Submodular Optimization: From Discrete to Continuous and Back

Hamed Hassani

Amin Karbasi

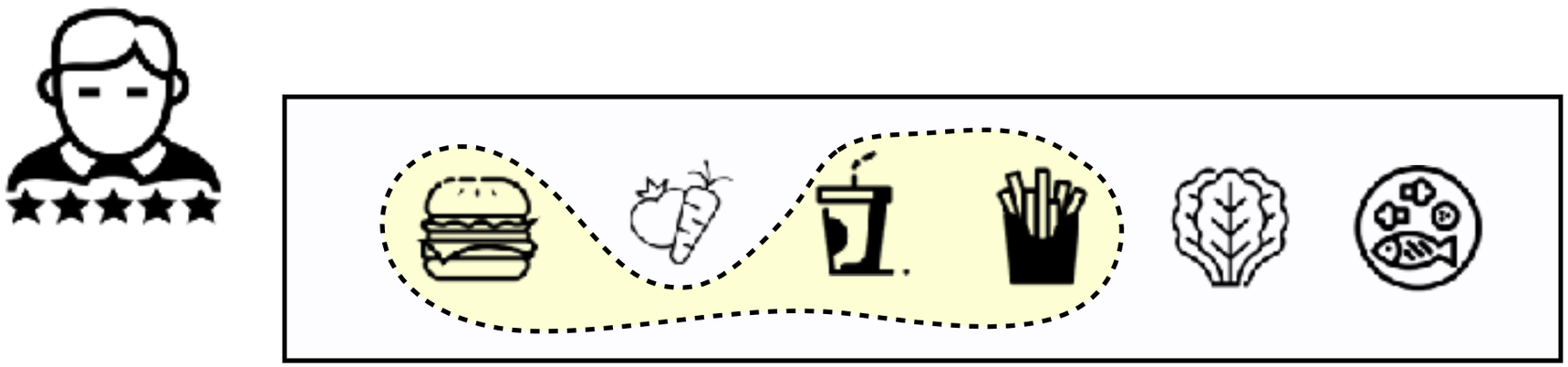
Penn

Yale

Slides + references: http://iid.yale.edu/icml/icml-20.md/

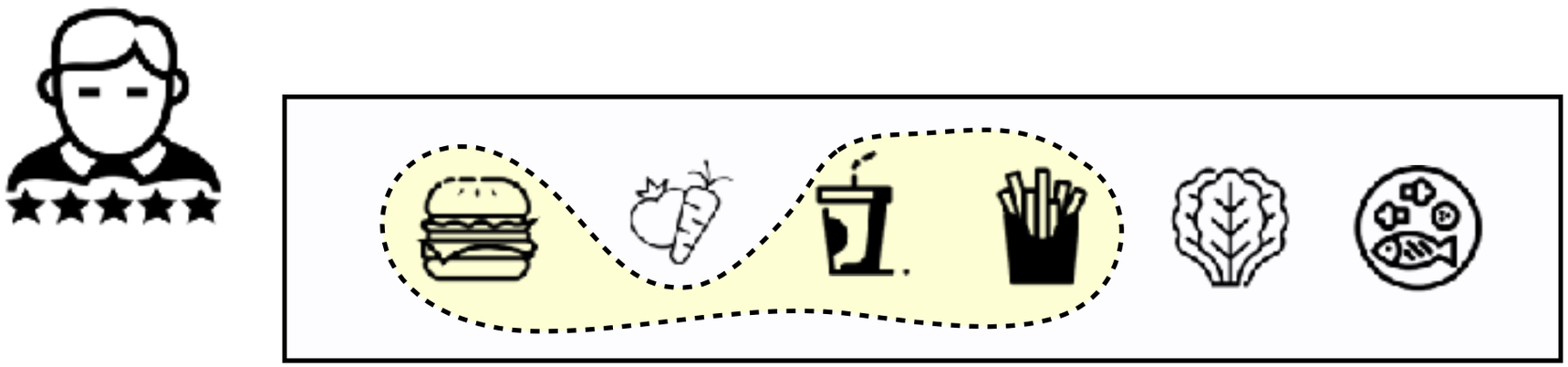
Subset Selection

- Propose a subset of items that make the user happy

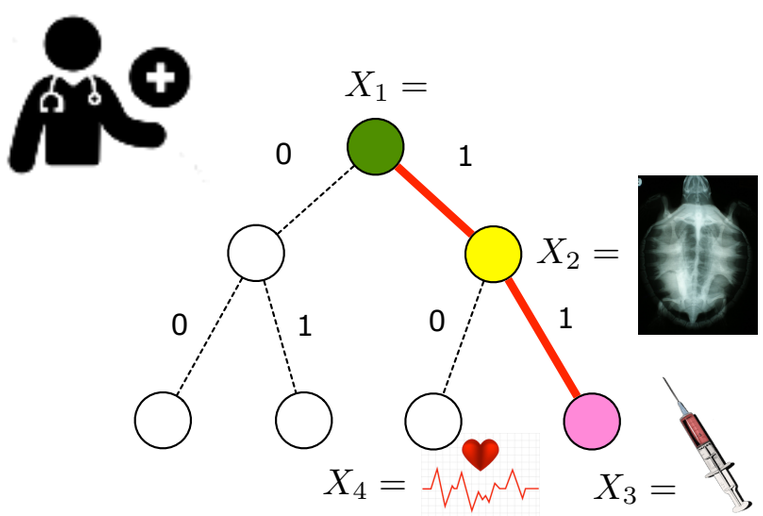


Subset Selection

- Propose a subset of items that make the user happy

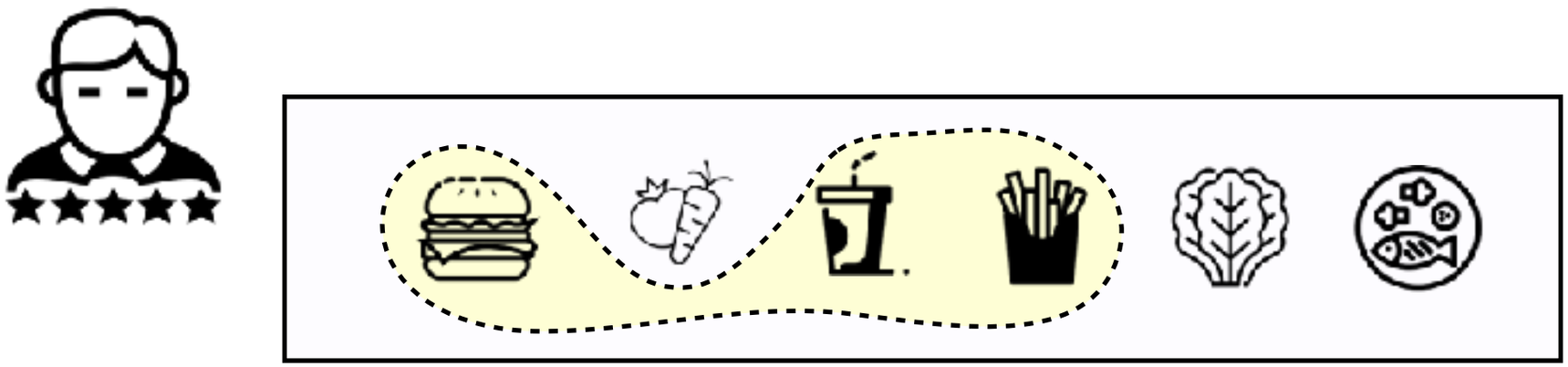


- Run the most informative subset of experiments to reach a conclusion

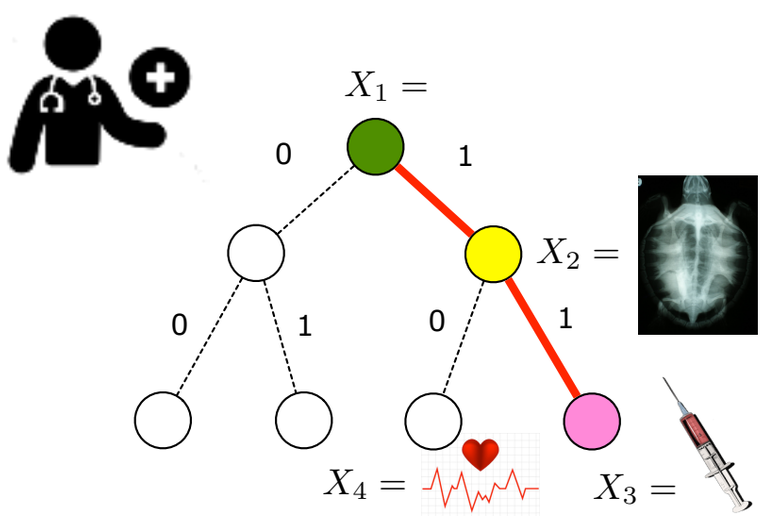


Subset Selection

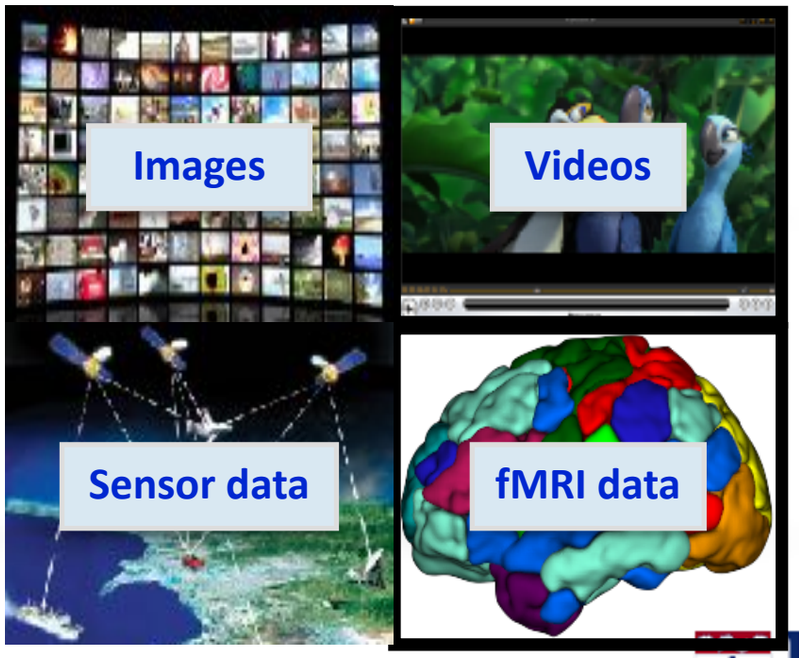
- Propose a subset of items that make the user happy



- Run the most informative subset of experiments to reach a conclusion



- Select the subset of data points best represents the dataset



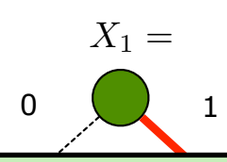
Subset Selection

- Propose a subset of items that make the user happy

$V =$ items

$f(S) =$ **How satisfied he/she is?**

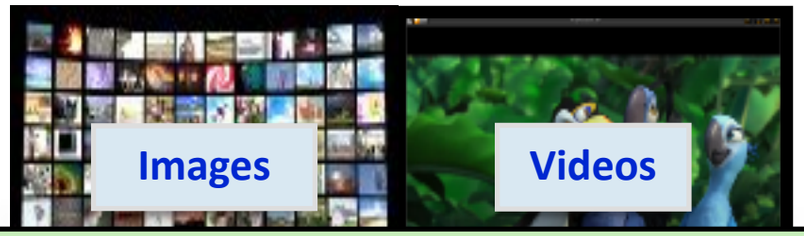
- Run the most informative subset of experiments to reach a conclusion



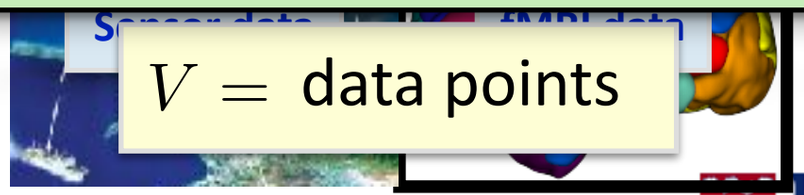
$f(S) =$ **How informative is it?**

$V =$ tests

- Select the subset of data points best represents the dataset



$f(S) =$ **How representative is it?**



$V =$ data points

Structure Is the Key

In all these problems the goal is to:

$$\max_{S \subseteq V, S \in \mathcal{I}} f(S)$$

Structure Is the Key

In all these problems the goal is to:

$$\max_{S \subseteq V, \boxed{S \in \mathcal{I}}} f(S)$$

constraints

Cardinality constraint

Knapsack constraint

Matroid constraint

Structure Is the Key

In all these problems the goal is to:

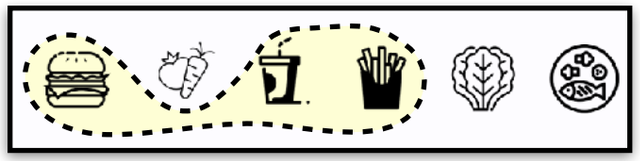
$$\max_{S \subseteq V, \boxed{S \in \mathcal{I}}} f(S)$$

constraints

Cardinality constraint

Knapsack constraint

Matroid constraint



Structure Is the Key

In all these problems the goal is to:

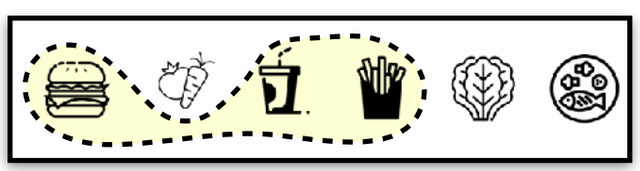
$$\max_{S \subseteq V, \boxed{S \in \mathcal{I}}} f(S)$$

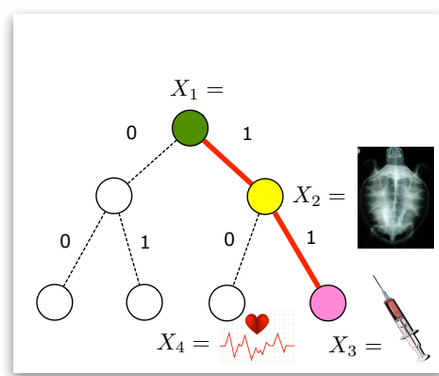
constraints

Cardinality constraint

Knapsack constraint

Matroid constraint





Structure Is the Key

In all these problems the goal is to:

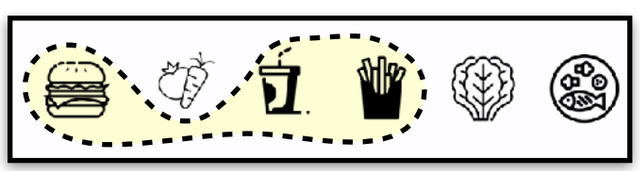
$$\max_{S \subseteq V, \boxed{S \in \mathcal{I}}} f(S)$$

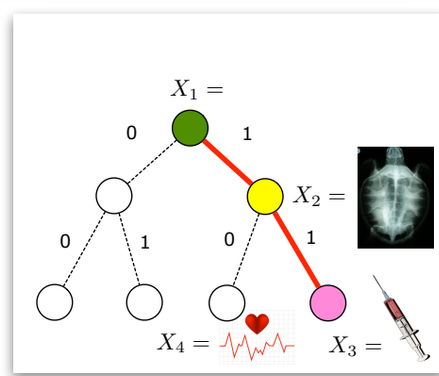
constraints

Cardinality constraint

Knapsack constraint

Matroid constraint





Structure Is the Key

In all these problems the goal is to:

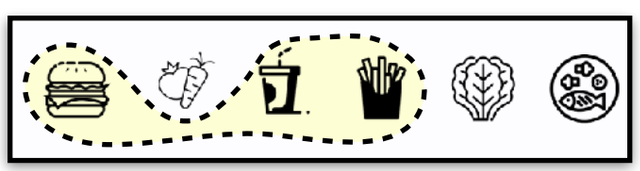
$$\max_{S \subseteq V, \boxed{S \in \mathcal{I}}} f(S)$$

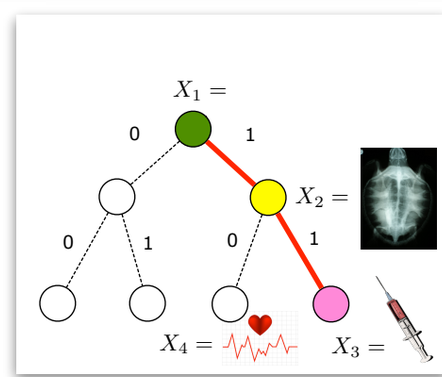
constraints

Cardinality constraint

Knapsack constraint

Matroid constraint





- Hopeless without further structures

Properties of a Good Structure

L. Lovász

1983

Submodular functions and convexity

L. Lovász
Eötvös Loránd University, Department of Analysis I, Múzeum krt. 6–8, H-1088 Budapest, Hungary

0. Introduction

In "continuous" optimization convex functions play a central role. Besides elementary tools like differentiation, various methods for finding the minimum of a convex function constitute the main body of nonlinear optimization. But even linear programming may be viewed as the optimization of very special (linear) objective functions over very special convex domains (polyhedra). There are several reasons for this popularity of convex functions:

– Convex functions occur in many mathematical models in economy, engineering, and other sciencies. Convexity is a very natural property of various functions and domains occuring in such models; quite often the only non-trivial property which can be stated in general.

– Convexity is preserved under many natural operations and transformations, and thereby the effective range of results can be extended, elegant proof techniques can be developed as well as unforeseen applications of certain results can be given.

– Convex functions and domains exhibit sufficient structure so that a mathematically beautiful and practically useful theory can be developed.

– There are theoretically and practically (reasonably) efficient methods to find the minimum of a convex function.

It is less apperant, but we claim and hope to prove to a certain extent, that a similar role is played in discrete optimization by *submodular set-functions*. These functions do not enjoy the nice geometric image of convex functions, and accordingly their significance has been discovered only gradually. The first class of submodular functions which was studied thoroughly was the class of matroid rank functions. Generalizing certain basic properties of matroid polyhedra, Edmonds (1970) began the systematical study of submodularity. Let us remark, however, that approaching from quite a different angle, Choquet (1955) also introduced these set-functions. He proved that the newtonian "capacity" of a subset of $\mathbb{R}^3$ defines a submodular set-function. Quite a few proof techniques in graph theory, but also in probability, geometry and lattice theory have made implicit use of submodularity of certain set-functions, thus forecasting the importance of this property.

