Theoretical Computer Science 407 (2008) 302–309

Contents lists available at ScienceDirect

Theoretical Computer Science

journal homepage: www.elsevier.com/locate/tcs

New results for finding common neighborhoods in massive graphs in the data stream model

A.L. Buchsbaum [a], R. Giancarlo [b,*], B. Racz [c,d]

[a] *Madison, NJ, USA*

[b] *University of Palermo, Dipartimento di Matematica ed Applicazioni, Via Archirafi 34, 90123 Palermo, Italy*

[c] *Computer and Automation Research Institute of the Hungarian Academy of Sciences (MTA SZTAKI), P.O. Box 63, Budapest, H-1518, Hungary*

[d] *Department of Algebra, Budapest University of Technology and Economics, P.O. Box 91, Budapest, H-1521, Hungary*

ARTICLE INFO

Article history:
Received 17 September 2007
Received in revised form 26 February 2008
Accepted 12 June 2008
Communicated by G. Ausiello

Keywords:
Graph algorithms for data streams
Extremal graph theory
Communication complexity
Space lower bounds

ABSTRACT

We consider the problem of finding pairs of vertices that share large common neighborhoods in massive graphs. We give lower bounds for randomized, two-sided error algorithms that solve this problem in the data-stream model of computation. Our results correct and improve those of Buchsbaum, Giancarlo, and Westbrook [On finding common neighborhoods in massive graphs, Theoretical Computer Science, 299 (1–3) 707–718 (2004)]

1. Introduction

We study the problem of finding pairs of vertices with large common neighborhoods in directed graphs. We consider the space complexity of the problem in the data-stream model proposed by Henzinger, Raghavan, and Sajagopalan [13]. In this model of computation, the input arrives as a sequence of elements (for a graph, e.g., a sequence of arcs). Complexity is measured in terms of the number of times an algorithm can scan the input (in order) and the amount of space it requires to store intermediate results. Buchsbaum, Giancarlo, and Westbrook [4] claimed results for common neighborhood problems (defined below) in these models, but some of their lower-bound proofs are incorrect. We present improved results that rectify these issues.

The motivation for studying such problems in data-stream models remains unchanged from the earlier paper [4], and we use it here verbatim. Many large-scale systems generate massive sequences of data: records of telephone calls in a voice network [6,14], transactions in a credit card network [5,20], alarms signals from network monitors [17,21], etc. From a practical standpoint, many applications require real-time decision making based on current information: e.g., fraud and intrusion detection [5,6,20] and fault recovery [17,21]. Data must be analyzed as they arrive, not off-line after being stored in a central database. From a theoretical (as well as practical) standpoint, processing and integrating the massive amounts of data generated by a myriad of continuously operating sources poses many problems. For example, external memory algorithms [23] are motivated by the fact that many classical algorithms do not scale when data sets do not fit in main memory. At some point, however, data sets become so large as to preclude most computations that require more than one scan of the data, as they stream by, without the ability to recall arbitrary pieces of input previously encountered.

* Corresponding author.
E-mail addresses: alb@adambuchsbaum.com (A.L. Buchsbaum), raffaele@math.unipa.it (R. Giancarlo), bracz@math.bme.hu (B. Racz).

0304-3975/$ – see front matter © 2008 Elsevier B.V. All rights reserved.
doi:10.1016/j.tcs.2008.06.056

Common neighborhoods represent a natural, basic relationship between pairs of vertices in a graph. In transactional data like telephone calls and credit card purchases, common neighborhoods indicate users with shared interests (such as who they call or what they buy); inverted, they also represent market-basket information [8,12,22] (e.g., which products tend to be purchased together). In graphs representing relationships such as hyperlinks in the World Wide Web or citations by articles in a scientific database, common neighborhoods can yield clues about authoritative sources of information [15] or seminal items of general interest [13].

Informally, we show that for infinitely many values of n and c, any $O(1)$-pass, randomized (two-sided error) data-stream algorithm that determines if *any* two vertices in a directed, n-vertex graph have more than c common neighbors requires $\Omega(\sqrt{c}n^{3/2})$ bits of space. The definitions in Section 2 formalize the problems, and the results are formally presented in Theorems 5, 6, 8 and 9. In addition to using reductions from communication complexity, we also use results from extremal graph theory to prove our claims.

2. Preliminaries

2.1. Data-stream models

In the *k-pass data-stream model*, an algorithm $\mathcal{A}$ accesses a one-way, read-only *input tape*, a two-way, read-write *work tape*, and a one-way, write-only *output tape*. $\mathcal{A}$ is allowed an arbitrary amount of internal computation (albeit restricted to use the tapes for input, working memory, and output) as well as an arbitrary number of the standard operations on any of the tapes: read or write the symbol under the head, and move the head to the next position. $\mathcal{A}$ is also allowed $k-1$ *rewind* operations on the input tape, each of which resets the head to point at the first symbol on the tape. The space required is the length of the work tape in the worst-case over all possible inputs. A *randomized k-pass data-stream algorithm* $\mathcal{A}$ can also access a one-way, read-only *random tape*, which contains an infinitely long string of random bits, and must report the correct answer with probability $1-\epsilon$ for a given ϵ. The space required is the length of the work tape in the worst-case over all possible inputs and random tapes.

