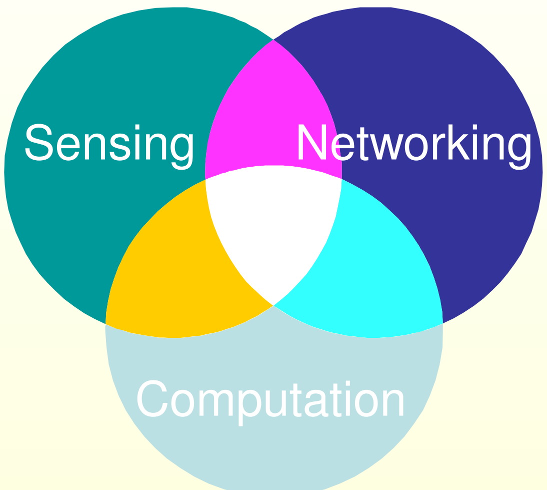
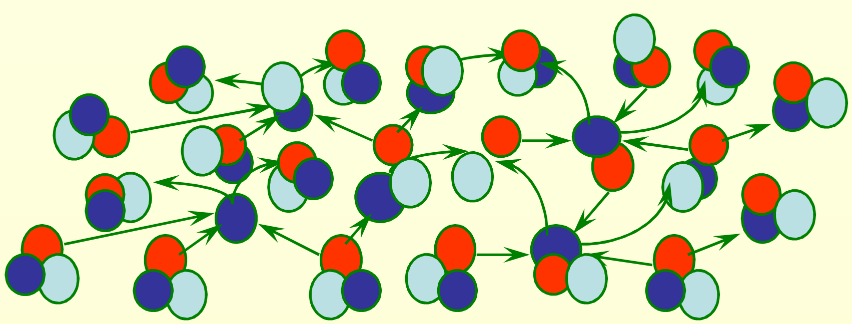
CS321: Sensor Tasking and Control



Leonidas Guibas
Computer Science Dept.
Stanford University



Sensor Deployment/Tasking

- How do we make good use of sensor nodes:
  - where should nodes be deployed?
  - which nodes should sense?
  - which nodes should communicate?

Sensing and Communication Costs ⟺ Adequacy/Quality of Information Obtained

We often need to predict the `value' of a sensor reading before we make it ...

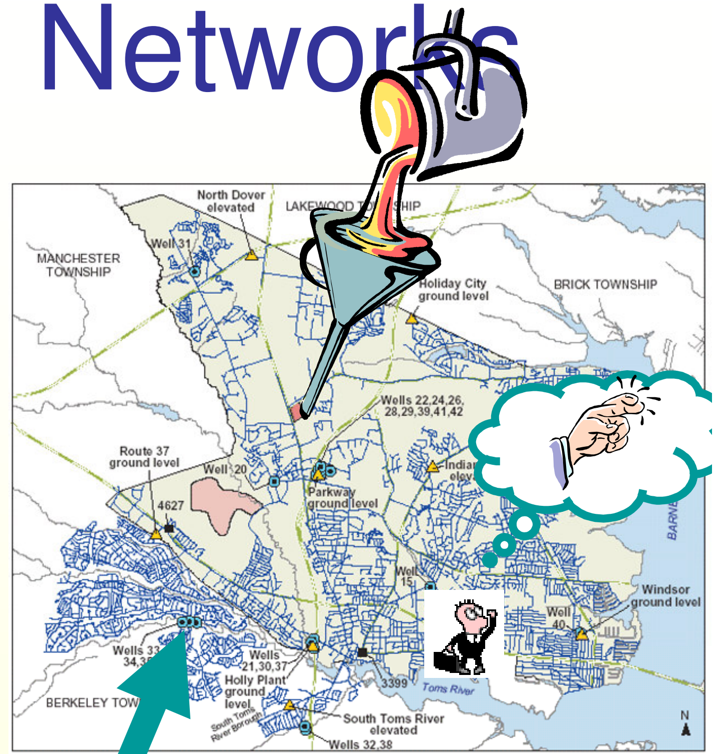
Papers

- Andreas Krause and Carlos Guestrin; Near-Optimal Nonmyopic Value of Information in Graphical Models; 21st Conference on **Uncertainty in Artificial Intelligence** (UAI 2005), Edinburgh, July 2005.
- Carlos Guestrin, Andreas Krause and Ajit Singh; Near-Optimal Sensor Placements in Gaussian Processes; 22nd International Conference on **Machine Learning** (ICML 2005), Bonn, August 2005.
- Andreas Krause, Carlos Guestrin, Anupam Gupta, Jon Kleinberg; Near-optimal Sensor Placements: Maximizing Information while Minimizing Communication Cost; Fifth International Conference on **Information Processing in Sensor Networks** (IPSN'06), April 2006.
- Andreas Krause and Carlos Guestrin; Near-optimal Observation Selection Using Submodular Functions; survey paper for the Nectar track in the 22nd Conference on **Artificial Intelligence** (AAAI), Vancouver, July 2007.
- Jure Leskovec, Andreas Krause, Carlos Guestrin, Christos Faloutsos, Jeanne VanBriesen and Natalie Glance; Cost-Effective Outbreak Detection in Networks; 13th ACM SIGKDD International Conference on **Knowledge Discovery and Data Mining** (KDD), San Jose, August 2007.

Sensor Placement/Tasking Examples

Chlorine

Water Distribution Networks



- **Water distribution** in a city → very **complex system**
- **Pathogens** in water can **affect thousands** (or millions) of people
- Currently: **Add chlorine** to the source and **hope for best**

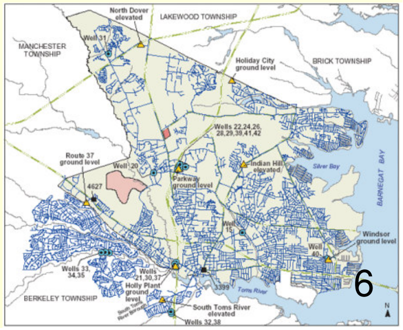
Barnegat Bay, NJ

ATTACK!
could deliberately
introduce pathogen

Sensing the Waterways

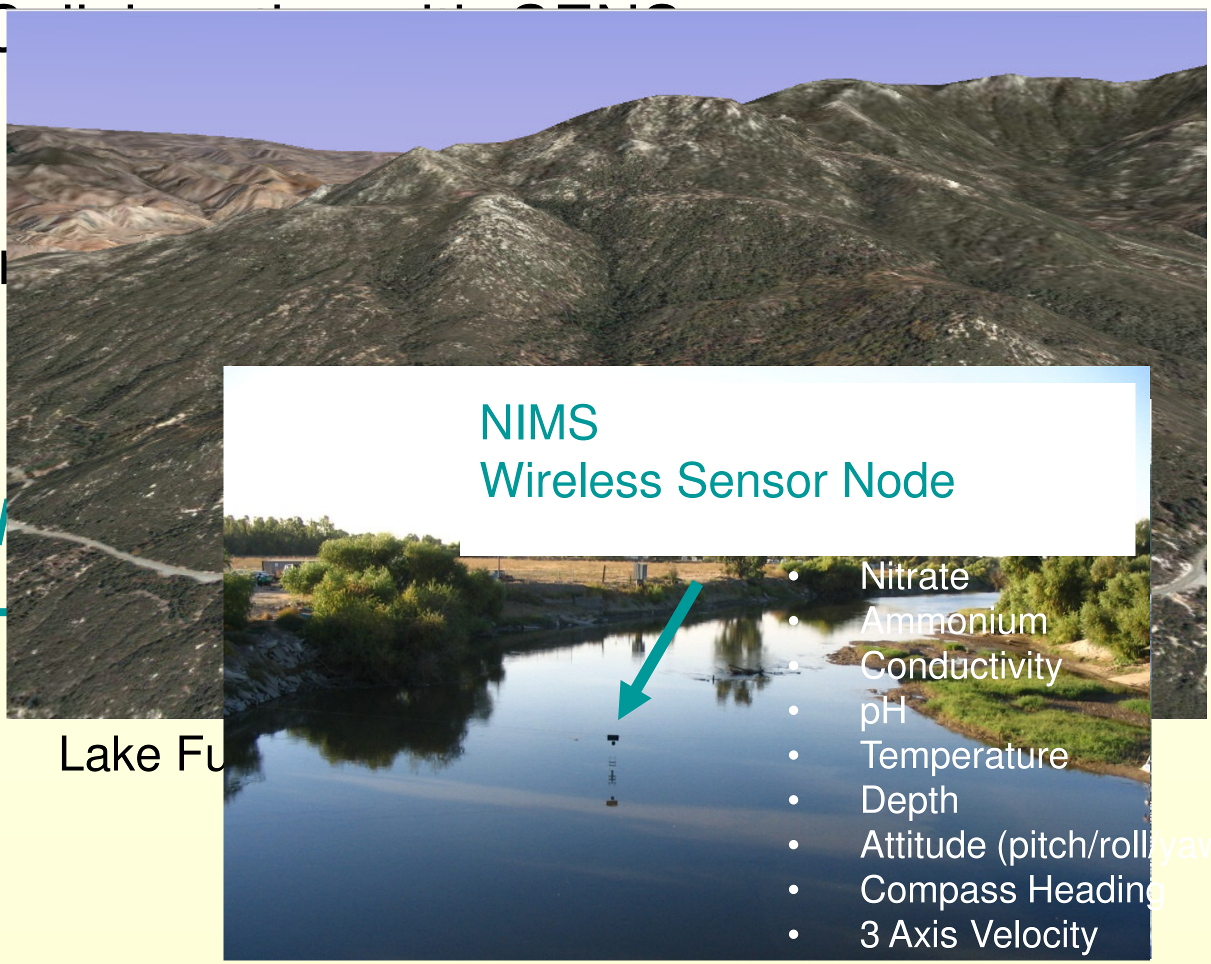
- Sensors in pipes (or taps) could detect pathogens quickly
  - minimize population affected, …
- 1 Sensor: $5,000 (just for chlorine)
- Local government not willing to spend much money
  - **Must be smart about where to place sensors**

- **Battle of the Water Sensor Networks** (competition Oct. 2006)
  - Get **model of a city**
  - **Simulator** of water flow provided by the **EPA**
  - Competition for **best placements**



Monitoring Algae Biomass in Lakes and Rivers

- Algae levels too high → Monitor biomass



Lake Fu

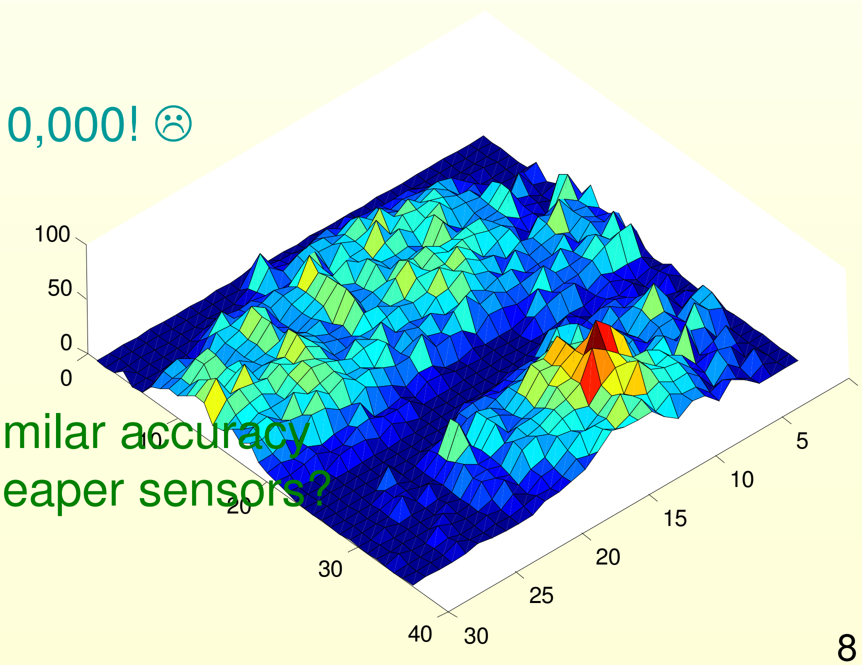
Chair Sensor Placement

- Help elderly stay active/healthy by activity recognition using a pressure sensing chair

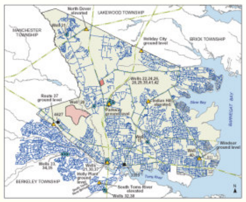
Equipped with
1 sensor per cm^2! ☺

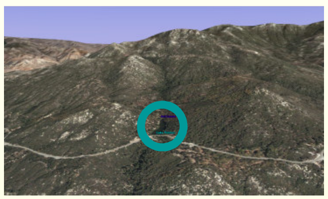
Costs $10,000! ☹

Can we get similar accuracy
with fewer, cheaper sensors?



Fundamental Question: Sensor Placement





Where should we place sensors to monitor spatial phenomena?

- pathogen distribution
- algae biomass
- temperature and light field
- rear-end pressure
- …

PART I:

Near-Optimal Sensor Placement:

Maximizing Information while

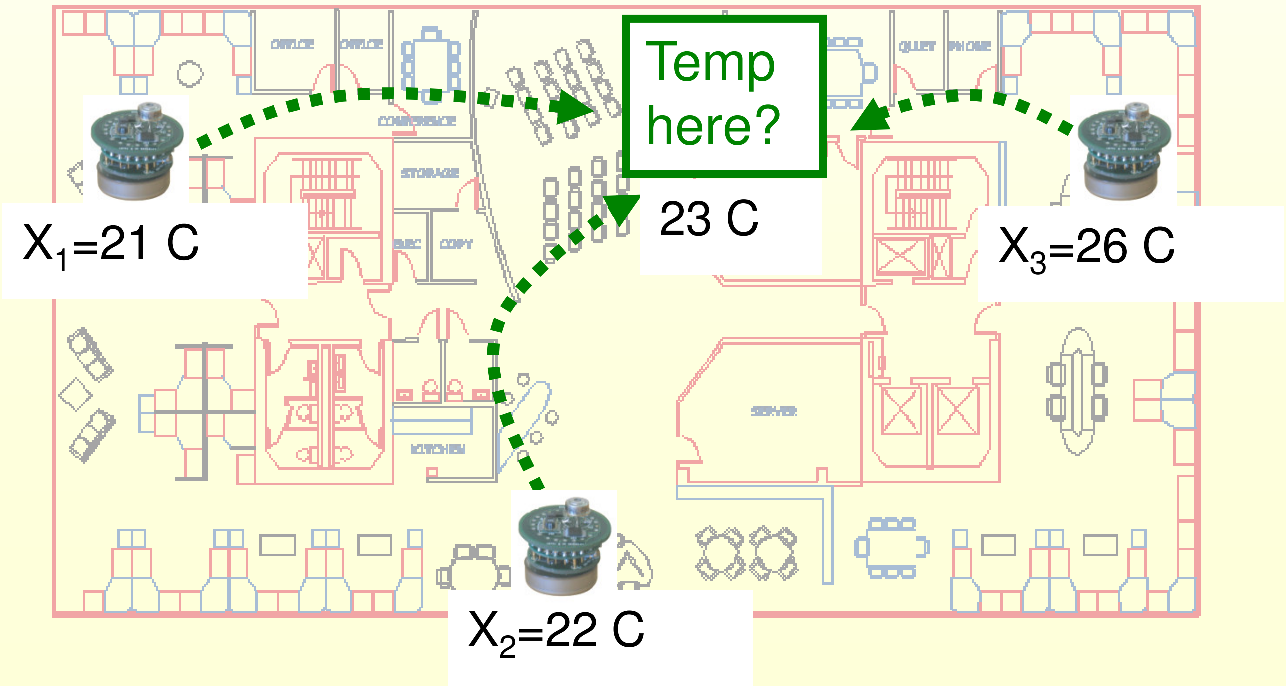
Minimizing Communication Cost

First question in presentation:
How do we use sensors to monitor
a spatial phenomena?

