

Decision making under uncertainty

- Robust optimization
 - Find decisions that work well regardless θ
- ➔ • Active learning/information gathering
 - Acquire more information about θ
- Machine learning
 - Use other data sources to estimate θ

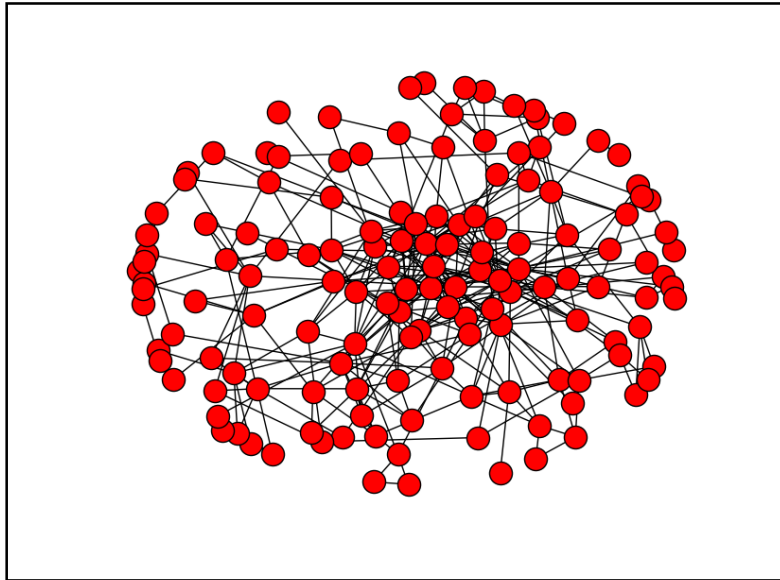
Information gathering

- What if we have the ability to collect more data?
 - Ask two people if they're friends
 - Test for organ donor compatibility
- Collecting data is usually expensive
- Want to balance cost vs quality of decision
- Two examples: influence maximization and matching

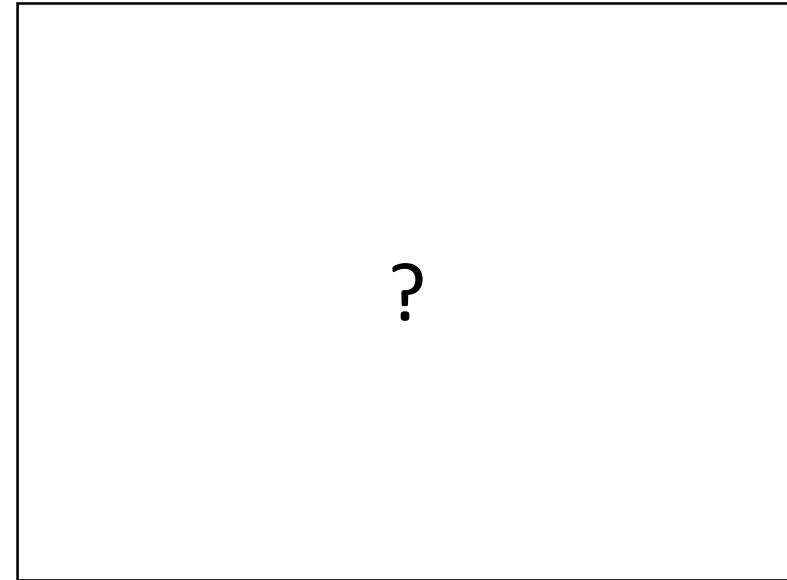
Influence maximization in the field

- Gathering network data requires in-person surveys, week+ of effort
- Approach: information gathering via network sampling

Assumed starting point



Real starting point



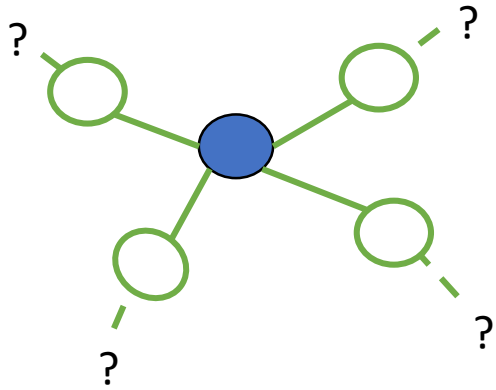
Where does the network come from?

- Data collection is costly and time consuming
 - Digital sources are often inaccurate or missing
 - Week+ for social workers to interview 100 or more people

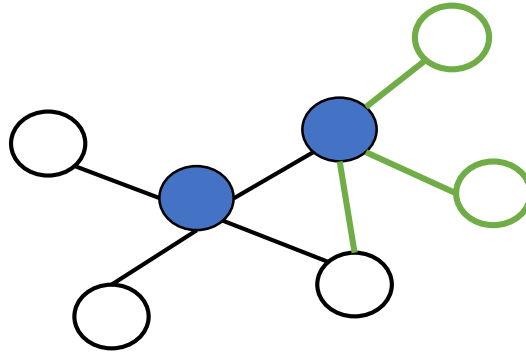
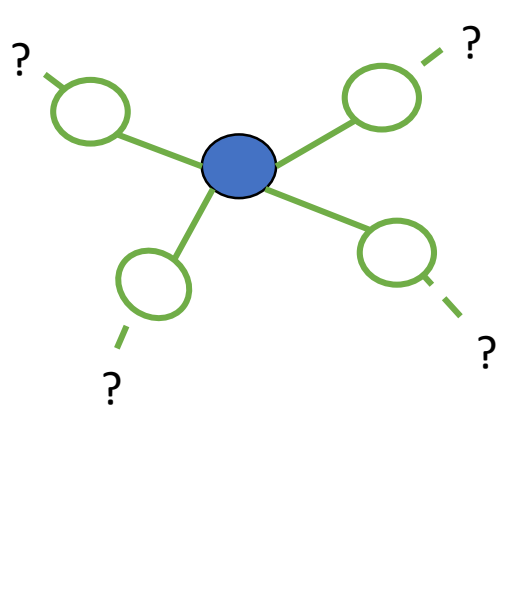
Where does the network come from?

- Data collection is costly and time consuming
 - Digital sources are often inaccurate or missing
 - Week+ for social workers to interview 100 or more people
- Do we really need to gather the entire network?

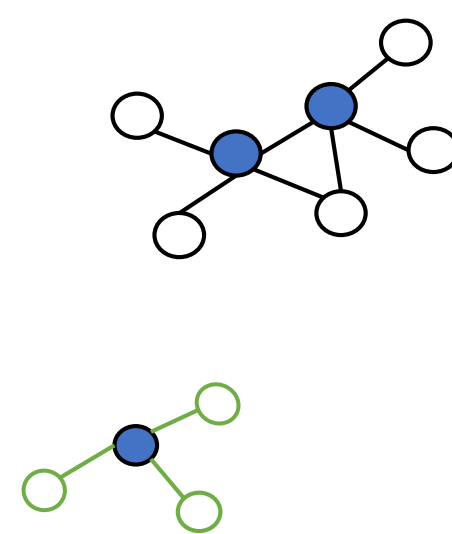
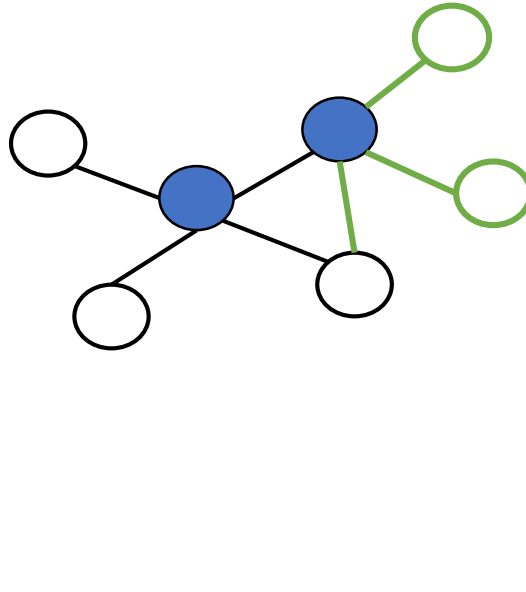
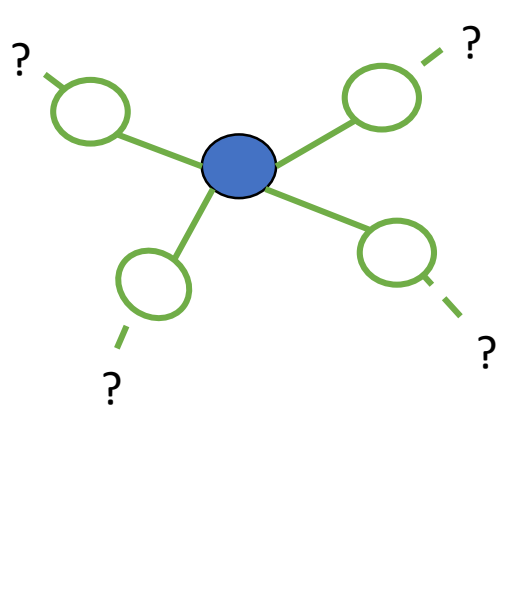
Network sampling



Network sampling



Network sampling



Objective

- Query cost: how many nodes were surveyed?
 - Should grow very slowly with n
- Influence spread: what is the expected number of nodes reached?
- Comparison to OPT , best influence spread by algorithm which sees entire network

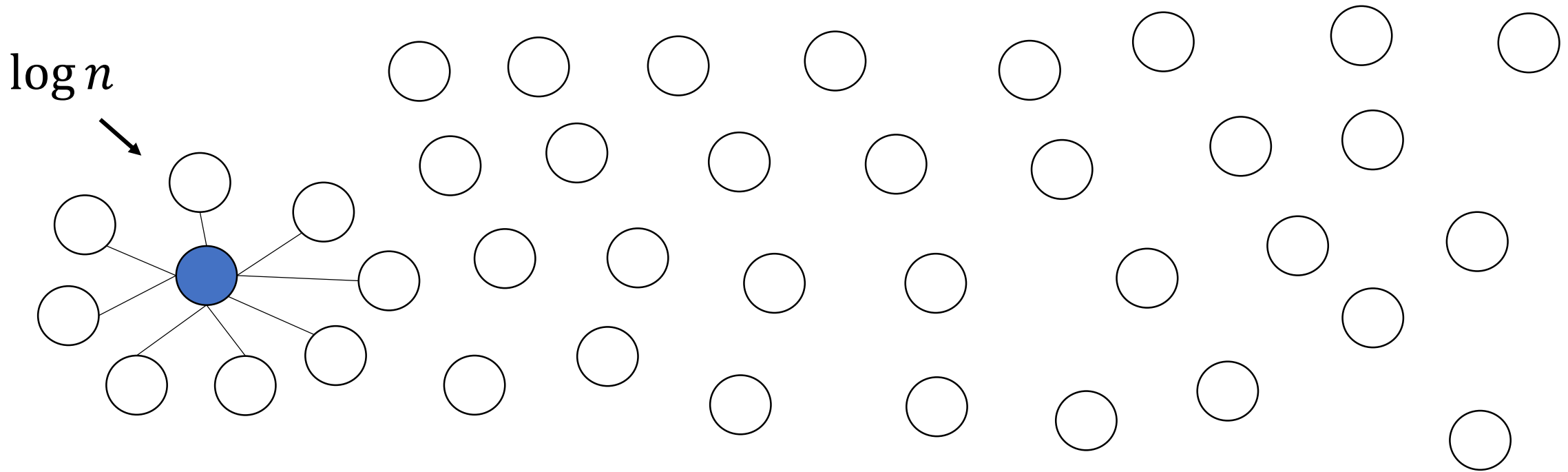
$$\text{approx. ratio} = \frac{E[\text{algorithm's influence spread}]}{OPT}$$

Hardness

Theorem: *There is a family of graphs on which any algorithm with strictly sublinear query cost has approximation ratio tending to 0 as $n \rightarrow \infty$*

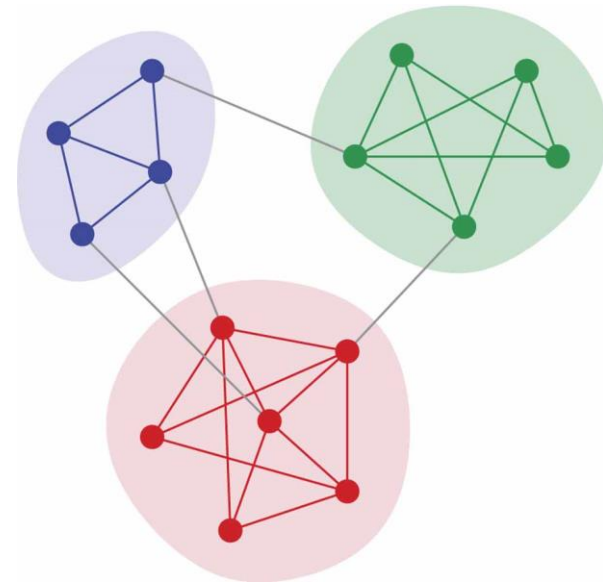
Hardness

Theorem: *There is a family of graphs on which any algorithm with strictly sublinear query cost has approximation ratio tending to 0 as $n \rightarrow \infty$*



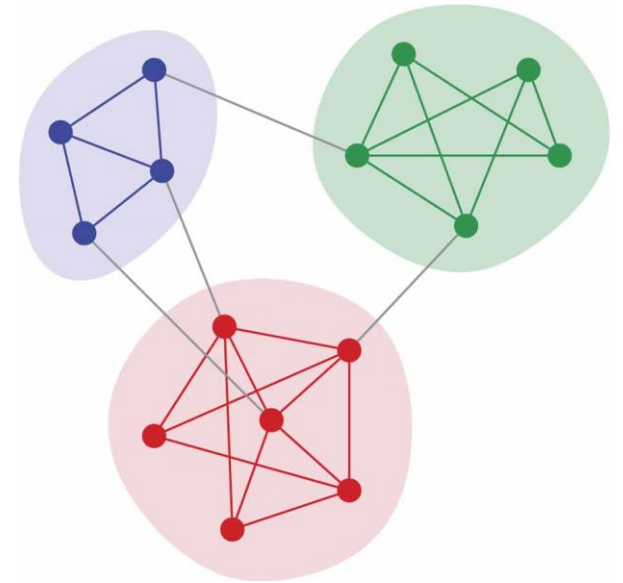
What now?

- Real networks have useful structure
- Here: two examples
 - Community structure
 - Friendship paradox



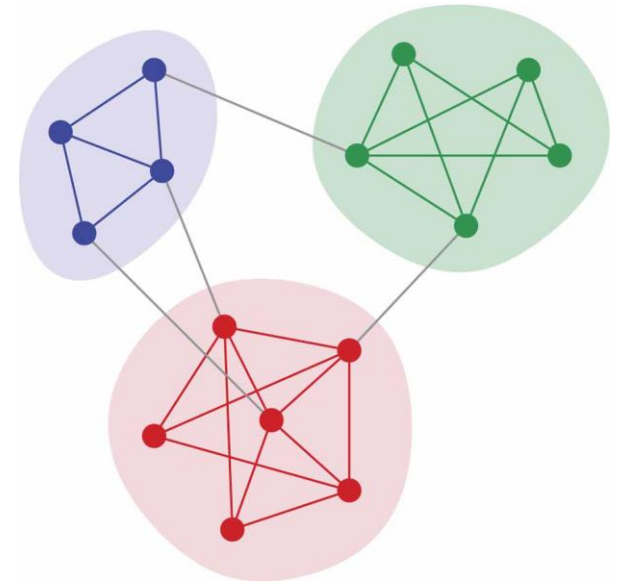
Community structure

- Intuition: influence mostly spreads locally, within communities
- We'd like to put one seed in each of the largest k communities



Community structure

- ARISEN algorithm repeatedly:
 - Randomly samples a node
 - Explores that node's neighborhood via a random walk
 - Estimates the size of that node's community
- And then seeds nodes that correspond to largest k communities



Community structure

Theorem: For community-structured graphs, ARISEN obtains a constant-factor approximation to the optimal influence spread using $\text{polylog}(n)$ queries.

Bryan Wilder, Nicole Immorlica, Eric Rice, Milind Tambe. Maximizing influence in an unknown social network. AAAI 2018.

Community structure

Theorem: For community-structured graphs, ARISEN obtains a constant-factor approximation to the optimal influence spread using $\text{polylog}(n)$ queries.

Asymptotically: exponential improvement over exhaustive surveys!

