ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

inf | Informatik
Computer Science

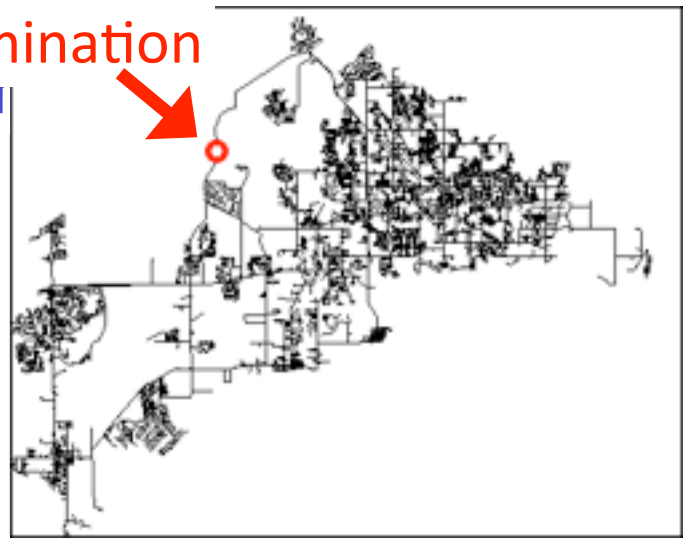
Tutorial on Submodular Optimization in Sensor and Social Networks PART I

Andreas Krause

Modern Aspects of Submodularity
GeorgiaTech March 2012

Monitoring water networks

Contamination of drinking water could affect millions of people



~$14K

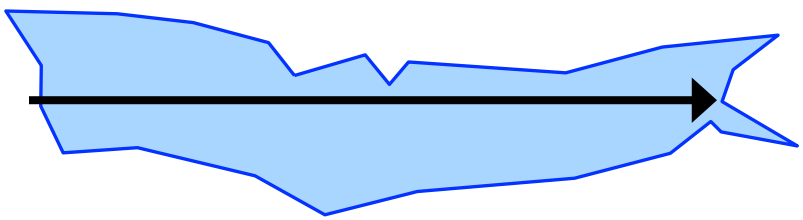
Place sensors to detect contaminations

Where should we place sensors to quickly detect contamination?

Robotic monitoring of rivers and lakes

Need to monitor large spatial phenomena

Temperature, nutrient distribution, fluorescence, …

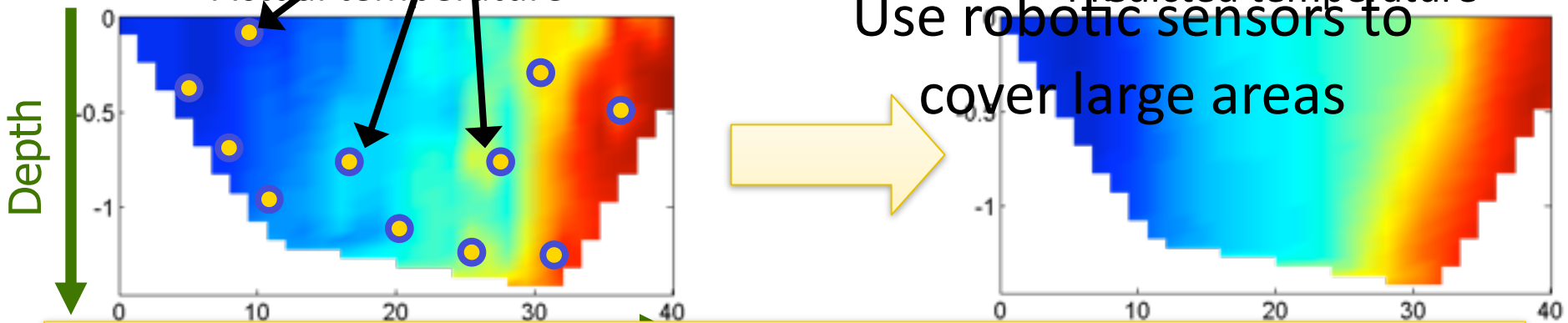


NIMS
Kaiser et.al. (UCLA)

Can only make a limited number of measurements!

Actual temperature

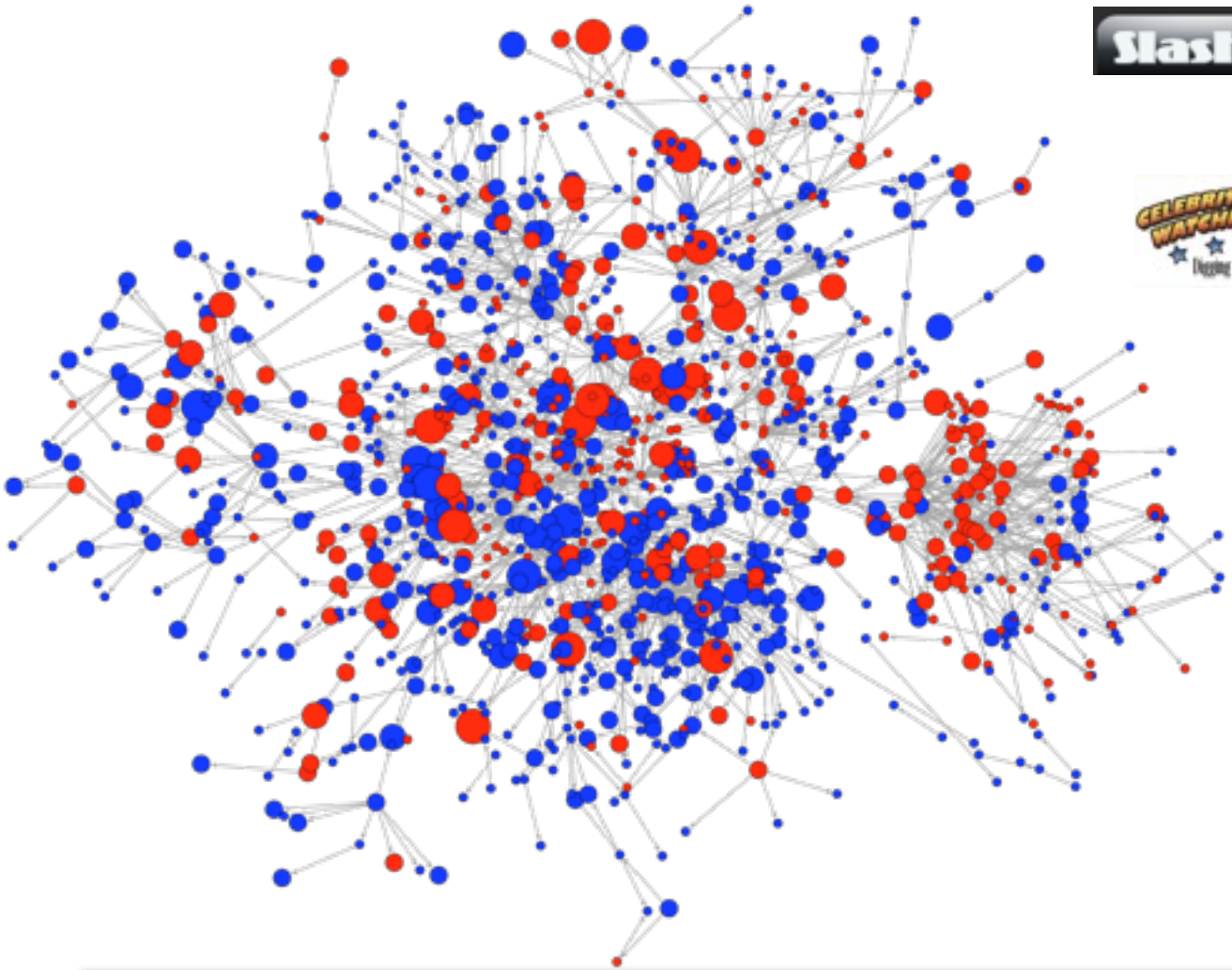
Predicted temperature



Use robotic sensors to cover large areas

Where should we sense to get most accurate predictions?

Inferring influential nodes



Slashdot

engadget

CELEBRITY DOG WATCHER.COM
Digging up celebrity dog news.

sisu

boingboing

Machine Learning (Theory)

the "blog" of "unnecessary" quotation marks

What are the most influential nodes (blogs, papers, …)?

Combinatorial optimization in sensor

- Many applications in sensor and social networks require solving discrete optimization problems

- Many of those can be formulated this way:
  Given finite set V (nodes in the graph, features, locations, …) wish to select subset A (subject to some constraints) maximizing submodular utility F(A)

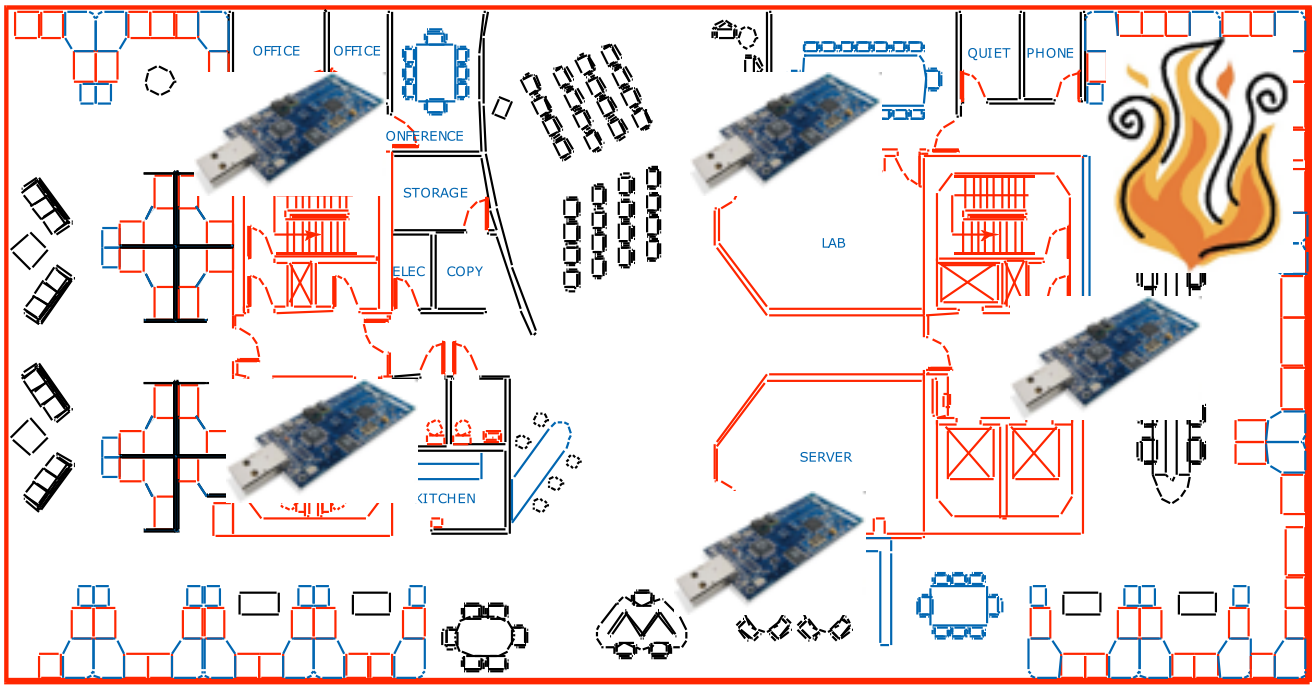
$$\max_{A \subseteq V} F(A)$$

Tutorial Overview

- Part I: Submodular maximization problems in sensor and social networks
  - Objective functions
  - Optimization problems and algorithms
  - Results on real data
- Part II: Learning to optimize submodular functions
  - No-regret online learning
  - Adaptive optimization

- *Only focus on constrained maximization problems*

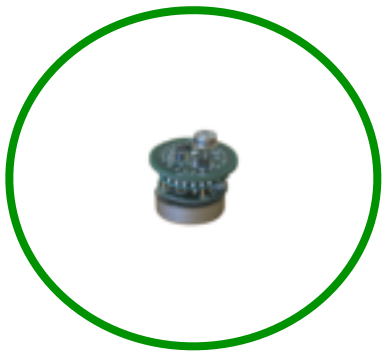
Sensor placement



Want to place sensors to monitor temperature

Example: Set cover

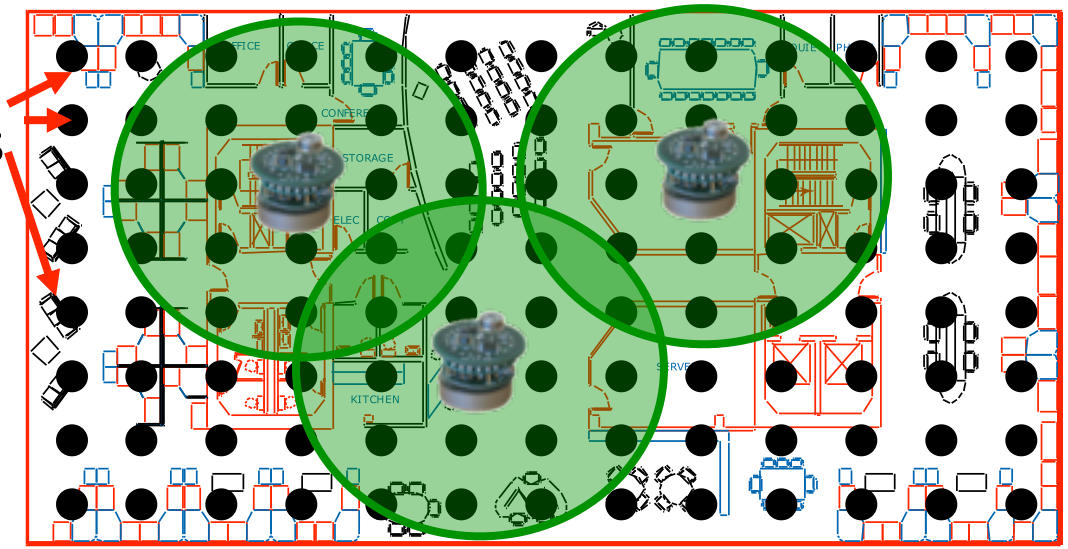
Place sensors in building



Node predicts values of positions with some radius

Possible locations V

Want to cover floorplan



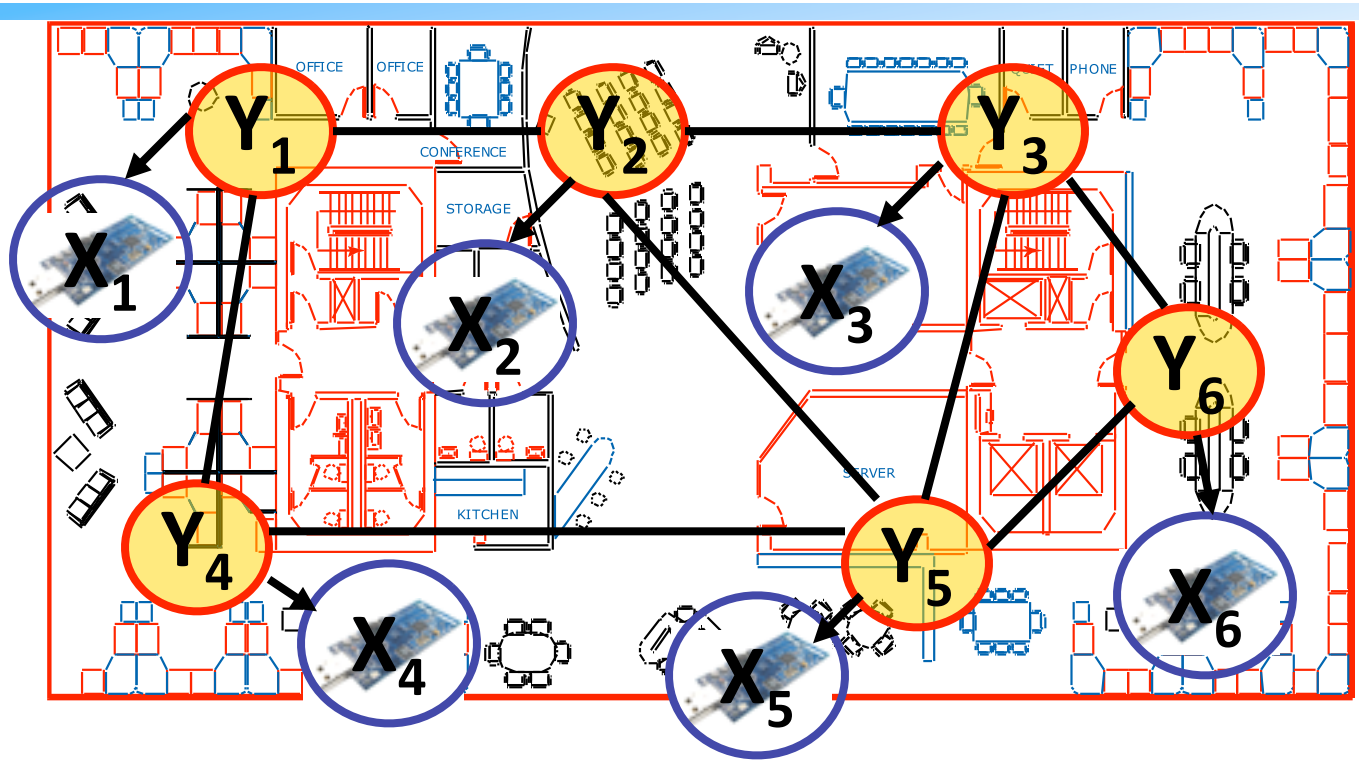
For $A \subseteq V$: F(A) = "area covered by sensors placed at A"

Formally:

W finite set, collection of n subsets $S_i \subseteq W$

For $A \subseteq V$ define $F(A) = |\bigcup_{i \in A} S_i|$

A better model for sensing



Y_s: temperature at location s

X_s: sensor value at location s

$$X_s = Y_s + \text{noise}$$

Joint probability distribution

$$P(X_1,\ldots,X_n,Y_1,\ldots,Y_n) = P(Y_1,\ldots,Y_n)\, P(X_1,\ldots,X_n \mid Y_1,\ldots,Y_n)$$

Prior **Likelihood**

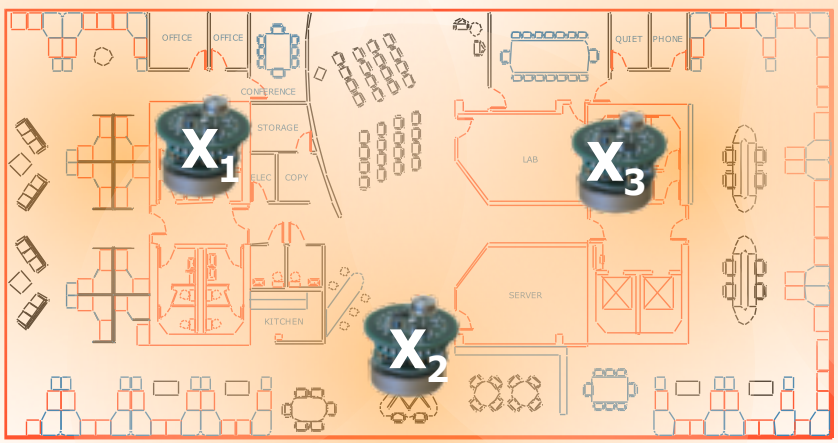
Information gain

Utility of having sensors at subset A of all locations

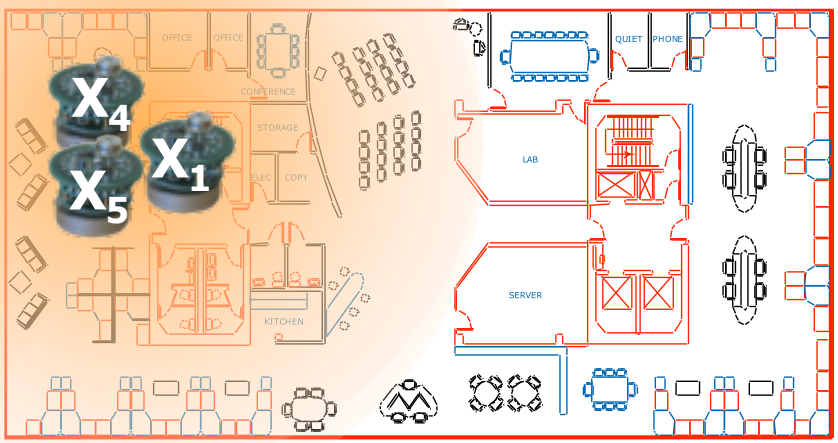
$$F(A) = H(\mathbf{Y}) - H(\mathbf{Y} \mid \mathbf{X}_A)$$

Uncertainty about temperature Y before sensing

Uncertainty about temperature Y after sensing



A={1,2,3}: High value F(A)



A={1,4,5}: Low value F(A)

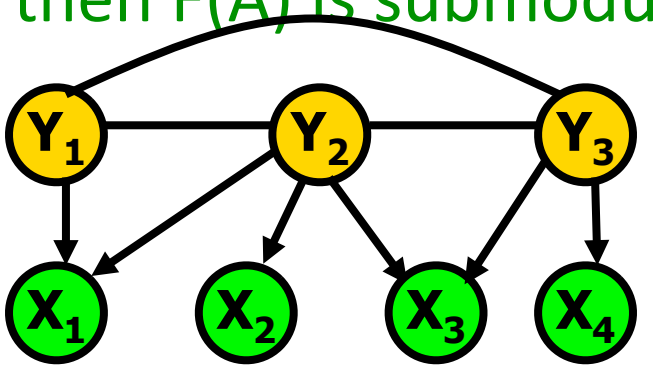
Example: Submodularity of info-gain

$Y_1,\ldots,Y_m$, X_1, …, X_n discrete RVs

$F(A) = I(Y; X_A) = H(Y)-H(Y \mid X_A)$

- F(A) is always monotonic
- However, NOT always submodular

Proposition: [Krause & Guestrin UAI' 05]
If X_i are all conditionally independent given Y, then F(A) is submodular!



