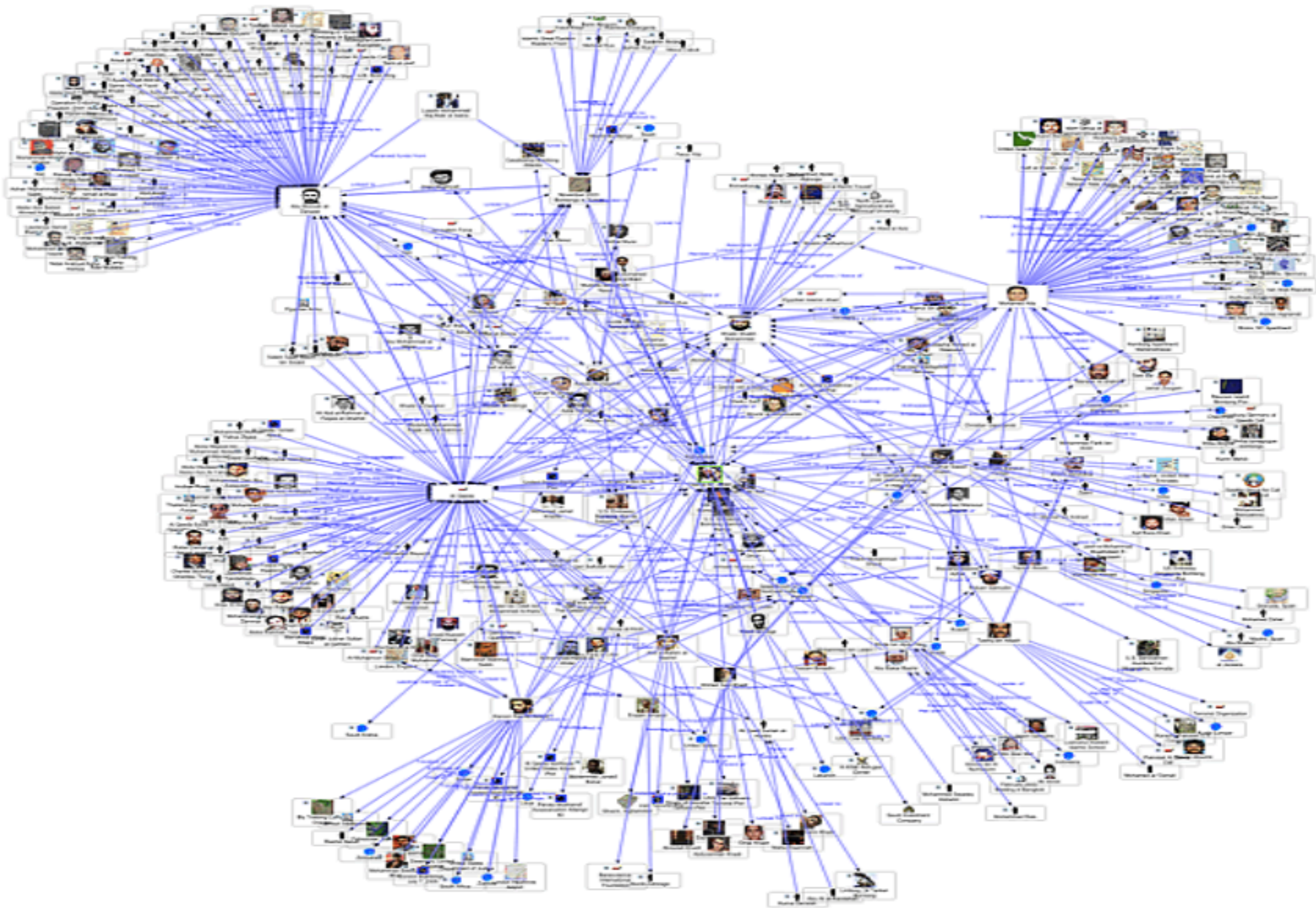
Inferring Likely Social Networks from (Ordered) Connectivity Information

Lev Reyzin
UIC MSCS
talk @ ITA 2017

based on papers with
Yi Huang and Mano Vikash Janardhanan (manuscript'17)
Dana Angluin, James Aspnes (ALT '10 & JOCO '15)

How do we learn social networks?



