Graph Sketching and Streaming: New Approaches for Analyzing Massive Graphs[*]

Andrew McGregor

University of Massachusetts, Amherst
mcgregor@cs.umass.edu

Abstract. In this invited talk, we will survey some of the recent work on designing algorithms for analyzing massive graphs. Such graphs may not fit in main memory, may be distributed across numerous machines, and may change over time. This has motivated a rich body of work on analyzing graphs in the data stream model and the development of general algorithmic techniques, such as graph sketching, that can help minimize the space and communication costs required to process these massive graphs.

1 Motivation and Definitions

If you pick up your favorite algorithms textbooks and turn to a random page, there is a reasonable chance that you will find an algorithm for solving a graph problem. This is perhaps unsurprising given that graphs are a natural abstraction whenever you have information about a set of basic entities and the relationships between these entities, e.g., people and their friendships, web-pages and hyperlinks; neurons and synapses; or IP addresses and network flows. However, many of the classical algorithms for analyzing graphs implicitly assume that the graphs are static and fit in the main memory of a single machine. Unfortunately, in a growing number of applications this is not the case and attention has turned to algorithms that can process streams of graph data and/or graph data that is distributed across numerous machines. In these scenarios, standard graph techniques and primitives such as constructing BFS or DFS trees, dynamic programming, and linear programming are no longer applicable and new techniques, such as graph sketching, are required.

In the accompanying talk, we will survey some of the recent work on these new approaches to analyzing massive graphs. In this document, we first present some of the relevant definitions and then collect together references for some of the main results that will be discussed.

Basic Definitions. The simplest version of the data stream model for processing graphs is the *insert-only model.* In this model, the input stream consists of a sequence of unordered pairs $e = \{u, v\}$ where $u, v \in [n]$. Such a stream,

$$S = \langle e_1, e_2, \ldots, e_m \rangle$$

[*] This work was supported in part by NSF Awards CCF-0953754, IIS-1251110, CCF-1320719, CCF-1637536, and a Google Research Award.

naturally defines an undirected graph $G = (V, E)$ where $V = [n]$ and $E = \{e_1, \ldots, e_m\}$. The goal is to design an algorithm that solves the required graph problem on the graph G while only accessing the input sequentially and using memory that is sublinear in m.

A natural extension of the model is the *insert-delete model* in which edges can be both inserted and deleted. In this case, the input is a sequence

$$S = \langle a_1, a_2, \ldots \rangle \quad \text{where} \quad a_i = (e_i, \Delta_i)$$

where e_i encodes an undirected edge as before and $\Delta_i \in \{-1, 1\}$. The multiplicity of an edge e is defined as $f_e = \sum_{i:e_i=e} \Delta_i$ and we typically restrict our attention to the case where $f_e \in \{0, 1\}$ for all edges e. Both the insert-only and insert-delete model can be extended to handle weighted graphs where the occurrence of each edge e in the input is replaced by the pair (e, w_e) where w_e indicates the weight of the edge.

An important algorithmic technique in the insert-delete model in particular, is that of *graph sketching.* A sketch is a random linear projection of a vector corresponding to the input data. In the context of graphs, this vector would be

$$\mathbf{f} \in \{0, 1\}^{\binom{n}{2}}$$

where entries correspond to the current f_e values and the sketch would be $M\mathbf{f} \in \mathbb{R}^d$ where $d \ll n^2$ is the dimension of the sketch and M is a random matrix chosen according to an appropriate distribution, i.e., one from which the relevant properties of $\mathbf{f}$ can be inferred given $M\mathbf{f}$. Such a sketch can then be used as the basis for a data stream algorithm since $M\mathbf{f}$ can be computed incrementally: when (e, Δ) arrives in the stream we can update $M\mathbf{f}$ as follows:

$$M\mathbf{f} \leftarrow M\mathbf{f} + \Delta \cdot M^e$$

where M^e is the eth column of M. Hence, it suffices to store the current sketch and any random bits needed to compute the matrix M. The main challenge is therefore to design low-dimensional sketches as this results in small-space algorithms. Sketches are also useful for the purpose of reducing communication in various distributed models since if each machine communicates a sketch of their local input, then a sketch of the entire data set can be recovered simply by adding together the individual sketches.

2 Some Results and References

In this section, we briefly summarize some of the main results in the area. This is not intended to be an exhaustive survey and we focus on the most representative or most recent results on each problem. Further details of some of these algorithms can also be found in the survey [30] although many of the results postdate that survey.

Connectivity and Sparsification. One of the most basic graph problems is determining whether a graph is connected. This and many related problems can be solved relatively easily in the insert-only model using $O(n \operatorname{polylog} n)$ space, e.g., to test connectivity when there are no edge deletions, it suffices to keep track of the connected components of the graph. Furthermore, it can be shown that $\Omega(n \log n)$ space is necessary to solve this problem [37]. A more surprising result is that $O(n \operatorname{polylog} n)$ space also suffices in the insert-delete model [3]; this was one of the first applications of the graph sketching technique. Furthermore, the basic algorithm can be extended to testing k-edge connectivity [4] and approximate testing of k-node connectivity [20] using $O(kn \operatorname{polylog} n)$ space. Lastly, in $O(\epsilon^{-2} n \operatorname{polylog} n)$ space it is possible to construct combinatorial and spectral sparsifiers of the input graph [20, 28]; these allow the size of all cuts to be approximated up to a $1+\epsilon$ factor along with various properties related to the Laplacian of the graph.