Proof: Submodularity of information gain

$Y_1,\ldots,Y_m$, X_1, …, X_n discrete RVs

$F(A) = I(Y; X_A) = H(Y)-H(Y \mid X_A)$

Variables X independent given Y

$F(A \cup \{s\}) - F(A) = H(X_s \mid X_A) - H(X_s \mid Y)$

Nonincreasing in A:

$A \subseteq B : H(X_s|X_A) \geq H(X_s|X_B)$

„information never hurts“

Constant (indep. of A)

$F(A \cup \{s\})-F(A)$ monotonically nonincreasing

⇔ F submodular ☺

Sensor placement

Given: finite set V of locations, sensing quality F

Want: $\mathcal{A}^* \subseteq \mathcal{V}$ such that

$$\mathcal{A}^* = \operatorname*{argmax}_{|\mathcal{A}| \leq k} F(\mathcal{A})$$

NP-hard!

Greedy algorithm:

Start with A = {}

For i = 1 to k

s* := argmax_s F(A U {s})

A := A U {s*}

Sensor Placement = Constrained maximization

Theorem [Nemhauser et al '78]

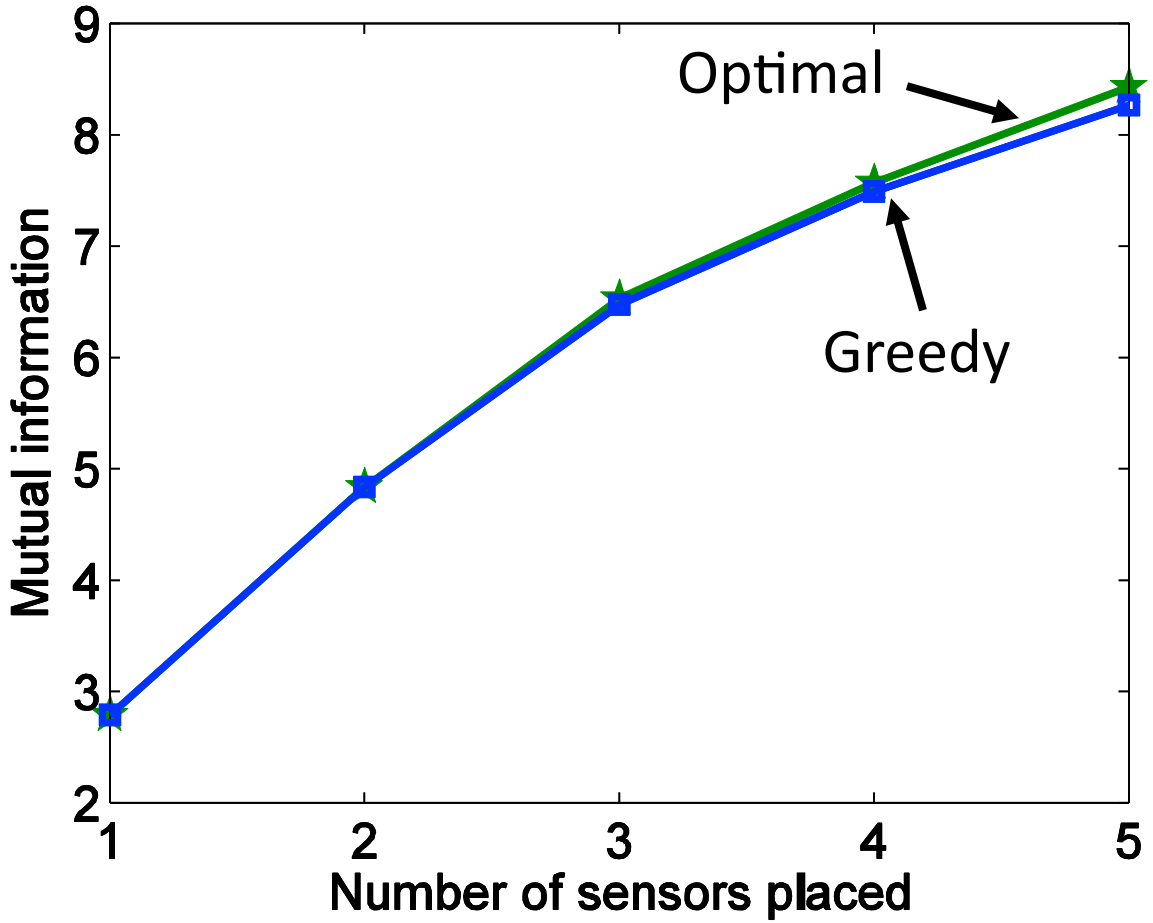
Suppose F is *monotonic* and *submodular*. Then

greedy algorithm gives constant factor approximation:

$$F(A_{greedy}) \geq \underbrace{(1-1/e)}_{\sim 63\%} \max_{|A| \leq k} F(A)$$

- Greedy algorithm gives near-optimal solution!
- For information gain: Guarantees best possible unless P = NP! [Krause & Guestrin '05]

Performance of greedy

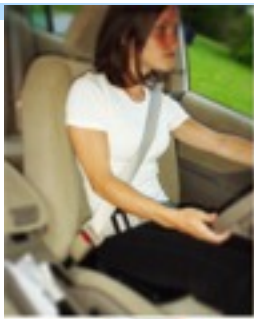


Temperature data from sensor network

- Greedy often empirically better than (1-1/e) OPT

Building a Sensing Chair
[Mutlu, Krause, Forlizzi, Guestrin, Hodgins UIST '07]

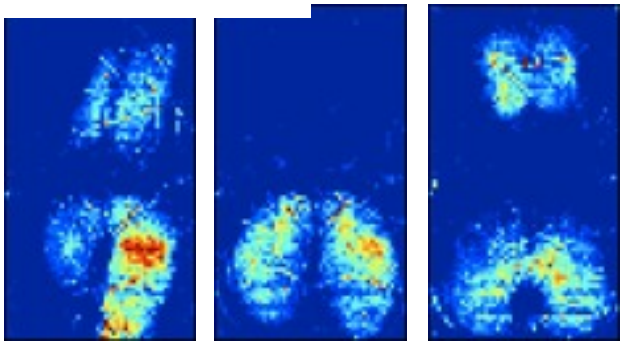
- People sit a lot
- Activity recognition in assistive technologies
- Seating pressure as user interface



Equipped with 1 sensor per cm^2!

Costs $6,000! ☹

Can we get similar accuracy with fewer, cheaper sensors?



Lean left | Lean forward | Slouch

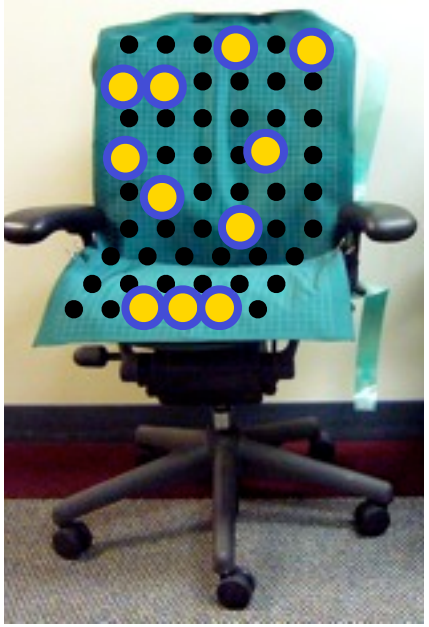
82% accuracy on 10 postures! [Tan et al]

How to place sensors on a chair?

- Sensor readings at locations V as random variables
- Predict posture Y using probabilistic model P(Y,V)
- Pick sensor locations A* from V to minimize entropy:

$$\mathcal{A}^* = \underset{|\mathcal{A}| \leq k}{\operatorname{argmax}} \operatorname{IG}(Y; X_{\mathcal{A}})$$

Possible locations **V**



Placed sensors, did a user study:

	Accuracy	Cost
Before	82%	**$6,000** ☹
After		

Similar accuracy at <2% of the cost!

Battle of the Water Sensor Networks Competition

[with Leskovec, Guestrin, VanBriesen, Faloutsos, J Wat Res Mgt 2008]

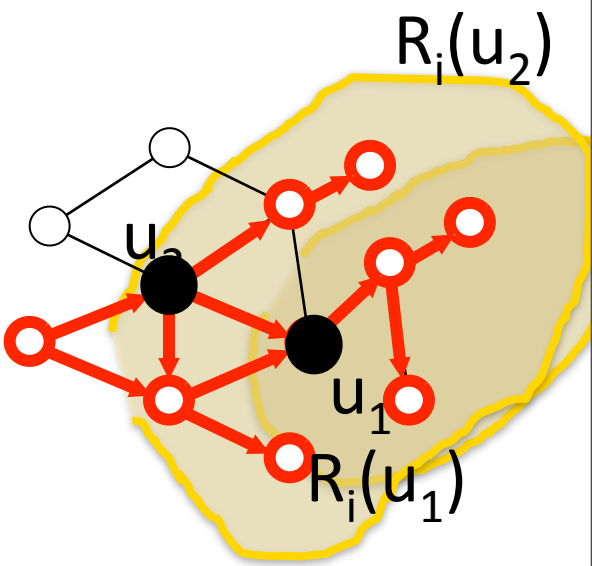
- Real metropolitan area network (12,527 nodes)
- Water flow simulator provided by EPA
- 3.6 million contamination events
- Multiple objectives: Detection time, affected population, …
- Place sensors that detect well “on average”

Reward function is submodular

- Claim:

  Reward function is monotonic submodular

- Consider event *i*:
  - $R_i(u_k)$ = benefit from sensor u_k in event i
  - $R_i(A) = \max R_i(u_k)$, $u_k \in A$
  - ⇒ R_i is submodular



Event i

- Overall objective:
  - $R(A) = \sum \text{Prob}(i)\, R_i(A)$

  ⇒ R is submodular

➔ Can use greedy algorithm to

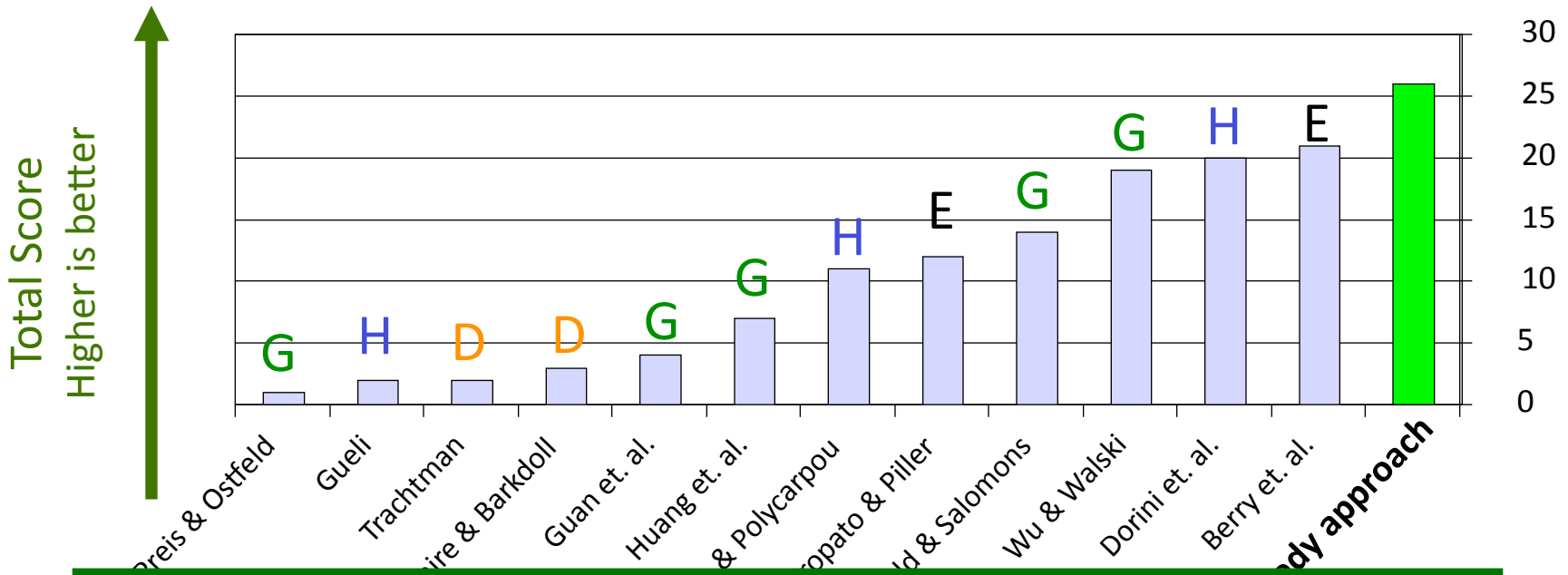
$$\max_{|A| \le k} R(A)$$

BWSN Competition results

- 13 participants
- Performance measured in 30 different criteria

G: Genetic algorithm D: Domain knowledge

H: Other heuristic E: "Exact" method (MIP)



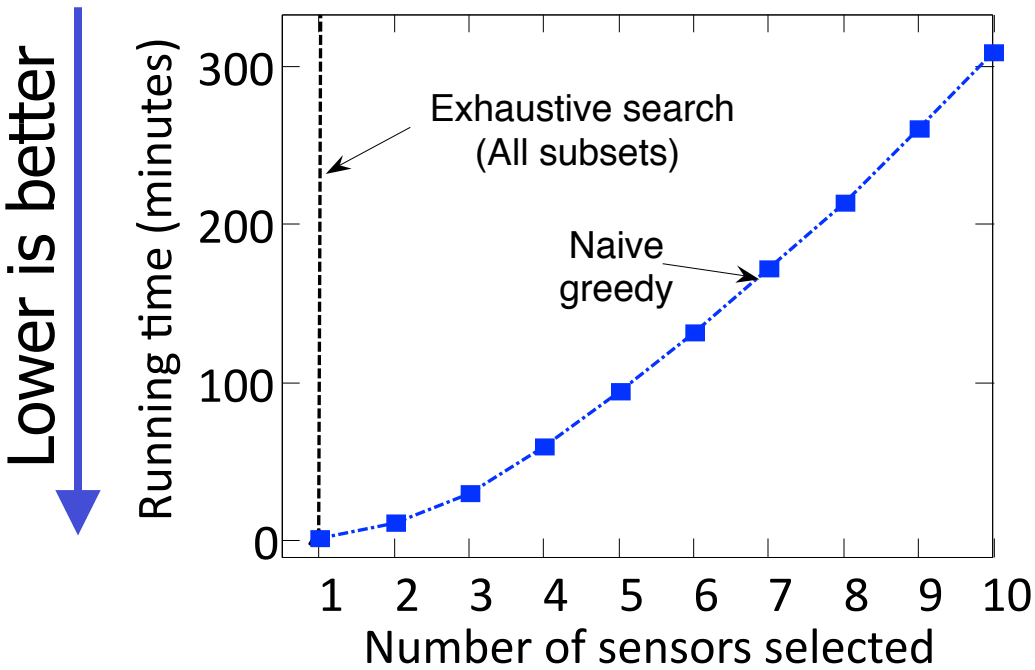
24% better performance than runner-up! ☺

What was the trick?

Simulated all **3.6M contaminations** on 2 weeks / 40 processors

152 GB data on disk, 16 GB in main memory (compressed)

➔ Very accurate computation of F(A) Very slow evaluation of F(A) ☹



30 hours/20 sensors

6 weeks for all

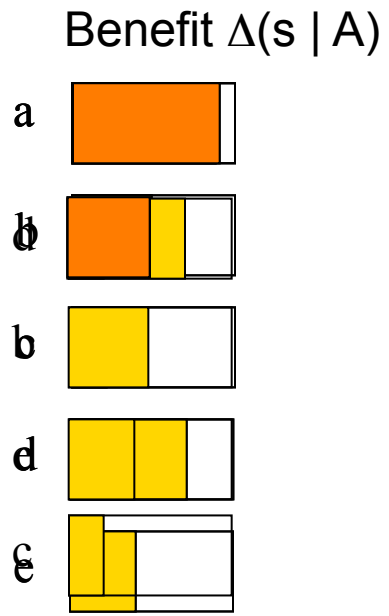
30 settings ☹

Submodularity to the rescue

"Lazy" greedy algorithm [Minoux '78]

Lazy greedy algorithm:

- First iteration as usual
- Keep an ordered list of marginal benefits Δ_i from previous iteration
- Re-evaluate Δ_i only for top element
- If Δ_i stays on top, use it, otherwise re-sort



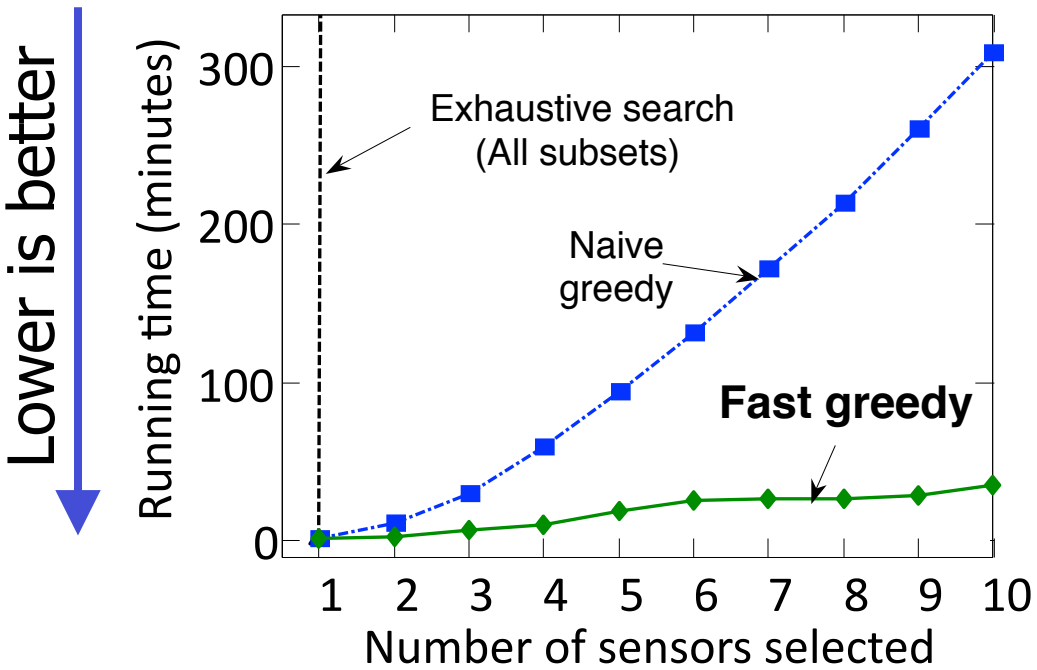
Note: Very easy to compute online bounds, use in other algo's, etc. [Leskovec, Krause et al. '07]

Result of lazy evaluation

Simulated all **3.6M contaminations** on 2 weeks / 40 processors

152 GB data on disk, 16 GB in main memory (compressed)

➔ Very accurate computation of F(A) Very slow evaluation of F(A) ☹



30 hours/20 sensors

6 weeks for all

30 settings ☹

ubmodularity to the rescue:

Using "lazy evaluations":

1 hour/20 sensors

Done after 2 days! ☺

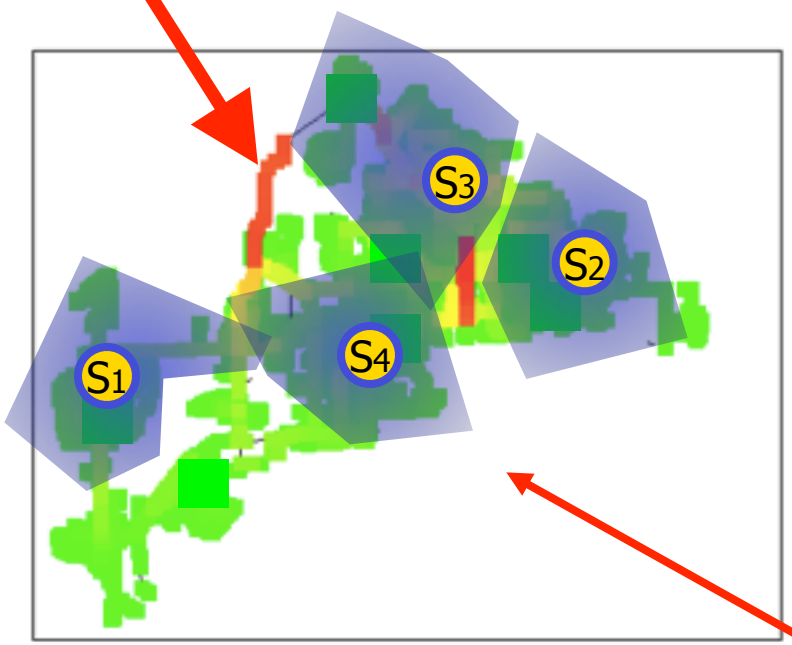
Other interesting directions

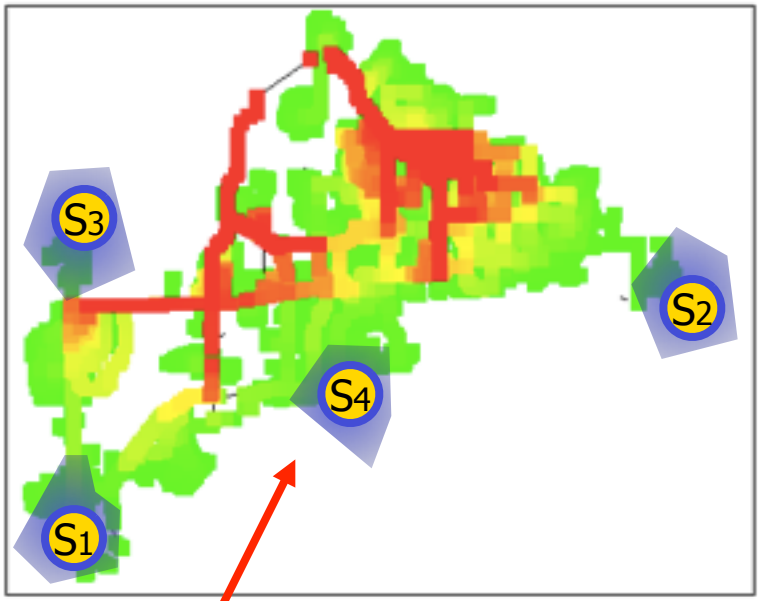
- Many sensing problems involve maximization of monotonic submodular functions
  - Can use greedy algorithm to get near-optimal solutions!