These models were formalized by Henzinger, Raghavan, and Sajagopalan [13], who considered some neighborhood and connectivity problems in directed graphs. Variations now underly a substantial literature in streaming algorithms, aptly surveyed by Muthukrishnan [18].

2.2. Common neighborhoods

Let $G=(V,A)$ be a directed graph. In what follows, $n=|V|$ and $m=|A|$. For each vertex a, define $N(a)=\{b:(a,b)\in A\}$; we call each $b\in N(a)$ a *neighbor* of a. Also define $deg(a)=|N(a)|$, the *out-degree* of a. Given two vertices a and b, define $N(\{a,b\})=N(a)\cap N(b)$; we call $N(\{a,b\})$ the *common neighborhood* of a and b.

Define $CN(G)=\{\{u,v\}:|N(\{u,v\})|\geq T(\{u,v\})\}$, where $T(\{u,v\})$ is a given threshold function, which may depend on u and v. Given the threshold function, we wish to find $CN(G)$. Since we are primarily interested in lower bounds, we concentrate on variations with uniform thresholds, in particular the following.

(1) For all $u,v\in V$, $T(\{u,v\})=c$, for some $c\in[1,n-1]$.
(2) For all $u,v\in V$, $T(\{u,v\})=\alpha\min(|N(u)|,|N(v)|)$, where $0<\alpha\leq 1$.

Also due to our focus on lower bounds, we consider only the corresponding decision problem of determining if $|CN(G)|\geq\tau$, for some integer parameter τ. When $\tau=1$, solving the decision problem determines whether there exists *any* pair of vertices whose common neighborhood size is greater than the given threshold function; we call this the *non-emptiness query* or the *non-emptiness problem*.

Define $f^{\epsilon}_{\mathrm{CN}}(n,c)$ to be the space required for a randomized data-stream algorithm to answer the non-emptiness query on an n-vertex graph, for $T(\{u,v\})=c$, with probability $1-\epsilon$ of being correct. We assume that the input graph G is given as an adjacency list; i.e., as a sequence of the form $\{(a_1,N(a_1));\ (a_2,N(a_2));\ \ldots;\ (a_n,N(a_n))\}$, for some arbitrary ordering of the vertices in V. Because any adjacency list corresponds to an edge list, while edge lists must be sorted to obtain adjacency lists, this assumption provides more powerful lower bounds than those assuming edge-list inputs.

2.3. Communication complexity

We use reductions from two basic problems in communication complexity. Henzinger, Raghavan, and Rajagopalan [13] also used these problems to lower bound several data-stream problems on graph algorithms. See Nisan and Kushilevitz [19] for a general introduction to communication complexity.

In the following problems there are two players, A and B; their goal is to compute a function value $f(x,y)$ with A having parameter x and B having parameter y. Thus they need to communicate. In a *one-round protocol*, A sends B a single message (sequence of bits), after which B must compute $f(x,y)$. In a *multi-round protocol*, A and B alternate the transmission of messages, and the protocol ends when one of them computes $f(x,y)$. The cost of a protocol is the total number of bits transmitted over all rounds in the worst case over all possible input pairs (x,y).

- In the *bit-vector index problem*, denoted IND, player A has an n-bit vector x, player B has an index $y \in \{1, 2, \ldots, n\}$, and the function to compute is $f(x, y) = x_y$; i.e., B must output the yth bit of the input vector x.
- In the *bit-vector disjointness problem*, denoted DISJ, each player A and B has an n-bit vector, x and y respectively, under the assumption that there is at most one index i such that $x_i = y_i = 1$; $f(x, y) = 1$ if there exists such an index i, and $f(x, y) = 0$ otherwise.[1]

We are particularly interested in randomized protocols for these problems. We assume the *private-coin, two-sided model*, in which both A and B have access to private sources of random bits and, for a given error parameter ϵ, the reported answer must be correct with probability $1 - \epsilon$. The cost in this model is the worst-case number of bits transmitted in all rounds over all possible inputs and random choices of A and B.

Denote by $f^{\epsilon}_{\mathrm{IND}}(n)$ and $f^{\epsilon}_{\mathrm{DISJ}}(n)$ the respective complexities of the IND and DISJ functions for some arbitrary but fixed protocol in this model. Kremer, Nisan, and Ron [16, Theorem 3.7] show that for any one-round protocol and constant $\epsilon < 1/3, f^{\epsilon}_{\mathrm{IND}}(n) = \Omega(n)$. Bar-Yossef et al. [1, Theorem 6.6] show that for any multi-round protocol, $f^{\epsilon}_{\mathrm{DISJ}}(n) \geq \frac{n}{4}\left(1 - 2\sqrt{\epsilon}\right)$.

2.4. Previous results on streaming graph problems

While plentiful results on streaming algorithms for many types of problems now grace the literature [18], such results for graph problems are still relatively few.

Henzinger, Raghavan, and Sajagopalan [13] give upper and lower bounds for solving various neighbor and path queries on graphs. Bar-Yossef, Kumar, and Sivakumar [2] give a general reduction tool for a class of "list-efficient" primitives and use it to devise a streaming algorithm for counting triangles in undirected graphs. Feigenbaum et al. [9,10] give streaming results for various distance problems on undirected graphs, including the construction of spanners, which was also discussed by Elkin and Zhang [7]. Feigenbaum et al. also give streaming results for computing matchings [9] as well as $\Omega(n)$ results for a general class of "balanced" graph problems [10].

