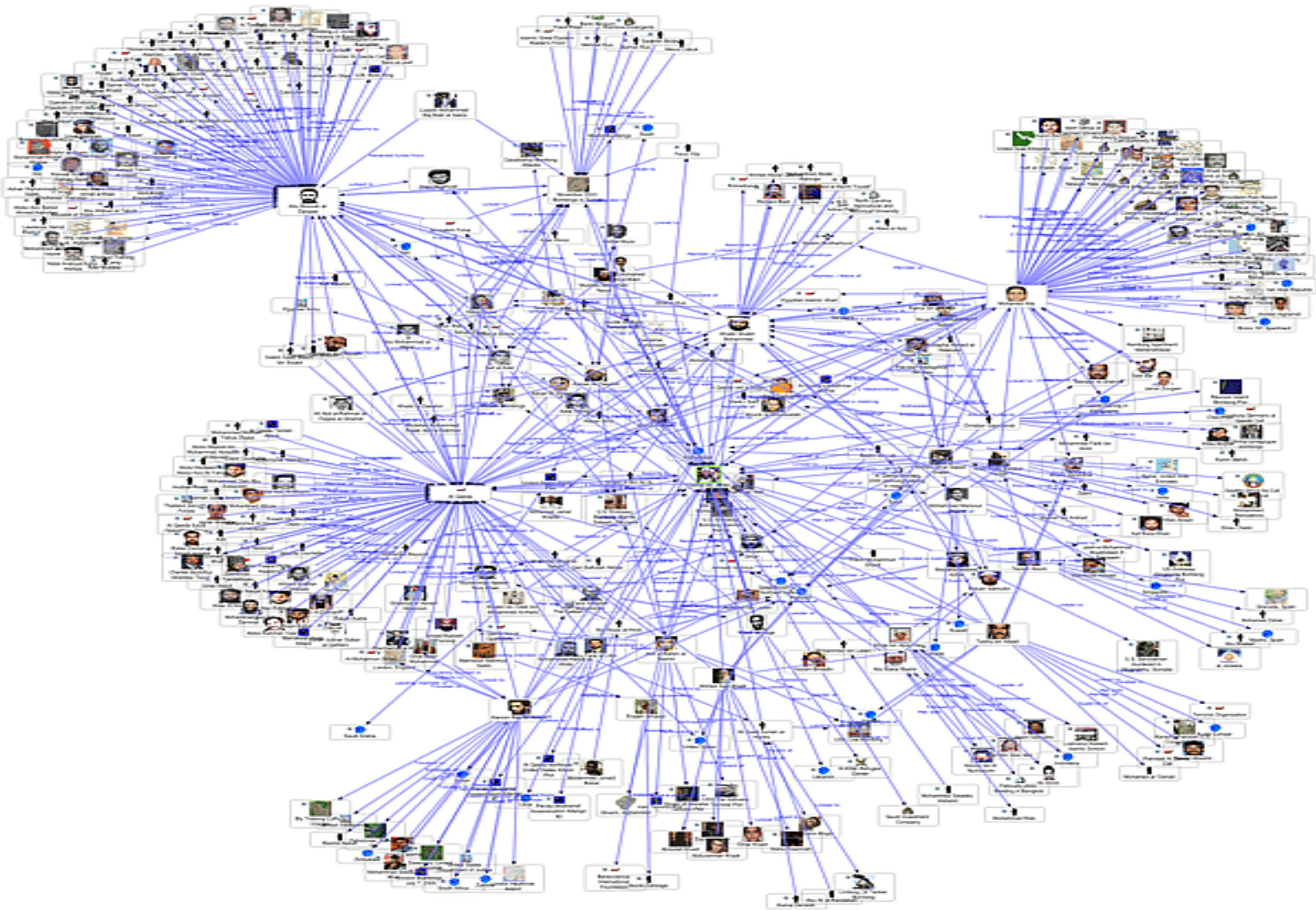
Inferring Social Networks from Outbreaks

Lev Reyzin
Georgia Institute of Technology
with
Dana Angluin and James Aspnes
Yale University

ALT 2010

How Do We Learn Social Networks?



Social Network Models

- A social network consists of nodes (agents) and edges (connections). The goal is to determine the structure of a target network.
- Active Learning – activate some nodes in the network and observe the effects.
  - eg. [Angluin, Aspnes, and Reyzin '08]
  - Often requires the learner having a lot of power.
- Passive Learning – observe the network from the outside and make conclusions about its structure.
  - This work lies here.

NIHI

2009 Swine Flu – Initial Cases

The Constraints

- The social network is an unknown graph, where nodes are agents.
  - The vertices are known to the learner.
  - The edges are to be learned or estimated.

- Let $p_{(u,v)}$ be the a priori probability of an edge between nodes u and v.

- Each observed outbreak induces (or exposes) a constraint.
  - Namely the graph is connected on the induced subset.

