Active Learning and Optimized Information Gathering

Lecture 12 – Submodularity

CS 101.2
Andreas Krause

Announcements

- **Homework 2: Due Thursday Feb 19**
- **Project milestone due: Feb 24**
  - 4 Pages, NIPS format: http://nips.cc/PaperInformation/StyleFiles
  - Should contain preliminary results (model, experiments, proofs, …) as well as timeline for remaining work
  - Come to office hours to discuss projects!
- Office hours
  - Come to office hours before your presentation!
  - Andreas: **Monday 3pm-4:30pm**, 260 Jorgensen
  - Ryan: Wednesday 4:00-6:00pm, 109 Moore

Course outline

1. Online decision making
2. Statistical active learning
3. $\boxed{\text{Combinatorial approaches}}$

Medical diagnosis

- Want to predict medical condition of patient given noisy symptoms / tests
  - Body temperature
  - Rash on skin
  - Cough
  - Increased antibodies in blood
  - Abnormal MRI

	healthy	*sick*
Treatment	-$$	$
No treatment	0	-$$$

- Treating a healthy patient is bad,
  not treating a sick patient is terrible
- Each test has a (potentially different) cost
- **Which tests should we perform to make most effective decisions?**

Value of information

Prior $P(Y)$ $\xrightarrow{\text{obs } X_i = x_i}$ **Posterior $P(Y \mid x_i)$** $\Rightarrow$ **Reward**

- Value of information:
  Reward[$P(Y \mid x_i)$] = $\max_a EU(a \mid x_i)$

- Reward can by **any function** of the distribution $P(Y \mid x_i)$
- Important examples:
  - Posterior variance of Y
  - Posterior entropy of Y

Optimal value of information

- Can we **efficiently** optimize value of information?

➔ Answer depends on properties of the distribution $P(X_1,\ldots,X_n,Y)$

Theorem [Krause & Guestrin IJCAI '05]:

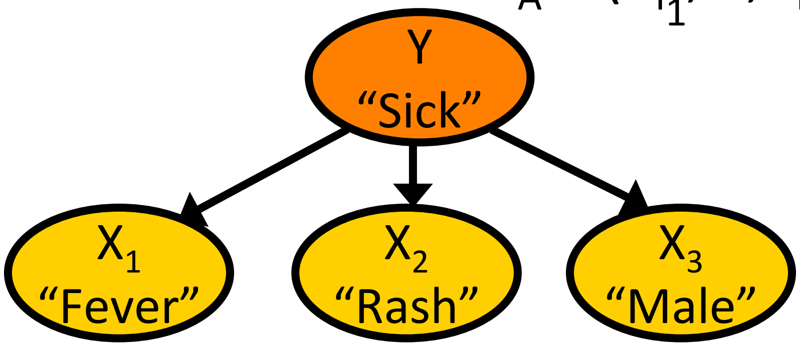
- If the random variables form a Markov Chain, can find optimal (exponentially large!) decision tree in polynomial time ☺
- There exists a class of distributions for which we can perform efficient inference (i.e., compute $P(Y|X_i)$), where finding the optimal decision tree is **NP^{PP} hard**

Approximating value of information?

- If we can’t find an optimal solution, can we find **provably near-optimal** approximations??

Feature selection

- Given random variables Y, X_1, … X_n
- Want to predict Y from subset $X_A = (X_{i_1},…,X_{i_k})$



Naïve Bayes Model

Want k most informative features:

$$\boxed{A^* = \operatorname{argmax} IG(X_A; Y) \text{ s.t. } |A| \leq k}$$

where $IG(X_A; Y) = H(Y) - H(Y \mid X_A)$

Uncertainty before knowing X_A

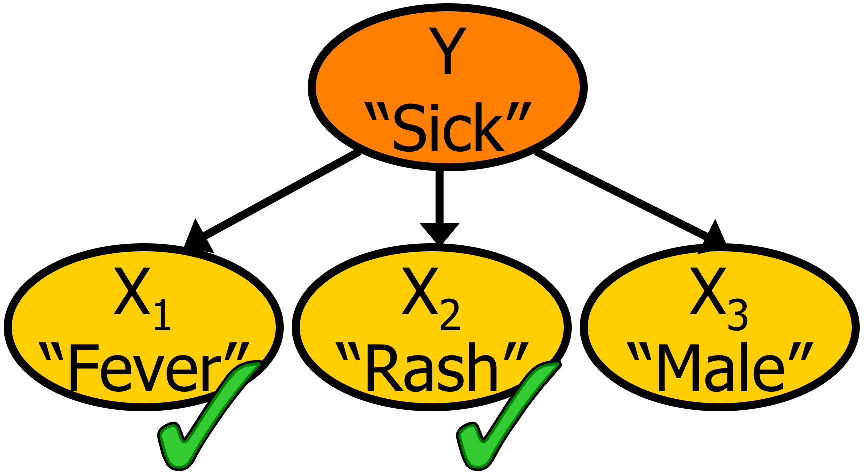
Uncertainty after knowing X_A

Example: Greedy algorithm for feature selection

- Given: finite set V of features, utility function $F(A) = IG(X_A; Y)$
- Want: $A^* \subseteq V$ such that

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

NP-hard!



<u>Greedy algorithm:</u>

Start with $A = \emptyset$

For i = 1 to k

$s^* := \operatorname{argmax}_s F(A \cup \{s\})$

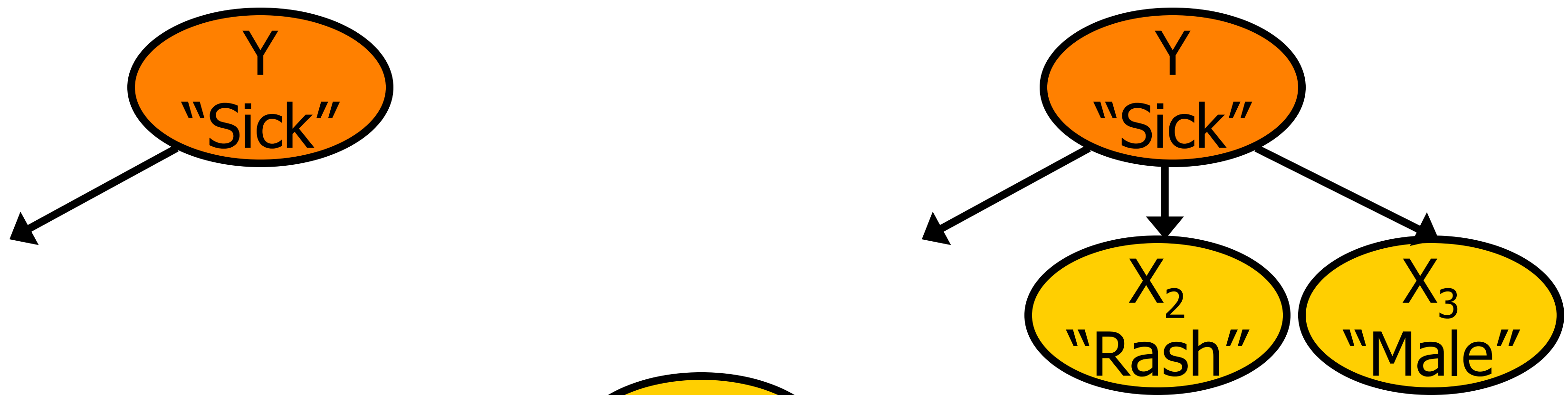
$A := A \cup \{s^*\}$

How well can this simple heuristic do?

Key property: Diminishing returns

Selection A = {}

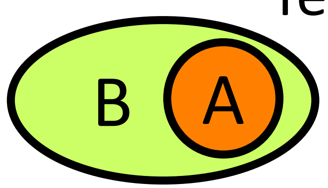
Selection B = $\{X_2, X_3\}$



Theorem [Krause, Guestrin UAI '05]**: Information gain F(A) in Naïve Bayes models is submodular!**

feature X_1

Submodularity:



+ • s   Large improvement

+ • s   Small improvement

For $A \subseteq B$, $F(A \cup \{s\}) - F(A) \geq F(B \cup \{s\}) - F(B)$

Why is submodularity useful?

Theorem [Nemhauser et al '78]

Greedy maximization algorithm returns A_{greedy}:

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

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

Submodularity is an incredibly useful and powerful concept!

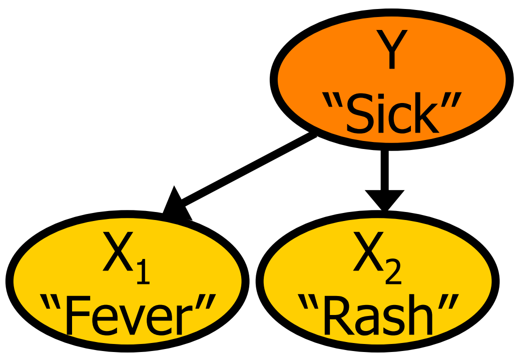
Set functions

- Finite set $V = \{1,2,\ldots,n\}$
- Function $F: 2^V \rightarrow R$
- Will always assume $F(\emptyset) = 0$ (w.l.o.g.)
- Assume black-box that can evaluate F for any input A
  - Approximate (noisy) evaluation of F is ok
- Example: $F(A) = IG(X_A; Y) = H(Y) - H(Y \mid X_A)$

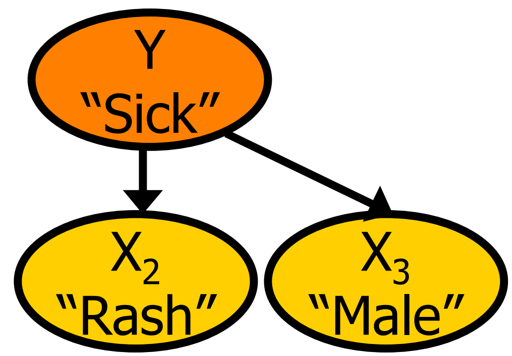
$$= \sum_{y,x_A} P(x_A)\,[\log P(y \mid x_A) - \log P(y)]$$

