Large-Scale Optimistic Adaptive Submodularity

Victor Gabillon[1], Branislav Kveton[2], Zheng Wen[3],
Brian Eriksson[2], & S. Muthukrishnan[3]

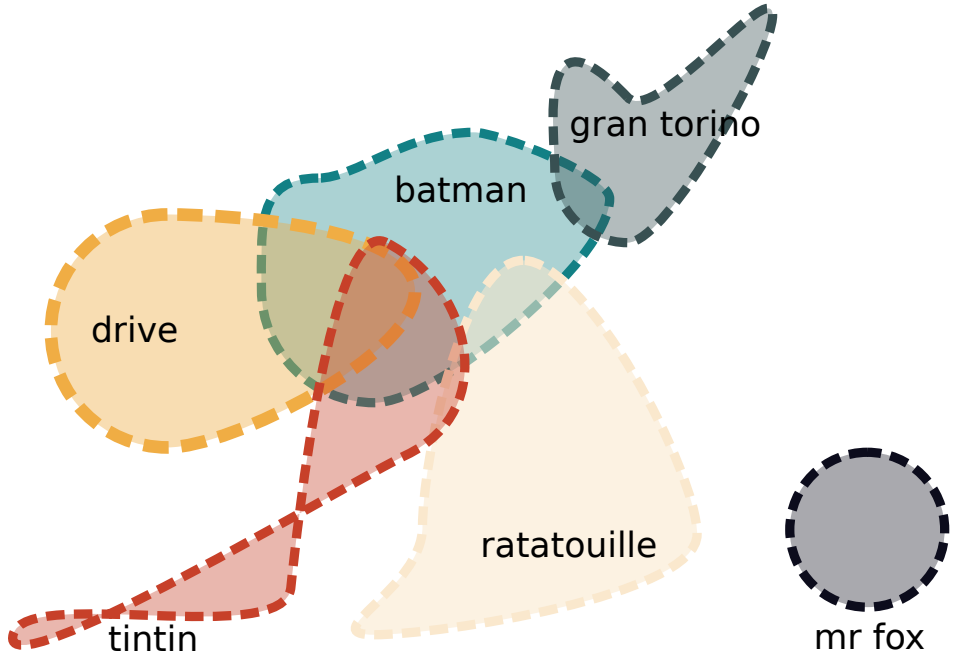
AAAI, Québec, July 29, 2014.

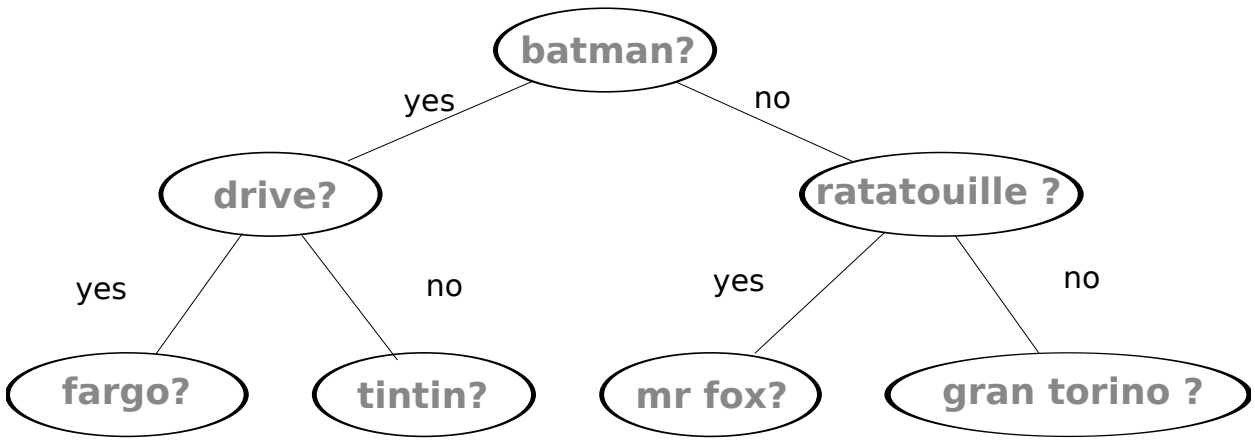
- We address a special **large scale POMDP** problem. The problem is initially combinatorial with parameters that need to be **learned**.

- We use **bandits** and a **submodularity** property to address this learning problem efficiently.

Preference elicitation example

Try to identify as many as possible interesting movies for a user in **K** steps.





The problem

Solving some special Large Scale POMDP problem:

For one user,

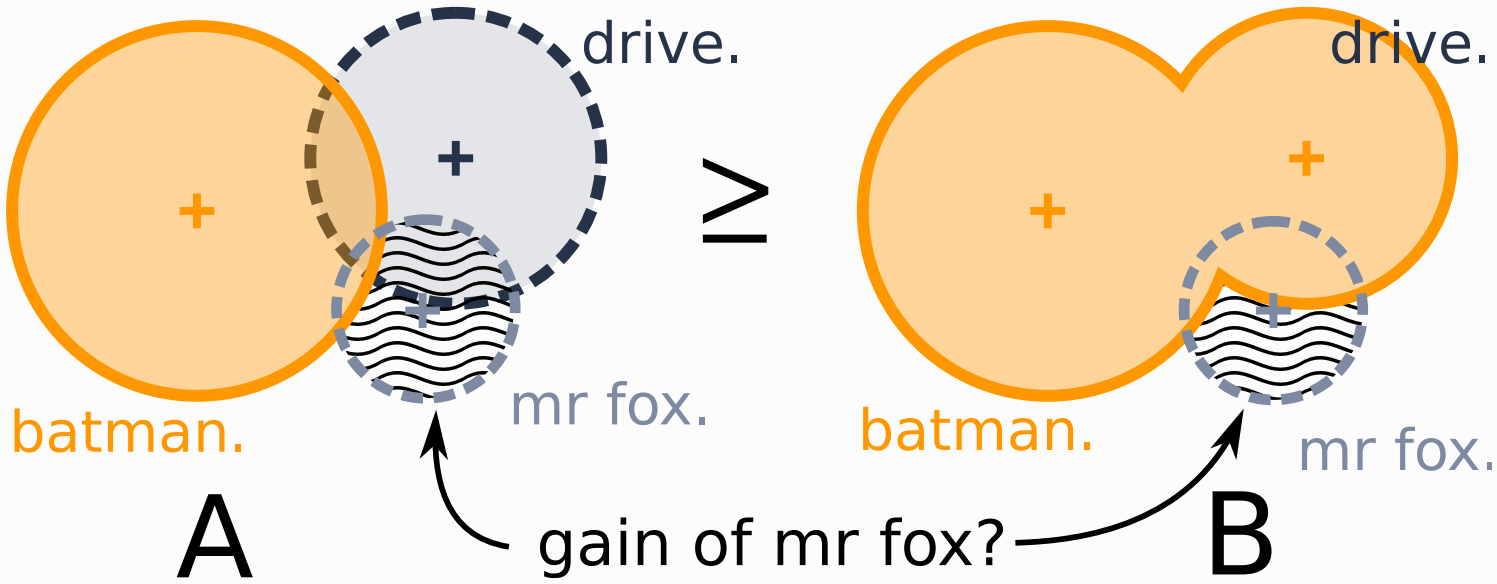
- The **state** is the answer to all the movie-questions: **unobserved!**
- The **state** space is **large** (exponential in the number of movies L). The state is fixed during the game.
- Each **action** (question) reveals one component of the state (yes/no, 0/1).

How to solve this combinatorial problem efficiently?

A greedy approach

Solving a submodular maximization problem:

- The pay-off are **submodular** and decreasing



- **Greedy algorithms** have performance 63% of the optimal in the worst case.
- At each time step just select the greedy action (the one with highest immediate gain g).

A learning problem

- Different users come in a sequence.
- We do not know the distribution generating the answers of the users.
- Need to **learn** this distribution in order to apply the greedy approach

Dealing with large scale: GLM

- [GKWEM 2013] proposed an optimistic submodular learning strategy. The learning time scales with L!!!
- Linear structure assumption using **Generalized Linear Model (GLM)**
- Use linear bandits tool to trade-off between exploration and exploitation in the greedy algorithm.

Notations

- $\phi \in \{-1, 1\}^L$, where $\phi[i]$ is the state of item i.
- ϕ is drawn i.i.d. from $P(\Phi)$.

An **observation** is a vector $\mathbf{y} \in \{-1, 0, 1\}^L$.

$$\begin{aligned} \phi &= (\ -1,\ -1,\ 1,\ -1,\ 1\ ) \leftarrow \text{State} \\ \mathbf{y} &= (\ 0,\ -1,\ 0,\ -1,\ 1\ ) \leftarrow \text{observation} \end{aligned}$$

We denote $\phi \sim \mathbf{y}$.

Model

- Each item (movie) is paired with a **d**-dimensional **feature vector** $\mathbf{x}_i$.