For computing non-emptiness in directed graphs, Buchsbaum, Giancarlo, and Westbrook [4] claimed results similar to ours. Their results for single-pass algorithms relied on a lemma [4, Lemma 2.1], the proof of which is incorrect. Specifically, they assume that a particular association from a graph class to a memory image of 2^g possible configurations is onto. This assumption, however, is unjustified: one is not free to reduce the memory size to make it true, because g is fixed by assumption. For $O(1)$-pass algorithms, they use a different proof technique to show an $\Omega(n)$-bit lower bound, which we strictly improve.

Our results show that finding common neighborhoods does not even fit into the "semi-streaming" model of Feigenbaum et al. [9], which allows $O(n \log^{O(1)} n)$ bits of space to process an n-vertex input graph.

3. Single-pass data-stream algorithms

Lemma 1. *Assume that for a given s, m, and integer $c \in [2, s]$, there exists an undirected graph G_u with s vertices and m edges that does not contain $K_{2,c}$ as a subgraph. Then there exists a family of 2^m directed graphs, each of which has $3s + c - 1$ vertices, for which any randomized, single-pass data-stream algorithm A that with probability $1 - \epsilon$ correctly answers the non-emptiness query for $T(\{u, v\}) = c$ must use at least $f^{\epsilon}_{\mathrm{IND}}(m)$ bits of space.*

Proof. Form directed graph G by arbitrarily directing the edges of G_u; there are 2^m possible such G's. Now augment G so that each original vertex has out-degree at least c and has a unique neighbor not shared by any other vertex, as follows.

Assign an arbitrary labeling $1, \ldots, m$ to the original arcs. For each original vertex u, add two new vertices, f_u and h_u, and add arc (u, f_u); this is the only arc to f_u. Each original vertex now has out-degree at least one. We call h_u the *shadow vertex of u* and will use it below. Finally add $c - 1$ new vertices $g_1, \ldots, g_{c-1}$, and for each original vertex u' with fewer than c neighbors, add arcs (u', g_i) for $1 \leq i \leq c - deg(u')$. Call this augmented graph G'. See Fig. 1(a–b).

Note that only original vertices have positive out-degree, each original vertex now has out-degree at least c, and G' still has no $K_{2,c}$. The latter claim follows because:

(1) Only original vertices have non-zero out-degree.
(2) Any original vertex u has f_u as a neighbor not shared with any other vertex and thus requires degree exceeding c to have c neighbors in common with another vertex.
(3) Any vertex u with degree exceeding c has only f_u in addition to its original neighbors, no other vertex is adjacent to f_u, and no $K_{2,c}$ existed before the augmentation.

Now we reduce IND to the non-emptiness problem. Players A and B both start with knowledge of G'.

Player A has an m-bit vector z. For each $1 \leq i \leq m$, if $z_i = 1$, he maintains the original ith arc, say (a, b), in the augmented graph; otherwise, he changes this arc to (a, h_b). Call the result G''. Player A then runs A on G'' and transmits the final memory image (contents of the work tape), which represents the entire state of A, to Player B.

[1] The assumption that x and y share at most one 1-bit is typical in the literature but not strictly necessary for our reduction.

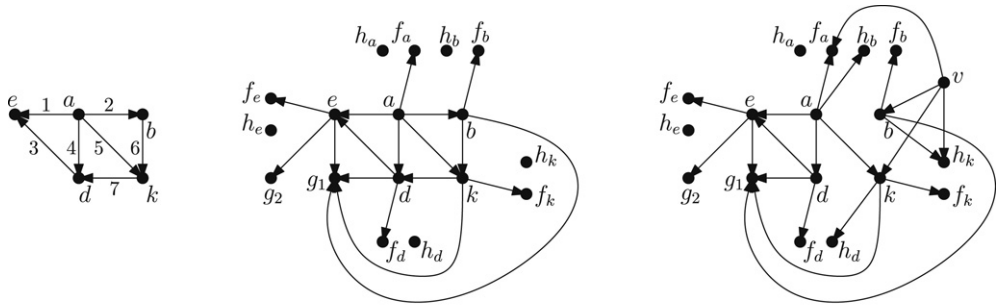

Fig. 1. (a) A directed graph G on vertices a, b, d, e, k with 7 labeled arcs and no $K_{2,3}$. (b) The augmented graph G', for $c = 3$. (c) A possible instance of G'''. In this example, Player A's bit vector is 1011100, so the original arcs (a, b), (b, k), and (k, d) are replaced by (a, h_b), (b, h_k), and (k, h_d). Player B's query index is 2, corresponding to arc (a,b), so he adds vertex v with $N(v) = \{f_a, b, k, h_k\}$; alternatively, he could set $N(v) = \{f_a, b, d, h_d\}$ or $N(v) = \{f_a, b, e, h_e\}$, not shown.

Player B has a query index $i \in [1, m]$, which he will use to create a new vertex v and an appropriate neighborhood. He will finish running algorithm A, starting from the received memory image, by feeding $(v, N(v))$ to it, so that the answer to the non-emptiness query will be identical to bit i in A's input vector z.