In this paper we survey some of the most important aspects of submodularity. In particular, we shall see that submodularity shares the above-listed four

Properties of a Good Structure

L. Lovász

Nontrivial property that can be stated in general

1983

Submodular functions and convexity

L. Lovász
Eötvös Loránd University, Department of Analysis I, Múzeum krt. 6–8, H-1088 Budapest, Hungary

0. Introduction

In "continuous" optimization convex functions play a central role. Besides elementary tools like differentiation, various methods for finding the minimum of a convex function constitute the main body of nonlinear optimization. But even linear programming may be viewed as the optimization of very special (linear) objective functions over very special convex domains (polyhedra). There are several reasons for this popularity of convex functions:

– Convex functions occur in many mathematical models in economy, engineering, and other sciences. Convexity is a very natural property of various functions and domains occuring in such models; quite often the only non-trivial property which can be stated in general.

– Convexity is preserved under many natural operations and transformations, and thereby the effective range of results can be extended, elegant proof techniques can be developed as well as unforeseen applications of certain results can be given.

– Convex functions and domains exhibit sufficient structure so that a mathematically beautiful and practically useful theory can be developed.

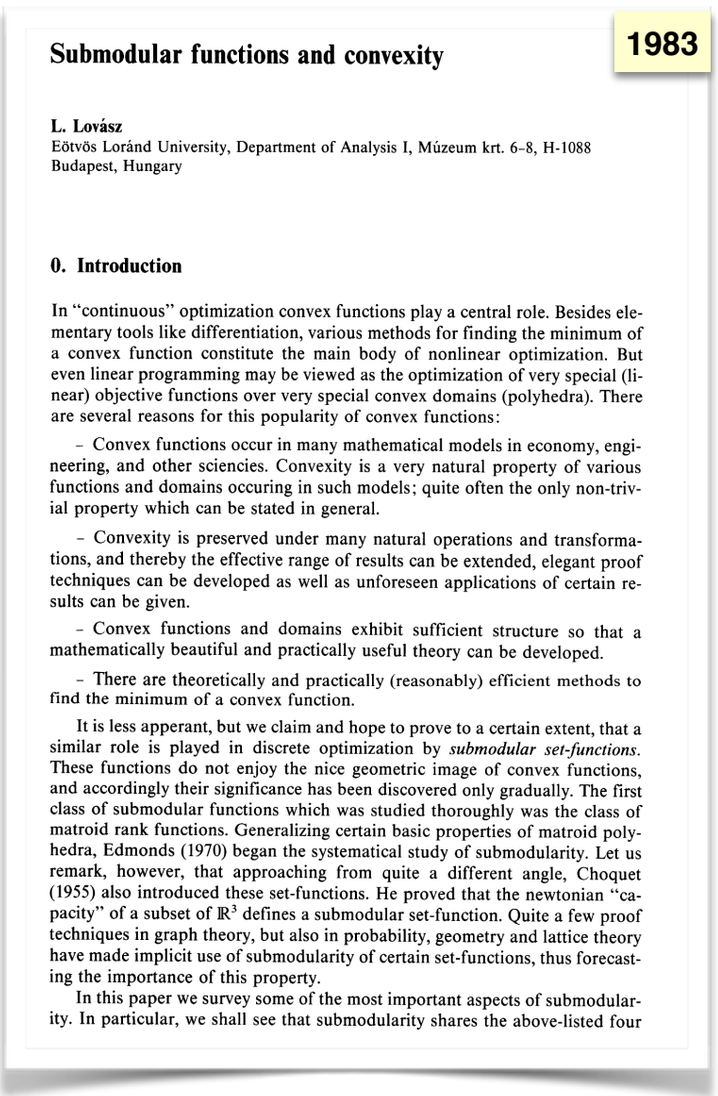
– There are theoretically and practically (reasonably) efficient methods to find the minimum of a convex function.

It is less apperant, but we claim and hope to prove to a certain extent, that a similar role is played in discrete optimization by *submodular set-functions*. These functions do not enjoy the nice geometric image of convex functions, and accordingly their significance has been discovered only gradually. The first class of submodular functions which was studied thoroughly was the class of matroid rank functions. Generalizing certain basic properties of matroid polyhedra, Edmonds (1970) began the systematical study of submodularity. Let us remark, however, that approaching from quite a different angle, Choquet (1955) also introduced these set-functions. He proved that the newtonian "capacity" of a subset of $\mathbb{R}^3$ defines a submodular set-function. Quite a few proof techniques in graph theory, but also in probability, geometry and lattice theory have made implicit use of submodularity of certain set-functions, thus forecasting the importance of this property.

In this paper we survey some of the most important aspects of submodularity. In particular, we shall see that submodularity shares the above-listed four

Properties of a Good Structure

L. Lovász



1983

Submodular functions and convexity

L. Lovász
Eötvös Loránd University, Department of Analysis I, Múzeum krt. 6–8, H-1088 Budapest, Hungary

0. Introduction

In "continuous" optimization convex functions play a central role. Besides elementary tools like differentiation, various methods for finding the minimum of a convex function constitute the main body of nonlinear optimization. But even linear programming may be viewed as the optimization of very special (linear) objective functions over very special convex domains (polyhedra). There are several reasons for this popularity of convex functions:

– Convex functions occur in many mathematical models in economy, engineering, and other sciencies. Convexity is a very natural property of various functions and domains occuring in such models; quite often the only non-trivial property which can be stated in general.

– Convexity is preserved under many natural operations and transformations, and thereby the effective range of results can be extended, elegant proof techniques can be developed as well as unforeseen applications of certain results can be given.

– Convex functions and domains exhibit sufficient structure so that a mathematically beautiful and practically useful theory can be developed.

– There are theoretically and practically (reasonably) efficient methods to find the minimum of a convex function.

It is less apperant, but we claim and hope to prove to a certain extent, that a similar role is played in discrete optimization by *submodular set-functions*. These functions do not enjoy the nice geometric image of convex functions, and accordingly their significance has been discovered only gradually. The first class of submodular functions which was studied thoroughly was the class of matroid rank functions. Generalizing certain basic properties of matroid polyhedra, Edmonds (1970) began the systematical study of submodularity. Let us remark, however, that approaching from quite a different angle, Choquet (1955) also introduced these set-functions. He proved that the newtonian "capacity" of a subset of $\mathbb{R}^3$ defines a submodular set-function. Quite a few proof techniques in graph theory, but also in probability, geometry and lattice theory have made implicit use of submodularity of certain set-functions, thus forecasting the importance of this property.

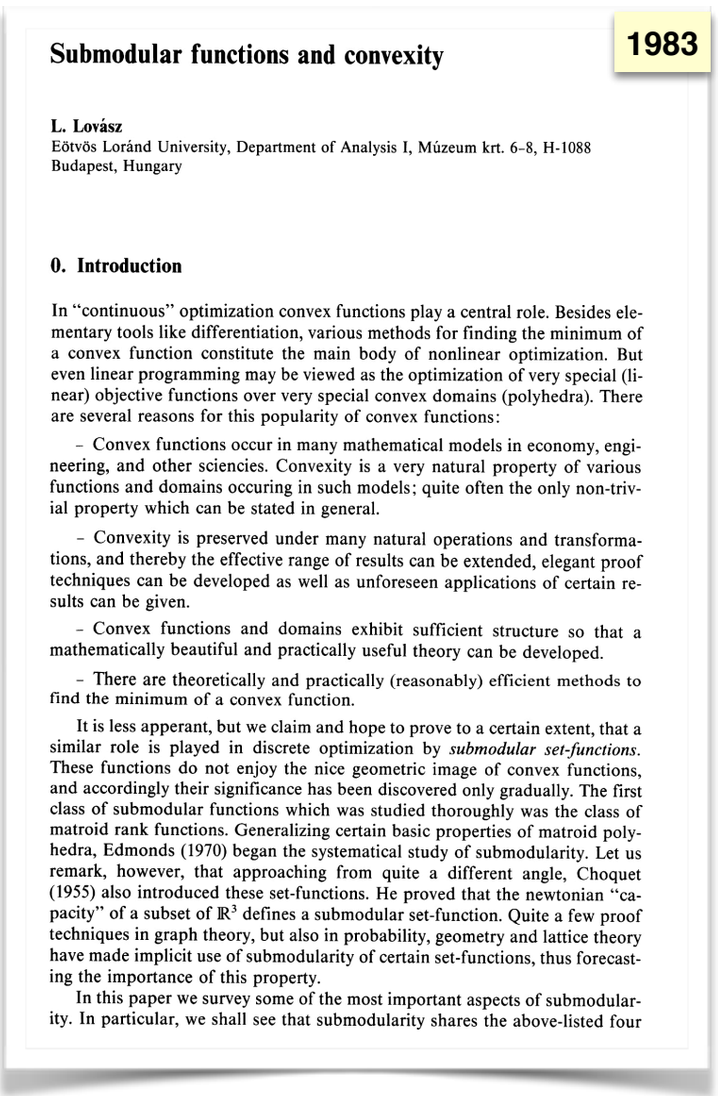
In this paper we survey some of the most important aspects of submodularity. In particular, we shall see that submodularity shares the above-listed four

Nontrivial property that can be stated in general

Preserved under many operations and transformations

Properties of a Good Structure

L. Lovász



1983

Submodular functions and convexity

L. Lovász
Eötvös Loránd University, Department of Analysis I, Múzeum krt. 6–8, H-1088
Budapest, Hungary

0. Introduction

In "continuous" optimization convex functions play a central role. Besides elementary tools like differentiation, various methods for finding the minimum of a convex function constitute the main body of nonlinear optimization. But even linear programming may be viewed as the optimization of very special (linear) objective functions over very special convex domains (polyhedra). There are several reasons for this popularity of convex functions:

– Convex functions occur in many mathematical models in economy, engineering, and other sciencies. Convexity is a very natural property of various functions and domains occuring in such models; quite often the only non-trivial property which can be stated in general.

– Convexity is preserved under many natural operations and transformations, and thereby the effective range of results can be extended, elegant proof techniques can be developed as well as unforeseen applications of certain results can be given.

– Convex functions and domains exhibit sufficient structure so that a mathematically beautiful and practically useful theory can be developed.

– There are theoretically and practically (reasonably) efficient methods to find the minimum of a convex function.

It is less apperant, but we claim and hope to prove to a certain extent, that a similar role is played in discrete optimization by *submodular set-functions*. These functions do not enjoy the nice geometric image of convex functions, and accordingly their significance has been discovered only gradually. The first class of submodular functions which was studied thoroughly was the class of matroid rank functions. Generalizing certain basic properties of matroid polyhedra, Edmonds (1970) began the systematical study of submodularity. Let us remark, however, that approaching from quite a different angle, Choquet (1955) also introduced these set-functions. He proved that the newtonian "capacity" of a subset of $\mathbb{R}^3$ defines a submodular set-function. Quite a few proof techniques in graph theory, but also in probability, geometry and lattice theory have made implicit use of submodularity of certain set-functions, thus forecasting the importance of this property.

In this paper we survey some of the most important aspects of submodularity. In particular, we shall see that submodularity shares the above-listed four

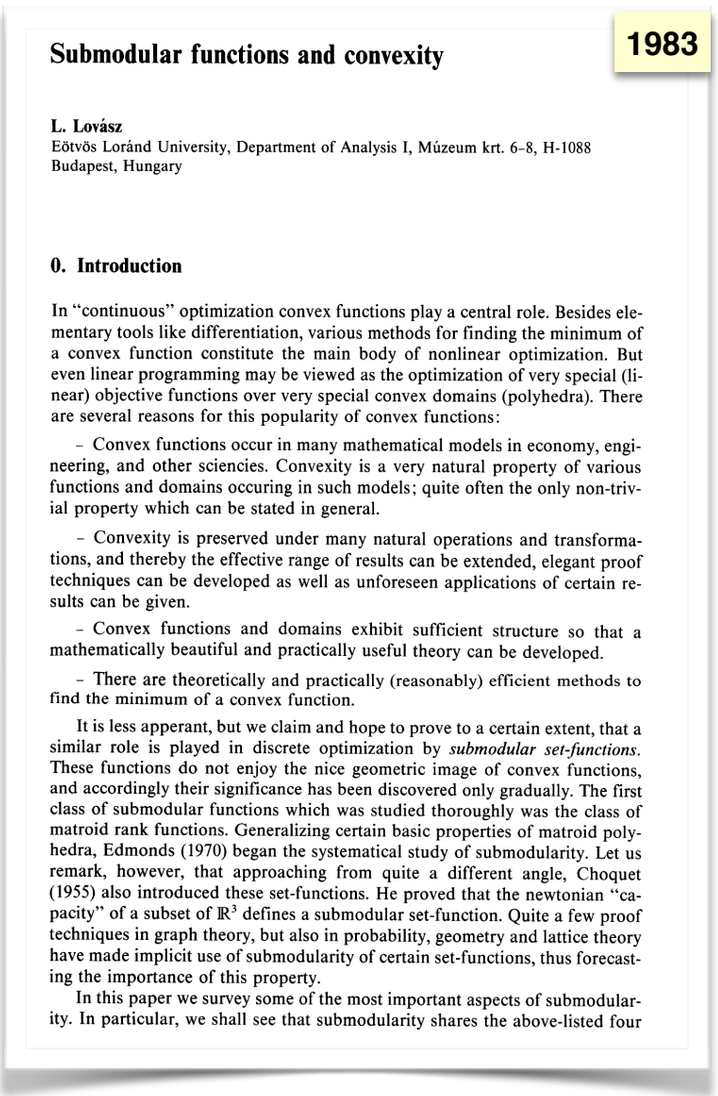
Nontrivial property that can be stated in general

Preserved under many operations and transformations

Sufficient structure for a practically useful theory

Properties of a Good Structure

L. Lovász



1983

Submodular functions and convexity

L. Lovász
Eötvös Loránd University, Department of Analysis I, Múzeum krt. 6–8, H-1088 Budapest, Hungary

0. Introduction

In "continuous" optimization convex functions play a central role. Besides elementary tools like differentiation, various methods for finding the minimum of a convex function constitute the main body of nonlinear optimization. But even linear programming may be viewed as the optimization of very special (linear) objective functions over very special convex domains (polyhedra). There are several reasons for this popularity of convex functions:

– Convex functions occur in many mathematical models in economy, engineering, and other sciencies. Convexity is a very natural property of various functions and domains occuring in such models; quite often the only non-trivial property which can be stated in general.

– Convexity is preserved under many natural operations and transformations, and thereby the effective range of results can be extended, elegant proof techniques can be developed as well as unforeseen applications of certain results can be given.

– Convex functions and domains exhibit sufficient structure so that a mathematically beautiful and practically useful theory can be developed.

– There are theoretically and practically (reasonably) efficient methods to find the minimum of a convex function.

It is less apperant, but we claim and hope to prove to a certain extent, that a similar role is played in discrete optimization by *submodular set-functions*. These functions do not enjoy the nice geometric image of convex functions, and accordingly their significance has been discovered only gradually. The first class of submodular functions which was studied thoroughly was the class of matroid rank functions. Generalizing certain basic properties of matroid polyhedra, Edmonds (1970) began the systematical study of submodularity. Let us remark, however, that approaching from quite a different angle, Choquet (1955) also introduced these set-functions. He proved that the newtonian "capacity" of a subset of $\mathbb{R}^3$ defines a submodular set-function. Quite a few proof techniques in graph theory, but also in probability, geometry and lattice theory have made implicit use of submodularity of certain set-functions, thus forecasting the importance of this property.

In this paper we survey some of the most important aspects of submodularity. In particular, we shall see that submodularity shares the above-listed four

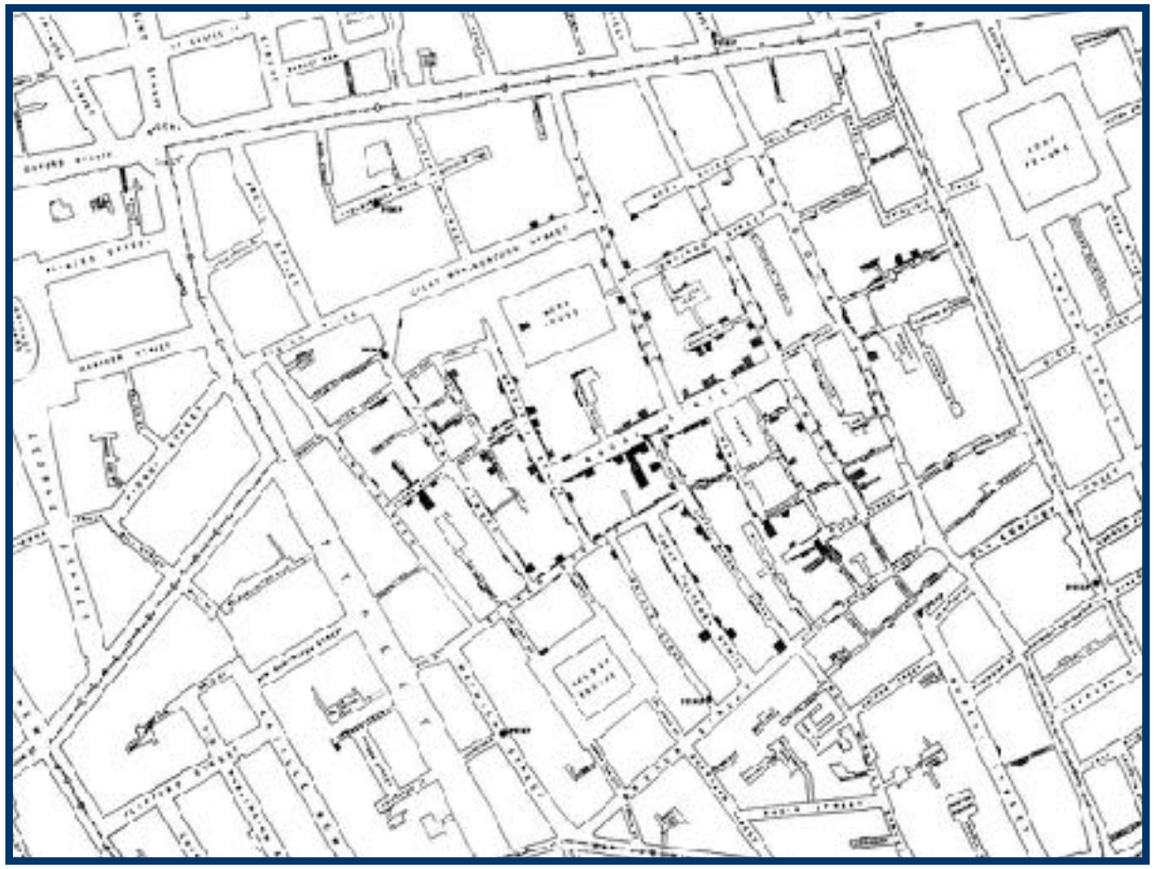
Nontrivial property that can be stated in general