- How can we handle more complex settings:
  - Adversarial settings?
  - Complex constraints / cost functions?
  - Sequential decisions?

What about worst-case? [Krause et al., JMLR'08]

Knowing the sensor locations, an adversary contaminates here!





Placement detects well on **"average-case"** (accidental) contamination

Very different average-case impact, **Same** worst-case impact

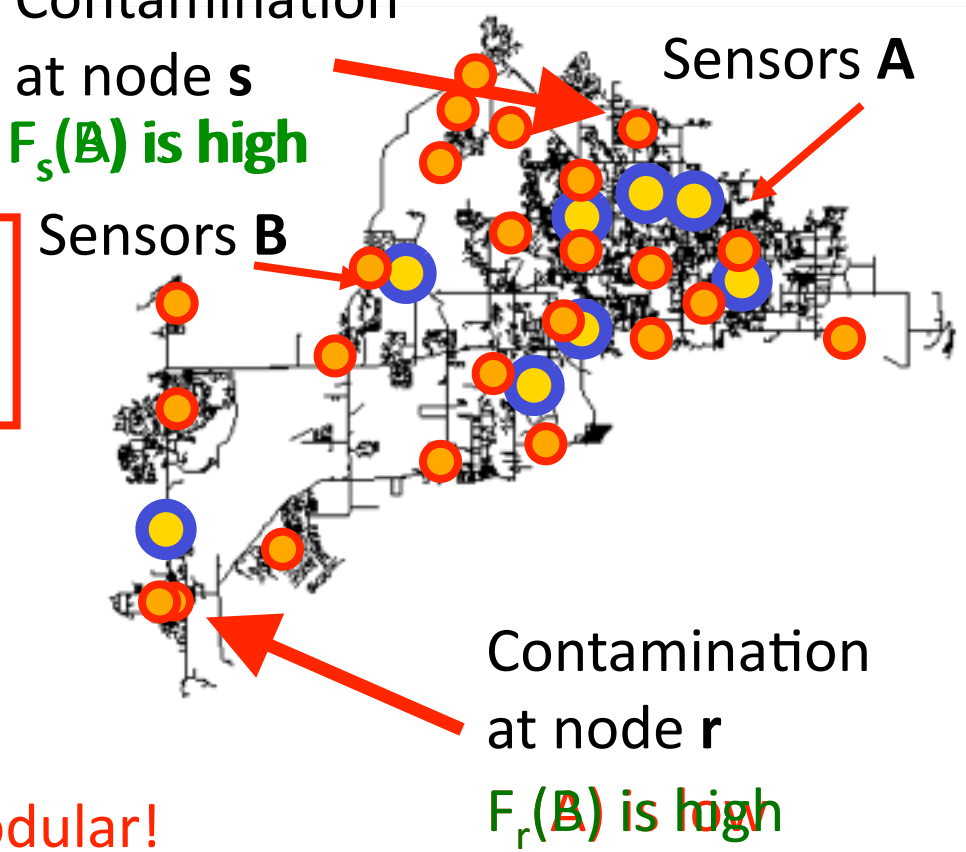
Where should we place sensors to quickly detect in the **worst case**?

Optimizing for the worst case

- Separate utility function F_i for each contamination i
- $F_i(A)$ = impact reduction by sensors A for contamination i

Want to solve

$$\mathcal{A}^* = \operatorname*{argmax}_{|\mathcal{A}| \leq k} \min_i F_i(\mathcal{A})$$



Each of the F_i is submodular

Unfortunately, $\min_i F_i$ not submodular!

The *SATURATE* algorithm

Implement using binary search

1. Guess optimal value c
2. Find cheapest solution A s.t. $F_i(A) \geq c$ for all i

   (i.e., all objectives F_i are "saturated")

Implement using a greedy algorithm!

Theoretical guarantees

Theorem: *SATURATE* finds a solution A_S such that

$$\min_i F_i(A_S) \geq OPT_k \text{ and } |A_S| \leq \alpha\, k$$

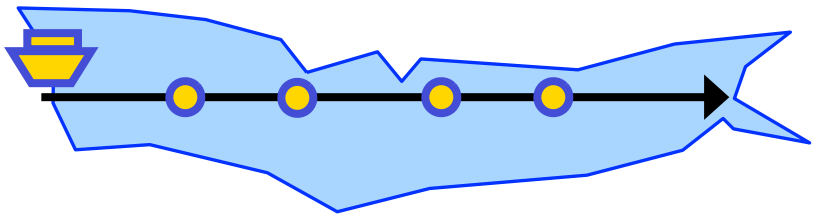
Optimal value achievable using k sensors

$\alpha > 1$, depends *logarithmically* on problem instance

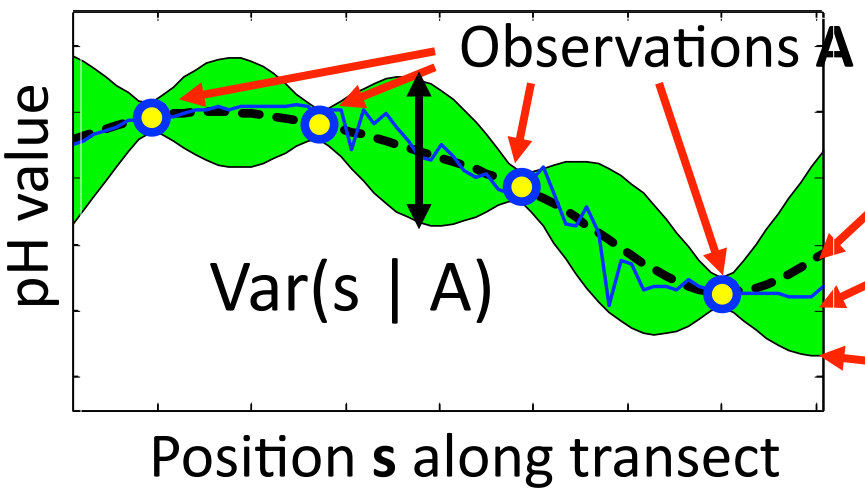
No better guarantees possible under reasonable complexity theoretic assumptions!

Example: Lake monitoring

- Monitor pH values using robotic sensor



transect



Prediction at unobserved locations

True (hidden) pH values

Use **probabilistic model** (Gaussian processes) to estimate prediction error

Where should we sense to **minimize our maximum error**?

$$A^* = \operatorname{argmin} \max \operatorname{Var}(s \mid A) = \operatorname{argmax} \min_s \underbrace{\operatorname{Var}(s) - \operatorname{Var}(s \mid \mathcal{A})}$$

(often) submodular [Das & Kempe '08]

➔ **Robust sensing problem!**

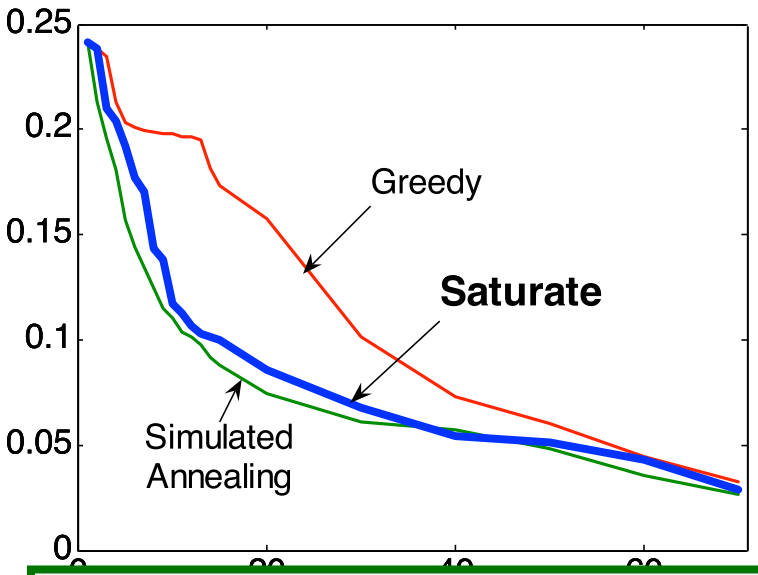
Comparison with state of the art

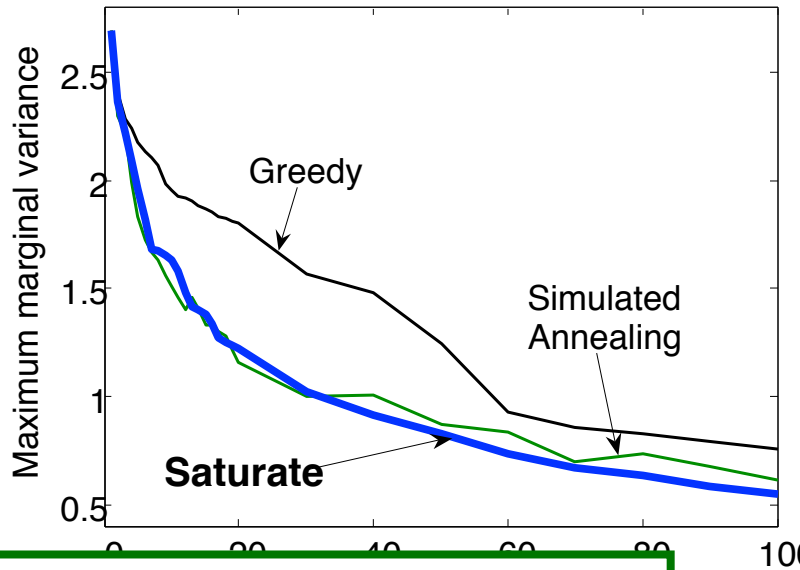
Algorithm used in geostatistics: *Simulated Annealing*

[Sacks & Schiller '88, van Groeningen & Stein '98, Wiens '05,…]

7 parameters that need to be fine-tuned

better

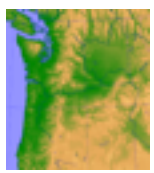




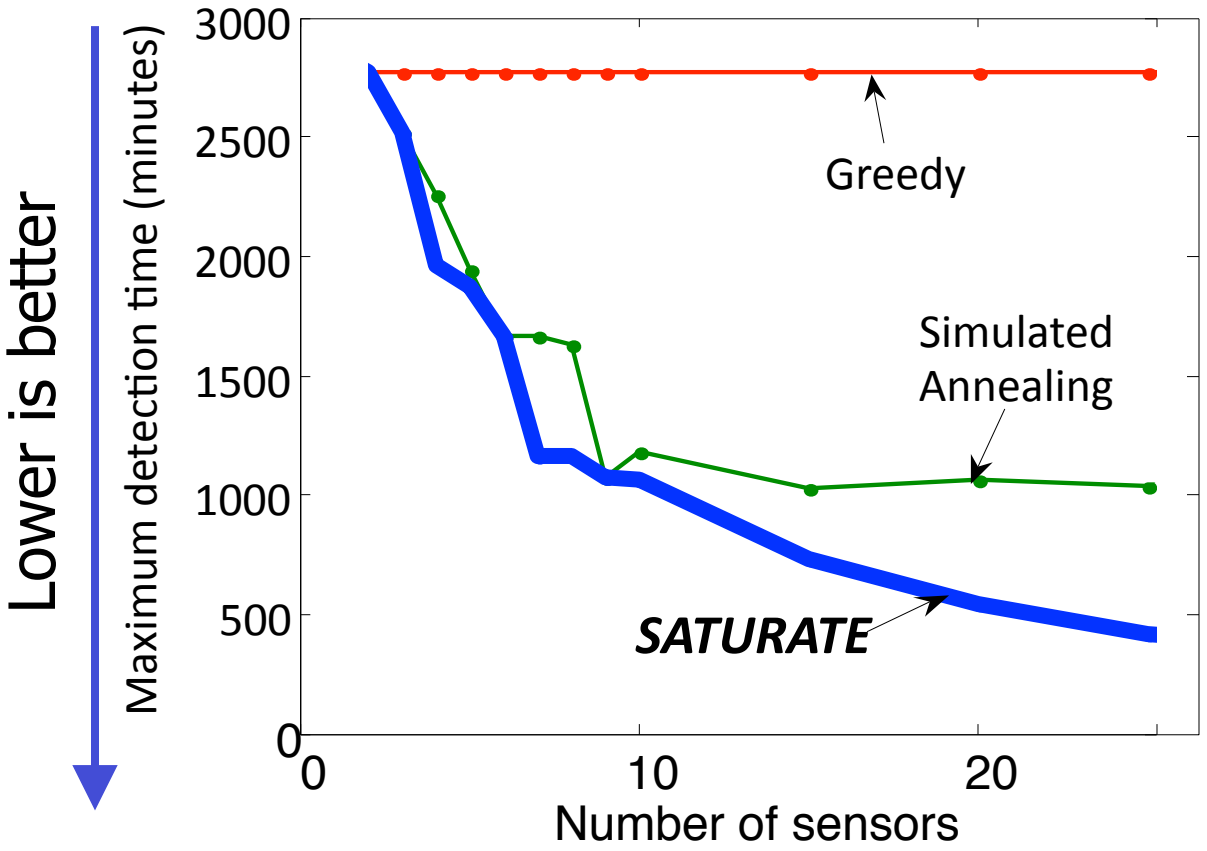
Enviro

***SATURATE* is competitive & 10x faster**

No parameters to tune!



Results on water networks



No decrease until **all** contaminations detected!

Water networks

60% lower worst-case detection time!

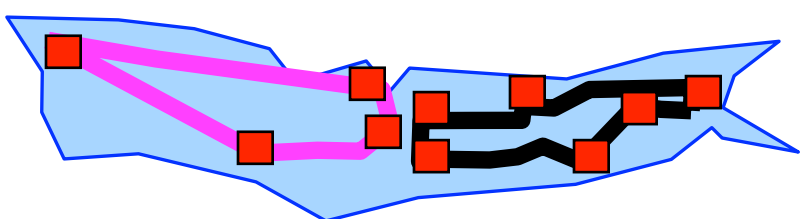
Other aspects: Complex constraints

$\max_{\mathbf{A}} F(\mathbf{A})$ or $\max_{\mathbf{A}} \min_i F_i(\mathbf{A})$ subject to

- So far: $|\mathbf{A}| \leq k$
- In practice, more complex constraints:
- Non-uniform modular cost $C(A) \leq B$ [e.g., Sviridenko '04, Leskovec et al '07]
- Time → Matroid constraints [e.g., Williams et al '07, Krause et al '08]

Locations need to be connected by paths

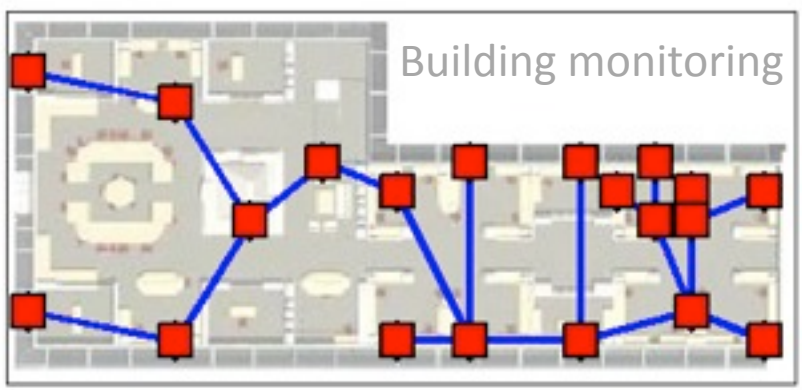
[Chekuri & Pal, FOCS '05]
[Singh et al, IJCAI '07 & '09]



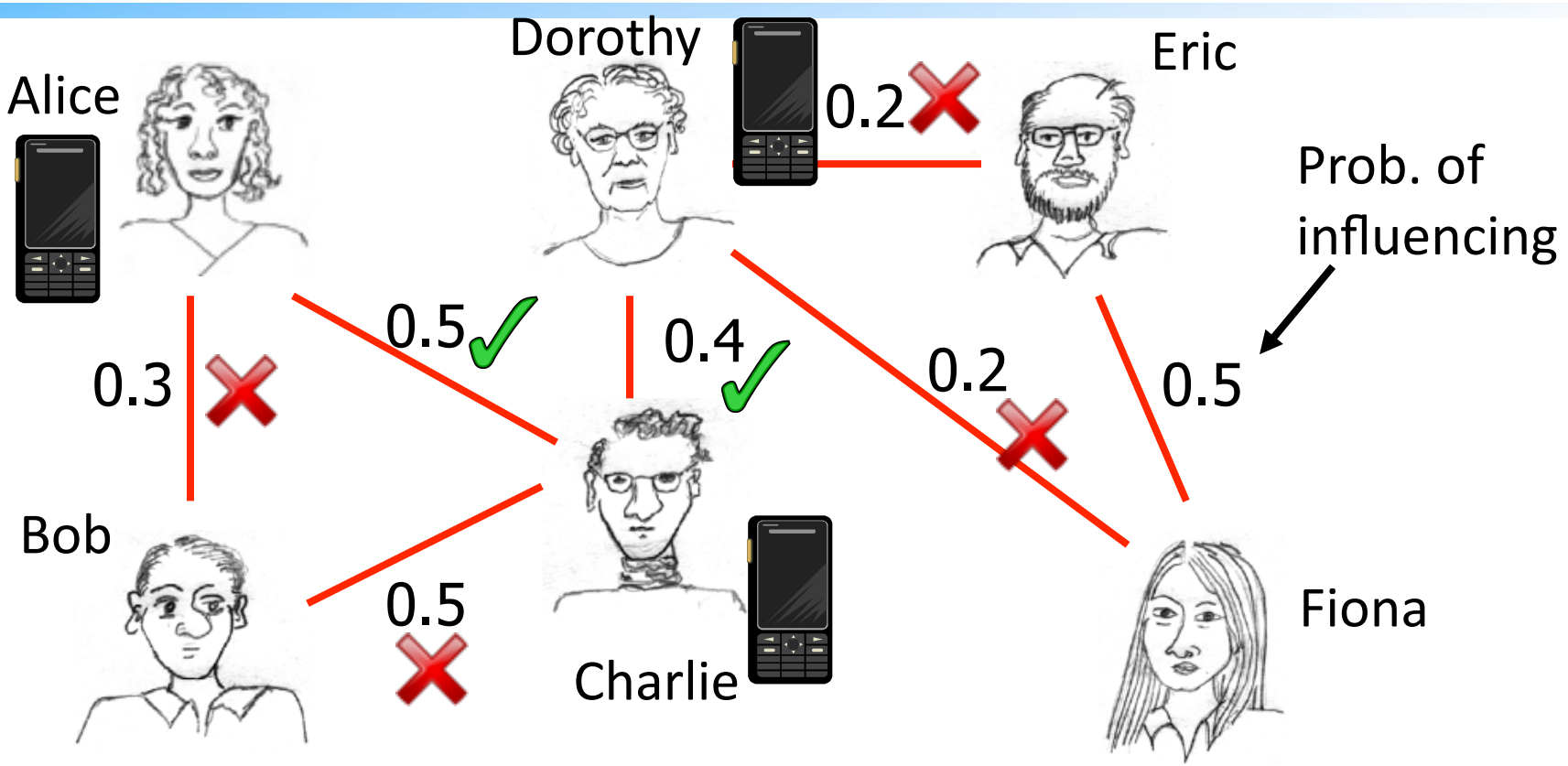
Lake monitoring

Sensors need to communicate (form a routing tree)

[Calinescu et al., ISAAC'04;
Krause et al., IPSN 06, TOSN '11]