Finding the Cheapest Network

- If the prior distribution is independent (and probabilities are small), the maximum likelihood social network maximizes

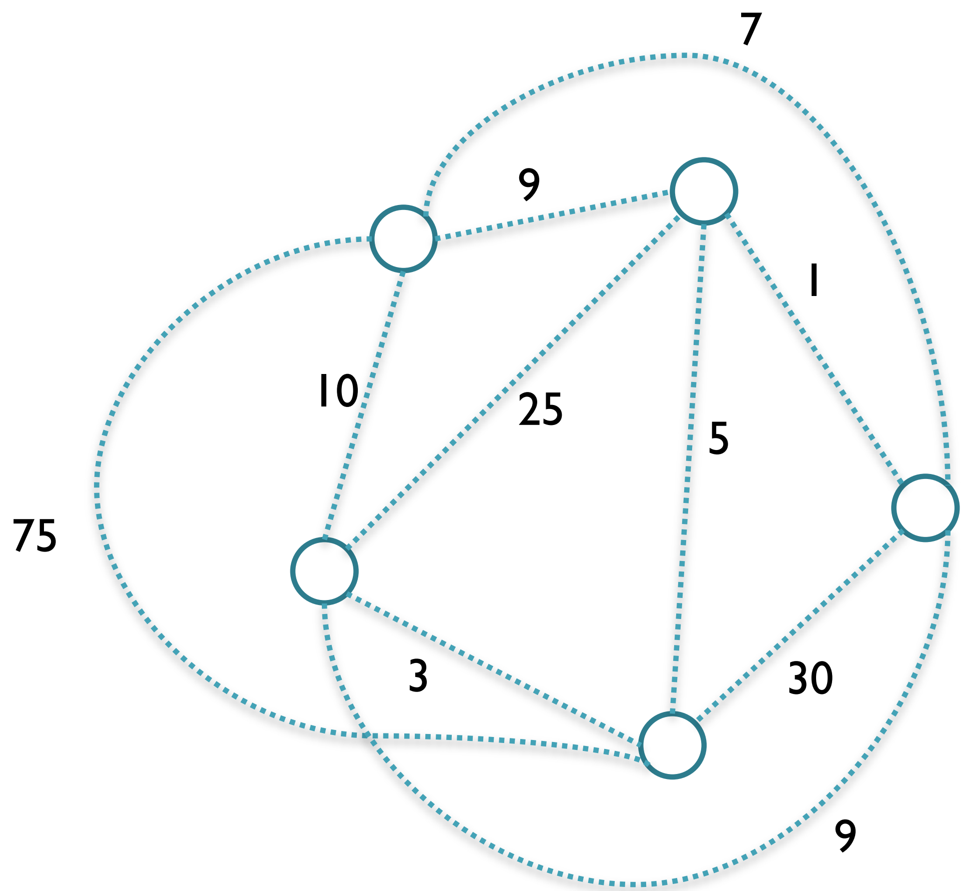
$$\prod_{u,v \in V} p_{(u,v)}$$

- This is equivalent to minimizing the sum of the log-likelihood costs

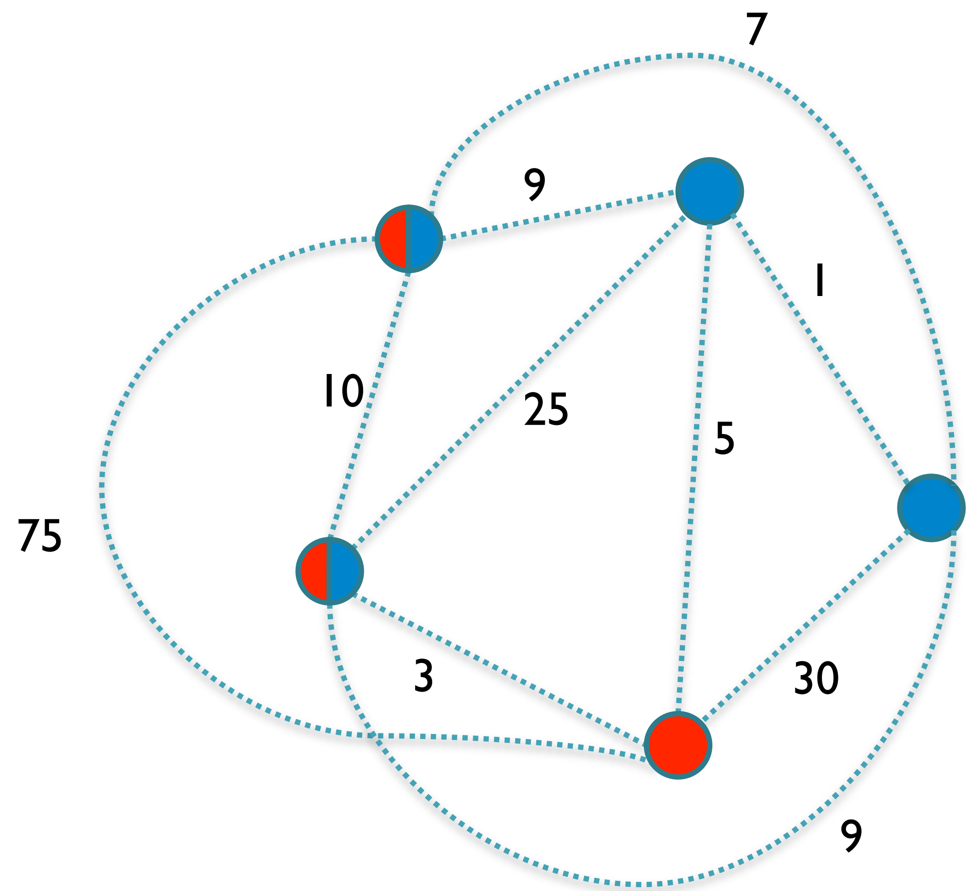
$$\sum_{v,u \in V} -\log(p_{(u,v)})$$

while satisfying the connectivity constraints.