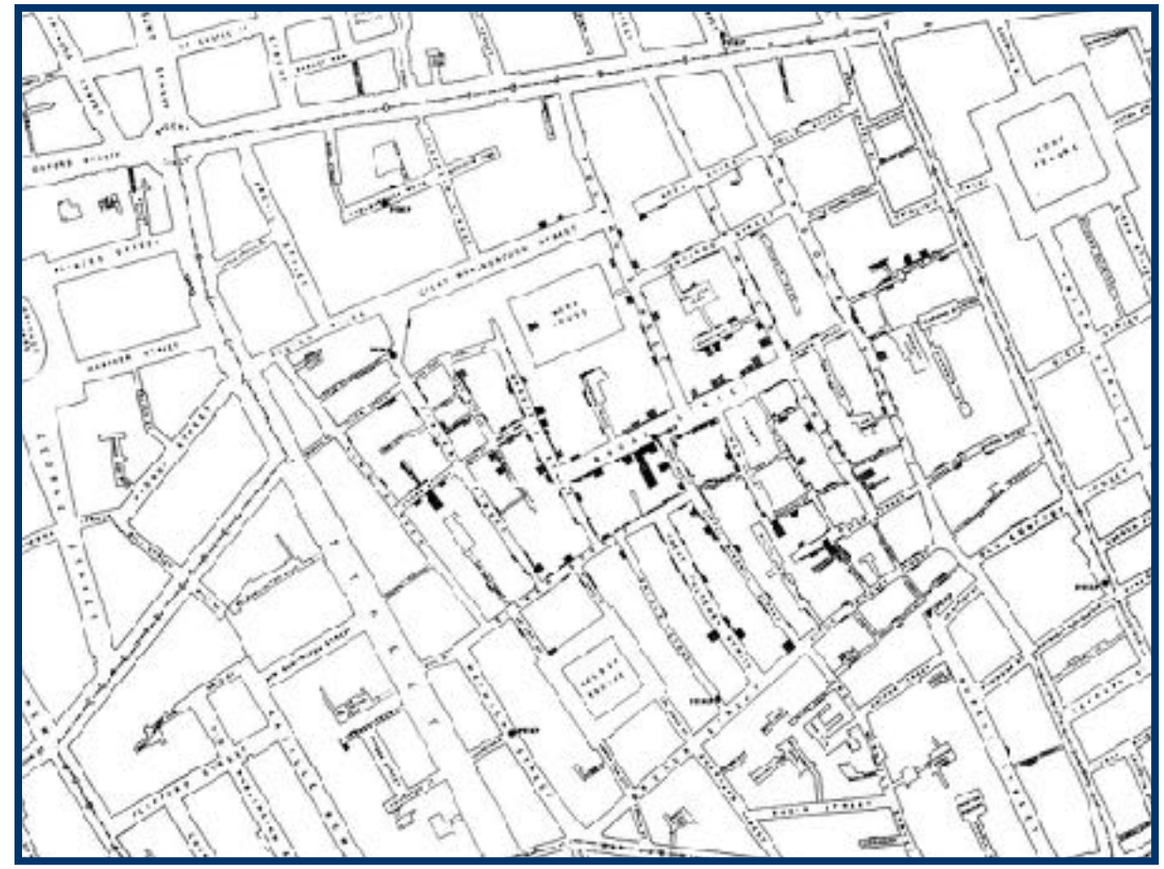
Preserved under many operations and transformations

Sufficient structure for a practically useful theory

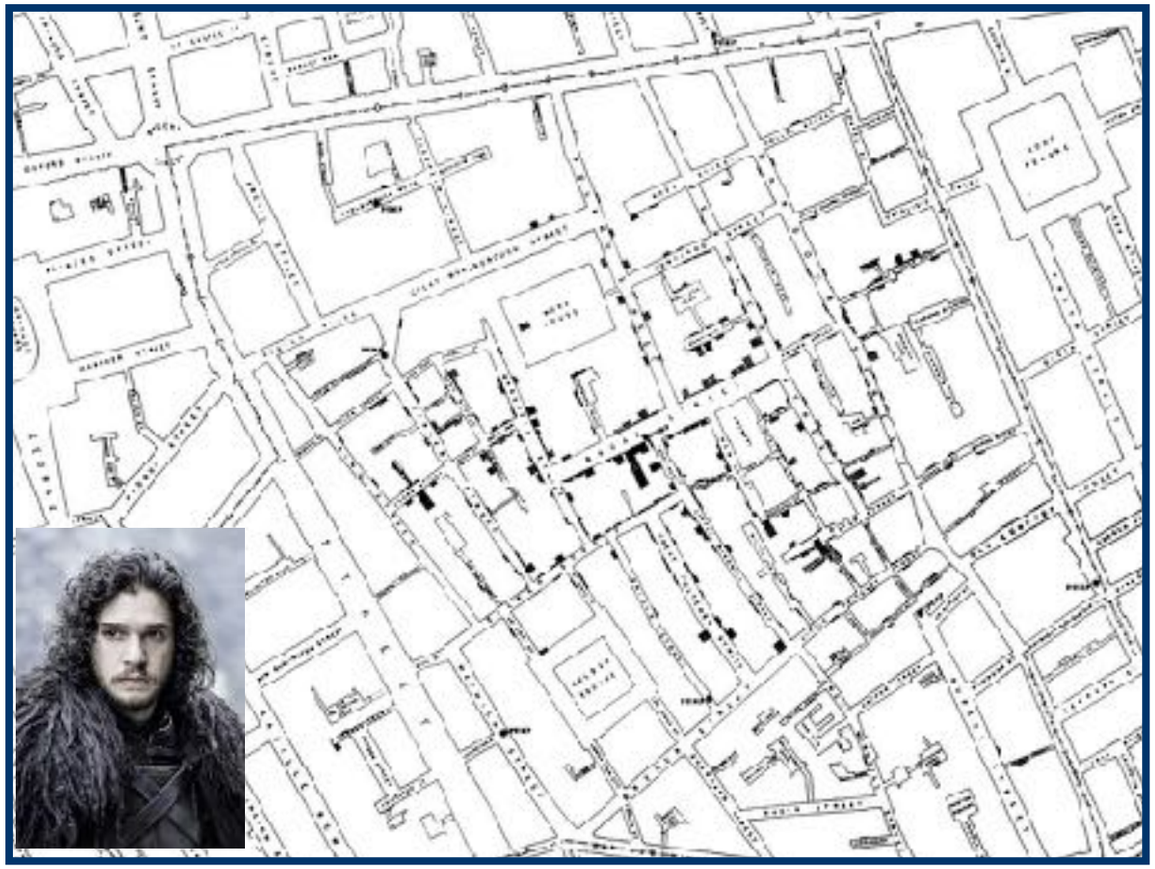
Continuous Optimization	Discrete Optimization
↓	↓
Convexity	Submodularity