$A = \{i_1, \ldots i_k\} \subset V$

$X_A = (X_{i_1}, \ldots, X_{i_k})$



$F(\{X_1,X_2\}) = 0.9$

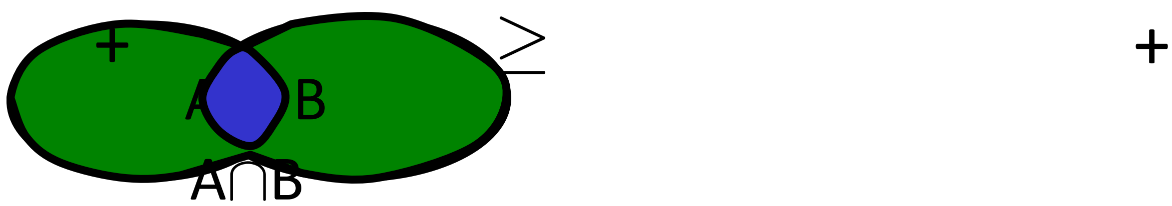


$F(\{X_2,X_3\}) = 0.5$

Submodular set functions

- Set function F on V is called submodular if

  For all $A,B \subseteq V$: $F(A)+F(B) \geq F(A\cup B)+F(A\cap B)$



- Equivalent diminishing returns characterization:

For $A\subseteq B$, $s\notin B$, $F(A \cup \{s\}) - F(A) \geq F(B \cup \{s\}) - F(B)$

Submodularity and supermodularity

- Set function F on V is called submodular if

  1) For all $A,B \subseteq V$: $F(A)+F(B) \geq F(A \cup B)+F(A \cap B)$

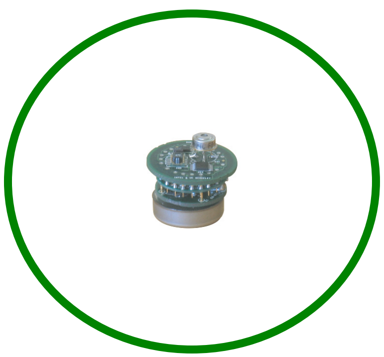
  $\Leftrightarrow$ 2) For all $A \subseteq B$, $s \notin B$, $F(A \cup \{s\}) - F(A) \geq F(B \cup \{s\}) - F(B)$

- F is called supermodular if $-F$ is submodular
- F is called modular if F is both sub- and supermodular

  for modular ("additive") F, $F(A) = \sum_{i \in A} w(i)$

Example: Set cover

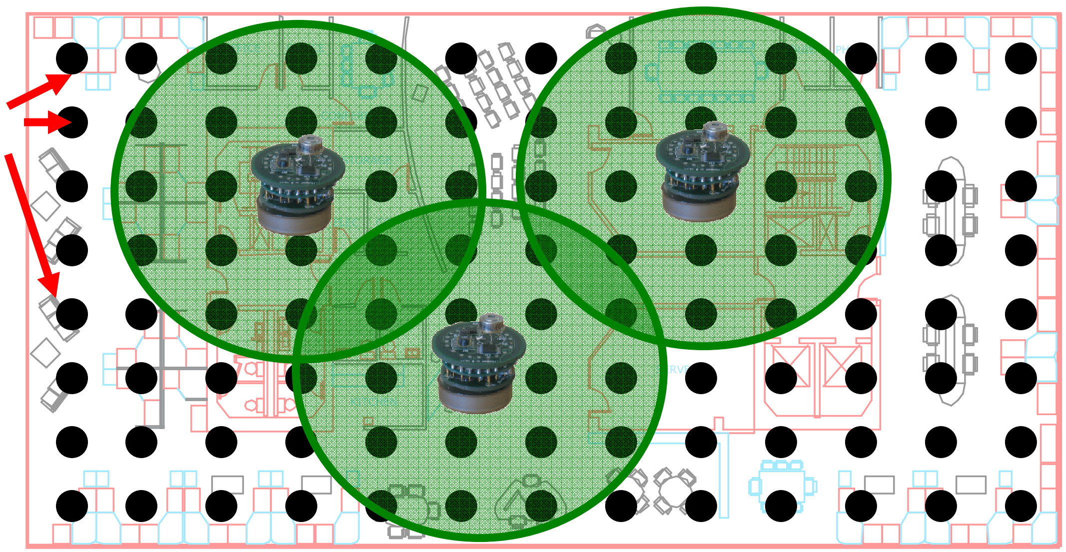
Place sensors in building



Node predicts values of positions with some radius

Want to cover floorplan with discs

Possible locations V



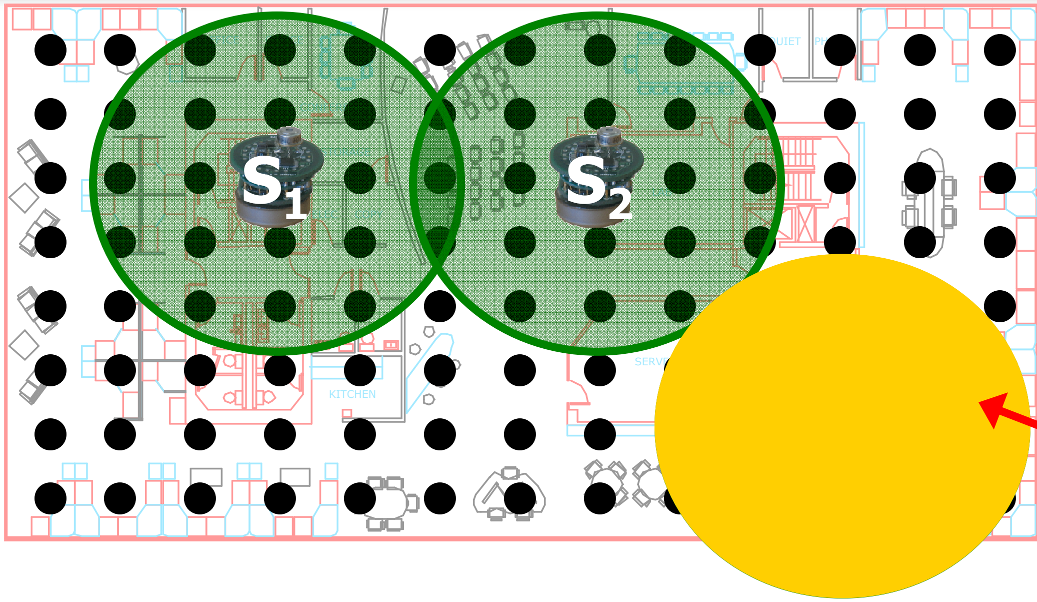
For $A \subseteq V$: $F(A)$ = "area covered by sensors placed at A"

Formally:

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

For $A \subseteq V=\{1,\ldots,n\}$ define $F(A) = |\bigcup_{i\in A} S_i|$

Set cover is submodular

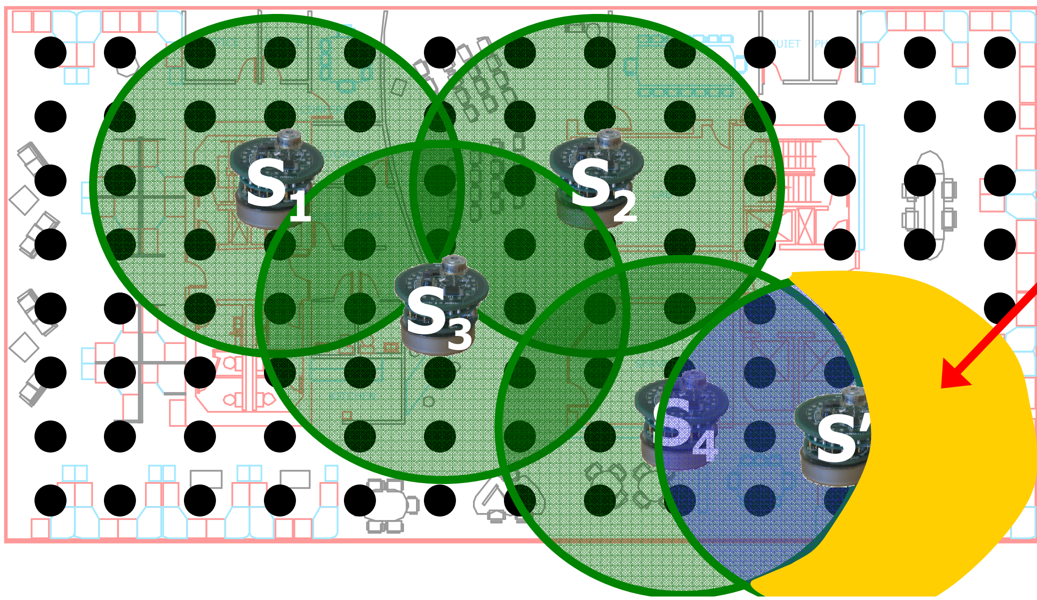


A={S_1,S_2}

F(A∪{S'})-F(A)

$\geq$

F(B∪{S'})-F(B)



B = {S_1,S_2,S_3,S_4}

Example: Mutual information

- Given random variables $X_1,\ldots,X_n$
- $F(A) = I(X_A; X_{V\setminus A}) = H(X_{V\setminus A}) - H(X_{V\setminus A} \mid X_A)$

Lemma: Mutual information F(A) is submodular

$$F(A \cup \{s\}) - F(A) = \underbrace{H(X_s \mid X_A)} - H(X_s \mid X_{V\setminus(A\cup\{s\})})$$