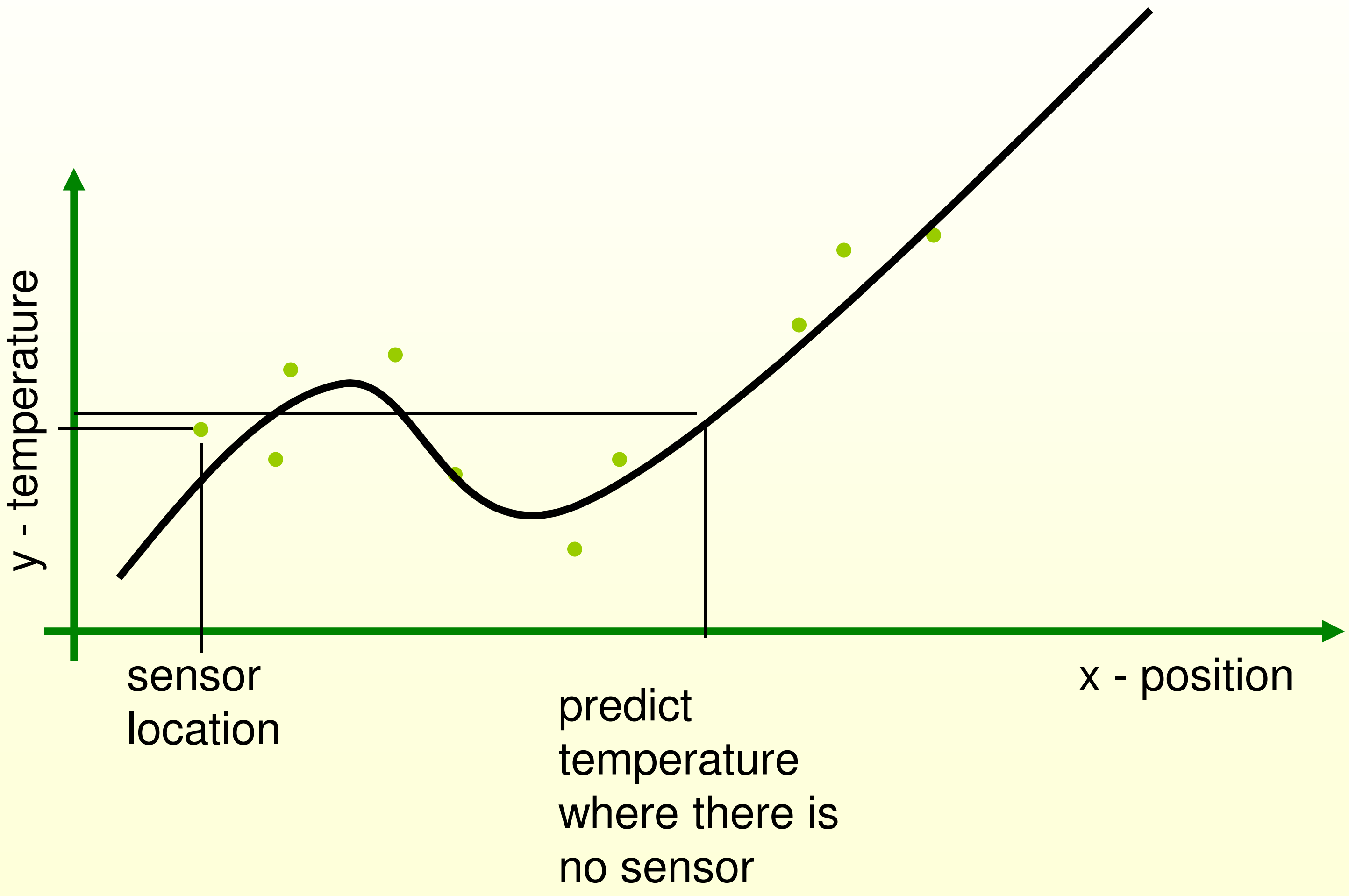
[Guestrin, Krause, Singh, '05]

Predicting Spatial Phenomena from Sensor Readings

- Can only measure where we have sensors
- **Multiple sensors** can be used to predict phenomena at **uninstrumented** locations



Prediction is a Regression Problem

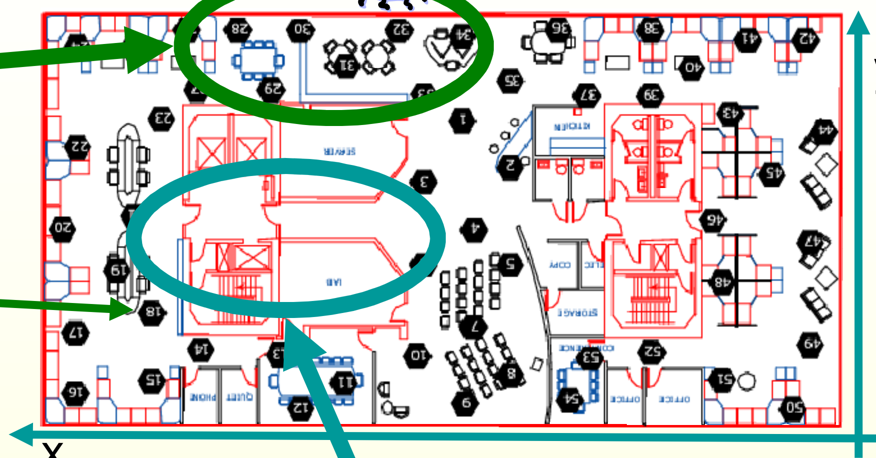


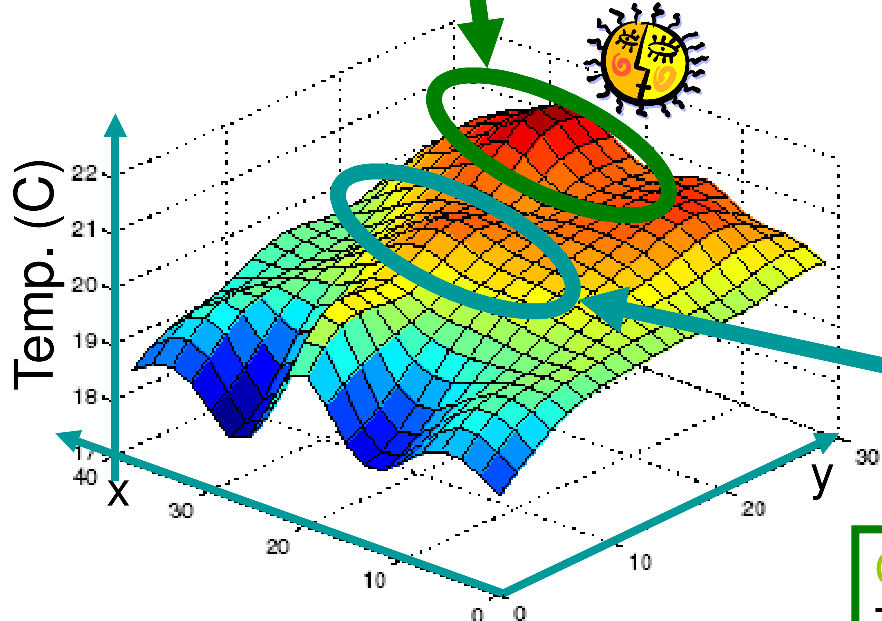
Regression Models for Spatial Phenomena

Real deployment of temperature sensors

many sensors around → trust estimate here

measurements from 52 sensors (black dots)





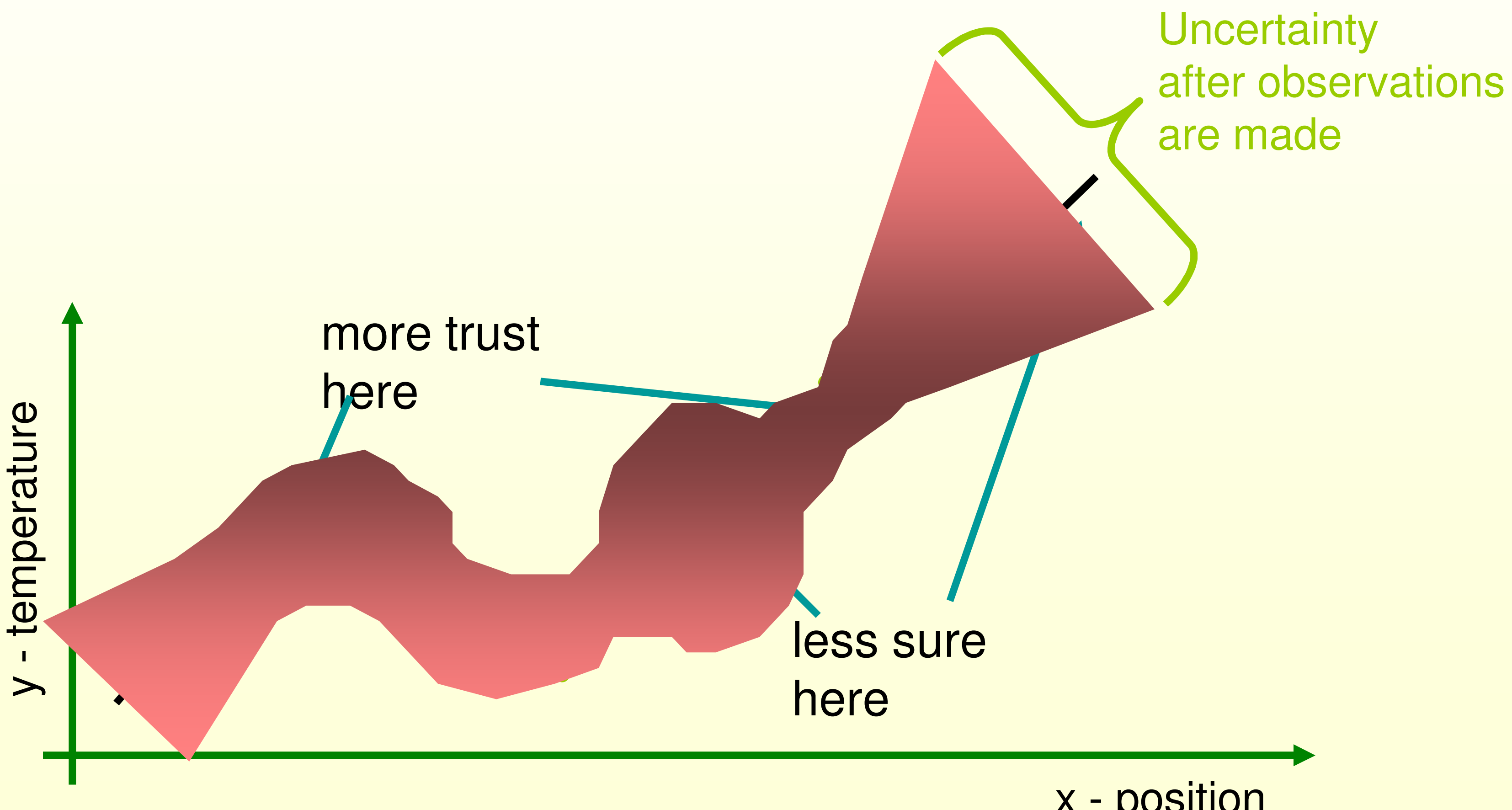
few sensors around → don't trust estimate

Predicted temperature throughout the space

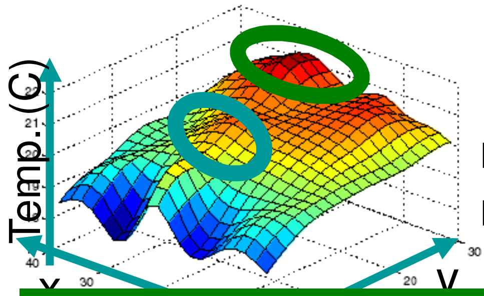
Good sensor placements: Trust estimate everywhere!

Uncertainty about Prediction – Intuition

trusting estimate ↔ low uncertainty about prediction

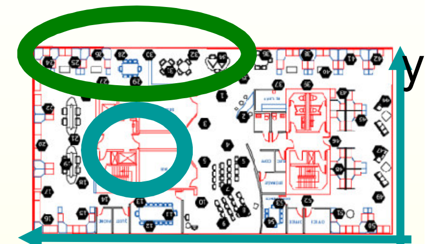


Probabilistic Models for Spatial Phenomena



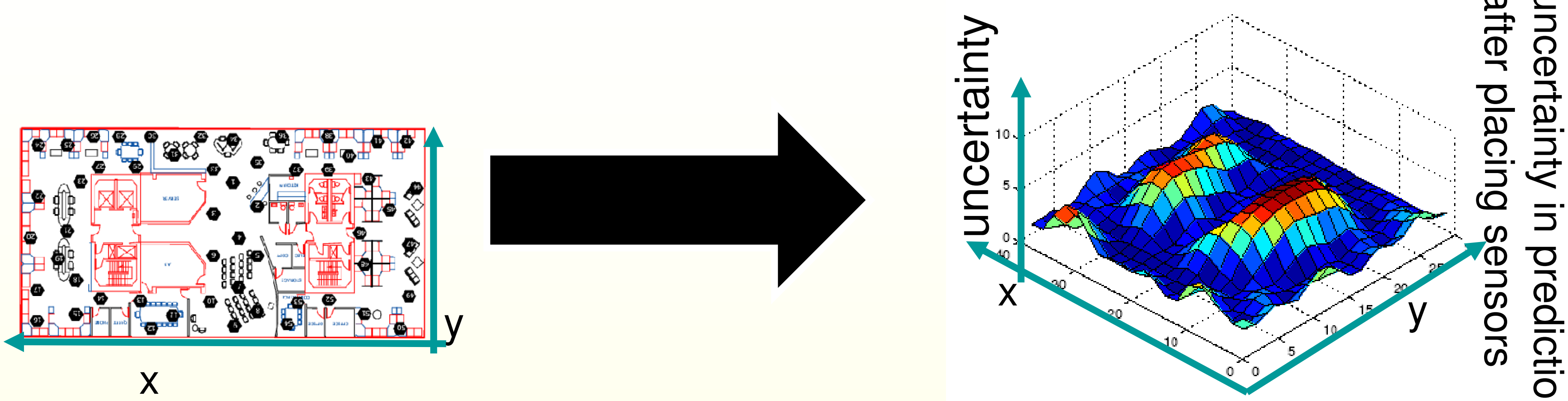
regression model

sensor locations



- Modeling uncertainty is fundamental!
- Many rich probabilistic models
  - gaussian processes, a *non-parametric* model [O'Hagan ’78]
  - Learned from pilot data or computed from expert model (e.g., EPA)
- Learning model is well-understood → focus on optimizing sensor locations

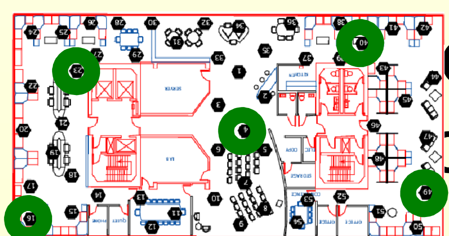
Information Quality



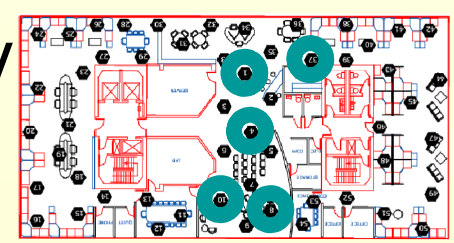
sensor placement *A*

information quality *I(A)*

- Pick **locations *A*** with **highest information quality**
  - lowest "uncertainty" after placing sensors
  - d in terms of entropy ior
    on



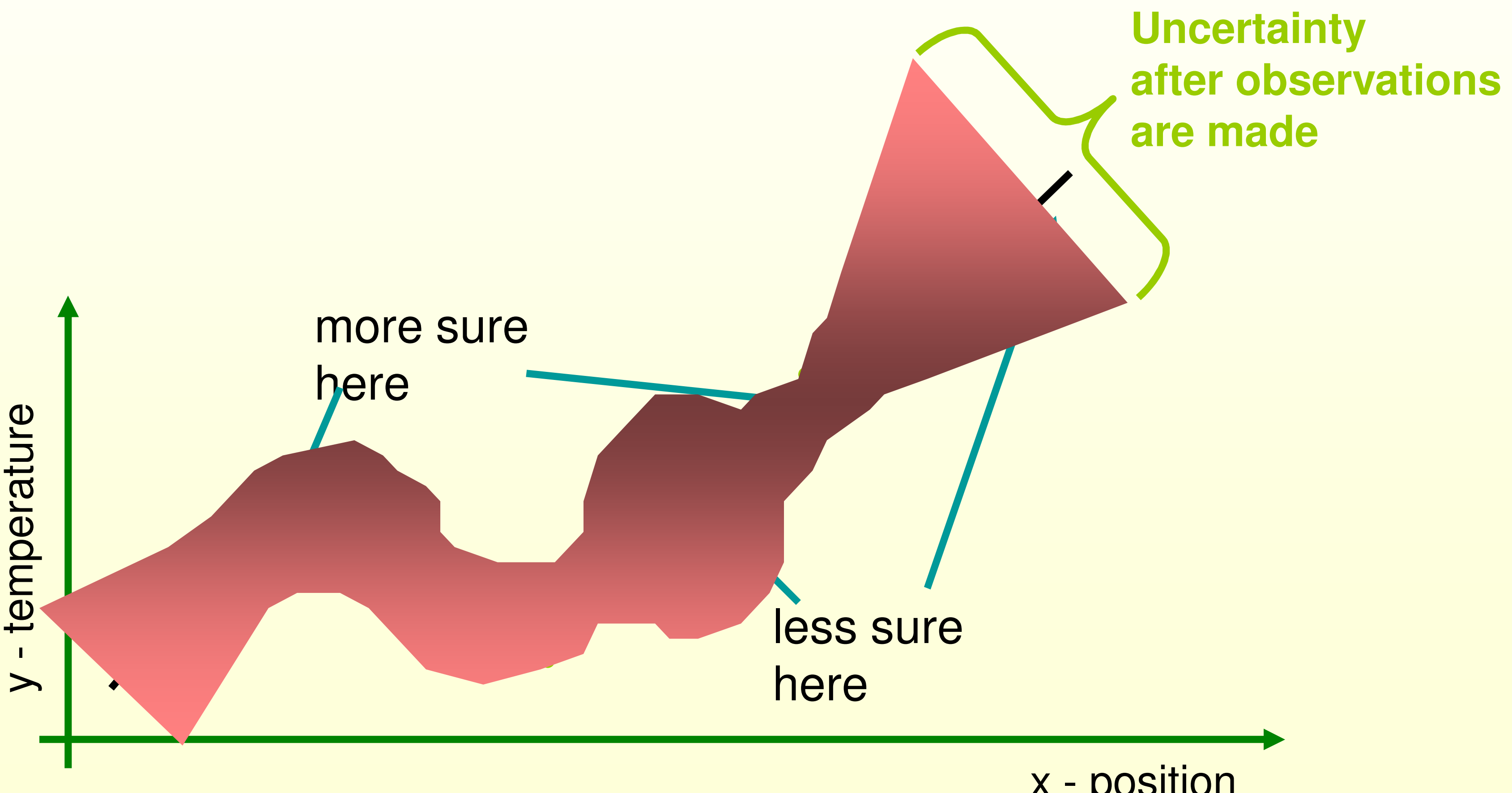
placement *A* $I(A) = 10$



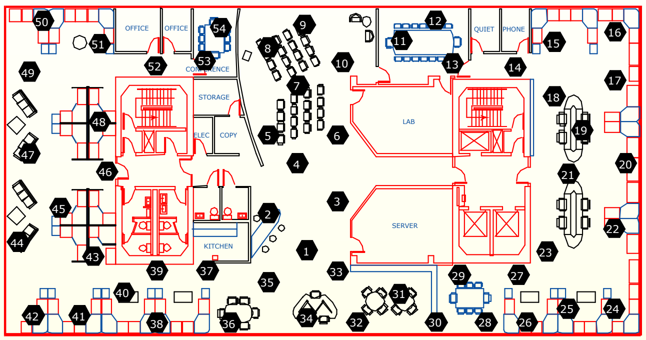
placement *B* $I(B) = 4$

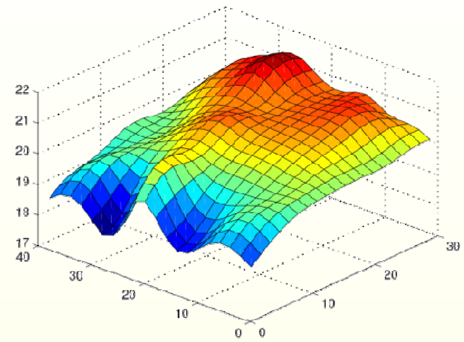
Gaussian Processes (GP) - Intuition

GP – Non-parametric; represents uncertainty;
complex correlation functions (kernels)