Matching. Most of the work on approximating maximum matchings has focused on the insert-only model. The trivial greedy approach yields a 2-approximation using $O(n \log n)$ space in the unweighted case and after a long sequence of papers, a $(2+\epsilon)$-approximation algorithm using $O(\epsilon^{-1} n \log n)$ space in the weighted case is now known [19, 35]. The best known lower bound is that no algorithm can beat a factor $e/(e-1) \approx 1.58$ while using only $O(n \operatorname{polylog} n)$ space [25] and closing the gap remains an open problem. Better approximation guarantees or lower space requirements are possible if the algorithm may take a small number of additional passes over the data stream [2, 17, 23, 29] or if the edges of the graph are assumed to arrive in a random order [26, 29]. Another line of work considers low arboricity graphs, e.g., the size of the maximum matching in a planar graph can be approximated up to a $(5+\epsilon)$ factor using $O(\epsilon^{-2} \log n)$ space [15, 32].

In the insert-delete model, it is known that $\Theta(n^2/\alpha^3 \cdot \operatorname{polylog} n)$ space is necessary and sufficient to find a matching that is at least $1/\alpha$ times the size of the maximum matching [8, 13]. This can be reduced to $O(n^2/\alpha^4 \cdot \operatorname{polylog} n)$ space if we are only interested in estimating the size of the maximum matching [7]. Furthermore, if the size of the maximum matching is bounded by k, then $\Theta(k^2 \operatorname{polylog} n)$ space is necessary and sufficient to find a matching of maximum size [11, 13].

And more... Other graph problems considered in the data stream model include finding the densest subgraph [10,18,31]; correlation clustering [1]; counting triangles [9,24,33], estimating the size of the maximum cut [27], finding large independent sets and cliques [14, 21], and performing random walks [36]. Some of the above problems have also been considered in the *sliding window model*, a variant of the insert-only model in which the relevant graph is defined by only the most recent edges [16]. Another notable body of related work considers problems that can be described in terms of hypergraphs, i.e., every edge in the stream includes an arbitrary number of nodes rather than just two. Such problems include minimum set cover [5,6,12,22], maximum coverage [5,34], and minimum hitting set [13].

References

1. Ahn, K.J., Cormode, G., Guha, S., McGregor, A., Wirth, A.: Correlation clustering in data streams. In: International Conference on Machine Learning. pp. 2237–2246 (2015)
2. Ahn, K.J., Guha, S.: Linear programming in the semi-streaming model with application to the maximum matching problem. Inf. Comput. 222, 59–79 (2013)

3. Ahn, K.J., Guha, S., McGregor, A.: Analyzing graph structure via linear measurements. In: ACM-SIAM Symposium on Discrete Algorithms. pp. 459–467 (2012)
4. Ahn, K.J., Guha, S., McGregor, A.: Graph sketches: sparsification, spanners, and subgraphs. In: ACM Symposium on Principles of Database Systems. pp. 5–14 (2012)
5. Assadi, S.: Tight Space-Approximation Tradeoff for the Multi-Pass Streaming Set Cover Problem. ArXiv e-prints (Mar 2017)
6. Assadi, S., Khanna, S., Li, Y.: Tight bounds for single-pass streaming complexity of the set cover problem. In: ACM Symposium on Theory of Computing. pp. 698–711 (2016)
7. Assadi, S., Khanna, S., Li, Y.: On estimating maximum matching size in graph streams. In: ACM-SIAM Symposium on Discrete Algorithms. pp. 1723–1742 (2017)
8. Assadi, S., Khanna, S., Li, Y., Yaroslavtsev, G.: Maximum matchings in dynamic graph streams and the simultaneous communication model. In: ACM-SIAM Symposium on Discrete Algorithms. pp. 1345–1364 (2016)
9. Bera, S.K., Chakrabarti, A.: Towards tighter space bounds for counting triangles and other substructures in graph streams. In: Symposium on Theoretical Aspects of Computer Science. pp. 11:1–11:14 (2017)
10. Bhattacharya, S., Henzinger, M., Nanongkai, D., Tsourakakis, C.E.: Space- and time-efficient algorithm for maintaining dense subgraphs on one-pass dynamic streams. In: ACM Symposium on Theory of Computing. pp. 173–182 (2015)
11. Bury, M., Schwiegelshohn, C.: Sublinear estimation of weighted matchings in dynamic data streams. In: European Symposium on Algorithms. pp. 263–274 (2015)
12. Chakrabarti, A., Wirth, A.: Incidence geometries and the pass complexity of semi-streaming set cover. In: ACM-SIAM Symposium on Discrete Algorithms. pp. 1365–1373 (2016)
13. Chitnis, R., Cormode, G., Esfandiari, H., Hajiaghayi, M., McGregor, A., Monemizadeh, M., Vorotnikova, S.: Kernelization via sampling with applications to finding matchings and related problems in dynamic graph streams. In: ACM-SIAM Symposium on Discrete Algorithms. pp. 1326–1344 (2016)
14. Cormode, G., Dark, J., Konrad, C.: Independent set size approximation in graph streams. CoRR abs/1702.08299 (2017), `http://arxiv.org/abs/1702.08299`
15. Cormode, G., Jowhari, H., Monemizadeh, M., Muthukrishnan, S.: The sparse awakens: Streaming algorithms for matching size estimation in sparse graphs. CoRR abs/1608.03118 (2016), `http://arxiv.org/abs/1608.03118`
16. Crouch, M.S., McGregor, A., Stubbs, D.: Dynamic graphs in the sliding-window model. In: European Symposium on Algorithms. pp. 337–348 (2013)
17. Esfandiari, H., Hajiaghayi, M., Monemizadeh, M.: Finding large matchings in semi-streaming. In: IEEE International Conference on Data Mining Workshops. pp. 608–614 (2016)
18. Esfandiari, H., Hajiaghayi, M., Woodruff, D.P.: Brief announcement: Applications of uniform sampling: Densest subgraph and beyond. In: ACM Symposium on Parallel Algorithms and Architectures. pp. 397–399 (2016)
19. Ghaffari, M.: Space-optimal semi-streaming for $(2 + \epsilon)$-approximate matching. CoRR abs/1701.03730 (2017), `http://arxiv.org/abs/1701.03730`
20. Guha, S., McGregor, A., Tench, D.: Vertex and hyperedge connectivity in dynamic graph streams. In: ACM Symposium on Principles of Database Systems. pp. 241–247 (2015)
21. Halldórsson, B.V., Halldórsson, M.M., Losievskaja, E., Szegedy, M.: Streaming algorithms for independent sets. In: International Colloquium on Automata, Languages and Programming. pp. 641–652 (2010)
22. Har-Peled, S., Indyk, P., Mahabadi, S., Vakilian, A.: Towards tight bounds for the streaming set cover problem. In: ACM Symposium on Principles of Database Systems. pp. 371–383 (2016)