N1H1 Initial Infections (2009)

Learning Model

- A **social network** consists of **agents** and **connections**. The goal is to determine the **graph** of a target network.
- **Passive Learning** – observing a network from the outside and make conclusions about its structure.
- Each observed outbreak induces (or exposes) a **connectivity constraint**.
    - Namely the graph is connected on the induced subset of nodes.

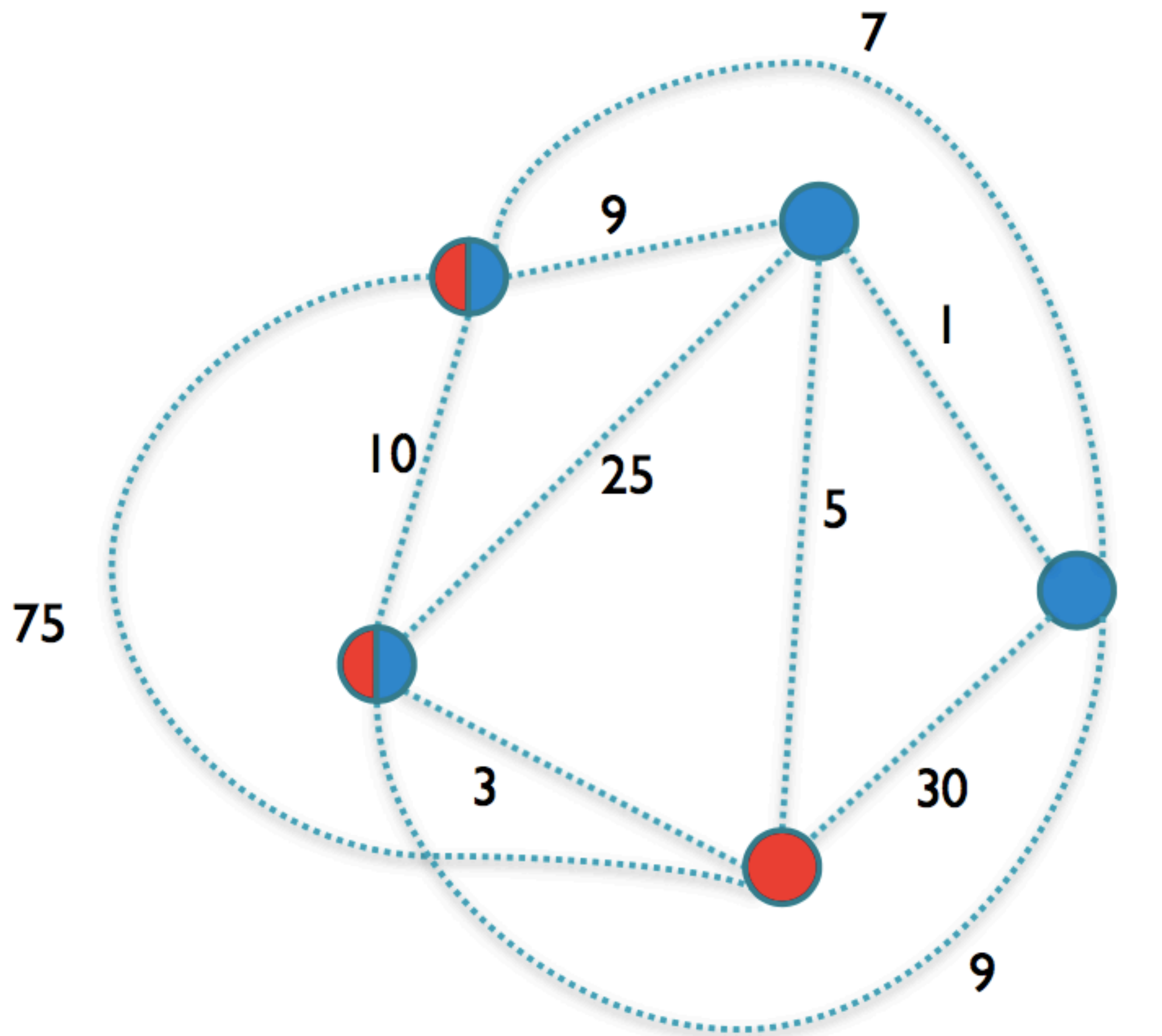
The Constraints

- Let **p(u,v)** be the a priori probability of an edge between nodes u and v.