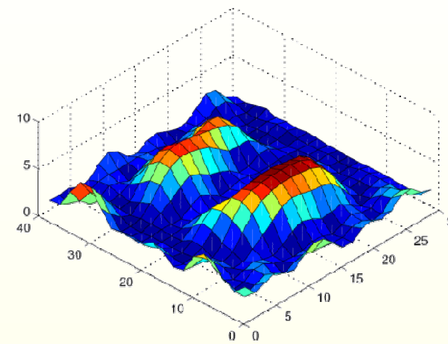


Gaussian Processes





Posterior
mean temperature



Posterior
variance

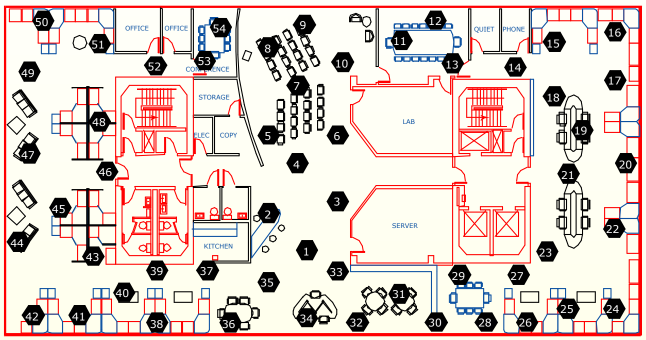
Kernel function:

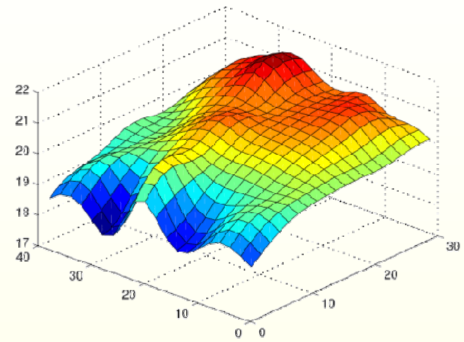
$$\sigma_{ij} = K(x_i, x_j)$$

Prediction after observing
set of sensors *A*:

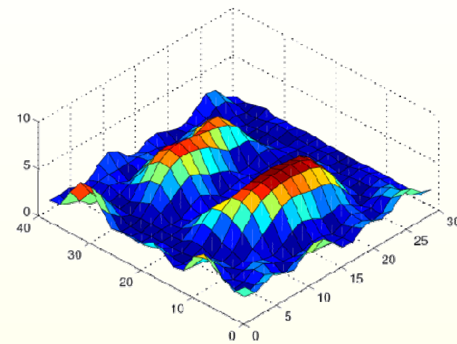
$$\mu_{Y|A} = \mu_Y + \Sigma_{YA}\Sigma_A^{-1}(\mathbf{x}_A - \mu_A)$$
$$\sigma_{Y|A} = \sigma_Y + \Sigma_{YA}\Sigma_A^{-1}\Sigma_{AY}$$

Gaussian Processes for Sensor Placement





Posterior mean temperature



Posterior variance

Goal:
Find sensor placement with least uncertainty *after* observations

Problem is still NP-complete ☹
Need approximation

Sensor Placements

- Consider **myopically** selecting

$$H(A_1) + H(A_2 \mid \{A_1\}) + \ldots + H(A_k \mid \{A_1 \ldots A_{k-1}\})$$

most uncertain

most uncertain given A_1

This is exactly the joint entropy

$H(\mathbf{A}) = H(\{A_1 \ldots A_k\})$

most uncertain given $A_1 \ldots A_{k-1}$

- This can be seen as an attempt to maximize

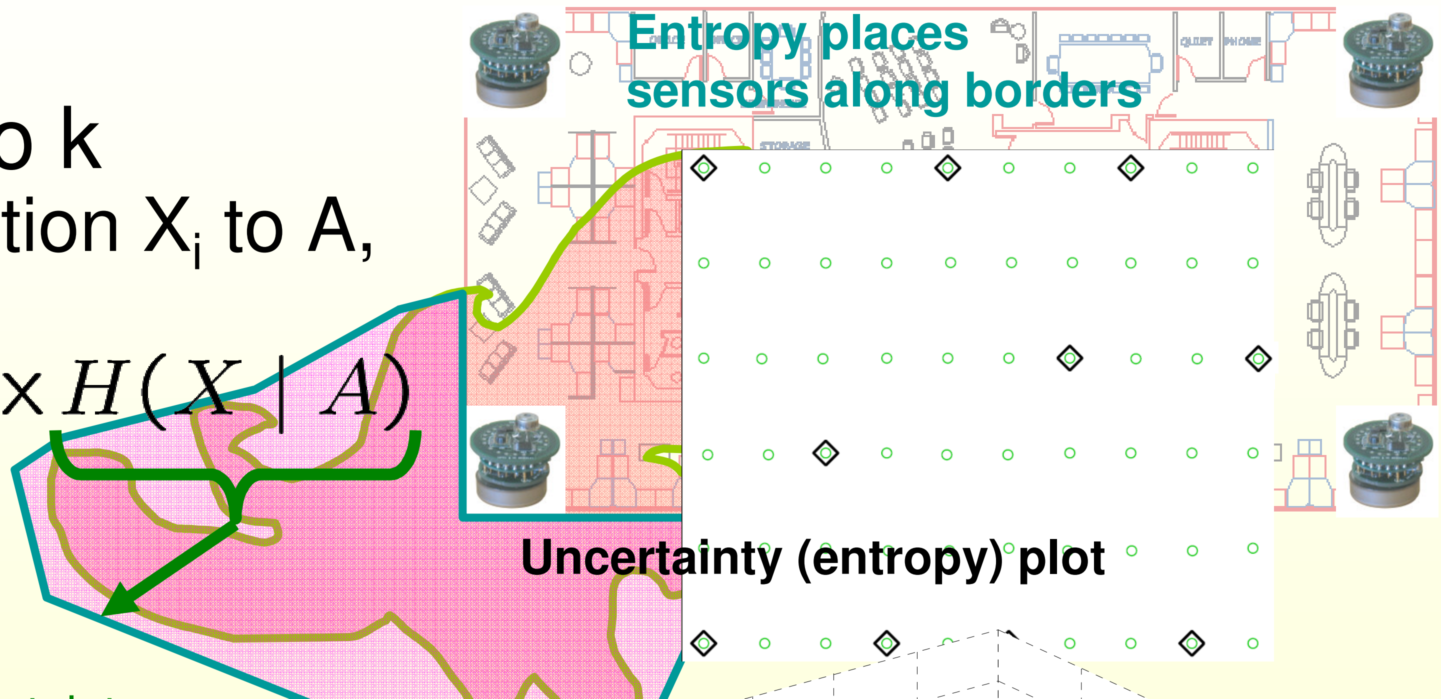
$$\max_{\mathbf{A} \subset \mathbf{S}} H(\mathbf{A}) \text{ subject to } |\mathbf{A}| \leq k$$

Entropy Criterion (*c.f.*, [Cressie '91])

- A ← ∅
- For i = 1 to k
  - Add location X_i to A, s.t.:

$$X_i = \underset{X}{\operatorname{argmax}}\, H(X \mid A)$$

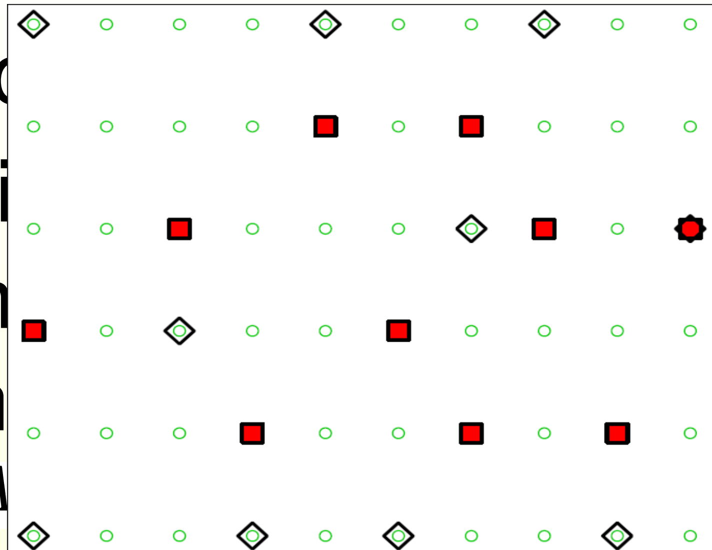
“Wasted” information observed by



Entropy criterion wastes information [O'Hagan '78], Indirect, doesn't consider sensing region –
No formal guarantees ☹

Objective Function: Mutual Information

- Lo V
- Fi m in



Temperature data placements:
Entropy
Mutual information

$I(A$ $H(V \setminus A \mid A)$

Intuitive criterion – Locations that are both different and informative
Formal guarantees ☺

a

given A –
X is different

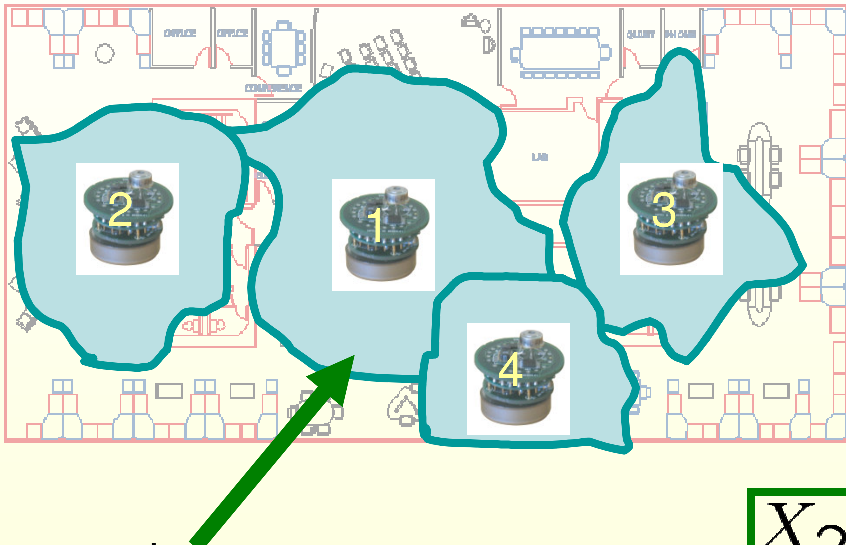
given rest –
X is informative

The Placement Problem

- Let **V** be set of locations we can choose from
- For subset of locations $\mathbf{A} \subseteq \mathbf{V}$, let
  - *I(A)* be **information quality** of placement A
- Want to optimize
  $\max_A \; I(A)$ subject to $|A| \leq \mathbf{k}$
- A well-studied problem: *c.f.* [Lindley '56, O'Hagan '78, Sacks et al. '89, Cressie '91, Rasmussen '96, Atkinson '96, Flaherty et al. '05,…]
- **NP-hard** → most **existing methods** are heuristics with **no quality guarantees**
- **Let's look at one such heuristic…**

Simple Greedy Algorithm

Greedy:



area where sensor provides "information"

- First pick most informative location:

$$X_1 = \operatorname*{argmax}_X I(X)$$

- Then, pick most **improvement** given previous one:

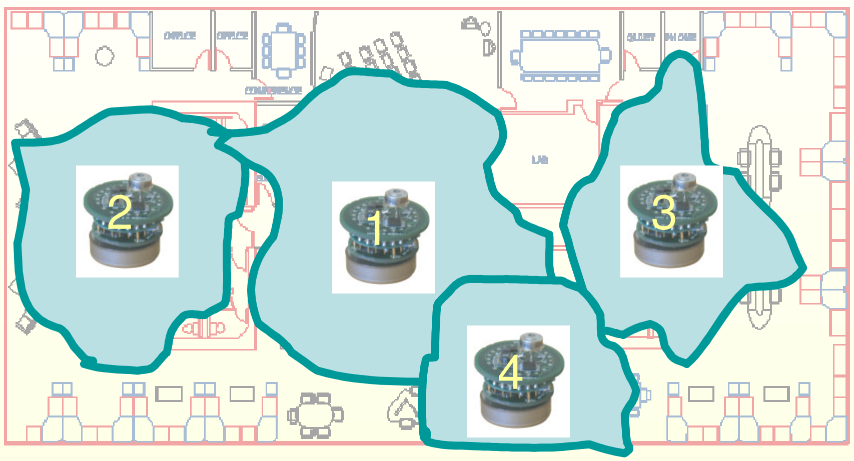
$$X_2 = \operatorname*{argmax}_X I(X \cup X_1) - I(X_1)$$

- And so on…

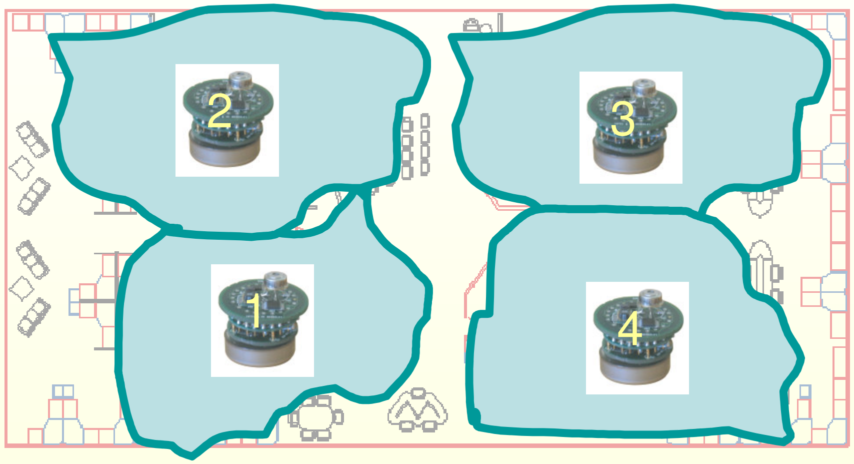
$$X_3 = \operatorname*{argmax}_X I(X \cup X_1 \cup X_2) - I(X_1 \cup X_2)$$

Greedy Algorithms are Generally Suboptimal

Greedy:

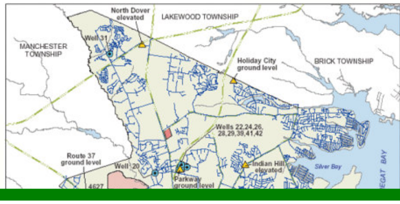


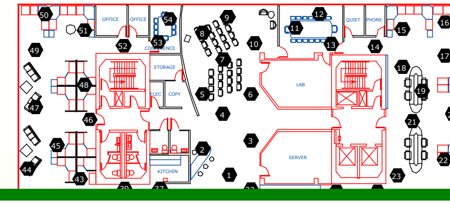
Optimal:



- **Greedy** algorithm provides **suboptimal** solution ☹
- But greedy **doesn’t look that bad**… ☺

Greedy is Pretty Good!

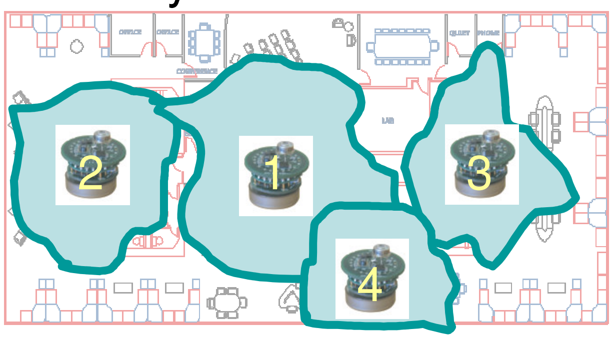




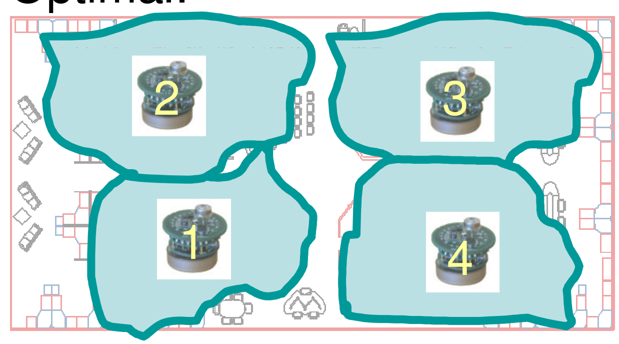
simple greedy algorithm works rather well!

Why ????