Influence maximization in social networks

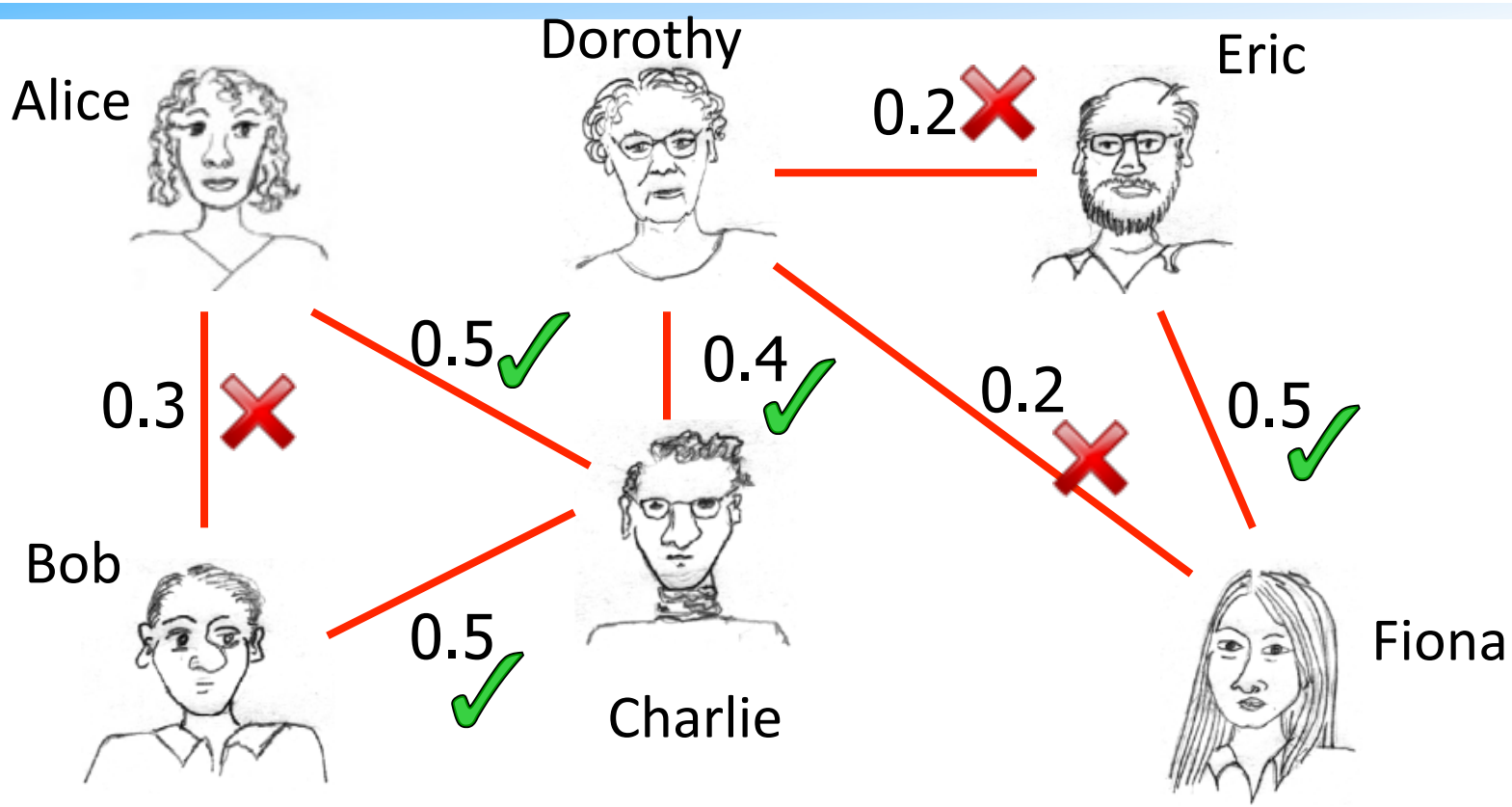


Who should get free cell phones?

V = {Alice,Bob,Charlie,Dorothy,Eric,Fiona}

F(A) = Expected number of people influenced when targeting A

Influence in social networks is submodular



Key idea: Flip coins **c** in advance ➔ "live" edges

$F_c(A)$ = People influenced under outcome **c** (set cover!)

$F(A) = \sum_c P(c)\, F_c(A)$ is submodular as well!

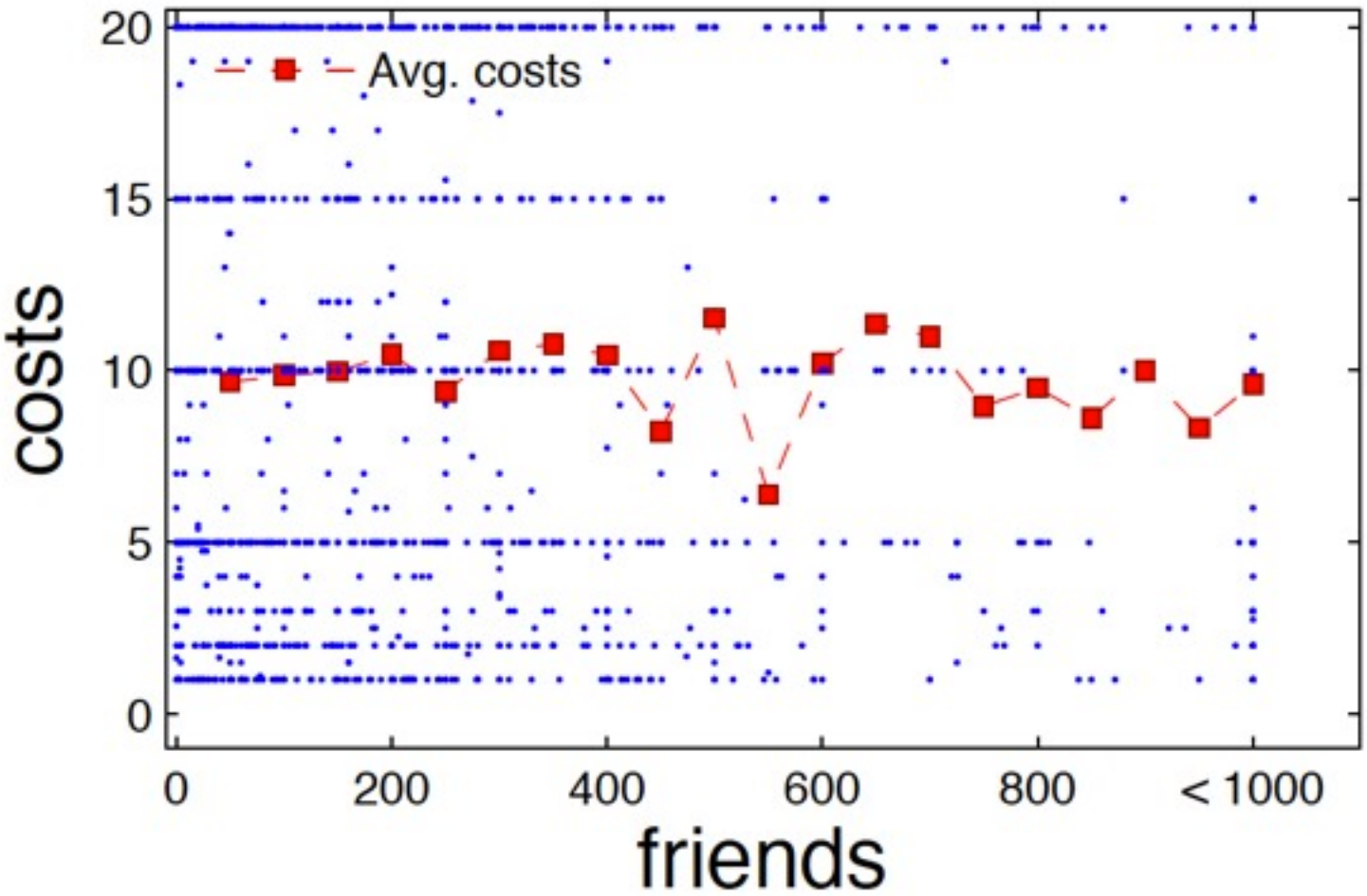
Incentive compatible submodular maximization

- Suppose each agent has a private cost
- How should we incentivize agents to truthfully report their true cost and simultaneously maximize influence?

Theorem [Singer '10]: There is a randomized mechanism for monotonic submodular maximization that

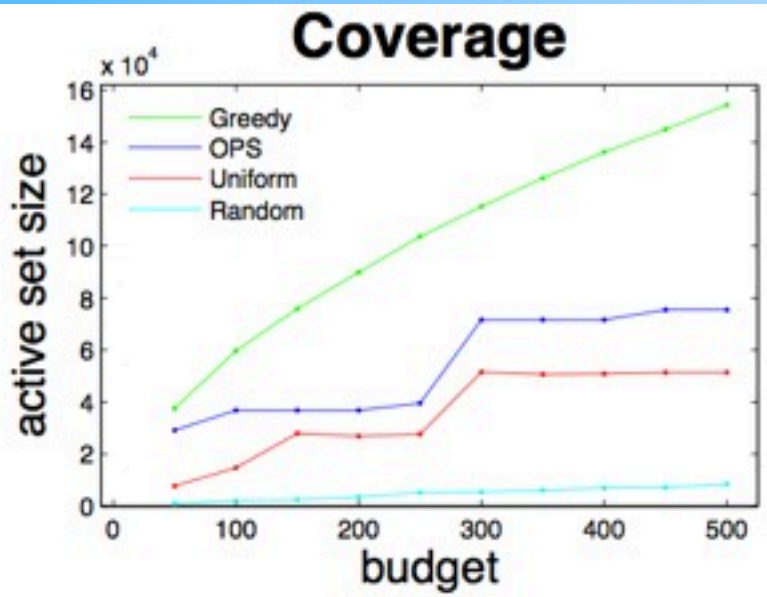
- is incentive compatible,
- budget feasible and
- provides a constant factor approximation guarantee

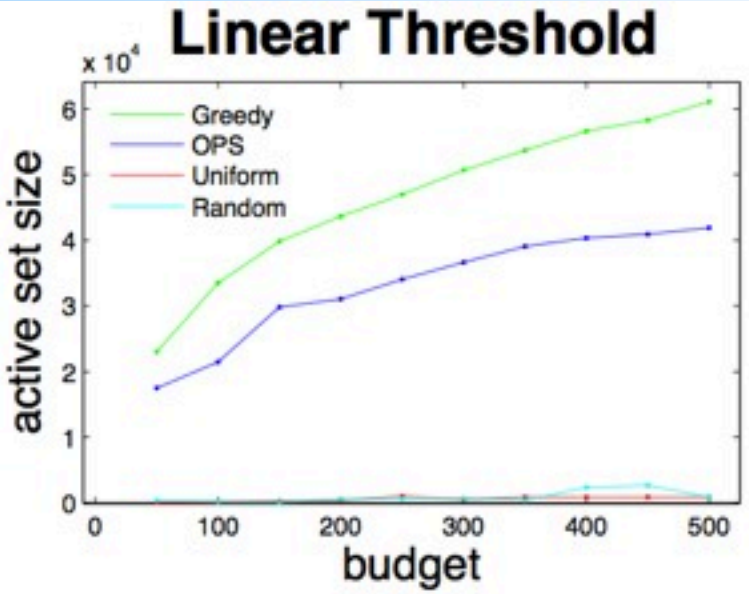
Amazon Mechanical Turk experiment

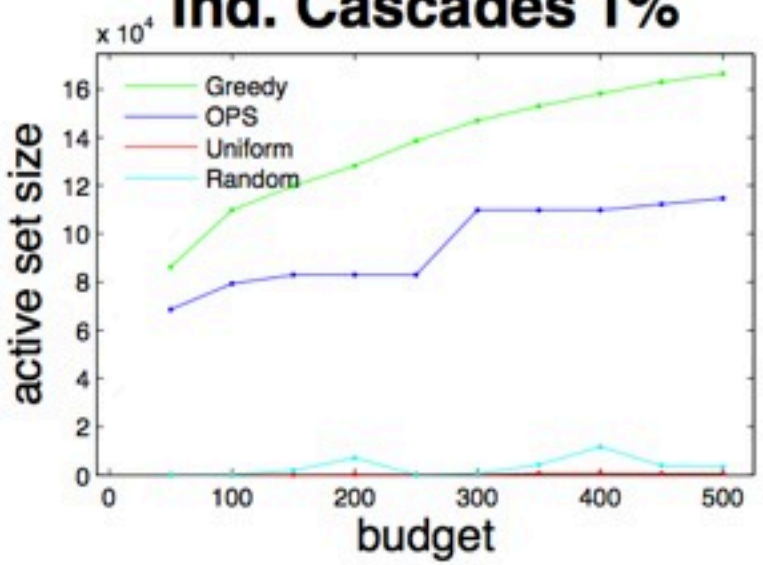


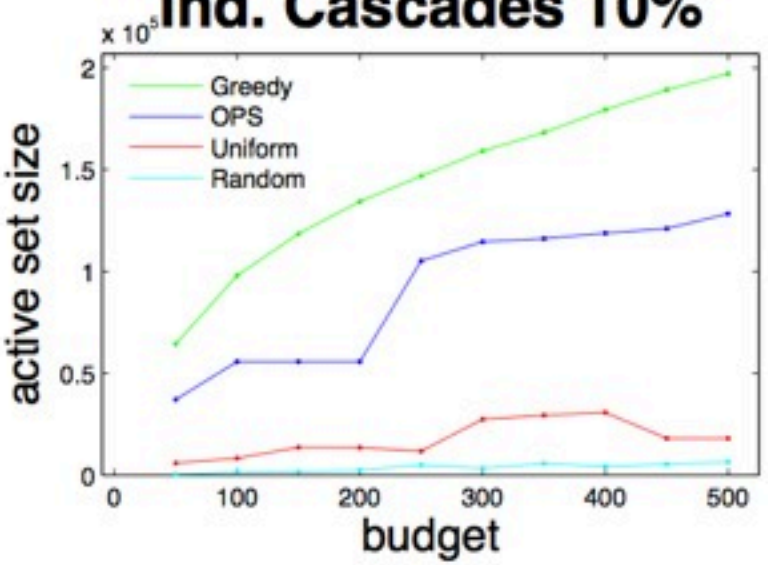
- What‘s your „cost“ to post an ad on your Facebook page?

Simulations on 1M node Facebook subgraph [Singer WSDM '12]

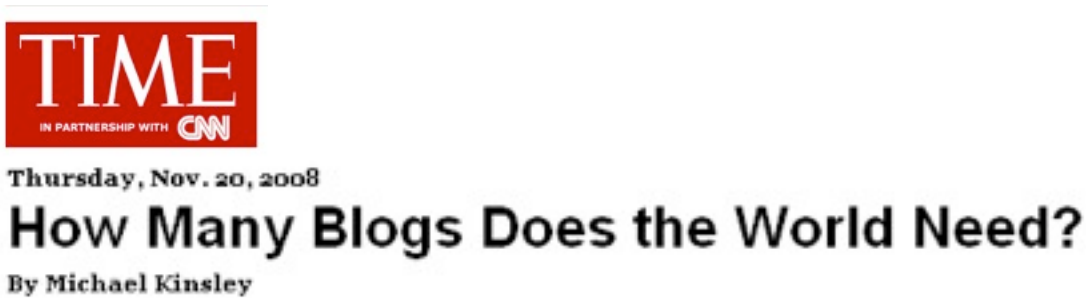








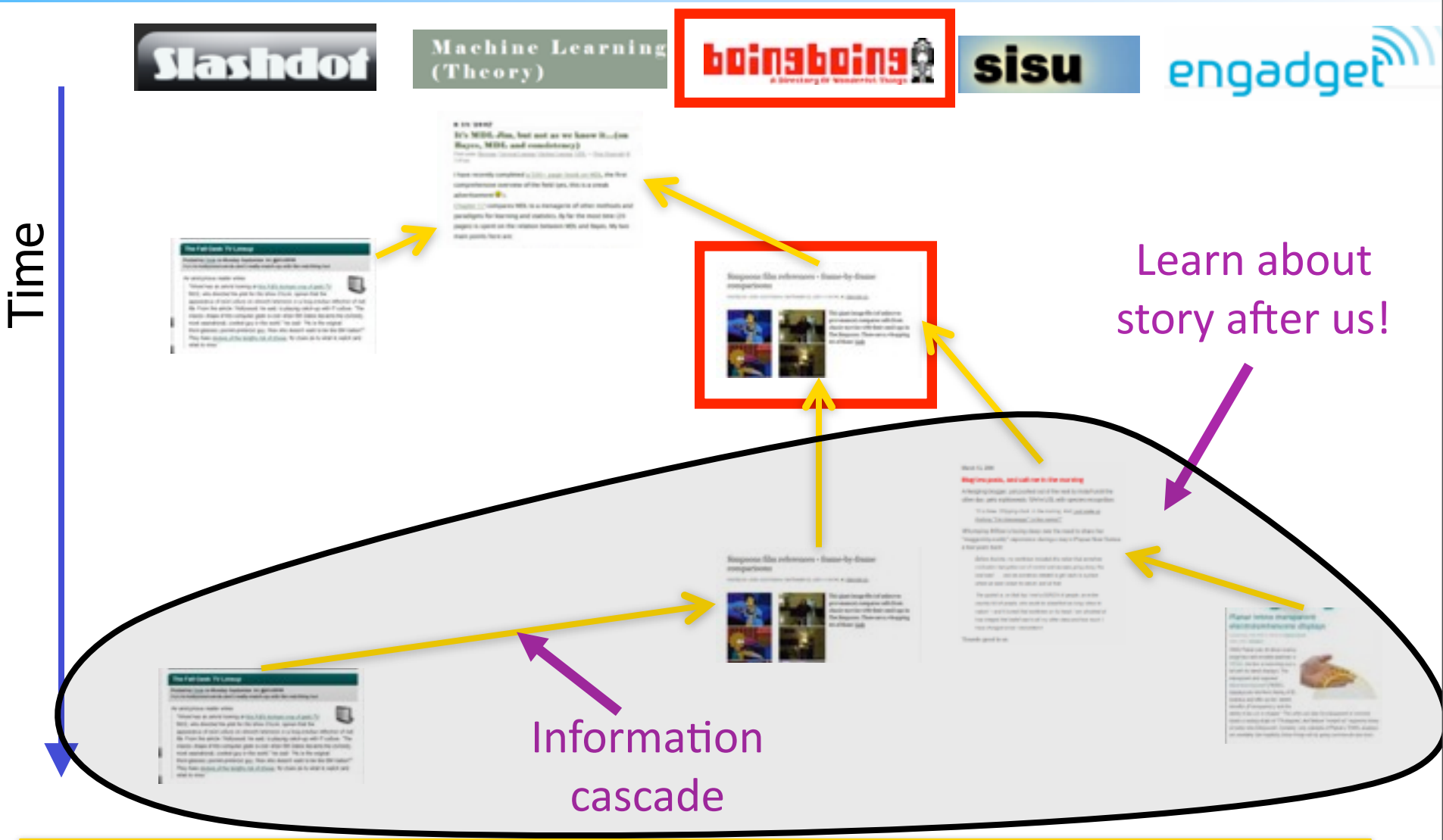
Submodularity in Social Media



I have 10 minutes. Which blogs should I read to be most up to date?

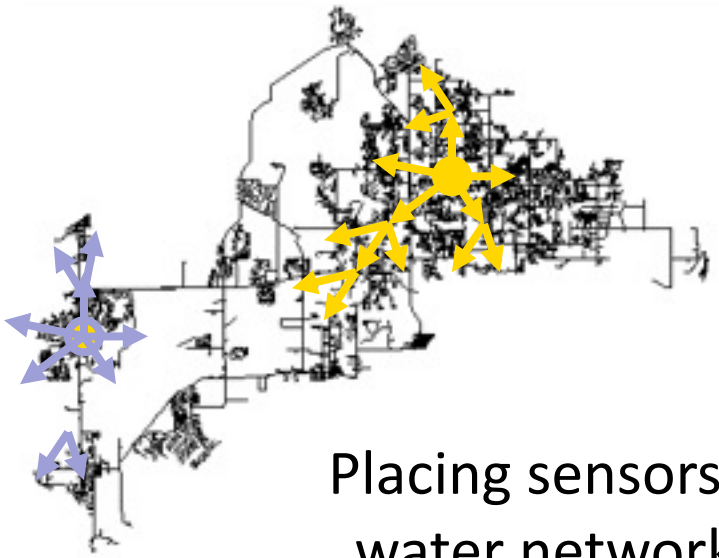
Cascades in the Blogosphere

[with Leskovec, Guestrin, Faloutsos, VanBriesen, Glance KDD 07 – *Best Paper*]



Which blogs should we read to learn about big cascades early?

