This is a summary of the talk I gave on February 16th, 2012, in University of Waterloo, in which I talked about the paper

> Testing Properties of Directed Graphs: Acyclicity and Connectivity, by Bender and Ron,

which appeared in ICALP 2000 and then in RSA 2002.

Abbas Mehrabian, February 2012

1. **Definition.** A tester for property P is a randomized algorithm that inputs a distance parameter $\epsilon > 0$, can ask questions about a mathematical object O, and
   - If O has P then says YES with probability at least $2/3$.
   - If O is ϵ-far from satisfying P then says NO with probability at least $2/3$.
   - Otherwise, it does not matter what it says.
2. Talk about graph representations (mention the zeros in the incidence list)
3. Refine the definition of a tester for directed graphs
4. Query complexity is important. We are interested in testers whose query complexity is sublinear in the size of G.
5. This paper studies the property "being acyclic" and proves that:
   (a) In the adjacency matrix representation, there is a tester with query complexity $O(\frac{\log^2(1/\epsilon)}{\epsilon^2})$.
   (b) In the incidence list representation, every tester asks $\Omega(n^{1/3})$ queries.

Acyclicity Testing Algorithm.

1. Uniformly and independently select a set of $\Theta\left(\log\left(\frac{1}{\epsilon}\right)/\epsilon\right)$ vertices, called U.

2. For every $i, j \in U$ query ask $ij \in E$? and ask $ji \in E$?

3. If $G[U]$ contains a cycle, say NO; otherwise say YES.

Theorem 1. The algorithm is a tester.

Given $W \subseteq V$ say v has high outdegree w.r.t. W if v has at least $\frac{\epsilon}{2}n$ edges to W.

Lemma 2. If G is ϵ-far from acyclic, the there exists $W \subseteq V$ s.t. $|W| \geq \sqrt{\frac{\epsilon}{2}}n$, and every vertex $v \in W$ has high outdegree w.r.t. W.

Proof. Put vertices in a linear order, such that the number of backward edges is small. Start from the end and as long as $|V| \geq \sqrt{\frac{\epsilon}{2}}n$, put a vertex v that has low outdegree at the beginning of the list. In each step you add $\epsilon/2n$ backward edges, giving a total of $\frac{\epsilon}{2}n^2$.

Once $|V| < \sqrt{\frac{\epsilon}{2}}n$ order the rest arbitrarily. This adds at most $\frac{\epsilon}{2}n^2$ backward edges. Hence total backward edges are at most ϵn^2. $\square$

Lemma 3. Let $W \subseteq V$ be s.t. every $v \in W$ has at least $\delta|W|$ edges to W, for some $0 < \delta \leq 1/2$. For $c > 10$, if we randomly choose $m = c(\log(1/\delta)/\delta)$ vertices from W, call it U then with probability at least $9/10$ the subgraph induced by U induces a cycle.

Proof. We prove whp every $u \in U$ has an outgoing edge to U. Assume that $m = c(\log(1/\delta)/\delta) = |U|$. The probability that a fixed $u \in U$ has no outgoing edge to U is at most

$$(1-\delta)^{m-1} \leq \exp(-(m-1)\delta) \sim \exp(-c\log(1/\delta)) = \delta^c.$$