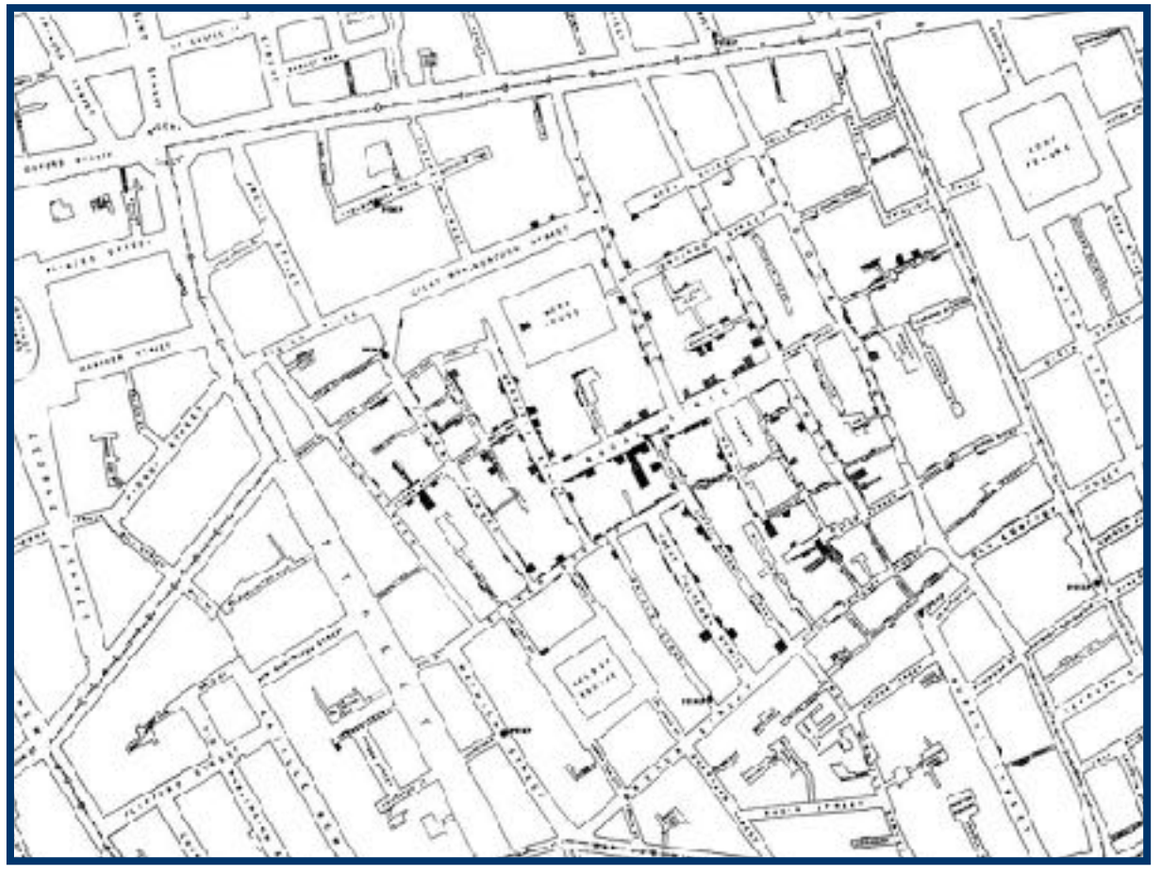
Jon Snow and London Cholera Epidemic



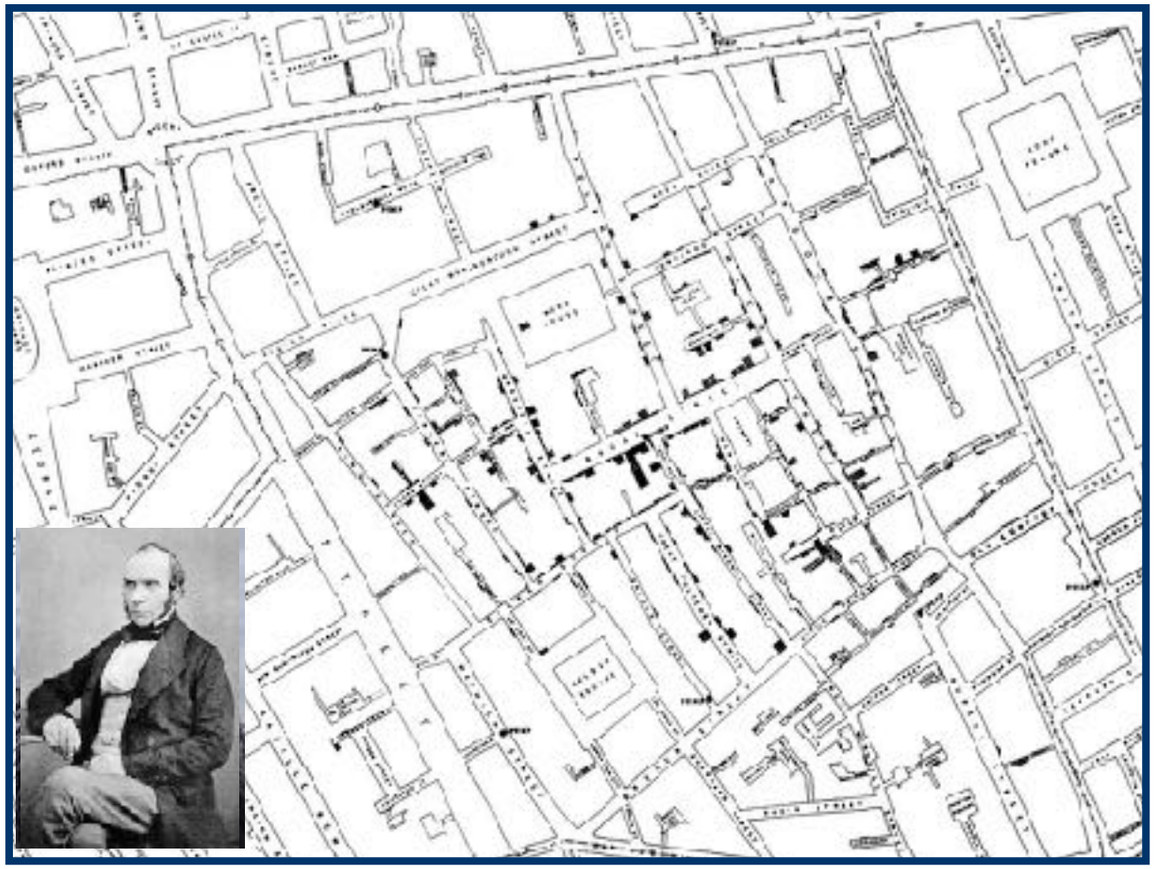


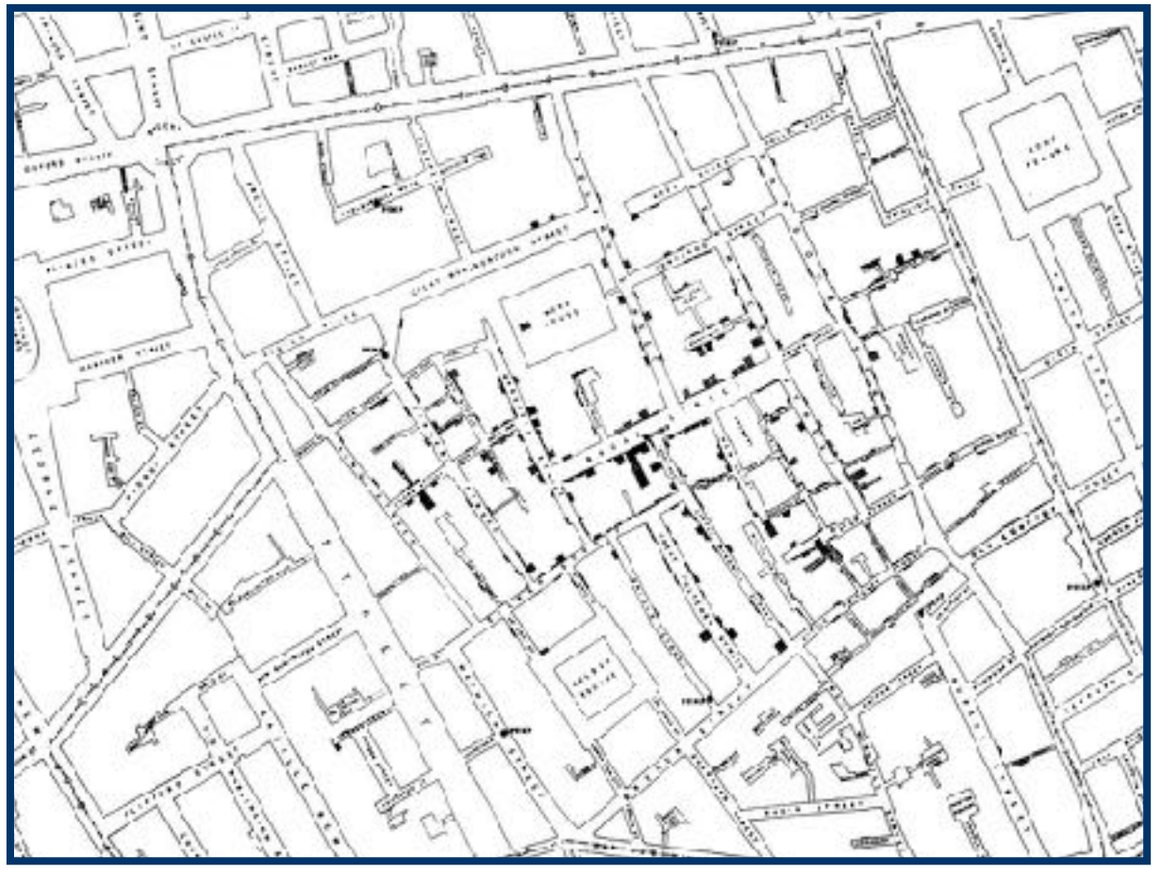
Jon Snow and London Cholera Epidemic



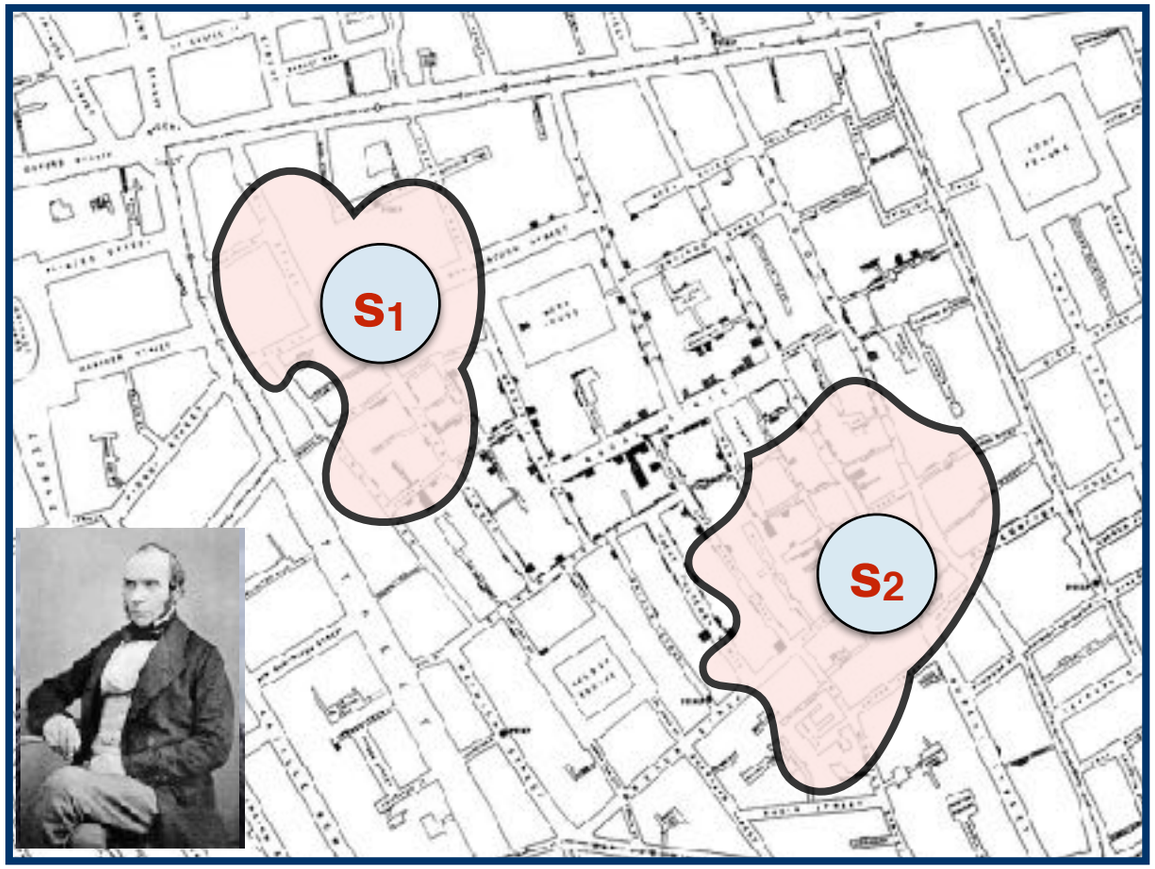


Jon Snow and London Cholera Epidemic





Jon Snow and London Cholera Epidemic

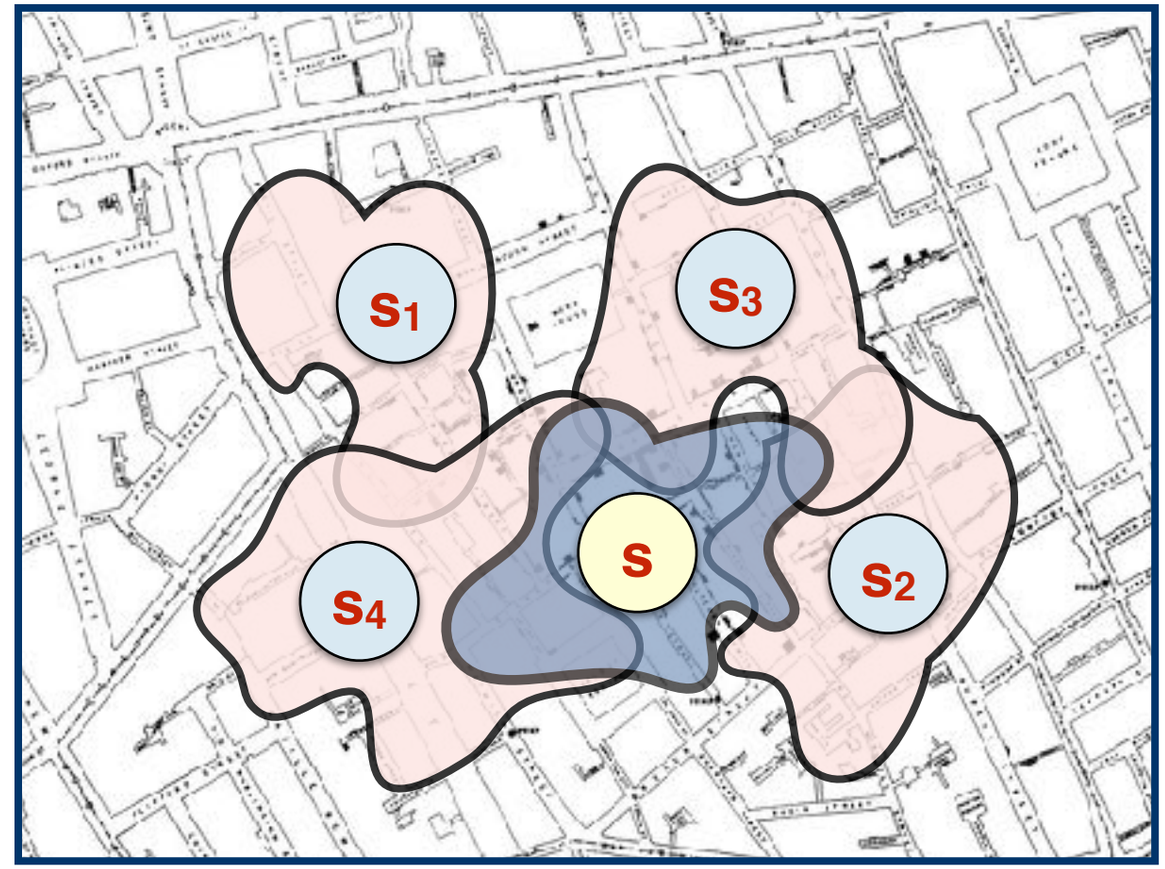


$$A = \{s_1, s_2\}$$

$$B = \{s_1, s_2, s_3, s_4\}$$

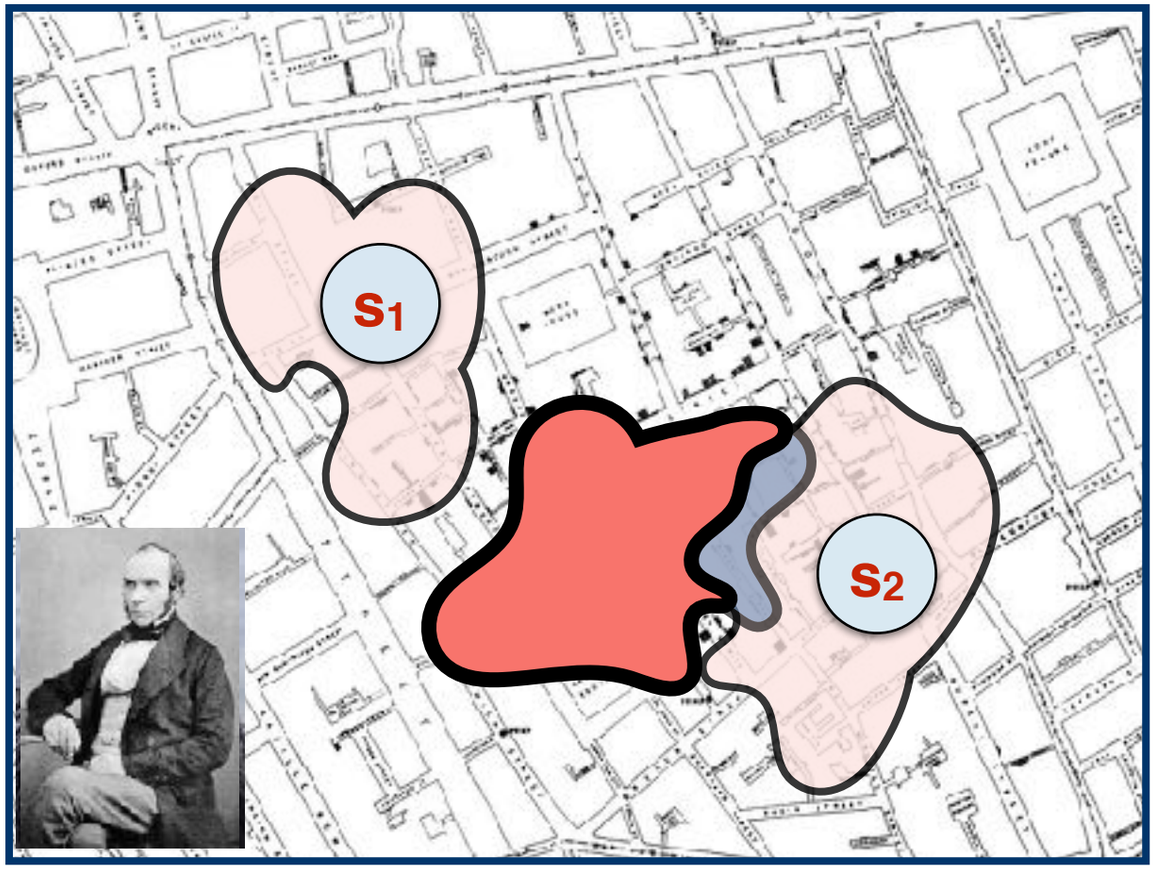
Jon Snow and London Cholera Epidemic

$$A = \{s_1, s_2\}$$

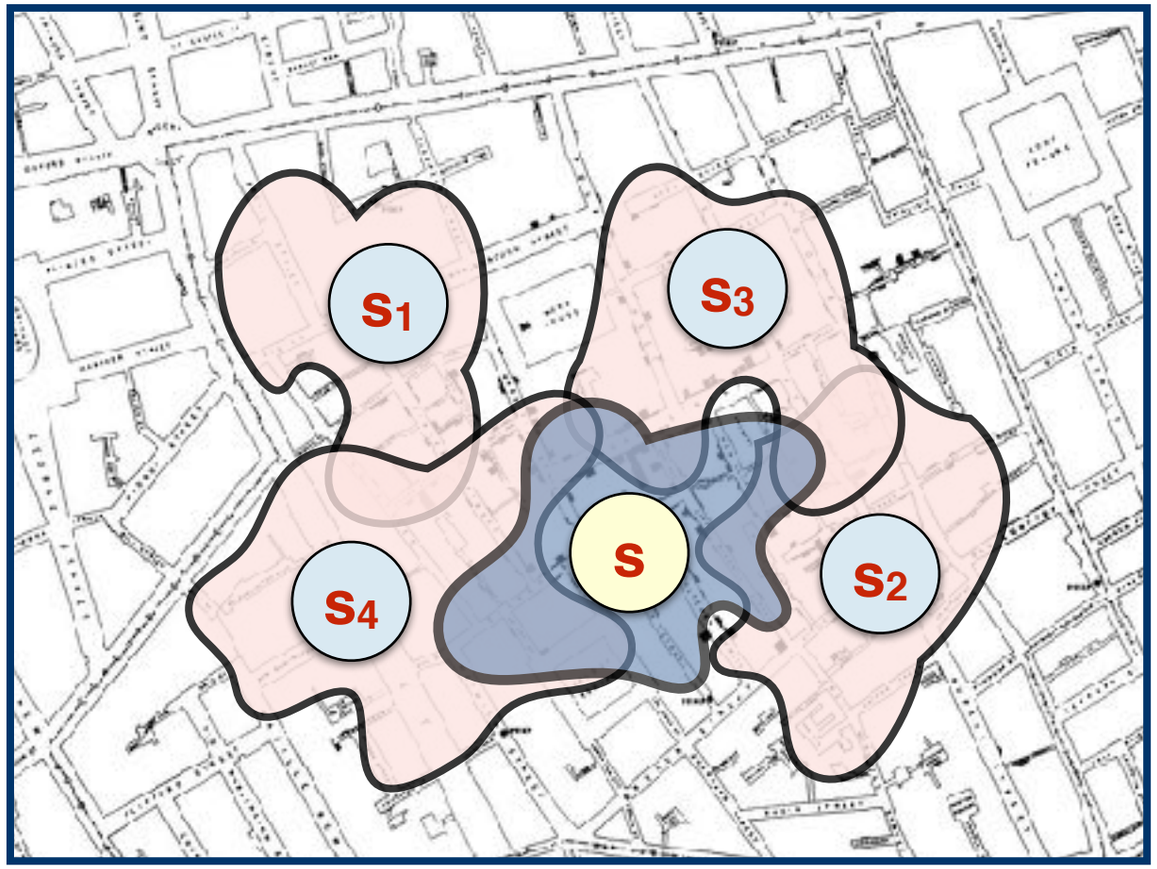


$$B = \{s_1, s_2, s_3, s_4\}$$

Jon Snow and London Cholera Epidemic

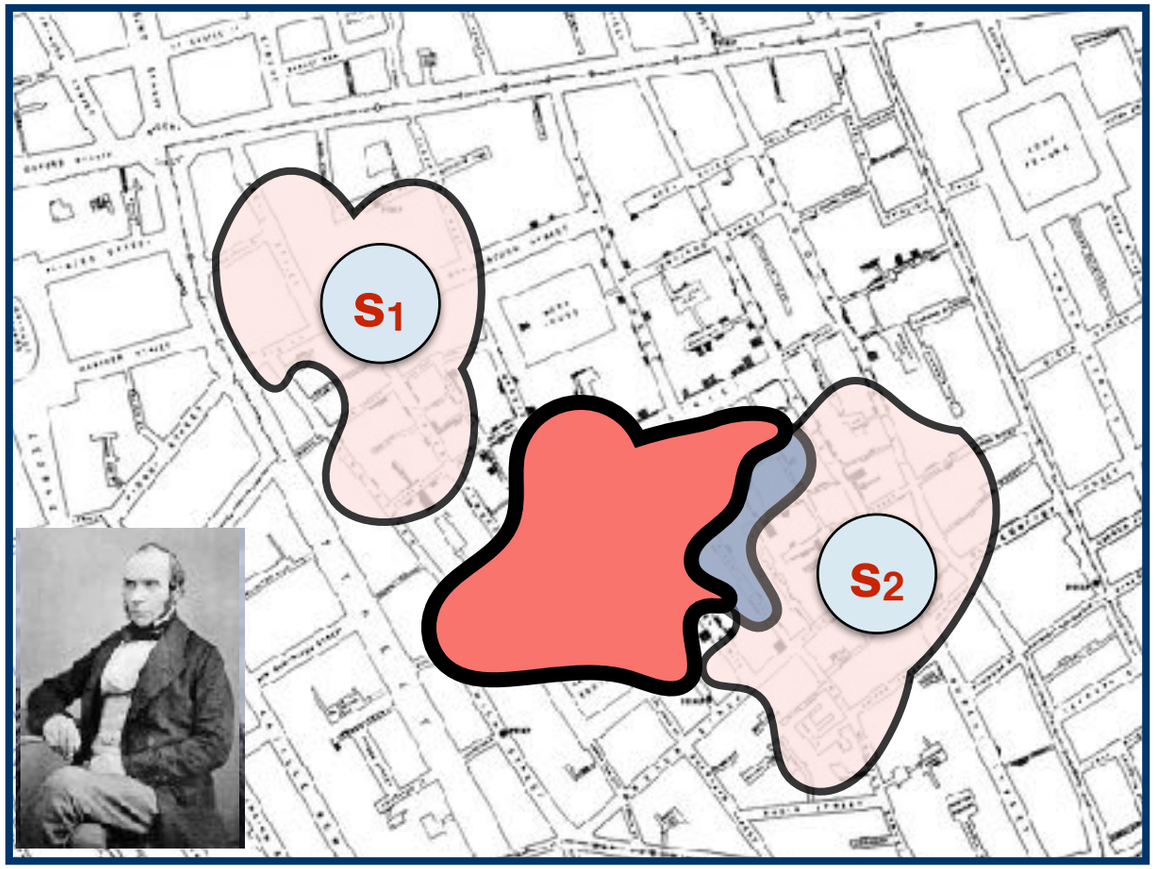


$$A = \{s_1, s_2\}$$

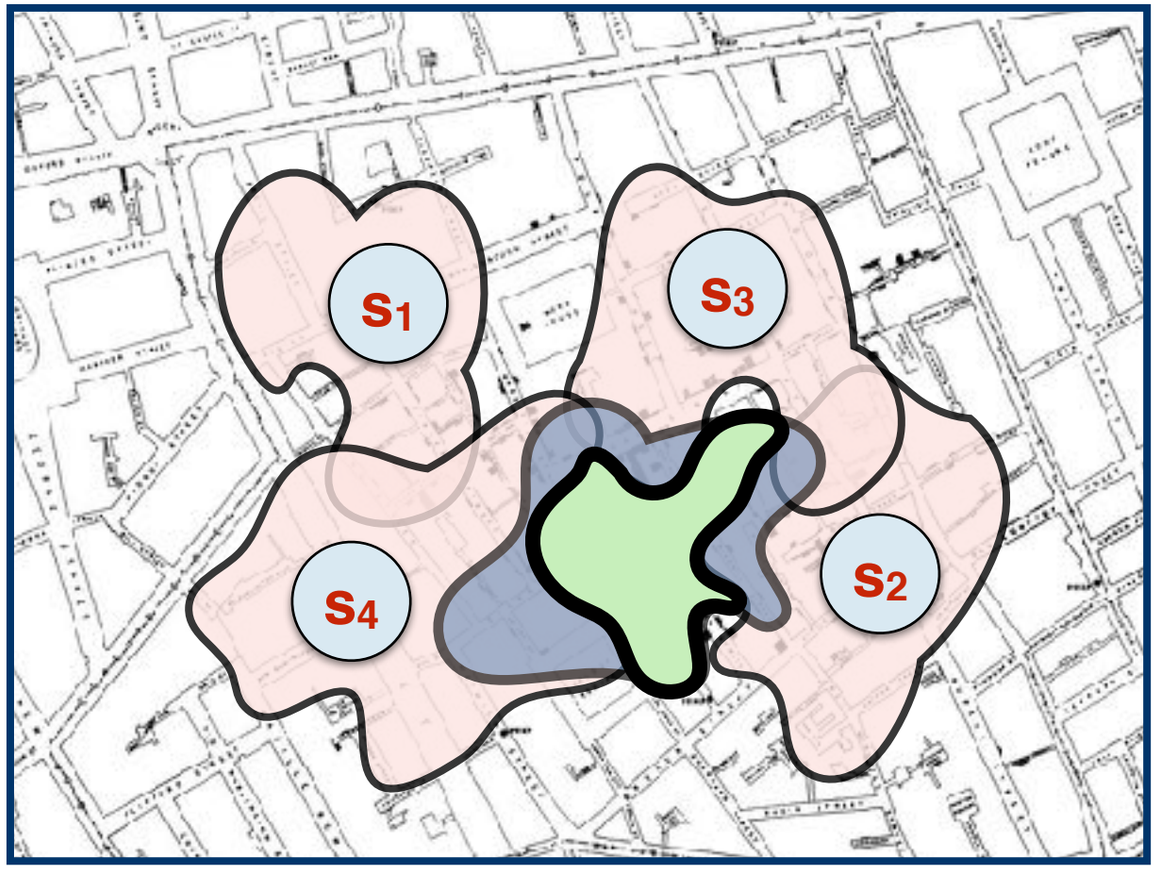


$$B = \{s_1, s_2, s_3, s_4\}$$

Jon Snow and London Cholera Epidemic

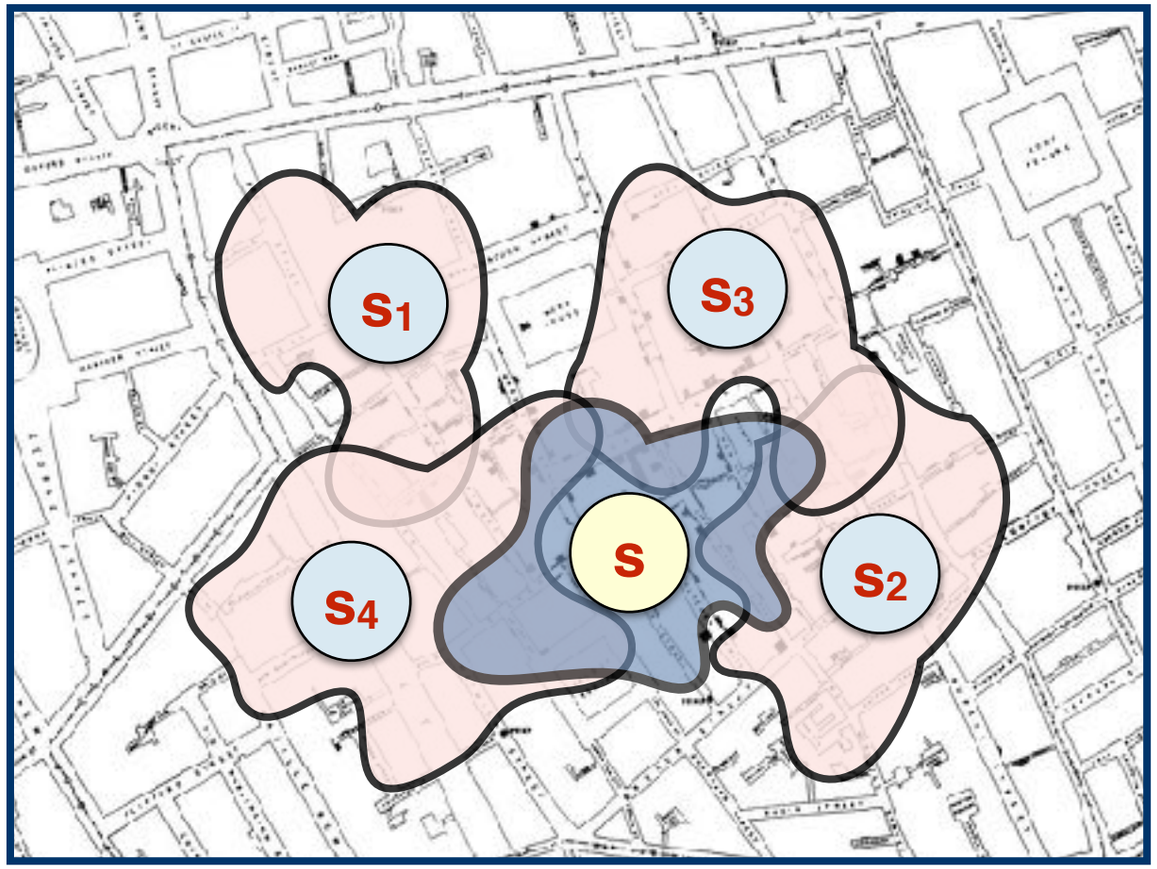