Water vs. Web



Placing sensors in water networks

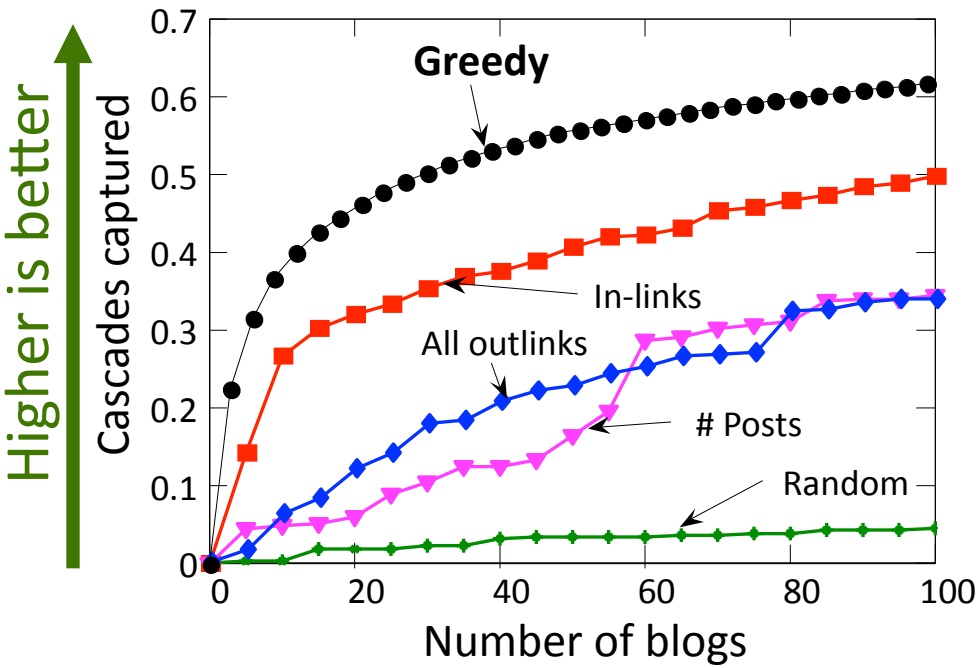
vs.

Selecting informative blogs

- In both problems we are given
  - Graph with nodes (junctions / blogs) and edges (pipes / links)
  - Cascades spreading dynamically over the graph (contamination / citations)
- Want to pick nodes to **detect big cascades early**

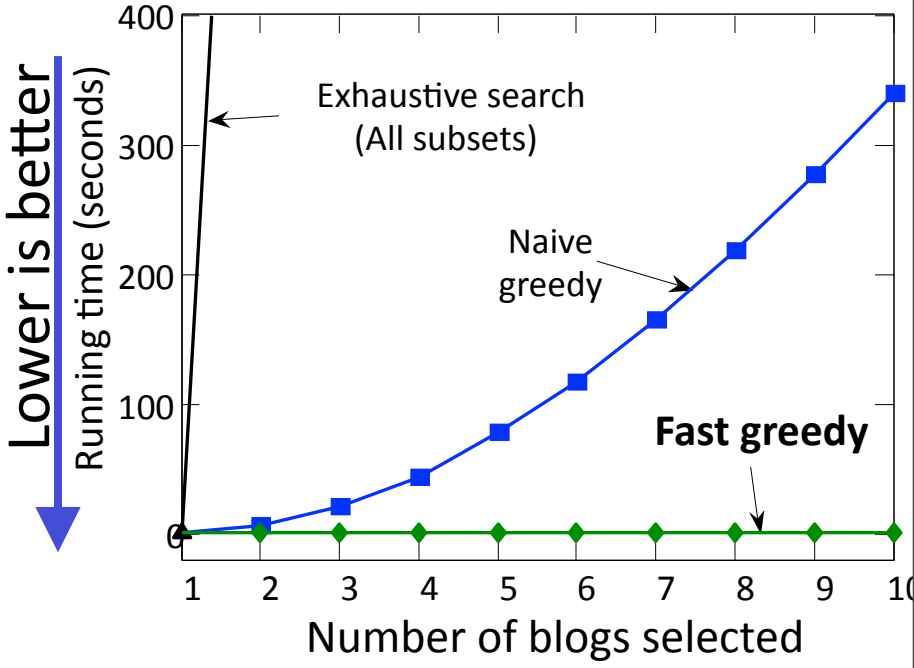
In both applications, utility functions submodular

Performance on Blog selection



Blog selection

~45k blogs



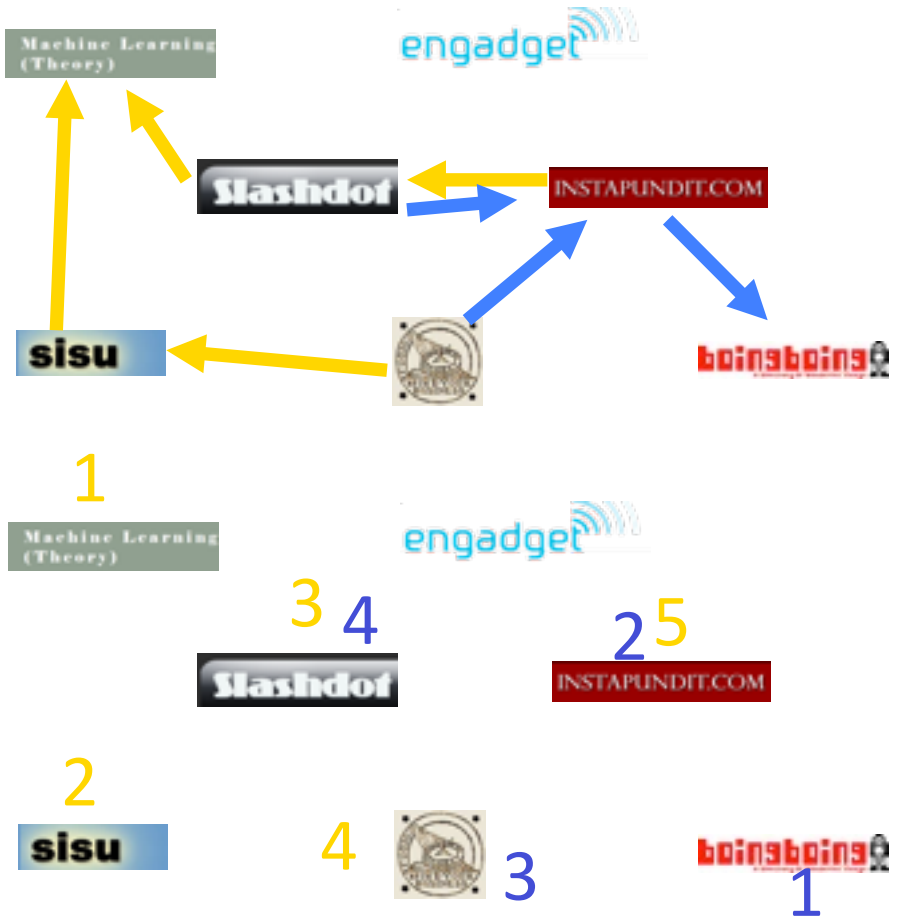
Blog selection

Outperforms state-of-the-art heuristics

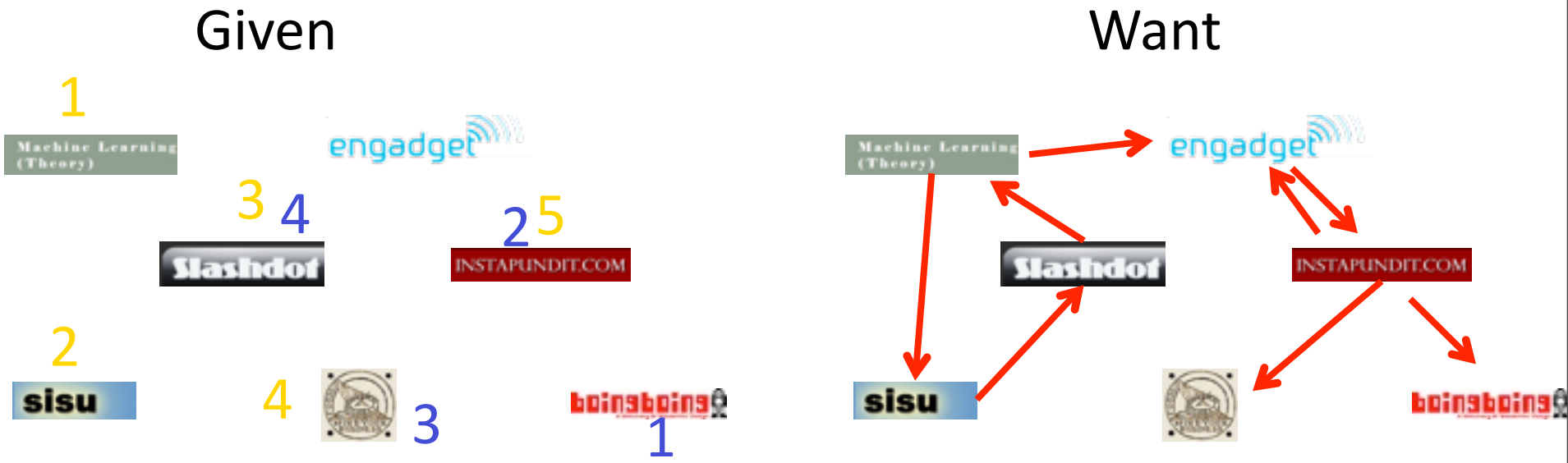
700x speedup using submodularity!

Inferring Networks of Diffusion
[with Leskovec, Gomez-Rodriguez ACM Trans. KDD 2012]

- Want to detect information cascades
- Often, only know time of occurrence, not links
- Examples:
  - Information propagation
  - Epidemics
- Want to infer underlying network (edges and directions)



Inferring diffusion networks



- Given traces of influence, wish to infer sparse directed network G=(V,E)
- Goal: Formulate as optimization problem

$$E^* = \arg\max_{|E| \leq k} F(E)$$

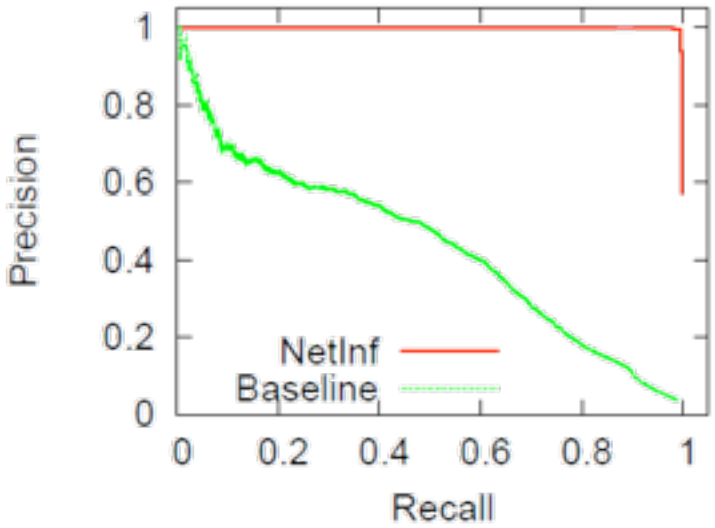
Estimation problem

- Many influence trees T consistent with data
- For cascade C_i, model $P(C_i \mid T)$
- Find sparse graph that maximizes likelihood for all observed cascades

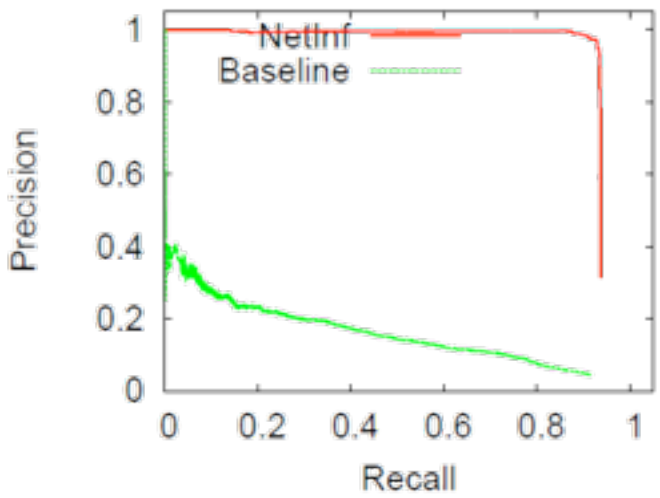
➔ Log likelihood monotonic submodular in selected edges

$$F(E) = \sum_i \log \max_{\text{tree } T \subseteq E} P(C_i \mid T)$$

Evaluation: Synthetic networks



1024 node hierarchical Kronecker exponential transmission model

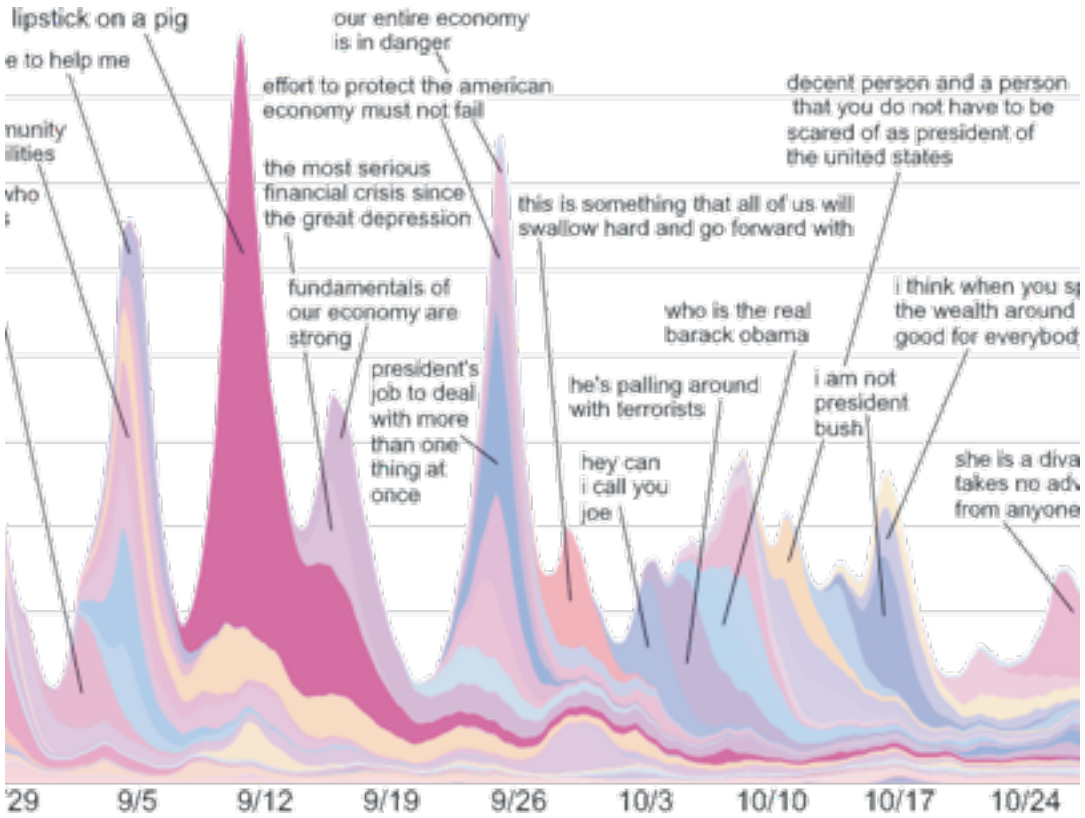


1000 node Forest Fire (α = 1.1) power law transmission model

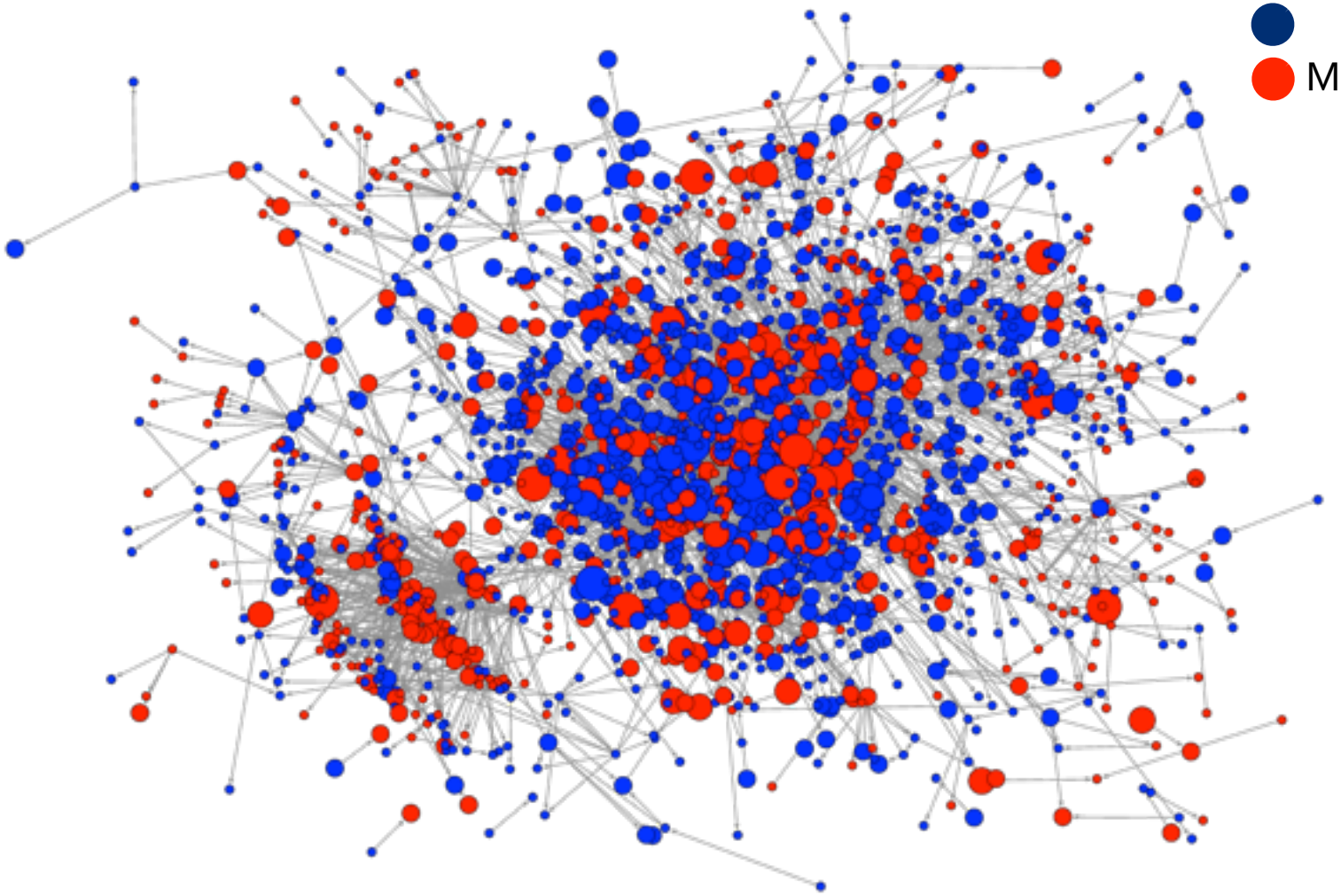
- Performance does not depend on the network structure:
  - Synthetic Networks: Forest Fire, Kronecker, etc.
  - Transmission time distribution: Exponential, Power Law
- Break-even point of > 90%

Experiments on real data

- MemeTracker data set [Leskovec et al]
  - 172m news articles
  - Aug ’08 – Sept ‘09
  - 343m textual phrases (quotes)
- We infer the network of information diffusion

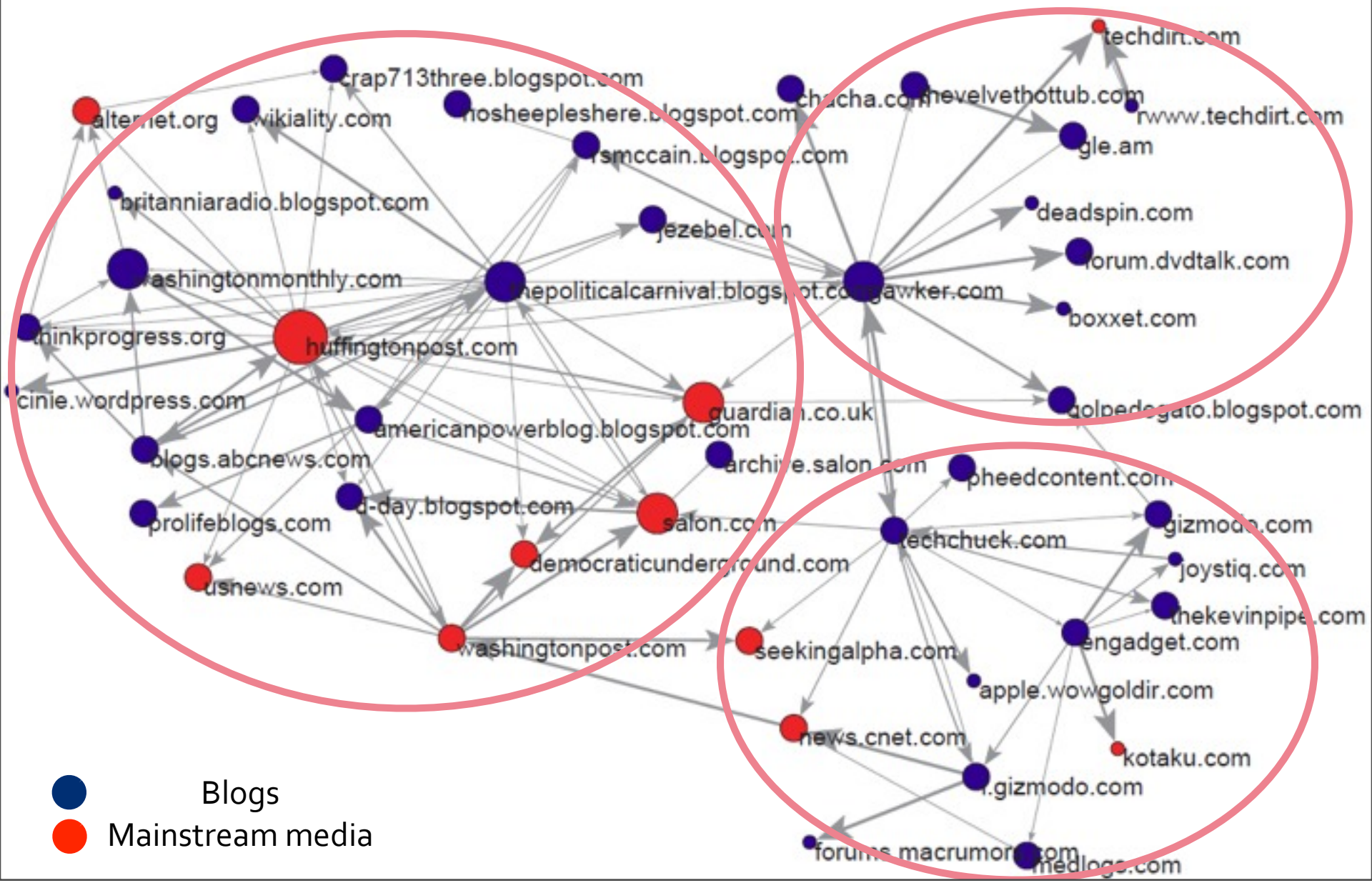


Diffusion Network



Actual network inferred from 172 million articles from 1 million news sources

Diffusion Network (small part)