Factorization assumption: Let $p_i(\mathbf{x}_i) = P(\phi[i] = 1)$

$$P(\Phi = \phi) = \prod_{i=1}^{L} p_i(\mathbf{x}_i)^{\mathbb{1}\{\phi[i]=1\}} \left(1 - p_i(\mathbf{x}_i)\right)^{\mathbb{1}\{\phi[i]=-1\}},$$

Since $\phi[i]$ is binary, we model using **GLM**, here, *logistic regression*:

$$p_i(\mathbf{x}) = P(\phi[i] = 1 \mid \mathbf{x}) = \mu(\mathbf{x}^\mathsf{T}\theta^*), \qquad \mu(u) = 1/(1 + e^{-u}),$$

- θ^* is a column vector of parameters to be learned.
- The (immediate) gain: $g_i(\mathbf{y}) = p_i(\mathbf{x}_i)\bar{g}_i(\mathbf{y})$
  where $\bar{g}_i(\mathbf{y})$ is the expected gain of choosing item i in state 1 and can be computed without knowing $P(\Phi)$.

The framework & algorithm

We repetitively play the **K** step game.

Input: States $\phi_1, \ldots, \phi_n$

for $t = 1, 2, \ldots,$ **n do** ◁ **n** *episodes*

1. **Play the K-step game with** π^t

2. **Update all statistics of the model**

end for

We try to design π^t in order to minimize the **cumulative regret**

$$R(n) = \mathbb{E}_{\phi_1,\ldots,\phi_{\mathbf{n}}}\left[\sum_{t=1}^{\mathbf{n}} f(\pi^g_{\mathbf{K}}(\phi_t), \phi_t) - f(\pi^t_{\mathbf{K}}(\phi_t), \phi_t)\right].$$

The framework & algorithm

We repetitively play the **K** step game.

Input: States $\phi_1, \ldots, \phi_n$

for $t = 1, 2, \ldots,$ **n do** $\quad \triangleleft$ **n** *episodes*

1. **Play the K-step game with** π^t

   $\pi^t(\mathbf{y}) = \arg\max_i \widehat{g}_i(\mathbf{y})$ *(upper bound on* $g_i(\mathbf{y})$*)*

2. **Update all statistics of the model**

end for

We try to design π^t in order to minimize the **cumulative regret**

$$R(n) = \mathbb{E}_{\phi_1,\ldots,\phi_{\mathbf{n}}}\left[\sum_{t=1}^{\mathbf{n}} f(\pi^g_{\mathbf{K}}(\phi_t), \phi_t) - f(\pi^t_{\mathbf{K}}(\phi_t), \phi_t)\right].$$

The framework & algorithm

We repetitively play the **K** step game.

Input: States $\phi_1, \ldots, \phi_n$

for $t = 1, 2, \ldots,$ **n do** $\quad \triangleleft$ **n** *episodes*

1. **Play the K-step game with** π^t

   $\pi^t(\mathbf{y}) = \arg\max_i \left(\mu(\mathbf{x}_i^\top \tilde{\theta}_t) + \rho_{k,t}(\delta) \|\mathbf{x}_i\|_{M_t^{-1}} \right) \bar{g}_i(\mathbf{y})$

2. **Update all statistics of the model**

   Regression to estimate $\tilde{\theta}_t$

end for

We try to design π^t in order to minimize the **cumulative regret**

$$R(n) = \mathbb{E}_{\phi_1,\ldots,\phi_{\mathbf{n}}} \left[\sum_{t=1}^{\mathbf{n}} f(\pi^g_{\mathbf{K}}(\phi_t), \phi_t) - f(\pi^t_{\mathbf{K}}(\phi_t), \phi_t) \right].$$

Analysis

- We prove a gap-dependent bound on the expected cumulative regret of `LinOASM`.
- Our analysis is based on counting the number of times when the policy π^t selects a different item from the policy π^g at step k.

$$R(n) \leq \underbrace{\sum_{k=1}^{K} G_k(\ell_k + O(1))}_{O(\log^3 n)},$$

$\ell_k =$ Max # of wrong action at step k. Scales with d^2.

$G_k \geq$ expected gain of π^g after level k

Experiments on MovieLens

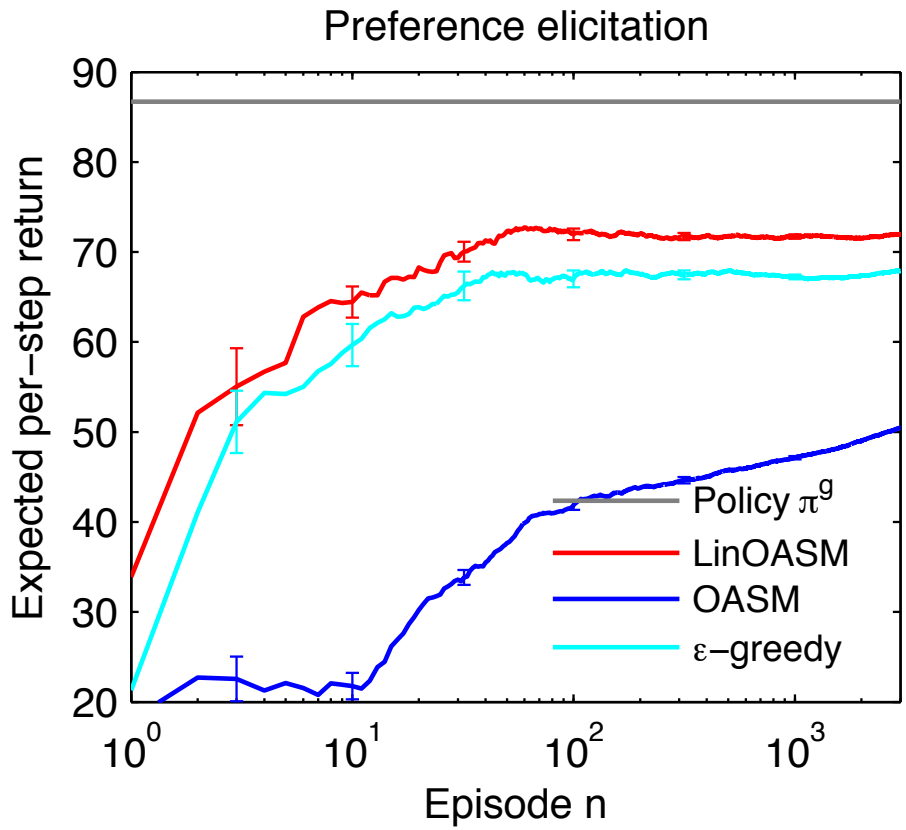
- All movie preference vectors ϕ_j are estimated from historical ratings of the users using low-rank matrix factorization.
- $\mathbf{x}_i$ describes the movies. $d = 10$.

The return $f(A, \phi)$:

$$f(A,\phi) = \sum_{\ell=1}^{500} 0.1 * \mathbb{1}\{\exists i \in A : \|\mathbf{x}_\ell - \mathbf{x}_i\|_2 \leq 0.2\} + 0.9 * \mathbb{1}\{\exists i \in A : \|\mathbf{x}_\ell - \mathbf{x}_i\|_2 \leq 0.2, \phi[i] = 1\}.$$

The function $f(A, \phi)$ is adaptive submodular and is maximized when A is a diverse set of liked movies.

Experiments on Movie-Lens



The expected per-step return up to $n = 3$k.

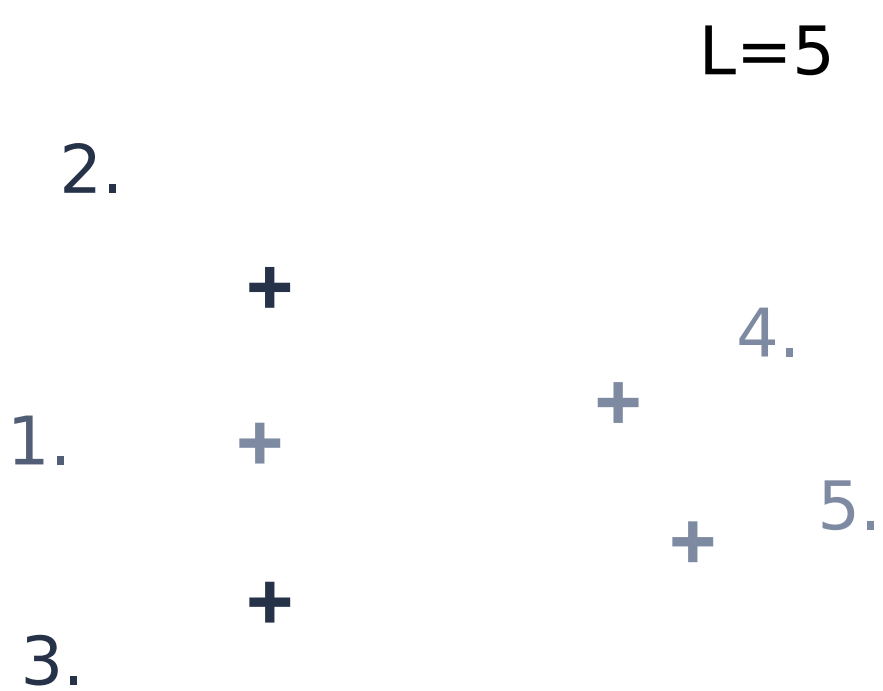
Conclusions

- We propose a method that solves a large special POMDP problem.
- We bring together adaptive submodularity and generalized linear bandits.
- Our analysis shows that the regret scales with d^2 and is polylogarithmic in n.
- Experiments demonstrates its applicability in a face detection problem ($d = 5, L = 81$) and a movie recommander system ($d = 10, L = 500$)

Thank you!