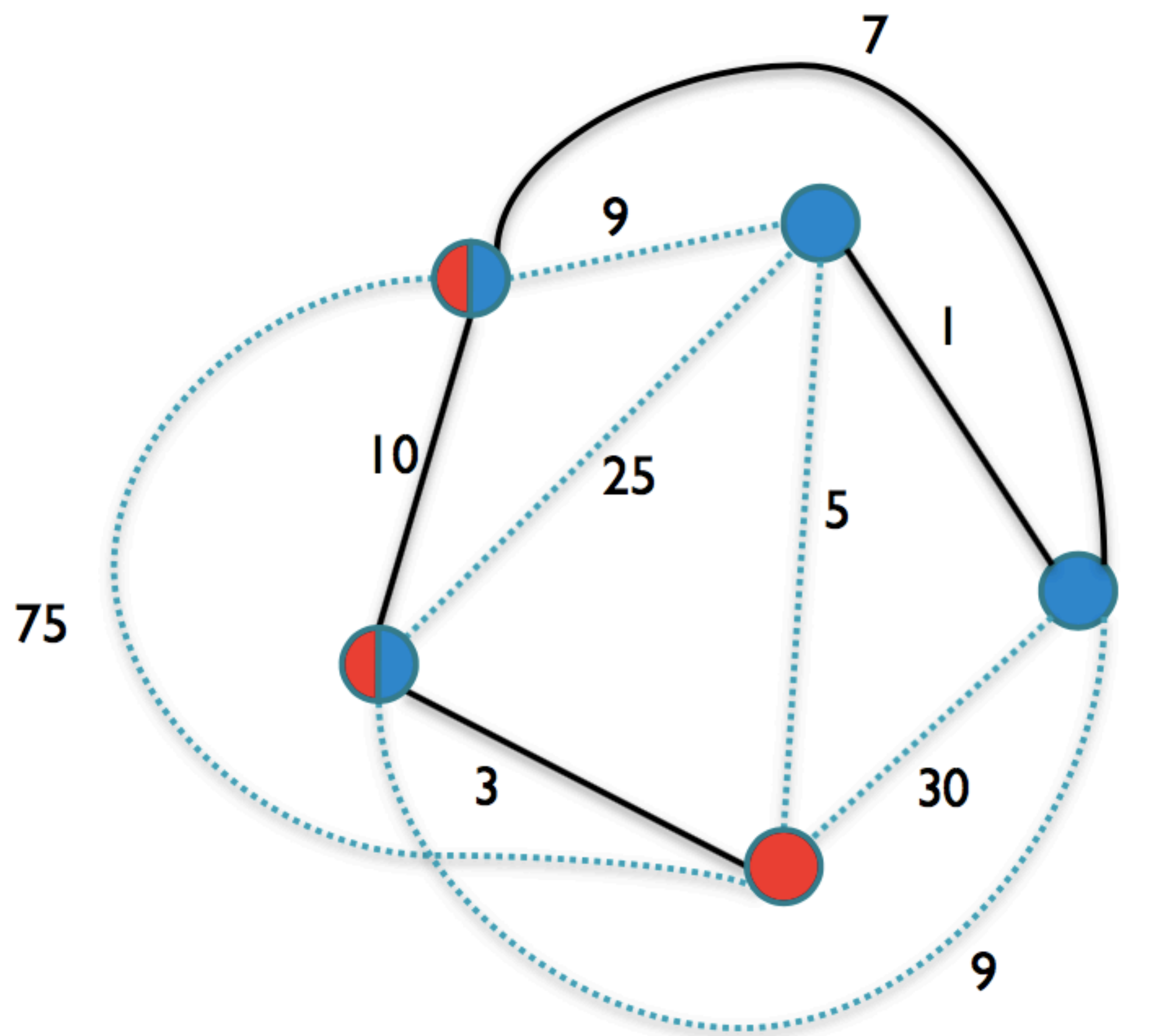
- If the prior distribution is **independent** (and probabilities are small), the maximum likelihood social network maximizes $\prod_{u,v \in V} p_{(u,v)}$.