Conclusions

- Discrete optimization abundant in sensor and social network problems
- Fortunately, many of them are submodular
- Interesting optimization problems in these domains
  - Robust optimization (minimax and randomized)
  - Complex constraints (path / tree constraints etc.)
  - Game-theoretic considerations (incentive-compatibility, …)

ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

inf | Informatik
Computer Science

Tutorial on Submodular Optimization in Sensor and Social Networks PART II

Andreas Krause

Modern Aspects of Submodularity
GeorgiaTech March 2012

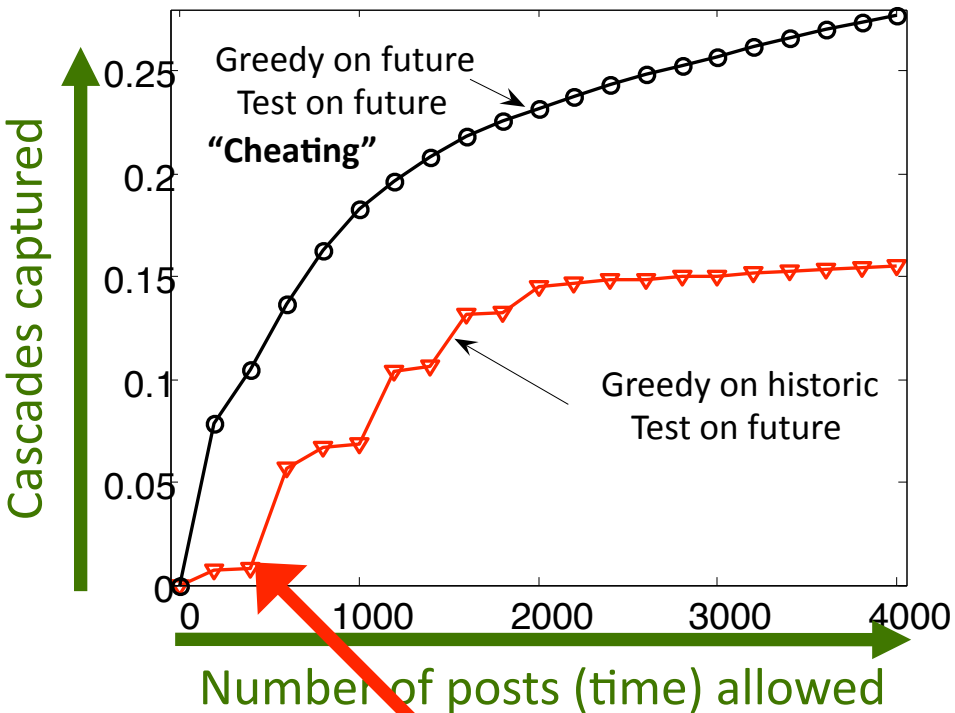
Learning to optimize submodular functions

- Online submodular optimization
  - Learn to pick a sequence of sets to maximize a sequence of (unknown) submodular functions
  - *Application*: Learning to rank; sponsored search

- Adaptive submodular optimization
  - Gradually build up a set, taking into account feedback
  - *Application*: Sequential experimental design

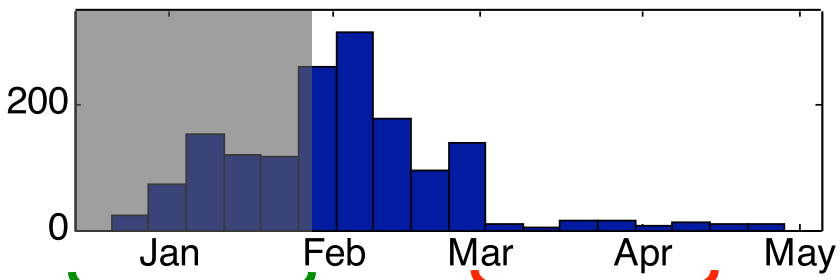
Predicting the "hot" blogs

- Want blogs that will be informative in the future
- Split data set; train on historic, test on future



Let's see what goes wrong here.

Detects on training set



Blog selection "overfits" to training data!

Detect well here!

Detect poorly here!

Poor generalization!

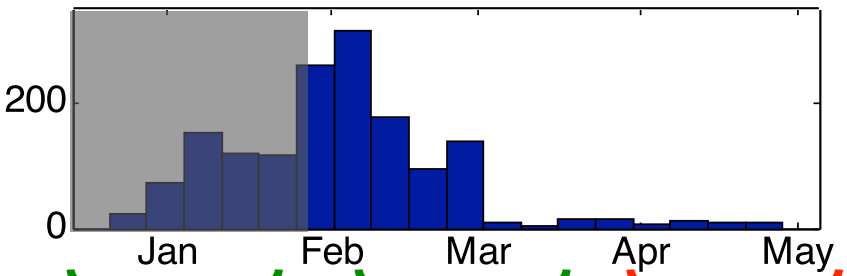
Why should we?

Want blogs that continue to do well!

Online optimization

"Overfit" blog selection **A**

$F_i(A)$ = detections in interval i



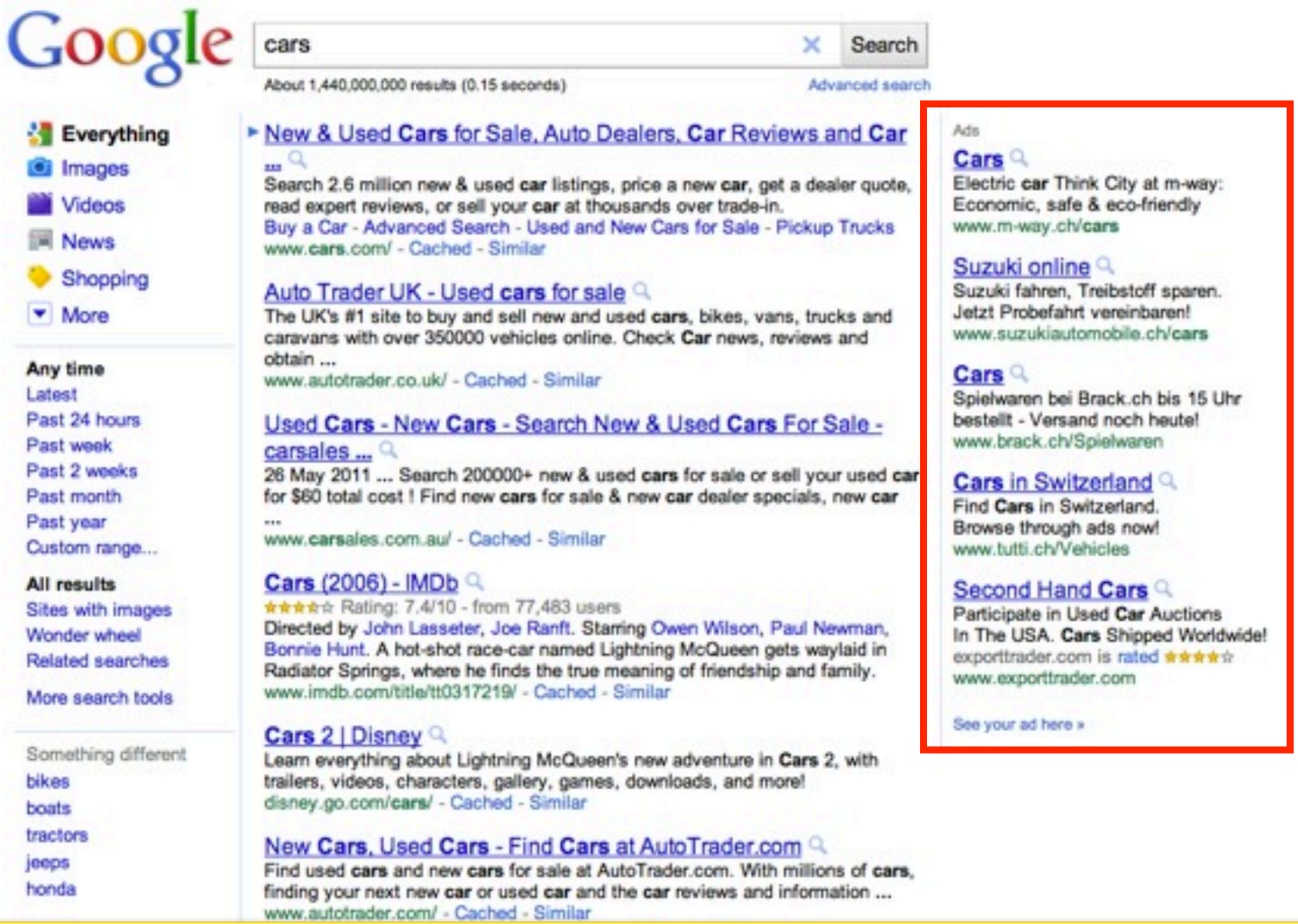
$F_1(A)=.5$ $F_3(A)=.6$ $F_5(A)=.02$

$F_2(A)=.8$ $F_4(A)=.01$

Online optimization: $\max \sum_{t=1}^{T} F_t(A_t)$

How should we choose $A_1, \ldots, A_T$ to maximize performance??

Sponsored search



Which **set of ads** should be displayed to maximize revenue?

Pay per click model

- Search engine only gets paid if user clicks on ad

- Simplistic model:
  - Every round t we pick set A_t of ads
  - A user arrives, and is willing to click on an (unknown) subset S_t of ads
  - Search engine receives payoff:

$$F_t(A_t) = \min(|A_t \cap S_t|, 1)$$

- Want to maximize total number of clicks

Online learning

- For round t=1:T
  - Decision maker chooses action $a_t \in A = \{1, \ldots, n\}$
  - Nature chooses reward function $f_t : A \to \mathbb{R}$ and reveals f_t (full information) or only $f_t(a_t)$ („bandit“)
- Decision maker suffers regret

$$R_T = \max_{a \in A} \left[\sum_{t=1}^{T} f_t(a) \right] - \sum_{t=1}^{T} f_t(a_t)$$

- **Goal**: wish to obtain „no-regret“ $R_T / T \to 0$

Online learning algorithms

- Long history in machine learning
- Can get no-regret algorithms in full information and „bandit“ setting
- Regret bounds scale polynomially with number n of choices
- What if $A = 2^V$ and f_t are submodular??

Online Greedy Algorithm

Replace each stage of greedy algorithm with an expert / multi-armed bandit algorithm.

Select $\{ a_1, a_2, a_3, \ldots, a_k \}$.

Feedback to $\mathcal{E}_j$ for action a_j is

$F_t(\{a_1, a_2, \ldots, a_{j-1}, a_j\}) - F_t(\{a_1, a_2, \ldots, a_{j-1}\})$

In bandit setting, using unbiased estimate suffices

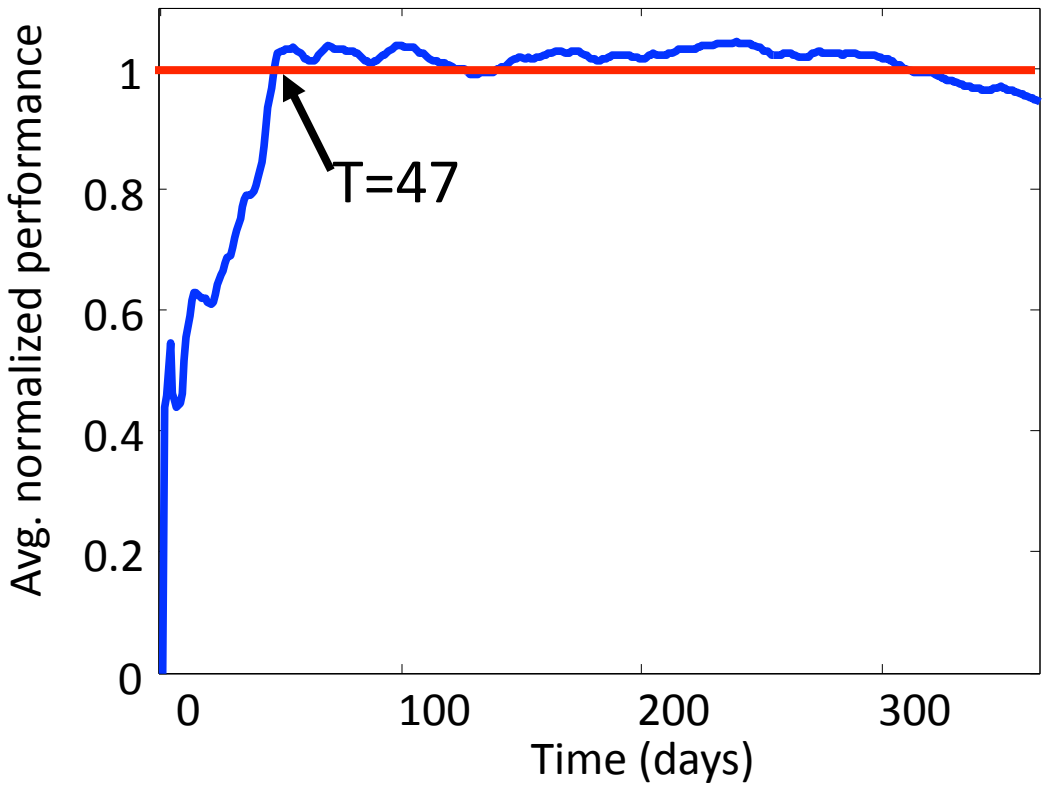
Online maximization of submodular functions

Theorem
Online greedy algorithm chooses $A_1,\ldots,A_T$ s.t. in expectation, for any sequence $F_1,\ldots,F_T$ as $T \rightarrow \infty$

$$\sum_{t=1}^{T} F_t(A_t) \geq \max_{|A|\leq k} \sum_{t=1}^{T} F_t(A) - O\left(nT^{2/3}\right)$$

Can get 'no-regret' over greedy algorithm in hindsight

Results on blogs



Performance of online algorithm converges quickly to "cheating" offline greedy algorithm!

Other results on online submodular optimization

- Online submodular maximization
  - No (1-1/e) regret for ranking (partition matroids) [Streeter, Golovin, K NIPS 2009]
  - Distributed implementation [Golovin, Faulkner, K IPSN 2010]
  - Improved bounds for bandits with linear combinations of SFs [Yue, Guestrin, NIPS 2011]
- Online submodular coverage
  - Min-cost / Min-sum submodular cover [Streeter & Golovin NIPS 2008]
  - Guillory & Bilmes [NIPS 2011]
- Online Submodular Minimization

Learning to optimize submodular functions

- Online submodular optimization
  - Learn to pick a sequence of sets to maximize a sequence of (unknown) submodular functions
  - *Application*: Learning to rank; sponsored search

- Adaptive submodular optimization
  - Gradually build up a set, taking into account feedback
  - *Application*: Sequential experimental design

Adaptive Sensing / Diagnosis

Want to effectively diagnose while minimizing cost of testing!

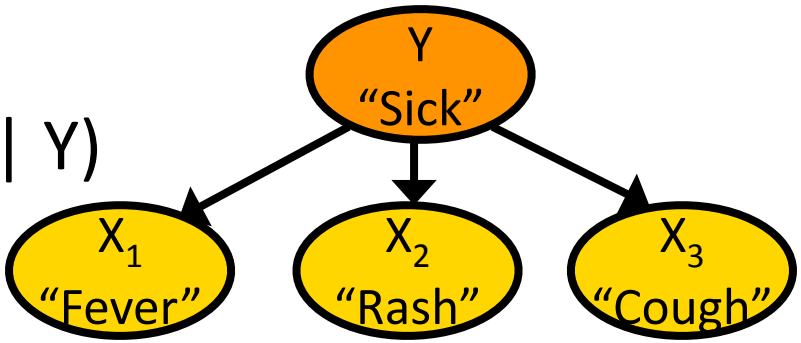
Interested in a *policy* (decision tree), not a *set.*

Can't use submodular set functions!!

Can we generalize submodularity for sequential decision making?

Adaptive selection in diagnosis

- Prior over diseases P(Y)
- Deterministic test outcomes P($\mathbf{X}_V$ | Y)



- Each test eliminates hypotheses y
- How should we test to eliminate all incorrect hypotheses?
- The value of a test depends on previous observations!

States y

Problem Statement

Given:

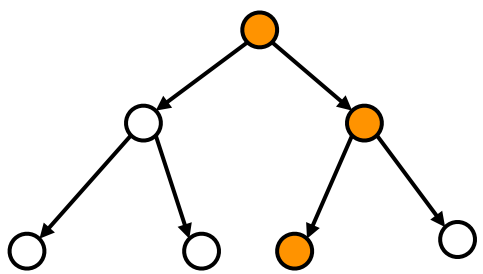
- Items (tests, experiments, actions, …) V={1,…,n}
- Associated with random variables $X_1,\ldots,X_n$ taking values in O
- Objective: $f : 2^V \times O^V \rightarrow \mathbb{R}$

Value of policy π: $F(\pi) = \sum_{\mathbf{x}_V} P(\mathbf{x}_V) f(\pi(\mathbf{x}_V), \mathbf{x}_V)$

Items chosen by π if world in state x_V

Want $\pi^* \in \operatorname{argmax}_{|\pi| \leq k} F(\pi)$

NP-hard (also hard to approximate!)



Adaptive greedy algorithm

- Suppose we've seen $X_A = x_A$.
- Conditional expected benefit of adding item s:

$$\Delta(s \mid \mathbf{x}_A) = \mathbb{E}\Big[f(A \cup \{s\}, \mathbf{x}_V) - f(A, \mathbf{x}_V) \mid \mathbf{x}_A\Big]$$

Benefit if world in state $\mathbf{x}_V$

Adaptive Greedy algorithm:

Start with $A = \emptyset$

For i = 1:k

- Pick $s_k \in \operatorname{argmax}_s \Delta(s \mid \mathbf{x}_A)$
- Observe $X_{s_k} = x_{s_k}$
- Set $A \leftarrow A \cup \{s_k\}$

When does this adaptive greedy algorithm work??

Adaptive submodularity

f is *adaptive submodular*, iff

$$\Delta(s \mid \mathbf{x}_A) \geq \Delta(s \mid \mathbf{x}_B) \quad \text{whenever} \quad \mathbf{x}_A \preceq \mathbf{x}_B$$

$\mathbf{x}_B$ observes more than $\mathbf{x}_A$

f is *adaptive monotonic* iff

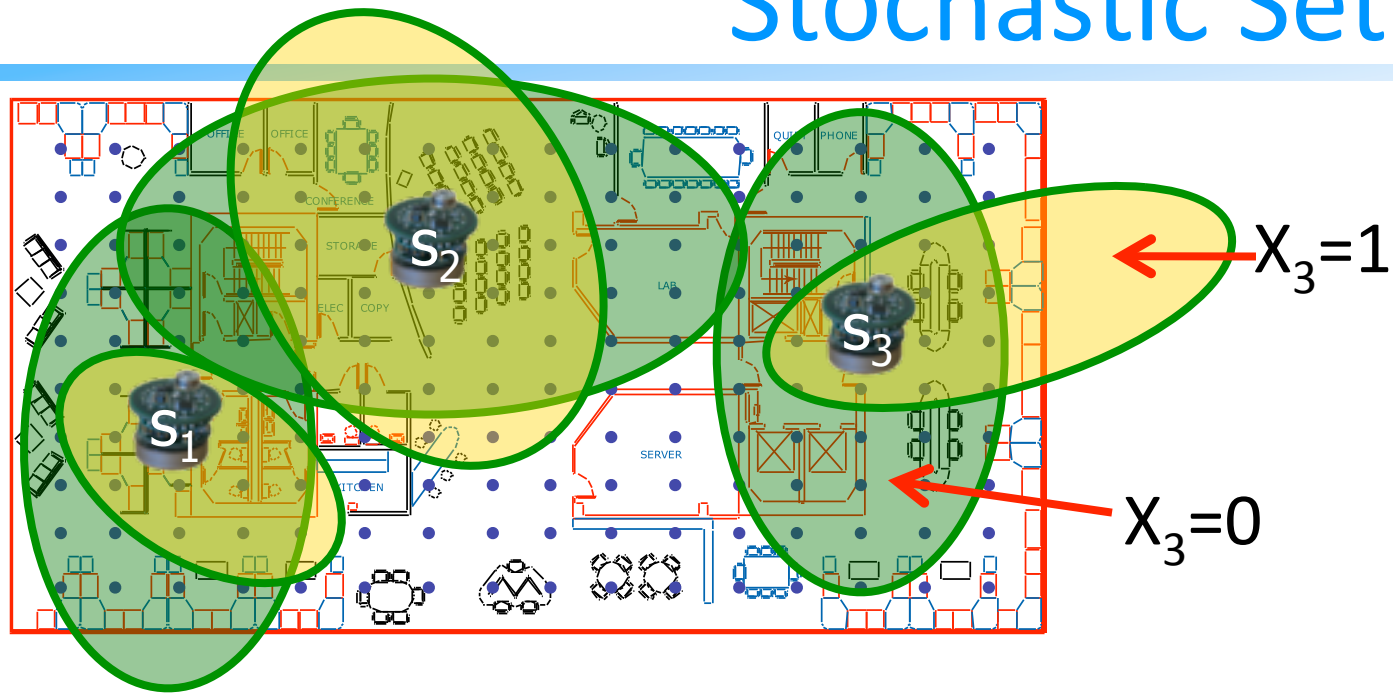
$$\Delta(s \mid \mathbf{x}_A) \geq 0$$

Theorem: If f is adaptive submodular and adaptive monotone w.r.t. to distribution P, then

$F(\pi_{greedy}) \geq (1\text{-}1/e)\ F(\pi_{opt})$

Many other results about submodular set functions can also be "lifted" to the adaptive setting!