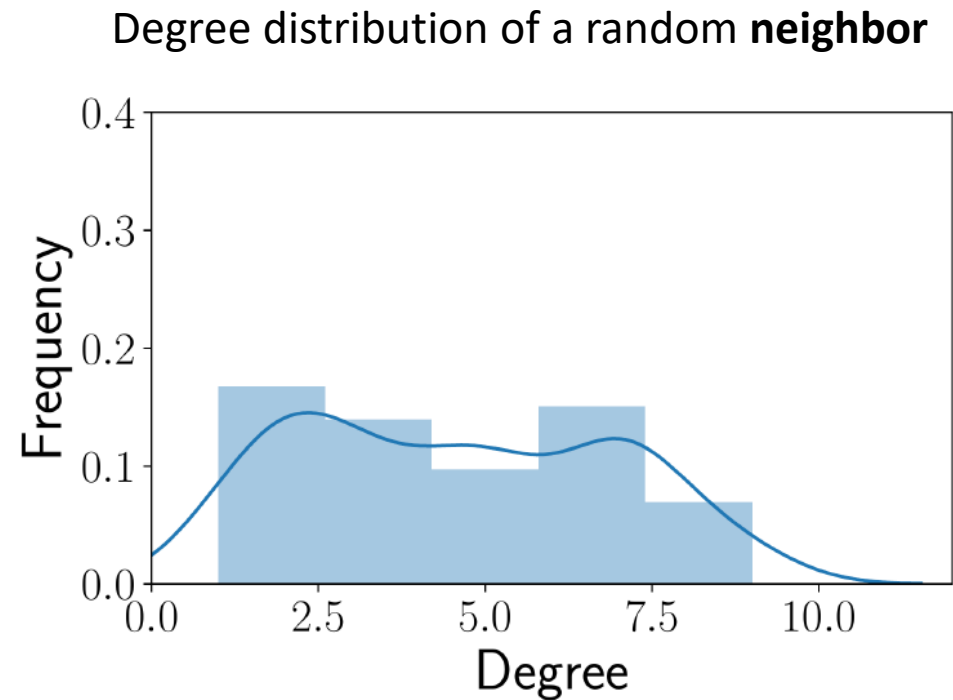
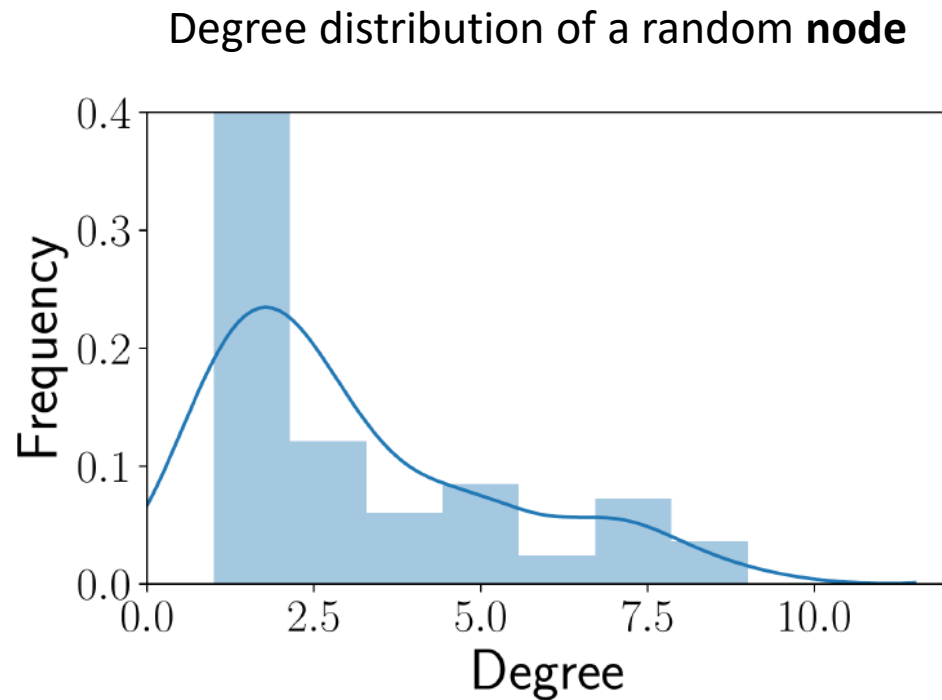
Bryan Wilder, Nicole Immorlica, Eric Rice, Milind Tambe. Maximizing influence in an unknown social network. AAAI 2018.

Community structure

- Downside: difficult to implement in some settings
- Homeless youth: can't find a series of 5-10 youth to simulate a random walk

Friendship paradox

- On average, your friends are more popular than you

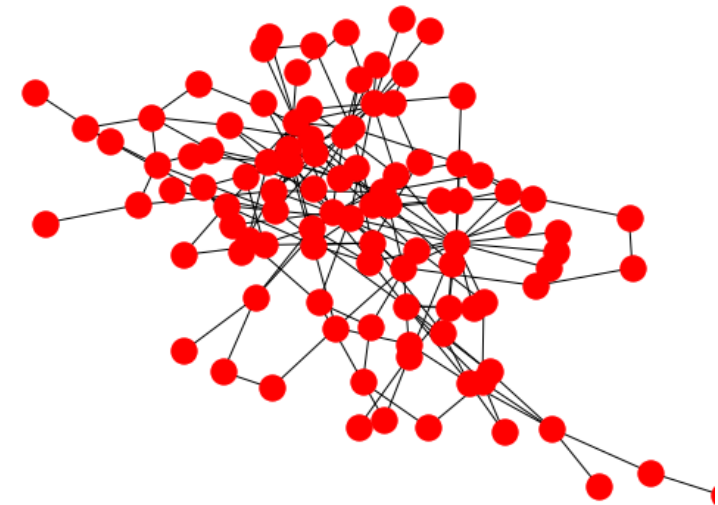


Friendship paradox

- Repeatedly
 - Survey a random node
 - Survey one of its neighbors
- First step encourages diversity, second biases towards high-degree/central nodes

Case study: HIV and homelessness

- Shelters conduct educational interventions
- Resource constraints: work with 4-6 youth at a time
- *Peer leaders*: spread message through social network



Recap: HIV and homelessness

- Influence maximization is well studied, but new challenges in the field
 - How will influence spread across the network?
 - How do we get network data in the first place?

Recap: HIV and homelessness

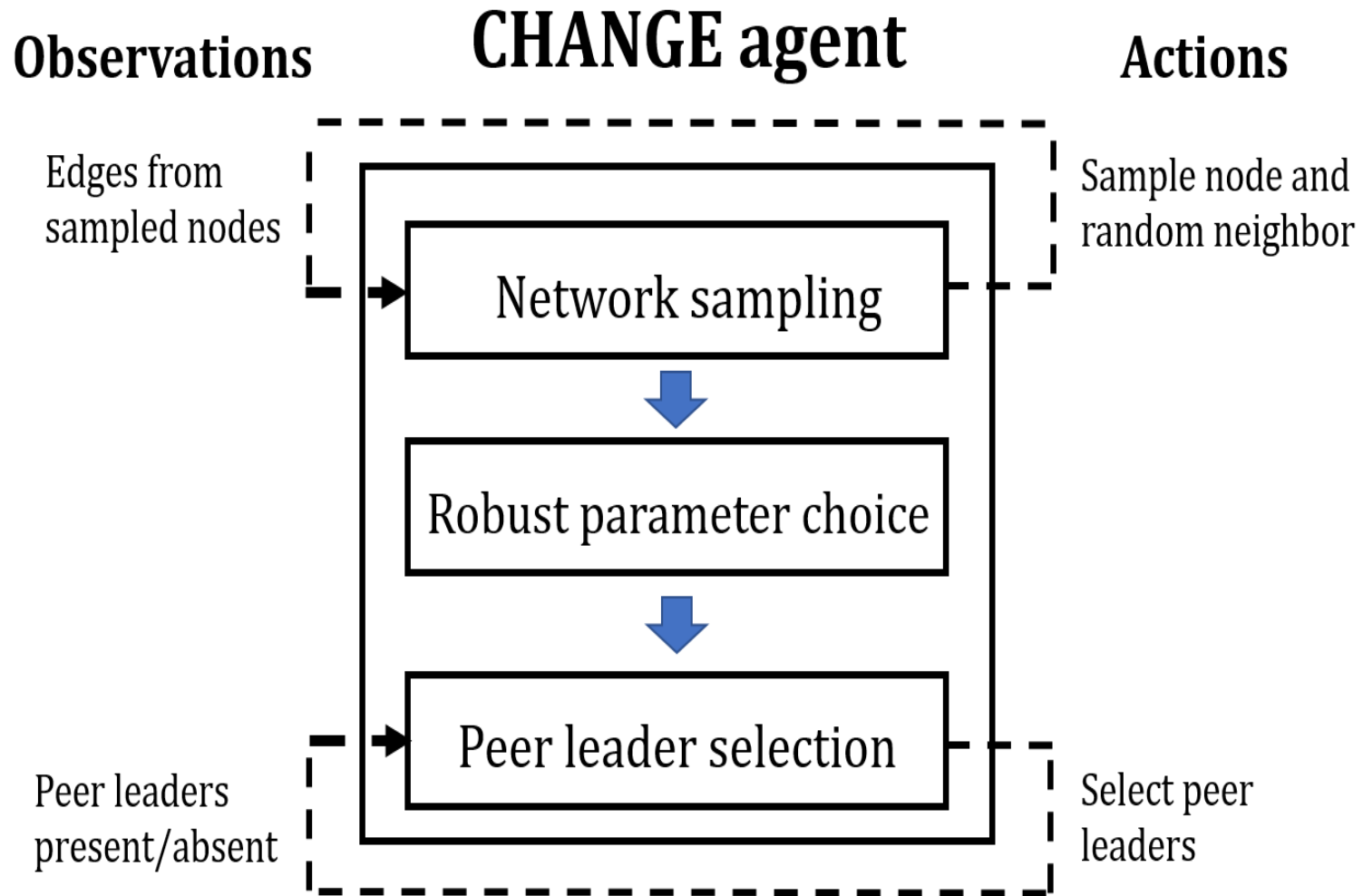
- Influence maximization is well studied, but new challenges in the field
 - How will influence spread across the network?
 - How do we get network data in the first place?
- Proposed algorithmic solutions
 - Modeling uncertainty → robust optimization
 - Cost of gathering network data → information gathering via subsampling

Recap: HIV and homelessness

- Influence maximization is well studied, but new challenges in the field
 - How will influence spread across the network?
 - How do we get network data in the first place?
- Proposed algorithmic solutions
 - Modeling uncertainty → robust optimization
 - Cost of gathering network data → information gathering via subsampling

Putting it all together

- Combine these ideas into a single system which works in the field
- Needs to minimize need for data, expertise, resources
- Needs to handle domain-specific challenges
 - Homeless youth: peer leaders often don't attend intervention



Bryan Wilder, Laura Onasch-Vera, Juliana Hudson, Jose Luna, Nicole Wilson, Robin Petering, Darlene Woo, Milind Tambe, Eric Rice. End-to-End Influence Maximization in the Field. AAMAS 2018.

Field study

Deployment in collaboration with social work and community partners





- Recruit 60 youth
- Survey social network

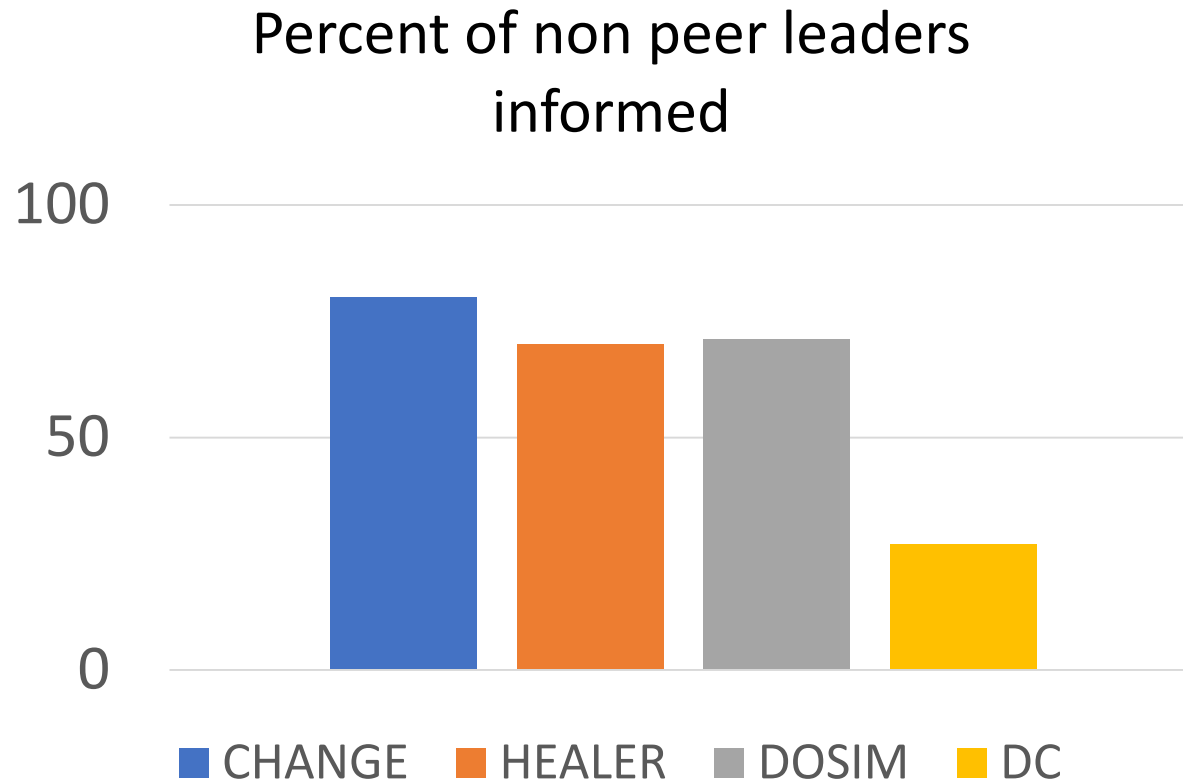
- Train 10-12 peer leaders
- Over 3 interventions

- 1 month: follow-up with all 60
- See who received information

Comparison

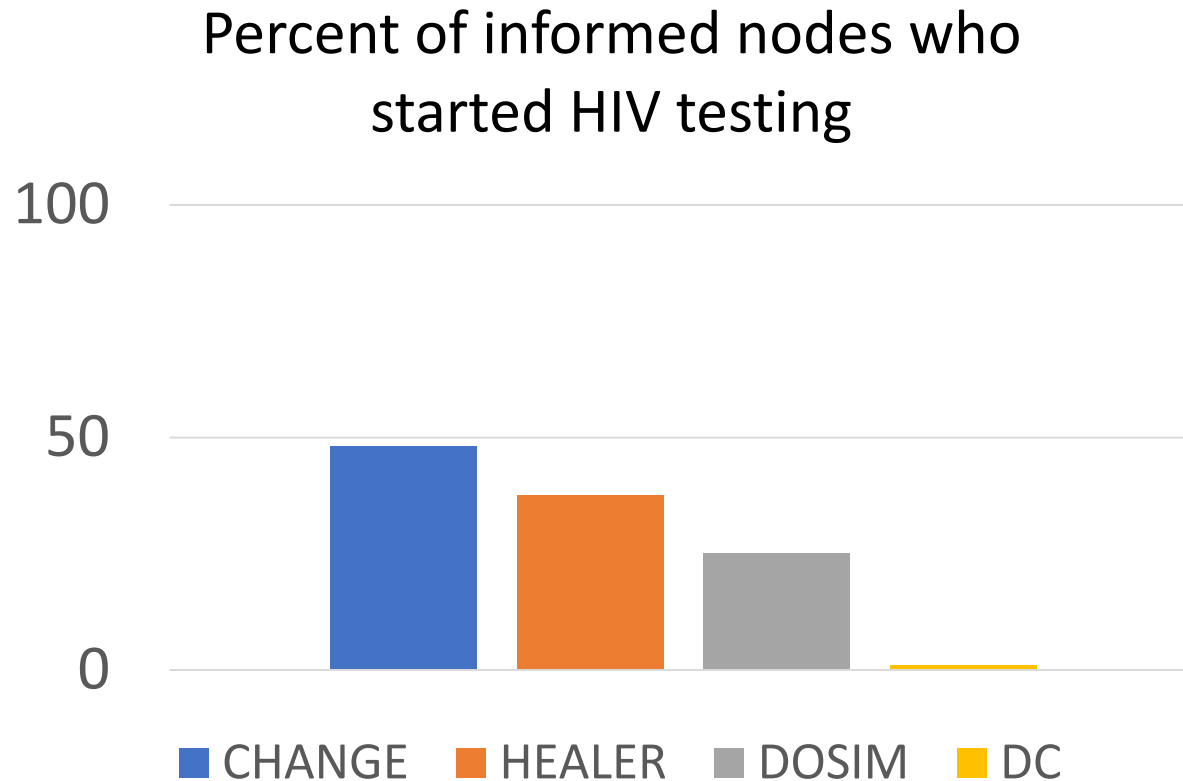
- Conducted (so far) 4 studies, each with different algorithm
- Status quo: degree centrality (DC)
- AI-based algorithms: CHANGE, HEALER, DOSIM
 - CHANGE only surveys ~20% of nodes
 - HEALER and DOSIM survey 100%

Results: information spread



- AI-based algorithms dramatically outperform status quo (**27% → 70+%**)
- CHANGE performs comparable to HEALER/DOSIM, but surveyed only **18%** of youth!

Results: behavior change



- Information spread translates into real behavior change!
- CHANGE: comparable/slightly higher conversion rate

Summary

- Influence maximization/social network interventions offer promising means to amplify interventions
- Doesn't work out of the box (yet)
- Think carefully about models and data
- Dealing with uncertainty is critical

Decision making under uncertainty

- Robust optimization
 - Find decisions that work well regardless θ
- Active learning/information gathering
 - Acquire more information about θ
- ➔ • Machine learning
 - Use other data sources to estimate θ

Predicting Outcomes: Techniques and Perils

Sanmay Das

Optimization & Learning Approaches to Resource Allocation
for Social Good (Tutorial @ AAMAS 2019)

Improving Allocations Using Predicted Outcomes

- ▶ Idea: Personalized intervention / resource allocation
- ▶ If we allocate resource α to agent A , how effective will that be in terms of the outcome we care about?
 - ▶ Bed space + counseling resources to a homeless single mother with two kids, ages 5 and 7? Household with alcoholic veteran father, mother and two children? (Kube et al, *AAAI* 2019)
 - ▶ Kidney of a 250 lb man with 2 HLA-B mismatches into a 150 lb woman of the same blood type ? (Massie et al, *Am. J. Transplantation* 2016)
 - ▶ Refugees to specific cities? “Current procedures for determining how to allocate refugees across domestic resettlement sites do not fully leverage synergies between refugees and geographic locations” (Bansak et al, *Science* 2018)

Forecasting Outcomes

- ▶ Sometimes a standard learning problem
 - ▶ Refugee assignments to cities can be randomized
 - ▶ Living donor kidney transplantation has typically always been carried out as long as recipient and donor are compatible
- ▶ However, often available data is based on current allocation policies
 - ▶ Creates serious estimation problems
 - ▶ Need: Causal / counterfactual prediction

Heterogeneous Treatment Effects

- ▶ Rubin's potential outcomes framework (Rubin, *JASA* 2005: Causal effect of treatment for i is $Y_i(1) - Y_i(0)$.
 - ▶ But we only observe one of these.
- ▶ Usual focus: Estimating average treatment effects across the population, e.g. $\sum_{i=1}^n [Y_i(1) - Y_i(0)]$, although recently also a focus on *conditional average treatment effects*
 $\sum_{i=1}^n [Y_i(1) - Y_i(0) | X_i]$
- ▶ Classic causal inference problem: Observational study. Outcomes typically not independent of decision to treat.
- ▶ Aside: Many clever empirical strategies for dealing with this in specific cases (RCTs, natural experiments, diff-in-diff, instrumental variables, and so on)
 - ▶ Not the typical "big data" setting (often administrative data without convincing instruments or good natural experiments)