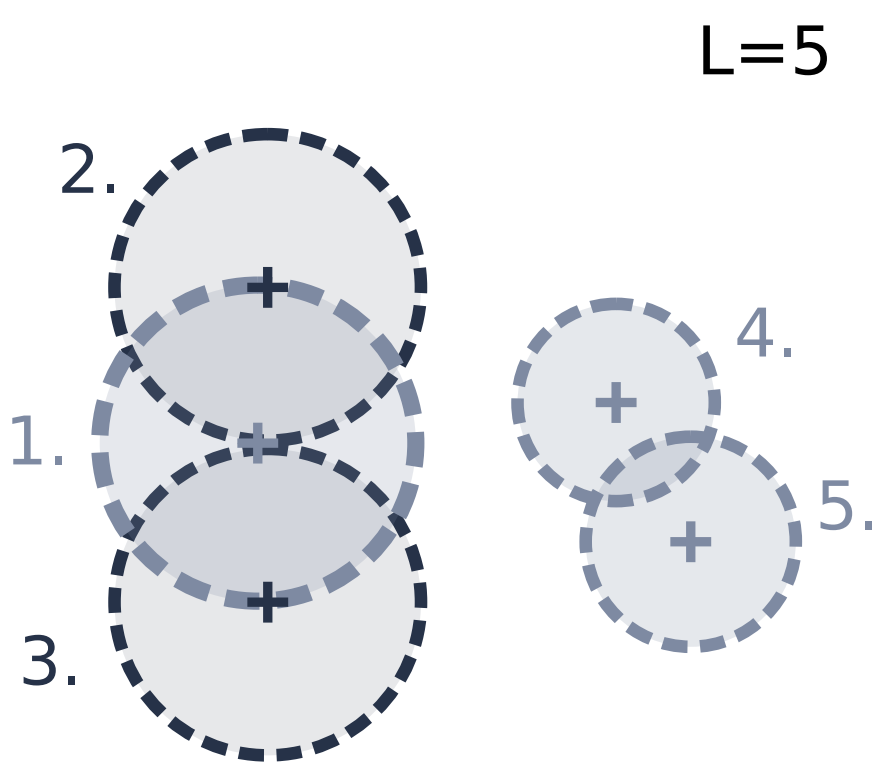
Sensor activation problem.

- 5 sensors ($L = 5$) whose placements is fixed (and known).



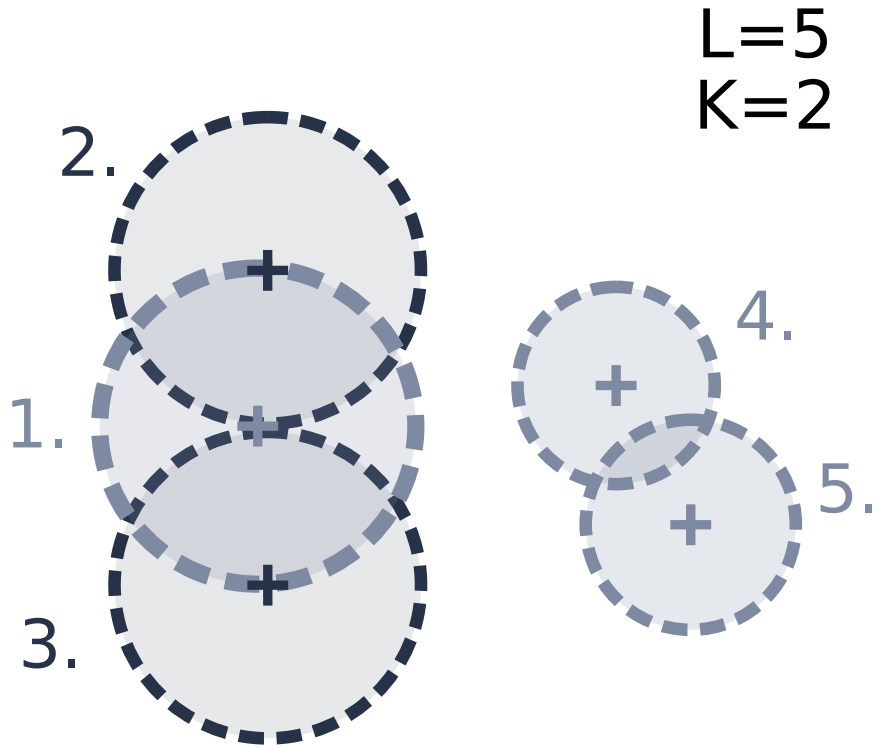
Sensor activation problem.

- 5 sensors ($L = 5$) whose placements is fixed (and known).
- Each area covered by a sensor is known



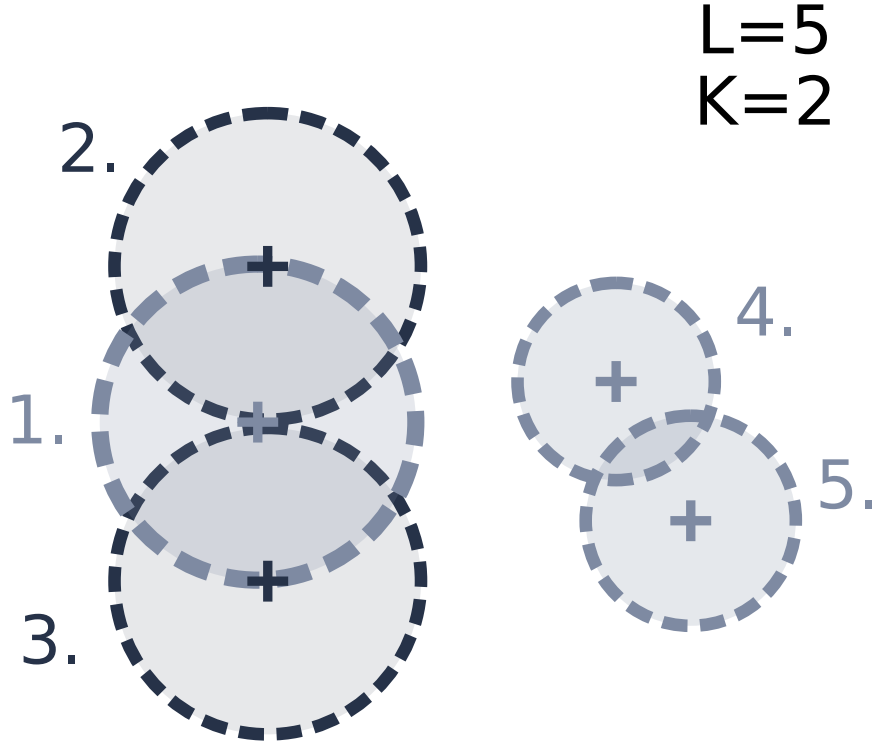
Sensor activation problem.

- 5 sensors ($L = 5$) whose placements is fixed (and known).
- Each area covered by a sensor is known
- **Goal:** Maximize the total covered area with 2 sensors ($K = 2$).



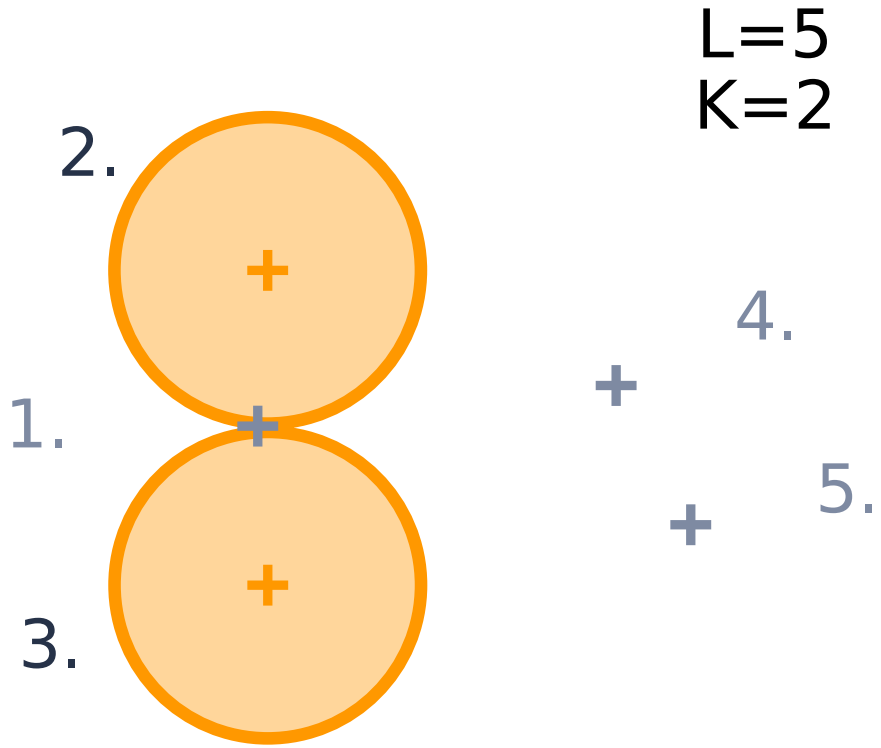
Sensor activation problem.

