Available online at www.sciencedirect.com
SCIENCE DIRECT®
Theoretical Computer Science 348 (2005) 207–216
Theoretical Computer Science
www.elsevier.com/locate/tcs

On graph problems in a semi-streaming model☆

Joan Feigenbaum[a,1,2], Sampath Kannan[b,2,3], Andrew McGregor[b,*,2,4], Siddharth Suri[b,2,5], Jian Zhang[a,2,6]

[a] *Department of Computer Science, Yale University, New Haven, CT 06520, USA*
[b] *Department of Computer and Information Science, University of Pennsylvania, Philadelphia, PA 19104, USA*

Abstract

We formalize a potentially rich new streaming model, the *semi-streaming model*, that we believe is necessary for the fruitful study of efficient algorithms for solving problems on massive graphs whose edge sets cannot be stored in memory. In this model, the input graph, $G = (V, E)$, is presented as a stream of edges (in adversarial order), and the storage space of an algorithm is bounded by $O(n \cdot \text{polylog } n)$, where $n = |V|$. We are particularly interested in algorithms that use only one pass over the input, but, for problems where this is provably insufficient, we also look at algorithms using constant or, in some cases, logarithmically many passes. In the course of this general study, we give semi-streaming constant approximation algorithms for the unweighted and weighted matching problems, along with a further algorithmic improvement for the bipartite case. We also exhibit $\log n / \log \log n$ semi-streaming approximations to the diameter and the problem of computing the distance between specified vertices in a weighted graph. These are complemented by $\Omega(\log^{(1-\varepsilon)} n)$ lower bounds.

Keywords: Graph; Streaming; Matching; Spanner; Girth; Articulation point

1. Introduction

Streaming [14,2,10] is an important model for computation on massive data sets. Recently, there has been a large body of work on designing algorithms in this model [11,2,10,15,13,12]. Yet, the problems considered fall into a small number of categories, such as computing statistics, norms, and histograms. Very few graph problems [4] have been considered in the streaming model.

☆ This work was supported by the DoD University Research Initiative (URI) administered by the Office of Naval Research under Grant N00014-01-1-0795.

* Corresponding author.
E-mail addresses: feigenbaum-joan@cs.yale.edu (J. Feigenbaum), kannan@cis.upenn.edu (S. Kannan), andrewm@cis.upenn.edu (A. McGregor), ssuri@gradient.cis.upenn.edu (S. Suri), zhang-jian@cs.yale.edu (J. Zhang).

[1] Supported in part by ONR Grant N00014-01-1-0795 and NSF Grant ITR-0331548.
[2] Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.
[3] Supported in part by ARO Grant DAAD 19-01-1-0473 and NSF Grant CCR-0105337.
[4] Supported in part by NSF grant ITR-0205456.
[5] Supported in part by NIH Grant T32 HG000046-05.
[6] Supported by ONR Grant N00014-01-1-0795 and NSF Grants CCR-0105337 and ITR-0331548.

0304-3975/$ - see front matter © 2005 Published by Elsevier B.V.
doi:10.1016/j.tcs.2005.09.013

The difficulty of graph problems in the streaming model arises from the memory limitation of the model combined with input-access constraints. We can view the amount of memory used by algorithms with sequential (one-way) input access as a spectrum. At one end of the spectrum, we have dynamic algorithms [9] that may use memory enough for the whole input. At the other end, we have streaming algorithms that use only polylog space. At one extreme, there is a lot of work on dynamic graph problems; on the other, general graph problems are considered hard in the (poly)log-space streaming model. Recently, it has been suggested by Muthukrishnan [19] that the middle ground, where the algorithms can use $O(n \cdot \text{polylog } n)$ bits of space, is an interesting and open area. This is the area that we explore.

Besides taking a middle position in the memory-size spectrum, the semi-streaming model allows multiple passes over the input stream. In certain applications with massive data sets, a small number of sequential passes over the data would be much more efficient than many random accesses to the data. Only a few works [8,6] have considered the multiple-pass model and a lot remains to be done.

Massive graphs arise naturally in many real-world scenarios. Two examples are the *call graph*, where nodes correspond to telephone numbers and edges to calls between numbers that call each other during some time interval, and the *web graph*, where nodes are web pages, and the edges are links between pages. The streaming model is necessary for the study of the efficient processing of such massive graphs. In [1], the authors introduce the semi-external model for computations on massive graphs, i.e., one in which the vertex set can be stored in memory, but the edge set cannot. However, this work addresses the problems in an external memory model in which random access to the edges, although expensive, is allowed. This is a major difference between their model and ours. Indeed, the authors of [18] argue that one of the major drawbacks of standard graph algorithms, when applied to massive graphs such as the web, is their need to have random access to the edge set. Furthermore, there are situations in which the graph is revealed in a streaming fashion, such as a web crawler exploring the web graph.

We consider a set of classical graph problems in this semi-streaming model. We show that, although the computing power of this model is still limited, there are semi-streaming algorithms for a variety of graph problems. Our main result is a semi-streaming algorithm that computes a $(\frac{2}{3} - \varepsilon)$-approximation in $O((\log(1/\varepsilon))/\varepsilon)$ passes for unweighted bipartite graph matching. We also provide a one-pass semi-streaming algorithm for $\frac{1}{6}$-approximating the maximum weighted graph matching. We also provide $\log n / \log\log n$ approximations for diameter and shortest paths in weighted graphs which we complement with $\Omega(\log^{(1-\varepsilon)} n)$ lower bounds for these problems in unweighted graphs.

2. Preliminaries

Unless stated otherwise, we denote by $G(V, E)$ a graph G with vertex set $V = \{v_1, v_2, \ldots, v_n\}$ and edge set $E = \{e_1, e_2, \ldots, e_m\}$. Note that n is the number of vertices and m the number of edges.

Definition 1. A *graph stream* is a sequence of edges $e_{i_1}, e_{i_2}, \ldots, e_{i_m}$, where $e_{i_j} \in E$ and $i_1, i_2, \ldots, i_m$ is an arbitrary permutation of $[m] = \{1, 2, \ldots, m\}$.

While an algorithm goes through the stream, the graph is revealed one edge at a time. This definition generalizes the streams of graphs in which the adjacency matrix or the adjacency list is presented as a stream. In a stream in the adjacency-matrix or adjacency-list models, the edges incident to each vertex are grouped together. We need the more general model to account for graphs such as call graphs where the edges might generated in any order.

The efficiency of a graph algorithm in the semi-streaming model is measured by the space it uses, the time it requires to process each edge, and the number of passes it makes over the graph stream.

Definition 2. A *semi-streaming graph algorithm* computes over a graph stream using $S(n, m)$ bits of space. The algorithm may access the input stream in a sequential order(one-way) for $P(n, m)$ passes and use $T(n, m)$ time to process each edge. It is required that $S(n, m)$ be $O(n \cdot \text{polylog } (n))$ bits.

To see the limitation of the (poly)log-space streaming model for graph problems, consider the following simple problem. Given a graph, determining whether there is a length-2 path between two vertices, x and y, is equivalent to deciding whether two vertex sets, the neighborhood of x and the neighborhood of y have a non-empty intersection. Because set disjointness has linear-space communication complexity [16], the length-2 path problem is impossible in