journal on Computing*, 27(4):942–959, 1998. doi:10.1137/S0097539796305109.

8 Anubhav Baweja, Justin Jia, and David P. Woodruff. An efficient semi-streaming PTAS for tournament feedback arc set with few passes. In *13th Innovations in Theoretical Computer Science Conference, ITCS 2022, January 31 - February 3, 2022, Berkeley, CA, USA*, volume 215 of *LIPIcs*, pages 16:1–16:23. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2022. doi:10.4230/LIPIcs.ITCS.2022.16.

9 Soheil Behnezhad, Moses Charikar, Weiyun Ma, and Li-Yang Tan. Single-pass streaming algorithms for correlation clustering. In Nikhil Bansal and Viswanath Nagarajan, editors, *Proceedings of the 2023 ACM-SIAM Symposium on Discrete Algorithms, SODA 2023, Florence, Italy, January 22-25, 2023*, pages 819–849. SIAM, 2023. doi:10.1137/1.9781611977554.CH33.

10 L.W. Beineke and R.J. Wilson. *Selected Topics in Graph Theory*. Number v. 1-3 in Selected Topics in Graph Theory. Academic Press, 1978.

11 Joshua Brody, Amit Chakrabarti, Ranganath Kondapally, David P. Woodruff, and Grigory Yaroslavtsev. Certifying equality with limited interaction. In *Proc. 18th International Workshop on Randomization and Approximation Techniques in Computer Science*, pages 545–581, 2014.

12 Paul Camion. Chemins et circuits hamiltoniens des graphes complets. *COMPTES RENDUS HEBDOMADAIRES DES SEANCES DE L ACADEMIE DES SCIENCES*, 249(21):2151–2152, 1959.

13 Amit Chakrabarti, Graham Cormode, Navin Goyal, and Justin Thaler. Annotations for sparse data streams. In *Proc. 25th Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 687–706, 2014.

14 Amit Chakrabarti, Prantar Ghosh, Andrew McGregor, and Sofya Vorotnikova. Vertex ordering problems in directed graph streams. In *Proceedings of the 2020 ACM-SIAM Symposium on Discrete Algorithms, SODA 2020, Salt Lake City, UT, USA, January 5-8, 2020*, pages 1786–1802. SIAM, 2020. doi:10.1137/1.9781611975994.109.

15 Amit Chakrabarti, Yaoyun Shi, Anthony Wirth, and Andrew C. Yao. Informational complexity and the direct sum problem for simultaneous message complexity. In *Proc. 42nd Annual IEEE Symposium on Foundations of Computer Science*, pages 270–278, 2001.

16 Pierre Charbit, Stéphan Thomassé, and Anders Yeo. The minimum feedback arc set problem is np-hard for tournaments. *Combinatorics, Probability & Computing*, 16:1–4, January 2007. doi:10.1017/S0963548306007887.

17 Lijie Chen, Gillat Kol, Dmitry Paramonov, Raghuvansh R. Saxena, Zhao Song, and Huacheng Yu. Almost optimal super-constant-pass streaming lower bounds for reachability. In *STOC '21: 53rd Annual ACM SIGACT Symposium on Theory of Computing, Virtual Event, Italy, June 21-25, 2021*, pages 570–583. ACM, 2021. doi:10.1145/3406325.3451038.

18 Lijie Chen, Gillat Kol, Dmitry Paramonov, Raghuvansh R. Saxena, Zhao Song, and Huacheng Yu. Near-optimal two-pass streaming algorithm for sampling random walks over directed graphs. In *48th International Colloquium on Automata, Languages, and Programming, ICALP 2021, July 12-16, 2021, Glasgow, Scotland (Virtual Conference)*, volume 198 of *LIPIcs*, pages 52:1–52:19. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2021. doi:10.4230/LIPICS.ICALP.2021.52.

19 Don Coppersmith, Lisa Fleischer, and Atri Rudra. Ordering by weighted number of wins gives a good ranking for weighted tournaments. *ACM Trans. Algorithms*, 6(3):55:1–55:13, 2010. doi:10.1145/1798596.1798608.

20 Abhik Kumar Das and Sriram Vishwanath. On finite alphabet compressive sensing. In *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 5890–5894, 2013. doi:10.1109/ICASSP.2013.6638794.

21 Michael Elkin. Distributed exact shortest paths in sublinear time. In *Proc. 49th Annual ACM Symposium on the Theory of Computing*, pages 757–770, 2017.

22 Joan Feigenbaum, Sampath Kannan, Andrew McGregor, Siddharth Suri, and Jian Zhang. On graph problems in a semi-streaming model. *Theor. Comput. Sci.*, 348(2–3):207–216, 2005. Preliminary version in *Proc. 31st International Colloquium on Automata, Languages and Programming*, pages 531–543, 2004.

23 S I Gass. Tournaments, transitivity and pairwise comparison matrices. *Journal of the Operational Research Society*, 49(6):616–624, 1998. doi:10.1057/palgrave.jors.2600572.