$$A = \{s_1, s_2\}$$

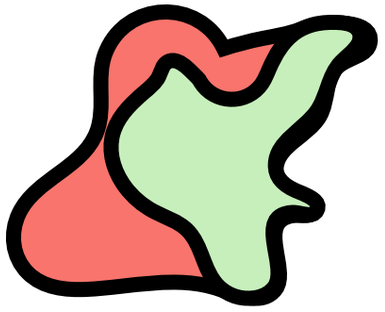


$$B = \{s_1, s_2, s_3, s_4\}$$

Jon Snow and London Cholera Epidemic

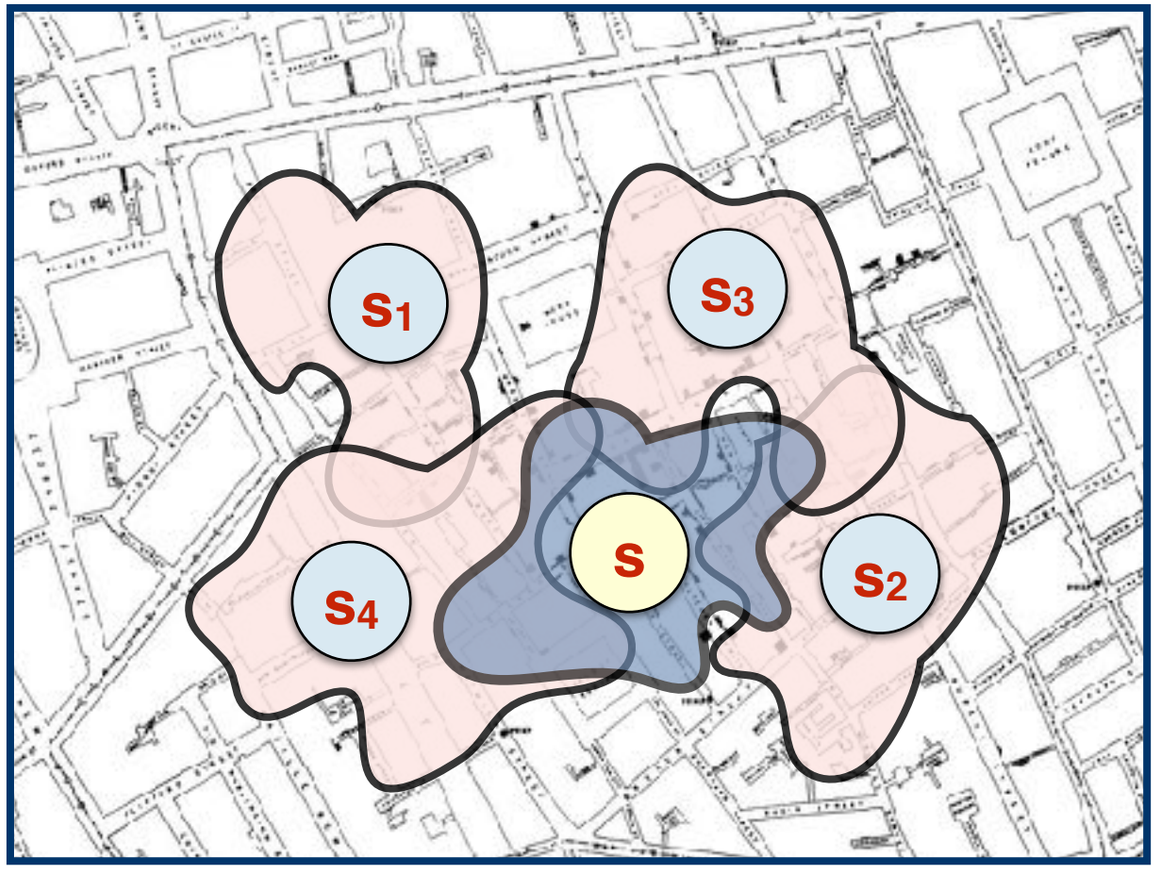


$$A = \{s_1, s_2\}$$

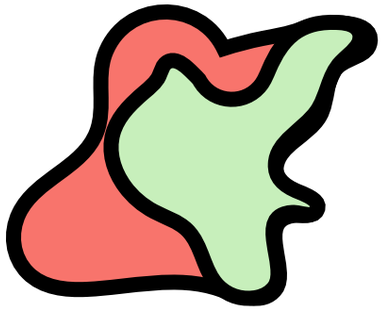


$$B = \{s_1, s_2, s_3, s_4\}$$

Jon Snow and London Cholera Epidemic



$A = \{s_1, s_2\}$



$B = \{s_1, s_2, s_3, s_4\}$

Submodularity

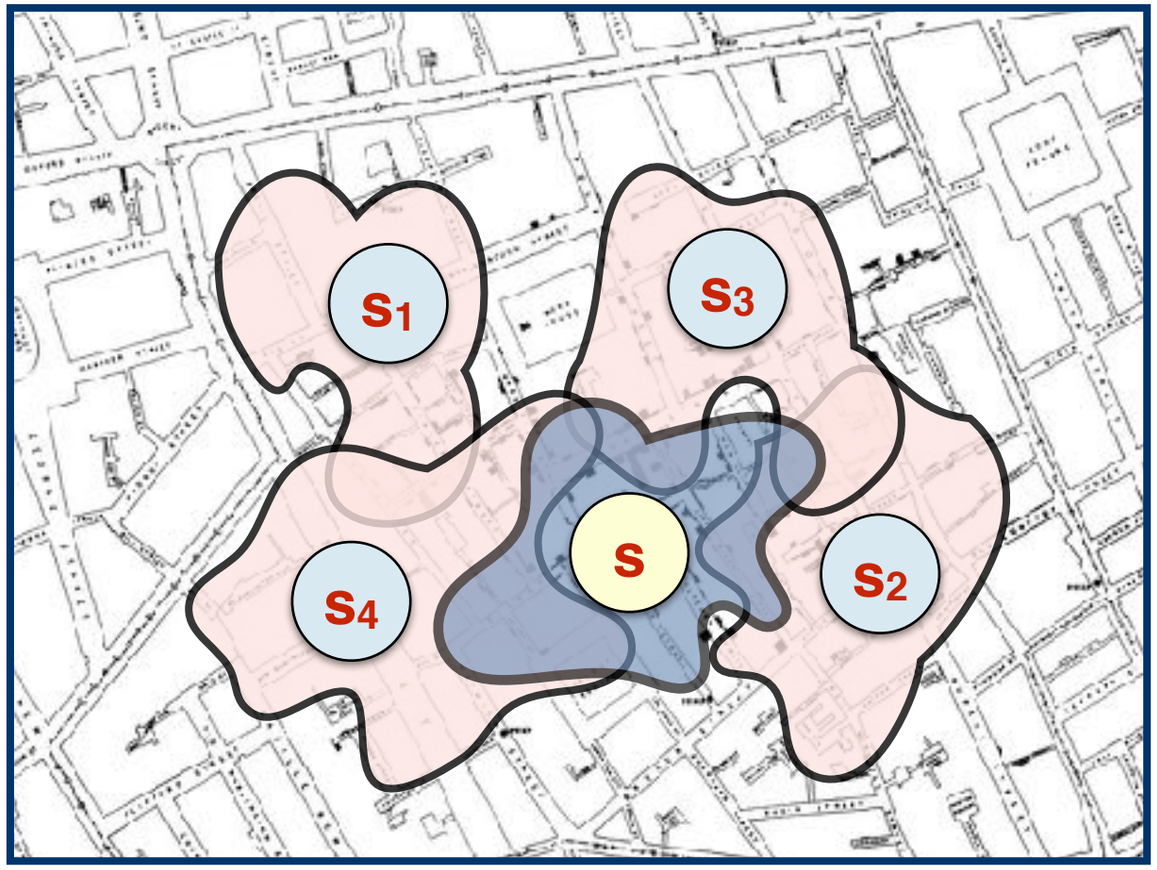
$$\forall\ A \subseteq B \subseteq V \text{ and } s \in V \setminus B$$

$$f(A \cup \{s\}) - f(A) \geq f(B \cup \{s\}) - f(B)$$

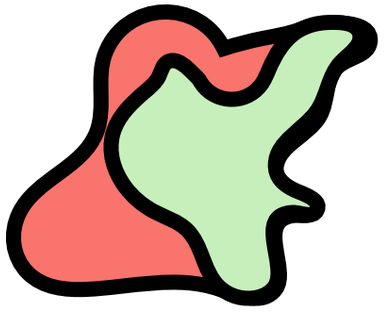
Yale

Penn

Jon Snow and London Cholera Epidemic



$A = \{s_1, s_2\}$



$B = \{s_1, s_2, s_3, s_4\}$

Submodularity

$$\forall\ A \subseteq B \subseteq V \text{ and } s \in V \setminus B$$

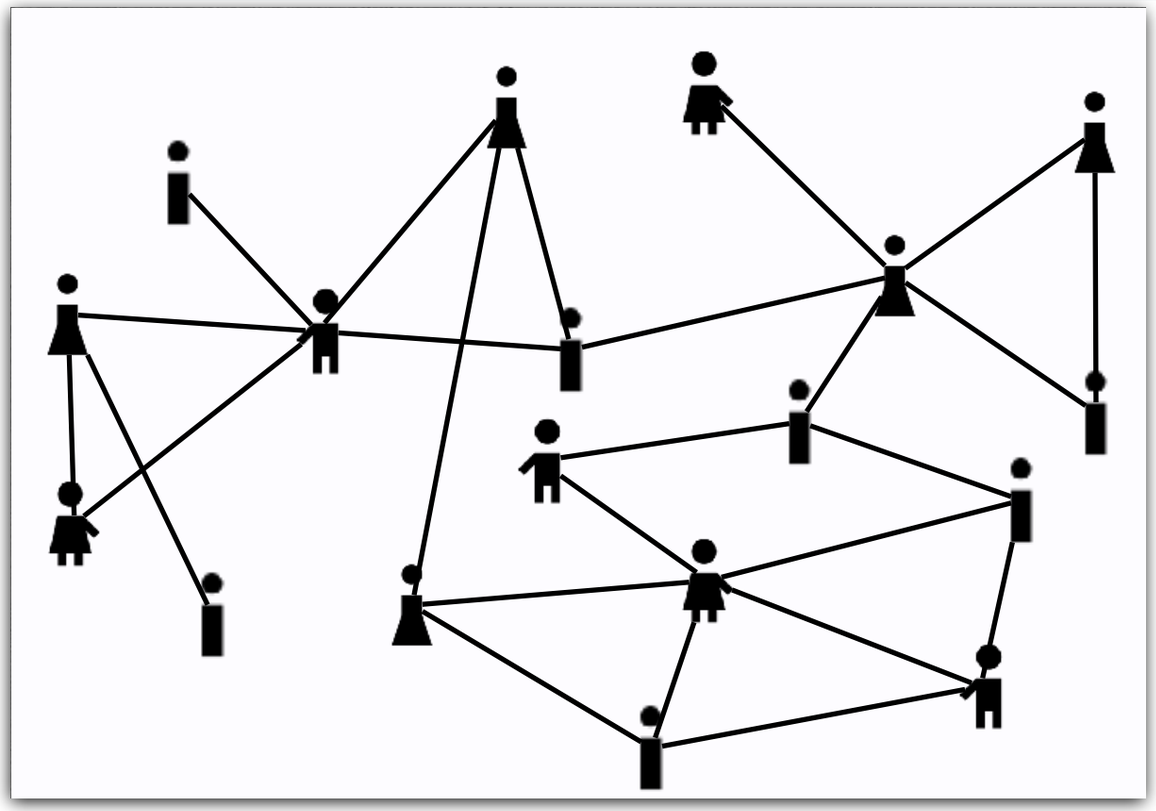
$$f(A \cup \{s\}) - f(A) \geq f(B \cup \{s\}) - f(B)$$