Let (a, b) be the ith original arc. First set the neighbors of v to be the following: b, f_a, and any $c - 2$ additional neighbors of a in G'. Then add to the neighbor set of v the shadow vertex for each original vertex in $N(v)$ except that for b. Call the result G'''. See Fig. 1(c). Notice that:

(1) Based on the choices of player A induced by vector z, for each pair x, y of original vertices, at most one of (x, y) and (x, h_y) appears as an arc in G''. Thus any $v' \neq v$ may have at most c common neighbors with v in G'''.
(2) Because only a and v have arcs to f_a, only a may have c common neighbors with v in G'''.
(3) If bit i was zero in the input x of player A, then b is not a common neighbor of a and v, and thus $CN(G''') = \emptyset$; if bit i was one, then $CN(G''') \supseteq \{\{a, v\}\}$.

Thus the output of A after B's additional input reports an answer to original instance of IND with the same correctness criteria. Because the only message transmitted from A to B was the memory image of A after the modified graph was input to it, the lemma follows.

Remark 1. If the directed graph G in the preceding proof has only vertices of degree r and 0, then any graph on which A is run has only vertices of degree $r + 1$ and 0 (except v). Thus we can change the threshold function to be $T(\{u, v\}) = \alpha \min(|N(u)|, |N(v)|)$ so long as $c \leq \alpha(r + 1)$, and the proof will still work if we ensure that v has at least $r + 1$ neighbors: b, f_a, and $\lceil \alpha(r + 1) \rceil - 2$ neighbors of the respective vertex a along with any shadow vertices of such, and possibly some new vertices. The number of extra vertices needed is linearly bounded, so the lower bound applies asymptotically to any algorithm solving the non-emptiness problem for $T(\{u, v\}) = \alpha \min(|N(u)|, |N(v)|)$.

For s and c denote by $ex(s, K_{2,c})$ the maximum number of edges a graph with s vertices may have without containing a subgraph isomorphic to $K_{2,c}$. Lemma 1 implies that $f^{\epsilon}_{\mathrm{CN}}(n, c) = \Omega(ex(n, K_{2,c}))$. Bounding $ex(n, K_{2,c})$ in terms of n and c is a problem in extremal graph theory, related to an open problem posed by Zarankiewicz [24]. Füredi [11] has given an algebraic graph construction to gain exact asymptotics for this problem. The following theorem and proof closely match Füredi's result. We specify them to make the constructed graphs regular, which will benefit the sequel.

Theorem 2 (*Füredi [11]*). *For any prime power q and c such that $\frac{q-1}{c-1}$ is an integer, there exists a bipartite q-regular graph on two classes of $\frac{q^2-1}{c-1}$ vertices that has no $K_{2,c}$ as a subgraph.*

Proof. For simplicity, let $t = c - 1$. Let $\mathbb{F}$ be the q-element field, and let $h \in \mathbb{F}$ be an element of order t. Let $H = \{h^\alpha : \alpha = 0, 1, \ldots, t - 1\}$ be the set of powers of h.

Consider the set $\mathbb{F} \times \mathbb{F} \setminus \{(0, 0)\}$ and the following equivalence relation $\sim$: for any pairs (x, y) and (x', y'), let $(x, y) \sim (x', y')$ if and only if there exists an α such that $x' = xh^\alpha$ and $y' = yh^\alpha$. Each equivalence class contains t pairs; thus the number of equivalence classes is $\frac{q^2-1}{t} = \frac{q^2-1}{c-1}$.

Consider a bipartite graph on vertex sets A and B, each vertex set corresponding to the set of equivalence classes of $\mathbb{F} \times \mathbb{F} \setminus \{(0, 0)\}$. For each $(x, y) \in A$ and $(x', y') \in B$, connect them with an edge if and only if $xx' + yy' \in H$.

Let $(x, y) \in A$ be a vertex. Assume without loss of generality that $x \neq 0$. Then for any $y' \in F$ and $\alpha \in \{0, 1, \ldots, t - 1\}$ there is a unique solution for x' to the equation $xx' + yy' = h^\alpha$. There are thus tq solutions (x', y') to $xx' + yy' \in H$, which form equivalence classes of size t each. Therefore (x, y) in A has q neighbors in B.

Consider a pair (x, y) and (x', y') of distinct vertices in A. For any common neighbor $(u, v) \in B$ of them, the following equations hold for some $\alpha, \beta \in \{0, 1, \ldots, t - 1\}$:

$$\begin{aligned} xu + yv &= h^\alpha \\ x'u + y'v &= h^\beta. \end{aligned}$$

Considering this as a linear system of equations in variables u, v it follows that:

- If the determinant
$$\begin{vmatrix} x & y \\ x' & y' \end{vmatrix}$$
is non-zero, then for each α, β the equation has a unique solution for (u, v). There are t^2 choices for the pair α, β. As the solutions come in equivalence classes of size t, the vertices (x, y) and (x', y') have exactly t common neighbors.
- If the above determinant is zero, then there is an element $g \in F \setminus \{0\}$ such that $x' = xg$ and $y' = yg$. (Recall that neither (x, y) nor (x', y') can be $(0, 0)$.) Then the system of equations has no solution unless $h^\beta = h^\alpha g$, from which $g \in H$, and thus $(x, y) \sim (x', y')$, which is a contradiction.

We formalize the end of the preceding proof for later use.