- 5 sensors ($L = 5$) whose placements is fixed (and known).
- Each area covered by a sensor is known
- **Goal:** Maximize the total covered area with 2 sensors ($K = 2$).



Sensor activation problem.

- 5 sensors ($L = 5$) whose placements is fixed (and known).
- Each area covered by a sensor is known
- **Goal:** Maximize the total covered area with 2 sensors ($K = 2$).

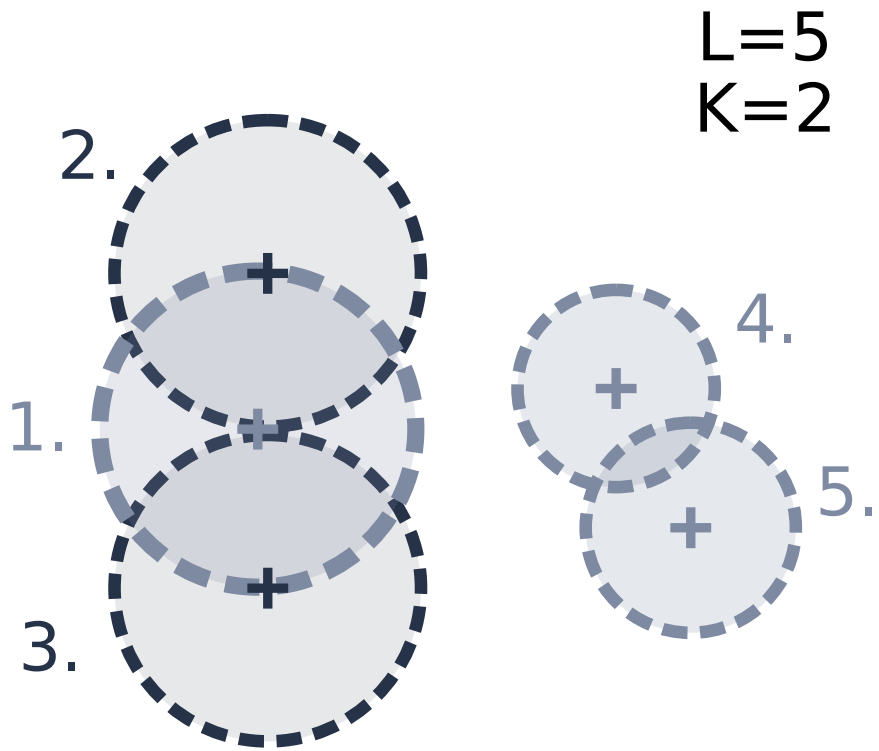


Good Solutions:

Optimal Solution: 2.& 3.

Sensor activation problem.

- 5 sensors ($L = 5$) whose placements is fixed (and known).
- Each area covered by a sensor is known
- **Goal:** Maximize the total covered area with 2 sensors ($K = 2$).



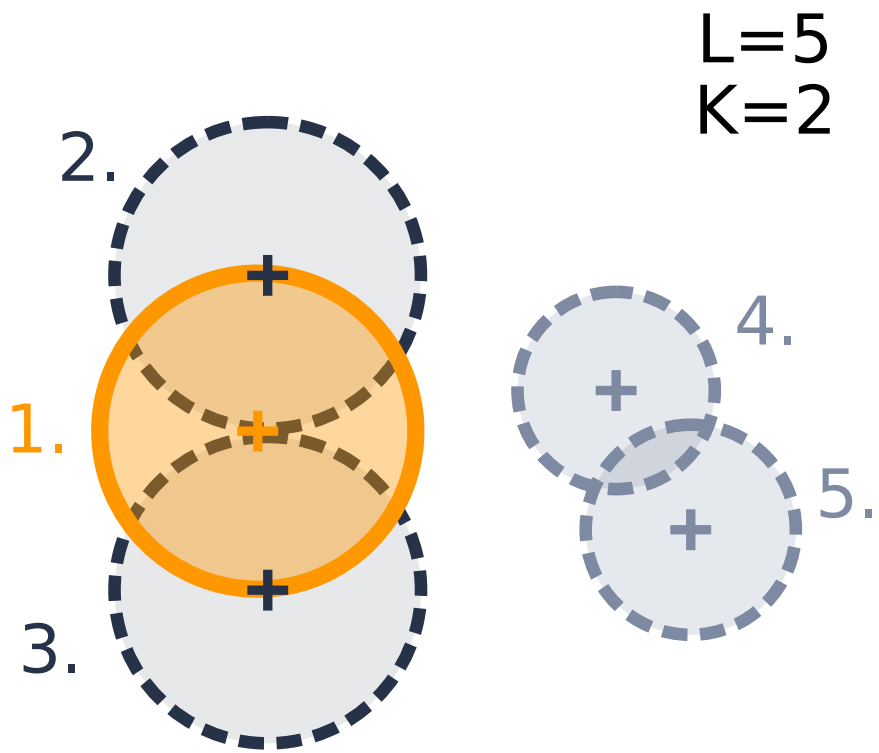
Good Solutions:

Optimal Solution: 2.& 3.

Greedy Solution:

Sensor activation problem.

- 5 sensors ($L = 5$) whose placements is fixed (and known).
- Each area covered by a sensor is known
- **Goal:** Maximize the total covered area with 2 sensors ($K = 2$).

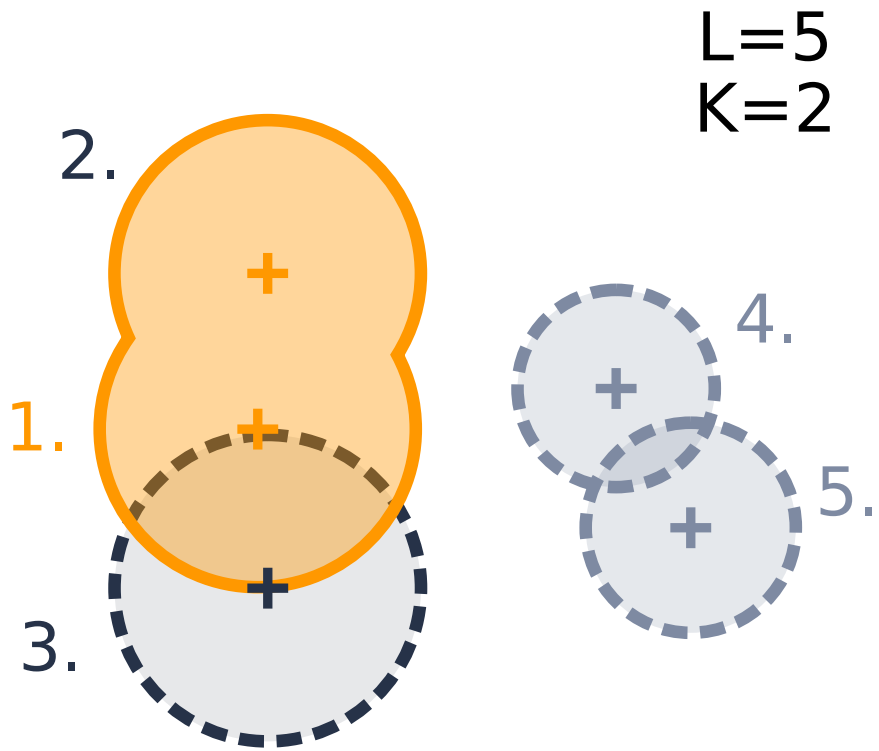


Good Solutions:

Optimal Solution: 2.& 3.
Greedy Solution: 1.

Sensor activation problem.

- 5 sensors ($L = 5$) whose placements is fixed (and known).
- Each area covered by a sensor is known
- **Goal:** Maximize the total covered area with 2 sensors ($K = 2$).



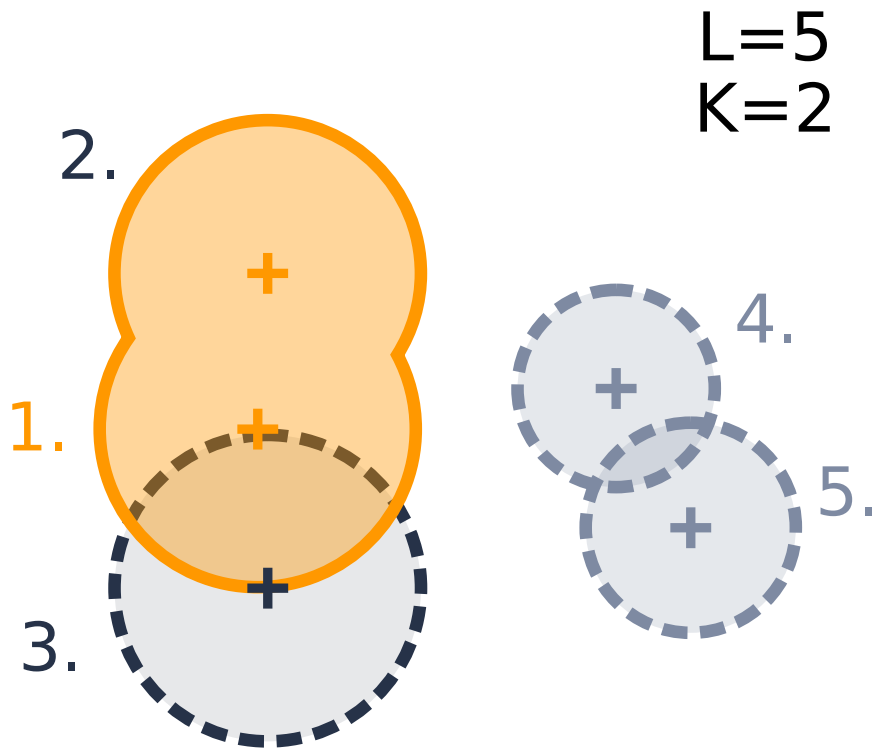
Good Solutions:

Optimal Solution: 2.& 3.