Monotonicity

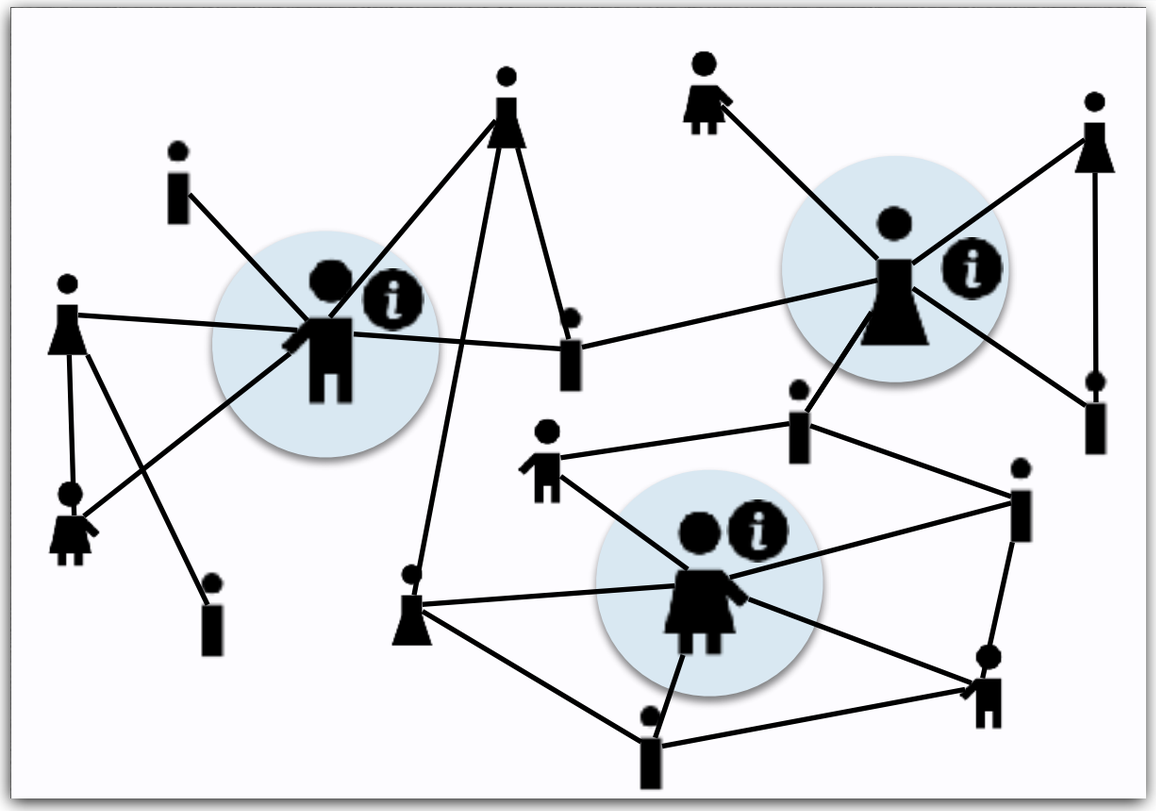
$$\forall A \subseteq B \subseteq V$$

$$f(A) \leq f(B)$$

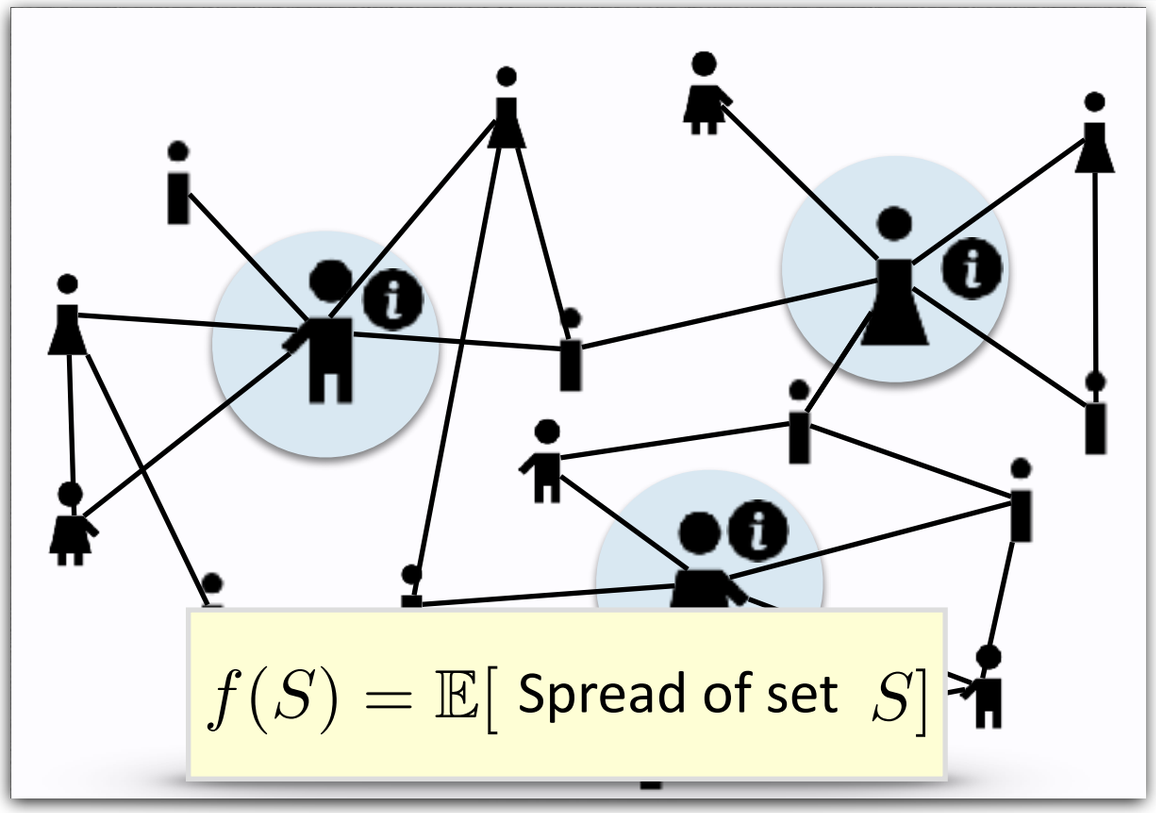
Examples



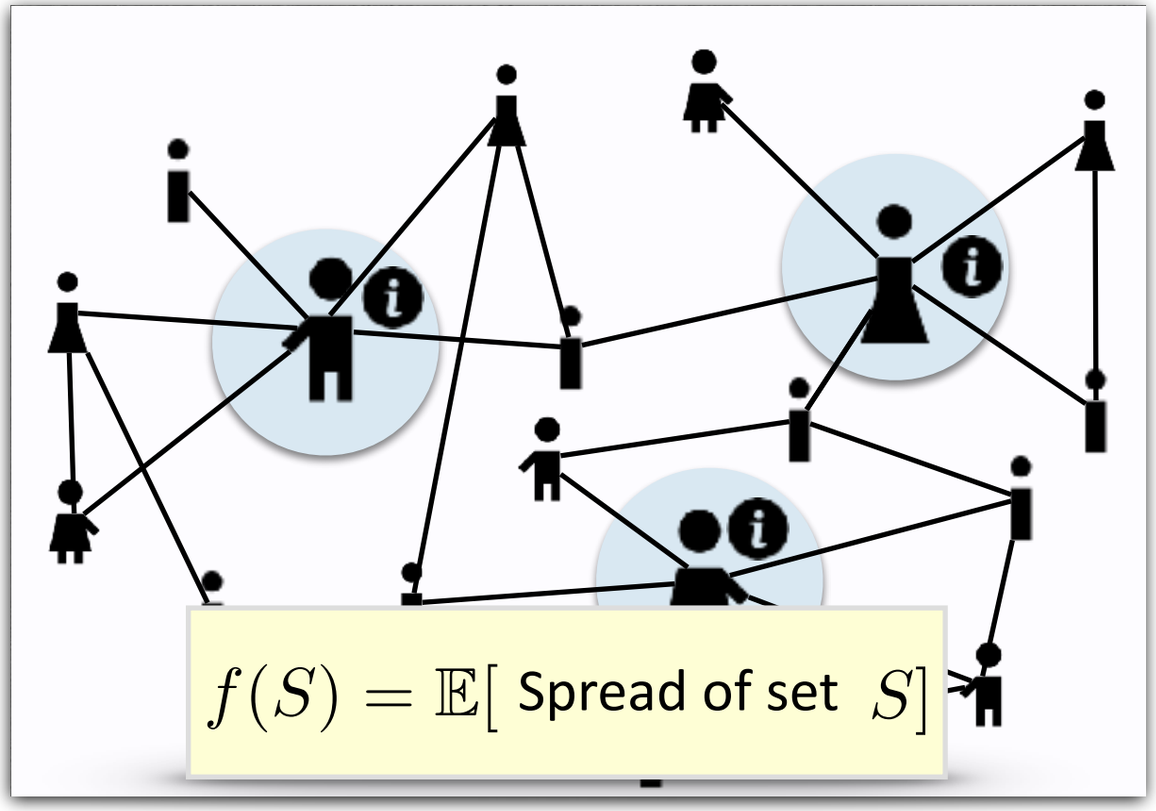
Examples

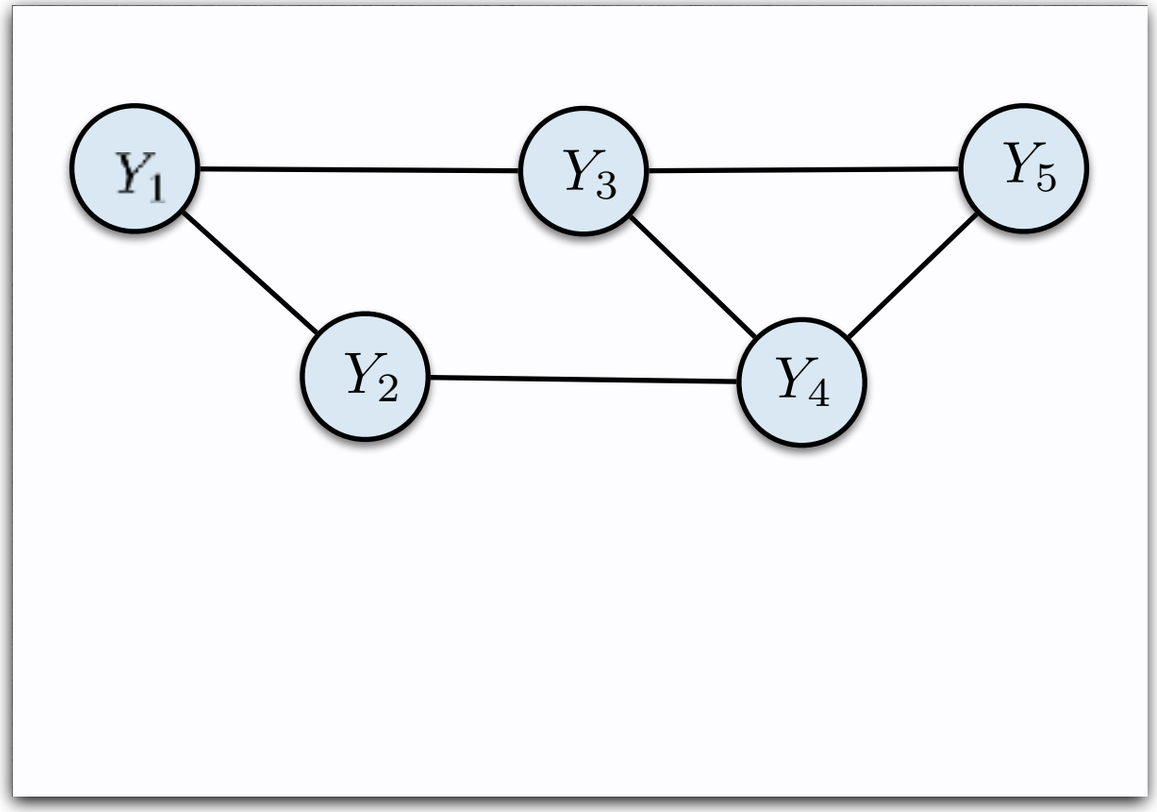


Examples

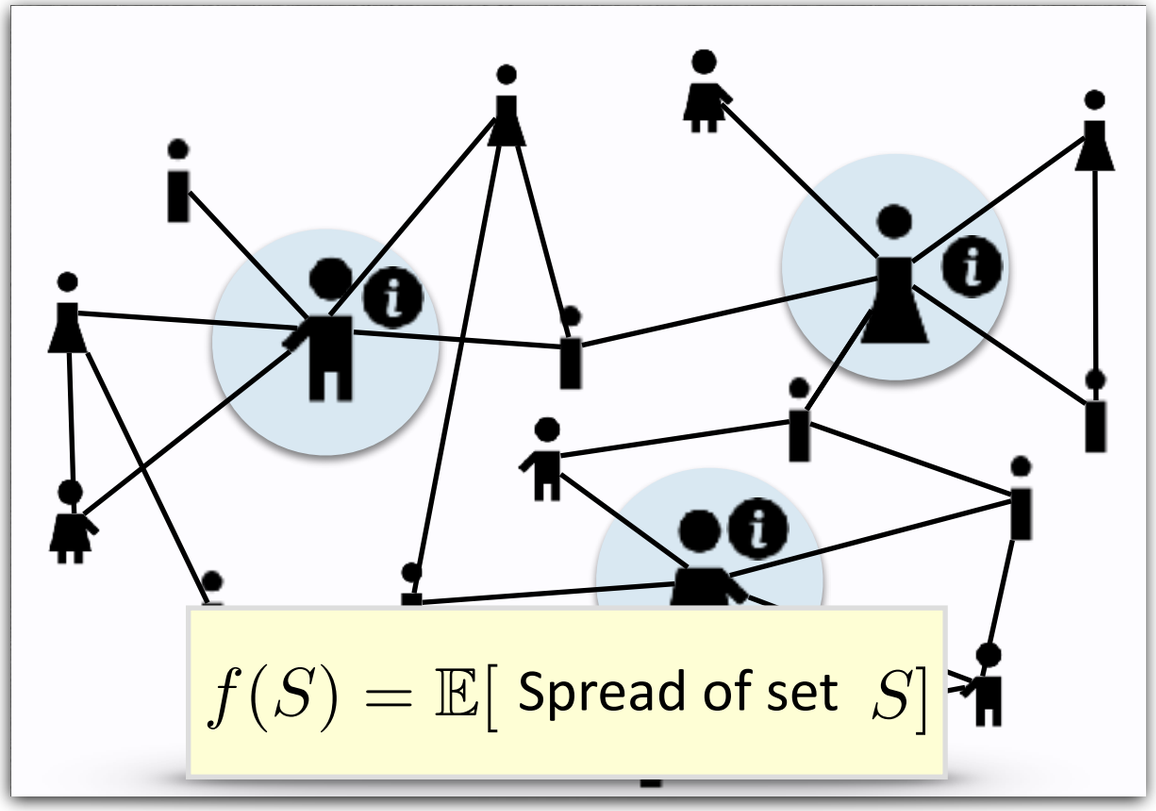


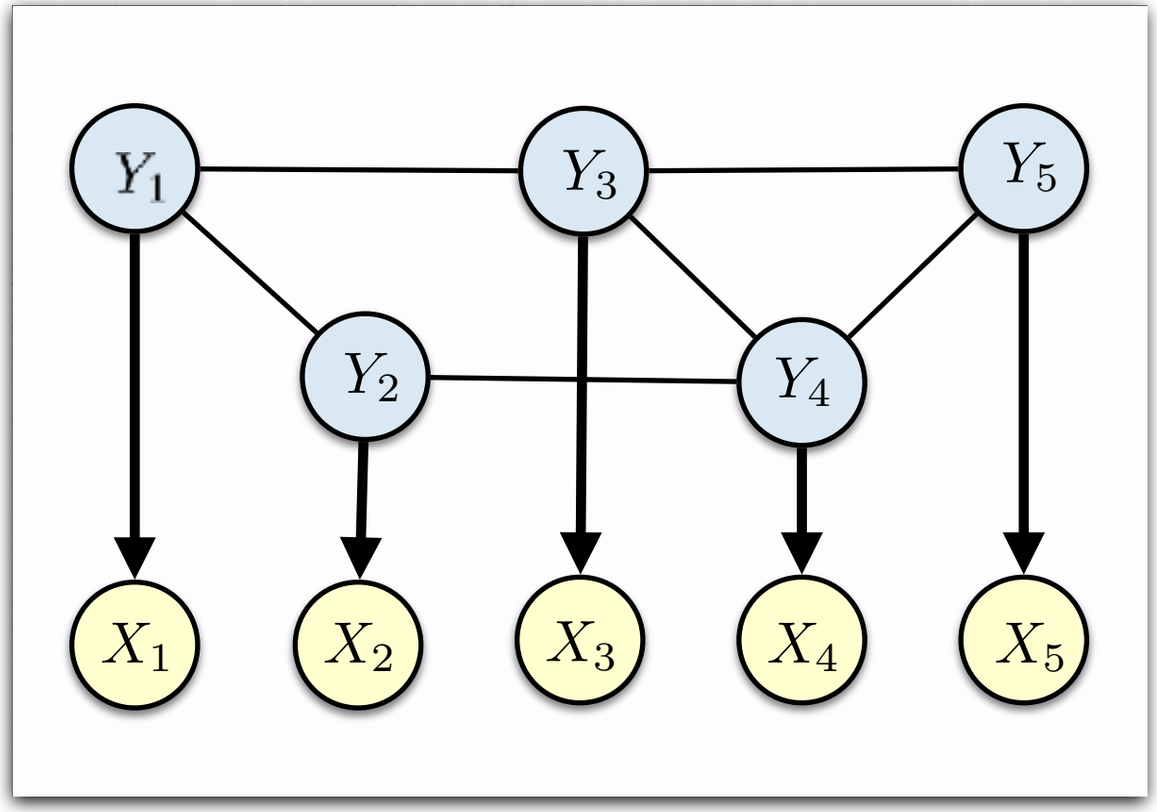
Examples