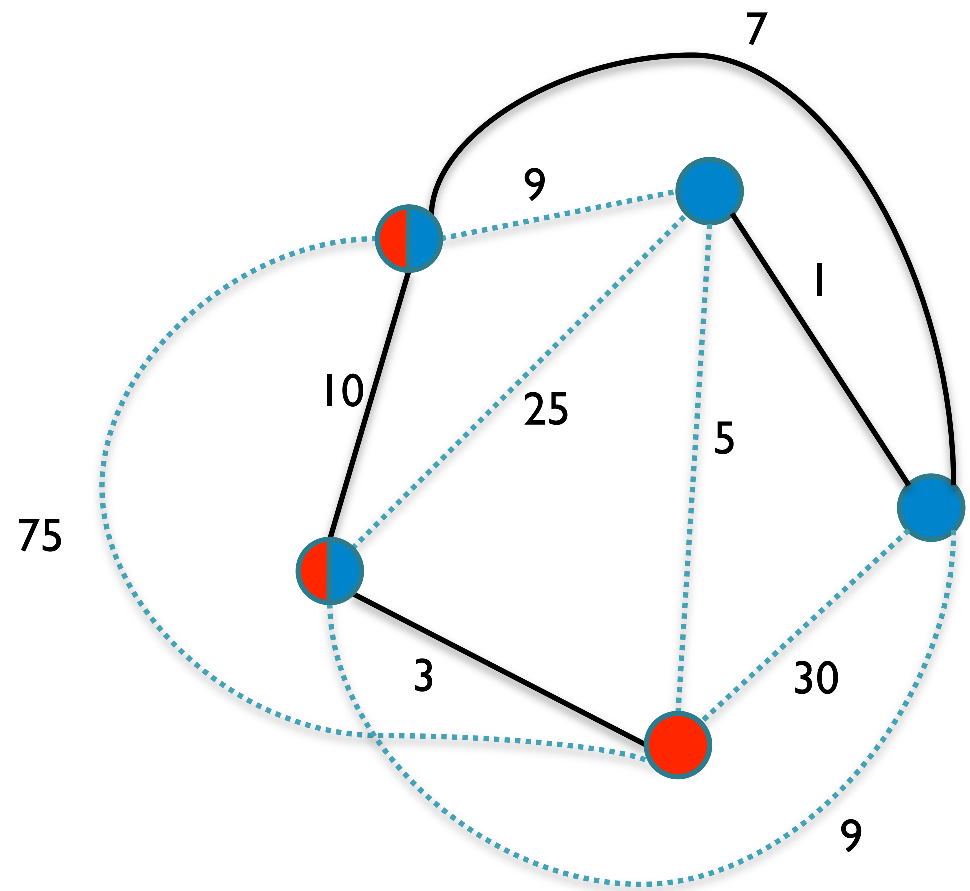
Finding the Cheapest Network Consistent with the Constraints



Finding the Cheapest Network Consistent with the Constraints



Finding the Cheapest Network Consistent with the Constraints



The Network Inference Problem

- Given:
  - vertices $V = \{v_1,\ldots,v_n\}$
  - costs c_e for each edge $e=\{v_i,v_j\}$
  - a constraint set $S = \{S_1,\ldots,S_r\}$, with $S_i \subseteq V$

- Find: a set E of edges of lowest cost such that each S_i induces a connected subgraph of $G=(V,E)$

Problem Variants

- We consider both the offline and online version of this problem. We also consider the arbitrary and uniform cost versions.

- Solved for the case where all constraints can be satisfied by a tree [Korach & Stern '03] – they left the general case open.
  - Our problem is closely related to many network optimization problems.

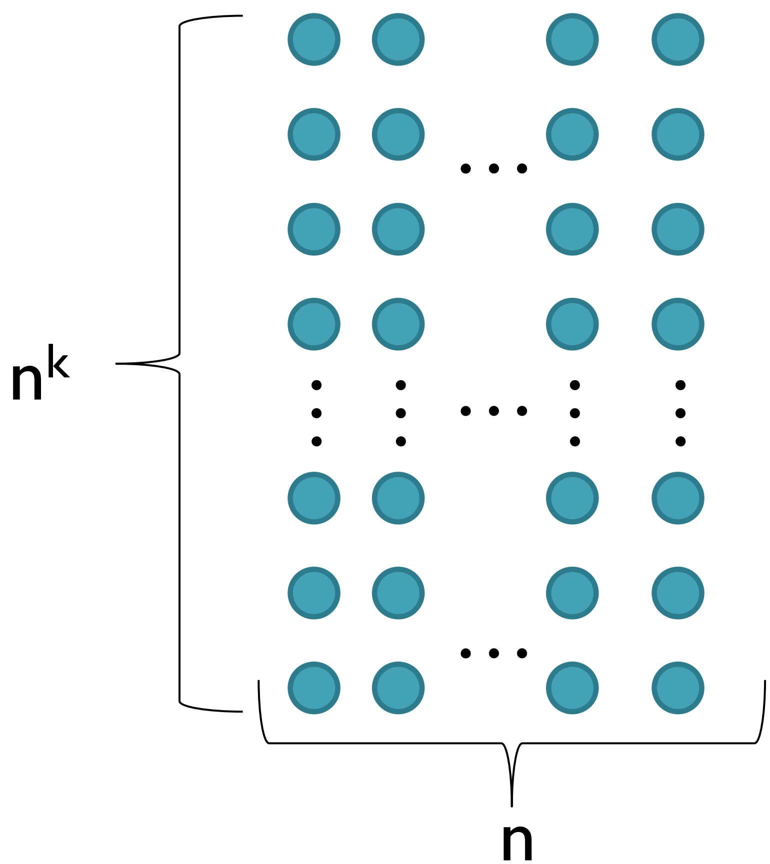
An Offline Lower Bound

- Theorem: If P ≠ NP, the approximation ratio for the Uniform Cost Network Inference problem is $\Omega(\log n)$.