Greedy:

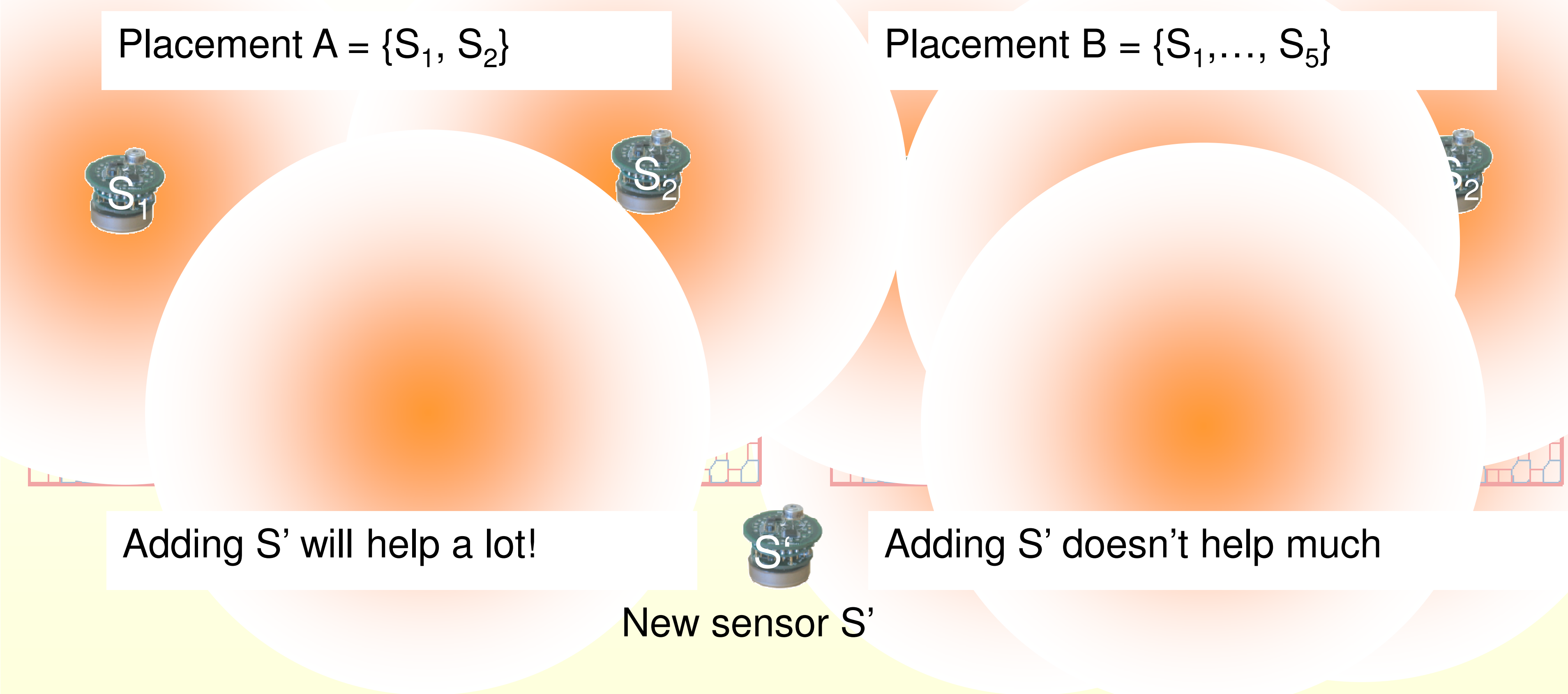


Optimal:

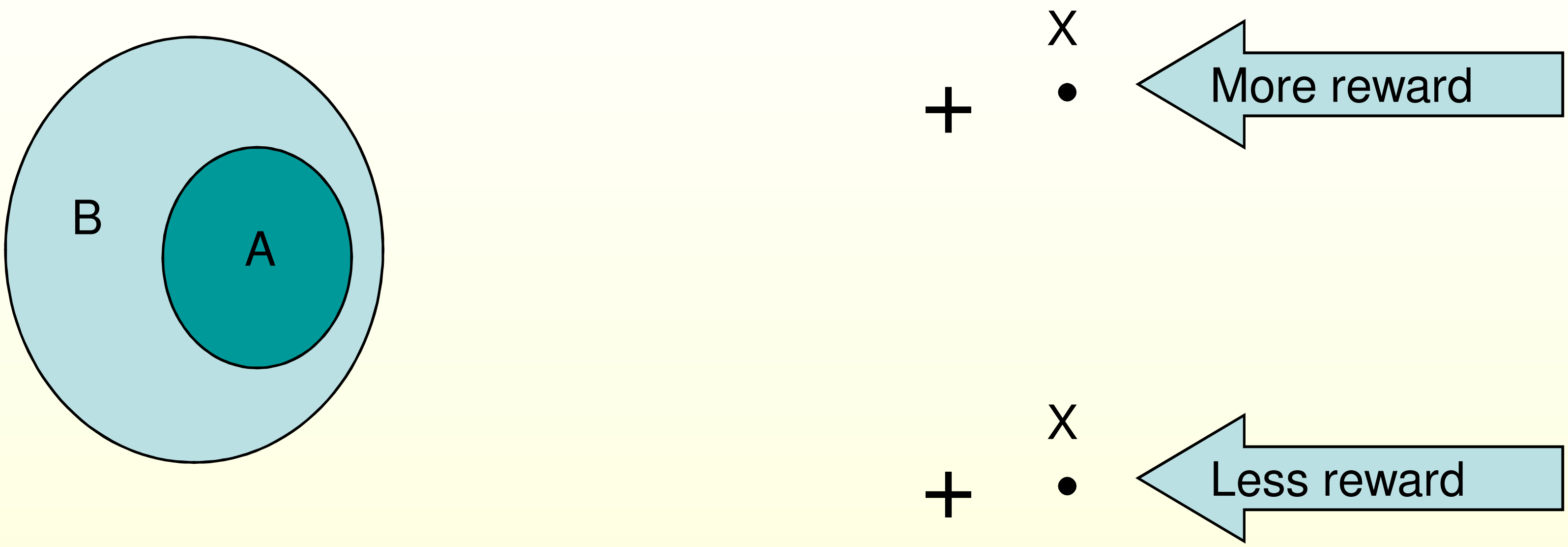


- Greedy behaves **well in the first few steps**
- By the time it becomes “**suboptimal**”, there **isn’t much information** to be had…

Key Observation: Law of Diminishing Returns



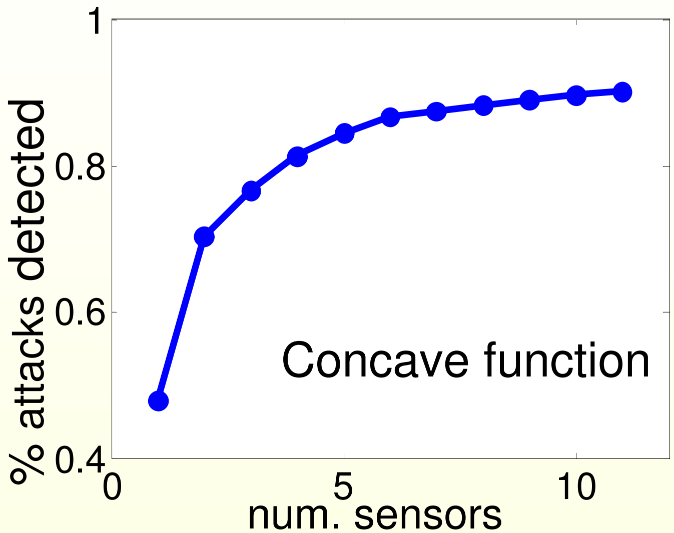
Submodular Functions: Functions on Sets



$$F(A \cup X) - F(A) \geq F(B \cup X) - F(B)$$
$$A \subseteq B$$

Submodularity and Sensor Placement

Many sensor placement objectives are submodular!!! :-)



If I(A) is a submodular function
Concave behavior: ↙ improvement provided by last sensor
diminishing returns → improvement small ☺

Sensor Placement for Information Quality

- **Theorem: Greedy** algorithm provides **constant factor approximation**: placing *k* sensors:

$$I(A_{\text{greedy}}) \geq (1 - 1/e) \max_{A:|A|=k} I(A)$$

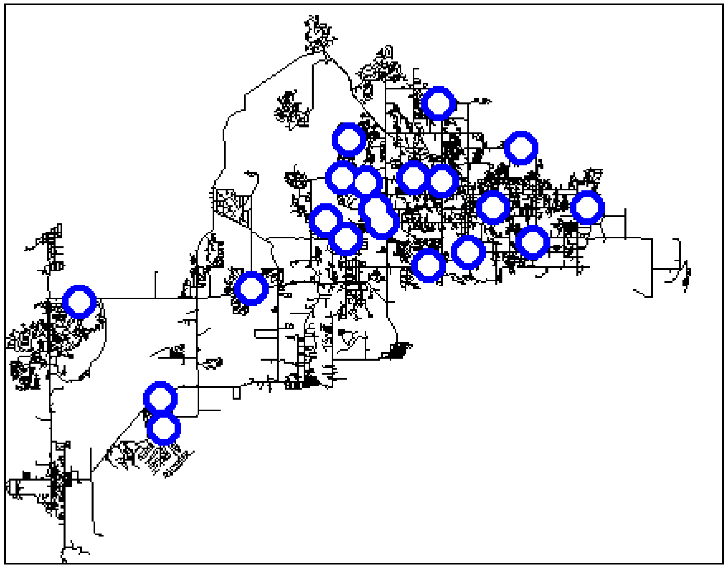
Result of the algorithm

Constant factor ~63%

Optimal solution

Example: Water Networks Competition

- 12,527 junctions
- 3.6 million contamination events

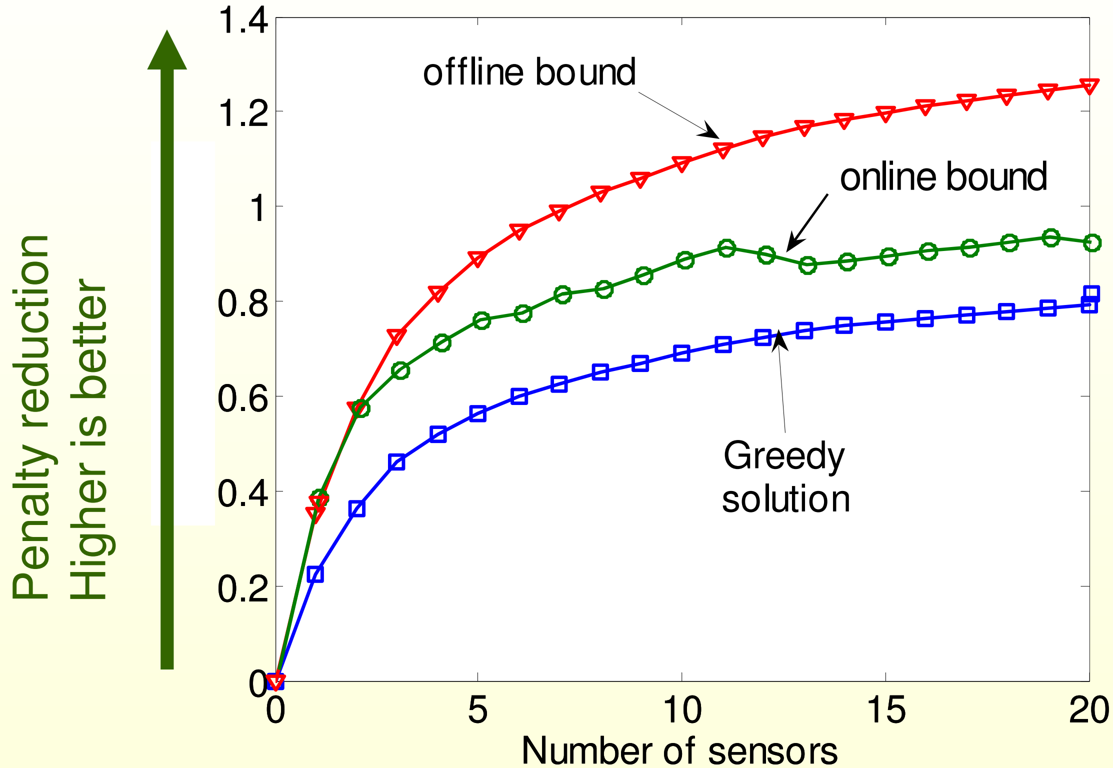


- Place 20 sensors to
  - maximize detection likelihood
  - minimize detection time
  - minimize population affected

in expectation over all possible contaminations

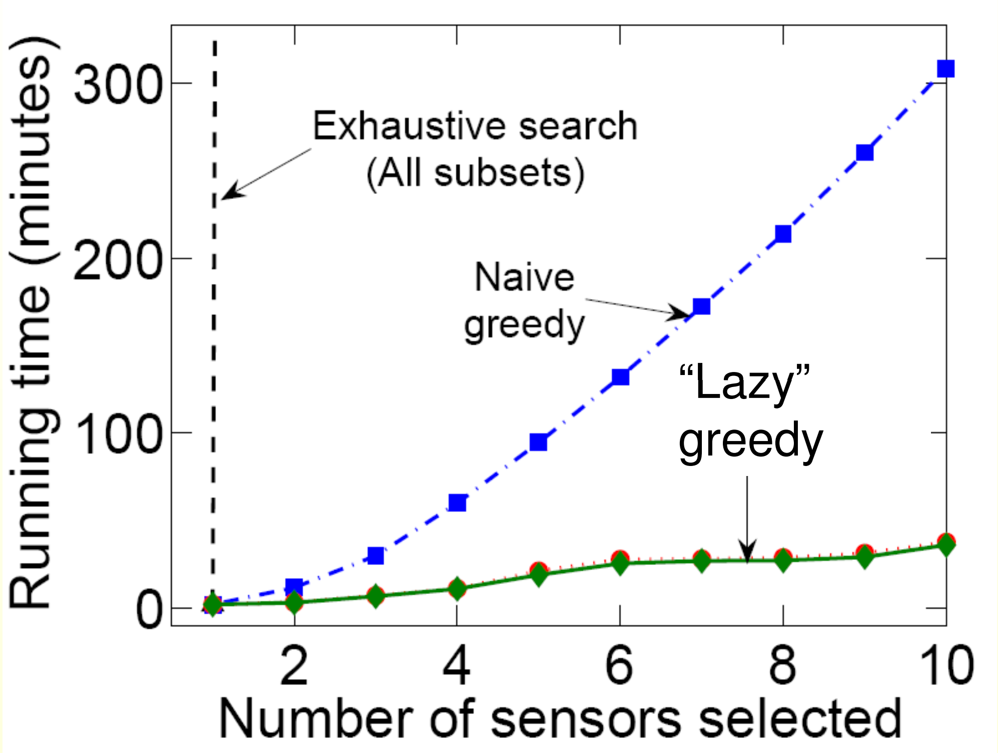
Observation:
All these objectives are submodular! ☺

Bounds on Optimal Solution



- Submodularity allows us to obtain **online** bounds on the performance of **any** algorithm
  - Use to evaluate quality of heuristic (e.g., simulated annealing)
  - Use in branch and bound algorithms

Speeding Up Algorithms



- Submodularity allows to **speed up** algorithms using "**lazy evaluations**"

Results of BWSN [Ostfeld et al]

- Multi-criterion optimization
  - "everybody is a winner"
- [Ostfeld et al '07]: count number of non-dominated solutions

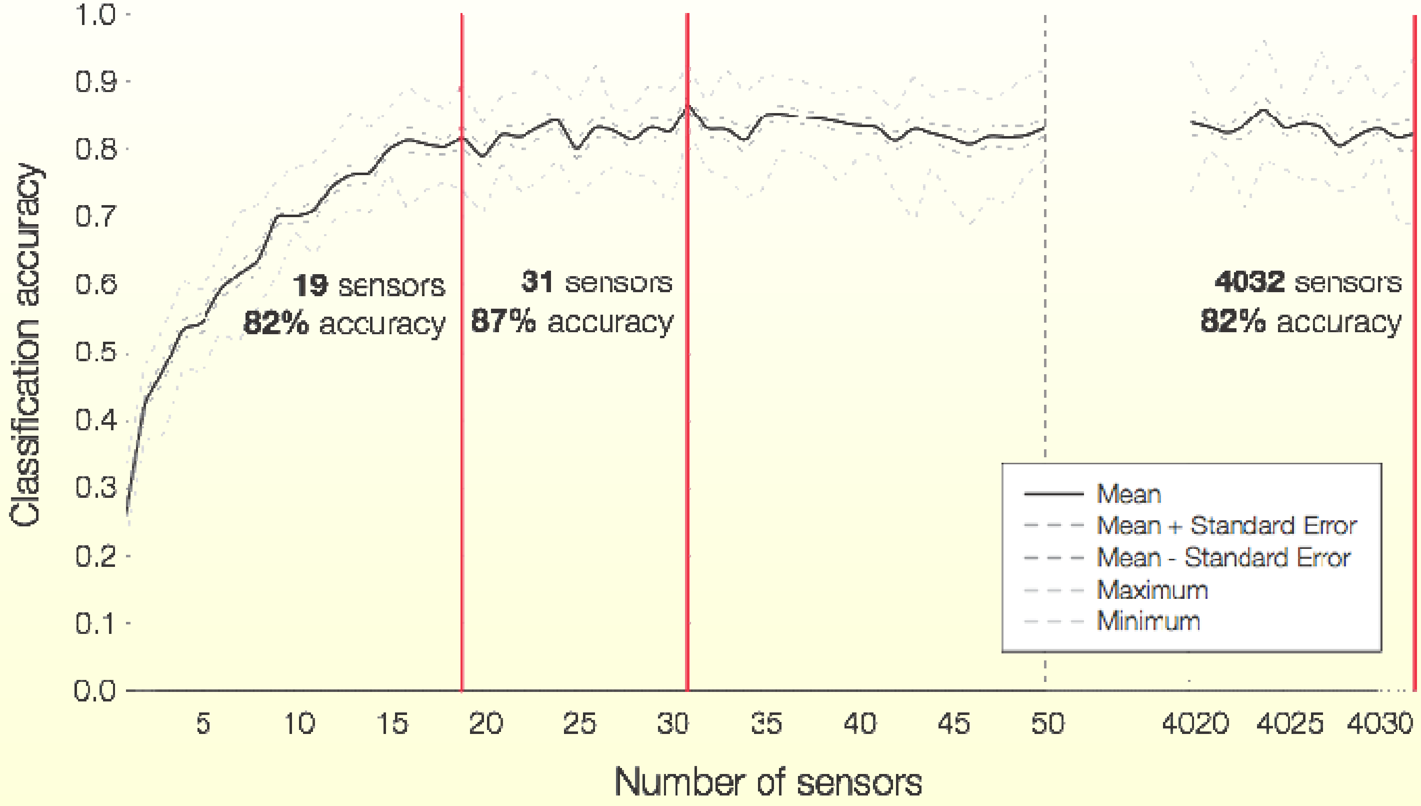
Author	**#non-dom.** (out of 30)
Krause et. al.	26
Berry et. al.	21
Dorini et. al.	20
Wu and Walski	19
Ostfeld and Salomons	14
Propato and Piller	12
Eliades and Polycarpou	11
Huang et. al.	7
Guan et. al.	4
Ghimire and Barkdoll	3
Trachtman	2
Gueli	2
Preis and Ostfeld	1

Smart Chair Prototype

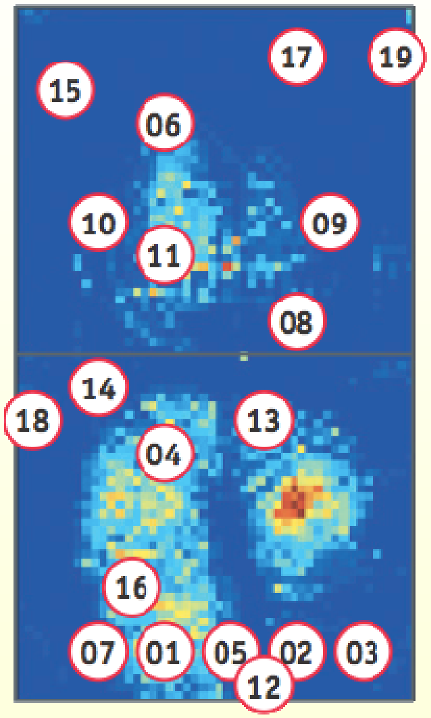
[Mutlu, Krause, Forlizzi, Guestrin, Hodgins ‘07]

- 20 naive participants
- Robust generalization
  Up to 87% accuracy in classifying 10 postures with new subjects
- Low cost
  Using 19 pressure sensors instead of 4032. Reducing sensor cost from $3K to ~$100
- 10 Hz real-time performance
  On a standard desktop computer