23. Kale, S., Tirodkar, S., Vishwanathan, S.: Maximum matching in two, three, and a few more passes over graph streams. CoRR abs/1702.02559 (2017), `http://arxiv.org/abs/1702.02559`
24. Kallaugher, J., Price, E.: A hybrid sampling scheme for triangle counting. In: ACM-SIAM Symposium on Discrete Algorithms. pp. 1778–1797 (2017)
25. Kapralov, M.: Better bounds for matchings in the streaming model. In: ACM-SIAM Symposium on Discrete Algorithms. pp. 1679–1697 (2013)
26. Kapralov, M., Khanna, S., Sudan, M.: Approximating matching size from random streams. In: ACM-SIAM Symposium on Discrete Algorithms. pp. 734–751 (2014)
27. Kapralov, M., Khanna, S., Sudan, M., Velingker, A.: $1+\Omega(1)$-approximation to MAX-CUT requires linear space. In: ACM-SIAM Symposium on Discrete Algorithms. pp. 1703–1722 (2017)
28. Kapralov, M., Lee, Y.T., Musco, C., Musco, C., Sidford, A.: Single pass spectral sparsification in dynamic streams. In: IEEE Symposium on Foundations of Computer Science. pp. 561–570 (2014)
29. Konrad, C., Magniez, F., Mathieu, C.: Maximum matching in semi-streaming with few passes. In: APPROX-RANDOM. pp. 231–242 (2012)
30. McGregor, A.: Graph stream algorithms: a survey. SIGMOD Record 43(1), 9–20 (2014)
31. McGregor, A., Tench, D., Vorotnikova, S., Vu, H.T.: Densest subgraph in dynamic graph streams. In: Mathematical Foundations of Computer Science. pp. 472–482 (2015)
32. McGregor, A., Vorotnikova, S.: A note on logarithmic space stream algorithms for matchings in low arboricity graphs. CoRR abs/1612.02531 (2016), `http://arxiv.org/abs/1612.02531`
33. McGregor, A., Vorotnikova, S., Vu, H.T.: Better algorithms for counting triangles in data streams. In: ACM Symposium on Principles of Database Systems. pp. 401–411 (2016)
34. McGregor, A., Vu, H.T.: Better streaming algorithms for the maximum coverage problem. In: International Conference in Database Theory. pp. 22:1–22:18 (2017)
35. Paz, A., Schwartzman, G.: A $(2+\epsilon)$-approximation for maximum weight matching in the semi-streaming model. In: ACM-SIAM Symposium on Discrete Algorithms. pp. 2153–2161 (2017)
36. Sarma, A.D., Gollapudi, S., Panigrahy, R.: Estimating pagerank on graph streams. J. ACM 58(3), 13 (2011)
37. Sun, X., Woodruff, D.P.: Tight bounds for graph problems in insertion streams. In: International Workshop on Approximation Algorithms for Combinatorial Optimization Problems. pp. 435–448 (2015)