- Proof (reduction from Hitting Set)
  - $U = \{v_1, v_2, \ldots, v_n\}$
  - $C = \{C_1, C_2, \ldots, C_j\}$, with $C_i \subseteq U$
  - The Hitting Set problem is to minimize |H|, where $H \subseteq U \text{ s.t. } \forall C_i \; H \cap C_i \neq \phi$

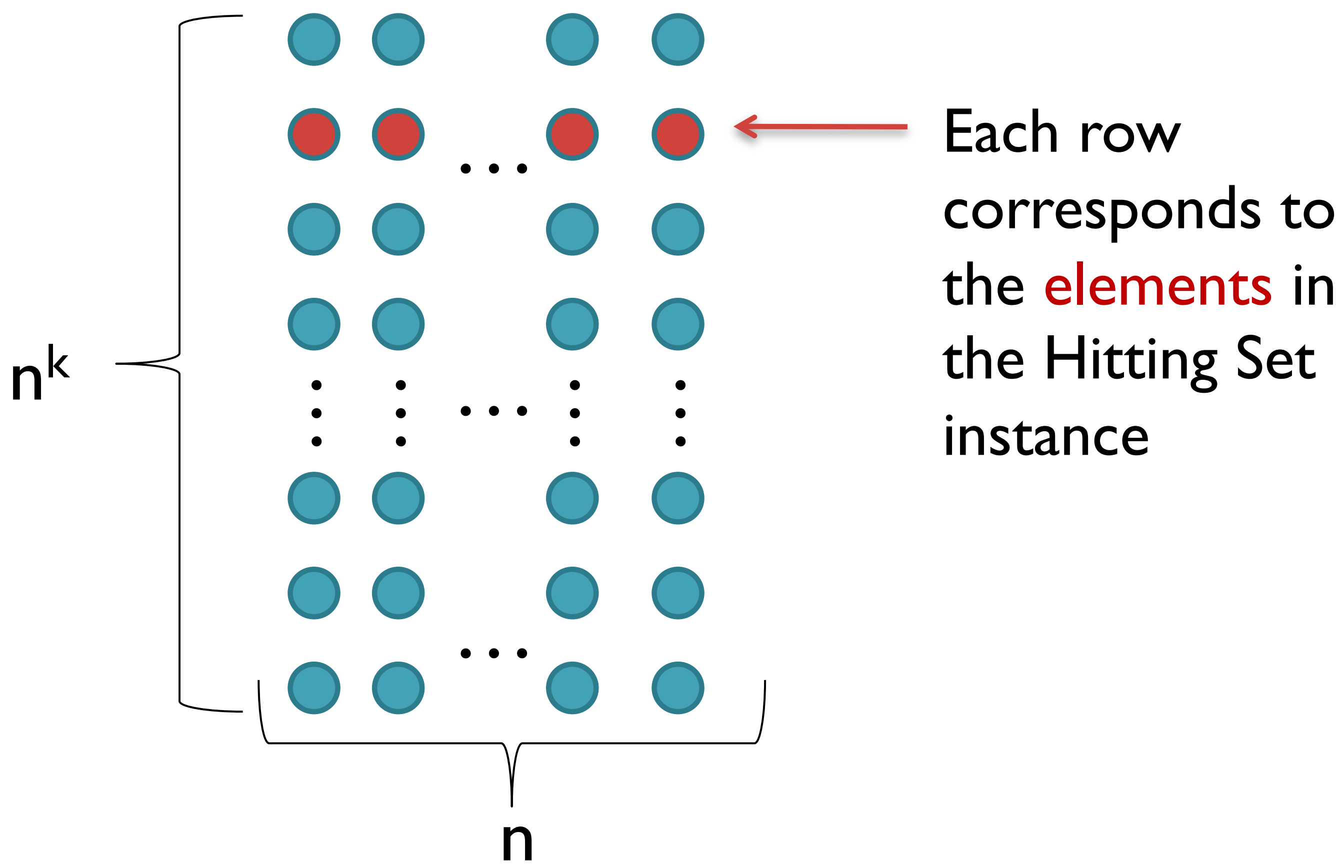
An Offline L.B. continued

- Reduction from Hitting Set
- For a constant k, We make a N.I. instance