- This is equivalent to satisfying the connectivity constraints while **minimizing the sum** of the log-likelihood costs $\sum_{v,u \in V} -\log(p_{(u,v)})$.

Finding the Cheapest Network Consistent with the Constraints



Finding the Cheapest Network Consistent with the Constraints

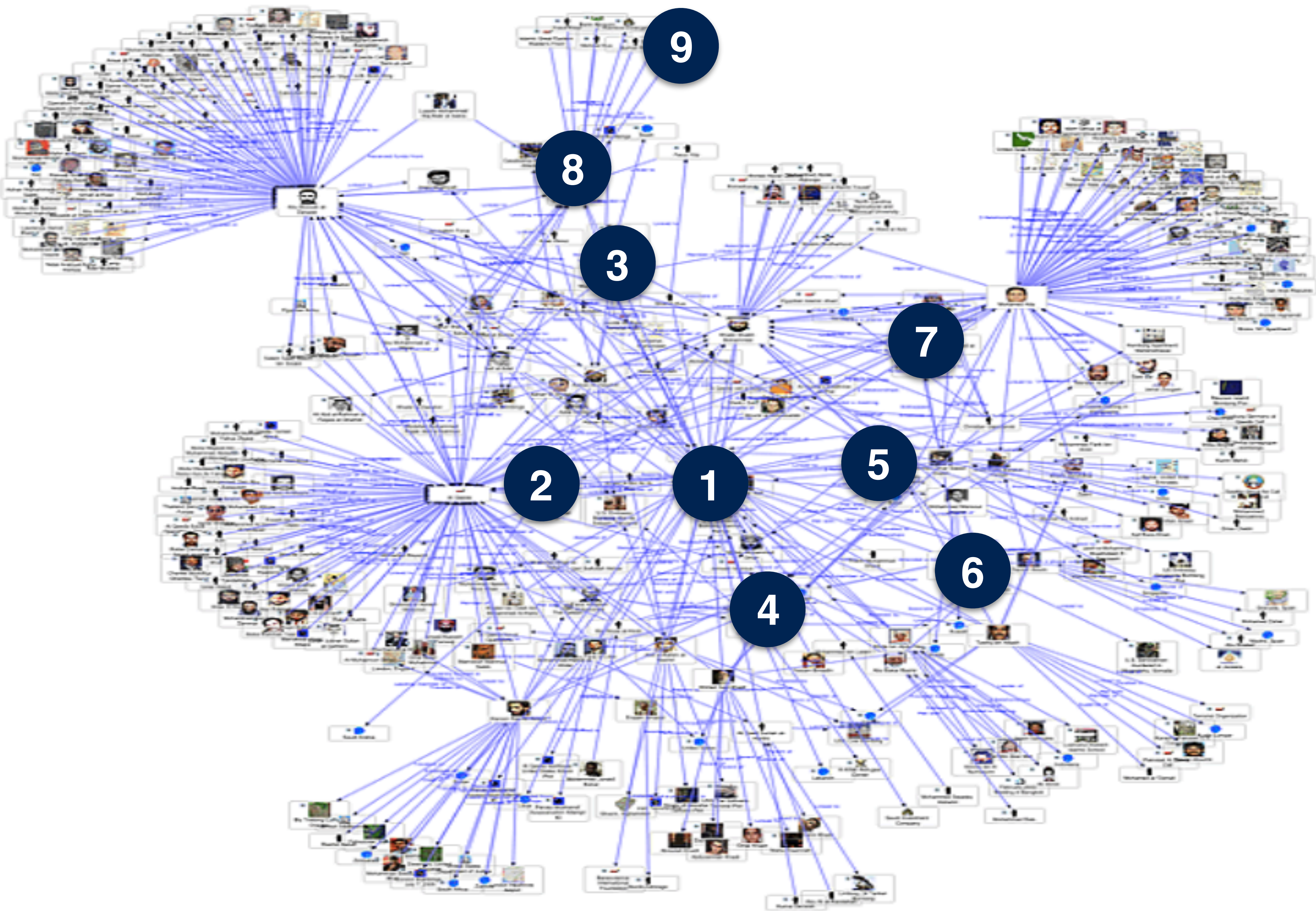


Our Network Inference Problem (aka Network Construction)

- **Given**:
  - vertices: $V = \{v_1,\ldots,v_n\}$
  - costs: c_e for each edge $e=\{v_i,v_j\}$
  - constraints: $S = \{S_1,\ldots,S_r\}$, with