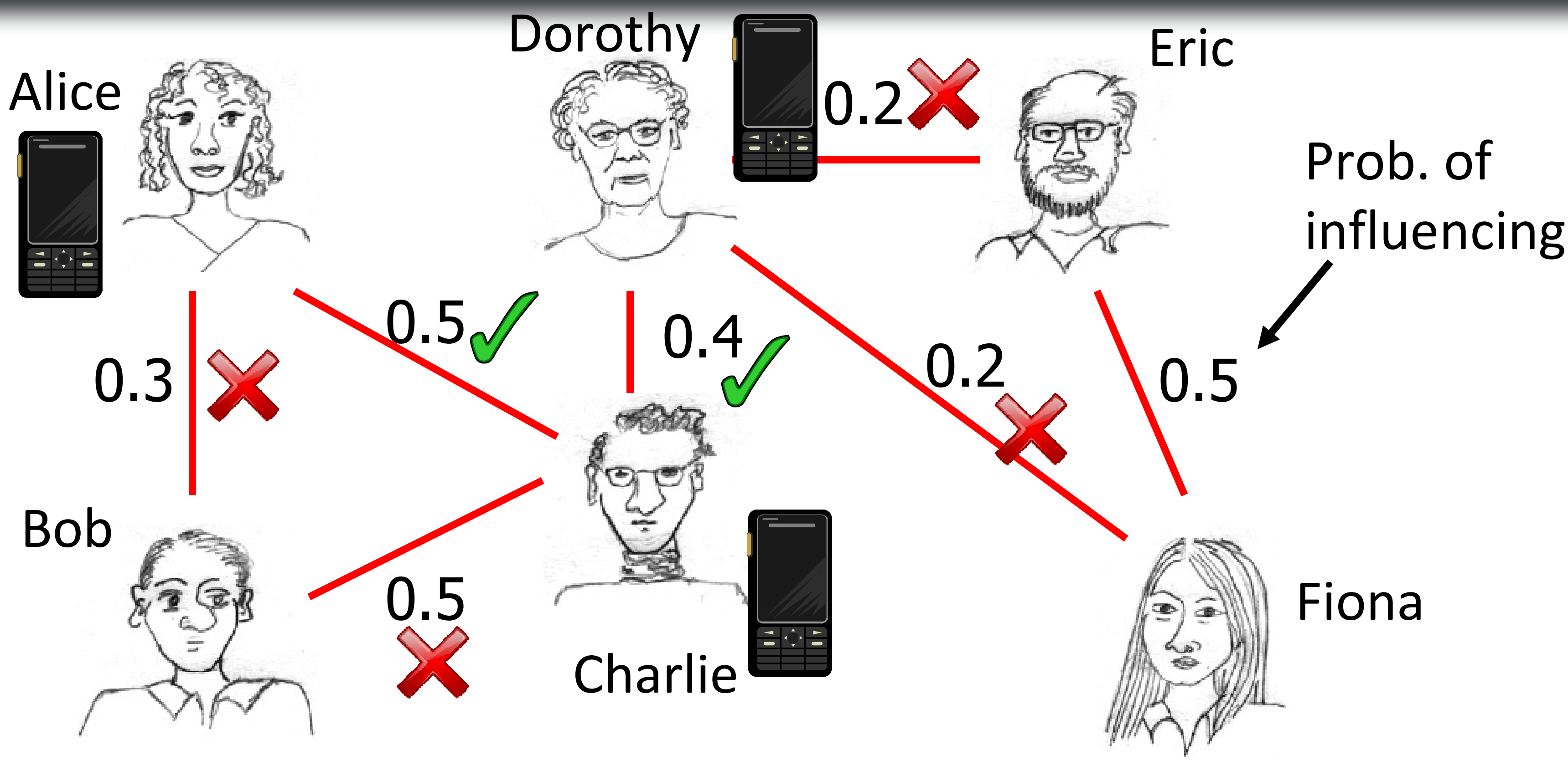
$A \subset B: H(X_s \mid X_A) \geq H(X_s \mid X_B)$

"information never hurts"

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

$\Leftrightarrow$ F submodular ☺

Example: Influence in social networks
[Kempe, Kleinberg, Tardos KDD '03]

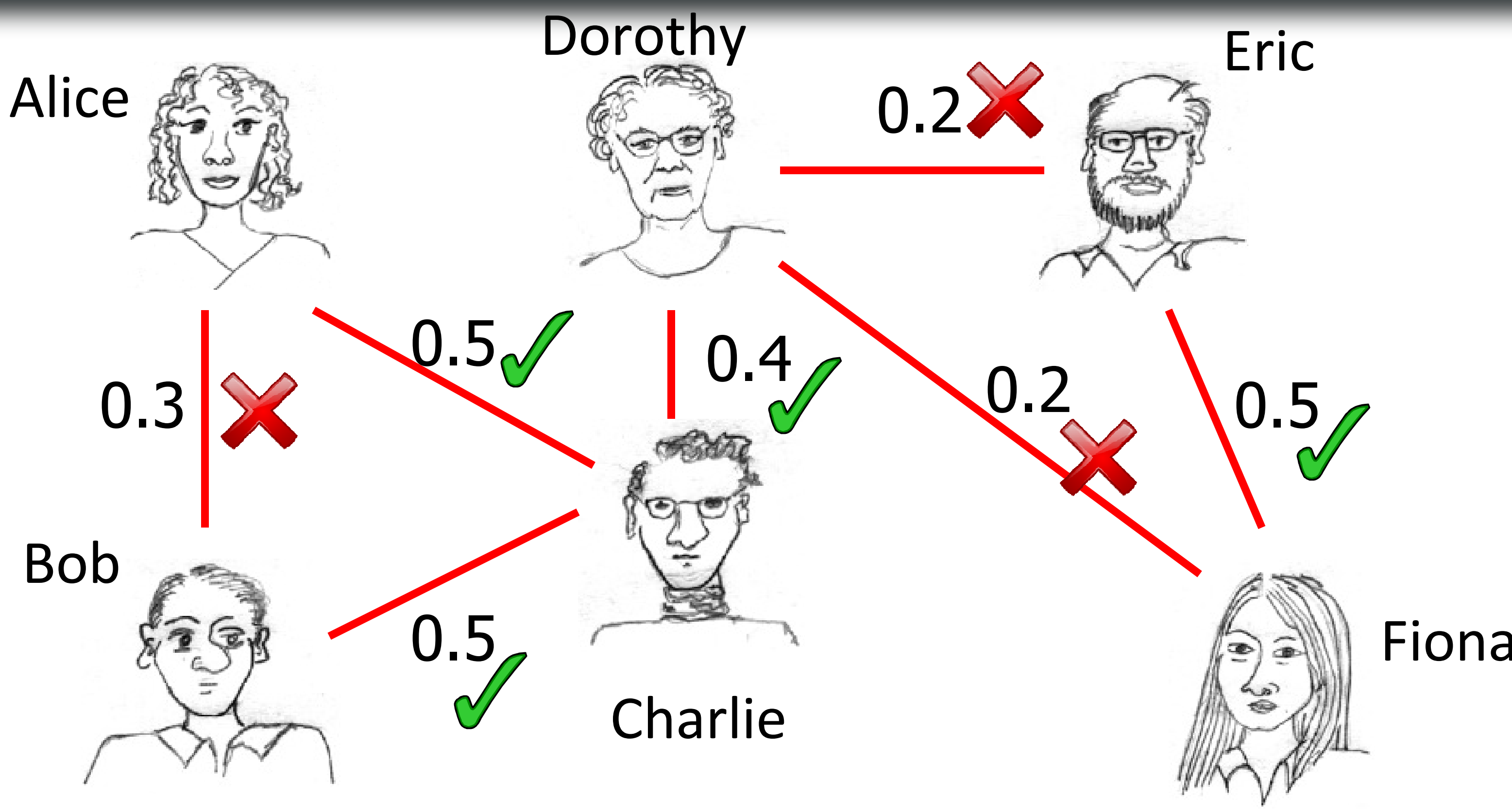


Who should get free cell phones?

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

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

Influence in social networks is submodular
[Kempe, Kleinberg, Tardos KDD '03]



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

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

$F(A) = \sum_c P(c)\ F_c(A)$

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

Closedness properties

$F_1,\ldots,F_m$ submodular functions on V and $\lambda_1,\ldots,\lambda_m > 0$

Then: $F(A) = \sum_i \lambda_i F_i(A)$ is submodular!

Submodularity closed under nonnegative linear combinations!

Extremely useful fact!!

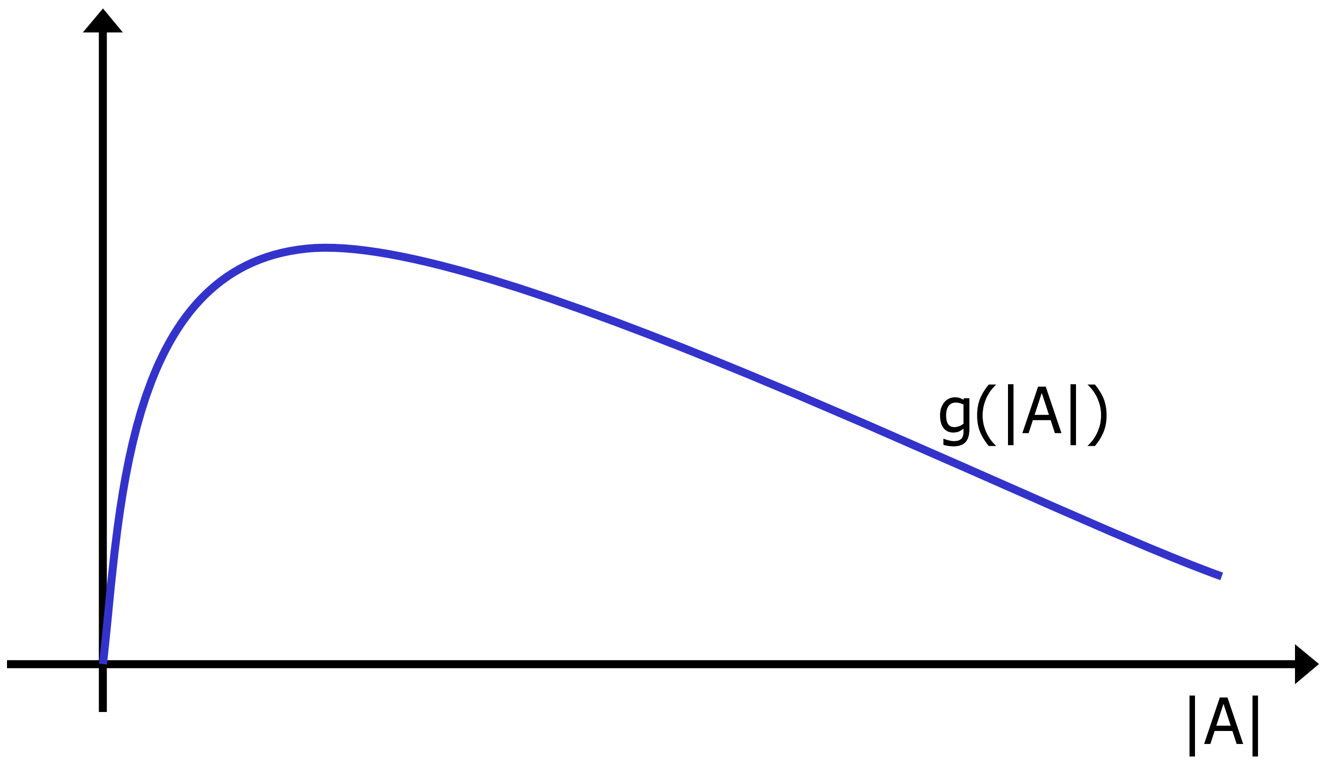
- $F_\theta(A)$ submodular $\Rightarrow \sum_\theta P(\theta) F_\theta(A)$ submodular!
- Multicriterion optimization:
  $F_1,\ldots,F_m$ submodular, $\lambda_i \geq 0 \Rightarrow \sum_i \lambda_i F_i(A)$ submodular