By union bound the probability that some u has no edge to U is at most

$$m\delta^c = (\frac{c\ln(1/\delta)}{\delta})\delta^c < 1/10.$$

□

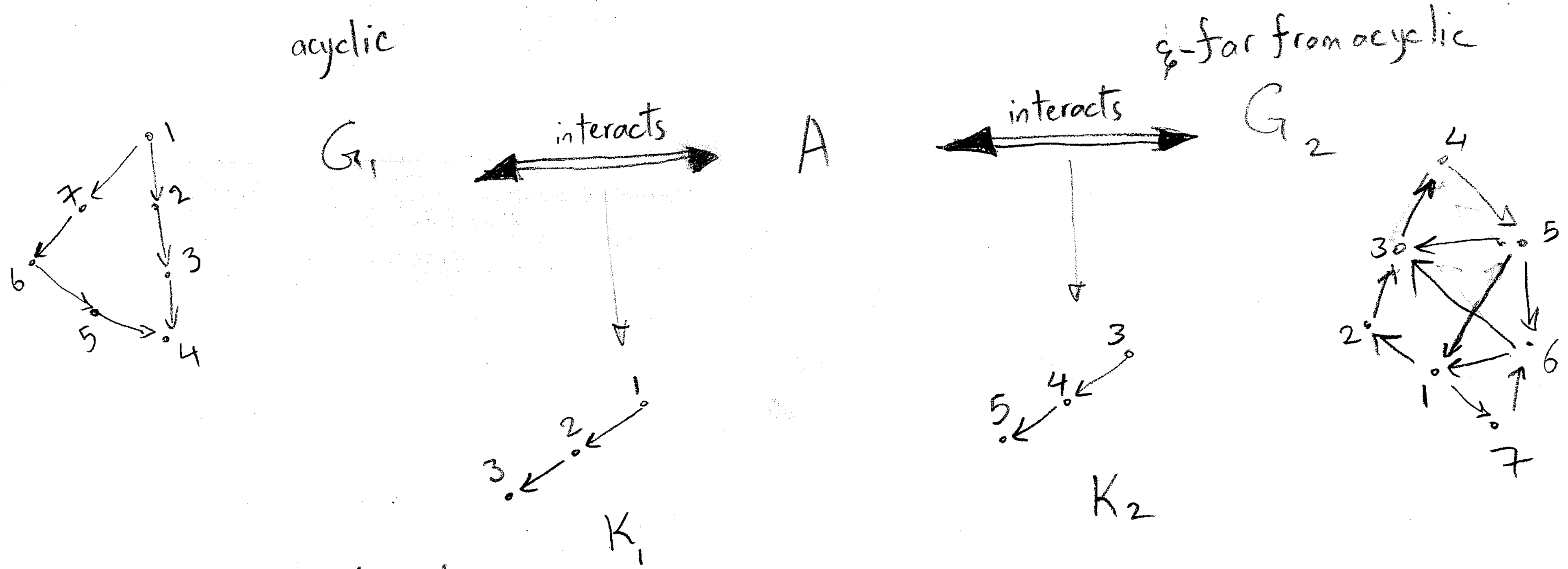
Proof of Theorem 1. If G is acyclic then the algorithm always says YES. Assume that G is ϵ-far from acyclic, let $\alpha = |W|/n$, where W is as in Lemma 2. Let $\delta = \min\{1/2, \frac{\epsilon}{2}\alpha\}$. By Lemma 2, $\alpha \geq \sqrt{\frac{\epsilon}{2}}$ and every $v \in W$ has at least $\frac{\epsilon}{2}n \geq \delta|W|$ edges to W. Let $m = 10\log(1/\delta)/\delta$. If we choose $2m/\alpha$ vertices from V, on average $2m$ of them are in W. By Chernoff's bound, with probability at least $9/10$, m of them are in W. By Lemma 3, with probability at least $9/10$, these m vertices induce a cycle and the algorithm says NO.

The number of vertices the algorithm samples equals

$$\frac{2m}{\alpha} \leq \frac{4\log(\log 1\delta)}{\delta\sqrt{\frac{\epsilon}{2}}} = O\left(\frac{\log(1/\epsilon)}{\epsilon}\right).$$

□

Theorem 4. Testing Acyclicity in the incidence-lists representation with distance parameter $\epsilon < 1/16$ requires $\Omega(n^{1/3})$ queries.



How can one prove a lower bound on the query complexity of a tester?

Knowledge graphs K_1 and K_2 are built based on the interaction of A with G_1 and G_2.

If they are isomorphic, then A cannot distinguish between G_1 and G_2.

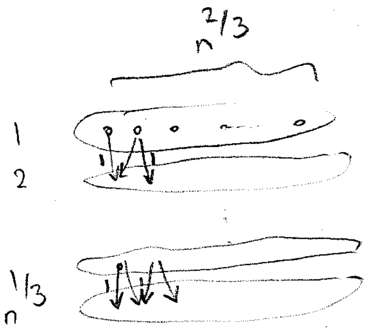
The approach does not work, because the algorithm may be specialized to distinguish between G_1 and G_2.