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

Ordered Constraints



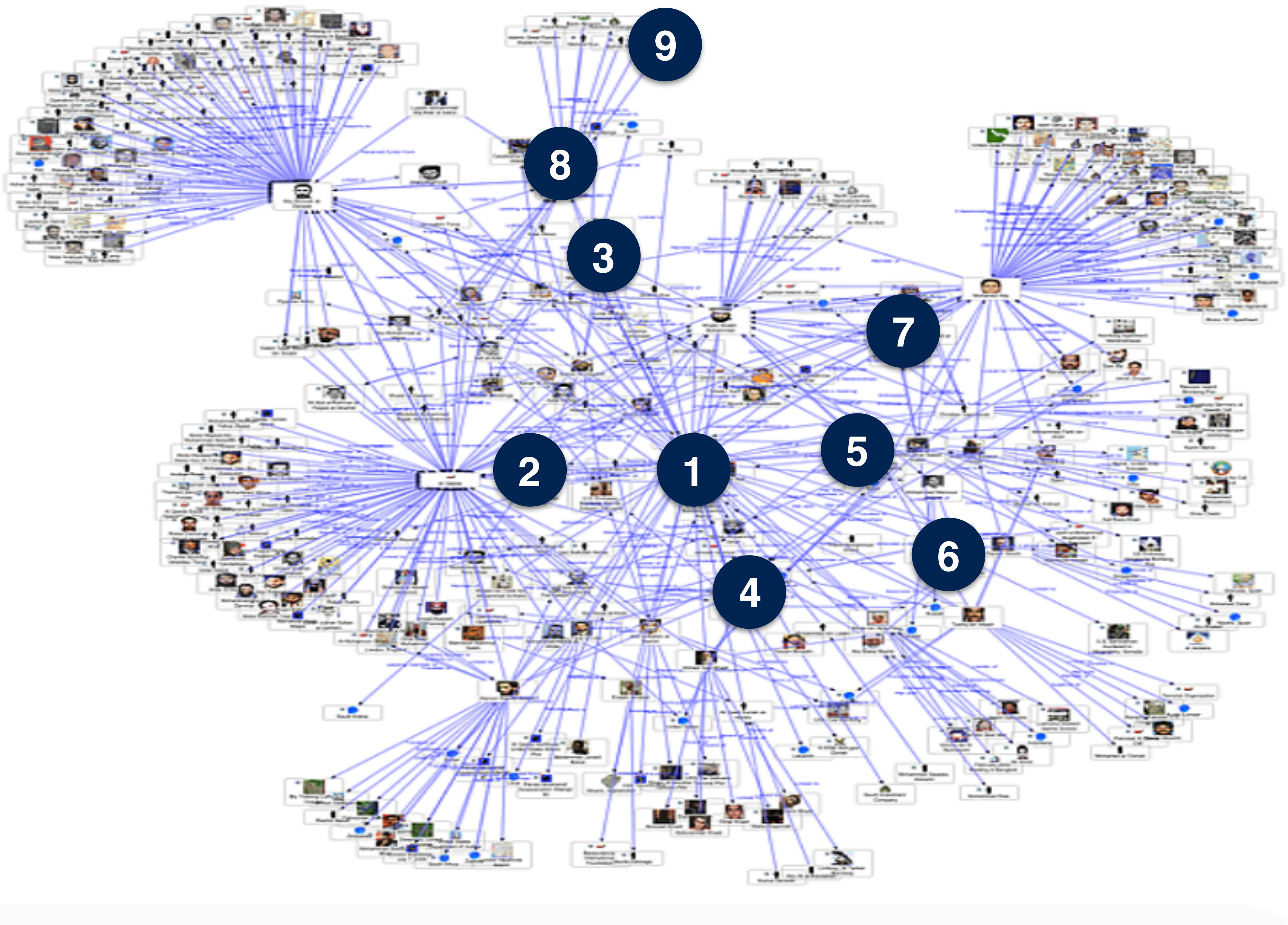
Ordered Constraint Model

- **an ordered constraint O** is an ordering on a subset of V of size $s > 1$. The ordered constraint $O = (v_{k1},...,v_{ks})$ is satisfied if for any $2 \leq i \leq s$, there exists an $1 \leq j < i$ such that the $e = \{v_{kj},v_{ki}\}$ is included in the solution.

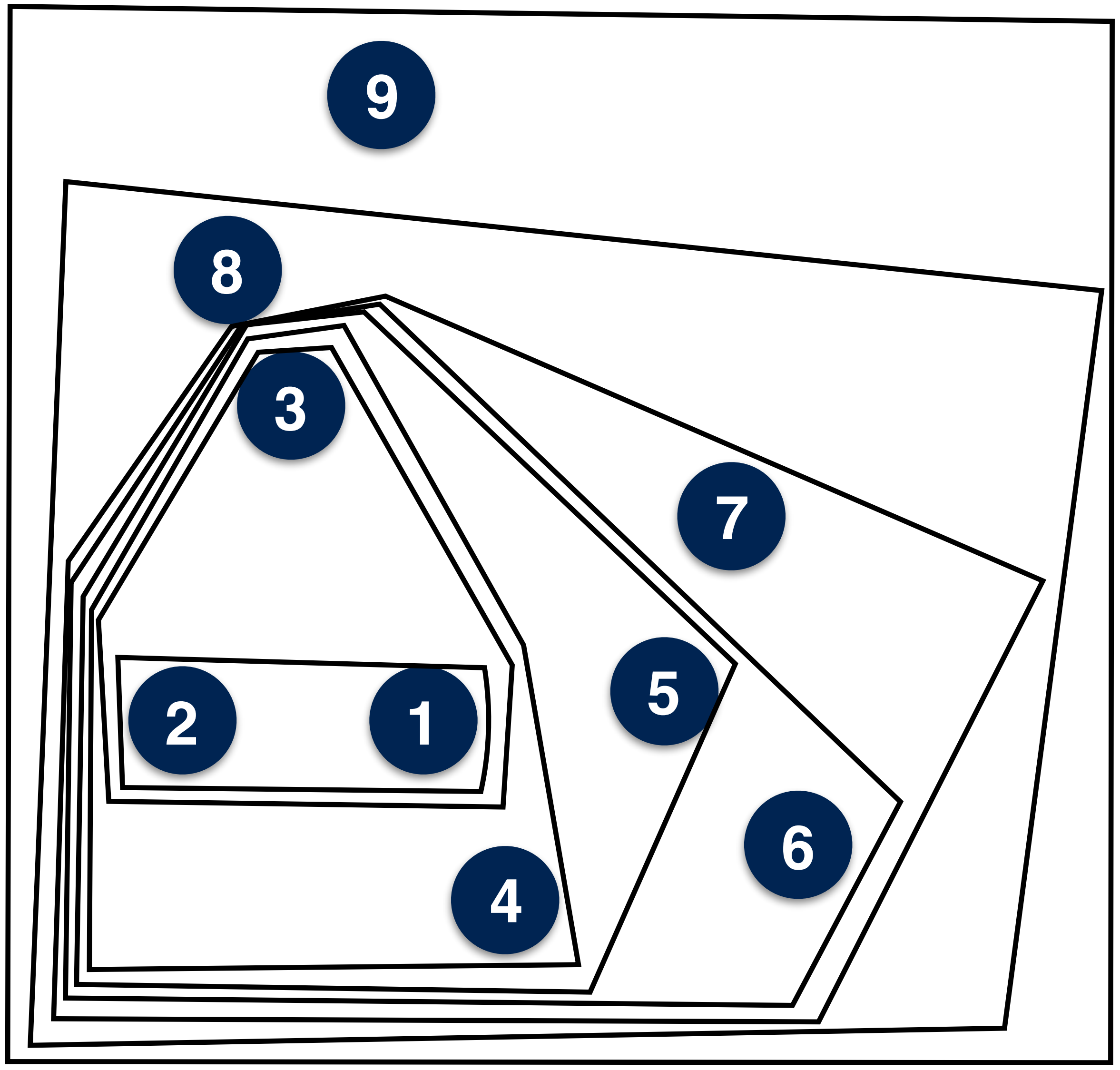
- **goal**: given a set of vertices, and edge costs, find a set E of edges of lowest cost that satisfies all the ordered constraints.