Submodularity and Concavity

Suppose g: N $\rightarrow$ R and F(A) = g(|A|)

Then F(A) submodular if and only if g concave!

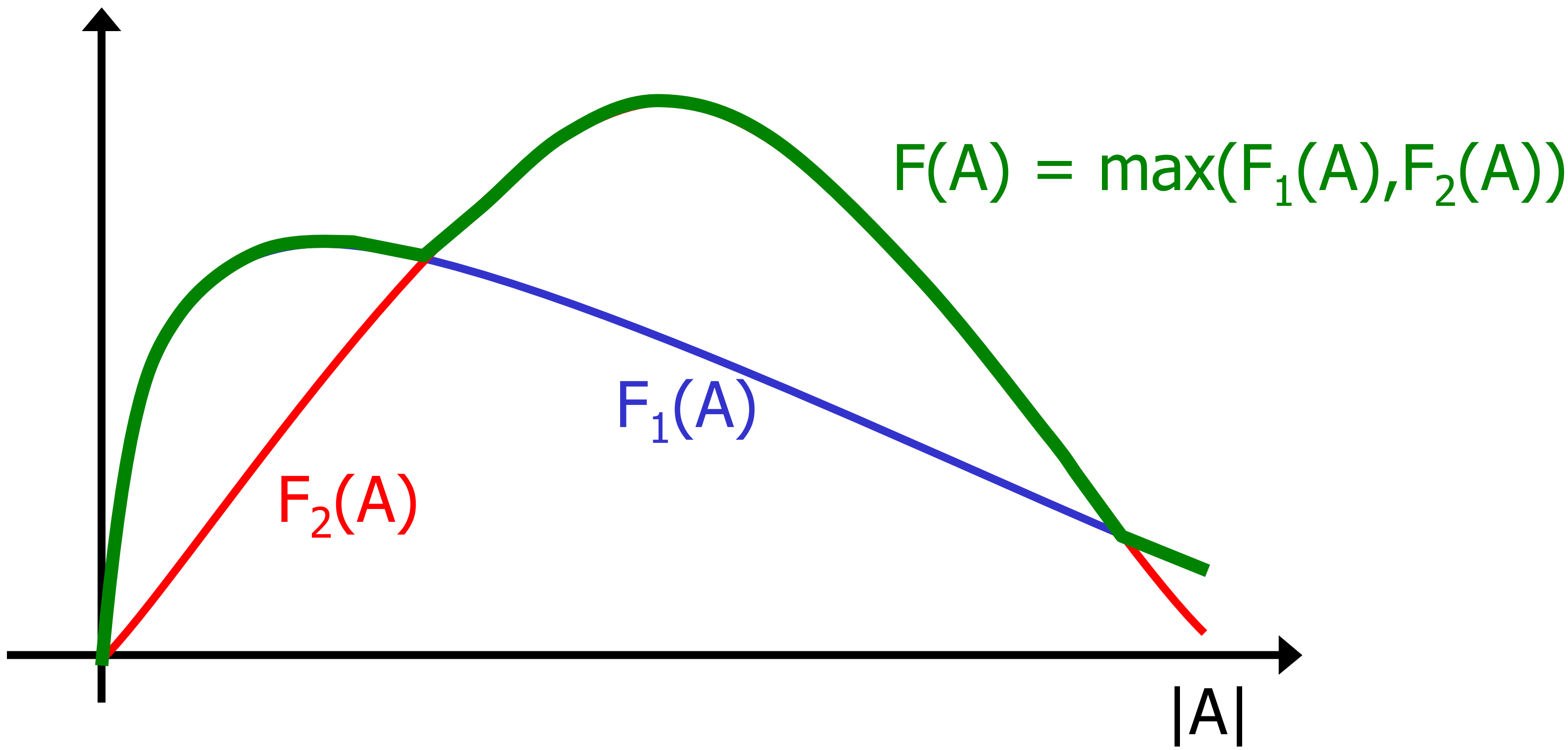
E.g., g could say "buying in bulk is cheaper"



Maximum of submodular functions

Suppose $F_1(A)$ and $F_2(A)$ submodular.

Is $F(A) = \max(F_1(A), F_2(A))$ submodular?



max(F_1,F_2) not submodular in general!

Minimum of submodular functions

Well, maybe $F(A) = \min(F_1(A),F_2(A))$ instead?

	$F_1(A)$	$F_2(A)$
$\emptyset$	0	0
{a}	1	0
{b}	0	1
{a,b}	1	1

$$F(\{b\}) - F(\emptyset)=0$$
$$<$$
$$F(\{a,b\}) - F(\{a\})=1$$

$\min(F_1,F_2)$ not submodular in general!

But stay tuned – we'll address $\min_i F_i$ later!

Maximizing submodular functions

Minimizing convex functions:

Polynomial time solvable!

Minimizing submodular functions:

Polynomial time solvable!

Maximizing convex functions:

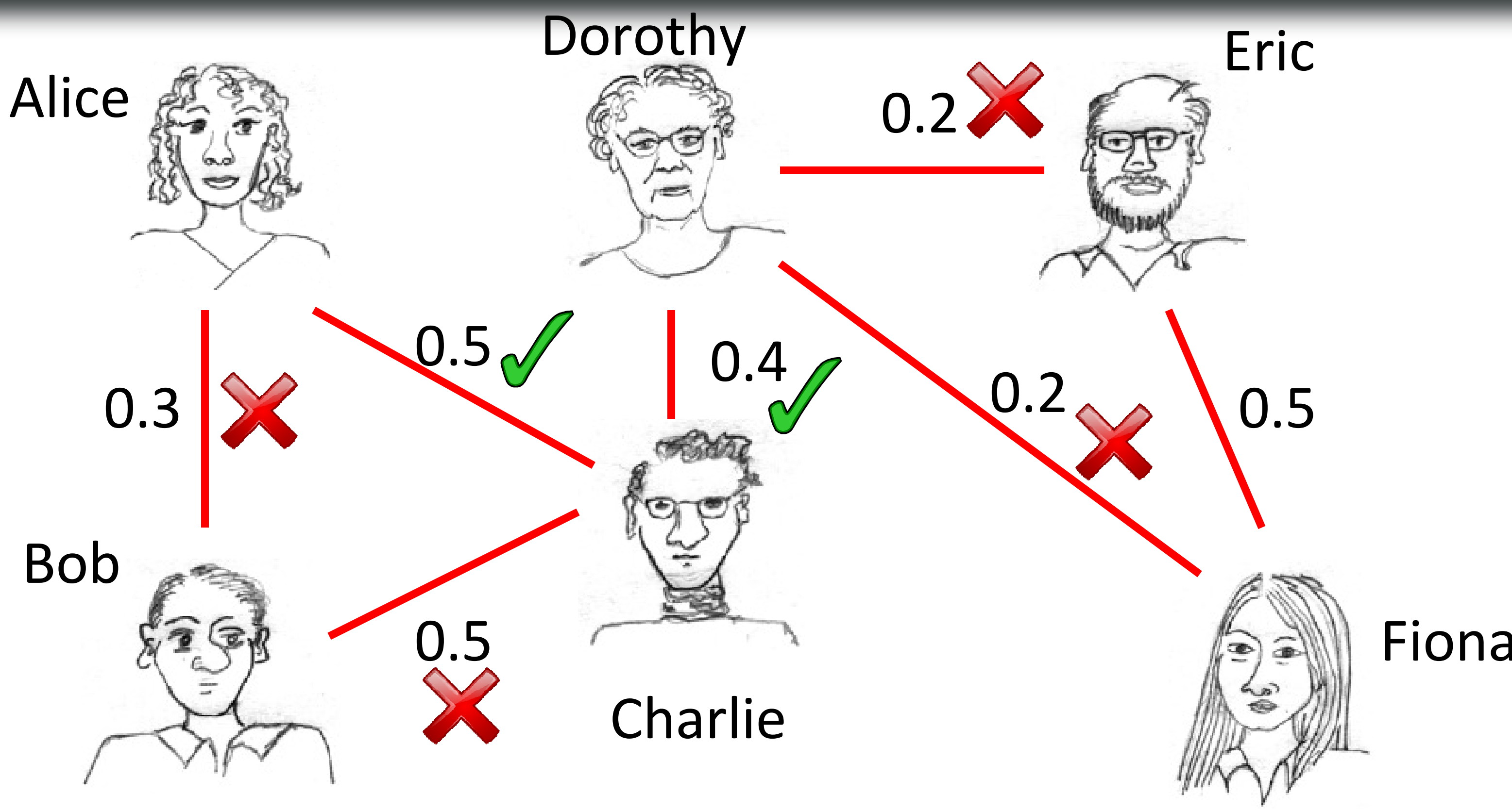
NP hard!