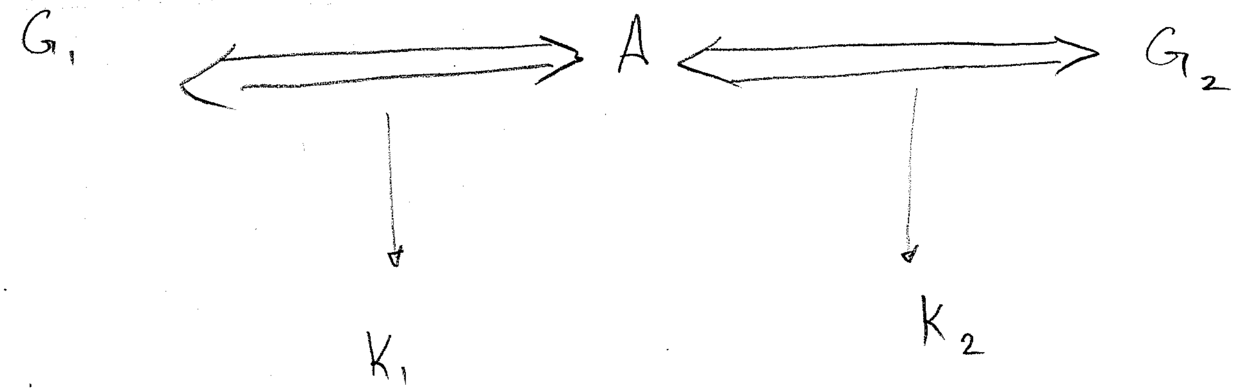
acyclic

Generate Random $G_1 \in \mathcal{G}_1$

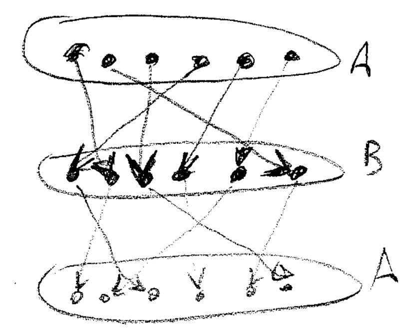
ε-far from acyclic

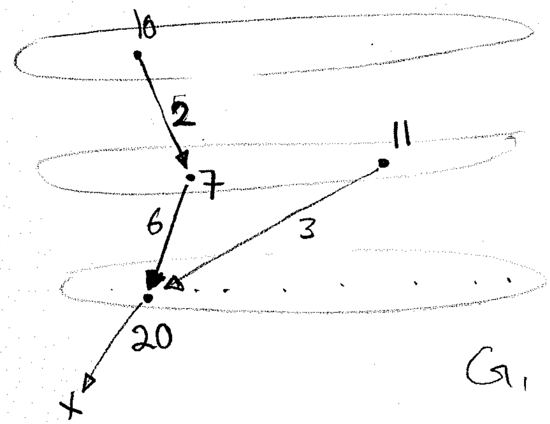
Generate random $G_2 \in \mathcal{G}_2$



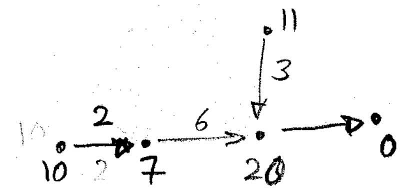


Lemma 5. Whp a randomly chosen $G_2 \in \mathcal{G}_2$ is $\frac{1}{16}$-far from acyclic.





G_1



K_1

Here wesee how to generate random graphs from the families $\mathcal{G}_1$ and $\mathcal{G}_2$, while interacting with algorithm

Note ~~that~~ the knowledge graph does not contain information about layers, unless if the algorithm queries about an outgoing edge from a vertex in the last layer!

Lemma 6. The processes described uniformly generate graphs from $\mathcal{G}_1$ and $\mathcal{G}_2$.

Lemma 7. If $o(n^{1/3})$ queries are asked, than whp the two knowledge graphs are isomporhic.

Proof. Query-answer pairs: $(v_i, m_i) - a_i$. Assume the algorithm does not ask a question twice. t queries are asked.

Observation: The knowledge graphs are isomorphic if none of the following happens:

1. A vertex in the knowledge graph is returned as the answer of a query.

2. Answer '0' is returned to some query.

For the first one, the worst case is that t vertices from the knowledge graph are in layer $i+1$, and an edge going out of a vertex in layer i is queried. Then the probability of (1) happening is at most $t/n^{2/3} = o(n^{-1/3})$. By union bound over the t queries, the total probability is $o(1)$.

For the second one, we may assume that the algorithm's objective is to find a sink. Assume the knowledge graph consists of s paths starting from vertices $h_1, \ldots, h_s$ and having lengths $l_1, \ldots, l_s$. Note that the sum of lengths is at most the number of queries, $o(n^{1/3})$. Then

$$\Pr[h_j + l_j \geq n^{1/3} \text{ for some } j] \leq \sum_j Pr[h_j + l_j \geq n^{1/3}] = \sum_j l_j / n^{1/3} = n^{-1/3} \sum_j l_j = o(1).$$

□

Remarks:

- The lower bound holds even if the algorithm is allowed to ask about incoming edges; the proof is almost the same.
- The property of being "strongly connected" has also been studied in the paper. If the algorithm can ask about incoming and outgoing edges than the query complexity is $O(1/\epsilon)$, but if the algorithm can only ask about outgoing edges then the query complexity is $\Omega(\sqrt{n})$.