Stochastic Set Cover

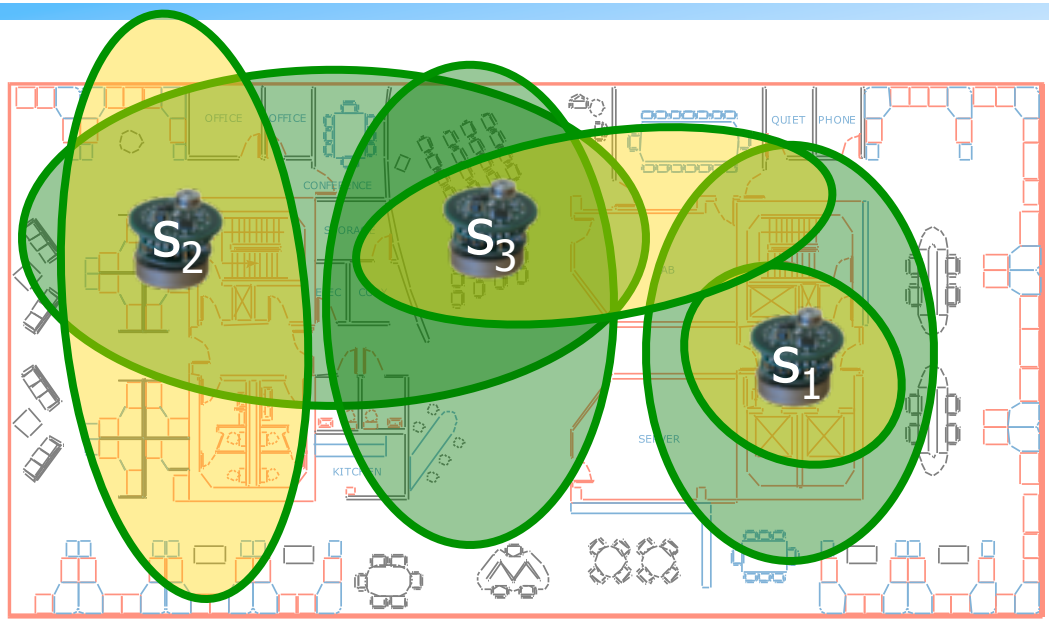


- Set V of possible locations
- Independent random variables $\{X_v : v \text{ in } V\}$
- Each set W_v depends on X_v, revealed after selection

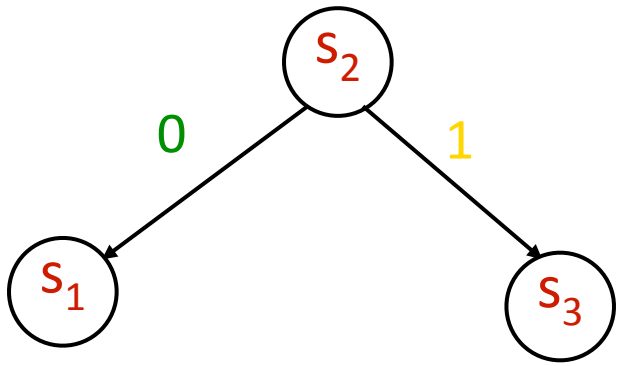
  $f(A, \mathbf{X_V})$ = Area covered by $\{W_v : v \text{ in } A\}$

  Can measure area using *any* mon. submodular function

Results for Stochastic Set Cover



Greedy policy

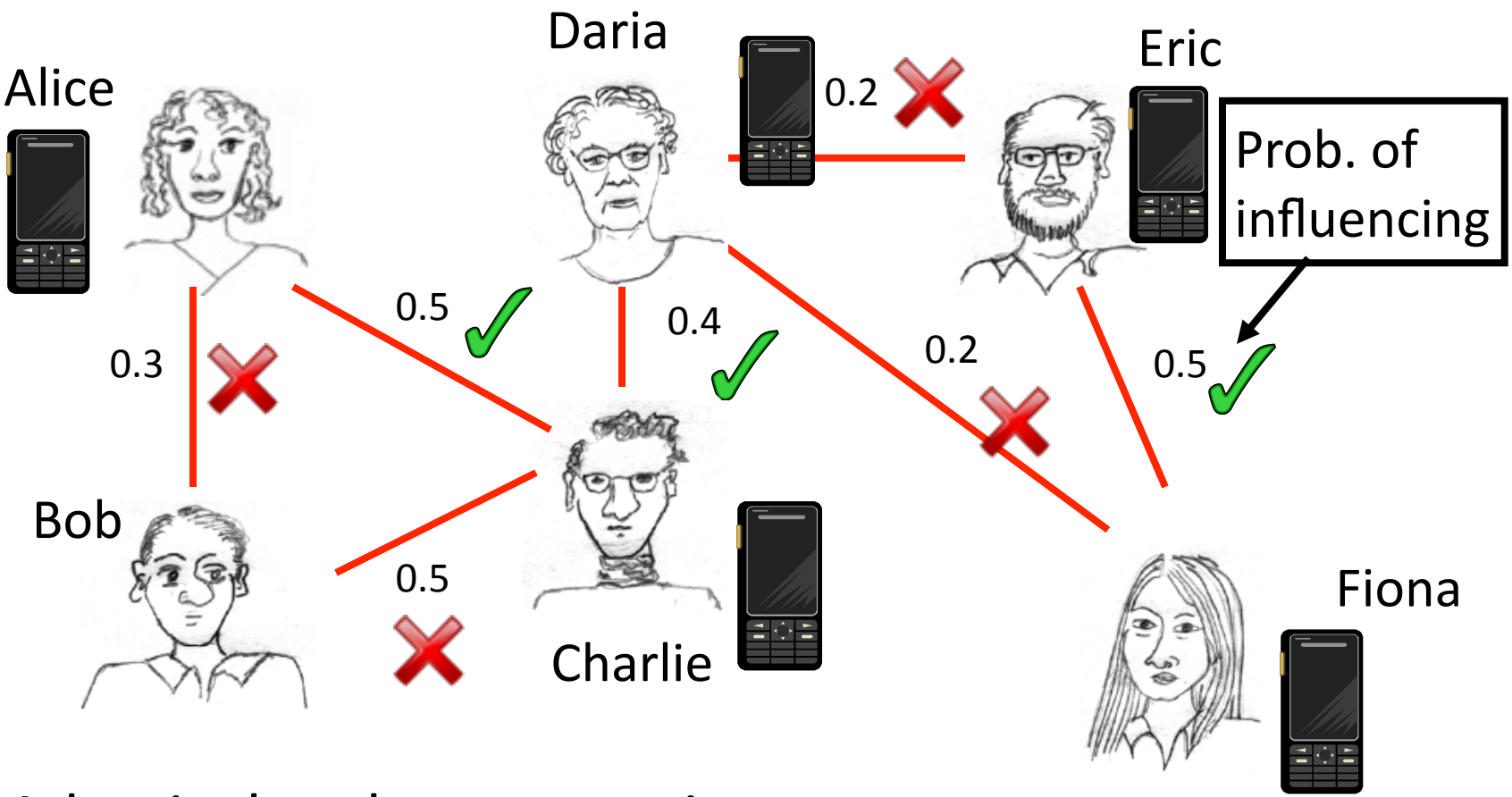


Theorem:

Maximization: Use adapt. greedy alg. to pick k sets

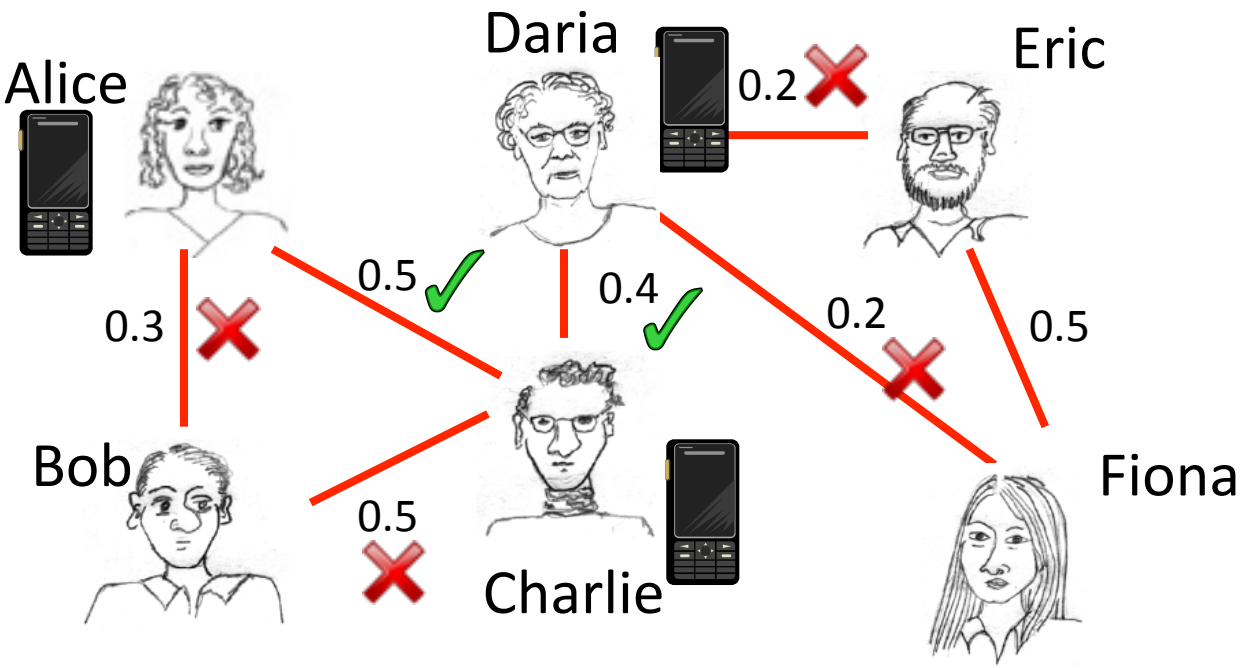
$$F(\pi_{\text{Greedy}})) \geq (1 - 1/e)F(\pi^*)$$

Example: Adaptive Viral Marketing



- Adaptively select promotion targets
- After *each* selection, see which people are influenced.

Application: Adaptive Viral Marketing

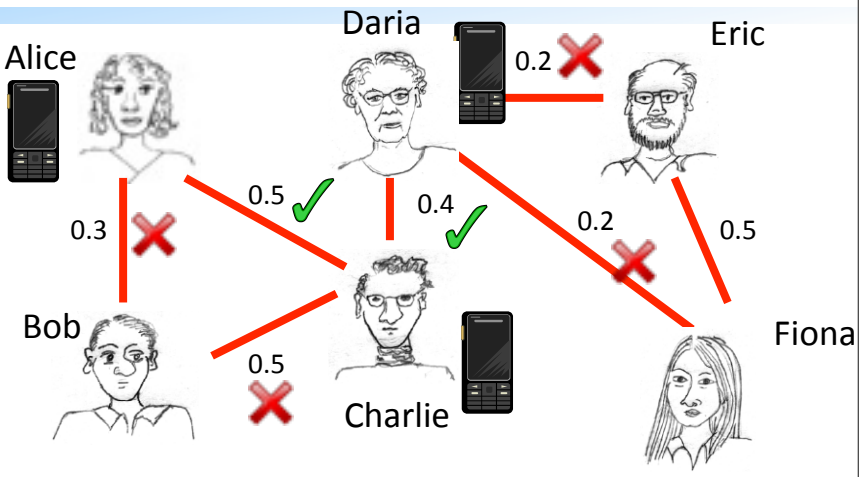


Theorem: Influence is adaptive submodular!

Hence, adaptive greedy choice is a (1-1/e) ≈ 63% approximation to the optimal adaptive solution.

Stochastic Minimum Cost Cover

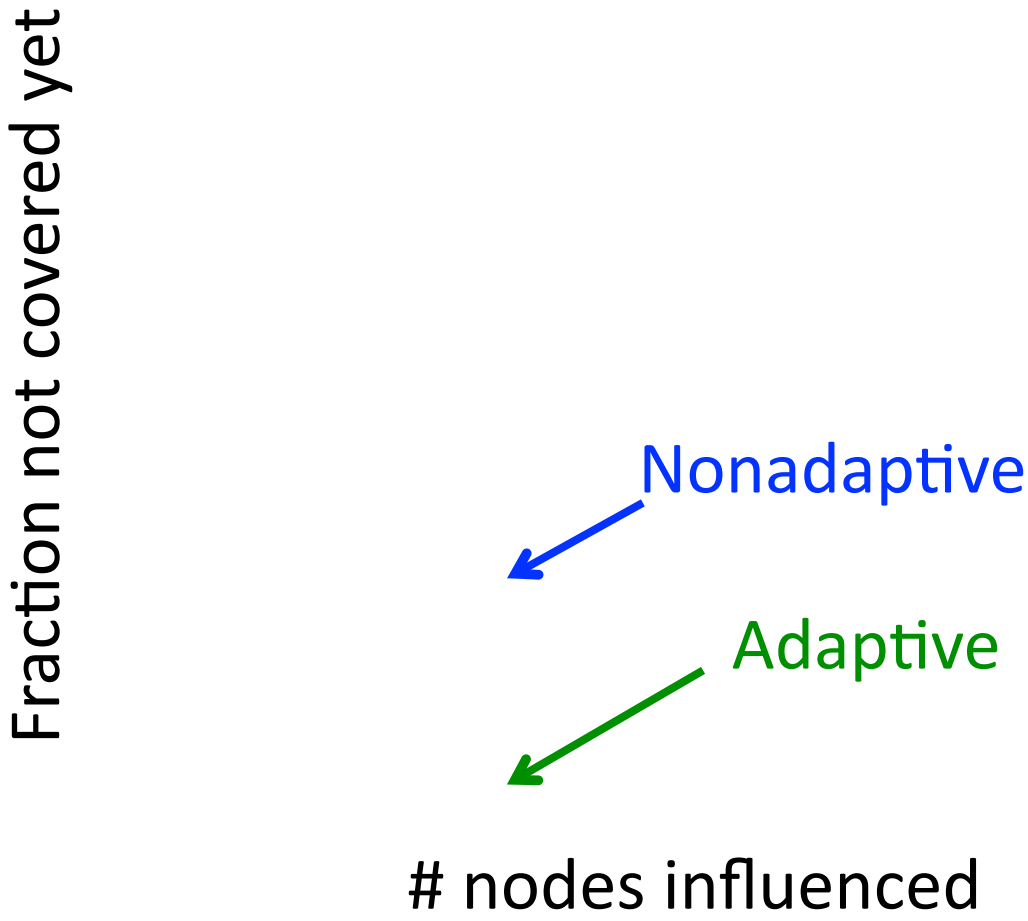
Suppose we wish to
get 25% market penetration
at minimum cost



Theorem [Golovin & Krause, '11]
If F is adaptive submodular and monotone:

$$\text{Cost}(\pi_{\text{greedy}}) \leq O(\log n)\ \text{Cost}(\pi_{\text{opt}})$$

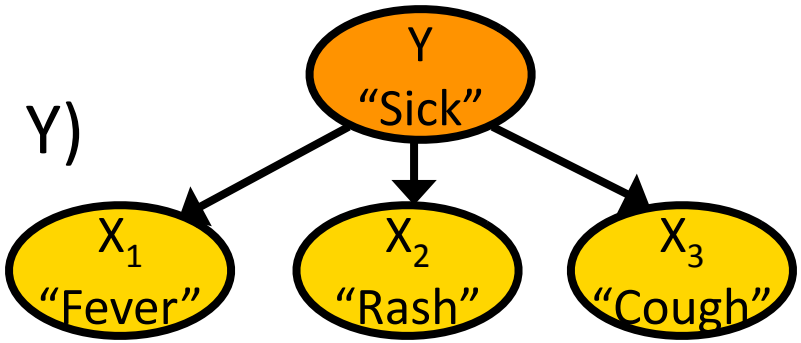
Simulation Results



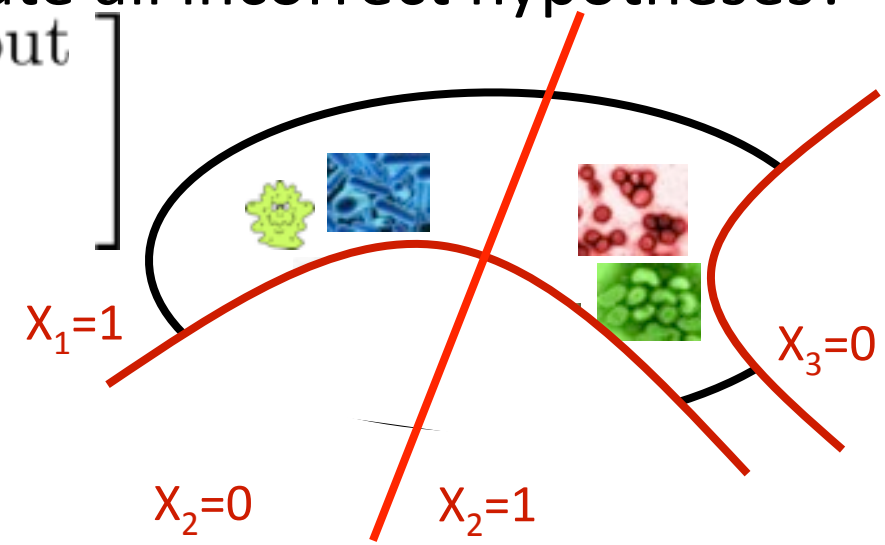
Significant savings through adaptive selection

Optimal Decision Trees

- Prior over diseases P(Y)
- Likelihood of test outcomes P($\mathbf{X_V}$ | Y)
- **Suppose that P($\mathbf{X_V}$ | Y) is deterministic (noise free)**
- Each test eliminates hypotheses y
- How should we test to eliminate all incorrect hypotheses?



$$\Delta(t \mid x_A) = \mathbb{E}\begin{bmatrix}\text{mass ruled out}\\ \text{by } t \text{ if we}\\ \text{know } x_A\end{bmatrix}$$



ODT is Adaptive Submodular

$b_0 := \mathbb{P}($  $)$

$g_0 := \mathbb{P}($  $)$

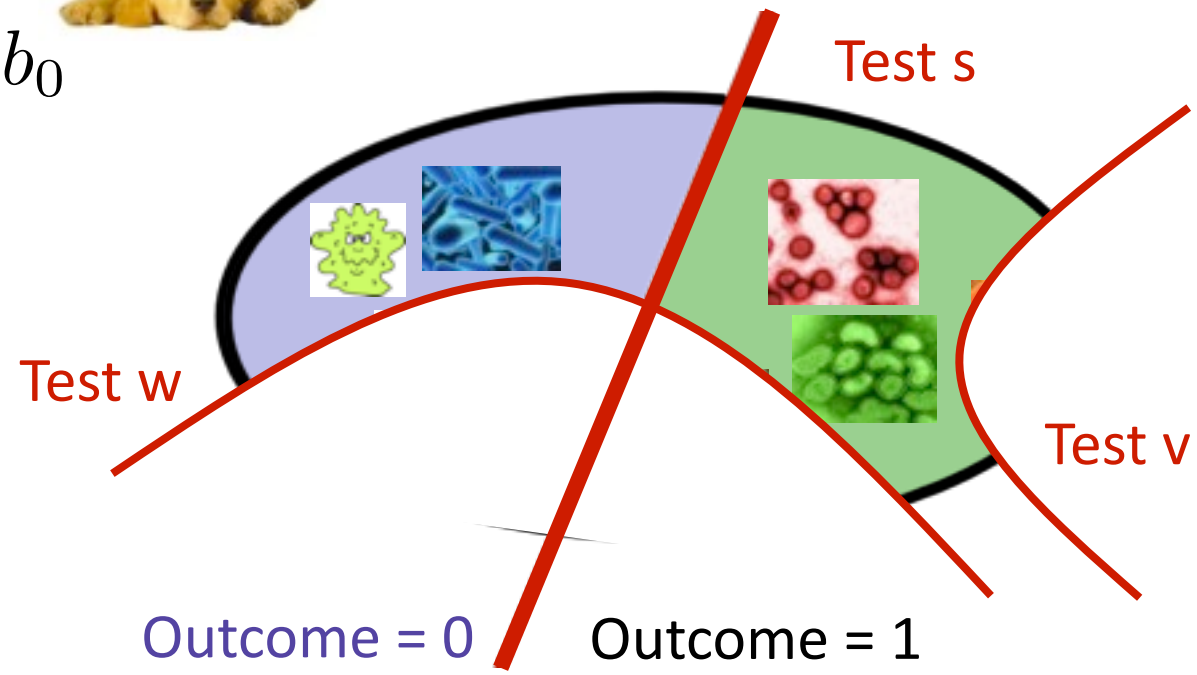
$\Delta(s \mid \{\}) = 2g_0b_0$

Objective = probability mass of hypotheses you have ruled out.