Ignorability and Matching

- ▶ Strong ignorability: Let Z be the treatment, Y the outcome, X the feature (covariate) vector (other than treatment)
 - ▶ Unconfoundedness: Y is independent of Z conditional on X
 - ▶ Common support / overlap: $0 < \Pr(Z = 1|X) < 1$
 - ▶ Note the underlying tension: as we go to more features, we're less likely to have confounding, but also less likely to satisfy common support
- ▶ Classic technique: Propensity score matching (Rosenbaum & Rubin, *Biometrika* 1983)
 - ▶ Estimate $\Pr(Z = 1|X)$ (propensity score) using logistic regression
 - ▶ Match each treated individual to one or more untreated using the propensity score
 - ▶ Checking for propensity score balance and covariate balance, differences in matched treatments and controls should be because of the treatment

Enter the Dragon (ML)

- ▶ Basic idea: Estimate the “response surfaces”
 $E[Y|X, Z = 0], E[Y|X, Z = 1]$.
- ▶ Key difference from prior work: Do not explicitly estimate the propensity to receive treatment.
- ▶ Intuition: If ML is really good at prediction, can we just treat this as a prediction problem?
- ▶ We are good at dealing with lots of features (so can maybe throw all confounders in there).
- ▶ Any flexible model for the response surface should work
- ▶ We'll consider BART (Chipman, George, and McCulloch, *NeurIPS 2007, Annals Appl. Stat 2010*, Hill *J. Comp. Graph. Stat. 2011*) and discuss causal forests (Wager & Athey, *JASA 2018*)

Bayesian Additive Regression Trees

- ▶ A flexible Bayesian model that easily captures interactions and nonlinearities (inspired by boosting).
- ▶ Sum of trees (T_j, M_j) where T_j is the tree structure (with binary splits at each interior node) and M_j is a vector of the predicted values at the leaves.
- ▶ A regularization prior.
 - ▶ Main effect: encodes preference for short individual trees (e.g. trees with 1, 2, 3, 4 \geq 5 terminal nodes receive prior probability of 0.05, 0.55, 0.28, 0.09, 0.03 respectively).
 - ▶ Normal priors on each value in M_j (encodes preference towards 0 values)
 - ▶ A prior on the variance of the error term σ
- ▶ Number of trees m is usually pre-set (200 is standard)

BART Algorithm

- ▶ Start by initializing m single node trees
- ▶ Gibbs sampler: At each iteration, consider changing one tree.
Need to sample from the posterior
 - ▶ $(T_j, M_j) | T_{(j)}, M_{(j)}, \sigma, \text{data}$ where $T_{(j)}, M_{(j)}$ is the ensemble other than tree j
 - ▶ T_j sampler is the complex part, but can be done using the following proposal algorithm based on the current tree:
growing a terminal node (0.25), pruning a pair of terminal nodes (0.25), changing a nonterminal rule (0.40), and swapping a rule between parent and child (0.10).
- ▶ Caveat: In our experience, if you want reasonable estimates of the posterior distribution, it is important to thin the draws from the sampler (50 works for us) to avoid autocorrelation problems.

Causal Forests

- ▶ Typical decision tree algorithm: Recursive greedy choice of splitting feature (typically combined with pruning)
- ▶ Bagging: Build an ensemble by training m trees on bootstrap samples of the training data
- ▶ Random forests: Decorrelate by only allowing a random subset of features to be chosen for splitting at each node
- ▶ Causal inference intuition: Leaves should be small enough (and therefore matched enough) that we can think of the (Y_i, Z_i) pairs as coming from a randomized experiment
- ▶ Athey and Wager show you can prove a lot of nice things about treatment effect estimation if trees are *honest*.

Honest Trees

- ▶ Honesty: Splitting rule must not inappropriately incorporate information about outcomes Y_i . Formally, in any tree Y_i can only be used to either estimate the treatment effect, or to decide where to place splits, not both.
- ▶ Two methods:
 1. Double-sample trees: Randomly subsample the data, divide into two equal halves, use one to grow the tree and the other to estimate responses at the leaves.
 2. Propensity trees: Randomly subsample the data, use Z_i to place the splits, but Y_i to estimate responses at the leaves.
- ▶ Note that these are subsampled without replacement, as opposed to the standard with replacement paradigm in random forests.

Example: Treatment Effects of Prevention and Rapid Rehousing

- ▶ Preliminary, unpublished work, with all the cautions inherent in that!
- ▶ Two homelessness interventions: Prevention and RRH were both introduced as part of the ARRA during the financial crisis
- ▶ Prevention is one of the lowest cost interventions: Cash, network of referrals
- ▶ Seems effective, but population receiving the treatment is very different than others (more likely to have been renting or owning their own homes in the immediate past; higher incomes)
- ▶ Definitions of what constitutes homelessness (criterion for treatment) are slippery
- ▶ Rapid rehousing: Place to stay and way to pay for it, but without support (counseling, etc)

Summary Statistics and Feature Information

Service Type	Number Assigned	Percent Reentered
Emergency Shelter	2897	56.20
Transitional Housing	1927	40.22
Rapid Rehousing	589	53.48
Homelessness Prevention	2061	24.16
Total	7474	43.03

Type	Number	Examples
Binary	3	Gender, Spouse Present, HUD Chronic Homeless
Other Categorical	61	Veteran, Disabling Condition, Substance Abuse
Continuous	4	Age, Income, Calls to Hotline, Duration of Wait
Total Features	68	

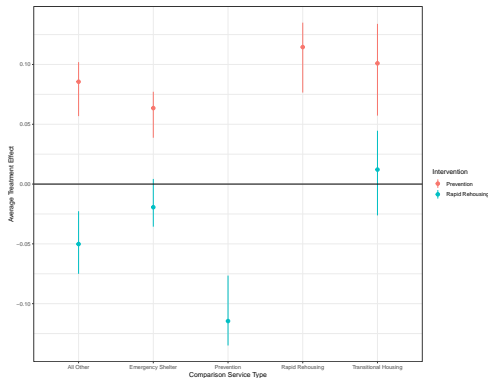
Summary Statistics and Feature Information

Service Type	Number Assigned	Percent Reentered
Emergency Shelter	2897	56.20
Transitional Housing	1927	40.22
Rapid Rehousing	589	53.48
Homelessness Prevention	2061	24.16
Total	7474	43.03

Type	Number	Examples
Binary	3	Gender, Spouse Present, HUD Chronic Homeless Veteran, Disabling Condition, Substance Abuse Age, Income, Calls to Hotline, Duration of Wait
Other Categorical	61	
Continuous	4	
Total Features	68	

Treatment Effects

- ▶ Only a few hundred in prevention are not immediately coming from their own homes, and only a few hundred others are.
- ▶ Strategies: Use (1) PSM with exact matching on prior residence, income quartile; (2) BART; (3) Causal forests on only the group that matches on those two features
- ▶ BART estimates of average treatment effects for homelessness prevention and rapid rehousing



Comparing Average Treatment Effects Using BART and PSM

Test	BART Treatment Effect			PSM Treatment Effect			Sample Size
	Average	Lower 5%	Upper 95%	Average	Lower 5%	Upper 95%	
Prevention vs All other	8.56	5.68	10.20	12.00	2.28	21.72	400
Rapid Rehousing vs All other	-5.02	-7.50	-5.02	-9.28	-15.55	-3.01	970
Prevention vs Emergency Shelter	6.35	3.87	7.72	3.42	-9.47	16.31	234
Prevention vs Transitional Housing	10.10	5.72	13.40	0	-17.20	17.20	134
Prevention vs Rapid Rehousing	11.46	7.65	13.50	12.41	0.60	24.22	274
Emergency Shelter vs Transitional Housing	-4.25	-10.31	3.47	-12.34	-16.03	-8.65	2772
Rapid Rehousing vs Emergency Shelter	-1.94	-3.56	0.43	-8.40	-16.36	-0.42	596
Rapid Rehousing vs Transitional Housing	1.21	-2.62	4.46	-2.34	-12.85	8.17	342

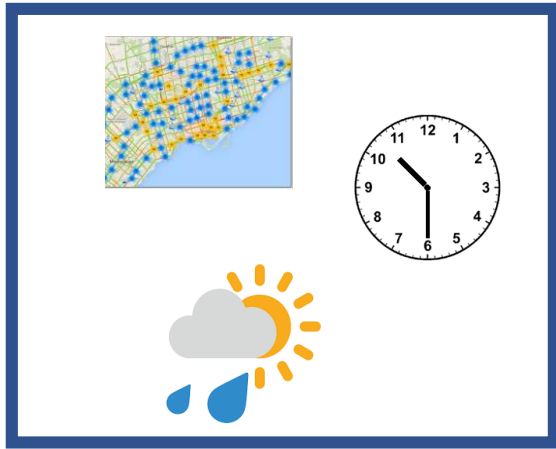
The data-decisions pipeline

Many real-world applications of AI involve a common template:

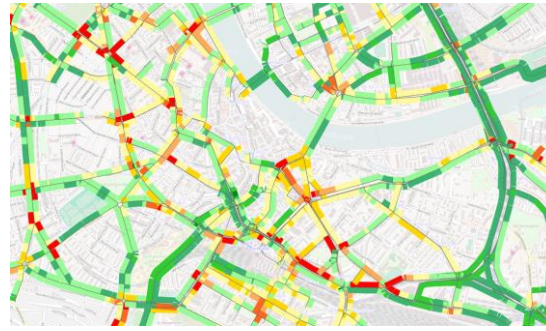
[Horvitz and Mitchell 2010; Horvitz 2010]



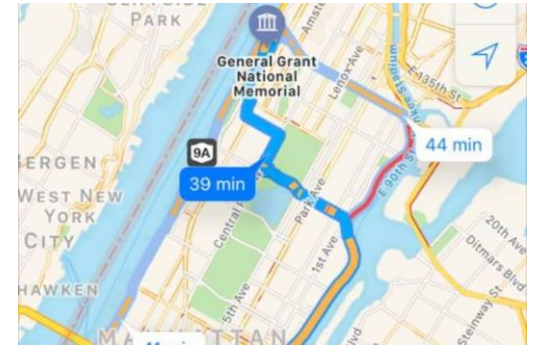
Google maps



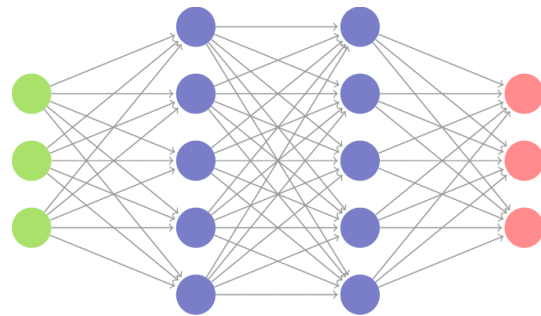
Data



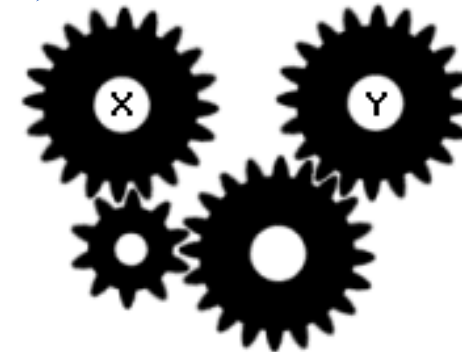
Predicted travel times



Shortest path



Predictive model



Routing algorithm



Formalization

- Objective function $f(x, \theta)$
 - $x \in \{0, 1\}^n$ is the **decision variable**
 - θ is an **unknown parameter** (e.g., true travel times)

Formalization

- Objective function $f(x, \theta)$
 - $x \in \{0, 1\}^n$ is the **decision variable**
 - θ is an **unknown parameter** (e.g., true travel times)

- Want to solve:

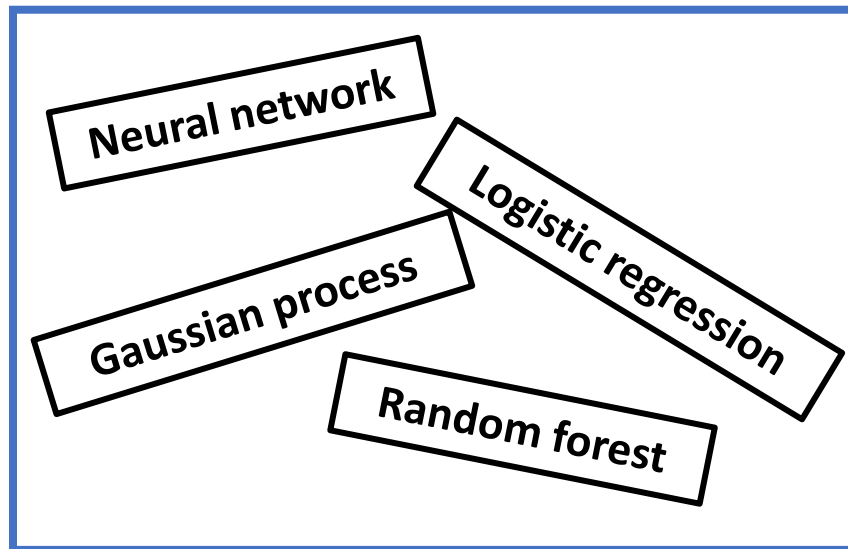
$$\max_{x \in X} f(x, \theta), \text{ where } X \subseteq \{0, 1\}^n \text{ is the feasible set}$$

Formalization

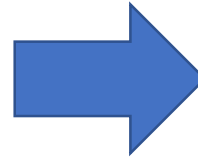
- Objective function $f(x, \theta)$
 - $x \in \{0, 1\}^n$ is the **decision variable**
 - θ is an **unknown parameter** (e.g., true travel times)
- Want to solve:
$$\max_{x \in X} f(x, \theta), \text{ where } X \subseteq \{0, 1\}^n \text{ is the feasible set}$$
- Challenge: θ is unknown and must be learned from data!

Typical two-stage approach

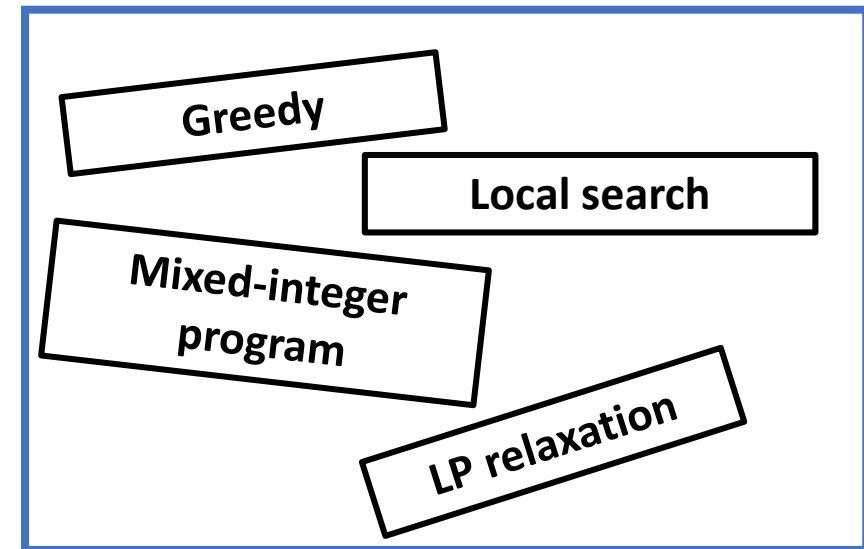
Machine learning models



Goal: maximize accuracy



Optimization algorithms



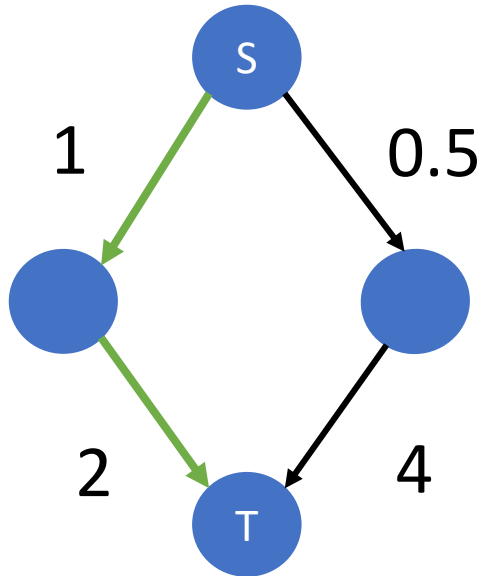
Goal: maximize decision quality

Challenge

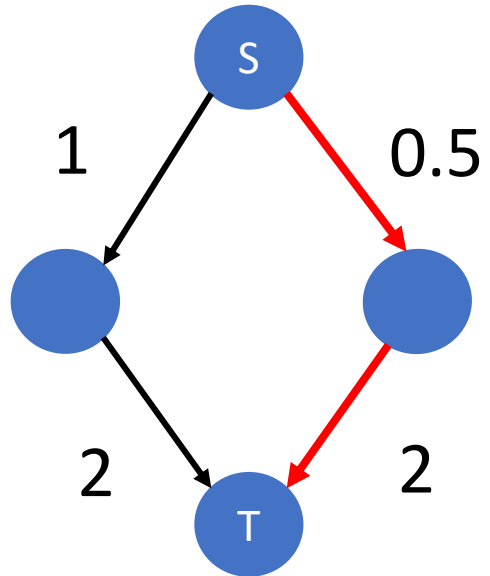
- Maximizing accuracy \neq maximizing decision quality
- “All models are wrong, some are useful”
- Two-stage training doesn't align with end goal

Example

Ground truth



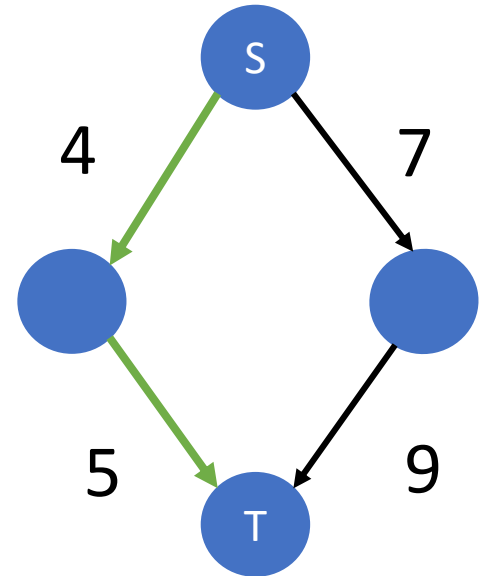
Model 1



MSE: 4

Ground truth length: 4.5

Model 2



MSE: 21.31

Ground truth length: 3

This work

Automatically shape the model's loss by incorporating the optimization problem into the training loop