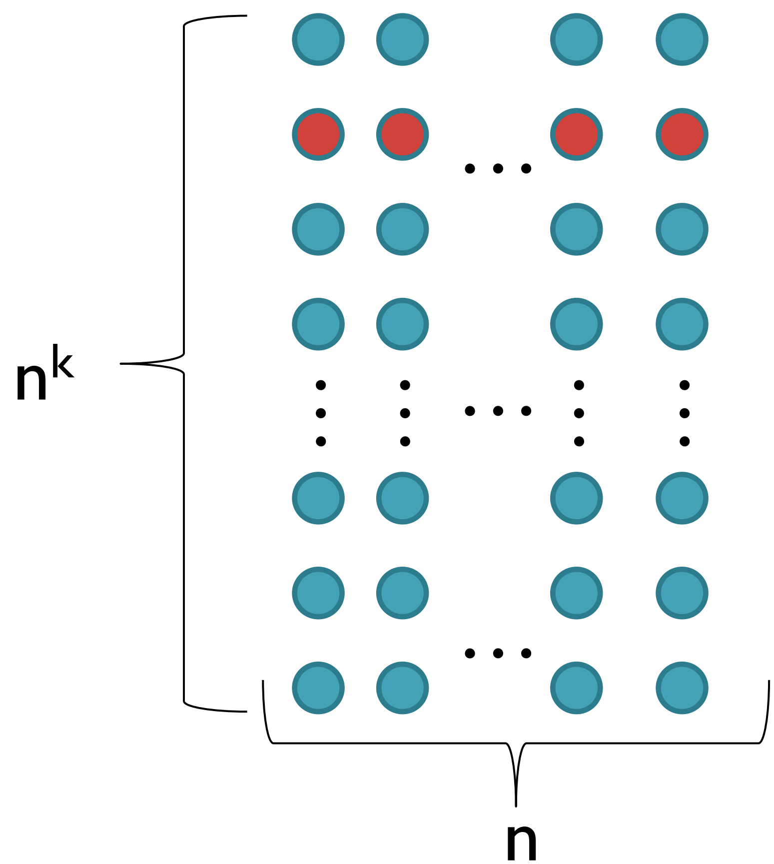
An Offline L.B. continued

- Reduction from Hitting Set
- For a constant k, We make a N.I. instance



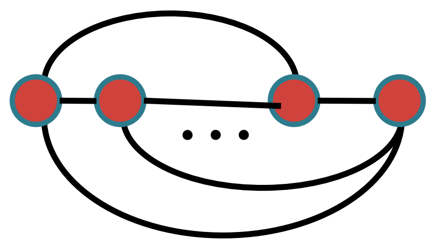
An Offline L.B. continued

- Constraints: first, for each row, give all pairwise constraints:



An Offline L.B. continued

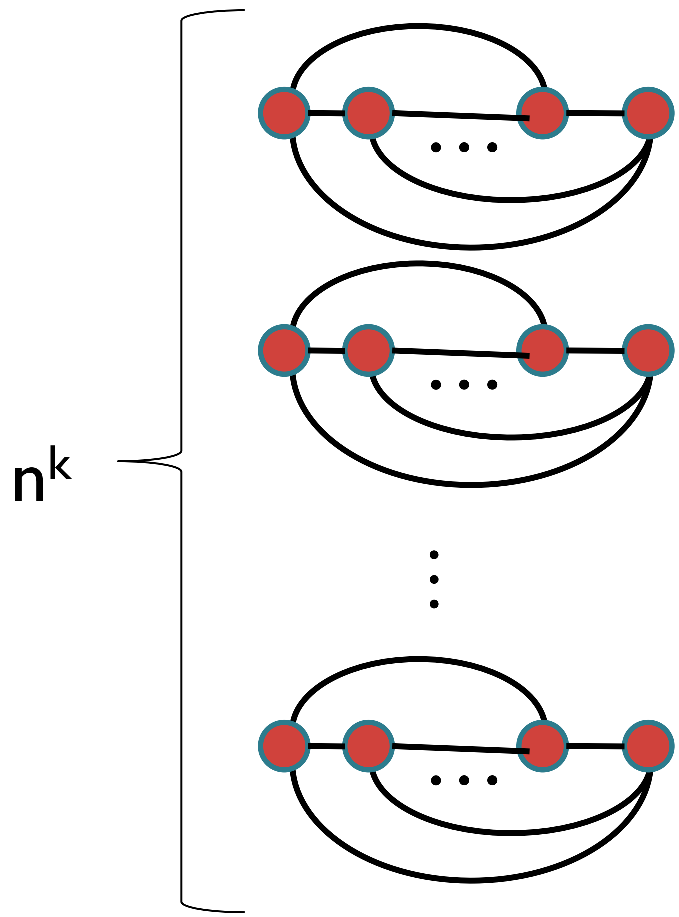
- Constraints: first, for each row, give all pairwise constraints:



- This will force the learner to put down a clique on each row

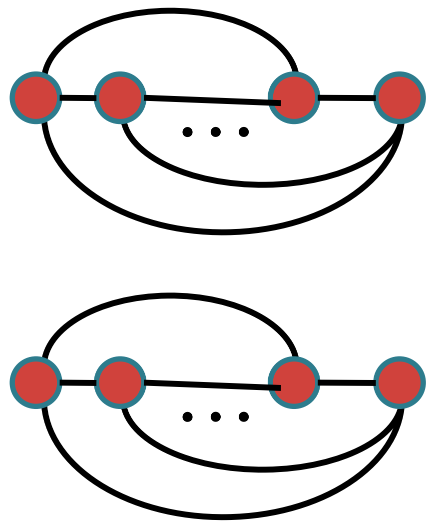
An Offline L.B. continued

- Now we have n^k rows of cliques



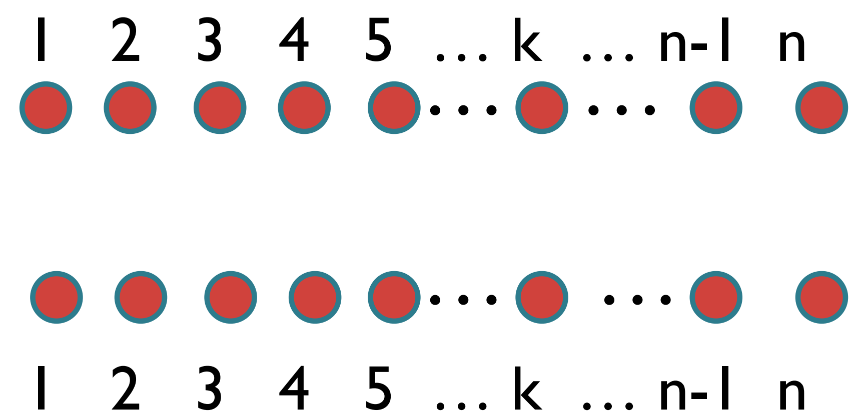
An Offline L.B. continued

- For each pair of rows:



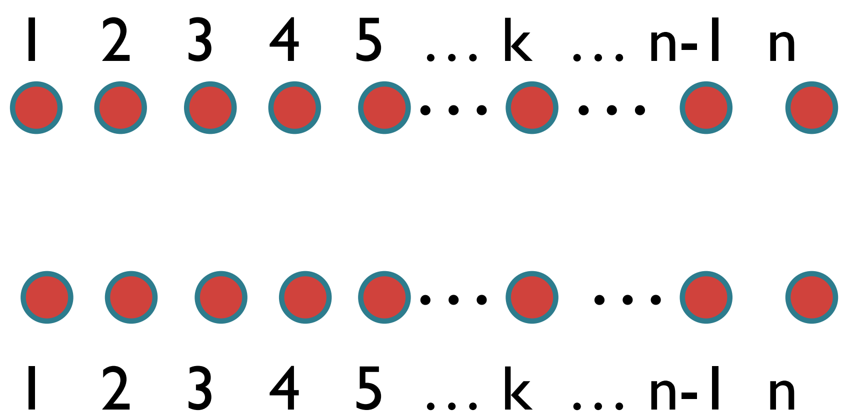
An Offline L.B. continued

- For each pair of rows:



An Offline L.B. continued

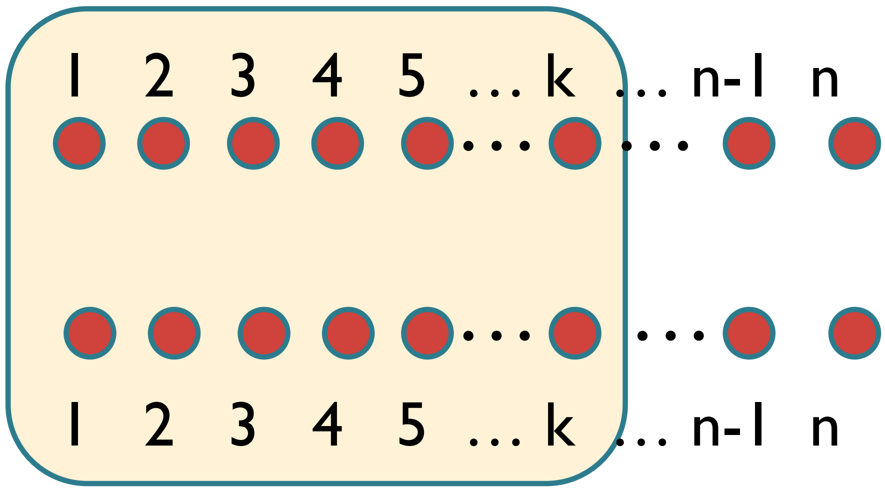
- For each pair of rows:



- w.l.o.g. for the Hitting Set constraint
  - $C_i = \{v_1, v_2, \ldots, v_k\}$
  - we will add the constraint:

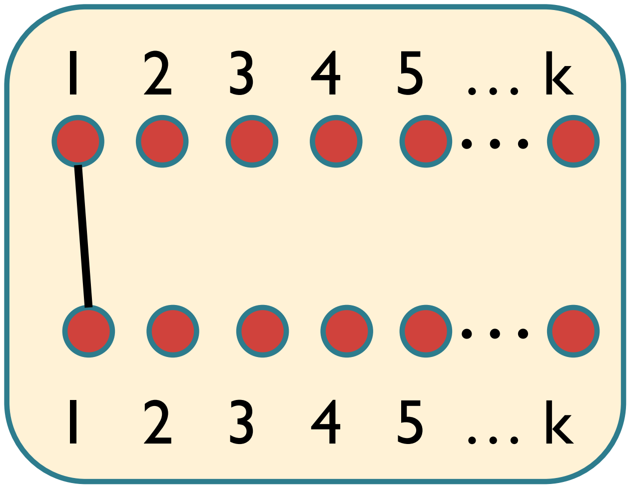
An Offline L.B. continued

- For each pair of rows:

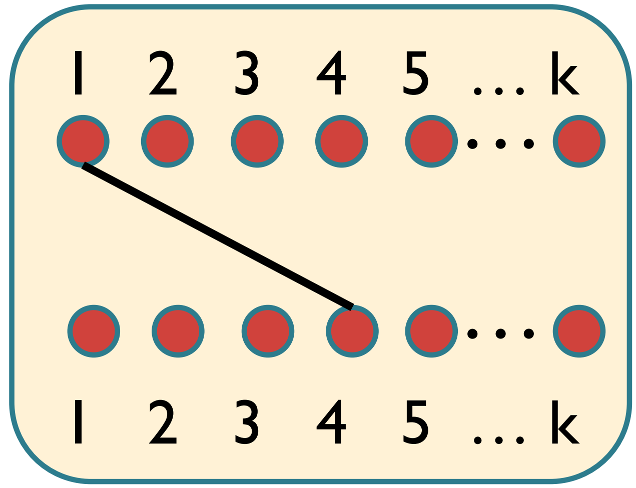


- w.l.o.g. for the Hitting Set constraint
  - $C_i = \{v_1, v_2, \ldots, v_k\}$
  - we will add the constraint:

An Offline L.B. continued

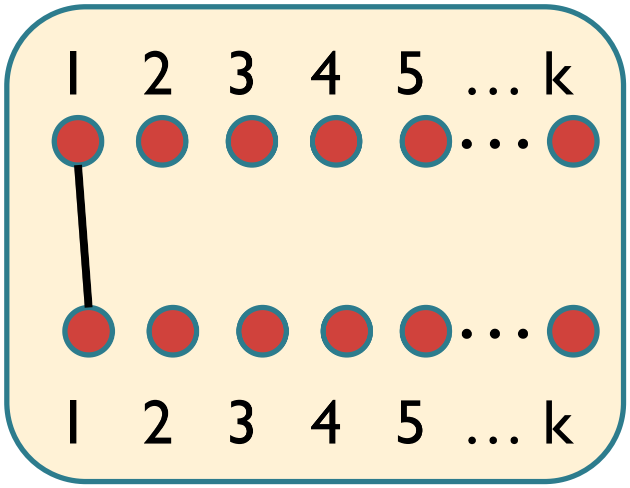


corresponds to adding v_1 to H

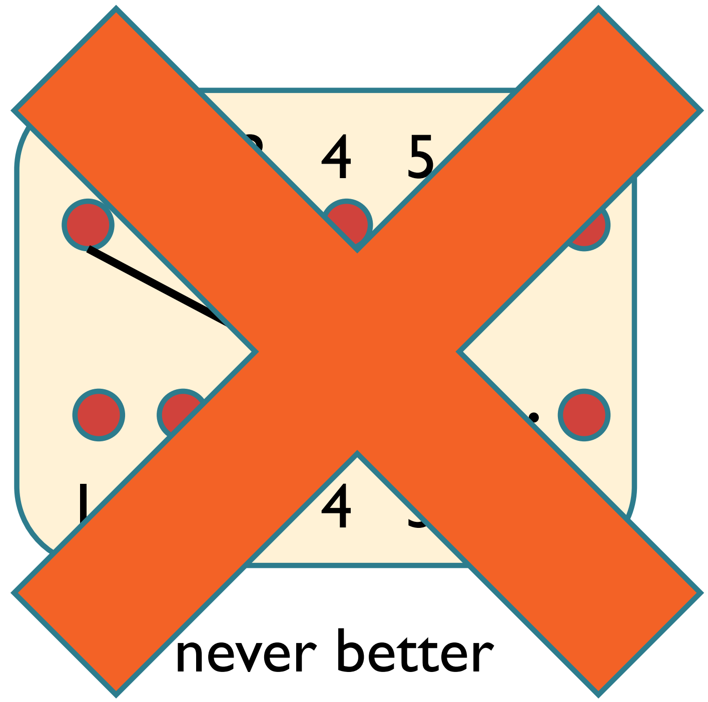


never better

An Offline L.B. continued



corresponds to adding v_1 to H



never better

Finishing the Lower Bound

- Unless P=NP, optimal Hitting Set approximation is Ω(log(n)) [Feige '98].
- The optimal algorithm pays:

$$n^k \binom{n}{2} + \mathrm{OPT}\binom{n^k}{2}$$

- But the learner pays:

$$n^k \binom{n}{2} + \Omega\left(\log(n)\mathrm{OPT}\binom{n^k}{2}\right)$$

- k can be chosen to be sufficiently large.

Offline Network Inference Algorithm

- Theorem: There is a O(log(r)) approximation algorithm to OPT

- Algorithm:
  - Let C(E) sum over all constraints S_i, 1 minus the number of components S_i induces in G(V,E).
  - Consider the greedy algorithm: while C(E) < 0, add to E the edge that has the lowest ratio of c_e to ΔC.
  - Note that at the completion of the algorithm, the constraints are all satisfied.