Greedy Solution: 1.& 2.

Sensor activation problem.

- 5 sensors ($L = 5$) whose placements is fixed (and known).
- Each area covered by a sensor is known
- **Goal:** Maximize the total covered area with 2 sensors ($K = 2$).



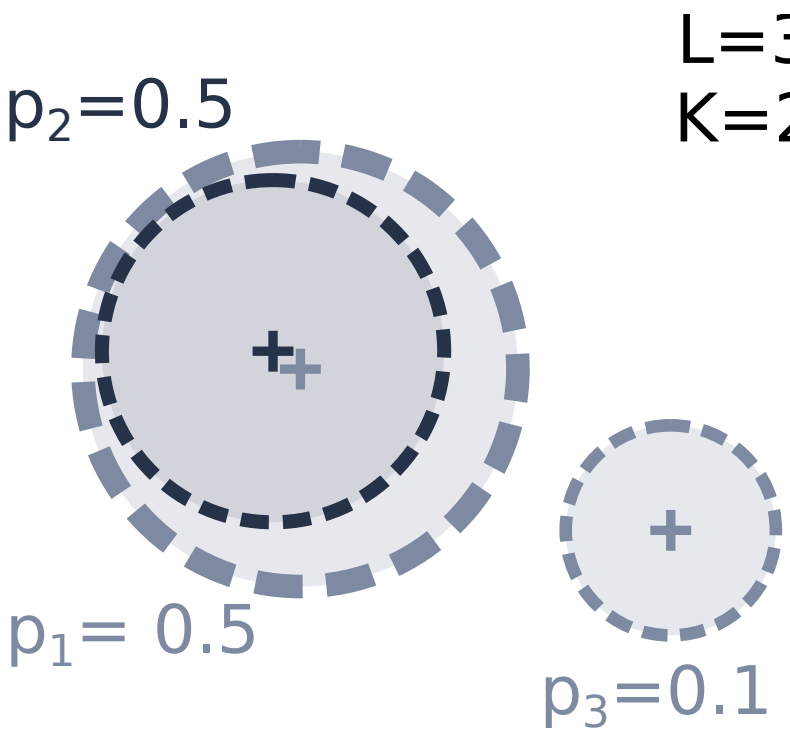
Good Solutions:

Optimal Solution: 2.& 3.
Greedy Solution: 1.& 2.

Because the problem is submodular, the greedy solution is optimal up to a constant multiplicative factor ($1 - 1/e \approx 0.63$)

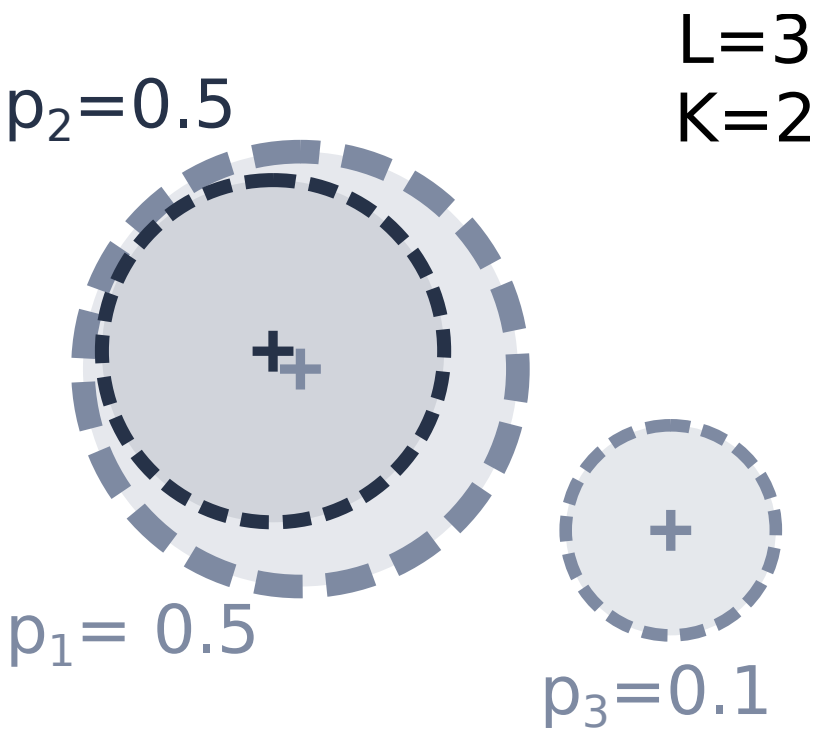
Stochastic sensor activation problem.

- Upon activation, each sensor i cover its area with probability p_i. There is 2 **states**: failure or success.
- **Goal:** Maximize the total **expected** covered area with 2 sensors, **chosen in advance**.



Stochastic sensor activation problem.

- Upon activation, each sensor i cover its area with probability p_i. There is 2 **states**: failure or success.
- **Goal:** Maximize the total **expected** covered area with 2 sensors, **chosen in advance**.



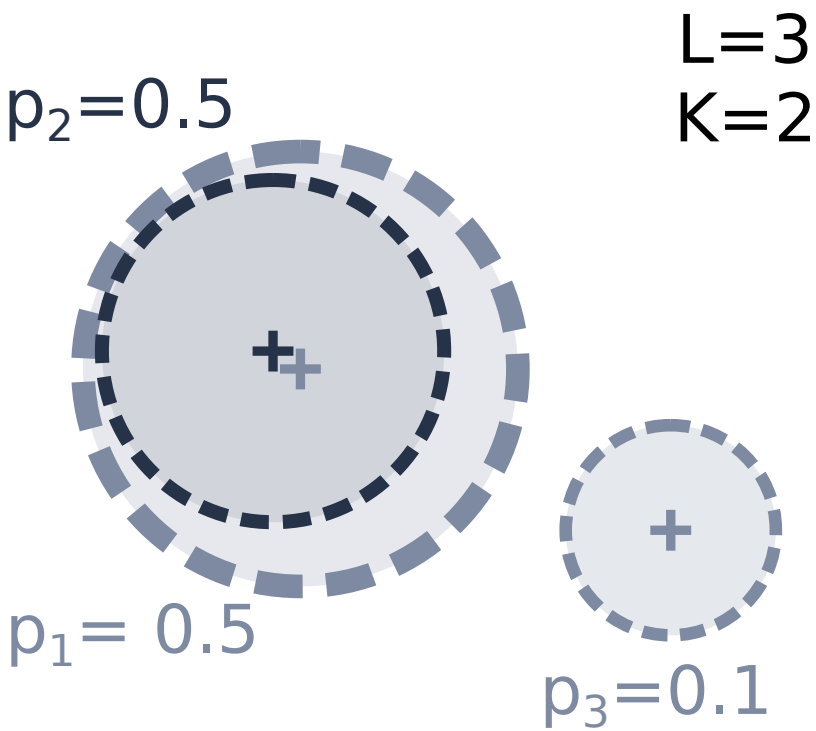
Good Solutions:

Optimal solution: 1.& 2.
Greedy solution: 1.& 2.

Again, the greedy solution is $(1-1/e)-$optimal

Adaptive sensor activation problem.

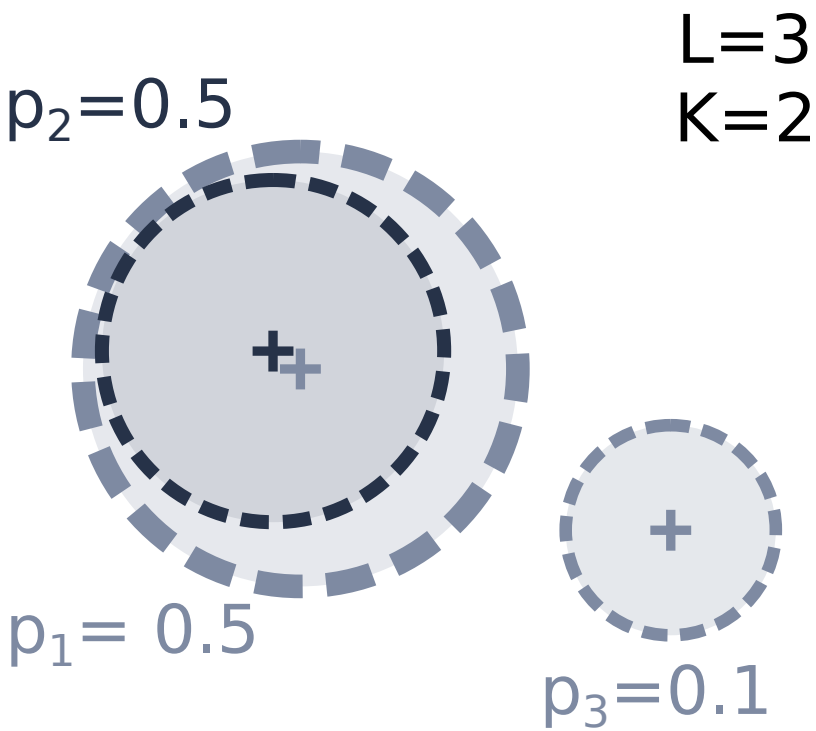
- Upon activation, each sensor i cover its area with probability p_i and the state of the sensor is **observed**.
- **Goal:** Maximize the total **expected** covered area with 2 sensors, **chosen adaptively**.