Near-Optimal Placement on Chair



Selected placements



Near-optimal placements shown on high-resolution pressure map

Near-optimall placements on the prototype

Summary thus far: **Optimizing information quality**

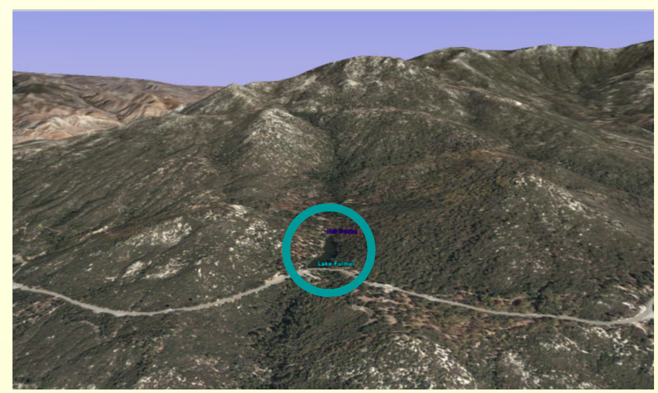
- **Greedy** algorithm for sensor placement is provably **near optimal**
  - **simple and effective** approach for finding **highly informative** sensor placements
  - **works very well on real data!**

- **Submodular functions are key in placing sensors in order to optimize information ☺**

Wireless Sensor Nodes

- In many applications, nodes use wireless links for communication
- No cables
  - huge cost savings
  - ease of deployment
  - simpler maintenance
  - enables mobility
  - allows us to go where no one has gone before... ☺

CMU's Intelligent Workplace



Lake Fulmor

PART II:

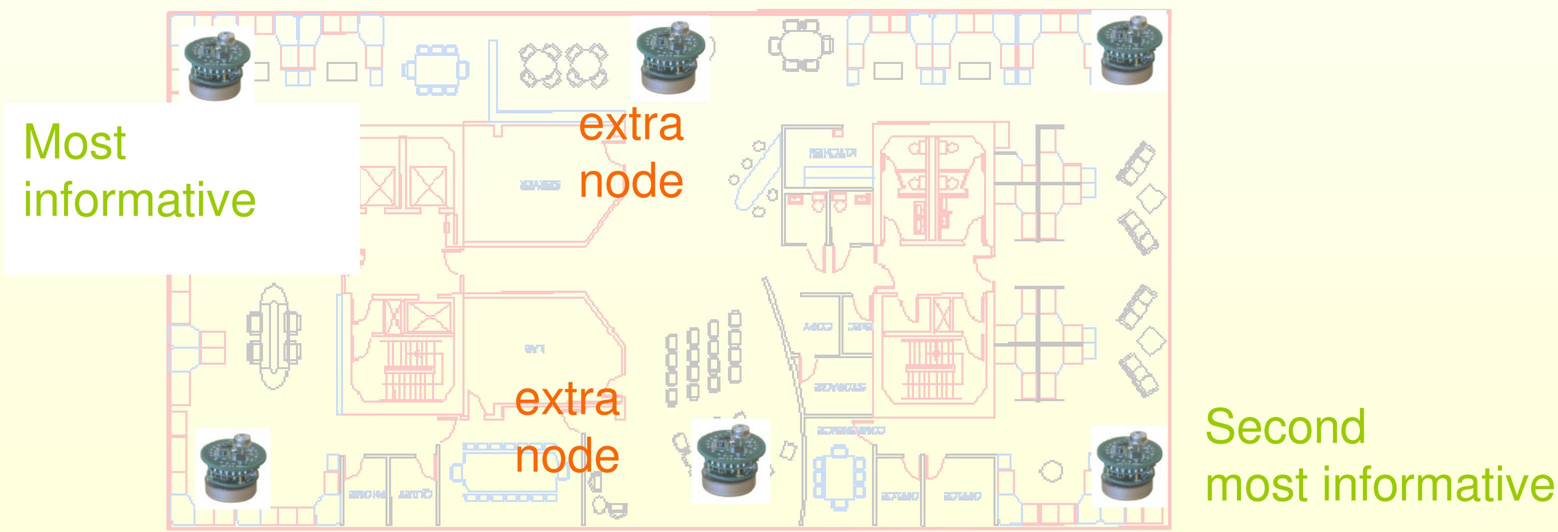
Near-Optimal Sensor Placement:

Maximizing Information while
Minimizing Communication Cost

[Krause, Guestrin, Gupta, Kleinberg, '06]

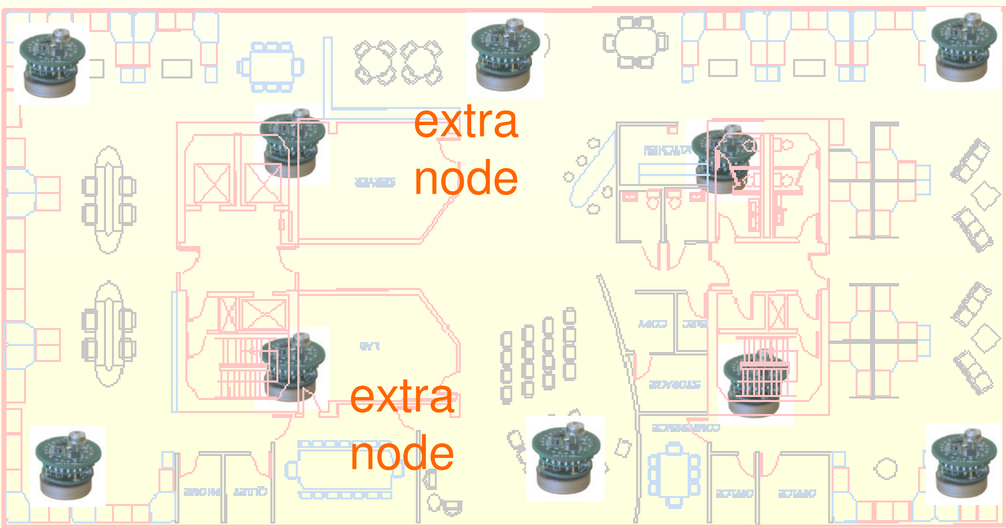
What About the Greedy Algorithm When Communication Matters?

- **Greedy-connect**: a simple heuristic
  - greedy obtains near-optimal information quality
    - usually selects nodes that are "far apart"
  - add **extra nodes** to guarantee communication



What if we used simple greedy algorithm when comm. matters?

- **Greedy-connect**: a simple heuristic
  - greedy obtains near-optimal information quality
    - usually selects nodes that are "far apart"
  - add **extra nodes** to guarantee communication

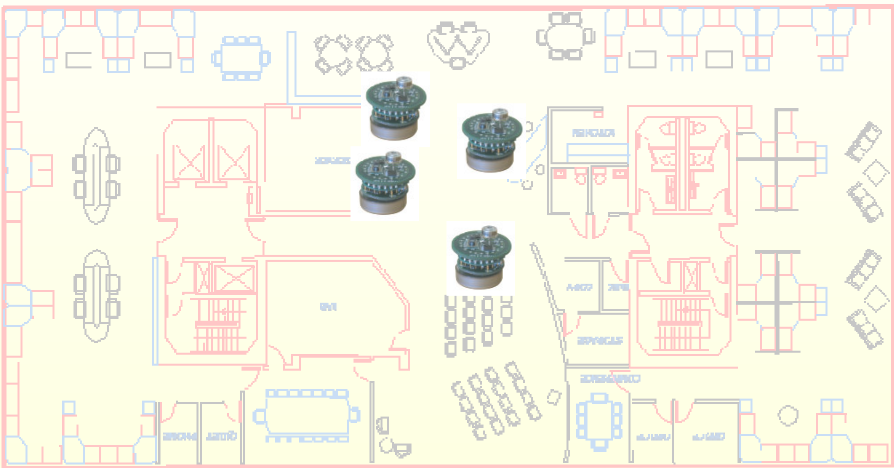


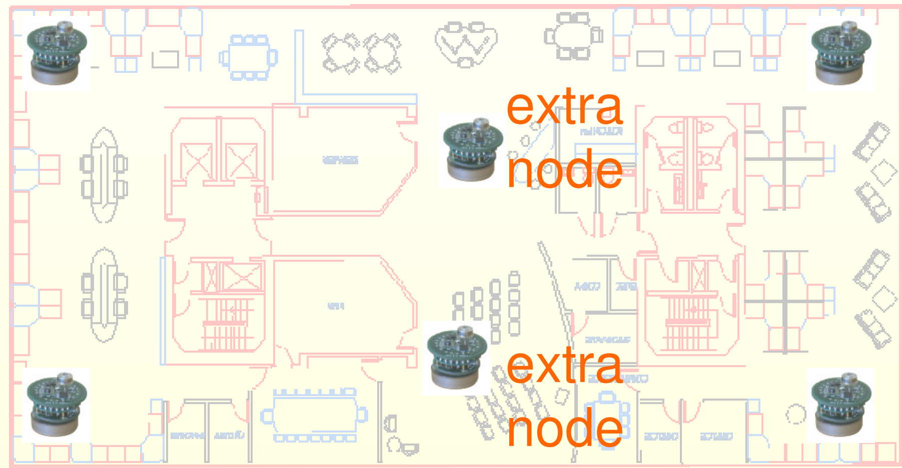
But, there might be a only slightly less informative, but much less costly solution!

Trade-off: Information vs. Communication Cost

The "closer" the sensors:

The "farther" the sensors:





efficient communication! ☺

better information quality! ☺

worse information quality! ☹

worse communication! ☹

The pSPIEL Algorithm

[Krause, G., Gupta, Kleinberg '06]

- ***pSPIEL***: Efficient, randomized algorithm

  (**p**added **S**ensor **P**lacements at **I**nformative and cost-**E**ffective **L**ocations)
- In expectation, **both** information quality and communication cost will be **close to optimum**
- Built a system on **real sensor nodes** for sensor placement using pSPIEL
- Evaluated the method on **real-world sensor placement** problems

The Placement Problem

- Let **V** be set of locations to choose from
- For subset of locations **A** $\subseteq$ **V**, let
  - *I(A)* be **information quality** and
  - *C(A)* be **communication cost** of placement A

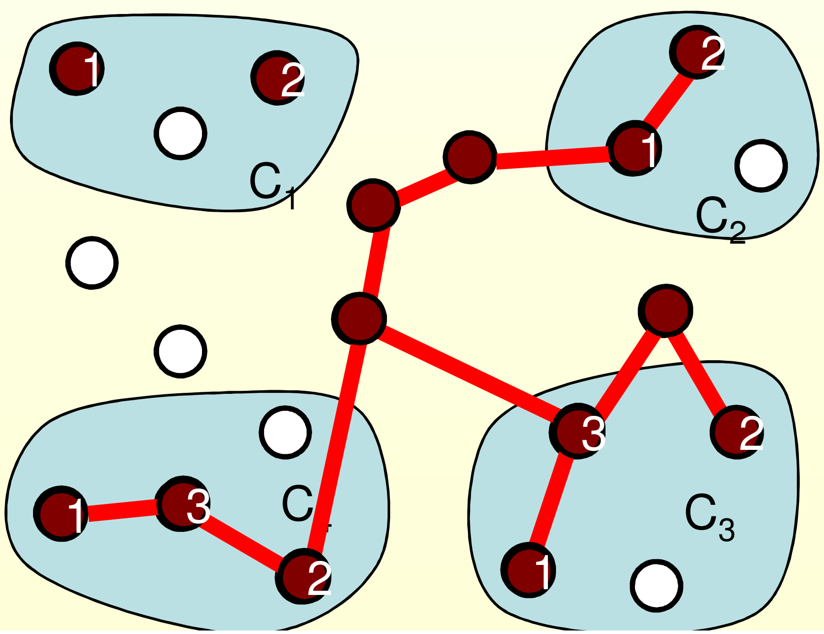
- Want to optimize

  min *C(A)* subject to *I(A)* $\geq$ *Q*

- Q>0 is information quota

Our approach: ***pSPIEL***

- **Decompose** sensing region into small, well-separated clusters
- Solve cardinality constrained problem **per cluster**
- **Combine** solutions using k-MST algorithm



Guarantees for Sensor Placement

Theorem:
pSPIEL finds a tree T with

info. quality $I(T) \geq \Omega(1)\ OPT_{quality,}$

comm. cost $C(T) \leq O(\log |V|)\ OPT_{cost}$

const. factor approx. info.

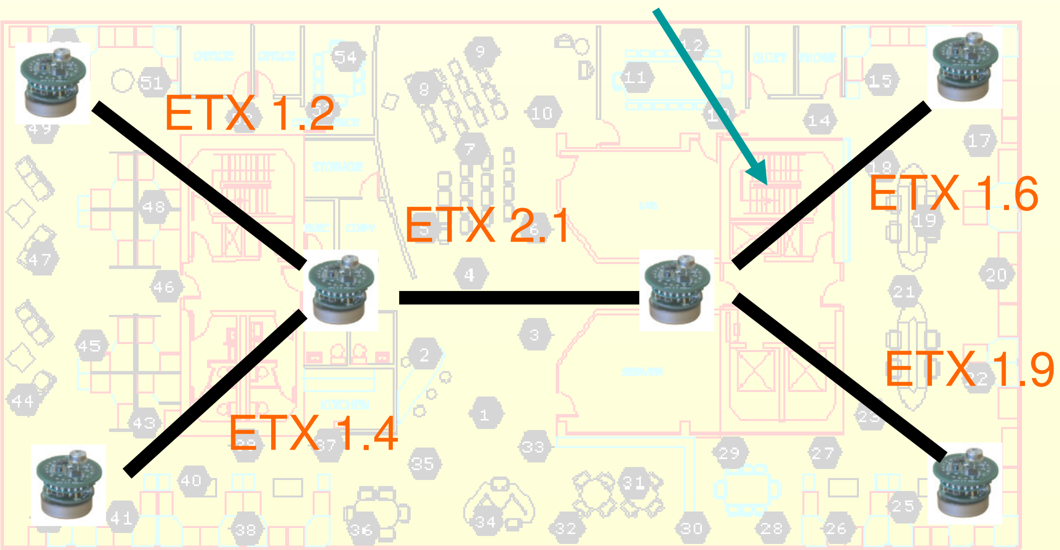
log factor approx. comm. cost

Summary of the Approach

- Use small, short-term “bootstrap” deployment to **collect** some **data** or obtain **expert model** (e.g., from the EPA)
- **Learn/Compute models** for information quality and communication cost
- **Optimize tradeoff** between information quality and communication cost using ***pSPIEL***
- **Deploy sensors**
- If desired, collect more data and continue with step 2

Communication Cost: a Simple Definition

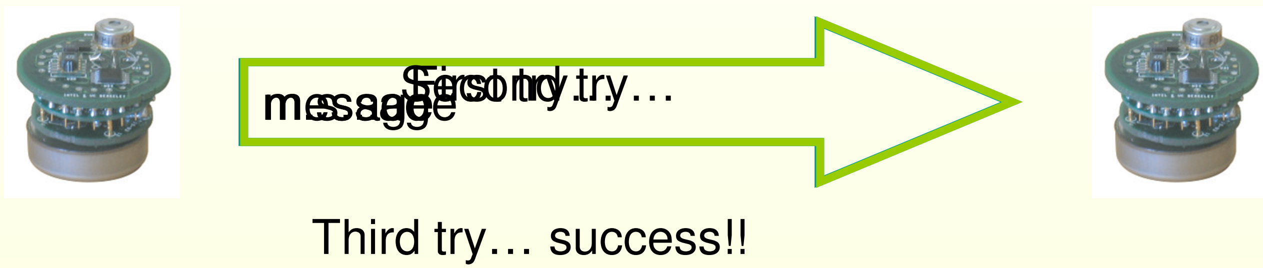
- Message loss requires retransmission
- This depletes the sensor's battery quickly
- Communication cost for two sensors means **expected number of transmissions** (ETX)
- Communication cost for placement is sum of all ETXs along routing tree



Total cost = 8.2

How to Compute ETX

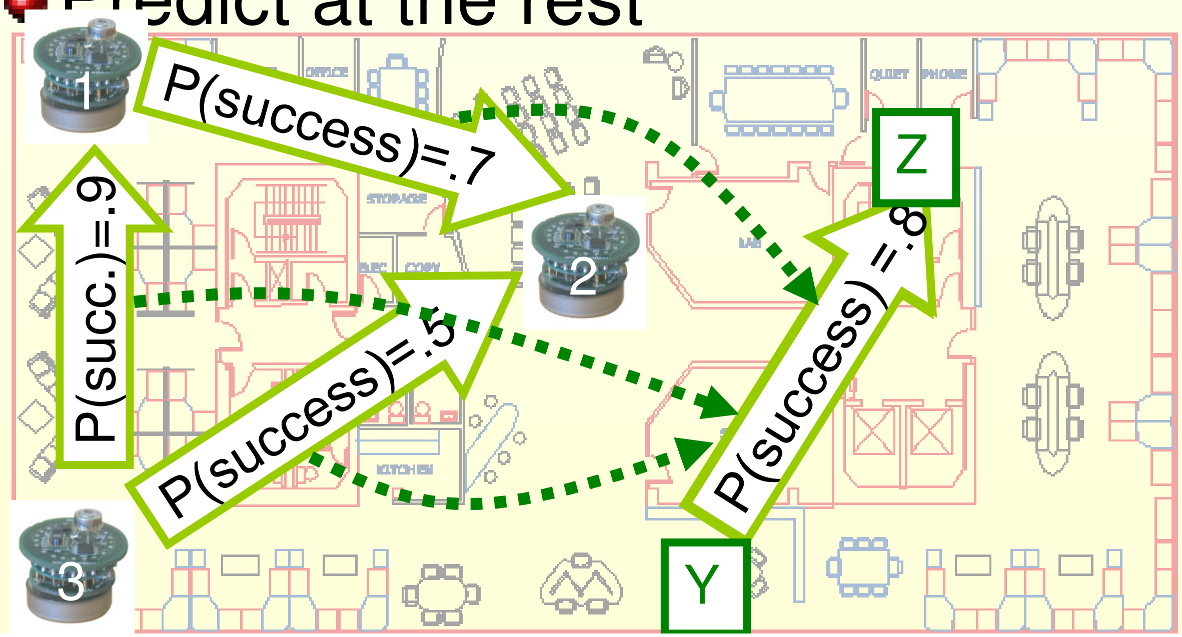
- Pick two sensors and collect transmission logs



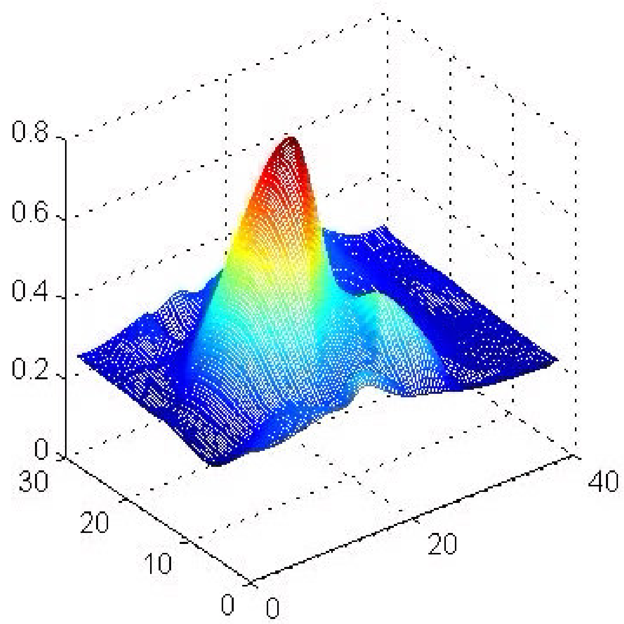
- If we do this long enough, we estimate e.g. P(success) = 1/3, and hence ETX = $1/_{1/3}$ = 3