- **notice**: ordered constraints are a special case of the subgraph constraints model

Ordered Constraints



Ordered Constraints are a special case



Results for Offline Problems

- For both problems, if $P \neq NP$, we have **Ω(log n)** hardness of approximation.
  - proofs reduce from Hitting Set

- For both problems, there exist polynomial time **O(log r + log n)**-approximation algorithms, where r is the number of constraints.
  - greedy algorithm minimizing a potential function

Online Version of the Problem

- Subgraph or ordered constraints, S_i or O_i, respectively, 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.
- Can consider **adaptive** or **oblivious** adversaries.

Fun Ideas

- An **$O(n^{2/3} \log^{2/3} n)$-competitive algorithm**: Initially, place a random graph w/ $p = c\, n^{-1/3} \log^{2/3} n$. Then place a clique on any unsatisfied constraint.

- Outline of analysis:
  - all constraints Si, $|S_i| \geq n^{1/3}\log^{1/3}(n)$ are <u>almost surely satisfied</u>.
  - For all constraints S_i, $|S_i| < n^{1/3} \log^{1/3}(n)$ that are not already covered, the <u>added clique hits at least 1 edge in OPT</u>.
  - We used $O(n^{5/3}\log^{2/3}(n)+n^{2/3}\log^{2/3}(n)\mathrm{OPT})$ edges in expectation. QED.

Less fun ideas

- **Algorithm**: we first solve <u>fractional version</u> of this problem via a <u>multiplicative updates</u> method. Then we use clever <u>rounding scheme</u> to integral solutions. [Techniques from Alon et al. ('06) and Buchbinder-Naor ('09).]
  - (less fun because the math gets messy)

- Against oblivious adversaries, gives **O((log r+log n) log n)-competitive** algorithm for **ordered case**.