Adaptive sensor activation problem.

- Upon activation, each sensor i cover its area with probability p_i and the state of the sensor is **observed**.
- **Goal:** Maximize the total **expected** covered area with 2 sensors, **chosen adaptively**.

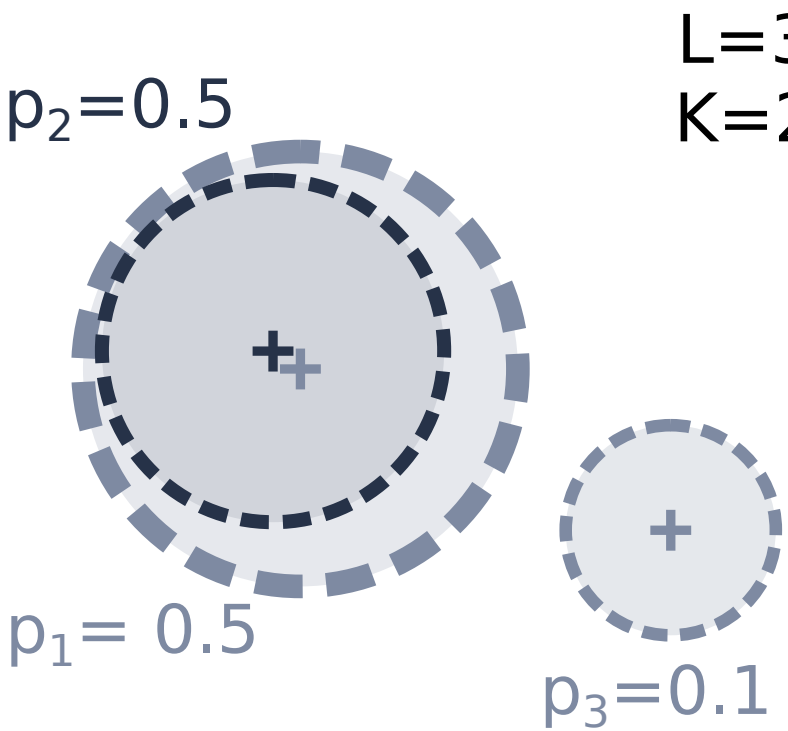
Good Policies:



Adaptive sensor activation problem.

- Upon activation, each sensor i cover its area with probability p_i and the state of the sensor is **observed**.
- **Goal:** Maximize the total **expected** covered area with 2 sensors, **chosen adaptively**.

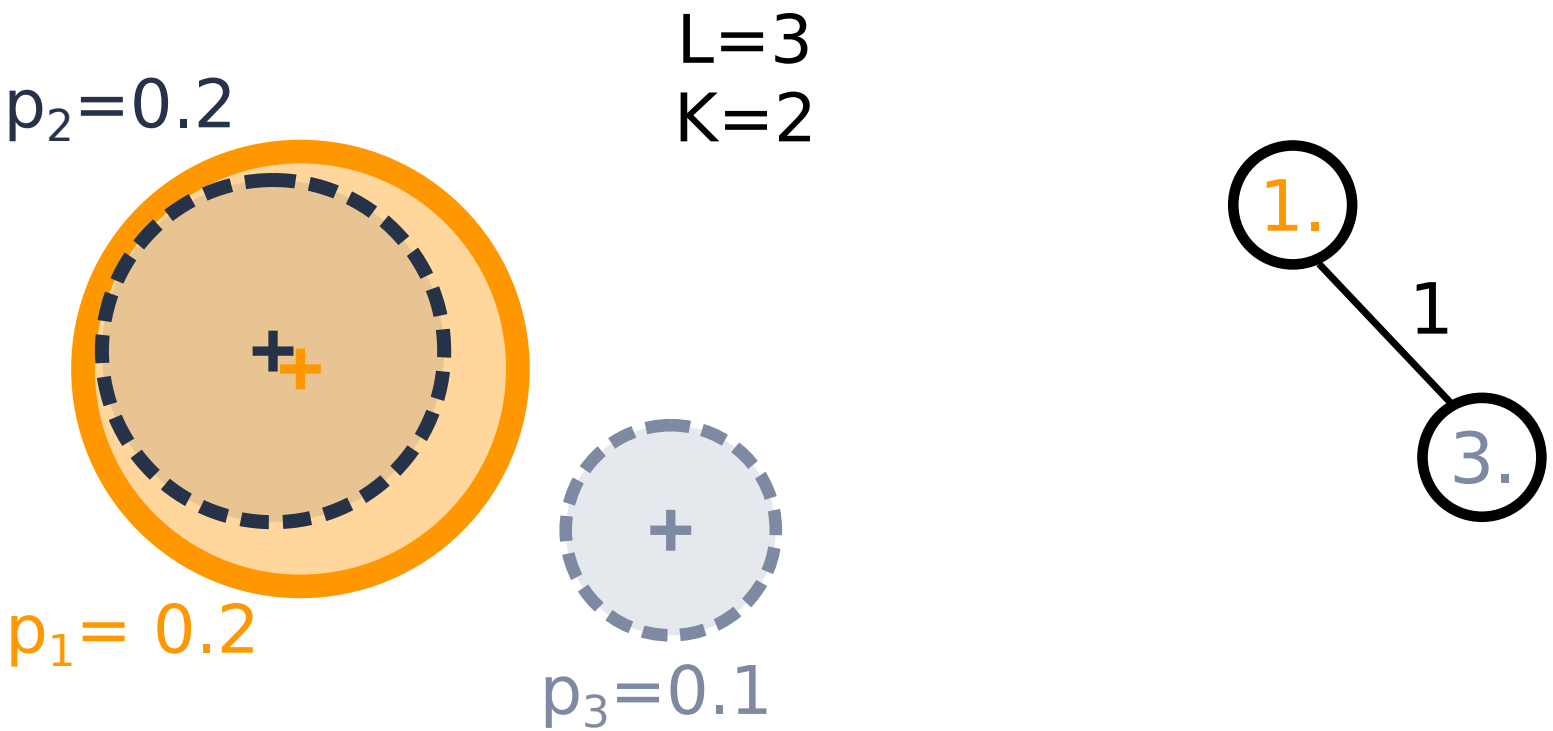
Good Policies:



Adaptive sensor activation problem.

- Upon activation, each sensor i cover its area with probability p_i and the state of the sensor is **observed**.
- **Goal:** Maximize the total **expected** covered area with 2 sensors, **chosen adaptively**.

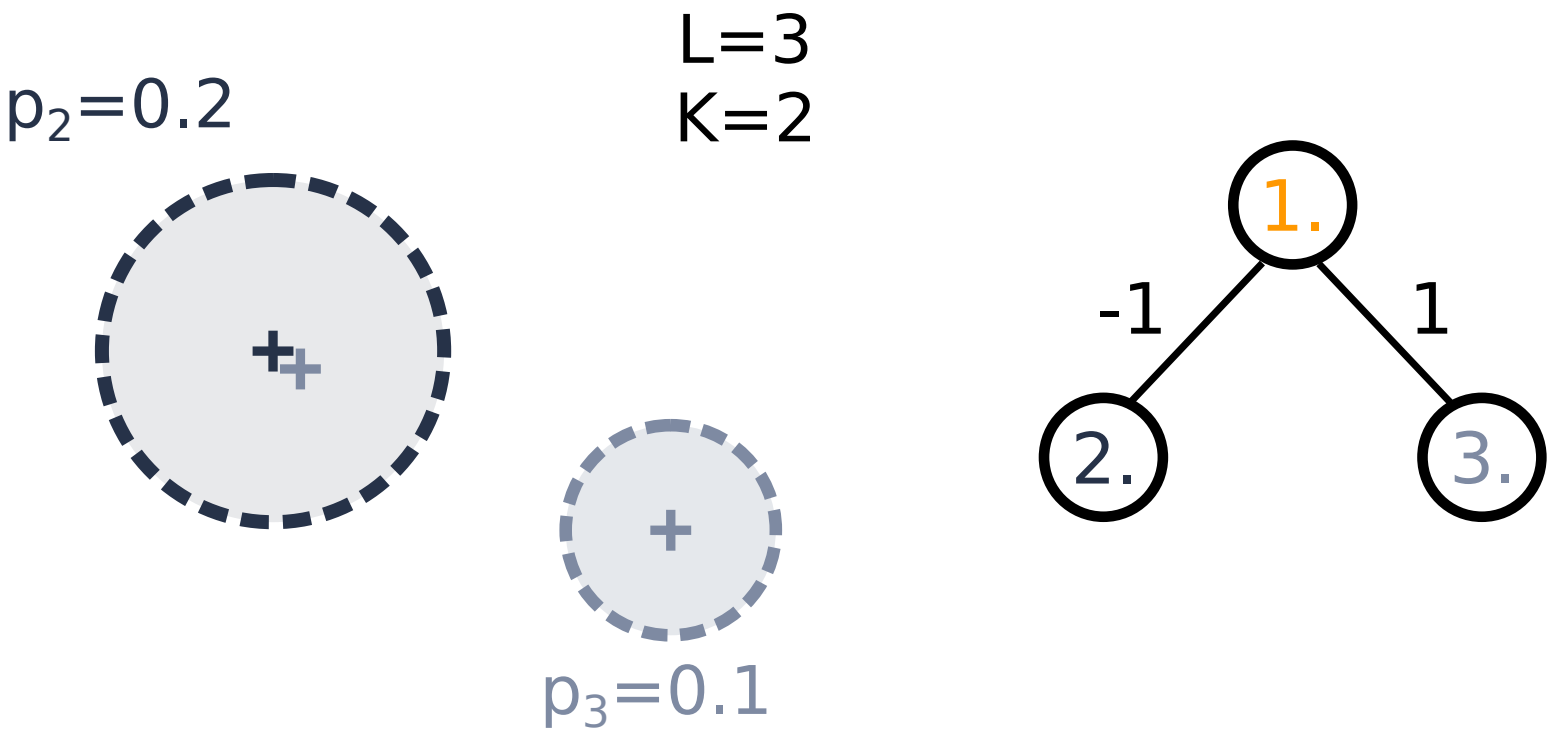
Good Policies:



Adaptive sensor activation problem.

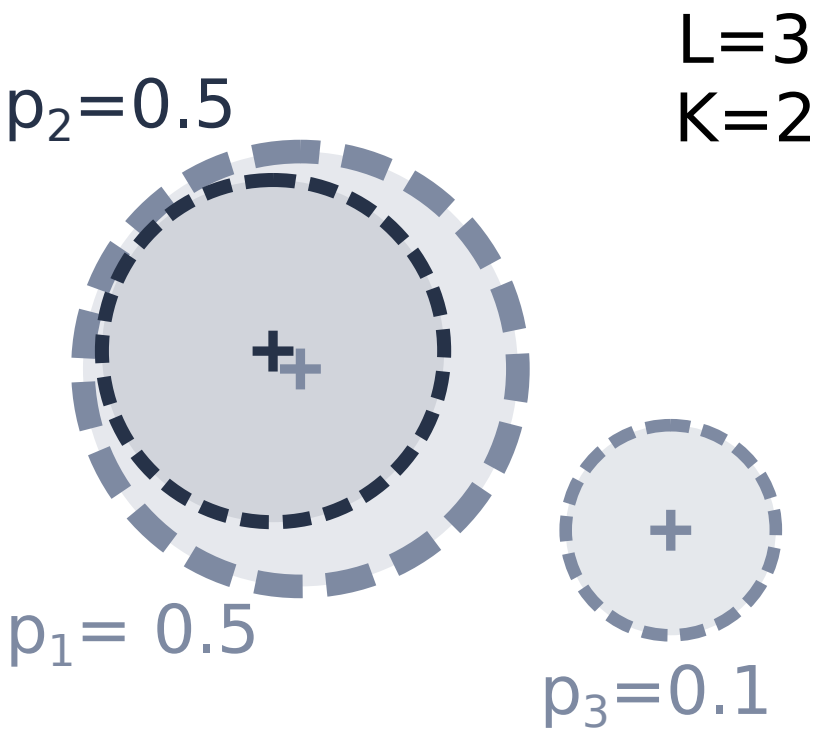
- Upon activation, each sensor i cover its area with probability p_i and the state of the sensor is **observed**.
- **Goal:** Maximize the total **expected** covered area with 2 sensors, **chosen adaptively**.

Good Policies:

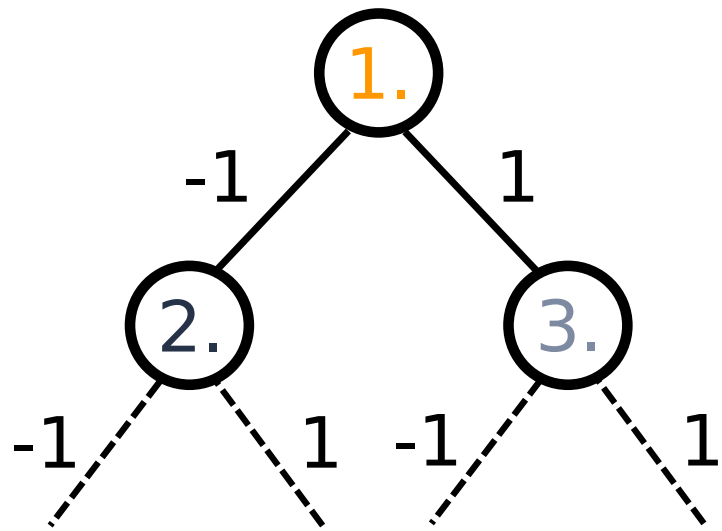


Adaptive sensor activation problem.

- Upon activation, each sensor i cover its area with probability p_i and the state of the sensor is **observed**.
- **Goal:** Maximize the total **expected** covered area with 2 sensors, **chosen adaptively**.

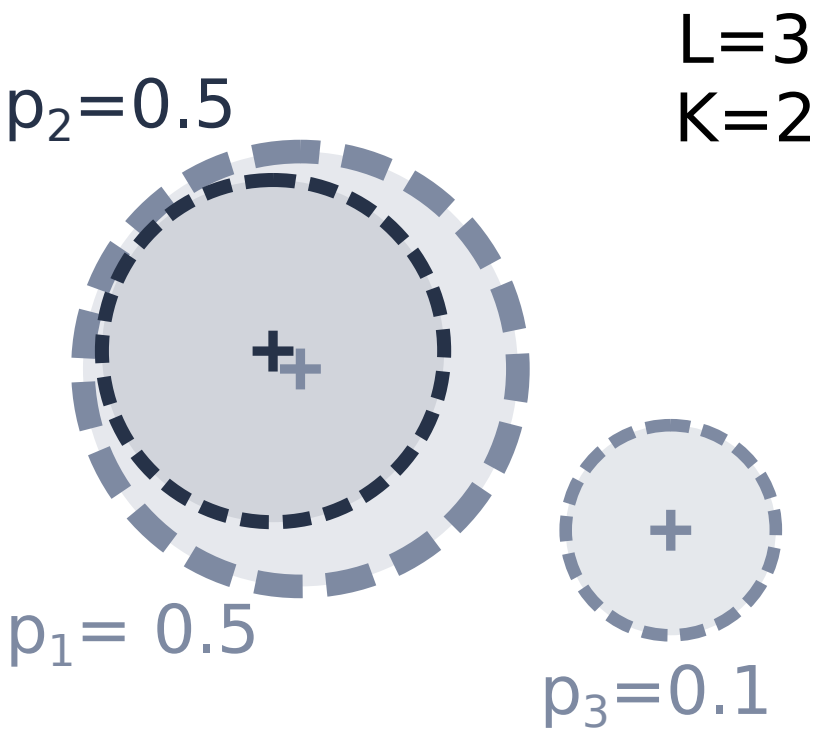


Good Policies:

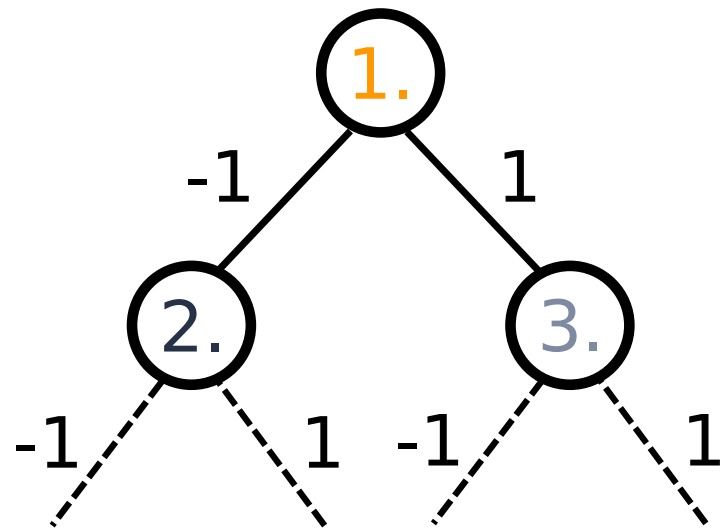


Adaptive sensor activation problem.

- Upon activation, each sensor i cover its area with probability p_i and the state of the sensor is **observed**.
- **Goal:** Maximize the total **expected** covered area with 2 sensors, **chosen adaptively**.



Good Policies:



Greedy is $(1-1/e)$−optimal

Adaptive submodular maximization

The objective is to maximize a **real** function of the form:

$$f(\underbrace{A}_{\text{Set of selected items}}, \underbrace{\phi}_{\text{State of the } L \text{ items}}) \to \mathbb{R}$$

- $\phi \in \{-1, 1\}^L$, where $\phi[i]$ is the state of item i.
- ϕ is drawn i.i.d. from $P(\Phi)$.