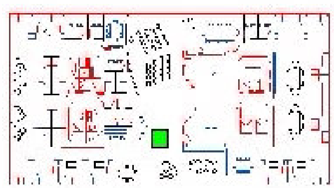
Predicting Communication Cost

- Must answer: **If I were to place sensors at Y and Z, what would the link quality be?**
- A **regression problem**:
  - Measure link qualities for some locations
  - Predict at the rest



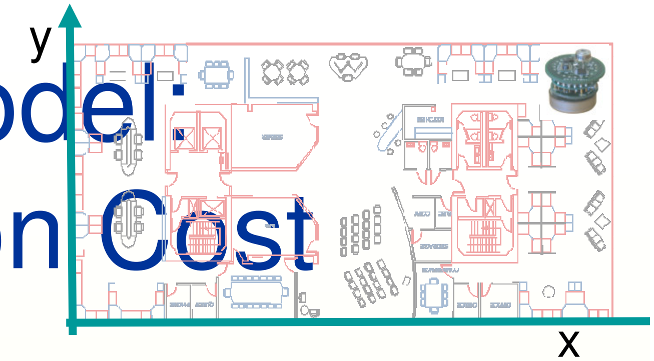
Interpolated Real Link Quality Data



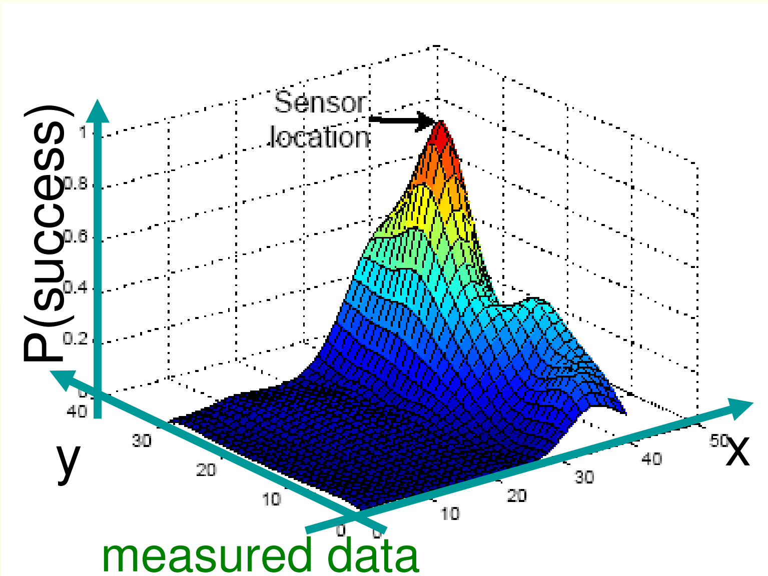


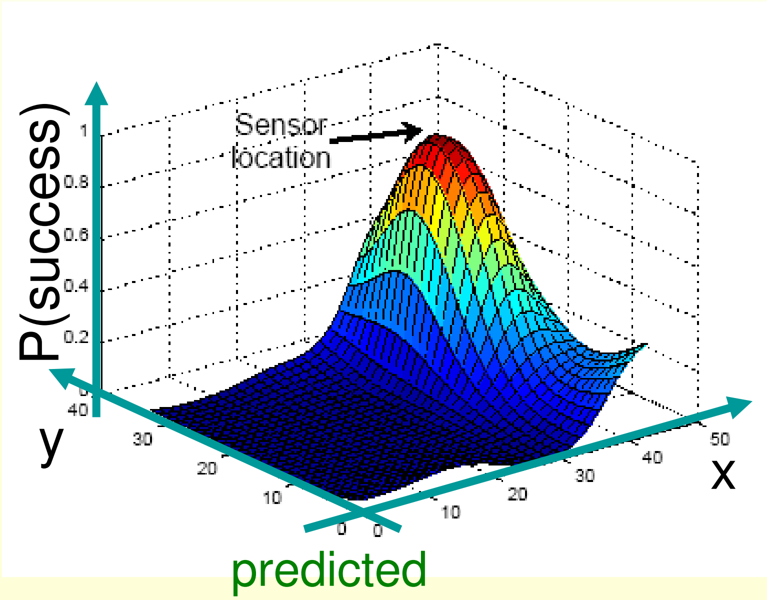
- Complex surface
- Looks very little like $1/r^2$
  - a typical assumption
- Observe
  - “corridor” effect
  - external interference
  - …
- How can we model this phenomena?

Learned Model: Communication Cost



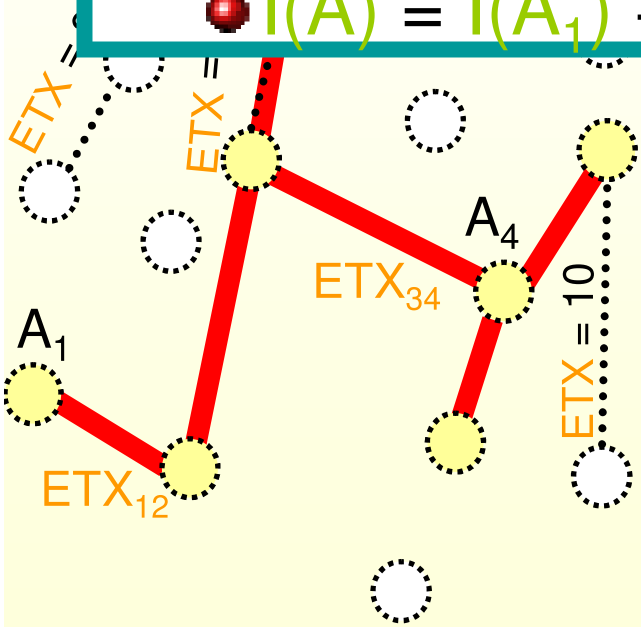
- Complex data
- Represent using probabilistic model
  - extension of non-parametric Gaussian process used for temperature
  - main difference: link qualities in [0,1]





Minimizing Communication Cost While Maximizing Information Quality

- First: **simplified** case, where each sensor provides **independent information**:
  - $I(A) = I(A_1) + I(A_2) + I(A_3) + I(A_4) + \ldots$



$\min_A C(A)$

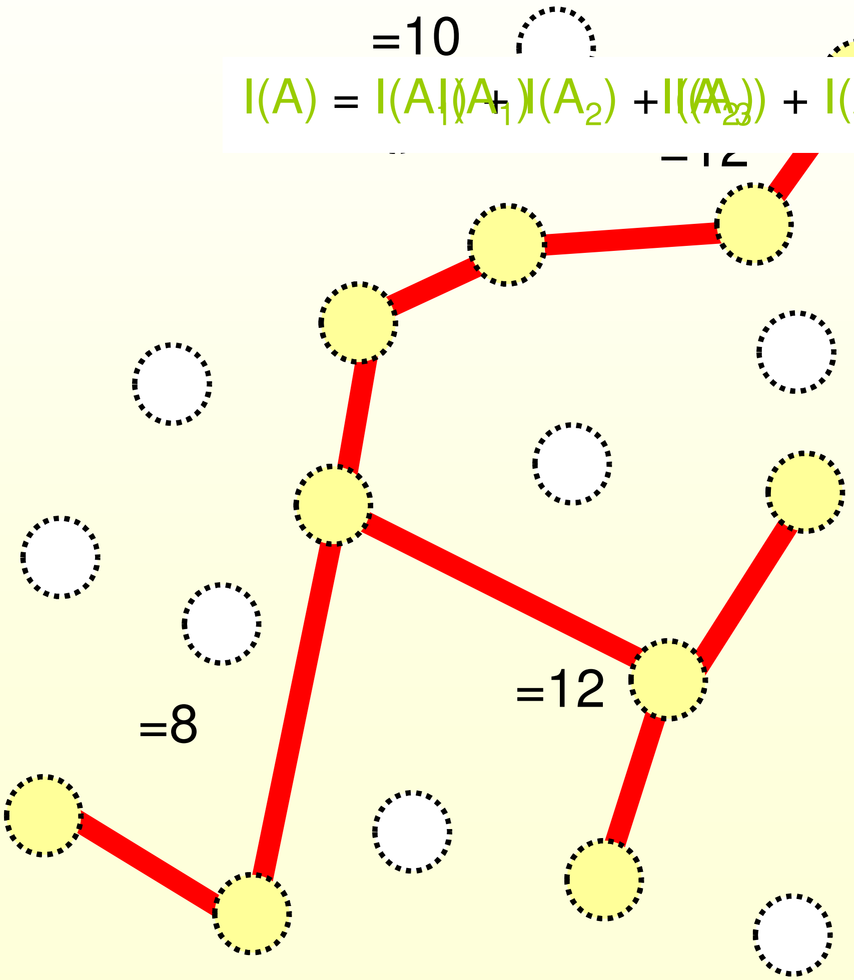
$C(A) =$

locations are informative:

$I(A) \geq Q$ + + + …

$I(A) = I($ $\cup$ $\cup$ $\cup \ldots)$

Quota Minimum Steiner Tree (Q-MST) Problem



$I(A) = I(A_1) + I(A_2) + I(A_3) + I(A_4) \ldots$ $I(A_4)$

Problem:

… reward $I(A_i)$

… ee that collects that least Q reward:

$+ \quad + \quad + \ldots \geq Q$

Perhaps could use to solve our problem!!! ☺

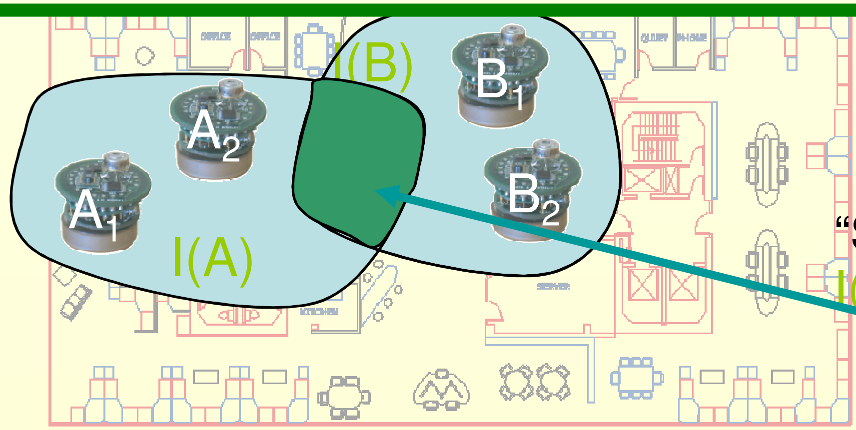
NP-hard… ☹

but very well studied [Blum, Garg, …]

Constant factor 2 approximation algorithm available! ☺

Getting Closer…

- Q-MST algorithm works if I(A) is *modular*, i.e., if *A* and *B* disjoint, $I(A \cup B)=I(A)+I(B)$
- **Makes no sense** for sensor placement!
  - Close by sensors are **not** independent
- For sensor placement, *I* is *submodular* $I(A \cup B) \leq I(A)+I(B)$



"Sensing regions" overlap, $I(A \cup B) < I(A) + I(B)$

Must Solve a New Problem

insight: our sensor problem has additional structure! ☺

- Want to optimize
  - min *C(A)* subject to *I(A)* $>$ *Q*

if sensors provide
independent information
$I(A) = I(A_1) + I(A_2) + I(A_3) + \ldots$

→

a modular problem
solve with Q-MST ☺
but info not independent ☹

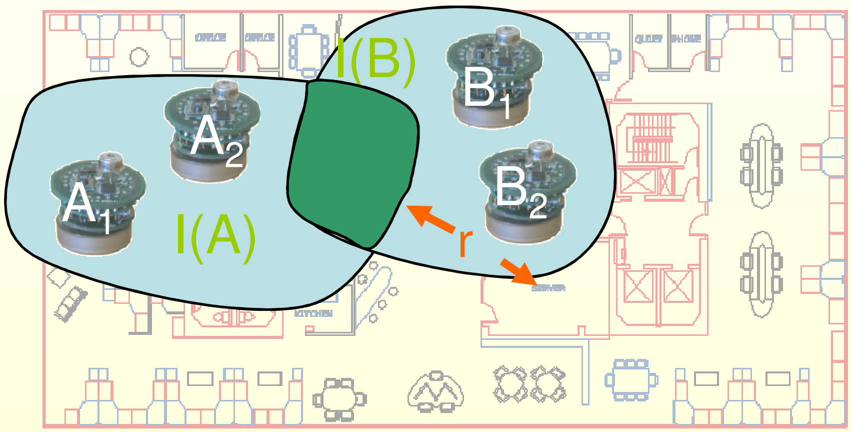
sensors provide
submodular information
$I(A_1 \cup A_2) \leq I(A_1) + I(A_2)$

→

a new open problem!
submodular steiner tree
strictly harder than Q-MST
generalizes existing problems
e.g., group steiner

Independence via Separability

- If A, B are placements closeby, then $I(A \cup B) < I(A) + I(B)$
- If A, B are placements, at least r apart, then $I(A \cup B) \approx I(A) + I(B)$



- Sensors that are **far apart** are **approximately independent**
  - We showed **locality is empirically valid**

Outline: **pSPIEL**

submodular
steiner tree
with locality
$I(A_1 \cup A_2) \leq I(A_1) + I(A_2)$

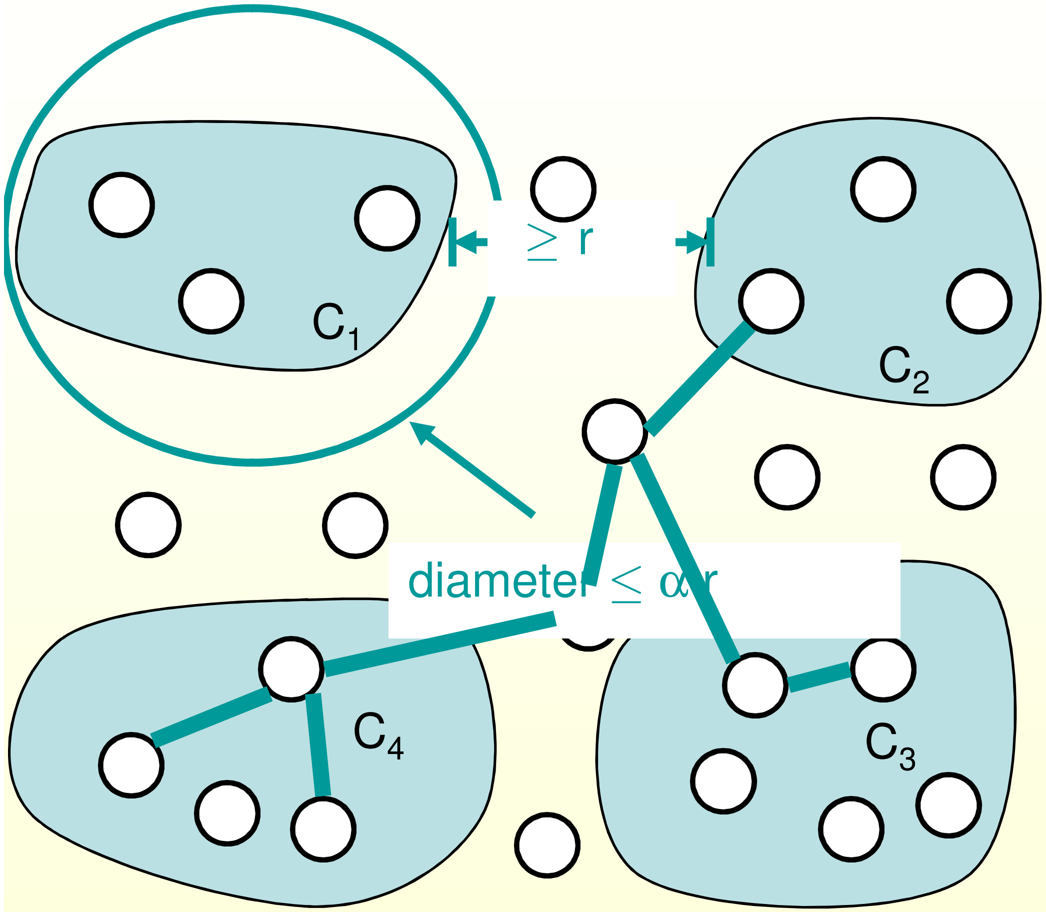
approximate by a modular problem:
for nodes A
sum of rewards A $\approx$ I(A)

solve modular approximation
with Q-MST

use off-the-shelf
Q-MST solver

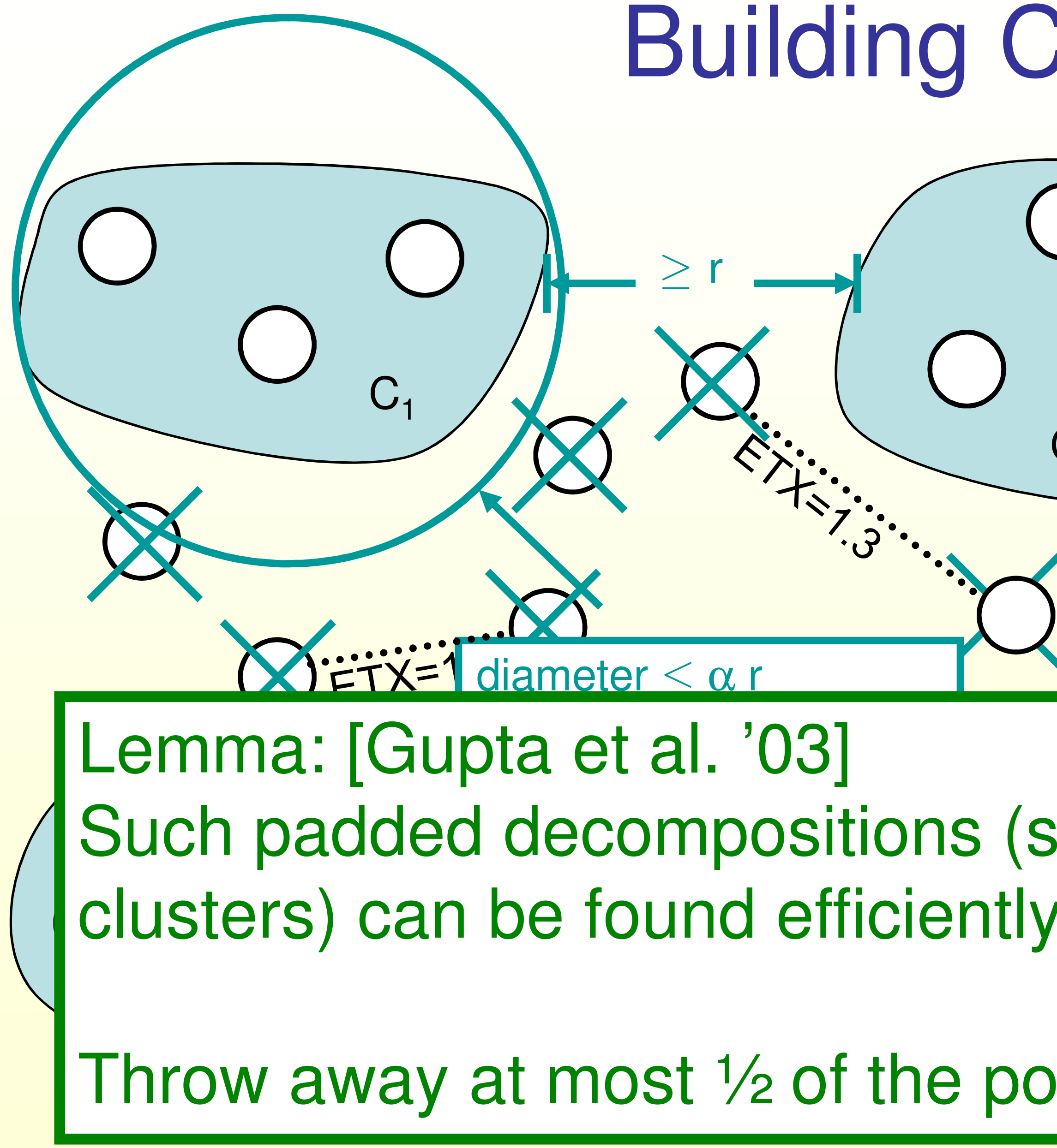
obtain solution of
original problem
(prove it's good)