Maximizing submodular functions:

NP hard!

But can get approximation guarantees ☺

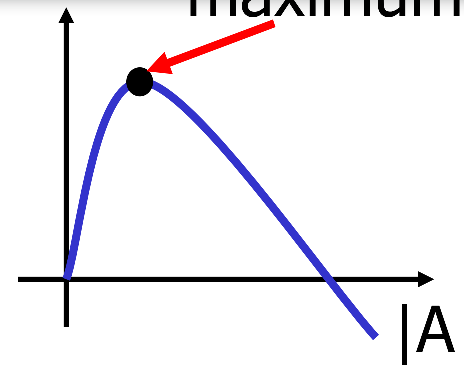
Maximizing influence
[Kempe, Kleinberg, Tardos KDD '03]



- $F(A)$ = Expected #people influenced when targeting A
- F monotonic: If $A \subseteq B$: $F(A) \leq F(B)$
  Hence $V = \text{argmax}_A\, F(A)$

  More interesting: $\text{argmax}_A\, F(A) - \text{Cost}(A)$

Maximizing non-monotonic functions



- Suppose we want for not monotonic F

$$\boxed{A^* = \operatorname{argmax} F(A) \;\; \text{s.t.} \;\; A \subseteq V}$$

- Example:
  - F(A) = U(A) – C(A) where U(A) is submodular utility, and C(A) is supermodular cost function

- In general: NP hard. Moreover:
- If F(A) can take negative values:
  As hard to approximate as maximum independent set
  (i.e., NP hard to get $O(n^{1-\varepsilon})$ approximation)

Maximizing positive submodular functions
[Feige, Mirrokni, Vondrak FOCS '07]

Theorem

There is an efficient randomized local search procedure, that, given a positive submodular function F, $F(\emptyset)=0$, returns set A_{LS} such that

$$F(A_{LS}) \geq (2/5) \max_A F(A)$$

- picking a random set gives ¼ approximation (½ approximation if F is symmetric!)
- we cannot get better than ¾ approximation unless P = NP

Scalarization vs. constrained maximization

Given monotonic utility F(A) and cost C(A), optimize:

Option 1:

$$\max_A F(A) - C(A)$$
$$\text{s.t. } A \subseteq V$$

"Scalarization"

Can get 2/5 approx…

if $F(A)-C(A) \geq 0$
for all $A \subseteq V$

Option 2:

$$\max_A F(A)$$
$$\text{s.t. } C(A) \leq B$$

"Constrained maximization"

coming up…

Positiveness is a strong requirement ☹

Constrained maximization: Outline

Monotonic submodular

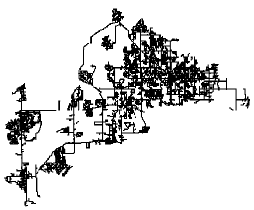
Selected set

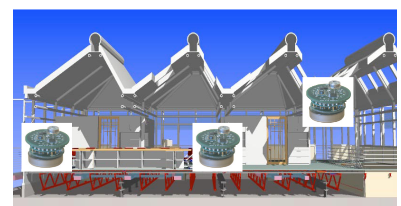
$$\max_{\mathcal{A} \subseteq \mathcal{V}} F(\mathcal{A})$$

Selection cost

Budget

$$\text{subject to } C(\mathcal{A}) \leq B$$





Subset selection: C(A) = |A|

Robust optimization

Complex constraints

Monotonicity

- A set function is called monotonic if

$$A \subseteq B \subseteq V \Rightarrow F(A) \leq F(B)$$

- Examples:
  - Influence in social networks [Kempe et al KDD '03]
  - For discrete RVs, entropy $F(A) = H(X_A)$ is monotonic:
    Suppose $B = A \cup C$. Then
    $$F(B) = H(X_A, X_C) = H(X_A) + H(X_C \mid X_A) \geq H(X_A) = F(A)$$
  - Information gain: $F(A) = H(Y) - H(Y \mid X_A)$
  - Set cover
  - Matroid rank functions (dimension of vector spaces, …)
  - …

Subset selection

- Given: Finite set V, monotonic submodular function F, F(∅) = 0
- Want: $\boxed{\begin{array}{l}\text{A}^{*} \subseteq \text{V such that} \\ \mathcal{A}^* = \underset{|\mathcal{A}| \leq k}{\operatorname{argmax}} F(\mathcal{A})\end{array}}$

NP-hard!

Exact maximization of monotonic submodular functions

1) Mixed integer programming [Nemhauser et al '81]

$$\begin{aligned} &\max \eta \\ &\text{s.t.} \quad \eta \leq F(B) + \textstyle\sum_{s \in V \setminus B} \alpha_s \delta_s(B) \text{ for all } B \subseteq S \\ &\qquad \textstyle\sum_s \alpha_s \leq k \\ &\qquad \alpha_s \in \{0,1\} \end{aligned}$$

where $\delta_s(B) = F(B \cup \{s\}) - F(B)$

Solved using constraint generation

2) Branch-and-bound: "Data-correcting algorithm" [Goldengorin et al '99]

Both algorithms worst-case exponential!

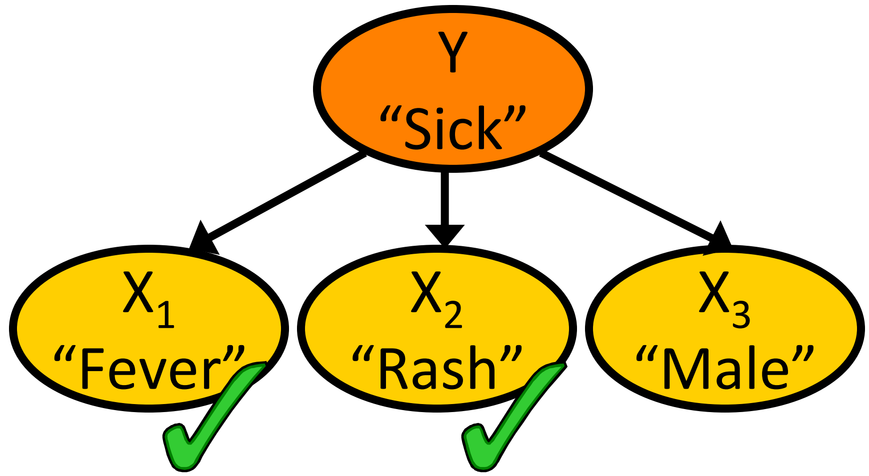
Approximate maximization

- Given: finite set V, monotonic submodular function F(A)

  Want: $A^* \subseteq V$ such that

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

NP-hard!



<u>Greedy algorithm:</u>

Start with $A_0 = \emptyset$

For i = 1 to k

$s_i := \text{argmax}_s\, F(A_{i-1} \cup \{s\}) - F(A_{i-1})$

$A_i := A_{i-1} \cup \{s_i\}$

Performance of greedy algorithm

Theorem [Nemhauser et al ‘78]

Given a monotonic submodular function F, $F(\emptyset)=0$, the greedy maximization algorithm returns A_{greedy}

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

Sidenote: Greedy algorithm gives 1/2 approximation for maximization over any matroid C! [Fisher et al ’78]

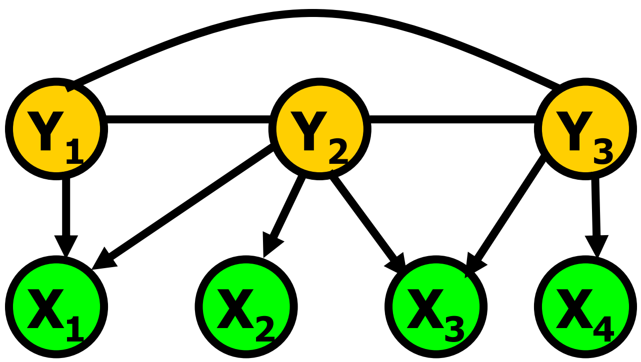
Example: Submodularity of info-gain

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

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

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

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

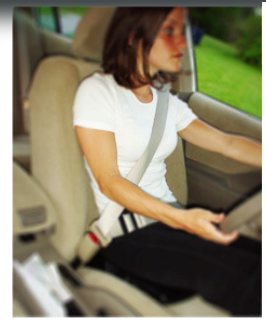


Hence, greedy algorithm works!

In fact, NO algorithm can do better than (1-1/e) approximation!

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

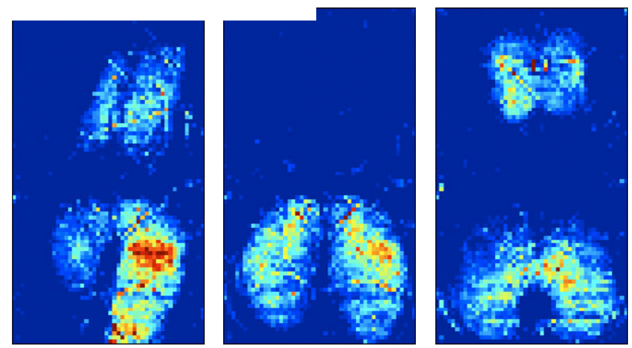
- People sit a lot
- Activity recognition in assistive technologies
- Seating pressure as user interface



Equipped with 1 sensor per cm^2!

Costs $16,000! ☹

Can we get similar accuracy with fewer, cheaper sensors?