$b_1 := \mathbb{P}($  $)$

$g_1 := \mathbb{P}($  $)$

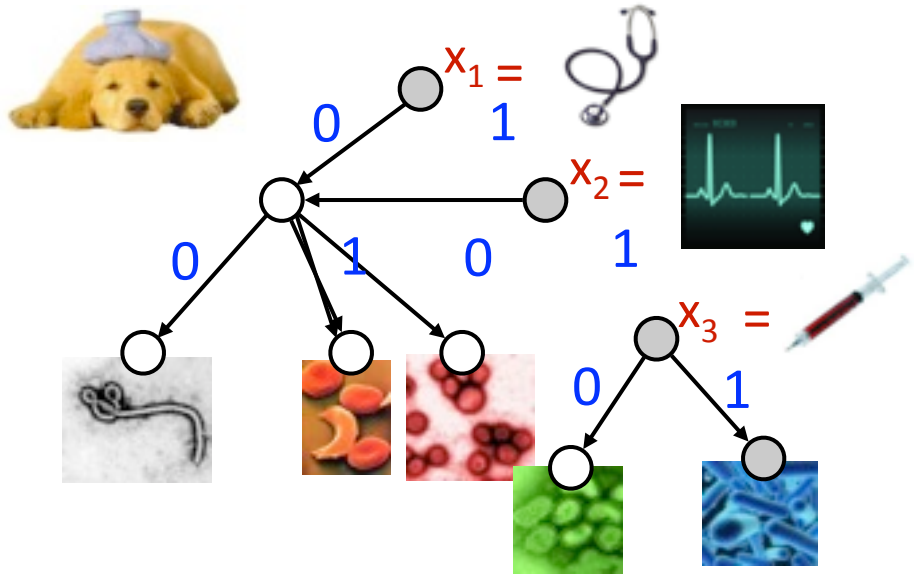
$\Delta(s \mid \mathbf{x}_{v,w}) = 2g_1b_1$



$b_0 \geq b_1, \quad g_0 \geq g_1$

Not hard to show that $\Delta(s \mid \{\}) \geq \Delta(s \mid \mathbf{x}_{v,w})$

Guarantees for Optimal Decision Trees



Garey & Graham, 1974;
Loveland, 1985;
Arkin et al., 1993;
Kosaraju et al., 1999;
Dasgupta, 2004;
Guillory & Bilmes, 2009;
Nowak, 2009;
Gupta et al., 2010

Adaptive-Greedy is a $(\ln(1/p_{\min}) + 1)$ approximation.

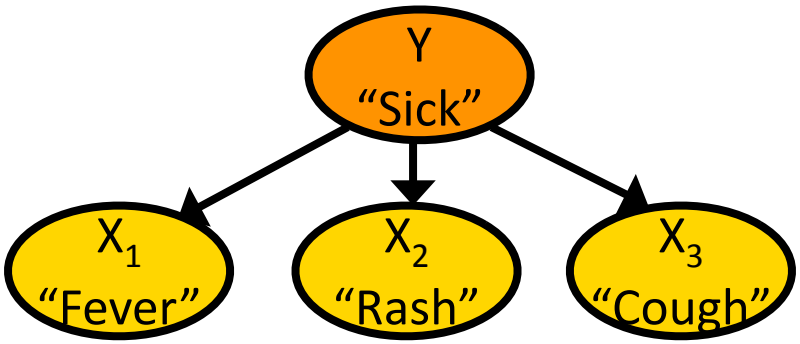
With adaptive submodular analysis!

Result requires that tests are *exact* (no noise)!

Noisy Bayesian diagnosis

In practice, observations typically are noisy

Results for noise-free case do not generalize ☹



Key problem: Tests no longer rule out hypotheses (only make them less likely)

Intuitively, want to gather *enough information* to make the *right decision*!

Noisy Bayesian diagnosis

Suppose I run all tests, i.e., see $\mathbf{x}_V$

Best I can do is to maximize expected utility

$$a^* = \arg\max_a \sum_y P(y \mid \mathbf{x}_V) U(a, y)$$

How should I cheaply test to guarantee that I choose a*?

Existing approaches:

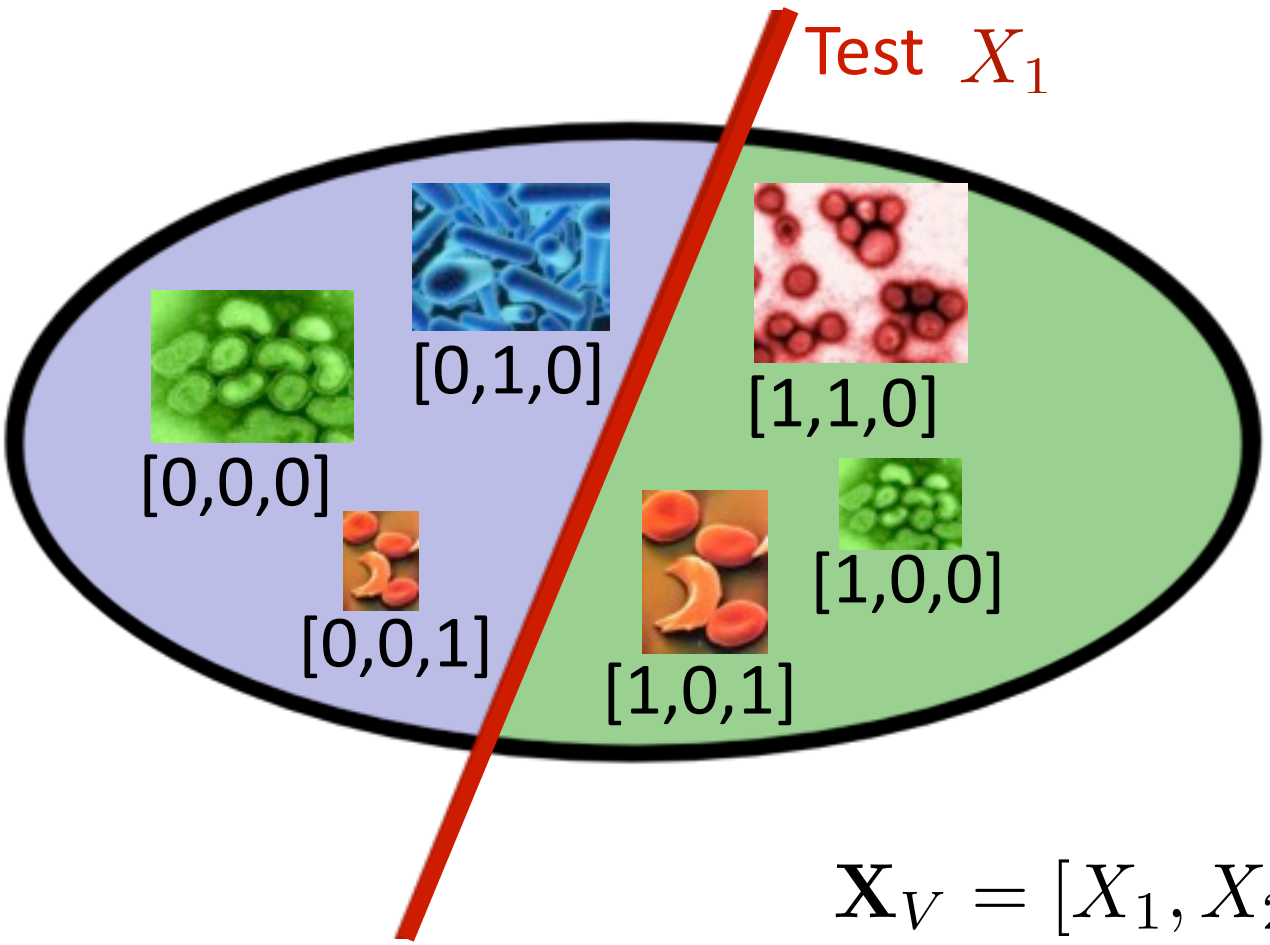
- Generalized binary search?
- Maximize information gain?
- Maximize value of information?

Not adaptive submodular in the noisy setting!

Theorem: All these approaches can have cost

A new criterion for nonmyopic VOI

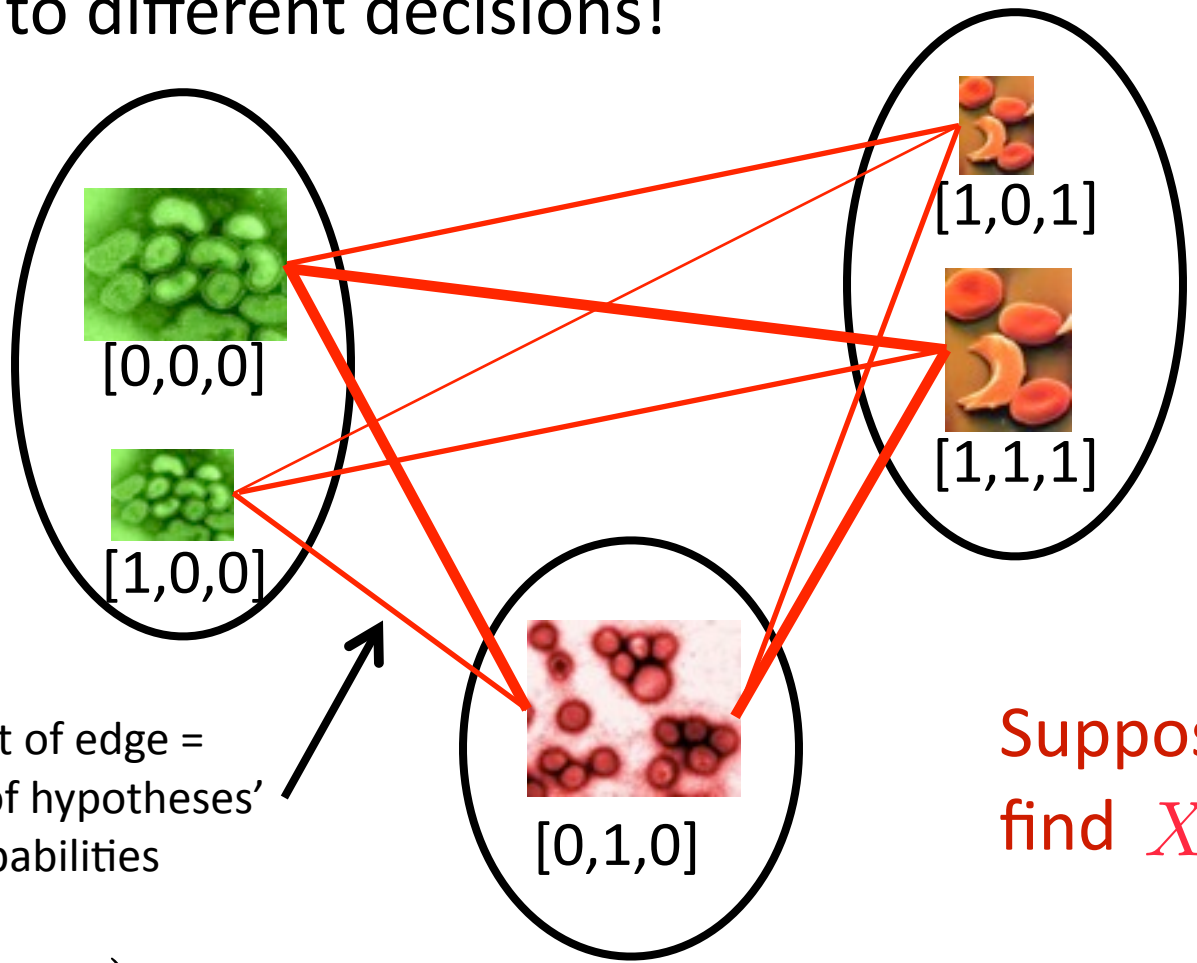
- **Strategy**: Replace noisy problem with noiseless problem
- **Key idea**: Make test outcomes part of the hypothesis



$$\mathbf{X}_V = [X_1, X_2, X_3]$$

A new criterion for nonmyopic VOI

Only need to distinguish between noisy hypotheses that lead to different decisions!



Tests

$[X_1, X_2, X_3]$

Weight of edge = product of hypotheses' probabilities

Suppose we find $X_1 = 1$

$f(A, \mathbf{x}_V)$ = total mass of edges cut if observing $\mathbf{X}_A = \mathbf{x}_A$

Theoretical guarantees

Theorem: Equivalence class edge-cutting (EC^2) is adaptive submodular and adaptive monotone.

Suppose $P(\mathbf{x}_V, h) \in \{0\} \cup [\delta, 1]$ for all $\mathbf{x}_V, h$

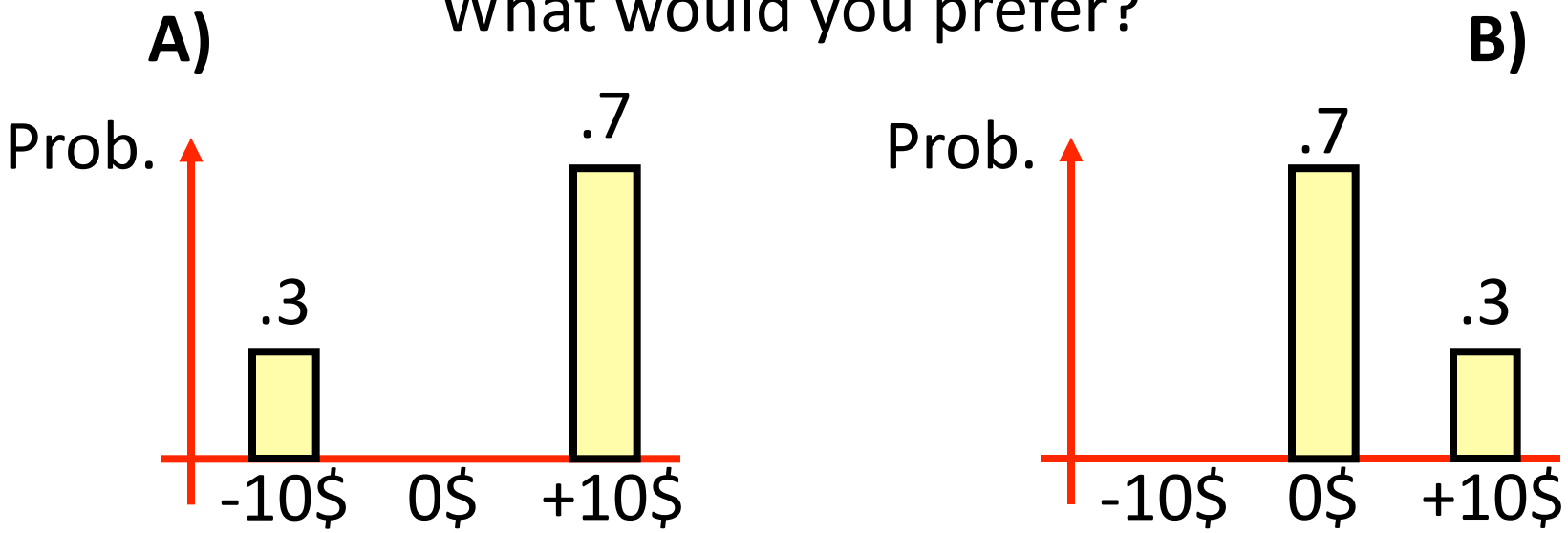
Then it holds that

$$\mathrm{Cost}(\pi_{\mathrm{Greedy}}) \leq \mathcal{O}\left(\log^{1}\right.$$

First approximation guarantees for **nonmyopic VOI** in general graphical models!

Example: The Iowa Gambling Task

What would you prefer?



Various competing theories on how people make decisions under uncertainty

- Maximize expected utility? [von Neumann & Morgenstern '47]
- Constant relative risk aversion? [Pratt '64]
- Portfolio optimization? [Hanoch & Levy '70]
- (Normalized) Prospect theory? [Kahnemann & Tversky '79]

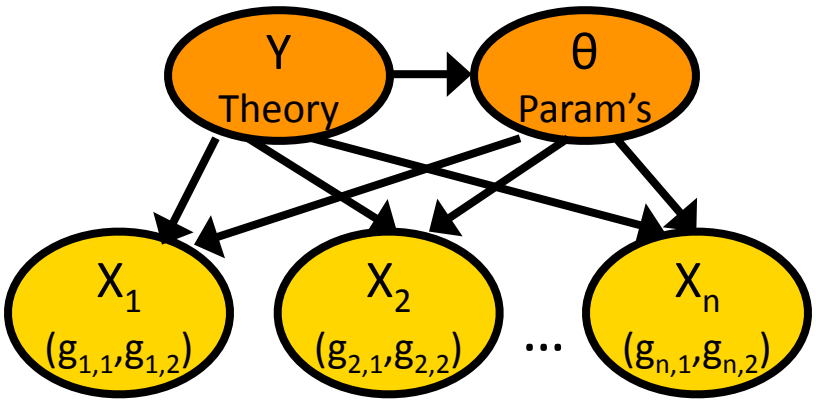
How should we design tests to distinguish theories?

Iowa Gambling as BED

Every possible test $X_s = (g_{s,1}, g_{s,2})$ is a pair of gambles

Theories parameterized by θ

Each theory predicts utility for every gamble U(g,y,θ)



$$P(X_s = 1 \mid y, \theta) = \frac{1}{}$$

Simulation Results

Adaptive submodular criterion (EC^2) outperforms existing approaches

Experimental Study

Study with 57 naïve subjects

32,000 designs in <5s; 8x faster through lazy!

- Indication of heterogeneity in population
- Unexpectedly **no support** for CRRA
- Submodularity crucial for real-time performance!

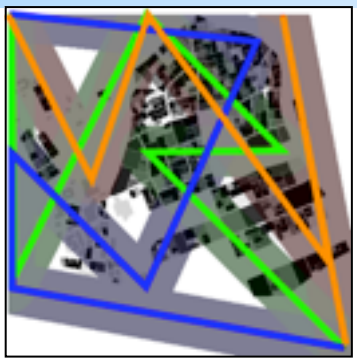
Interactive submodular coverage

- Alternative formalization of adaptive optimization [Guillory & Bilmes, ICML '10]
  - Addresses the worst case setting
- Applications to (noisy) active learning, viral marketing [Guillory & Bilmes, ICML '11]

Submodularity in ML / AI

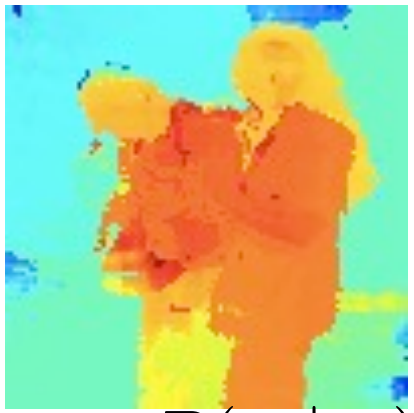
$$\mathbf{x} = \mathbf{\Phi w} + \mathbf{n}$$

Submodularity & Sparsity
[K & Cevher '10, Bach '10, …]



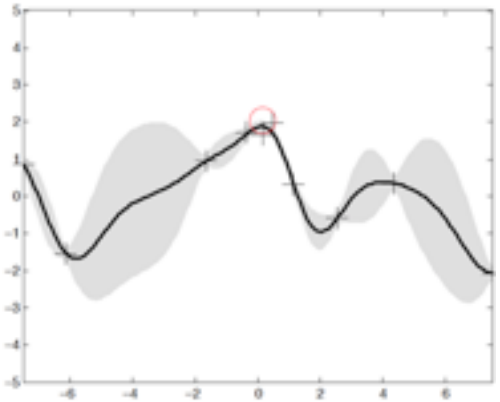
$$\max_P \min_i \mathbb{E}_{\mathcal{A}\sim P} F_i(\mathcal{A})$$

Zero-sum submodular games
[K, Golovin, Roper '11]



$$\max_{\mathbf{x}} P(\mathbf{x} \mid \mathbf{y})$$

Inference for MAP
[Kolmogorov & Zabih '03, …]



$$R_T = \mathcal{O}^*\left(\sqrt{}\right.$$

First convergence rates
GP optimization [Srinivas et al'10]

- Matlab Toolbox for Submodular Function Optimization
- NIPS Workshops on Discrete Optimization in ML (DISCML) (talks recorded on videolectures.net)