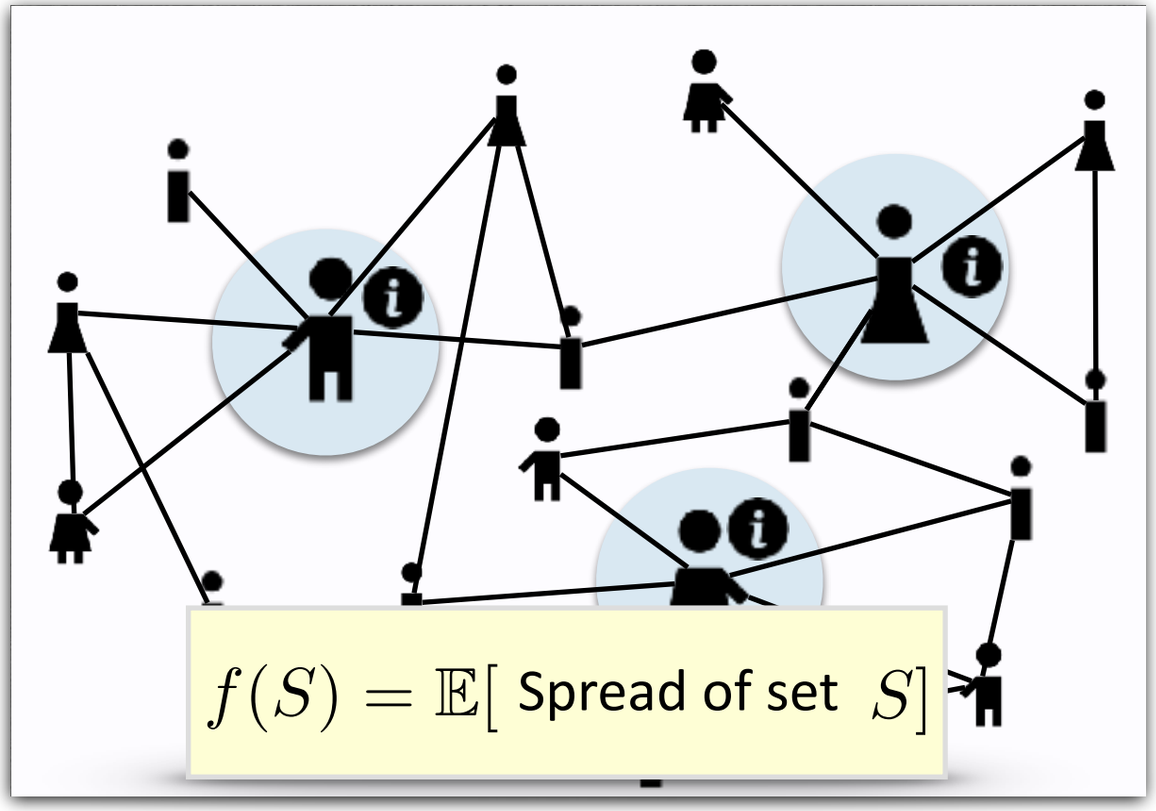


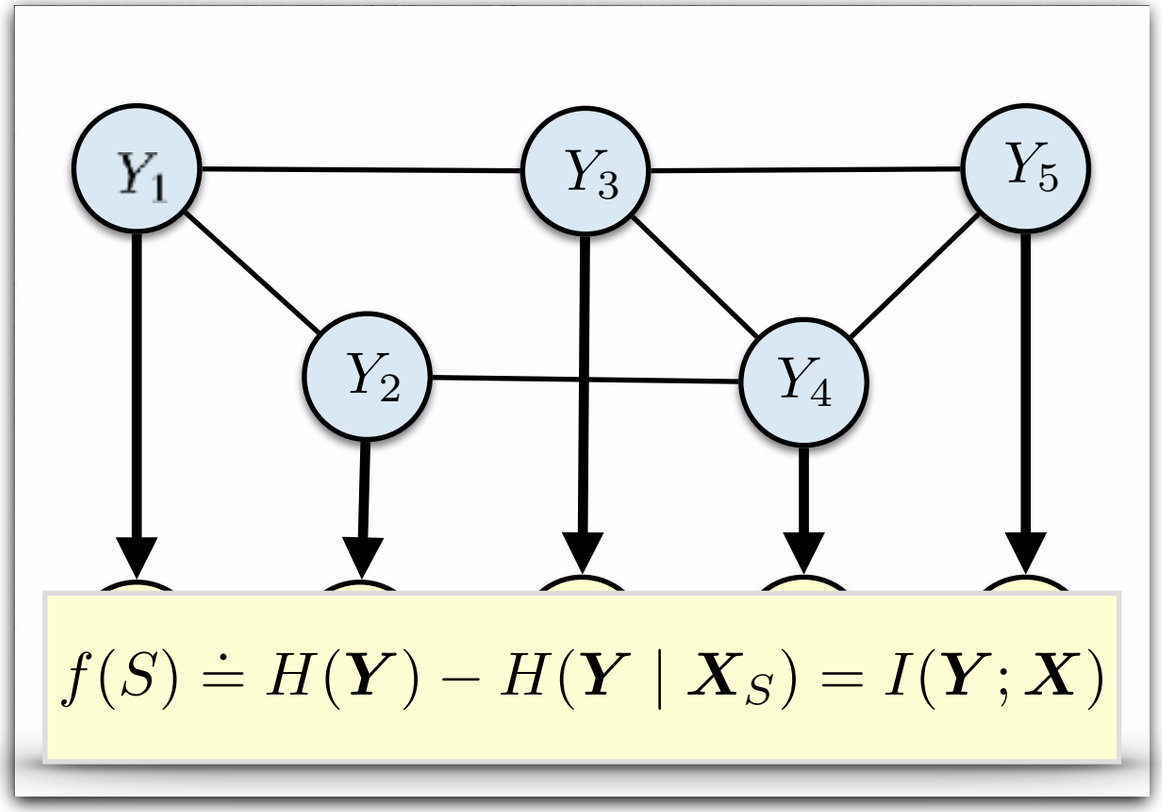
Examples



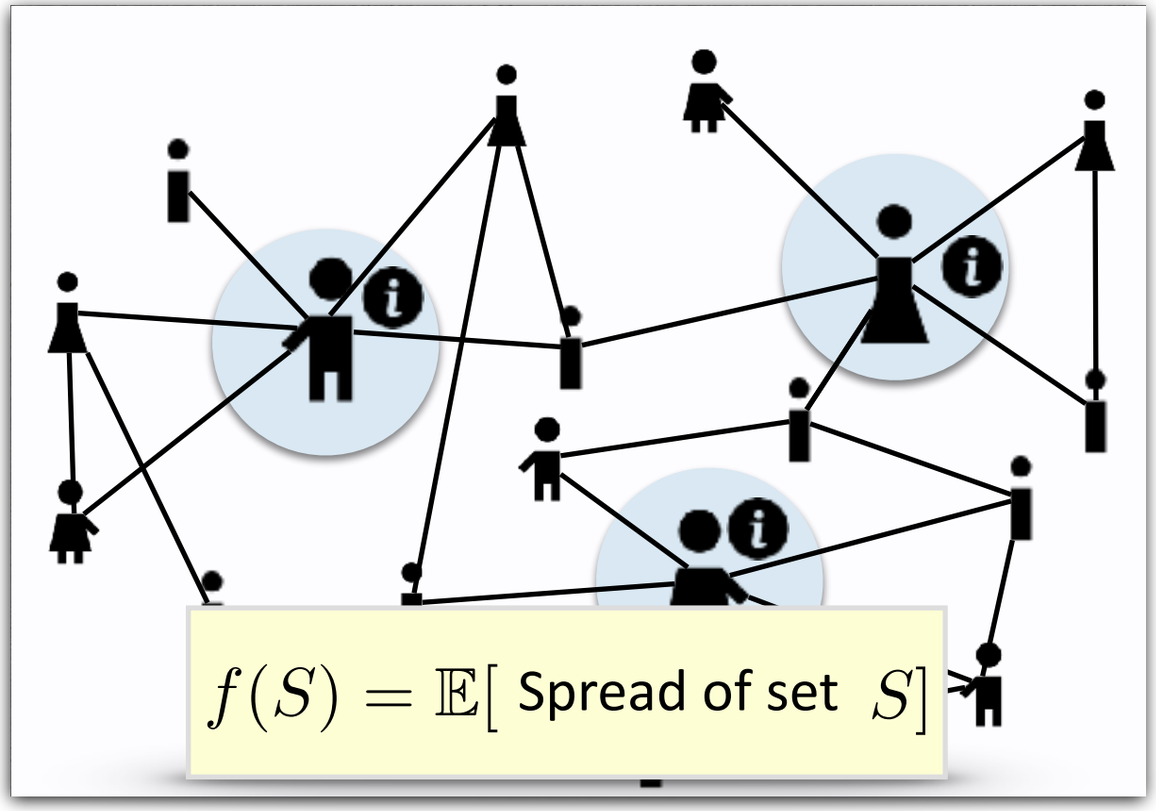


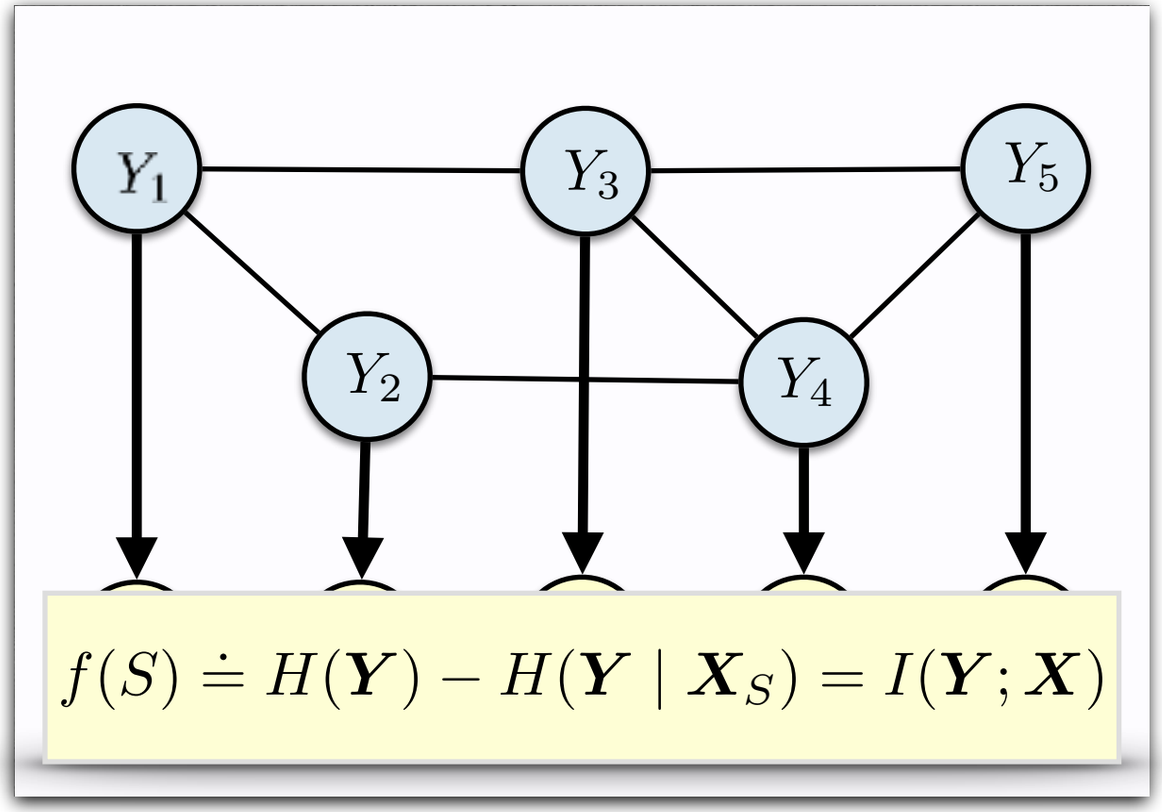
Examples

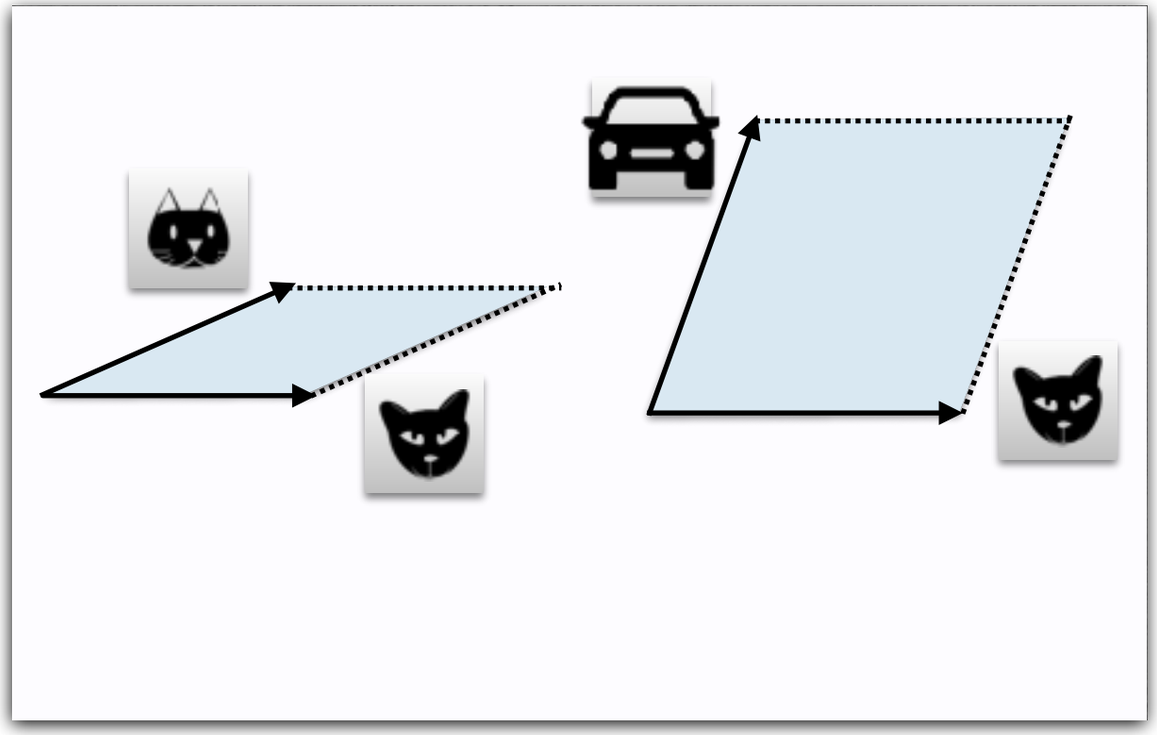

$f(S) = \mathbb{E}[$ Spread of set $S]$


Y_1
Y_2
Y_3
Y_4
Y_5
$f(S) \doteq H(\boldsymbol{Y}) - H(\boldsymbol{Y} \mid \boldsymbol{X}_S) = I(\boldsymbol{Y}; \boldsymbol{X})$