Lean left | Lean forward | Slouch

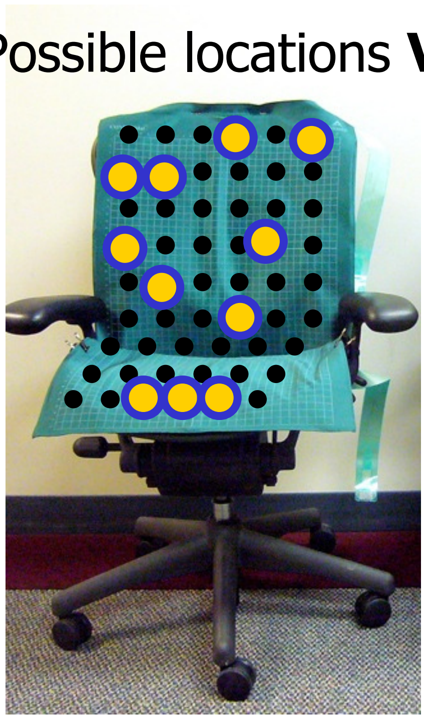
82% accuracy on 10 postures! [Tan et al]

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* $\subseteq$ 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%	**$16,000** ☹
After		

Similar accuracy at <1% of the cost!

Variance reduction

(a.k.a. Orthogonal matching pursuit, Forward Regression)

- Let $Y = \sum_i \alpha_i X_i + \varepsilon$, and $(X_1,\ldots,X_n,\varepsilon) \sim N(\cdot;\, \mu,\Sigma)$
- Want to pick subset X_A to predict Y
- $\mathrm{Var}(Y \mid X_A = x_A)$: conditional variance of Y given $X_A = x_A$
- Expected variance: $\mathrm{Var}(Y \mid X_A) = \int p(x_A)\, \mathrm{Var}(Y \mid X_A = x_A)\, dx_A$
- Variance reduction: $F_V(A) = \mathrm{Var}(Y) - \mathrm{Var}(Y \mid X_A)$

$F_V(A)$ is always monotonic

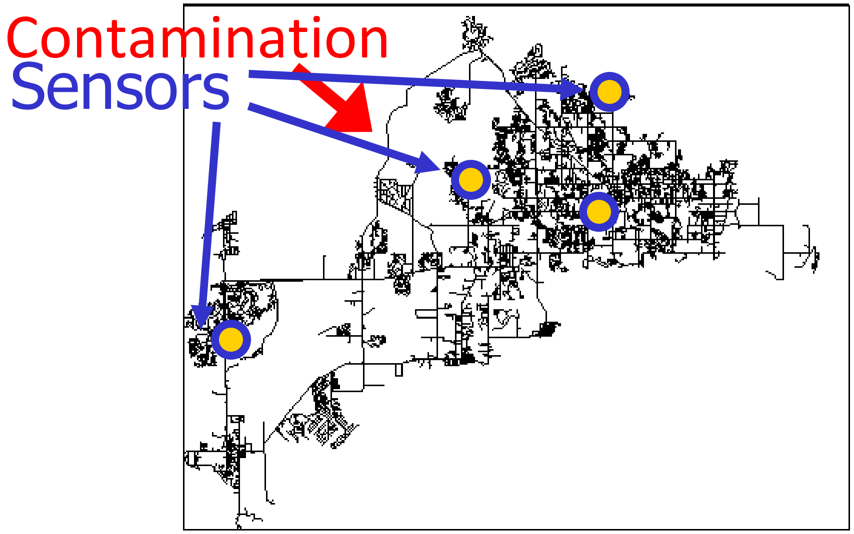
Theorem [Das & Kempe, STOC '08]
$F_V(A)$ is submodular*

*under some conditions on Σ

➔ **Orthogonal matching pursuit near optimal!**
[see other analyses by Tropp, Donoho et al., and Temlyakov]

Monitoring water networks
[Krause et al, J Wat Res Mgt 2008]

- Contamination of drinking water could affect millions of people



Simulator from EPA

Hach Sensor

~$14K

- Place sensors to detect contaminations
- "Battle of the Water Sensor Networks" competition

Where should we place sensors to quickly detect contamination?

Model-based sensing

- Utility of placing sensors based on model of the world
  - For water networks: Water flow simulator from EPA
- F(A)=Expected impact reduction placing sensors at A

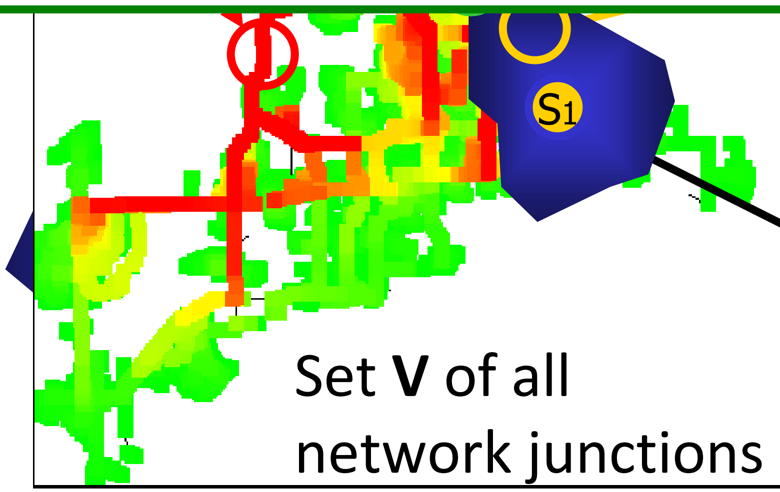
Model predicts

Low impact

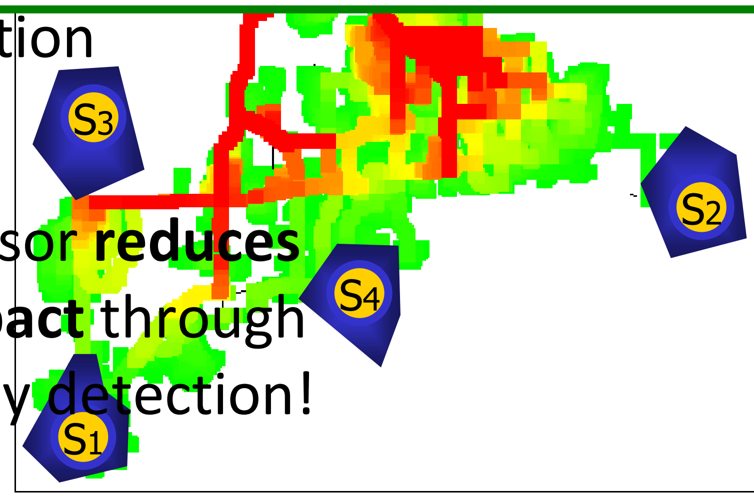
Theorem [Krause et al., J Wat Res Mgt '08]**:**

Impact reduction F(A) in water networks is submodular!

location



Sensor **reduces impact** through early detection!