Lemma 3. *For the bipartite graph for parameters q and c constructed by Theorem 2, the following holds: The vertices of A can be partitioned into $q + 1$ classes, each having $\frac{q-1}{c-1}$ elements, such that any two vertices in the same class have no common neighbors and any two vertices in different classes have exactly $c - 1$ common neighbors.*

Proof. Recall the construction in the proof of Theorem 2. For any $(x, y) \in A$, consider the vertices $(x', y') = (xg, yg) \in A$ for $g \in \mathbb{F} \setminus \{0\}$. As g traverses the cosets of H, we get a class of $\frac{q-1}{t}$ vertices in A, no pair of which has a common neighbor. Thus we can classify the vertices of A into $\frac{q^2-1}{t} \cdot \frac{t}{q-1} = q + 1$ classes, each having $\frac{q-1}{t}$ elements, such that any two vertices in the same class have disjoint neighborhoods, and any two vertices from different classes share exactly t common neighbors.

The following upper bound will be essential in giving near-optimal algorithms for the common neighborhood problem.

Claim 4. *For directed graphs $ex(n, \overrightarrow{K_{2,c}}) \leq \sqrt{c}n^{3/2}$.*

Proof. Theorem 2.3 of Bollobás [3, p. 310] states that

$$ex(n, K_{s,t}) \leq \frac{1}{2}(s-1)^{1/t}(n-t-1)n^{1-1/t} + \frac{1}{2}(t-1)n.$$

The claim follows by setting $s = c$ and $t = 2$.

We are now ready to state the main theorems of this section.

Theorem 5. *For any constant $\epsilon < 1/3$, there exist infinitely many values of n and c such that $f^{\epsilon}_{\mathrm{CN}}(n, c) = \Omega(\sqrt{c}n^{3/2})$ for one-pass data stream algorithms. There exists a deterministic, one-pass data stream algorithm that solves the non-emptiness problem with $T(\{u, v\}) = c$ using $O(\sqrt{c}n^{3/2})$ cells ($O(\sqrt{c}n^{3/2} \log n)$ bits) of space.*

Proof. By Theorem 2, for infinitely many values of s and c there exists a graph with s vertices and $\Omega(\sqrt{c}s^{3/2})$ arcs that does not contain $K_{2,c}$. The lower bound follows from Lemma 1 by setting $n = 3s + c - 1$ and using the result from Kremer, Nisan, and Ron [16, Theorem 3.7] that for any one-round protocol and constant $\epsilon < 1/3$, $f^{\epsilon}_{\mathrm{IND}}(k) = \Omega(k)$.

For the upper bound consider the following algorithm. Store the entire graph and process it off-line if the number of arcs does not exceed $\sqrt{c}n^{3/2}$. Otherwise by Claim 4 the proper answer to the non-emptiness query is "yes."

To get a similar lower bound for the non-emptiness problem with $T(\{u, v\}) = \alpha \min(|N(u)|, |N(v)|)$ we use Remark 1 with the graphs of Theorem 2.

Theorem 6. *For any $\alpha \in (0, 1]$, there exist infinitely many values of n such that $f^{\epsilon}_{\mathrm{CN}}(n, \alpha \min(|N(u)|, |N(v)|)) = \Omega(\alpha n^2)$.*

Proof. For any fixed $\alpha > 0$, we can fix a $\beta \in (\alpha/2, \alpha)$ such that there are infinitely many prime powers q and c dividing $q - 1$ with $\frac{c}{q-1} \in [\beta, \alpha]$. Thus there are graphs for infinitely many n having $n = 2\frac{q^2-1}{c} = \Theta(q/\alpha)$ vertices and $\Theta(nq) = \Theta(\frac{q^2}{\alpha}) = \Theta(\alpha\frac{q^2}{\alpha^2}) = \Theta(\alpha n^2)$ arcs. The theorem follows from Remark 1.

4. $O(1)$-pass data-stream algorithms

In this section we will derive bounds similar to those of the previous section. As the algorithms are allowed a constant number of passes over the input, the players can exchange messages in both directions, so we can no longer use a small and isolated augmentation of the graph depending only on the input of one player. This makes the reduction more complex, and our main tool will be the DISJ problem, for which multi-round protocols have been bounded.

Lemma 7. *Assume that for some m and $d > 0$ there exists a bipartite graph $G_0(X, Y, E)$ with m edges such that:*

(1) *X is partitioned into known* classes $X_1, \ldots, X_k$*;*
(2) *no pair of vertices in any one class of X has more than one common neighbor;*

(3) *no pair of vertices in two distinct classes of X has more than d common neighbors.*

Then for any $c > d$ there exists a family of directed graphs with $3|X| + |Y| + k(c-2)$ vertices each for which any randomized, $O(1)$-pass data-stream algorithm A that with probability $1-\epsilon$ correctly answers the non-emptiness query for $T(\{u, v\}) = c$ must use $\Omega(f^{\epsilon}_{\mathrm{DISJ}}(m))$ bits of space.