Application: resource allocation for tuberculosis treatment

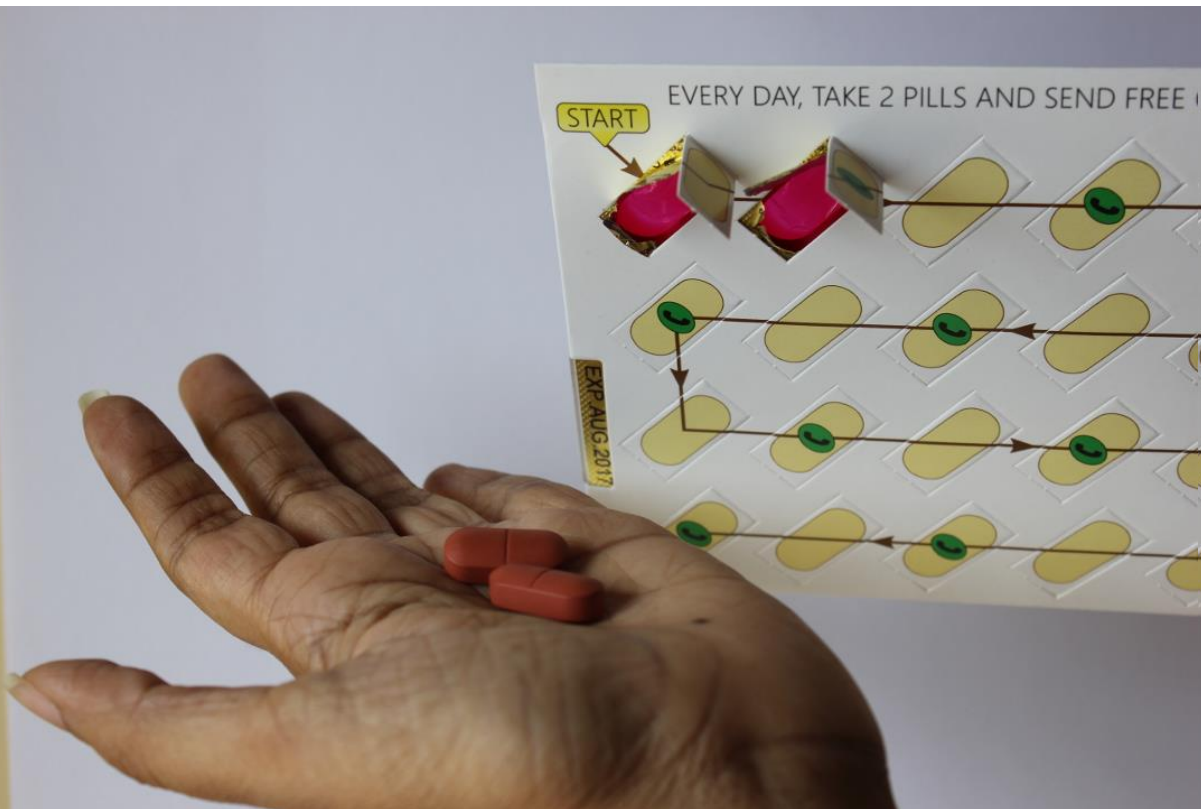
Tuberculosis in India

- **2.8 million cases** nationally
- Treatment: **6 months** of daily antibiotics
- Low adherence leads to **reinfection** and **drug-resistant strains**



Adherence tracking via phone calls

99DOTS



ID #	Attention Required	Patient	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26
6204	MEDIUM																											
6214	MEDIUM																											
6218	MEDIUM																											
6231	MEDIUM																											
6232	MEDIUM																											
6235	MEDIUM																											
6260	MEDIUM																											

Dataset

- Data shared by 99DOTS on 17,000 patients in Mumbai
- Anonymized call logs + basic demographic features

99DTS



What to do with data?

- Health workers have **limited resources**
- Text, call, make house visits to at-risk patients
- Goal: prevent missed doses

What to do with data?

- Status quo: reactive
 - This patient has missed 4 doses; better go check on them
- Ideal: proactive
 - This patient is showing warning signs; intervene preemptively

1	2	3	4	5	6	7
Green	Green	Green	Green	Green	Green	Green
Green	Green	Green	Green	Green	Red	Green
Red	Red	Red	Red	Red	Red	Red

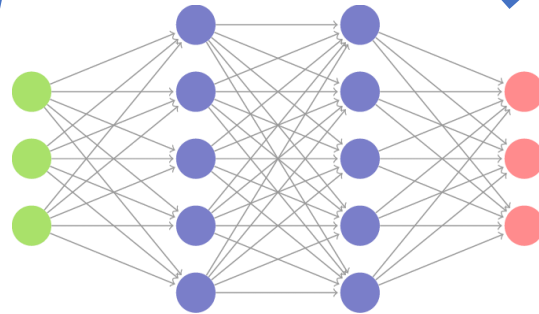
Data

8	9	10	11	12	13	14
Green	Green	Green	Green	Green	Green	Green
Green	Green	Green	Green	Green	Green	Red
Red	Red	Red	Red	Green	Light Green	Light Green

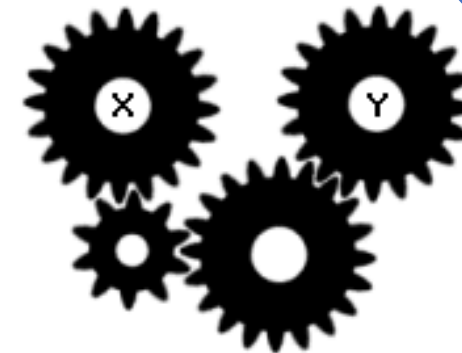
Predicted adherence



Intervention



Predictive model



Optimization algorithm

Resource allocation problem

- Let $\theta_t^i = \begin{cases} 1 & \text{if patient } i \text{ will miss a dose on day } t \\ 0 & \text{otherwise.} \end{cases}$
- Locations $1 \dots L$, patient i has location ℓ_i
- $x_t^j = \begin{cases} 1 & \text{if health worker goes to location } j \text{ on day } t \\ 0 & \text{otherwise.} \end{cases}$

Resource allocation problem

$$\max_x \sum_{t=1}^T \sum_{j=1}^L x_t^j \left(\sum_{i: \ell_i = j} \theta_i^t \right)$$

$$\sum_{j=1}^L x_t^j \leq 1, t = 1 \dots T$$

$$\sum_{t=1}^T x_t^j \leq 1, j = 1 \dots L$$

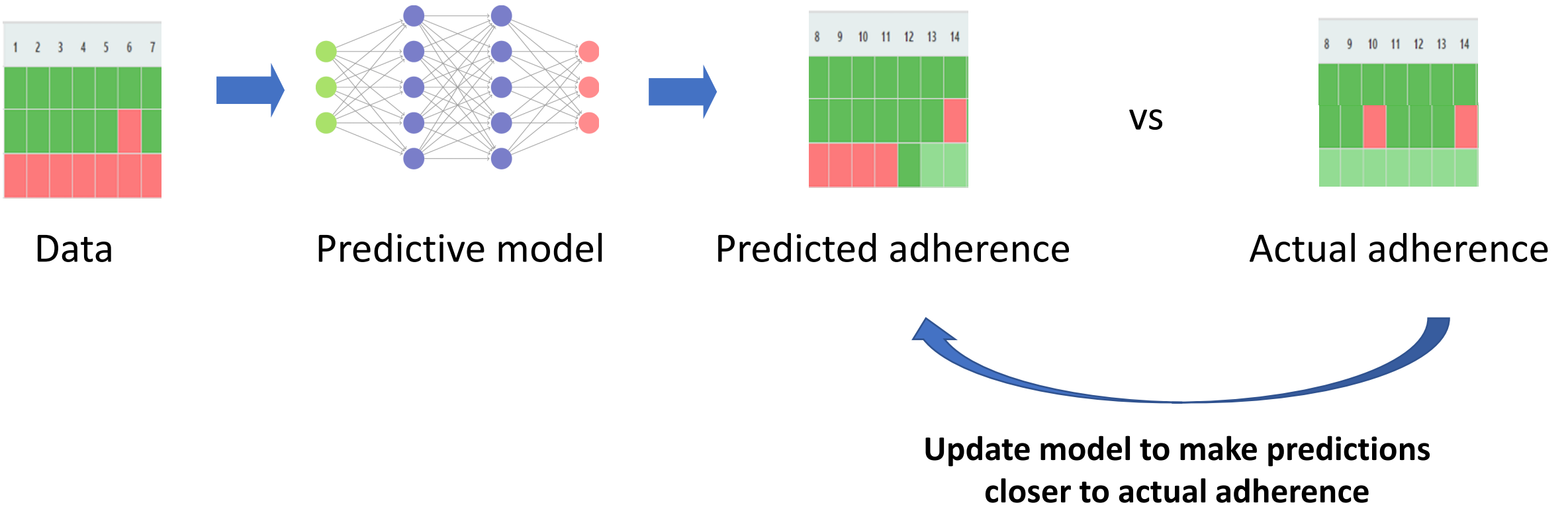
Resource allocation problem

$$\max_x \sum_{t=1}^T \sum_{j=1}^L x_t^j \left(\sum_{i:\ell_i=j} \theta_i^t \right) \longleftarrow \text{Future adherence isn't known!}$$

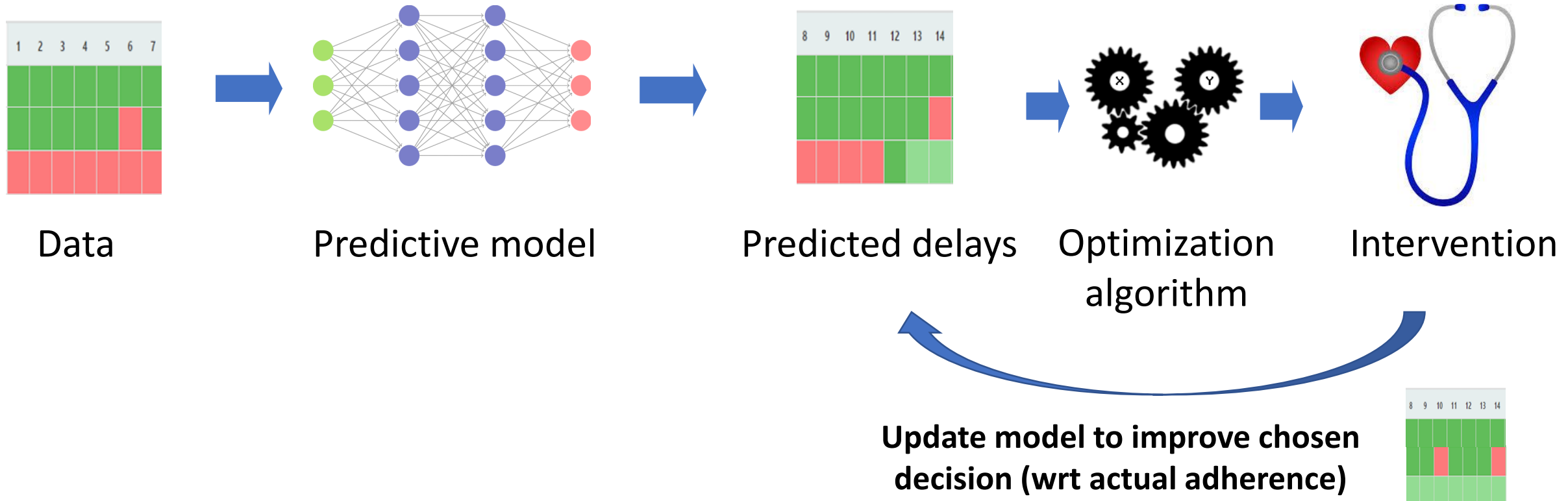
$$\sum_{j=1}^L x_t^j \leq 1, t = 1 \dots T$$

$$\sum_{t=1}^T x_t^j \leq 1, j = 1 \dots L$$

Two-stage training



Decision-focused learning

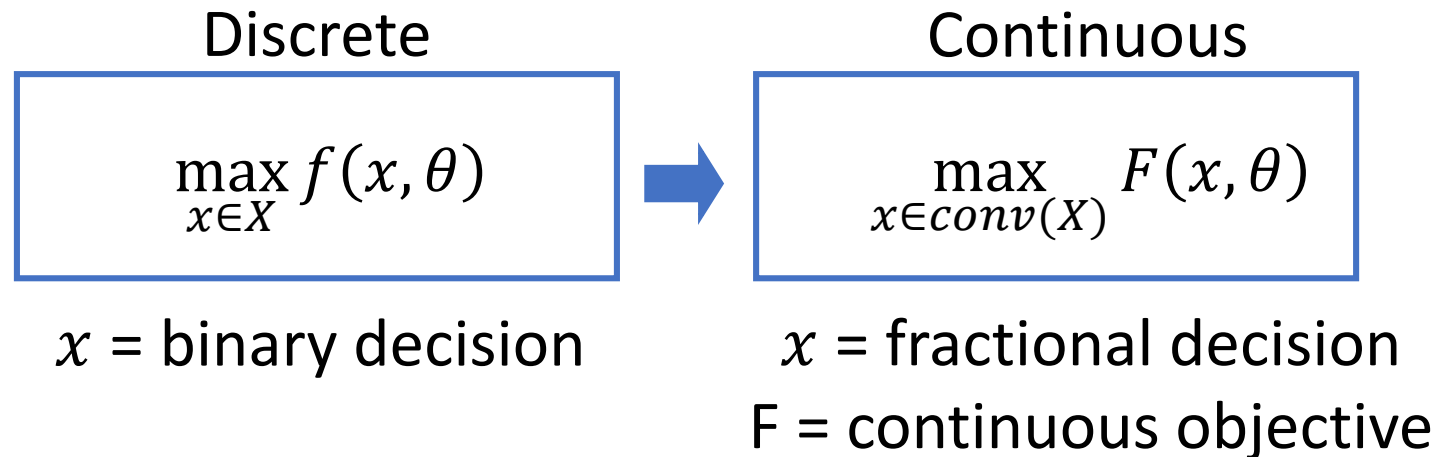


Approach

- Idea: differentiate optimal solution with respect to θ , train model via gradient descent
 - *Previous work: only convex problems [Rockafellar & Wets '09, Gould et al '16, Donti et al '17]*
- Challenge: the optimization problem is discrete!

Approach

- Idea: differentiate optimal solution with respect to θ , train model via gradient descent
 - *Previous work: only convex problems [Rockafellar & Wets '09, Gould et al '16, Donti et al '17]*
- Challenge: the optimization problem is discrete!
- Solution: relax to continuous problem, differentiate, round



Technical challenges

- What makes for a “good” continuous proxy F ?
- How to compute $\frac{dx^*}{d\theta}$?
 - Differentiate the output of the optimization algorithm wrt predictions

Differentiating through optimization

- Start by assuming that we have a good continuous relaxation
- How to backpropagate through the optimization step?
- For “nice” relaxations (convex programs), draw on known techniques

Differentiating through optimization

- How to compute $\frac{dx^*}{d\theta}$?
 - Differentiate the output of the optimization algorithm wrt predictions
- Idea: optimal solution must satisfy KKT conditions
- Differentiate through those equations via implicit function theorem

$$\begin{bmatrix} \nabla_x^2 f(x, \theta) & A^T \\ \text{diag}(\lambda)A & \text{diag}(Ax - b) \end{bmatrix} \begin{bmatrix} \frac{dx}{d\theta} \\ \frac{d\lambda}{d\theta} \end{bmatrix} = \begin{bmatrix} \frac{d\nabla_x f(x, \theta)}{d\theta} \\ 0 \end{bmatrix}$$

Linear programs

Standard form:

$$\begin{array}{ll} \max & \theta^T x \\ \text{s.t.} & Ax \leq b \end{array}$$

Includes bipartite matching, shortest path, mincut, etc.

Linear programs

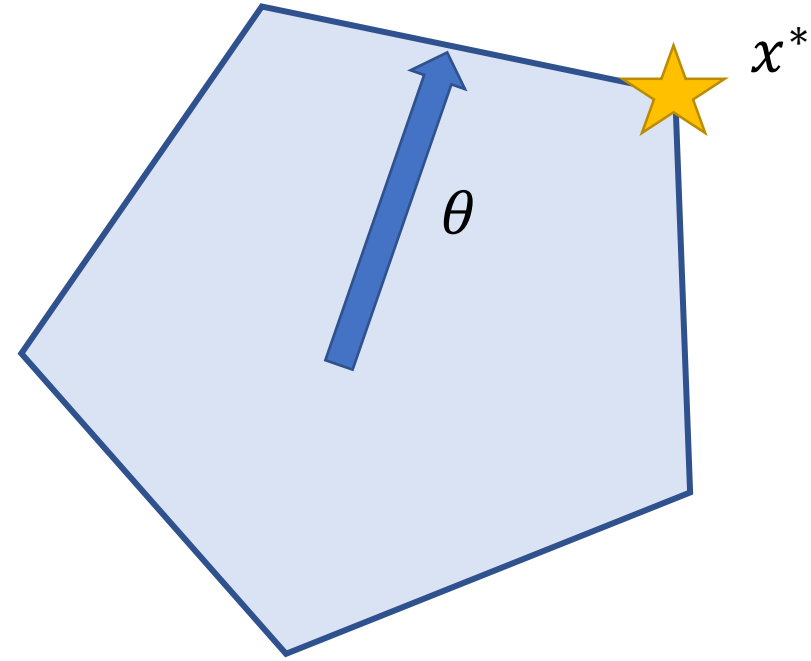
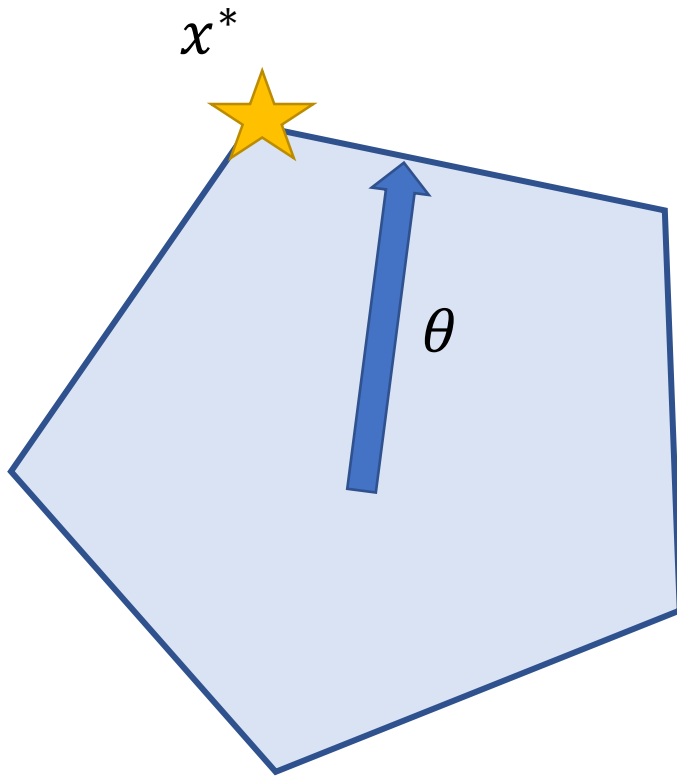
Standard form:

$$\begin{aligned} \max_x \quad & \theta^T x \\ \text{s.t.} \quad & Ax \leq b \end{aligned}$$

Includes bipartite matching, shortest path, mincut, etc.