24 Xiubo Geng, Tie-Yan Liu, Tao Qin, Andrew Arnold, Hang Li, and Heung-Yeung Shum. Query dependent ranking using k-nearest neighbor. In *Proceedings of the 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '08, pages 115–122. Association for Computing Machinery, 2008. doi:10.1145/1390334.1390356.

25 Prantar Ghosh and Sahil Kuchlous. New algorithms and lower bounds for streaming tournaments. *CoRR*, abs/2405.05952, 2024. doi:10.48550/arXiv.2405.05952.

26 Anna C. Gilbert and Piotr Indyk. Sparse recovery using sparse matrices. *Proceedings of the IEEE*, 98(6):937–947, 2010.

27 Venkatesan Guruswami and Krzysztof Onak. Superlinear lower bounds for multipass graph processing. In *Proceedings of the 28th Conference on Computational Complexity, CCC 2013, K.lo Alto, California, USA, 5-7 June, 2013*, pages 287–298. IEEE Computer Society, 2013. doi:10.1109/CCC.2013.37.

28 Venkatesan Guruswami, Ameya Velingker, and Santhoshini Velusamy. Streaming complexity of approximating max 2csp and max acyclic subgraph. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, APPROX/RANDOM 2017, August 16-18, 2017, Berkeley, CA, USA*, volume 81 of *LIPIcs*, pages 8:1–8:19. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2017. doi:10.4230/LIPICS.APPROX-RANDOM.2017.8.

29 Monika R. Henzinger, Prabhakar Raghavan, and Sridhar Rajagopalan. Computing on data streams. *External memory algorithms*, pages 107–118, 1999.

30 Ce Jin. Simulating random walks on graphs in the streaming model. In *10th Innovations in Theoretical Computer Science Conference, ITCS 2019, January 10-12, 2019, San Diego, California, USA*, pages 46:1–46:15, 2019. doi:10.4230/LIPIcs.ITCS.2019.46.

31 Richard Karp. Reducibility among combinatorial problems. *Complexity of Computer Computations*, 40:85–103, 1972. doi:10.1007/978-3-540-68279-0_8.

32 Claire Kenyon-Mathieu and Warren Schudy. How to rank with few errors. In *Proceedings of the Thirty-Ninth Annual ACM Symposium on Theory of Computing*, STOC '07, pages 95–103. Association for Computing Machinery, 2007. doi:10.1145/1250790.1250806.

33 Luigi Laura and Federico Santaroni. Computing strongly connected components in the streaming model. In *Theory and Practice of Algorithms in (Computer) Systems*, pages 193–205. Springer Berlin Heidelberg, 2011.

34 Nikhil S. Mande, Manaswi Paraashar, Swagato Sanyal, and Nitin Saurabh. On the communication complexity of finding a king in a tournament. *ArXiv preprint*, abs/2402.14751, 2024. arXiv:2402.14751.

35 Andrew McGregor. Graph stream algorithms: a survey. *SIGMOD Record*, 43(1):9–20, 2014. doi:10.1145/2627692.2627694.

36 Andrew McGregor and Hoa T. Vu. Better streaming algorithms for the maximum coverage problem. In *20th International Conference on Database Theory, ICDT 2017, March 21-24, 2017, Venice, Italy*, volume 68 of *LIPIcs*, pages 22:1–22:18. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2017. doi:10.4230/LIPICS.ICDT.2017.22.

37 Alexander Razborov. On the distributional complexity of disjointness. *Theor. Comput. Sci.*, 106(2):385–390, 1992. Preliminary version in *Proc. 17th International Colloquium on Automata, Languages and Programming*, pages 249–253, 1990.

38 Atish Das Sarma, Sreenivas Gollapudi, and Rina Panigrahy. Estimating pagerank on graph streams. *J. ACM*, 58(3):13, 2011. doi:10.1145/1970392.1970397.

39 Michael Simpson, Venkatesh Srinivasan, and Alex Thomo. Efficient computation of feedback arc set at web-scale. *Proc. VLDB Endow.*, 10(3):133–144, 2016. doi:10.14778/3021924.3021930.

40 Noah Singer, Madhu Sudan, and Santhoshini Velusamy. Streaming approximation resistance of every ordering CSP. In Mary Wootters and Laura Sanità, editors, *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, APPROX/RANDOM 2021, August 16-18, 2021, University of Washington, Seattle, Washington, USA (Virtual Conference)*, volume 207 of *LIPIcs*, pages 17:1–17:19. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2021. doi:10.4230/LIPICS.APPROX/RANDOM.2021.17.

41 Danny Soroker. Fast parallel algorithms for finding hamiltonian paths and cycles in a tournament. *J. Algorithms*, 9(2):276–286, June 1988. doi:10.1016/0196-6774(88)90042-9.

42 Andrew C. Yao. Some complexity questions related to distributive computing. In *Proc. 11th Annual ACM Symposium on the Theory of Computing*, pages 209–213, 1979.