Proof. We will give a reduction from DISJ, using a similar framework to the reduction used in the proof of Lemma 1. Using G_0 we will define a family of directed graphs G with three classes X', Y', and Z' of vertices. Neighborhoods of vertices in X' (rsp., Y') will depend on the input vector x of player A (rsp., y of player B); vertices in Z' will have out-degree zero. Then player A will run A on the adjacency lists of vertices in X', after which he will transmit the memory image of A to player B. Player B will then continue the run of A by inputting the adjacency lists of vertices in Y' and transmit the resulting memory image back to player A. A and B will iterate this procedure, always starting from the most current memory image received, until one declares the answer to the non-emptiness query. By construction, this answer will be identical to that for the instance of DISJ, and hence the lower bound for the latter will apply to the total number of bits transmitted in the protocol for A. By assumption only $O(1)$ passes are used, so the bound applies asymptotically to at least one pass.

Assign an arbitrary labeling $1, \ldots, m$ to the edges of graph G_0. Let the vertices of graph G be the following:

- for each vertex $u \in X$ in G_0, three vertices $\mu_0(u)$, $\mu_1(u)$, $\mu_2(u)$;
- for each vertex $v \in Y$ in G_0, one vertex $\nu(v)$;
- for $1 \le i \le k$, $c-2$ vertices $\gamma^i_1, \ldots, \gamma^i_{c-2}$.

Define the arcs of G as follows. (See Fig. 2.)

- For each $u \in X$ in G_0, let i be such that $u \in X_i$, and create arcs $(\mu_1(u), \mu_0(u))$, $(\mu_2(u), \mu_0(u))$, and for $1 \le j \le c-2$, $(\mu_1(u), \gamma^i_j)$ and $(\mu_2(u), \gamma^i_j)$. Thus $\mu_0(u), \gamma^i_1, \gamma^i_2, \ldots, \gamma^i_{c-2}$ are common neighbors of $\mu_1(u)$ and $\mu_2(u)$ in G.
- For each i such that bit $x_i = 1$, create arc $(\mu_1(u), \nu(v))$ corresponding to the ith edge (u, v) of G_0.
- For each i such that bit $y_i = 1$, create arc $(\mu_2(u), \nu(v))$ corresponding to the ith edge (u, v) of G_0.

Define vertex classes $X' = \{\mu_1(u) : u \in X\}$, $Y' = \{\mu_2(u) : u \in X\}$, and $Z' = \{\mu_0(u) : u \in X\} \cup \{\nu(v) : v \in Y\} \cup \{\gamma^i_j : 1 \le i \le k,\ 1 \le j \le c-2\}$. Then based upon G_0 and their respective input vectors, players A and B each know the vertex set of G; player A knows the neighbors of vertices in X'; and player B knows the neighbors of vertices in Y'. Vertices in Z' have out-degree zero.

For any $a, b \in X$ such that $a \ne b$, we claim $\mu_1(a)$ and $\mu_1(b)$ in G have at most $c-1$ common neighbors. If a and b are in the same class X_i, then $\mu_1(a)$ and $\mu_1(b)$ have $c-2$ common neighbors $\gamma^i_1, \ldots, \gamma^i_{c-2}$ plus a common neighbor $\nu(z)$ for any z that is a common neighbor of a and b in G_0; by assumption there is at most one such z. If, on the other hand, a and b are in distinct classes of X, then $\mu_1(a)$ and $\mu_1(b)$ only have common neighbors $\nu(z)$ for all z that are common neighbors of a and b in G_0; by assumption there are at most $d < c$ such z's. Similarly the pairs $\mu_1(a)$, $\mu_2(b)$; $\mu_2(a)$, $\mu_1(b)$; and $\mu_2(a)$, $\mu_2(b)$ have at most $c-1$ common neighbors in G.

Thus the only pairs of vertices in G with at least c common neighbors are of the form $\mu_1(u)$, $\mu_2(u)$ for some $u \in X$. Such a pair has c common neighbors if and only if there is some $v \in Y$ such that $(\mu_1(u), \nu(v))$ and $(\mu_2(u), \nu(v))$ are arcs in G, which occurs if and only if the ith bit is 1 in both input vectors, where (u, v) is the ith edge in G_0.

The answer to the non-emptiness query is therefore identical to that for the instance of DISJ. Note that it does not matter whether the specification of DISJ assumes that the input vectors have at most one common bit set.

We can use Füredi's constructions to bound the space requirement in terms of the number n of vertices for a fixed c.

Theorem 8. *For any fixed $c > 2$, there exist infinitely many values of n such that $f^{\epsilon}_{\mathrm{CN}}(n, c) = \Omega(n^{3/2}(1-2\sqrt{\epsilon}))$ for $O(1)$-pass data stream algorithms.*

Proof. Theorem 2 posits the existence of $K_{2,2}$-free bipartite graphs on $s+s$ vertices with $\Theta(s^{3/2})$ edges for infinitely many s. To any such graph, apply Lemma 7 with just one $K_{2,2}$-free class of vertices, and set $n = 4s + c - 2$. Bar-Yossef et al.'s result [1, Theorem 6.6] that for any multi-round protocol, $f^{\epsilon}_{\mathrm{DISJ}}(k) \ge \frac{k}{4}\left(1-2\sqrt{\epsilon}\right)$, completes the proof.

To gain asymptotics similar to Theorem 5 for the space needed in the $O(1)$-pass model, we exploit the more detailed view given by Lemma 3.