Why can't we just take derivatives using known techniques?

Linear programs



Linear programs

- $\frac{dx^*}{d\theta}$ doesn't exist!
- The Hessian $\nabla_x^2 f(x, \theta) = 0$ is singular

Linear programs

- Solution: add a regularizer to smooth things out

$$\begin{aligned} \max_x \quad & \theta^T x - \gamma \|x\|_2^2 \\ & Ax \leq b \end{aligned}$$

- Now, Hessian is $\nabla_x^2 f(x, \theta) = -2\gamma I \prec 0$

Linear programs

Theorem: *Provided the LP is feasible, $x^*(\theta)$ is differentiable almost everywhere. Moreover, $\theta^T x^* \geq OPT - \gamma \cdot \text{diameter}(X)$.*

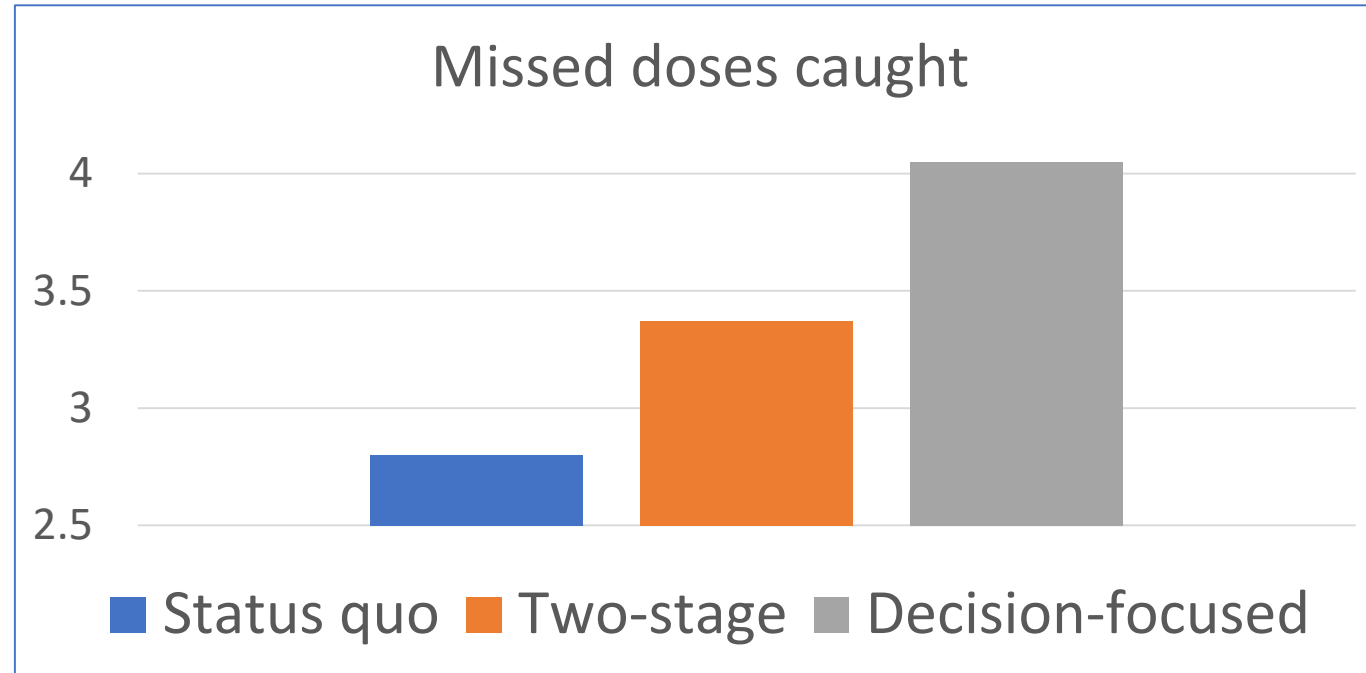
Application: tuberculosis

- Data shared by 99DOTS on 17,000 patients in Mumbai
- Train LSTM-based model to forecast adherence
- Three approaches
 - Status-quo (rule-based)
 - Standard cross-entropy loss (two-stage)
 - Decision-focused

99D  TS

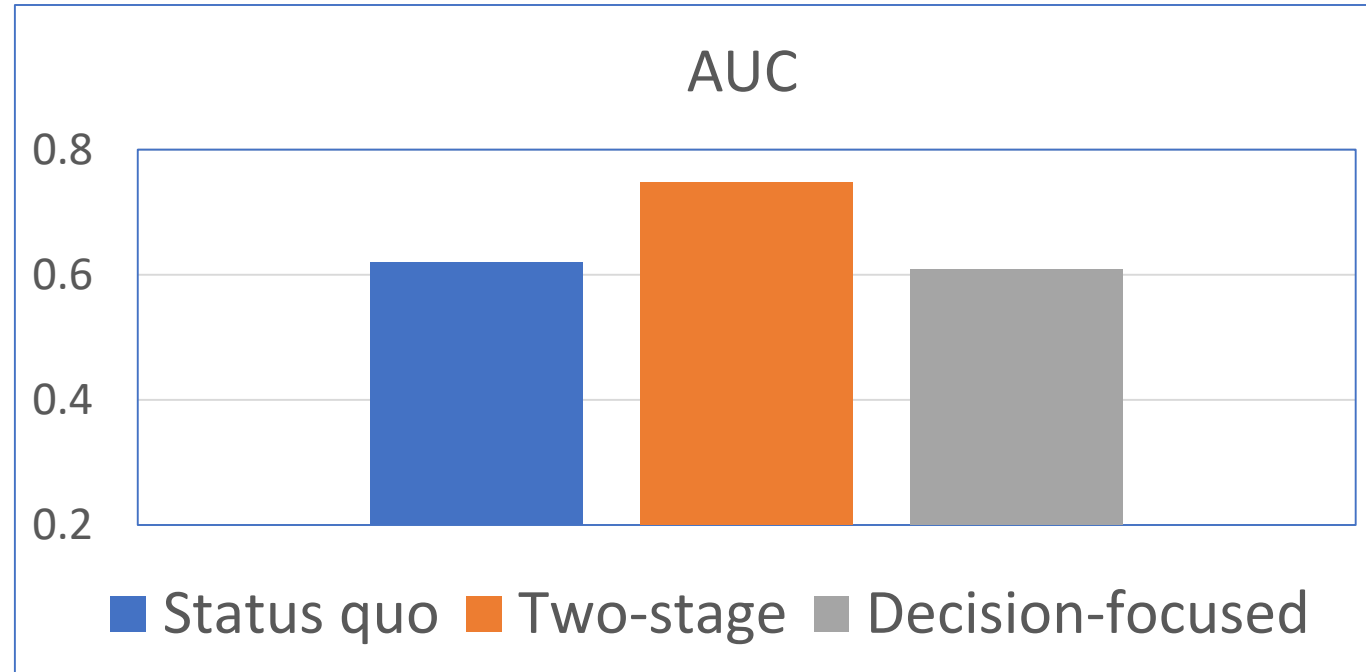


Results: tuberculosis treatment



Improvement in solution quality over baseline and two-stage

Results: tuberculosis treatment



Less “accurate” (but +15% successful interventions)!

Conclusion

- Data + optimization can better target TB interventions
- Need integrated approaches
 - Separate learning and optimization doesn't work for complex/noisy problems
- Opportunity to improve care for millions of people

99D  TS



MATCHING MARKETS & ALLOCATION UNDER UNCERTAINTY

In matching problems, prices do not do all – or any – of the work

Agents are **paired** with other (groups of) agents, transactions, or contracts

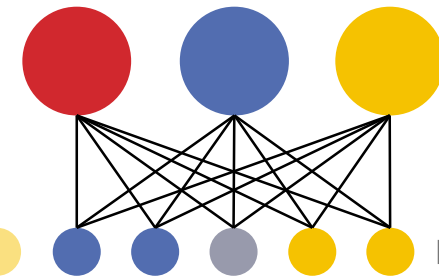
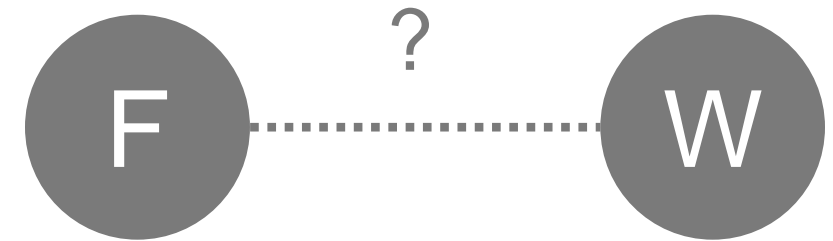
- Workers to firms
- Children to schools
- Residents to hospitals
- Patients to donors
- Advertisements to viewers
- Riders to rideshare drivers



UNCERTAINTY

- Does a matched edge truly exist?
- How valuable is a match?
- Will a better match arrive in the future?

upwork™
formerly oDesk



COMPETITION

Rival matching markets **compete** over the same agents

- How does this affect global social welfare?
- How to differentiate?



MATCH CADENCE

How quickly do new edges form?

How frequently does a market clear?

Is clearing centralized or decentralized?

Can agents reenter the market?



DECISION MAKING UNDER UNCERTAINTY!

Outline

- Intro & motivation
- Preliminaries & basic techniques
- Formulating objective functions: value judgement aggregation
- Decision making under uncertainty
- Offline allocation techniques & applications
- Online allocation techniques & applications
- Conclusion

Decision making under uncertainty

Decision making under uncertainty

- We have objective functions
- We have ways to optimize them
- Are we done?

Decision making under uncertainty

- No!
- Real world domains usually lack key information
- Need to make decisions that account for uncertainty about the objective or constraints

Decision making under uncertainty

- Objective function $f(x, \theta)$
 - x is the **decision variable**
 - θ is an **unknown parameter**
- Want to solve:

$$\max_{x \in X} f(x, \theta)$$

Decision making under uncertainty

- Robust optimization
 - Find decisions that work well regardless θ
- Active learning/information gathering
 - Acquire more information about θ
- Machine learning
 - Use other data sources to estimate θ

Decision making under uncertainty

- ➔ • Robust optimization
 - Find decisions that work well regardless θ
- Active learning/information gathering
 - Acquire more information about θ
- Machine learning
 - Use other data sources to estimate θ

Robust optimization

- We often don't know the “true” model
- One approach: robust optimization
- Given candidate objective functions f_θ induced by different parameters, solve

$$\max_{x \in X} \min_{\theta \in \Theta} f_\theta(x)$$

- Θ is the **uncertainty set**

Robust optimization

- In convex optimization: pretty “easy”
 - Maximum of convex functions remains convex
 - Lots of well-developed techniques
- Combinatorial problems: often become fundamentally harder
 - Approximable problems become inapproximable

Robust submodular optimization

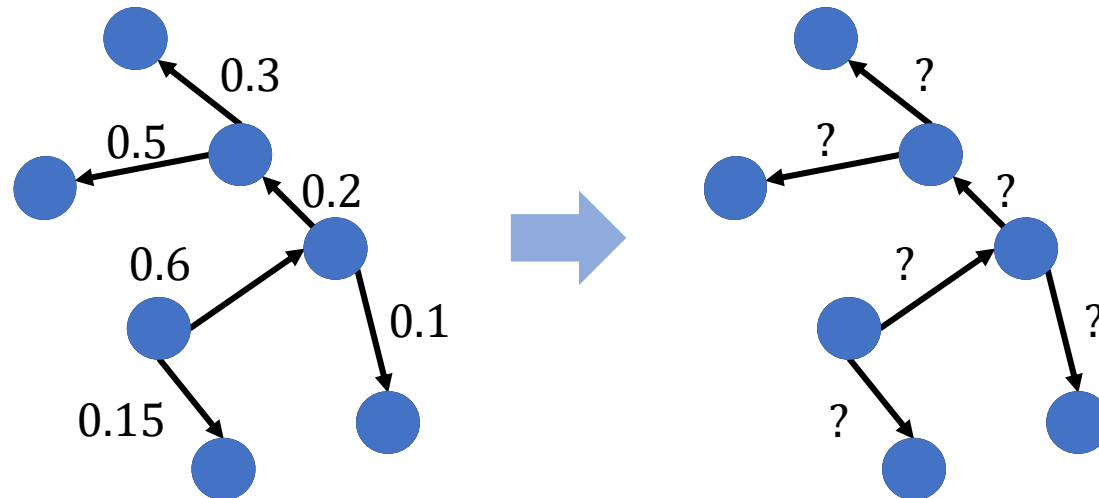
- Focus on this particular class of problems
- Recall: submodularity = set functions with diminishing returns
- Application: influence maximization

Influence maximization in the field

Lots of previous work on influence maximization...

[Kempe et al 2003, Chen et al 2011, Tang et al 2014...]

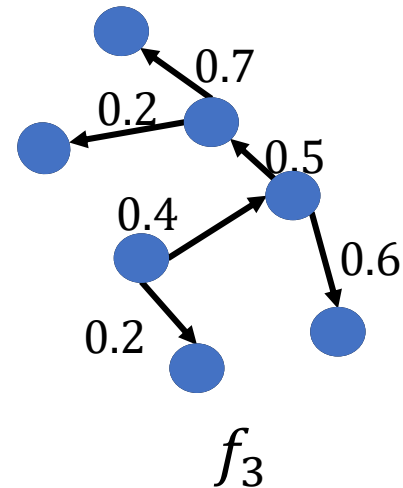
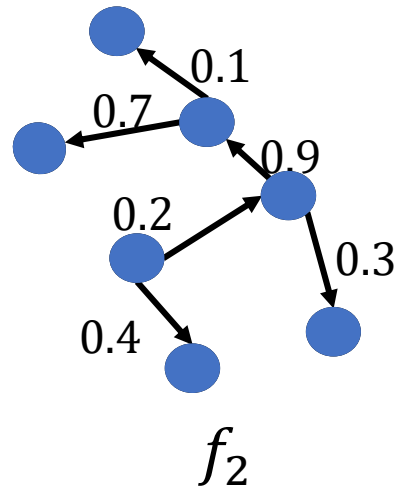
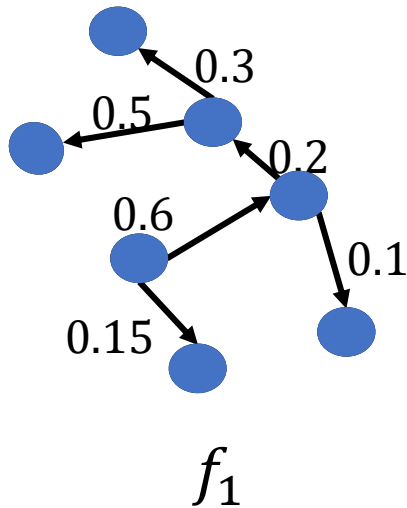
But assumes model of influence spread is known exactly!



Robust optimization

- Given candidate objective functions $f_1 \dots f_m$ induced by different models, solve

$$\max_{|S| \leq k} \min_{i=1 \dots m} f_i(S)$$



...

Hardness result

Theorem [Krause et al 2008]: finding an α -approximation for **any** $\alpha > 0$ for robust submodular maximization is NP-hard

[He and Kempe 2016]: This also holds specifically for influence maximization.

Solution concepts

- Bicriteria guarantee: Our algorithm can pick more than k nodes, but is only compared to OPT for k
- Mixed strategy: Instead of picking a single seed set, our algorithm picks a *distribution* over seed sets (randomized strategy, zero-sum game)

Solution methods

- Bicriteria guarantees: SATURATE algorithm
- Give increased budget αk ($\alpha > 1$)
- Binary search on the optimal objective value

Solution methods

- Bicriteria guarantees: SATURATE algorithm
- Give increased budget αk ($\alpha > 1$)
- Binary search on the optimal objective value
- For current guess c : greedily add nodes until $f_i(S) \geq c \ \forall i$

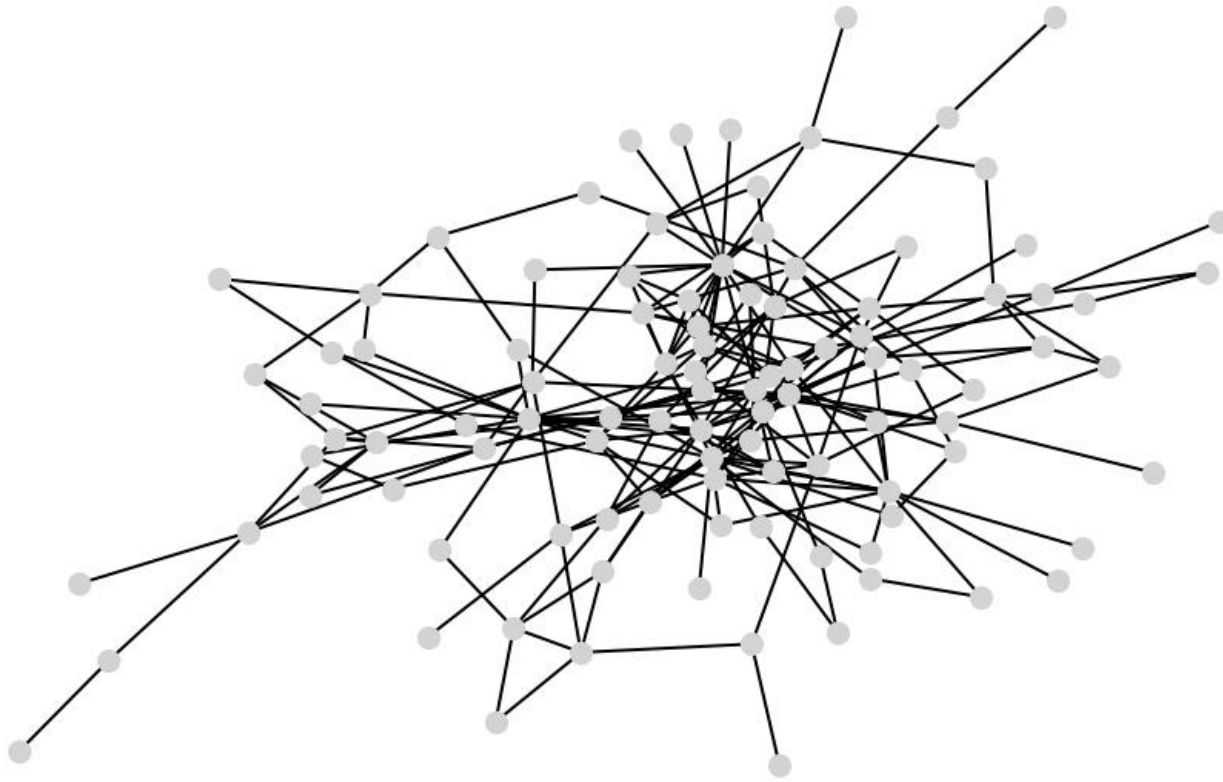
Solution methods

- Bicriteria guarantees: SATURATE algorithm
- Give increased budget αk ($\alpha > 1$)
- Binary search on the optimal objective value
- For current guess c : greedily add nodes until $f_i(S) \geq c \ \forall i$
- If achievable with αk nodes, try higher c . Else, try lower

Solution methods

- Bicriteria guarantees: SATURATE algorithm
- Give increased budget αk ($\alpha > 1$)
- Binary search on the optimal objective value
- For current guess c : greedily add nodes until $f_i(S) \geq c \ \forall i$
- If achievable with αk nodes, try higher c . Else, try lower

Theorem [Krause et al 2008]: Using a budget of $k \log m$, SATURATE matches the optimal objective value with budget k .