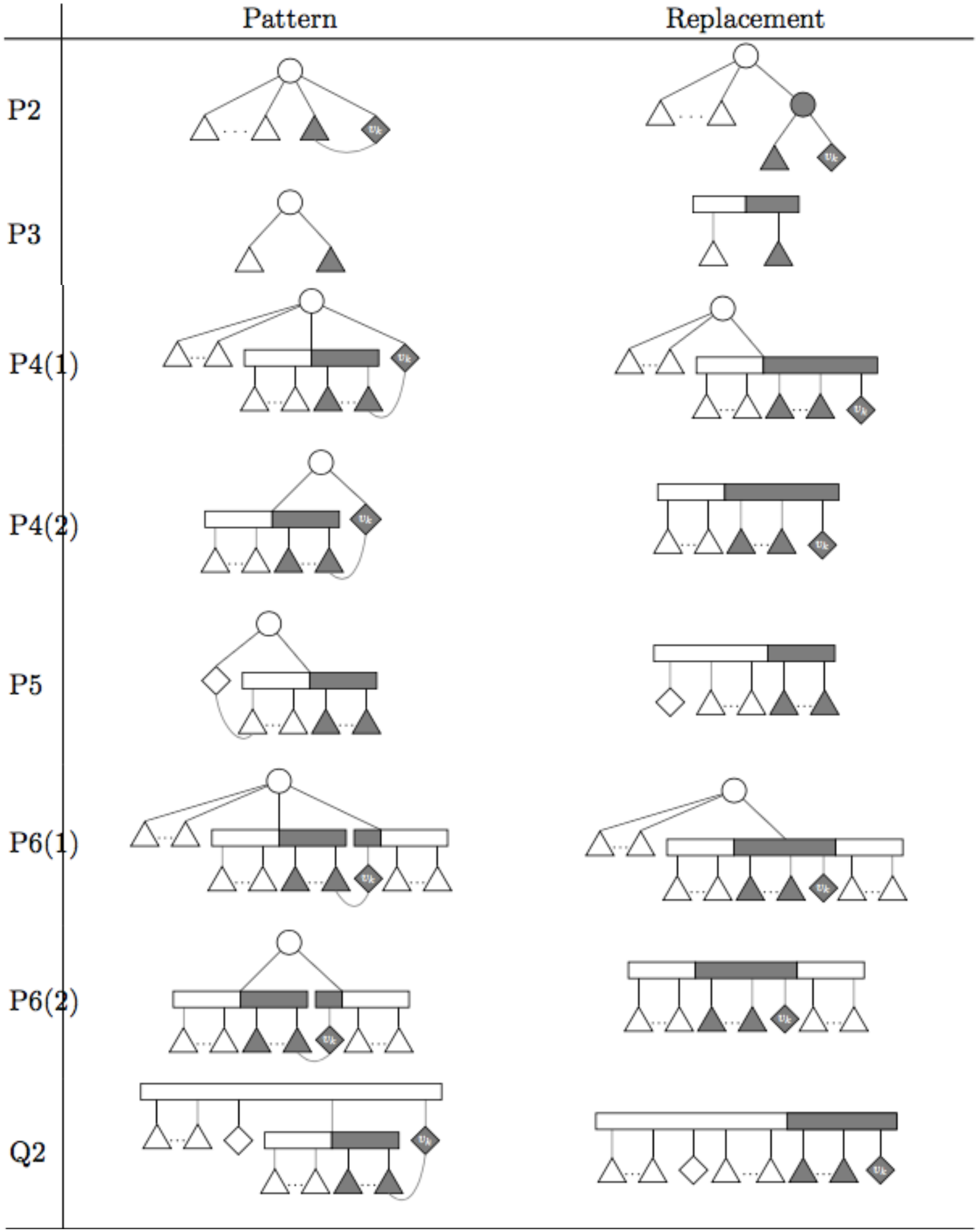
- There are also a **Ω(log n)-competitive lower bound** against oblivious adversaries.

Special Graph Structures

- when OPT is known to be a **star** and costs are uniform:
  - optimal ratio is **Ω(log(n)) in general** case
  - optimal ratio is **3/2 in ordered** case**.**

- when OPT is known to be a **path** and costs are uniform:
  - optimal ratio is **Ω(log(n)) in general** case
  - optimal ratio is **2 in ordered** case**.**

PQ-trees (not fun at all)



$$\phi(T) = a \sum_{p \in P} c(p) + b|P| + c|Q|$$

	$\sum_{p\in P} c(p)$	$\|P\|$	$\|Q\|$	$-\Delta\Phi$	number of edges added
P2	1	1	0	$-a-b$	1
P3	−2	−1	1	$2a+b-c$	0
P4(1)	−1	0	0	a	1
P4(2)	−2	−1	0	$2a+b$	1
P5	−2	−1	0	$2a+b$	1
P6(1)	−1	0	−1	$a+c$	1
P6(2)	−2	−1	−1	$2a+b+c$	1
Q2	0	0	−1	c	1
Q3	0	0	−2	$2c$	1

Summary

- Learning from constraints is just one **formalization** of a social network learning problem.
- Almost no matching bounds — **theory problems open** for pretty much every regime.
- Lots of **data to try out these models** on.
  - E.g. future work to experiment Twitter RT data and to test algorithms in practice.