Theorem 9. *There exist infinitely many values of n and c with $c = O(n^{1/3})$ such that $f^{\epsilon}_{\mathrm{CN}}(n, c) = \Omega(\sqrt{c}n^{3/2}(1-2\sqrt{\epsilon}))$ for $O(1)$-pass data stream algorithms. This is sharp up to a logarithmic factor; i.e., there exists an algorithm that solves the non-emptiness query with $O(\sqrt{c}n^{3/2}\log n)$ bits of space.*

Proof. For infinitely many values of q and c, Lemma 3 posits the existence of bipartite graphs $G_0(X, Y, E)$ such that $|X| = |Y| = \frac{q^2-1}{c-1}$; $|E| = q\frac{q^2-1}{c-1}$; and X can be partitioned into $q+1$ classes of $\frac{q-1}{c-1}$ vertices each, such that any two vertices in identical classes have disjoint neighborhoods, and any two vertices in different classes have $c-1$ common neighbors. To any such graph, apply Lemma 7 with $d = c-1$—the partition of X is given by Lemma 3—and set $n = 4\frac{q^2-1}{c-1} + (q+1)(c-2)$.

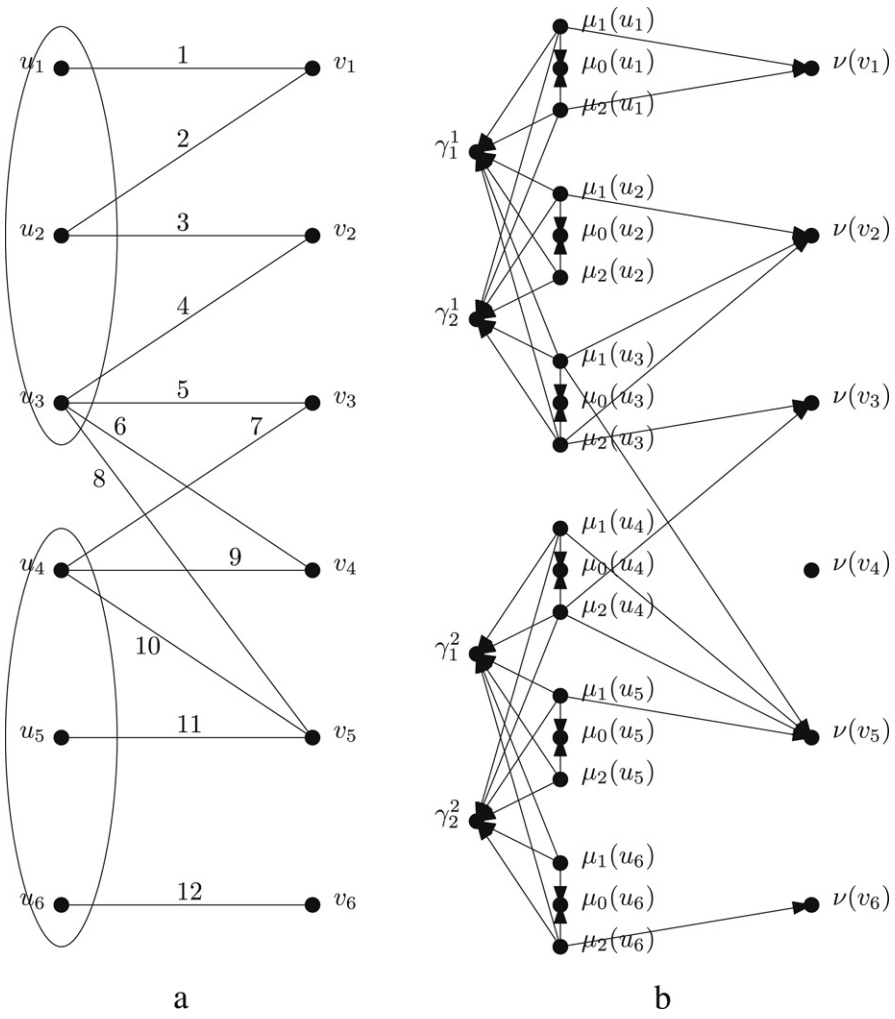

Fig. 2. (a) Bipartite graph $G_0(X, Y, E)$ with $X = \{u_1, \ldots, u_6\}$ partitioned into two, circled classes; $Y = \{v_1, \ldots, v_6\}$; and 12 labeled edges. G_0 satisfies the hypothesis for, e.g., $d = 3$. (b) Directed graph G for $c = 4$ induced by the bit vectors $x = 101100010110$ and $y = 100110100101$.

To keep $n = \Theta(\frac{q^2}{c})$, we must further bound $qc = O(\frac{q^2}{c})$, yielding the requirement $c = O(\sqrt{q})$. Thus $\frac{q^2}{c} = \Omega(q^{3/2})$, so it suffices to assume $c = O(n^{1/3})$.

Now, $|E| = q\frac{q^2-1}{c-1} = \Theta(\frac{q^3}{c}) = \Theta(\sqrt{c}\frac{q^3}{c^{3/2}}) = \Theta(\sqrt{c}n^{3/2})$. Again, Bar-Yossef et al.'s result [1, Theorem 6.6] completes the proof of the lower bound. The upper bound was presented in the proof of Theorem 5.