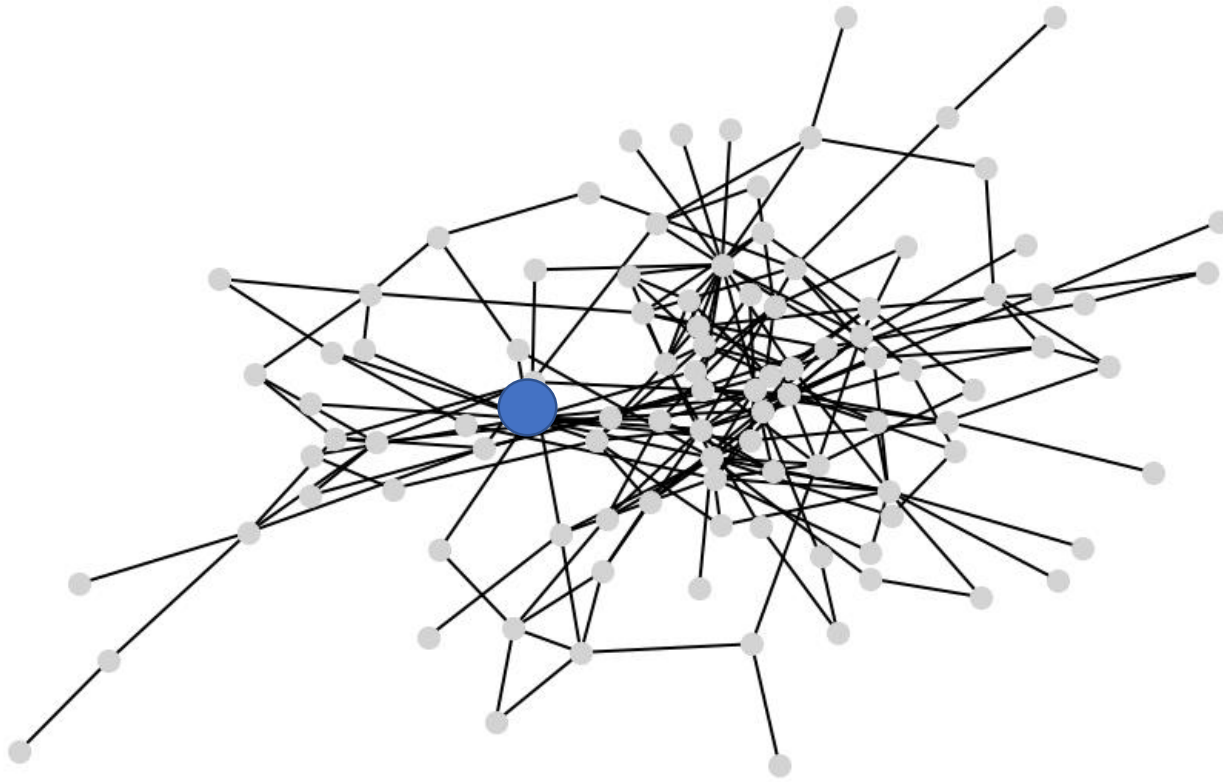


Target influence = 20

Budget = 5

Greedily maximize: $\sum_i \min(f_i(S), 20)$

f_1	f_2	f_3	f_4
0	0	0	0

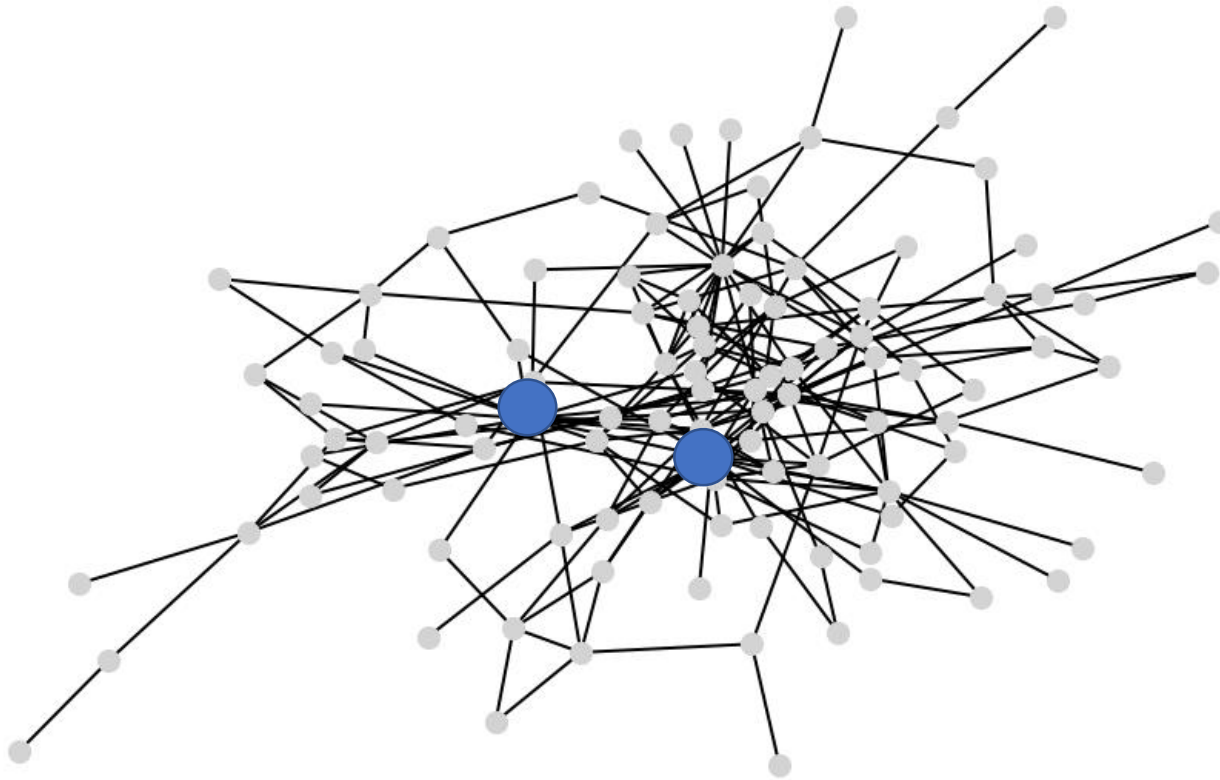


Target influence = 20

Budget = 5

Greedily maximize: $\sum_i \min(f_i(S), 20)$

f_1	f_2	f_3	f_4
5	7	2	10

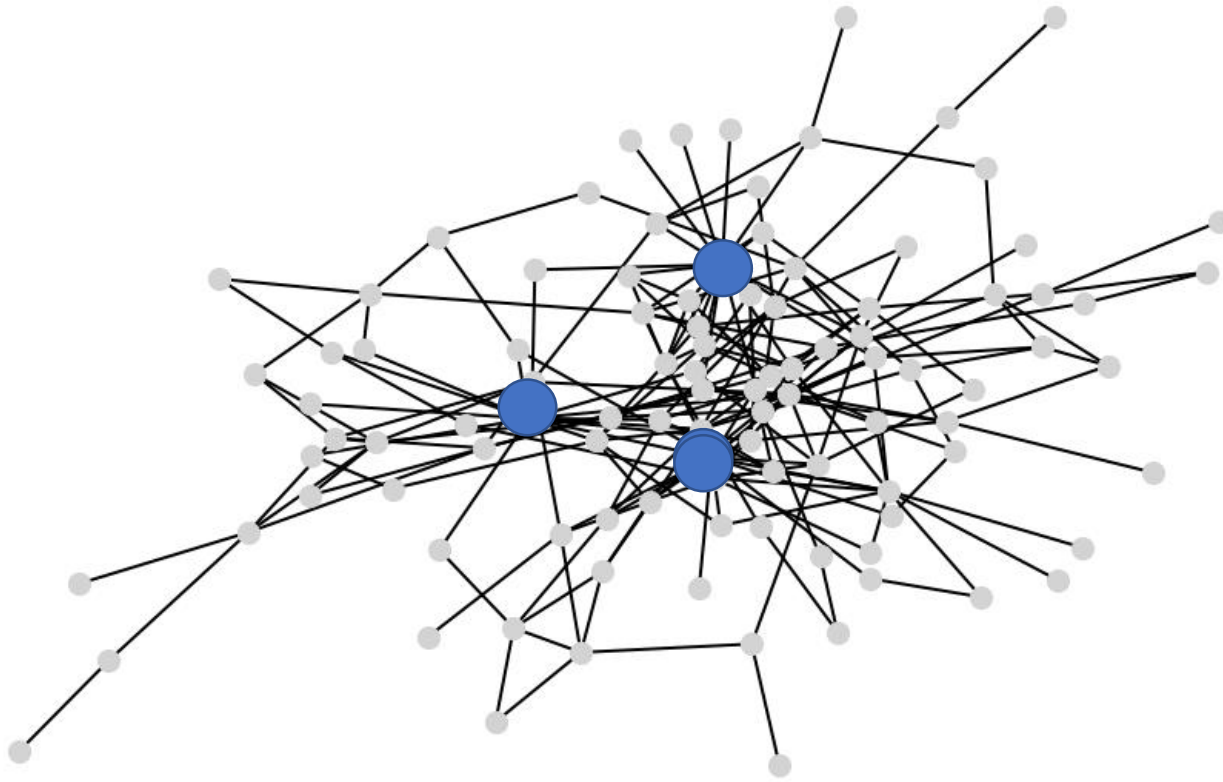


Target influence = 20

Budget = 5

Greedly maximize: $\sum_i \min(f_i(S), 20)$

f_1	f_2	f_3	f_4
8	9	4	15

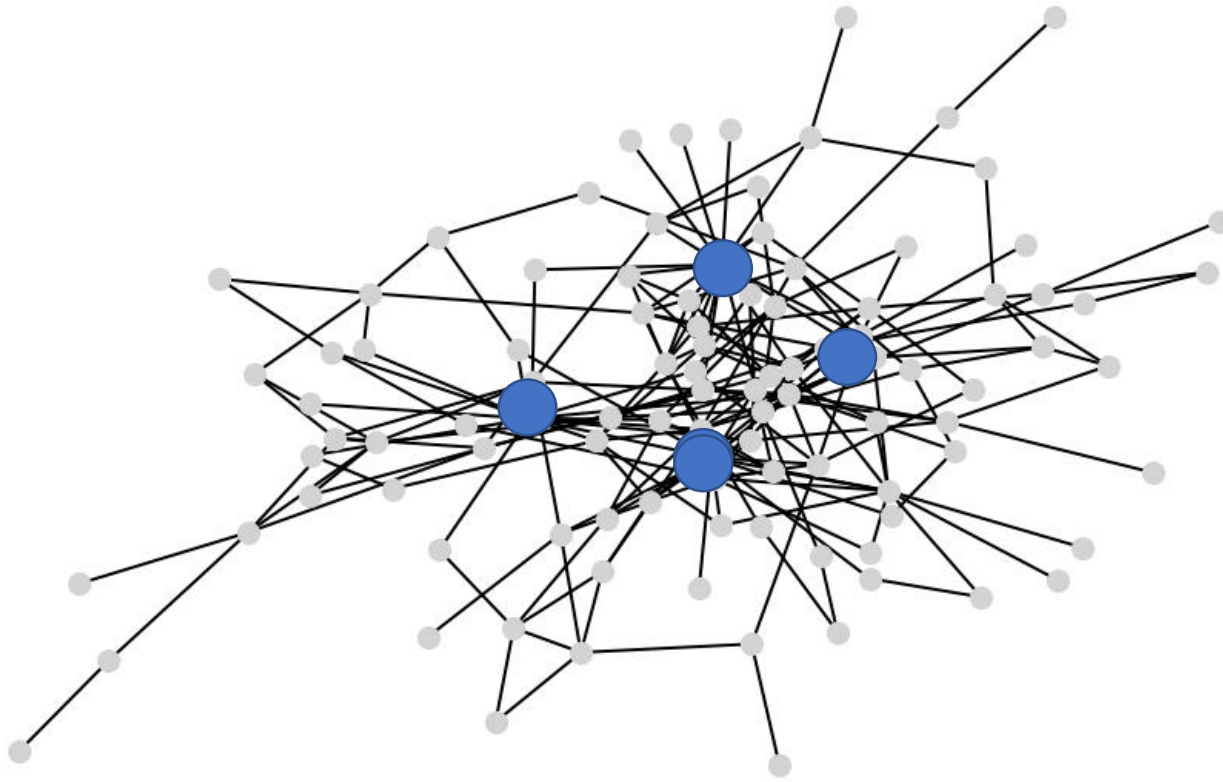


Target influence = 20

Budget = 5

Greedily maximize: $\sum_i \min(f_i(S), 20)$

f_1	f_2	f_3	f_4
11	10	7	20

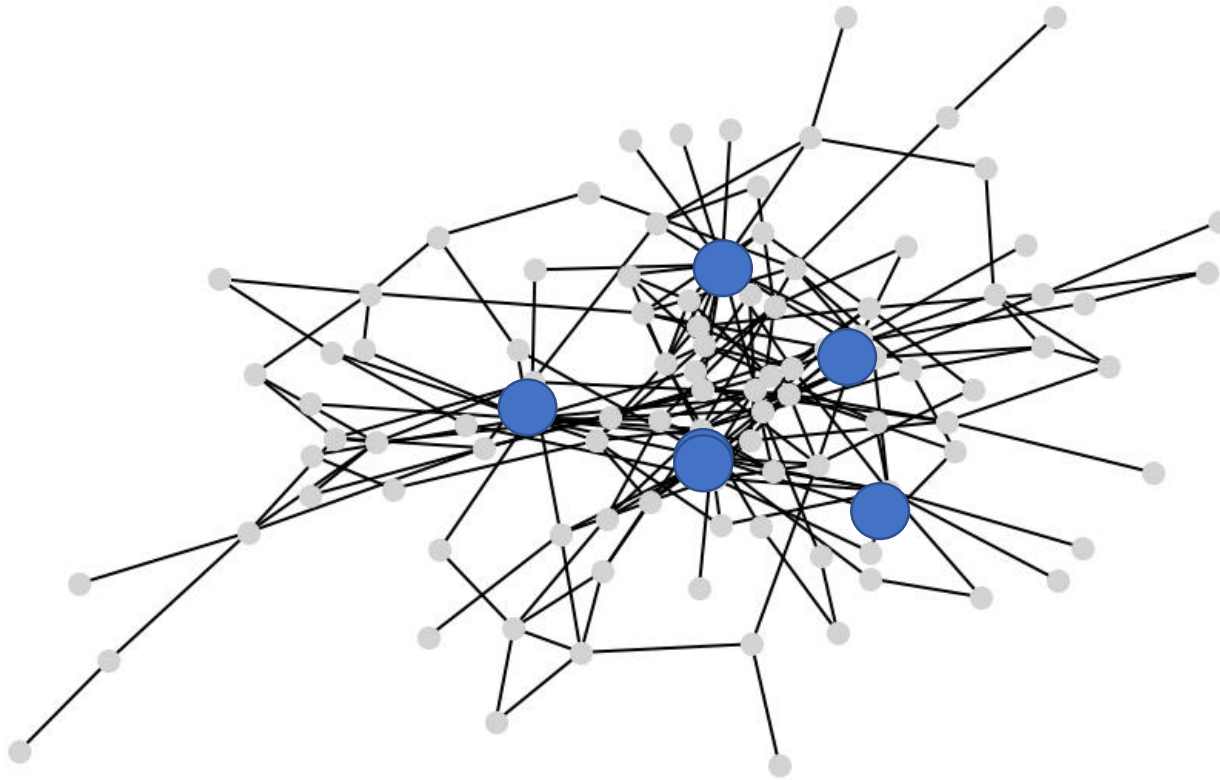


Target influence = 20

Budget = 5

Greedily maximize: $\sum_i \min(f_i(S), 20)$

f_1	f_2	f_3	f_4
12	12	9	20

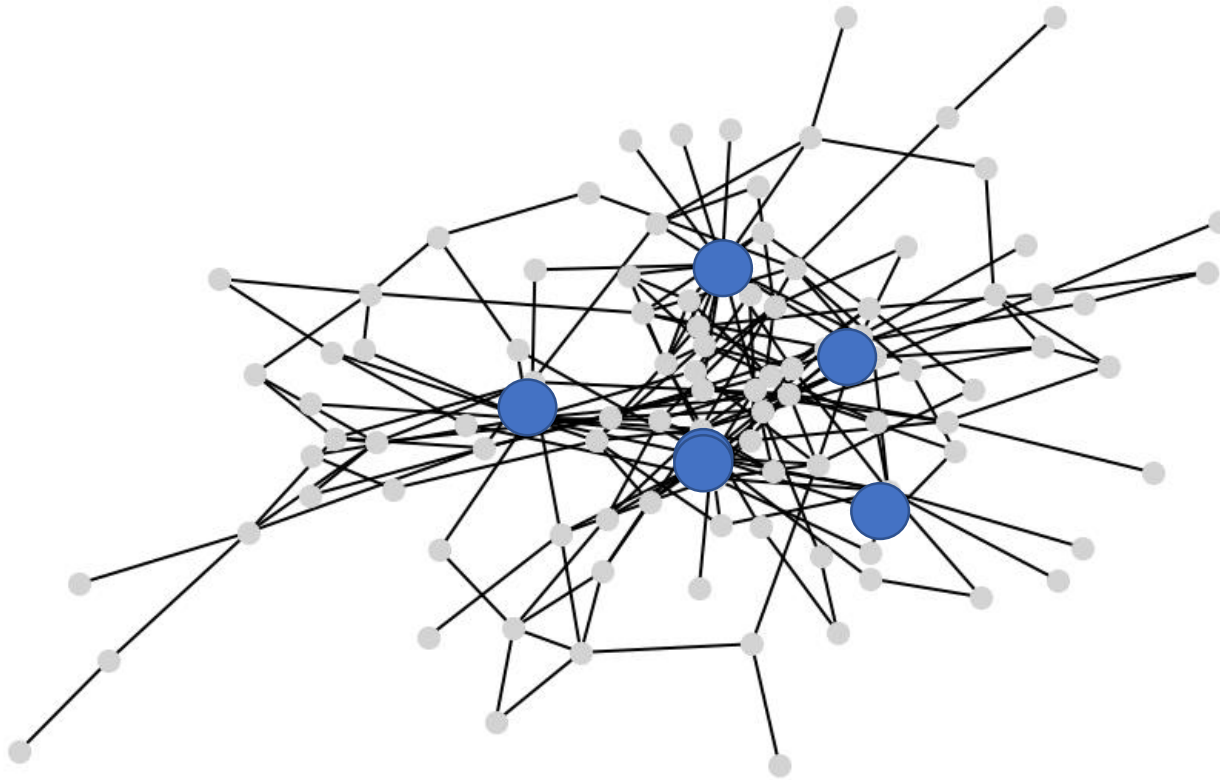


Target influence = 20

Budget = 5

Greedily maximize: $\sum_i \min(f_i(S), 20)$

f_1	f_2	f_3	f_4
15	17	10	20



Target influence = 20

Budget = 5

Greedly maximize: $\sum_i \min(f_i(S), 20)$

f_1	f_2	f_3	f_4
15	17	10	20



Failed with target = 20

Try again with target = 15

Example: model uncertainty

- Suppose we're not sure whether the ICM or the LTM is the right model
- SATURATE is a good way to optimize over both

Perturbation Interval ICM

- More complex example
 - *[He and Kempe 2016, Chen et al 2016]*
- Each edge (u, v) has a propagation probability $p_{u,v}$ for the ICM
- $p_{u,v}$ is not known exactly: belongs to interval $[a_{u,v}, b_{u,v}]$
- Every choice of $\{p_{u,v} \in [a_{u,v}, b_{u,v}] : (u, v) \in E\}$ defines an objective function (exponentially many)

Perturbation Interval ICM

- Since the worst-case value is always $p_{u,v} = a_{u,v}$, normalize by the optimal value
- $g_p = \frac{f_p(S)}{OPT(p)}$, solve $\max_{|S| \leq k} \min_p g_p(S)$

Perturbation Interval ICM

- Downside: can't apply SATURATE: exponentially many objectives
- He and Kempe: randomly sample a small set of them, use SATURATE

Scalable algorithms

- Need more powerful tools to deal with complicated uncertainty
- New approach: continuous optimization
 - Relax submodular set function to a continuous domain
 - Use (stochastic) gradient-based tools to optimize continuous function
 - Round back to a discrete set

Multilinear extension

- View set function f as defined on the vertices of the hypercube, $\{0, 1\}^n$
- Want to extend this to the entire hypercube $[0, 1]^n$

Multilinear extension

- View set function f as defined on the vertices of the hypercube, $\{0, 1\}^n$
- Want to extend this to the entire hypercube $[0, 1]^n$
- Canonical extension: *multilinear* extension [Calinescu et al 2011]
- Fractional variable $x \in [0, 1]^n$
- View x as marginals of a product distribution, $F(x) = \mathbb{E}_{S \sim x}[f(S)]$
- Equivalently, $F(x) = \sum_{S \subseteq X} f(S) \prod_{i \in S} x_i \prod_{i \notin S} 1 - x_i$

Multilinear extension

- F is not concave, no efficient global maximization
- But, it is *up-concave*
 - $F(x + \delta u)$, $u \geq 0$ is concave in $\delta > 0$

Multilinear extension

- F is not concave, no efficient global maximization
- But, it is *up-concave*
 - $F(x + \delta u)$, $u \geq 0$ is concave in $\delta > 0$
- Consequences:

Multilinear extension

- F is not concave, no efficient global maximization
- But, it is *up-concave*
 - $F(x + \delta u)$, $u \geq 0$ is concave in $\delta > 0$
- Consequences:
 - Any local optimum is a $\frac{1}{2}$ -approximation to the global optimum
 - [Chekuri et al 2014, Hassani et al 2017]

Multilinear extension

- F is not concave, no efficient global maximization
- But, it is *up-concave*
 - $F(x + \delta u)$, $u \geq 0$ is concave in $\delta > 0$
- Consequences:
 - Any local optimum is a $\frac{1}{2}$ -approximation to the global optimum
 - [Chekuri et al 2014, Hassani et al 2017]
 - Frank-Wolfe gradient-based algorithm gets a $\left(1 - \frac{1}{e}\right)$ -approximation
 - [Calinescu et al 2011, Bian et al 2017]

Multilinear extension

- Need a way to convert a fractional x back to a feasible set S
- There exist rounding algorithms which return a random set S with
$$E[f(S)] \geq F(x) \text{ (lossless rounding)}$$
- Works for general matroid constraints
 - Swap rounding, pipage rounding
 - See e.g. Calinescu et al 2009; Checkuri, Vondrak, Zenklusen 2009

Advantages of continuous approach

- More flexible/powerful
 - Robust or risk-averse objectives

Advantages of continuous approach

- More flexible/powerful
 - Robust or risk-averse objectives
- Often faster
 - Objectives often require random sampling (e.g. ICM)
 - Greedy needs many samples
 - Stochastic gradient methods can use 1, or small minibatch

Advantages of continuous approach

- More flexible/powerful
 - Robust or risk-averse objectives
- Often faster
 - Objectives often require random sampling (e.g. ICM)
 - Greedy needs many samples
 - Stochastic gradient methods can use 1, or small minibatch
- Better approximation guarantee for complex constraints
 - For general matroid, continuous gets $\left(1 - \frac{1}{e}\right)$ -approximation vs $\frac{1}{2}$ via greedy

Robust optimization

- Find a good *mixed strategy* against adversarial objective
 - [Krause et al 2010, Chen et al 2007, W 2018, Staib W Jegelka 2018]
 - Instead of relaxing budget, guarantee holds only in expectation
- Switch from discrete objectives $f_1 \dots f_m$ to their multilinear extensions $F_1 \dots F_m$, interpreting the fractional x as probability distribution

Solution methods

- Apply gradient-based method to the function $G(x) = \min F_i(x)$
- Get a (super)gradient of G by solving the inner adversarial problem to find the minimizing F_i

Solution methods

- Apply gradient-based method to the function $G(x) = \min F_i(x)$
- Get a (super)gradient of G by solving the inner adversarial problem to find the minimizing F_i

Theorem: This gives a $\left(1 - \frac{1}{e}\right)$ -approximation to the optimal mixed strategy