Overview: **pSPIEL**



- Build **clusters** over possible sensor locations
  - well-separated
  - not too large
- **Throw away** locations **not in clusters**
- **Information additive between clusters! ☺**
  - **separability or locality!!!**
- Use Q-MST to decide **which nodes** to use from **each cluster** and **how to connect** them

pSPIEL: Step 1
Building Clusters

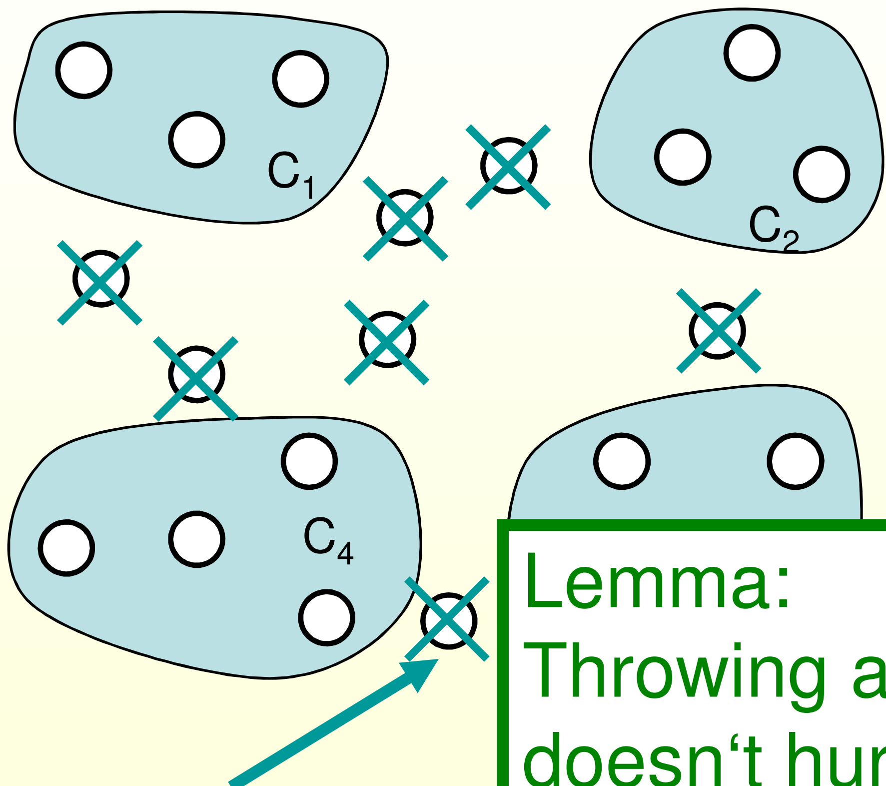


- **Throw away** locations **not in clusters → not allowed to place sensors there**

Lemma: [Gupta et al. '03]
Such padded decompositions (small, well-separated clusters) can be found efficiently

Throw away at most ½ of the possible locations

***pSPIEL*: Step 2 Throwing Away Nodes**



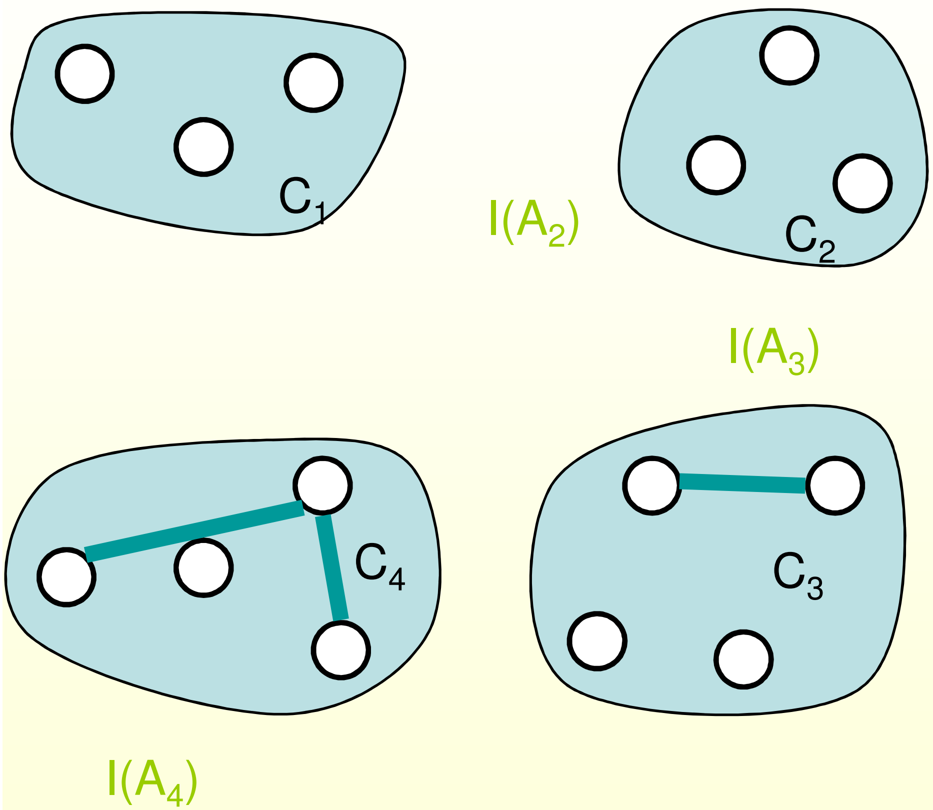
was this one too good to let go?

- We throw out unpadded possible sensor locations
  - at most ½ of all locations

Lemma:
Throwing away non-clustered locations doesn't hurt information quality too much in expectation! ☺

proof uses submodularity and an "uncrossing" procedure

pSPIEL: Well-Separated Small Clusters



- Under **separability**, clusters are mostly independent**!**

$I(A_2 \cup A_3 \cup A_4) \approx I(A_2)+I(A_3)+I(A_4)$

- Diameters are **small**, we **don't care** about comm. cost **within cluster**!
  - use greedy-connect in cluster

Outline: **pSPIEL**

submodular
steiner tree
with locality
$I(A_1 \cup A_2) \leq I(A_1) + I(A_2)$

approximate by a modular problem (MAG):
for nodes A
sum of rewards A $\approx$ I(A)

solve modular approximation
with Q-MST

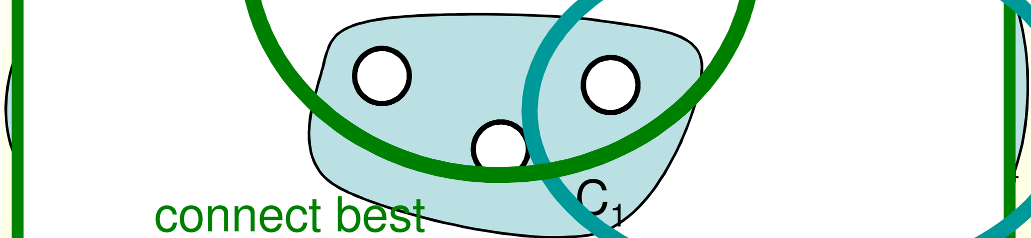
use off-the-shelf
Q-MST solver

obtain solution of
original problem
(prove it's good)

pSPIEL: Step 3
Modular Approximation Graph

How do we define weights and rewards?

Consider a cluster:



connect best nodes between clusters

greedy chains in clusters

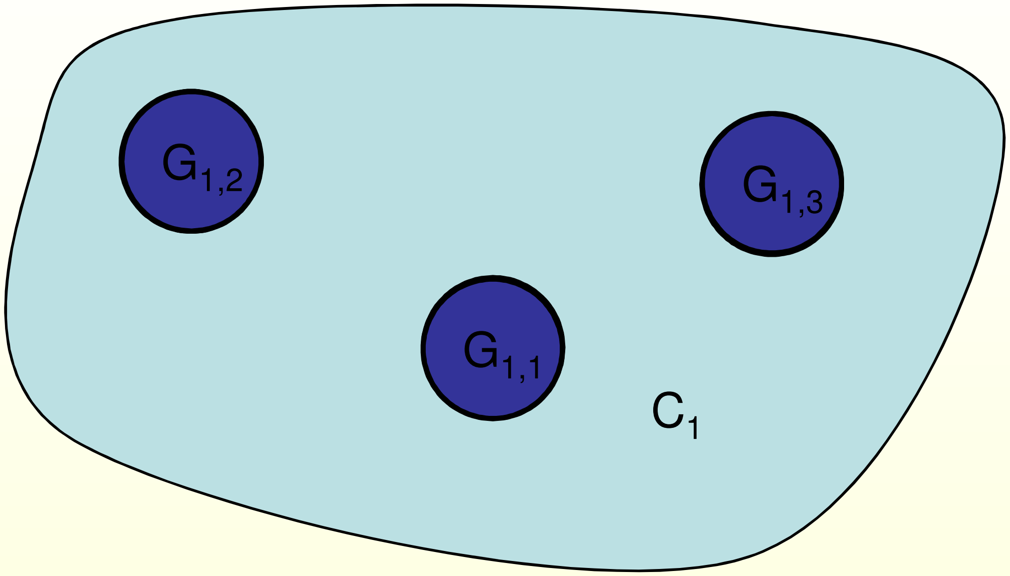
most importantly, additive rewards:

- Order nodes in "order of importance"
  - greedy defines order
- Build a modular approximation graph (MAG)
  - edge weights and node rewards → solution in MAG ≈ solution of original problem

Info: $I(G_{2,1} \cup G_{2,2} \cup G_{3,1} \cup G_{4,1} \cup G_{4,2}) \approx$ + + + +

Cost: $C(G_{2,1} \cup G_{2,2} \cup G_{3,1} \cup G_{4,1} \cup G_{4,2}) \approx$ + + +

pSPIEL: Greedy Algorithm within Cluster



Want to find sets with highest information quality in cluster

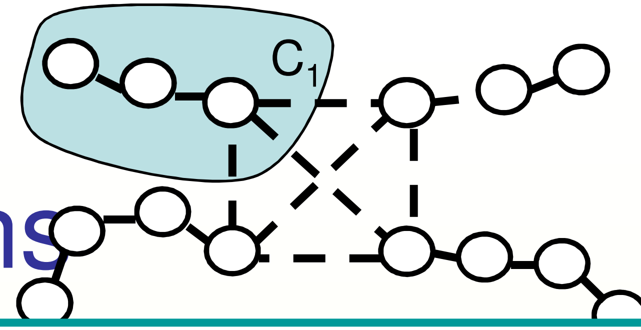
Run the greedy algorithm in every cluster, i.e., add most informative element

Again Theorem:

Greedy subsets of size k are at most a constant factor (1-1/e) worse in Information Quality than the best such set!

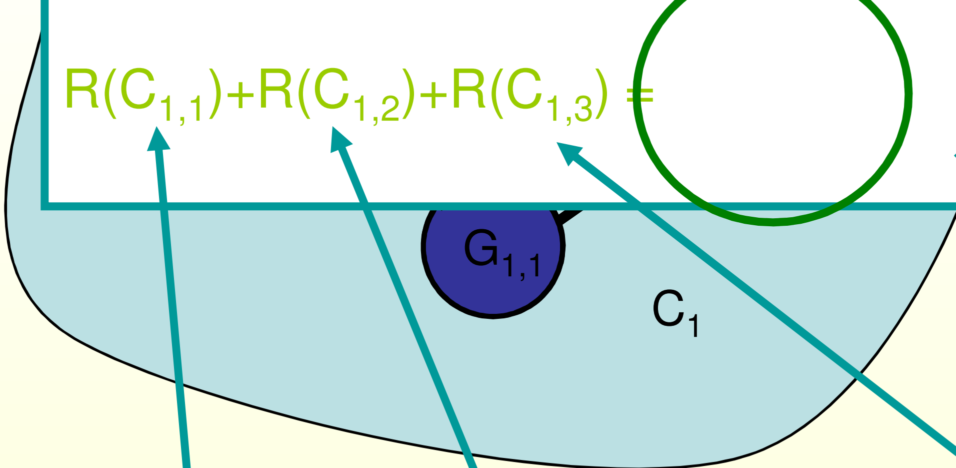
- e.g., pick nodes $G_{1,1}$ & $G_{1,2}$ 63% of optimal 2 nodes
- in practice, much better

pSPIEL: Step 3: Greedy Chains



Key property: clusters organized into chain

$R(C_{1,1})+R(C_{1,2})+R(C_{1,3}) =$ + +



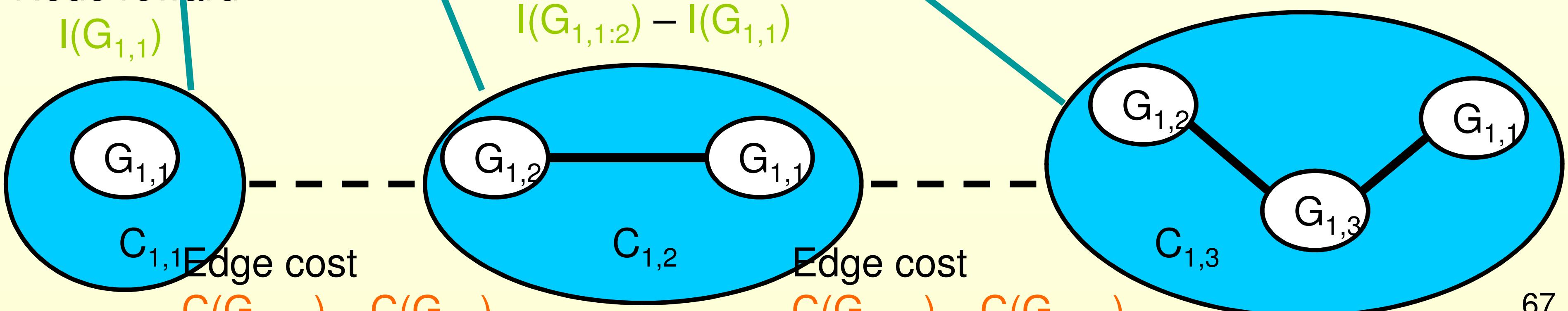
Get a modular functions via telescopic sums

Same trick for communication costs

Node reward $I(G_{1,1})$

Node reward $I(G_{1,1:2}) - I(G_{1,1})$

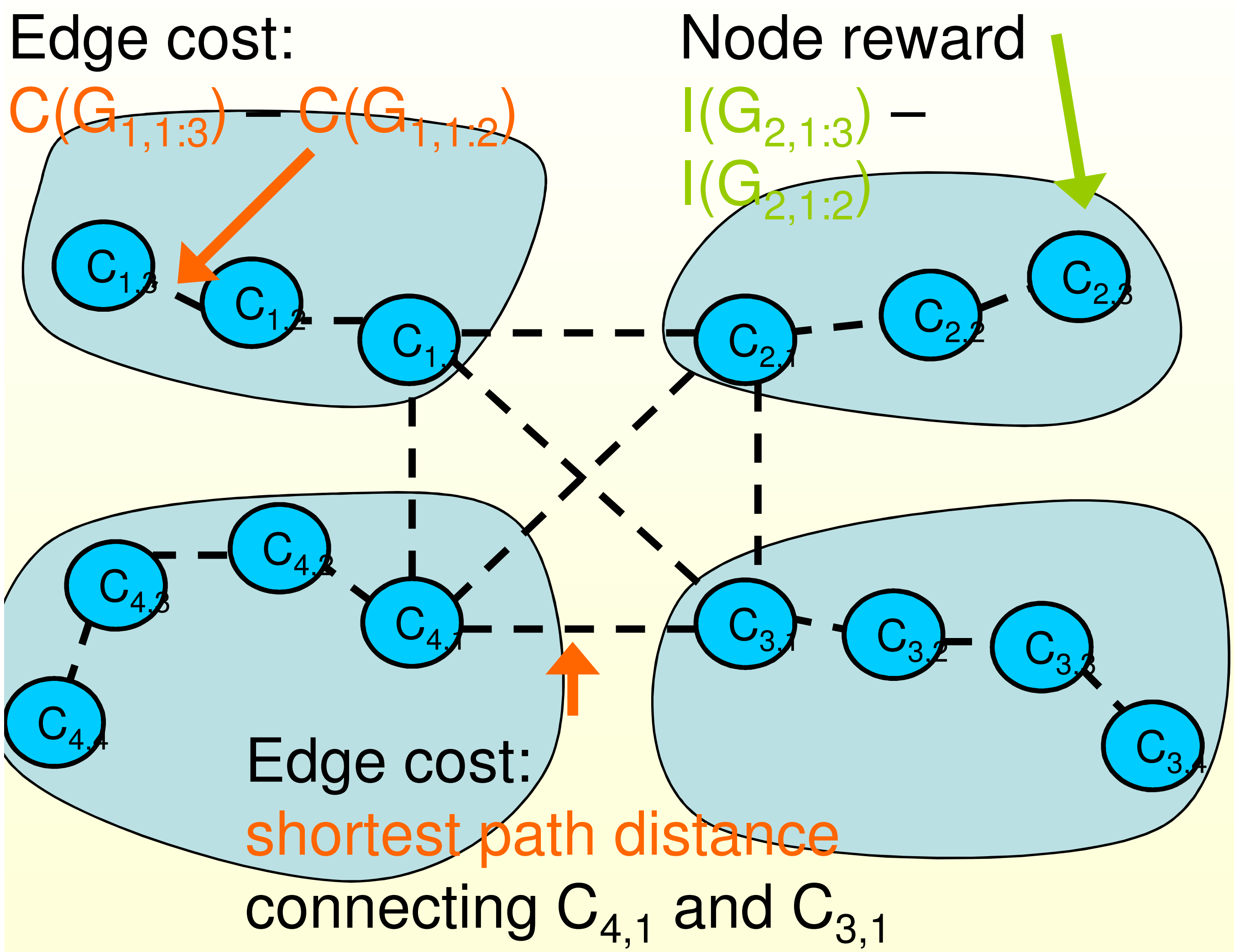
Node reward $I(G_{1,1:3}) - I(G_{1,1:2})$