5. Conclusion

We have provided lower bounds on the space needed for $O(1)$-pass, randomized data-stream algorithms to determine if a given directed graph has a pair of vertices with a common neighborhood of a given size. An open problem is to remove the restriction "$c = O(n^{1/3})$" from the result of Theorem 9; currently it is an artifact of the proof construction.

Acknowledgements

Work completed while the first author was a member of AT&T Labs–Research. The second author was partially supported by the Italian FIRB project "Bioinformatica per la Genomica e la Proteomica" and by MIUR FIRB Italy-Israel project "Pattern Matching and Discovery in Discrete Structures, with applications to Bioinformatics". Third author was supported by grants OTKA T 42481,T 42706 of the Hungarian National Science Fund, and NKFP-2/0017/2002 project Data Riddle.

References

[1] Z. Bar-Yossef, T.S. Jayram, R. Kumar, D. Sivakumar, An information statistics approach to data stream and communication complexity, Journal of Computer and System Sciences 68 (4) (2004) 702–732.

[2] Z. Bar-Yossef, R. Kumar, D. Sivakumar, Reductions in streaming algorithms, with an application to counting triangles in graphs, in: Proc. 13th ACM-SIAM Symp. on Discrete Algorithms, 2002, pp. 623–632.
[3] B. Bollobás, Extremal Graph Theory, Academic Press, New York, 1978.
[4] A.L. Buchsbaum, R. Giancarlo, J.R. Westbrook, On finding common neighborhoods in massive graphs, Theoretical Computer Science 299 (1-3) (2004) 707–718.
[5] P. Chan, W. Fan, A. Prodromidis, S. Stolfo, Distributed data mining in credit card fraud detection, IEEE Intelligent Systems 14 (6) (1999) 67–74.
[6] C. Cortes, K. Fisher, D. Pregibon, A. Rogers, F. Smith, Hancock: A language for analyzing transactional data streams, ACM Transactions on Programming Languages and Systems 26 (2) (2004) 301–338.
[7] M. Elkin, J. Zhang, Efficient algorithms for constructing $(1+\epsilon, \beta)$-spanners in the distributed and streaming models, in: Proc. 23rd ACM Symp. of Principles of Distributed Computing, 2004, pp. 160–168.
[8] M. Fang, N. Shivakumar, H. Garcia-Molina, R. Motwani, J.D. Ullman, Computing iceberg queries efficiently, in: Proc. 24th Int'l. Conf. on Very Large Databases, 1998, pp. 299–310.
[9] J. Feigenbaum, S. Kannan, A. McGregor, S. Suri, J. Zhang, On graph problems in a semi-streaming model, in: Proc. 31st Int'l. Coll. on Automata, Languages, and Programming, in: Lecture Notes in Computer Science, vol. 3142, Springer, 2004, pp. 531–543.
[10] J. Feigenbaum, S. Kannan, A. McGregor, S. Suri, J. Zhang, Graph distances in the streaming model: The value of space, in: Proc. 16th ACM-SIAM Symp. on Discrete Algorithms, 2005, pp. 745–54.
[11] Z. Füredi, New asymptotics for bipartite Turán numbers, Journal of Combinatorial Theory (A) 75 (1) (1996) 141–144.
[12] V. Ganti, J. Gehrke, R. Ramakrishnan, Mining very large databases, IEEE Computer 32 (8) (1999) 38–45.
[13] M.R. Henzinger, P. Raghavan, S. Rajagopalan, Computing on data streams, Technical Report 1998-011, DEC SRC, 1998.
[14] A. Hume, S. Daniels, A. MacLellen, Gecko: Tracking a very large billing system, in: Proc. USENIX Ann. Technical Conference, 2000, pp. 93–106.
[15] J. Kleinberg, Authoritative sources in a hyperlinked environment, Journal of the ACM 46 (5) (1999) 604–632.
[16] I. Kremer, N. Nisan, D. Ron, On randomized one-round communication complexity, Computational Complexity 8 (1) (1999) 21–49; 10 (4) (2001) 314–315 (Errata).
[17] C.-S. Li, R. Ramaswami, Automatic fault detection, isolation, and recovery in transparent all-optical networks, Journal of Lightwave Technology 15 (10) (1997) 1784–1793.
[18] S. Muthukrishnan, Data streams: Algorithms and applications, Foundations and Trends in Theoretical Computer Science 1 (2) (2005).
[19] N. Nisan, E. Kushilevitz, Communication Complexity, Cambridge University Press, Cambridge, UK, 1997.
[20] S. Stolfo, W. Fan, W. Lee, A. Prodromidis, P. Chan, Cost-based modeling for fraud and intrusion detection: Results from the JAM project, in: Proc. DARPA Information and Survivability Conference and Exposition, vol. 2, 2000, pp. 130–144.
[21] S.G. Tzafestas, P.J. Dalianis, Fault diagnosis in complex systems using artificial neural networks, in: Proc. 3rd IEEE Conf. on Control Applications, vol. 2, 1994, pp. 877–82.
[22] J.D. Ullman, The MIDAS data-mining project at Stanford, in: Proc. 1999 IEEE Symp. on Database Engineering and Applications, 1999, pp. 460–4.
[23] J.S. Vitter, External memory algorithms and data structures: Dealing with massive data, ACM Computing Surveys 33 (2) (2001) 209–271.
[24] K. Zarankiewicz, Problem P 101, Colloquium Mathematicum 2 (1951) 301.