An **observation** is a vector $\mathbf{y} \in \{-1, 0, 1\}^L$.

$$\begin{aligned}
\phi &= (\ -1,\ -1,\ 1,\ -1,\ 1\ ) \leftarrow \text{State} \\
\mathbf{y} &= (\ 0,\ -1,\ 0,\ -1,\ 1\ ) \leftarrow \text{observation} \\
\mathbf{y}' &= (\ 0,\ 0,\ 0,\ -1,\ 0\ ) \leftarrow \text{another observation}
\end{aligned}$$

We denote $\mathbf{y} \succeq \mathbf{y}'$ and $\phi \sim \mathbf{y}$.
The selected items of $\mathbf{y}$ are $\mathrm{dom}(\mathbf{y}) = A = \{2, 4, 5\}$.

Adaptive submodular maximization

- a *policy* $\pi : \{-1, 0, 1\}^L \to \{1, \ldots, L\}$
- $\pi_k(\phi)$ are the first k items chosen by policy π in state ϕ.

The optimal **K**-step policy satisfies:

$$\pi_K^* = \arg\max_{\pi_K} \mathbb{E}_\phi[f(\pi_K(\phi), \phi)] .$$

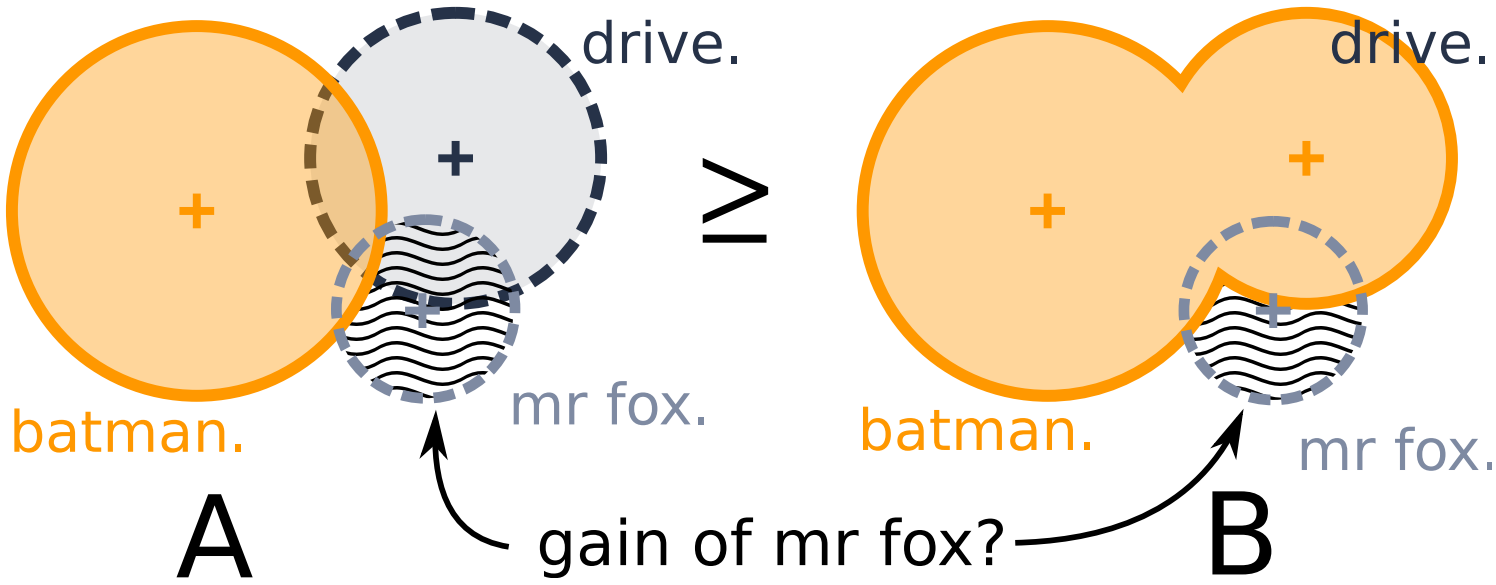
Computing π_K^* is NP-hard. **BUT**, if the function is adaptive submodular and adaptive monotonic...

Assumptions

f is adaptive submodular if:

$$\mathbb{E}_\phi[\, f(A \cup \{i\}, \phi) - f(A, \phi) \,|\, \phi \sim \mathbf{y}_A \,] \geq \mathbb{E}_\phi[\, f(B \cup \{i\}, \phi) - f(B, \phi) \,|\, \phi \sim \mathbf{y}_B \,]$$

$i \in I \setminus B$ and $\mathbf{y}_B \succeq \mathbf{y}_A$, where $A = \mathrm{dom}(\mathbf{y}_A)$ and $B = \mathrm{dom}(\mathbf{y}_B)$.



Assumptions

f is adaptive submodular if:

$$\mathbb{E}_\phi[\, f(A \cup \{i\}, \phi) - f(A, \phi) \,|\, \phi \sim \mathbf{y}_A \,] \geq \mathbb{E}_\phi[\, f(B \cup \{i\}, \phi) - f(B, \phi) \,|\, \phi \sim \mathbf{y}_B \,]$$

$i \in I \setminus B$ and $\mathbf{y}_B \succeq \mathbf{y}_A$, where $A = \mathrm{dom}(\mathbf{y}_A)$ and $B = \mathrm{dom}(\mathbf{y}_B)$.

f is adaptive monotonic if

$$\mathbb{E}_\phi[\, f(A \cup \{i\}, \phi) - f(A, \phi) \,|\, \phi \sim \mathbf{y}_A \,] \geq 0$$

$i \in I \setminus A$ and $\mathbf{y}_A$, where $A = \mathrm{dom}(\mathbf{y}_A)$.

The greedy policy π^g

π^g always selects the item with the highest expected gain:

$$\pi^g(\mathbf{y}) = \arg \max_{i \in I \setminus \mathrm{dom}(\mathbf{y})} g_i(\mathbf{y}),$$

where:

$$g_{\mathbf{i}}(\mathbf{y}) = \mathbb{E}_\phi[\, f(\mathrm{dom}(\mathbf{y}) \cup \{\mathbf{i}\}, \phi) - f(\mathrm{dom}(\mathbf{y}), \phi) \,|\, \phi \sim \mathbf{y}\,]$$

is the *expected gain* of choosing item i after observing $\mathbf{y}$.

- π^g is **simple**.
- Then π^g is a $(1 - 1/e)$-approximation to π^*.

The greedy policy π^g

π^g always selects the item with the highest expected gain:

$$\pi^g(\mathbf{y}) = \arg \max_{i \in I \setminus \mathrm{dom}(\mathbf{y})} g_i(\mathbf{y}),$$

where:

$$g_{\mathbf{i}}(\mathbf{y}) = \mathbb{E}_\phi[\, f(\mathrm{dom}(\mathbf{y}) \cup \{\mathbf{i}\}, \phi) - f(\mathrm{dom}(\mathbf{y}), \phi) \mid \phi \sim \mathbf{y} \,]$$

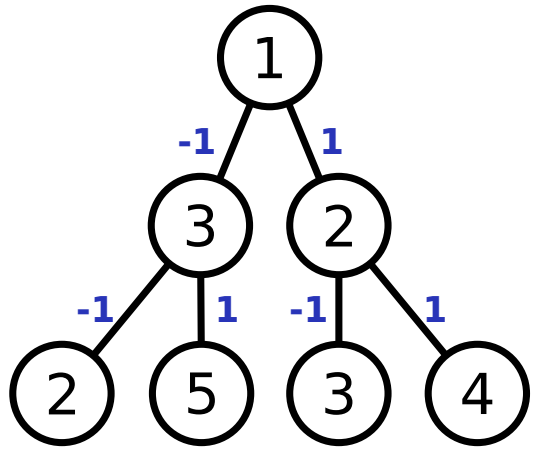
is the *expected gain* of choosing item i after observing $\mathbf{y}$.

- π^g is **simple**.
- Then π^g is a $(1 - 1/e)$-approximation to π^*.

Our approach

Our approach

Trying to **mimic** π^g while **learning** $P(\Phi)$.



What model for Φ?

Trade-off between

- Simple model for $\Phi \rightarrow$ Easy to learn
- Complex model $\rightarrow$ More realistic.

Two models

- Assuming structure

- Modelling the structure.

Two models

- Assuming structure

- Modelling the structure.

Independence assumption

The states of one item is independent of the other.

$$\phi = \{Bernoulli(p_1), Bernoulli(p_2), \ldots, Bernoulli(p_L)\}$$

$\rightarrow$ Only L parameters to learn.

We define the expected gain as:

$$g_i(\mathbf{y}) = p_i \bar{g}_i(\mathbf{y}),$$

$\bar{g}_i(\mathbf{y})$ = gain associated with state 1. (area covered if sensor is active)

Independence assumption

The states of one item is independent of the other.

$$\phi = \{Bernoulli(p_1), Bernoulli(p_2), \ldots, Bernoulli(p_L)\}$$

$\rightarrow$ Only L parameters to learn.

We define the expected gain as:

$$g_i(\mathbf{y}) = p_i \bar{g}_i(\mathbf{y}),$$

$\bar{g}_i(\mathbf{y})$ = gain associated with state 1. (area covered if sensor is active)

- We assume is easy to compute.
- Might not even be an expectation.