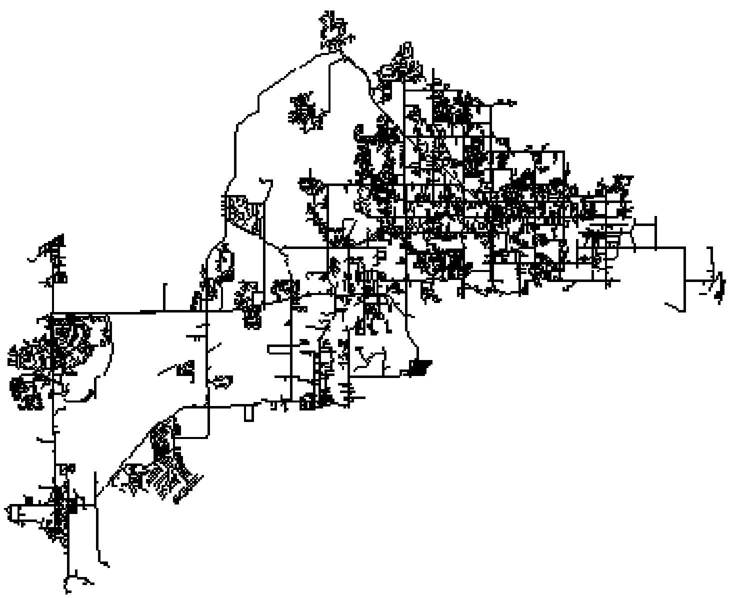


High impact reduction F(A) = 0.9

Low impact reduction F(A)=0.01

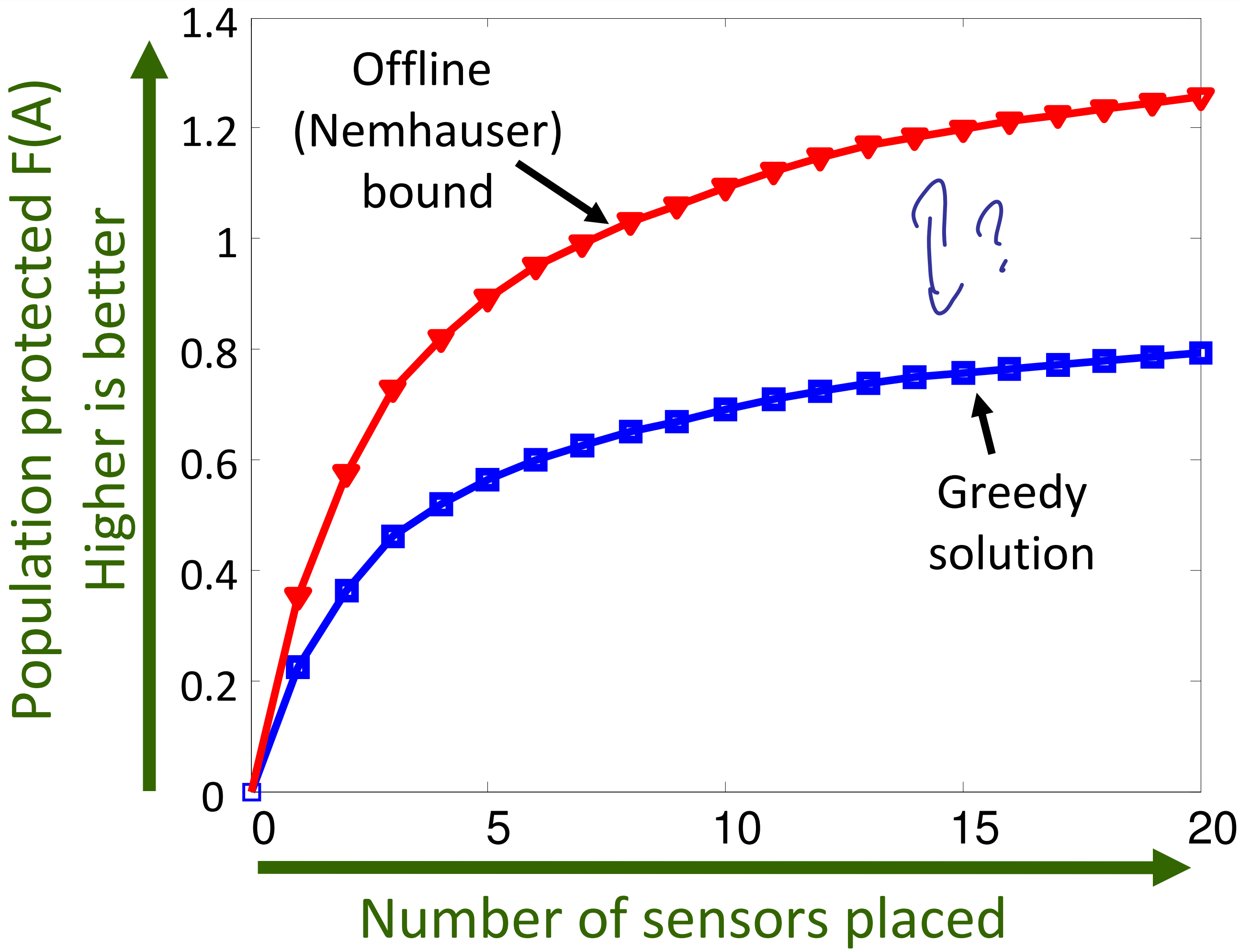
Battle of the Water Sensor Networks Competition

- 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"



Bounds on optimal solution

[Krause et al., J Wat Res Mgt '08]



Water networks data

(1-1/e) bound quite loose... can we get better bounds?

Data dependent bounds [Minoux '78]

- Suppose A is candidate solution to

$$\boxed{\operatorname{argmax} F(A) \ \text{ s.t. } |A| \leq k}$$

and $A^* = \{s_1,\dots,s_k\}$ be an optimal solution

$A \cup \{o_1 \dots o_m\}$

$$F(A^*) \leq F(\overbrace{A \cup A^*}) = F(A) + \sum_{i=1}^{m} \underbrace{\left(F(A \cup \{o_1,\dots,o_i\}) - F(A \cup \{o_1,\dots,o_{i-1}\})\right)}_{\leq \underbrace{F(A \cup \{o_i\}) - F(A)}_{\delta_i}}$$

$$\leq F(A) + \sum_{i=1}^{m} \delta_i$$

$$\leq F(A) + \sum_{i=1}^{k} \delta_i^*$$

$$\left[\leq F(A) + k \cdot \underbrace{\max_i \delta_i}_{\delta_1^*}\right]$$

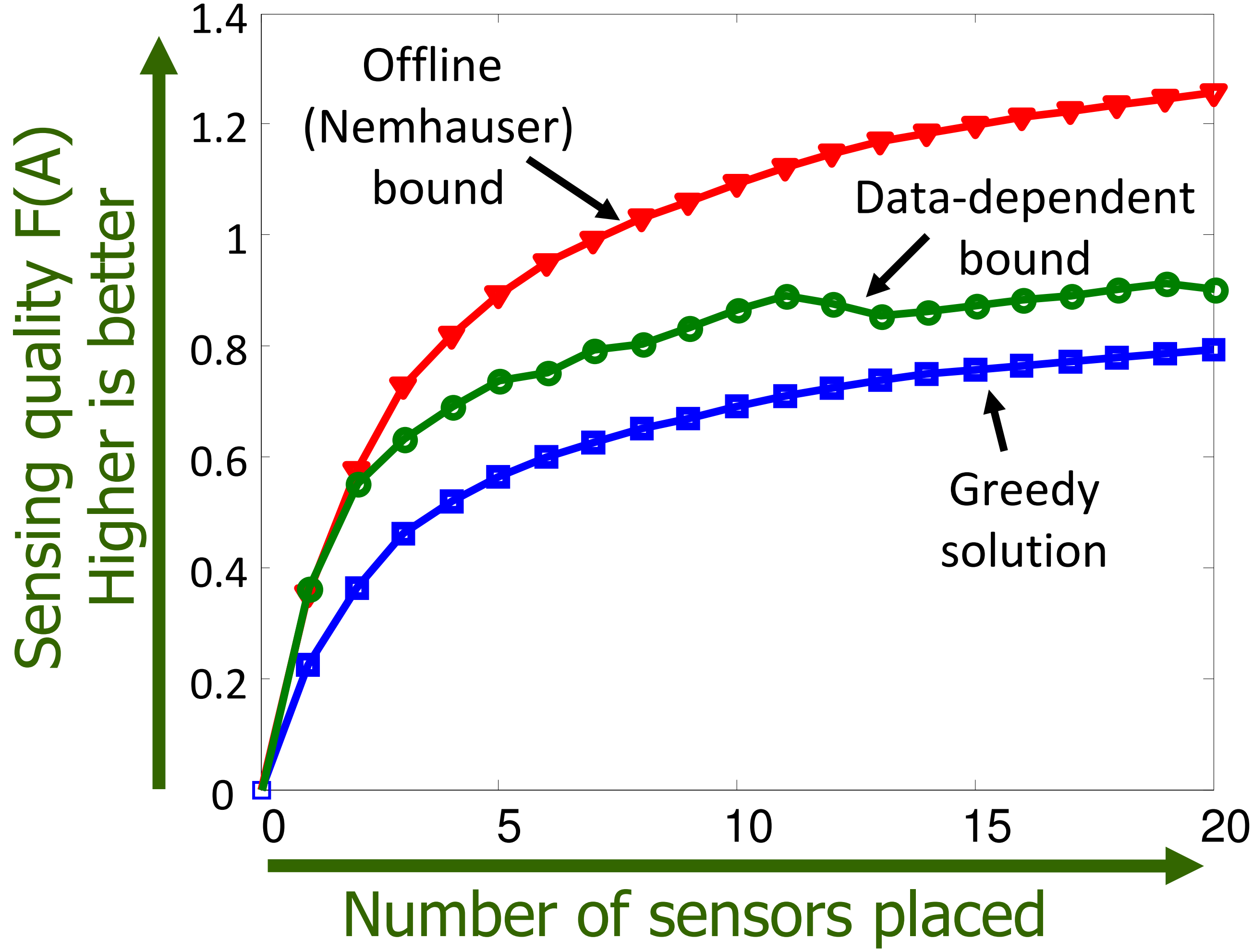
$\delta_1^* \geq \delta_2^* \geq \delta_3^* \dots$

For each $s \in V \setminus A$, let $\delta_s = F(A \cup \{s\}) - F(A)$

Order such that $\delta_1 \geq \delta_2 \geq \dots \geq \delta_n$

Then: $F(A^*) \leq F(A) + \sum_{i=1}^{k} \delta_i$

Bounds on optimal solution
[Krause et al., J Wat Res Mgt ’08]



Water networks data

Submodularity gives **data-dependent** bounds on the performance of **any** algorithm

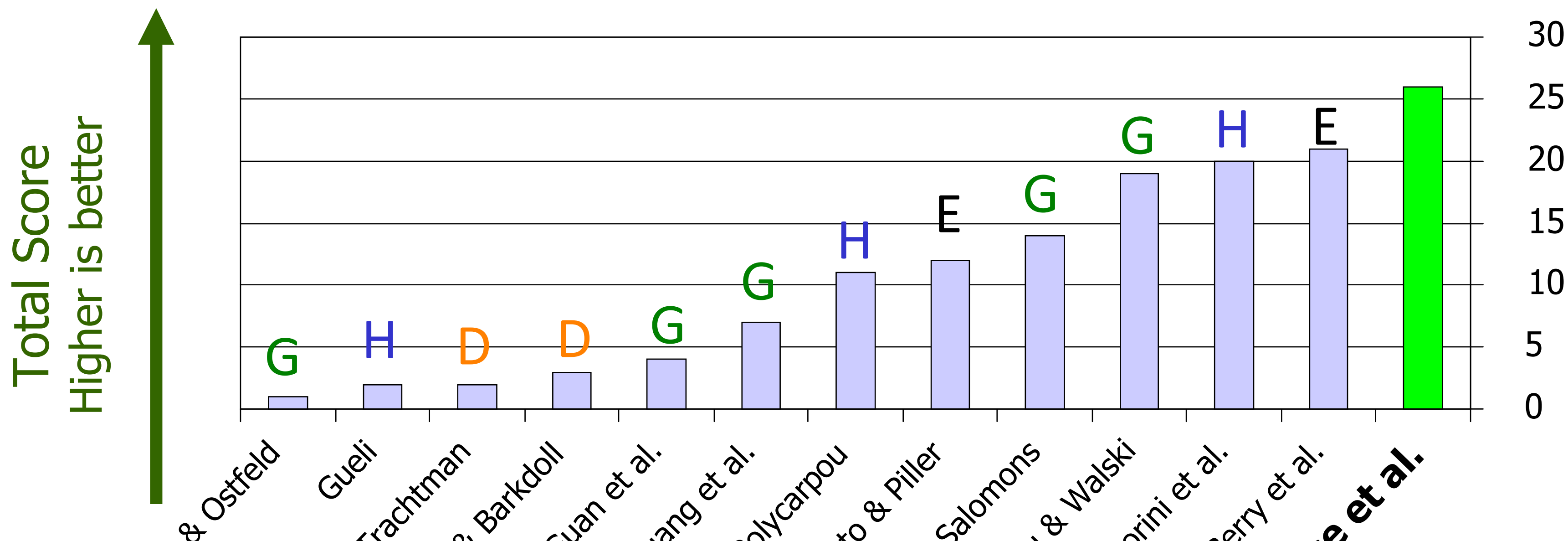
BWSN Competition results

[Ostfeld et al., J Wat Res Mgt 2008]