Edge cost $C(G_{1,1:2}) - C(G_{1,1})$

Edge cost $C(G_{1,1:3}) - C(G_{1,1:2})$

pSPIEL: Modular Approximation Graph



modular approximation graph (MAG) combines chains

approximates original optimization problem! ☺

Outline: **pSPIEL**

submodular

steiner tree

$I(A_1 \cup A_2) \leq I(A_1) + I(A_2)$

approximate by a modular problem (MAG):

for nodes A

sum of rewards A $\approx$ I(A)

solve modular approximation

with Q-MST

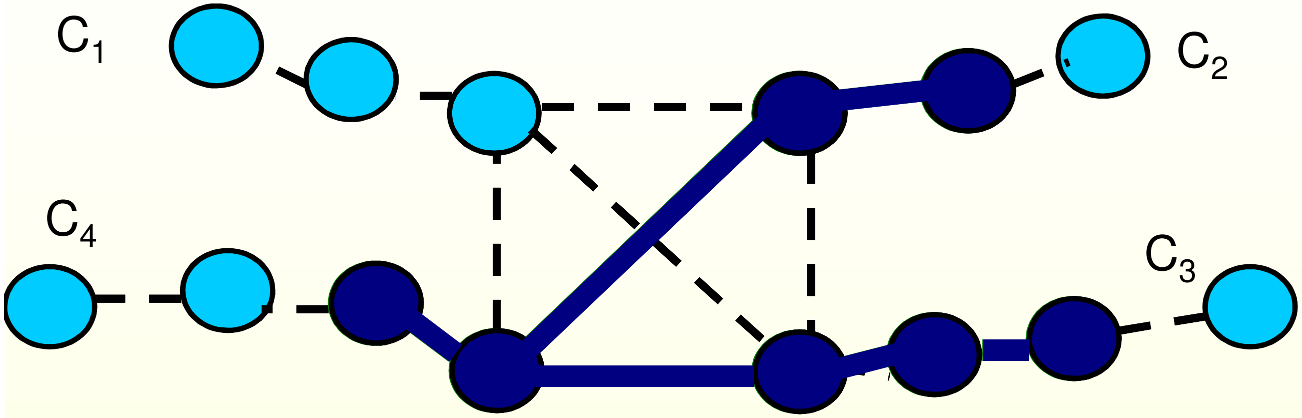
use off-the-shelf Q-MST solver

obtain solution of

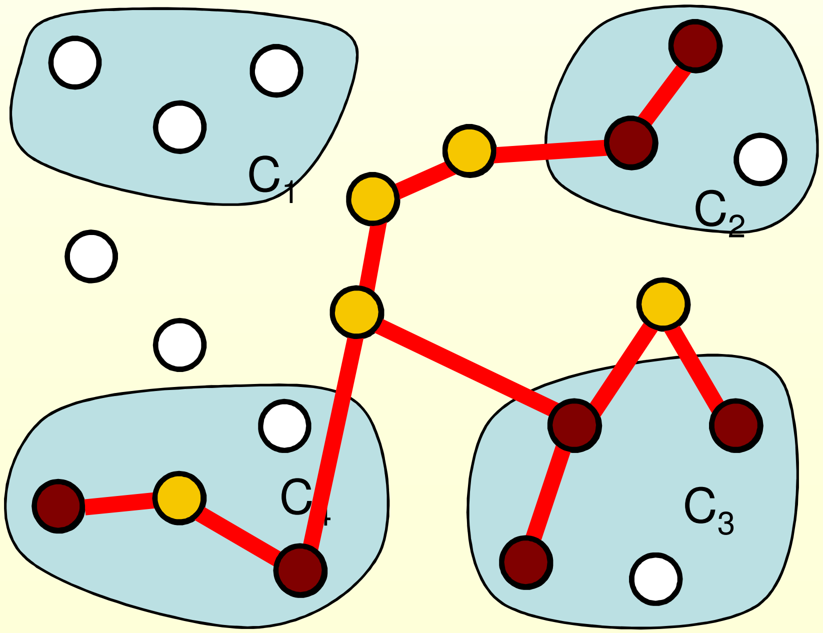
original problem

(prove it's good)

pSPIEL: Using Q-MST



tree in MAG → solution in original graph



Q-MST on MAG → solution to original problem! ☺

Outline: **pSPIEL**

submodular
steiner tree
$I(A_1 \cup A_2) \leq I(A_1) + I(A_2)$

approximate by a modular problem (MAG):
for nodes A
sum of rewards A $\approx$ I(A)

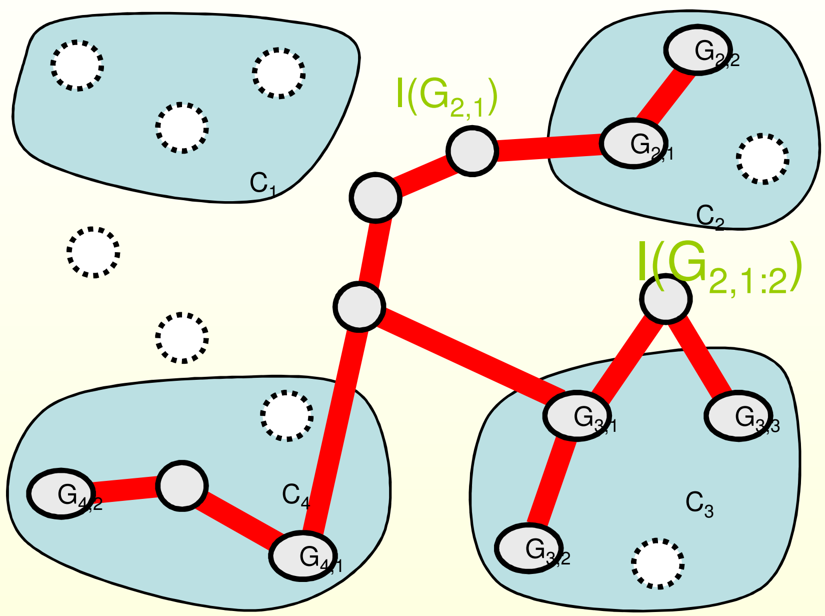
solve modular approximation
with Q-MST

use off-the-shelf
Q-MST solver

obtain solution of
original problem
(prove it's good)

pSPIEL: Solving Sensor Placement Problem

$I(G_{2,1:2}) - I(G_{2,1})$



$I(G_{4,1:2})$ $I(G_{3,1:3})$

Inside a cluster:

$I(G_{2,1:2}) =$ +

Total reward by Q-MST:

$I(G_{2,1:2} \cup G_{3,1:3} \cup G_{4,1:2}) \approx$ locality!

+ + $\geq Q$

- Want to optimize
  - min *C(A)* subject to *I(A)* $\geq$ *Q*

Proof of $\approx$ min *C(A)*:
near opt. Q-MST &
small cluster diameter $\rightarrow$
comm. not too expensive in cluster

Guarantees for Sensor Placement

Theorem:
pSPIEL finds a tree T with

info. quality $I(T) \geq \Omega(1)\ OPT_{quality,}$

comm. cost $C(T) \leq O(r \log |V|)\ OPT_{cost}$

r depends on locality property

const. factor approx. info.

log factor approx. comm. cost

Summary of the Approach

1. Use small, short-term "bootstrap" deployment to **collect** some **data** or **expert model** (e.g., from the EPA)
2. **Learn/Compute models** for information quality and communication cost
3. **Optimize tradeoff** between information quality
and communication cost using ***pSPIEL***
4. **Deploy sensors**
5. If desired, collect more data and continue with step 2

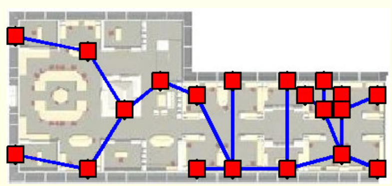
Implemented at CMU…

- Implemented on Tmote Sky motes from MoteIV
- Collect measurement and link information and send to base station
- We can now deploy nodes, learn models and come up with placements!

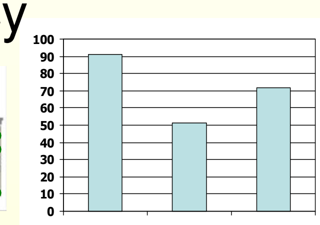
Proof of Concept Study

CMU's Intelligent Workplace

- Learned model from short deployment of 46 sensors at the Intelligent Workplace



accuracy

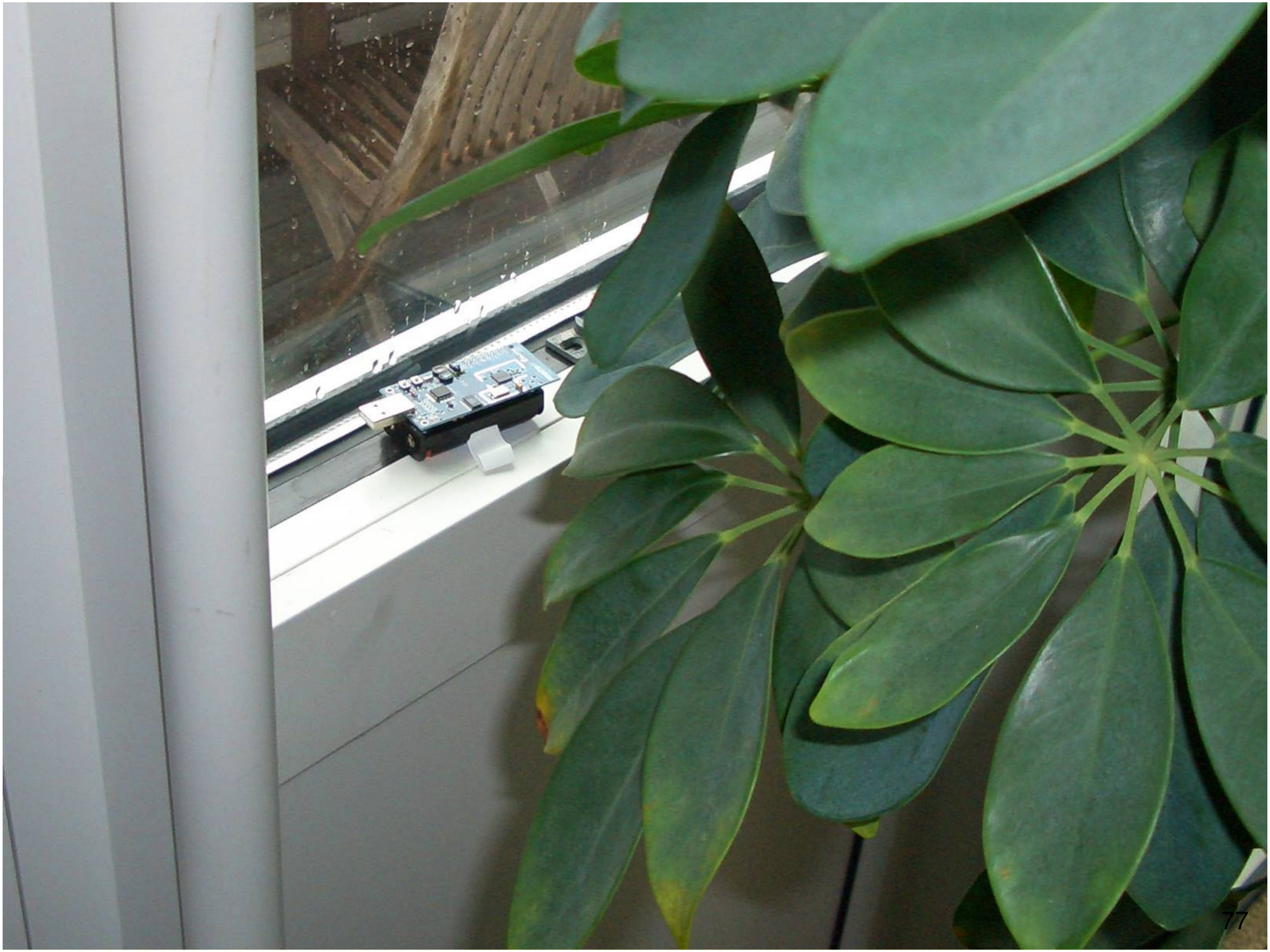


Time

learned GPs for light field & link qualities

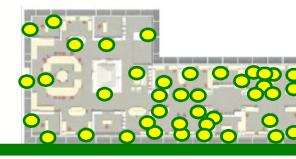
deployed 2 sets of sensors: pSPIEL and manually selected locations

evaluated both deployments on 46 locations

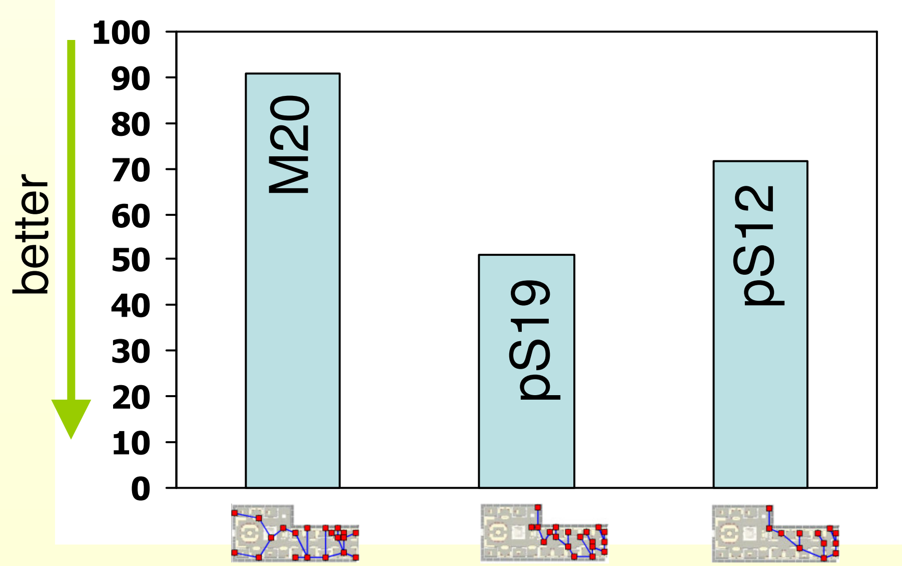


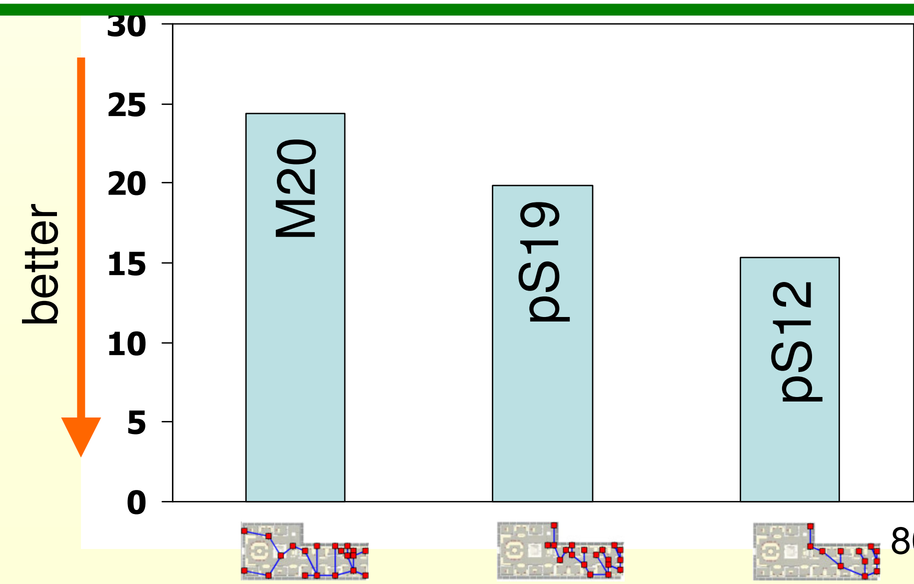
Proof of Concept Study

accuracy on 46 locations



- *pSPIEL* improved solution over intuitive manual placement:
  - 50% better prediction and 20% less comm. cost, or
  - 20% better prediction and 40% less comm. cost
- Poor placements can hurt a lot!
  - Good solution can be unintuitive





Conclusions

- **Unified approach** for deploying wireless sensor networks – **uncertainty is fundamental**
  - models for phenomena, link qualities, and optimization alg.
- ***pSPIEL***: Efficient, randomized algorithm **optimizes tradeoff**: info. quality and comm. cost with **strong guarantees**
  - Proved both info. quality and comm. cost close to optimum
- Built a **complete system** on Tmote Sky motes, deployed sensors, evaluated placements
- *pSPIEL* **significantly outperforms** alternative methods

Conclusions

- **Submodular functions are cool!**
  - Design very effective algorithms, exploit this problem structure
  - Simple setting → Greedy algorithm
  - Constraints (comm., adversary, path,…) → Beyond greedy, but still exploit submodular fns
- **Not just sensor placement**
  - Observation selection, feature selection, experimental design, …
- **Algorithms actually work on real world problems!** :-)
  - Real sensor deployment, water distribution competition, built smart chair,…

The End