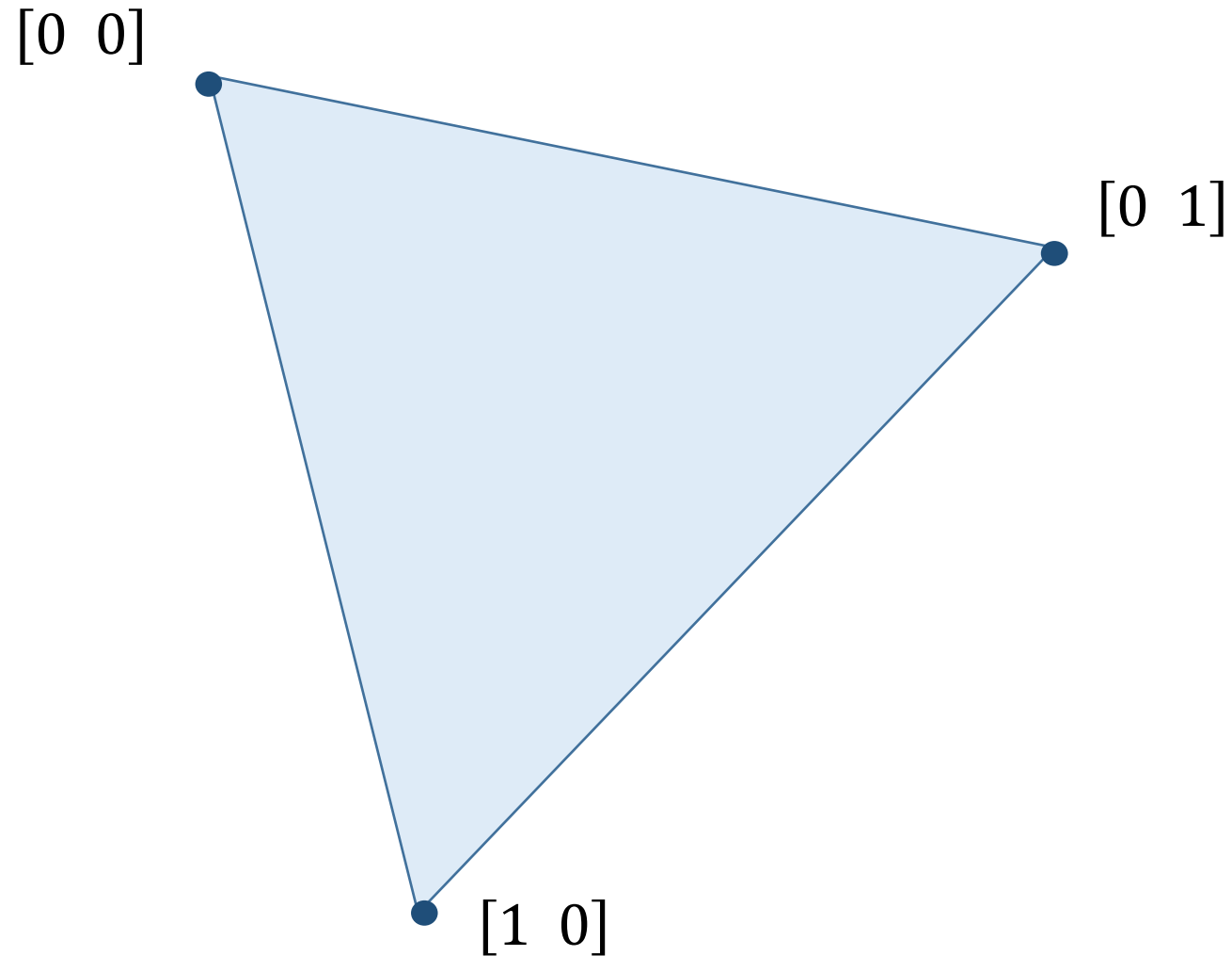
Continuous optimization

$[0 \ 0]$ •

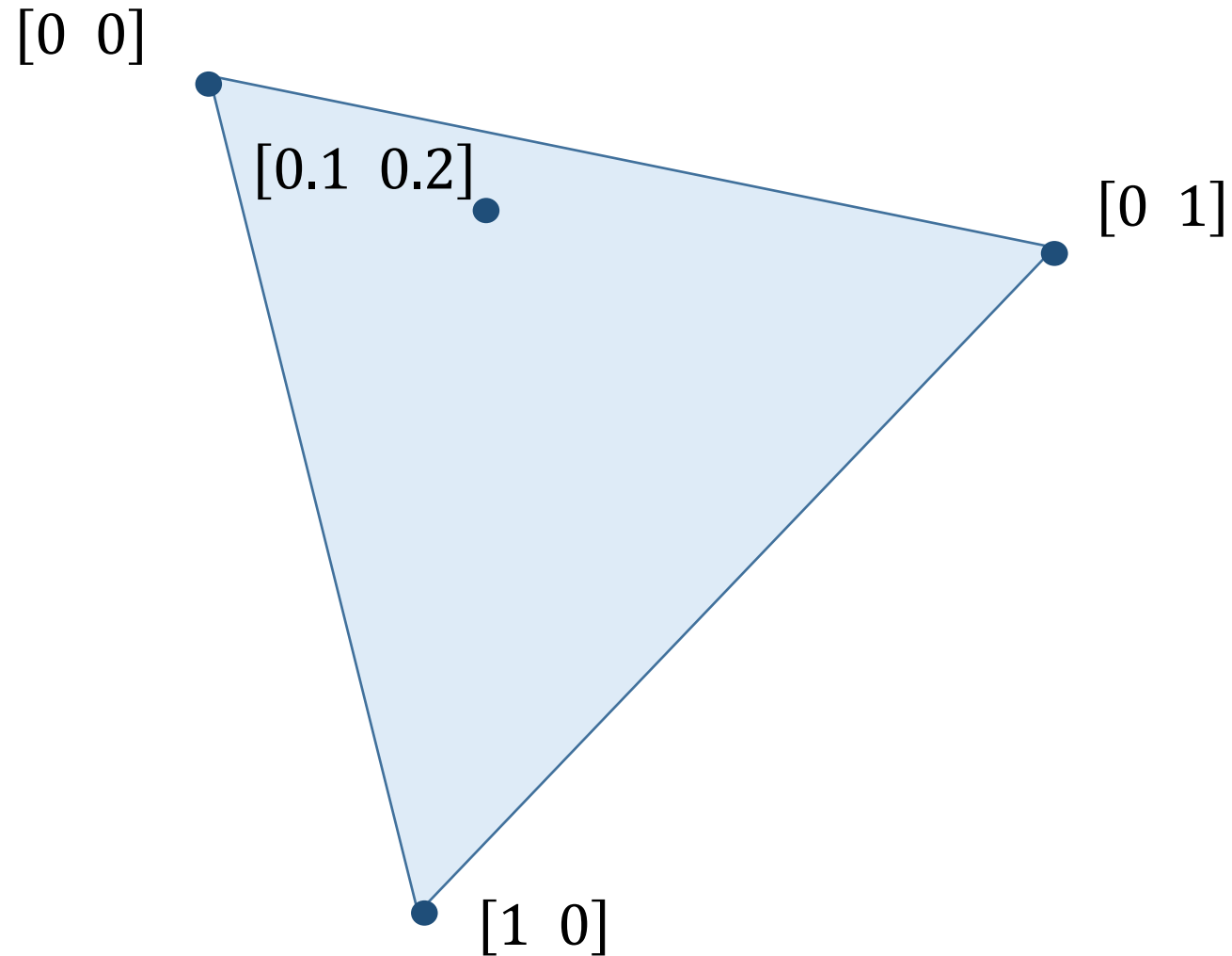
• $[0 \ 1]$

• $[1 \ 0]$

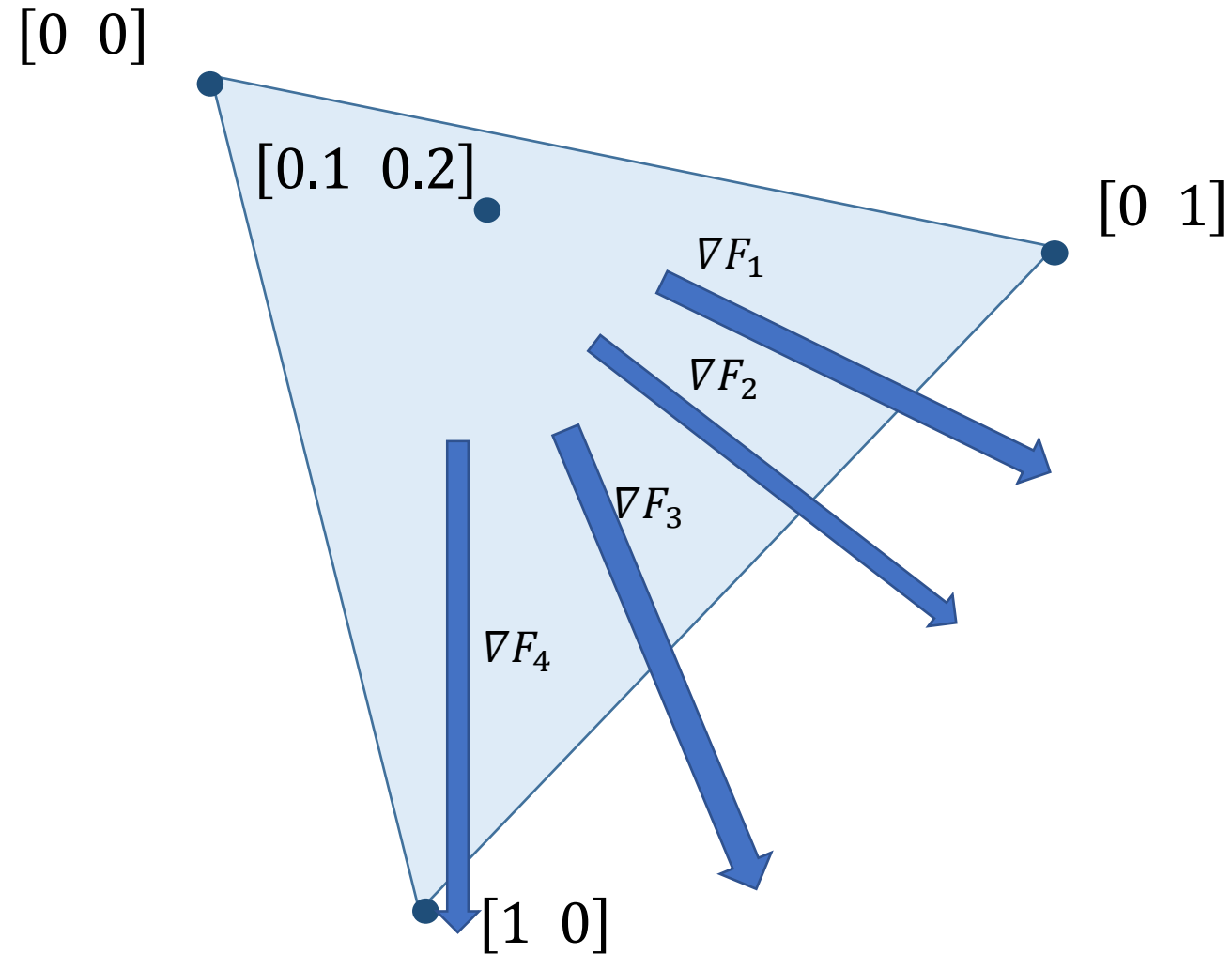
Continuous optimization



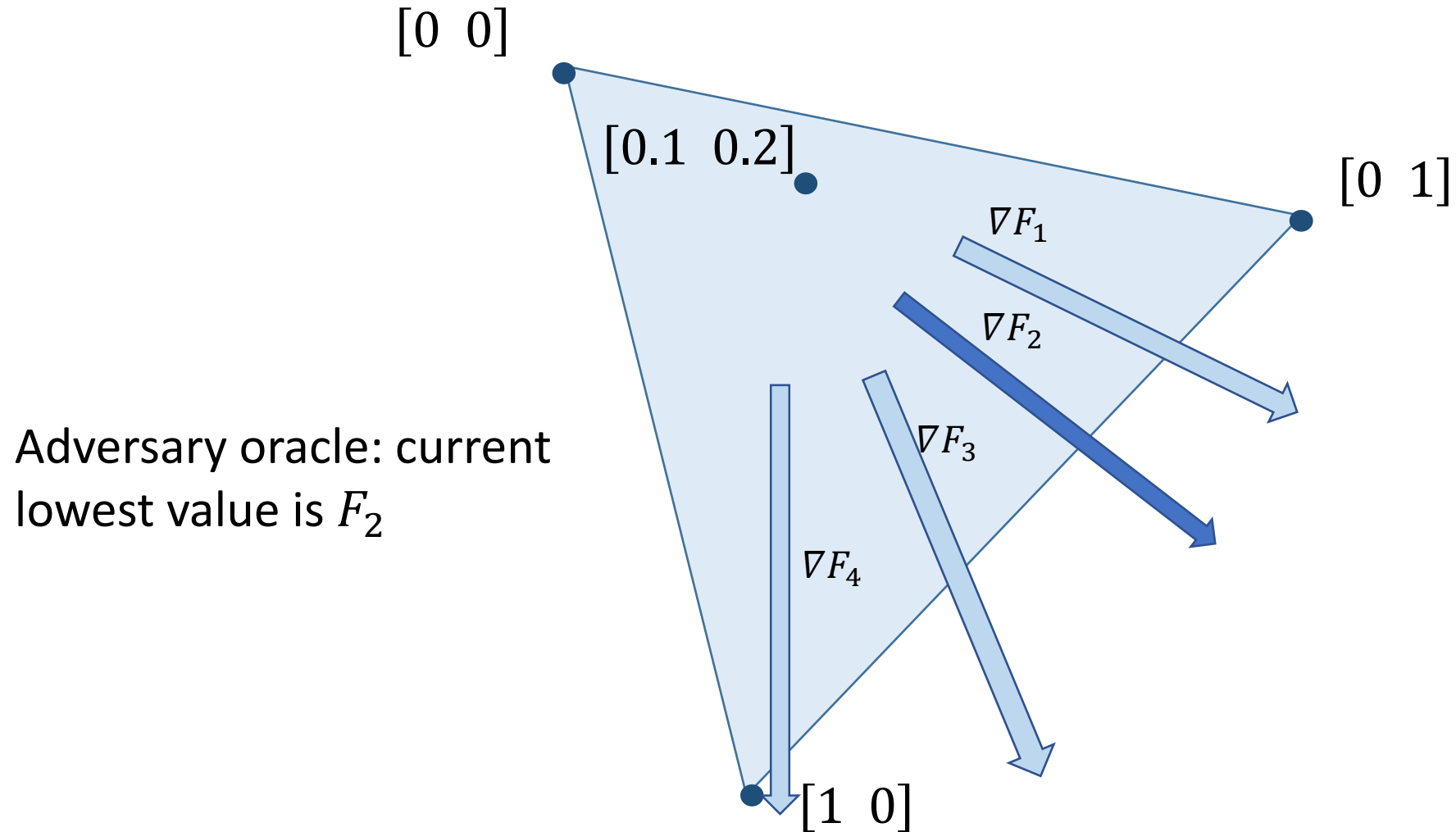
Continuous optimization



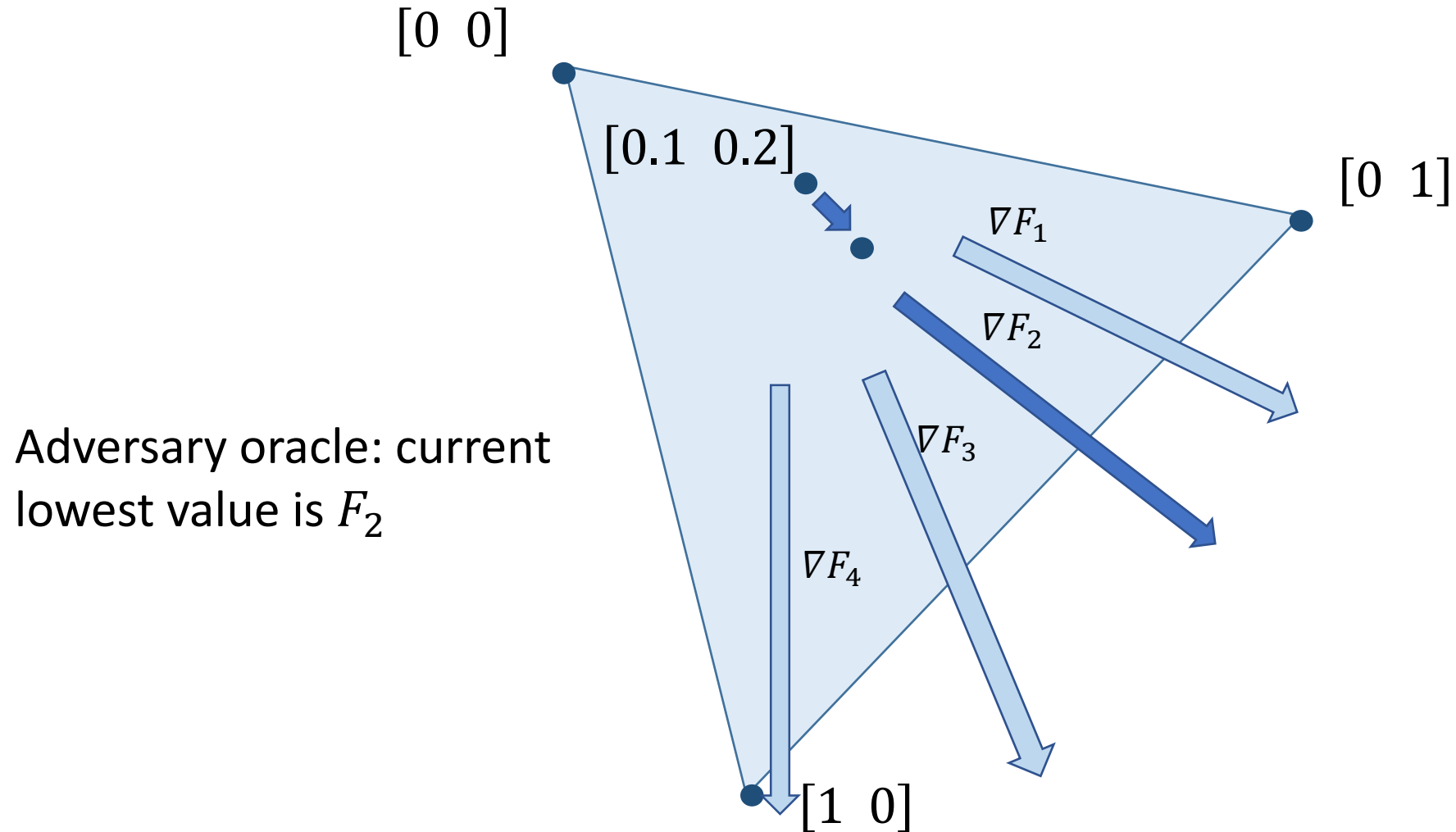
Continuous optimization



Continuous optimization



Continuous optimization



Pros and cons

- Discrete methods (SATURATE):
 - Pro: Good bicriteria performance ($\log m$ is usually overly pessimistic)
 - Con: slow, and only tractable when m (# objectives) is small
- Continuous methods:
 - Pro: fast, and works for **any** m so long as you can solve adversary's problem
 - Con: weaker guarantee for the actual action that's sampled
 - Workaround: can often translate a mixed strategy into a (somewhat worse) bicriteria guarantee; see [Chen et al 2017, Anari et al 2018]

PRE-MATCH EDGE TESTING

Idea: perform a small amount of costly testing before a match run to test for (non)existence of edges

E.g., more extensive medical testing, donor interviews, surgeon interviews, ...

Cast as a stochastic matching problem:

Given a graph $G(V, E)$, choose subset of edges S such that:

$$|M(S)| \geq (1-\epsilon) |M(E)|$$

Need: “sparse” S , where every vertex has $O(1)$ incident tested edges

GENERAL THEORETICAL RESULTS

Adaptive: select one edge per vertex per *round*, test, repeat

T
H
E
O
R
E
M

Stochastic matching:

$(1-\varepsilon)$ approximation with $O_\varepsilon(1)$ queries per vertex, in $O_\varepsilon(1)$ rounds

Stochastic k-set packing:

$(2/k - \varepsilon)$ approximation with $O_\varepsilon(1)$ queries per vertex, in $O_\varepsilon(1)$ rounds

Non-adaptive: select $O(1)$ edges per vertex, test all at once

T
H
E
O
R
E
M

Stochastic matching:

$(0.5-\varepsilon)$ approximation with $O_\varepsilon(1)$ queries per vertex, in 1 round

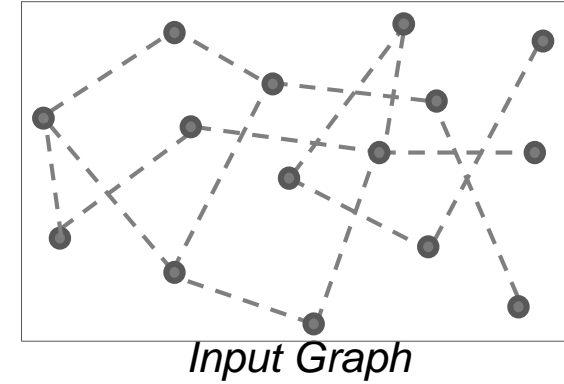
Stochastic k-set packing:

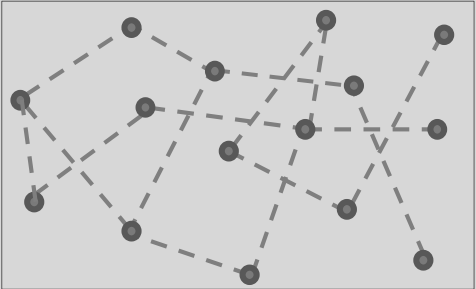
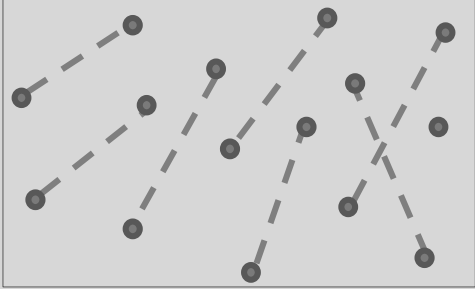
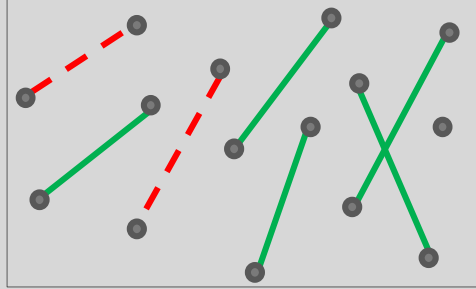
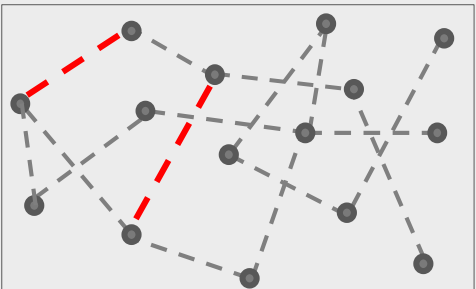
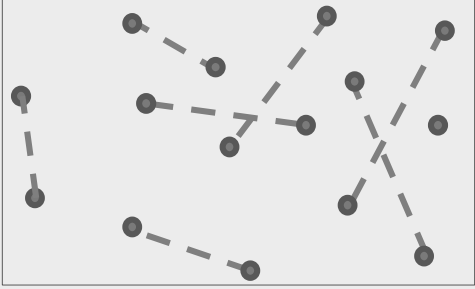
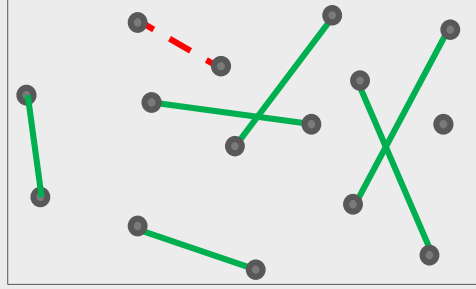
$(2/k - \varepsilon)^2$ approximation with $O_\varepsilon(1)$ queries per vertex, in 1 round

ADAPTIVE ALGORITHM

For R rounds, do:

1. Pick a max-cardinality matching M in graph G , minus already-queried edges that do not exist
2. Query all edges in M



r	Base graph	Matching picked	Result of queries
1:			
2:			

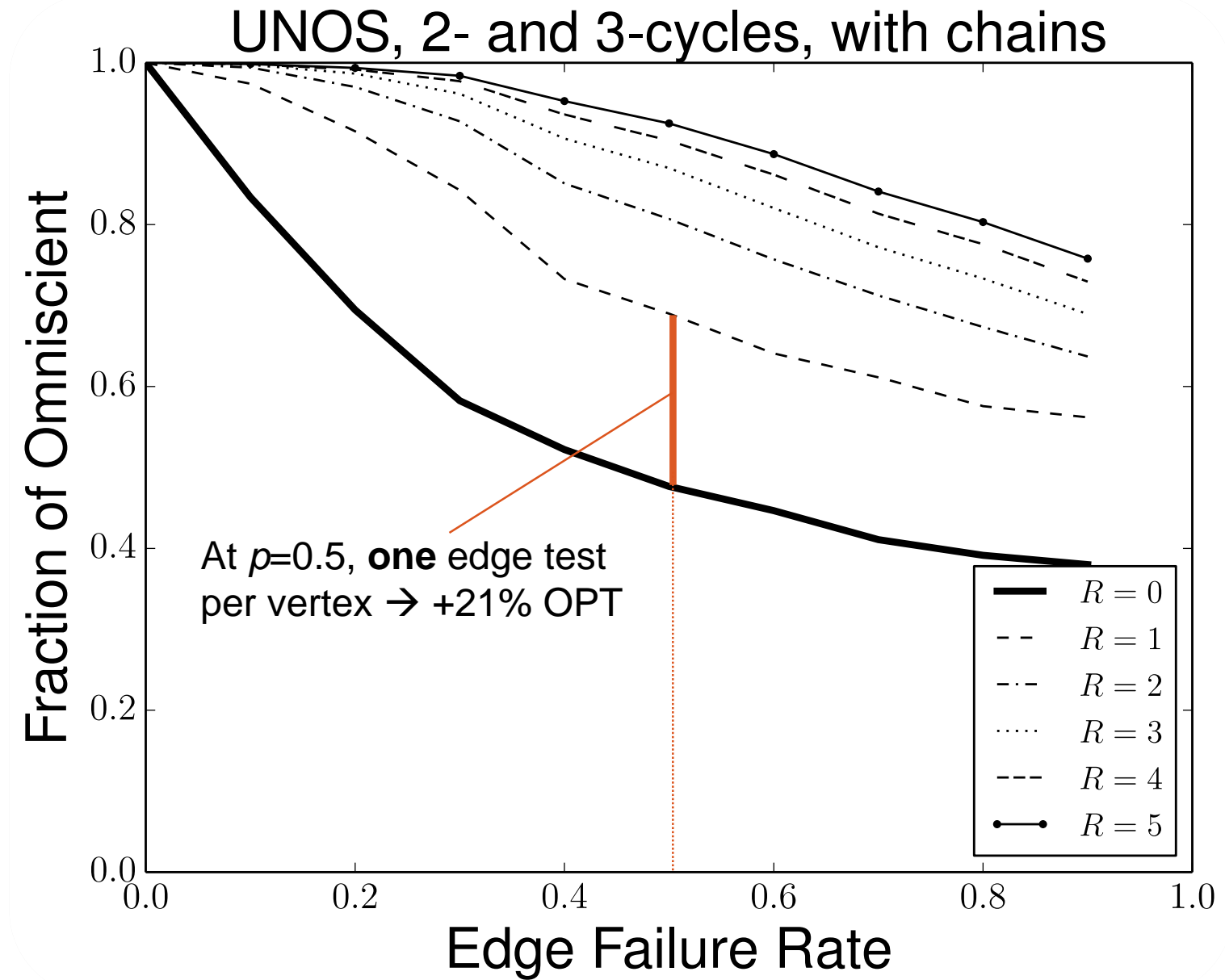
INTUITION FOR ADAPTIVE ALGORITHM

If at any round r , the best solution on edges queried so far is **small** relative to omniscient ...

- ... then current structure admits large number of unqueried, disjoint augmenting structures
- For $k=2$, aka normal matching, simply augmenting paths

Augmenting structures might not exist, but can query in parallel in a single round

- Structures are constant size \rightarrow exist with constant probability
- Structures are disjoint \rightarrow queries are independent
- \rightarrow Close a constant gap per round



Even 1 or 2 extra tests would result in a huge lift