- 13 participants
- Performance measured in 30 different criteria

G: Genetic algorithm   D: Domain knowledge

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



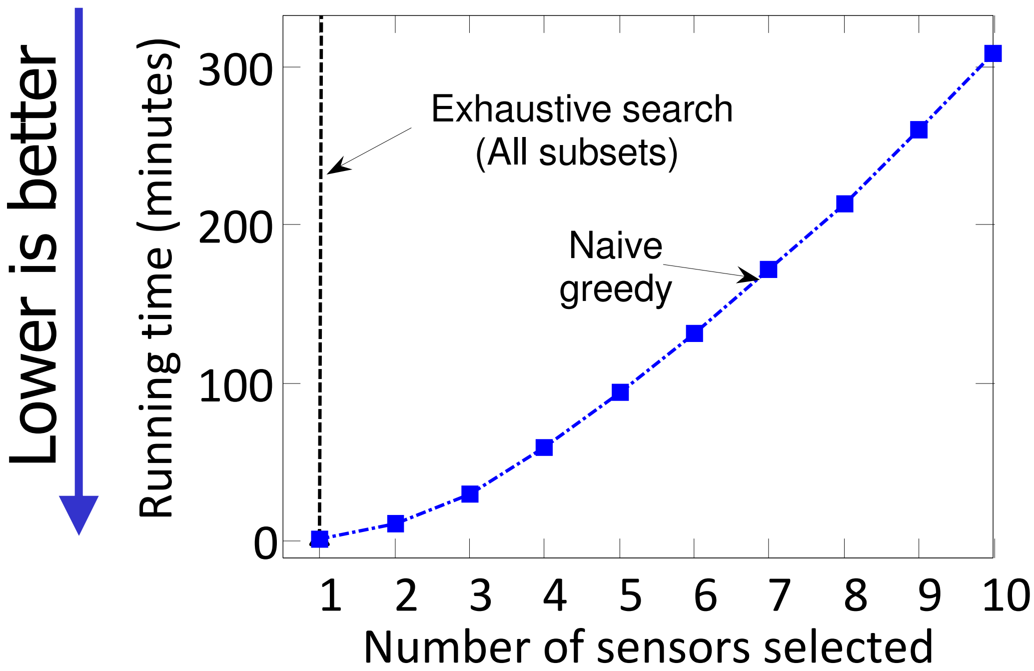
24% better performance than runner-up! ☺

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

Scaling up greedy algorithm [Minoux '78]

In round i+1,

- have picked $A_i = \{s_1,\ldots,s_i\}$
- pick $s_{i+1} = \text{argmax}_s\, F(A_i \cup \{s\})\text{-}F(A_i)$

I.e., maximize "marginal benefit" $\delta_s(A_i)$

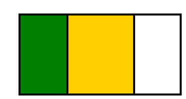
$$\delta_s(A_i) = F(A_i \cup \{s\})-F(A_i)$$

Key observation: Submodularity implies

$$i \leq j \Rightarrow \delta_s(A_i) \geq \delta_s(A_j)$$

$\delta_s(A_i) \geq \delta_s(A_{i+1})$

s

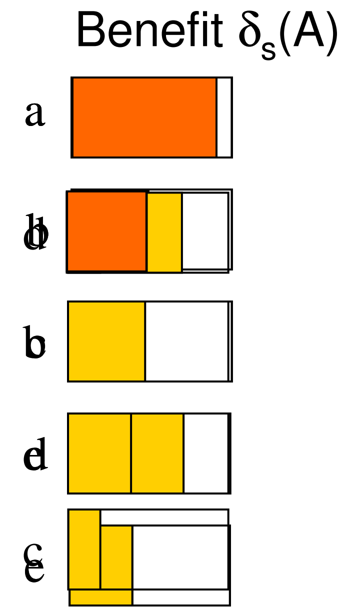


Marginal benefits can never increase!

“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



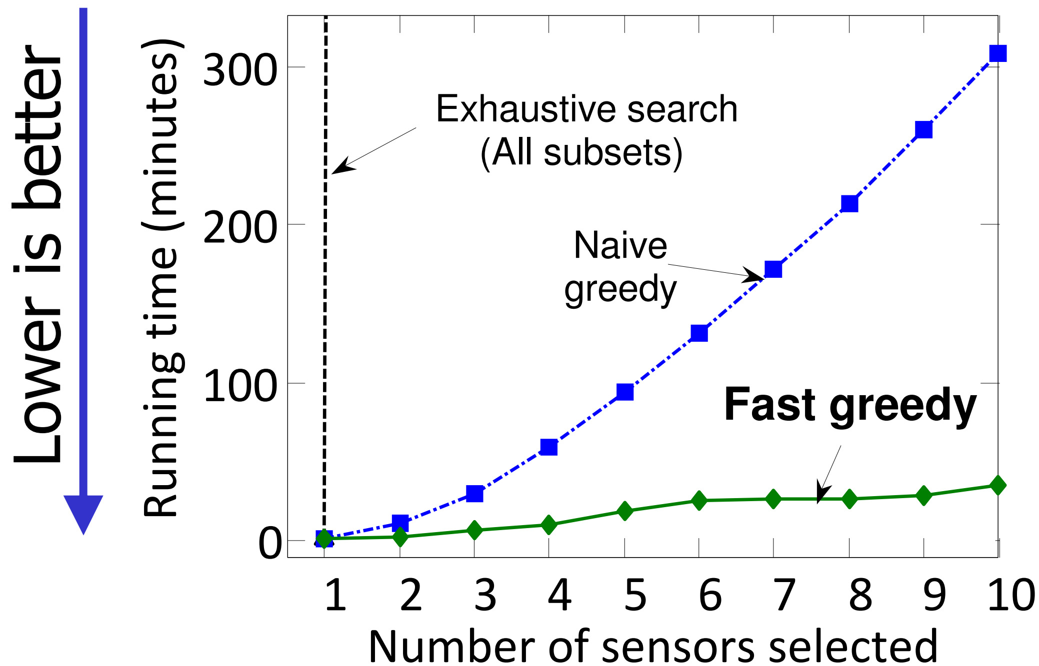
Note: Very easy to compute online bounds, lazy evaluations, etc. [Leskovec et al. ’07]

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:

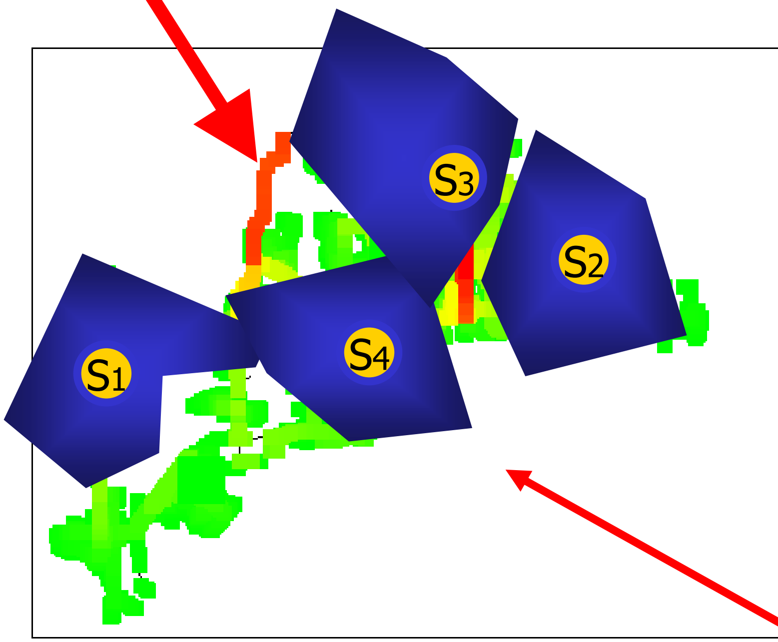
Using "lazy evaluations":

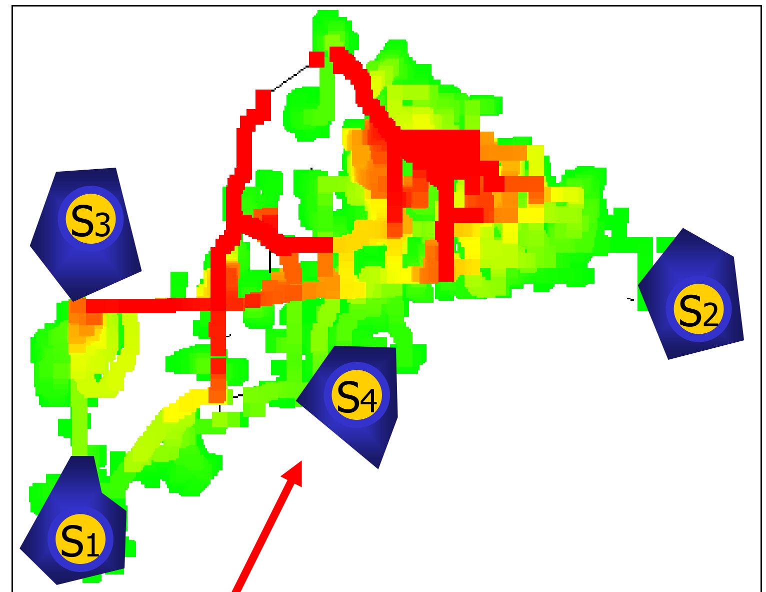
1 hour/20 sensors

Done after 2 days! ☺

What about worst-case?
[Krause et al., NIPS ’07]

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**?