Algorithm Analysis

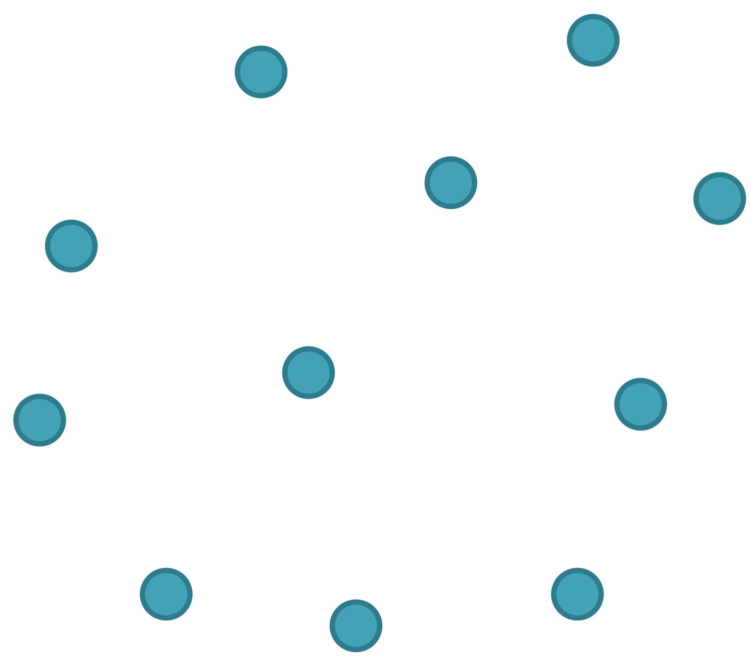
- The greedy algorithm: while $C(E) < 0$, add to E the edge that has the lowest ratio of c_e to ΔC.
- $C(E)$ is submodular in its edge set.
- A Greedy algorithm for maximizing an integer-valued submodular function on x gives an $H(m)$ approx, where $m = \max_x f(\{x\})$ [Wolsey '82]
- Each edge can increase C by at most r, so $m < r$.

The Online Problem

- Constraints S_i come in online
- Must satisfy each constraint as it comes in.
- Can add but not remove edges.
- Seemingly good ideas like placing an MST on each constraint can perform very badly.

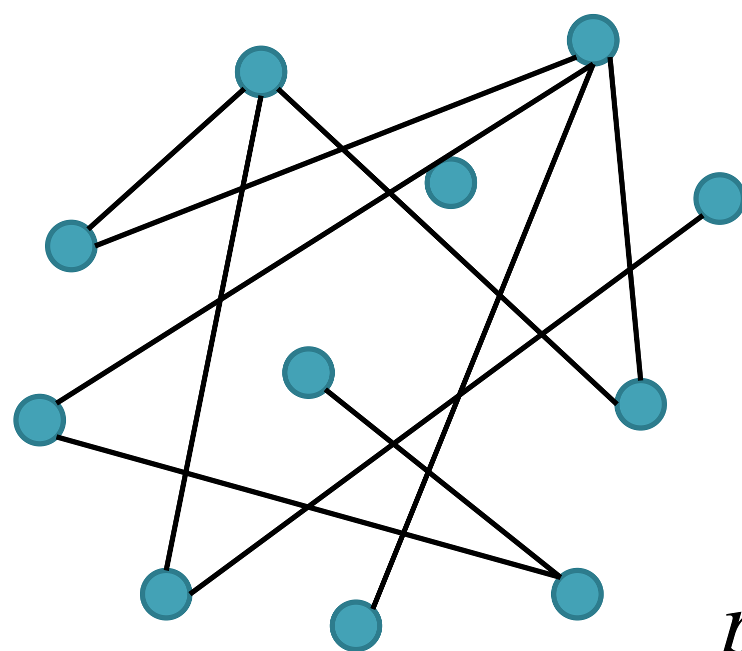
Online Algorithm Against Oblivious Adversary

$O(n^{2/3}\log^{2/3}n)$-competitive algorithm for the uniform cost problem.



Online Algorithm Against Oblivious Adversary

$O(n^{2/3}\log^{2/3}n)$-competitive algorithm for the uniform cost problem.



$$p = \frac{c\log^{2/3} n}{n^{1/3}}$$

Online Algorithm Against Oblivious Adversary

$O(n^{2/3}\log^{2/3}n)$-competitive algorithm

- All constraints S_i, $|S_i| \geq n^{1/3}\log^{1/3}(n)$ are almost surely connected
- All constraints S_i, $|S_i| < n^{1/3}\log^{1/3}(n)$ that are not already covered, we can put a clique on, and hit at least 1 edge in OPT
- We used $O(n^{5/3}\log^{2/3}(n)+n^{2/3}\log^{2/3}(n)\text{OPT})$ edges in expectation.
- Because OPT = $\Omega(n)$, we are done.

Other Online Results

- The uniform cost problem has a $\Omega\left(\sqrt{n}\right)$ - competitive lower bound

- The competitive ratio for uniform cost stars and paths is θ(log n).
  - for paths, we use pq-trees [Booth and Lueker '76]

- The arbitrary cost problem has an Ω(n)-competitive lower bound and O(n log n)-competitive algorithm.

Online Results: Competitive Ratios for Adaptive Adversaries

	any G	trees	stars	paths
uniform cost	$O(n)$ $\Omega(n^{1/2})$	?? $\Omega(\log n)$	$\Theta(\log n)$	$\Theta(\log n)$
arbitrary cost	$O(n \log n)$ $\Omega(n)$	$O(n \log n)$ $\Omega(n)$	$O(n \log n)$??	$O(n \log n)$ $\Omega(n)$

Summary

- New model for passively learning social networks.
  - Also relevant to network optimization..
- Many interesting results.
  - Solve open problem from network optimization.
- Lots of good open problems left!