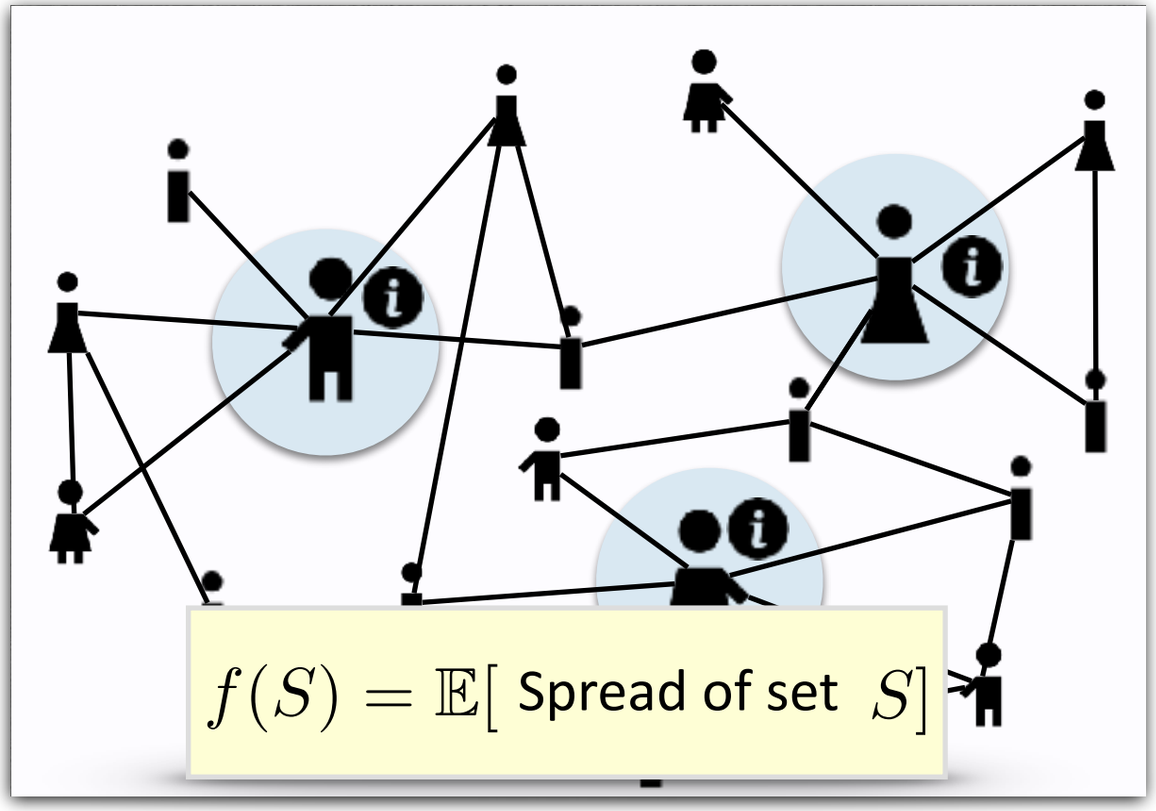
Examples

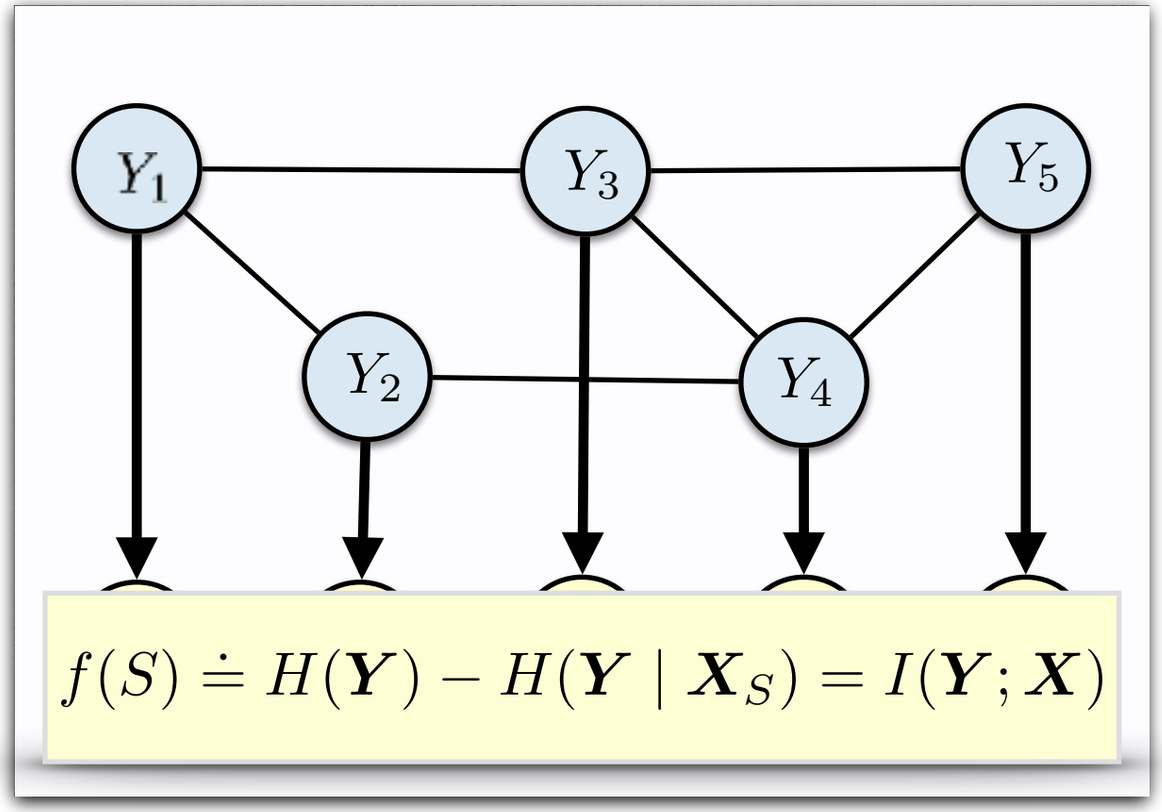

$f(S) = \mathbb{E}[$ Spread of set $S]$

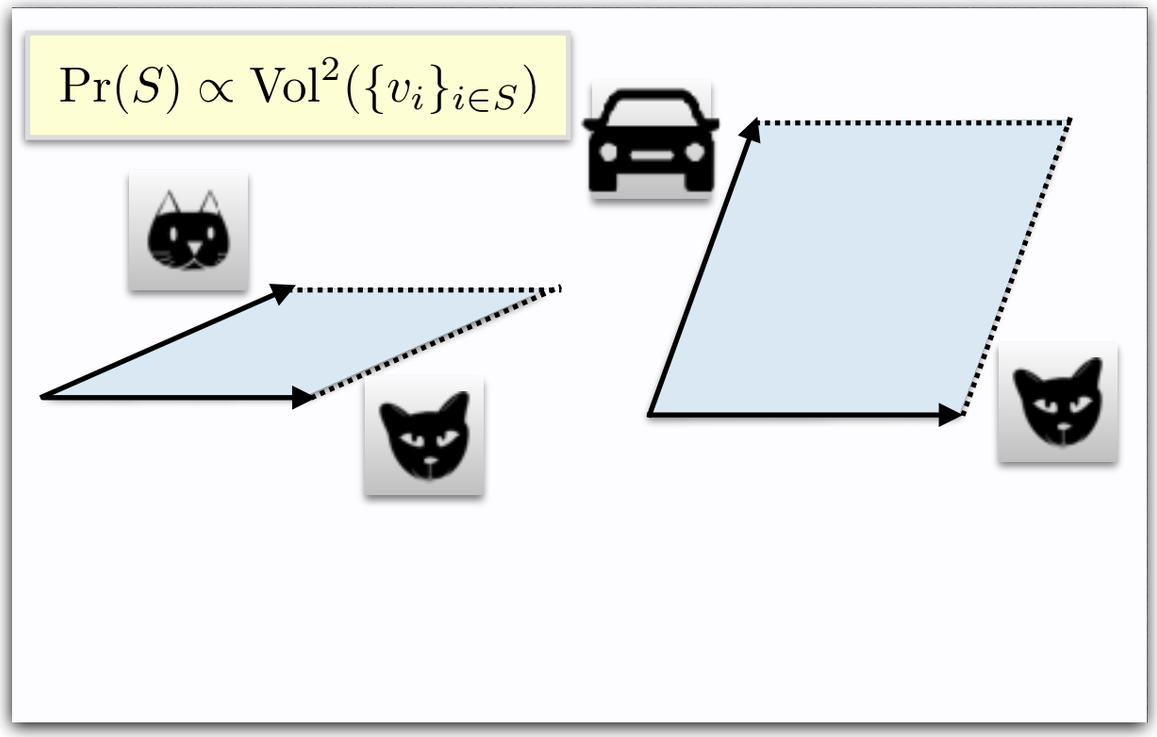

Y_1
Y_2
Y_3
Y_4
Y_5
$f(S) \doteq H(\boldsymbol{Y}) - H(\boldsymbol{Y} \mid \boldsymbol{X}_S) = I(\boldsymbol{Y}; \boldsymbol{X})$



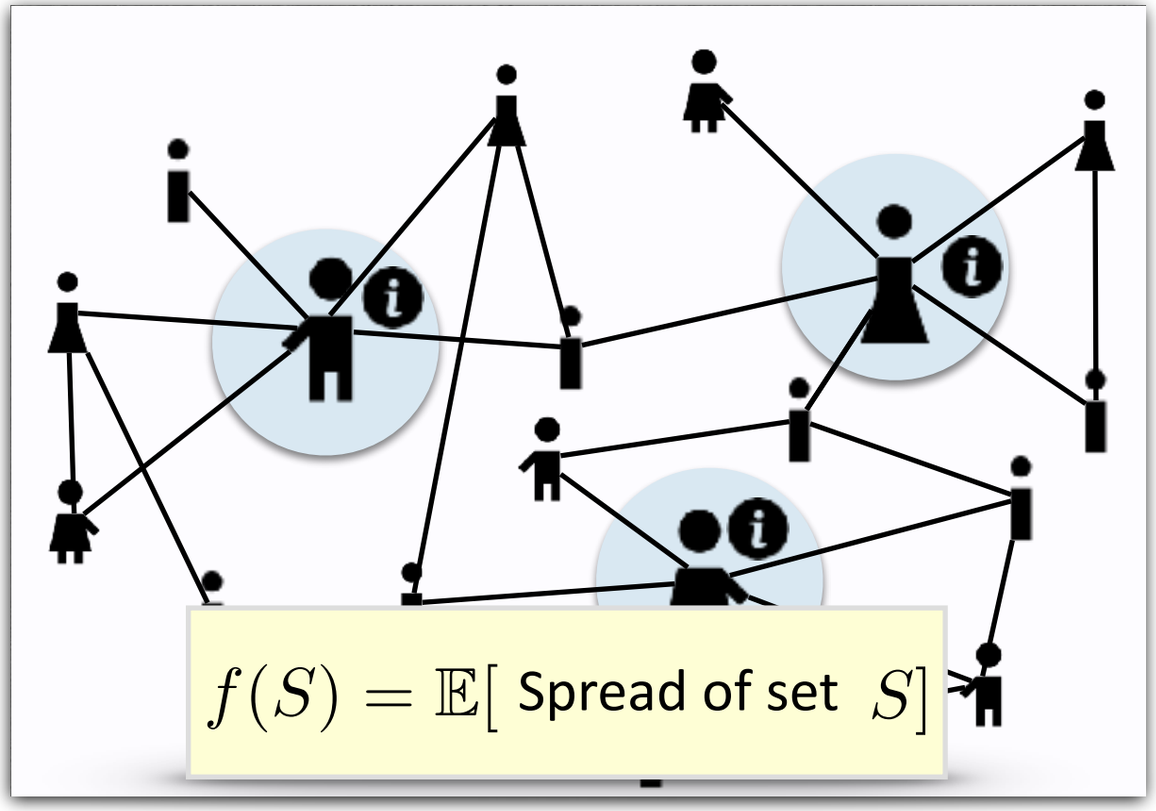
Examples

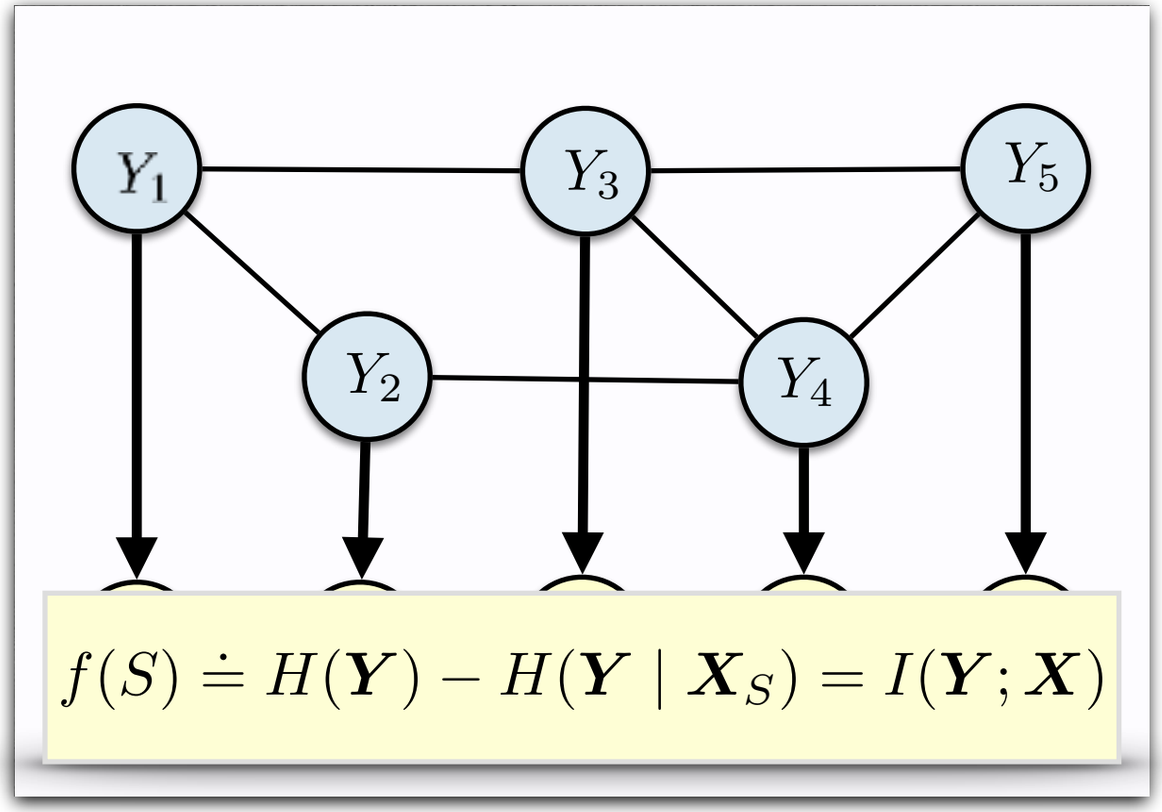

$f(S) = \mathbb{E}[$ Spread of set $S]$

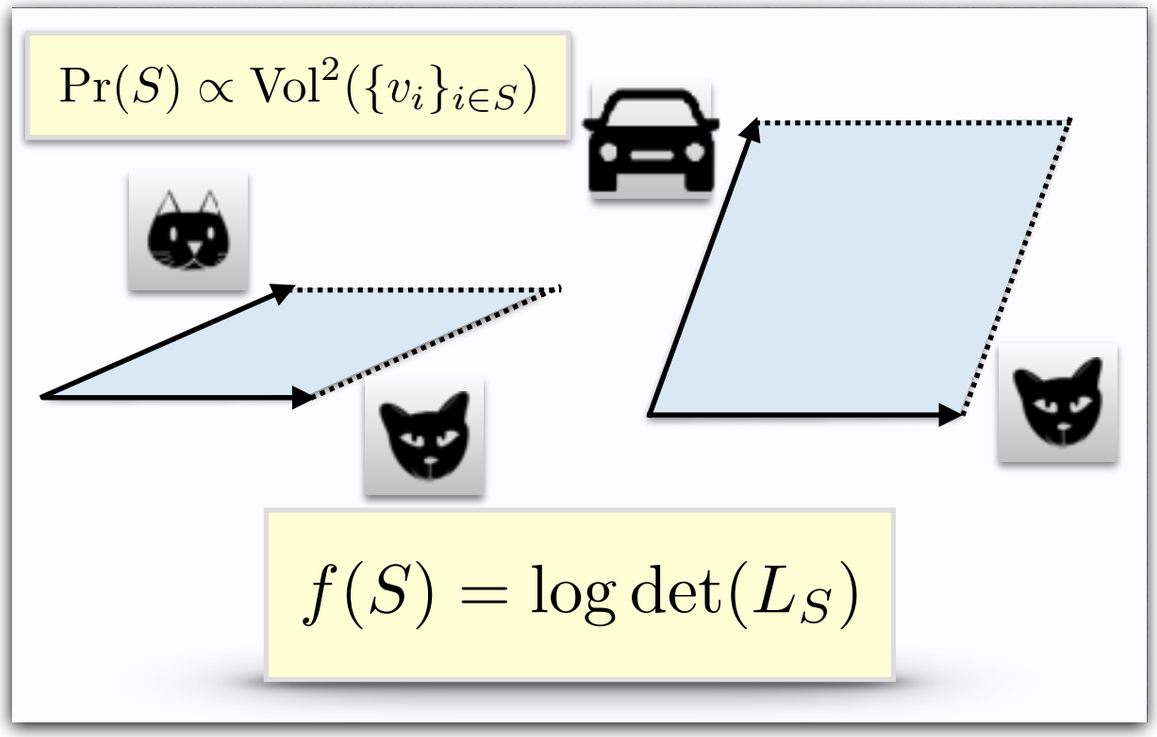

Y_1
Y_2
Y_3
Y_4
Y_5
$f(S) \doteq H(\boldsymbol{Y}) - H(\boldsymbol{Y} \mid \boldsymbol{X}_S) = I(\boldsymbol{Y}; \boldsymbol{X})$


$\Pr(S) \propto \mathrm{Vol}^2(\{v_i\}_{i \in S})$

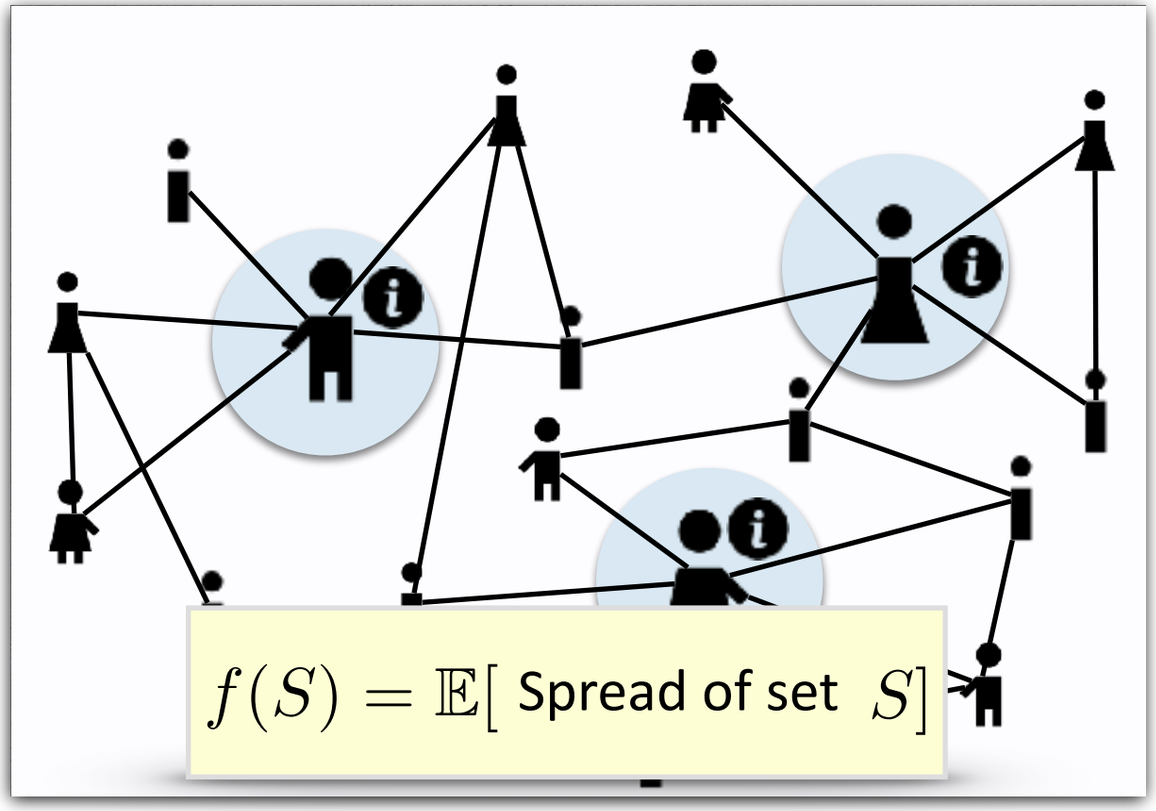
Examples

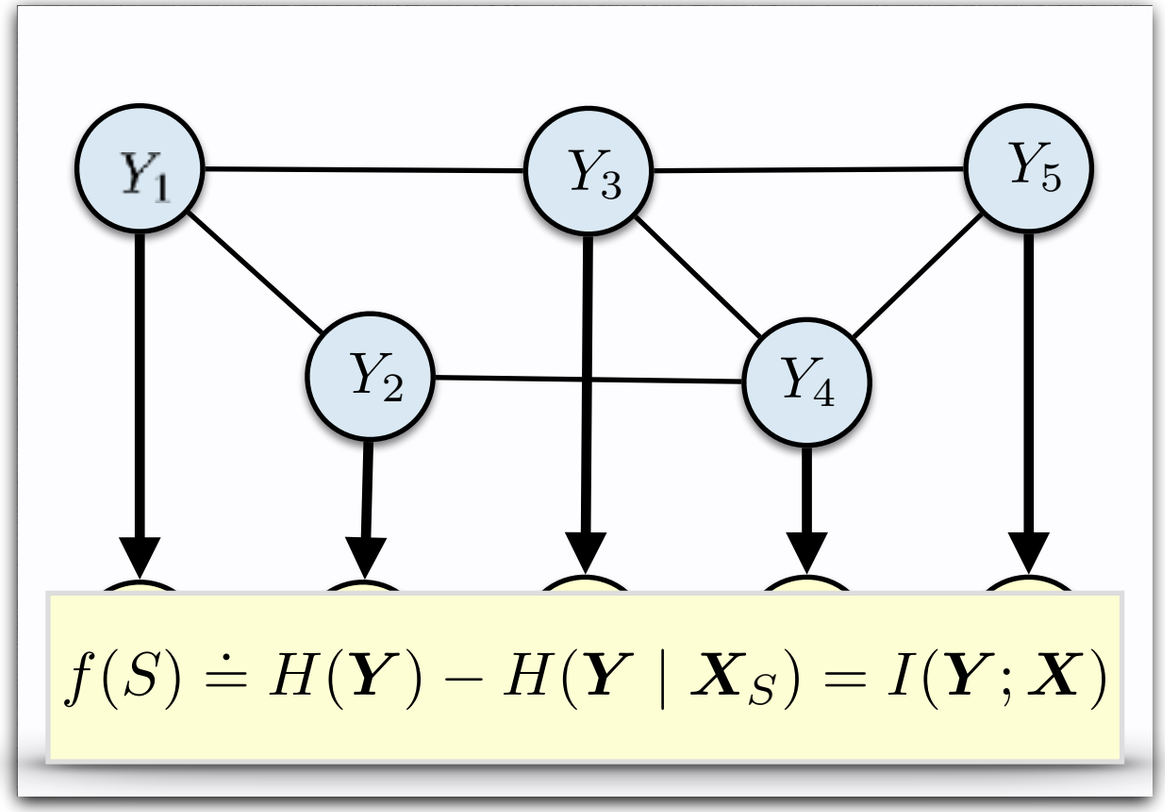

$f(S) = \mathbb{E}[$ Spread of set $S]$

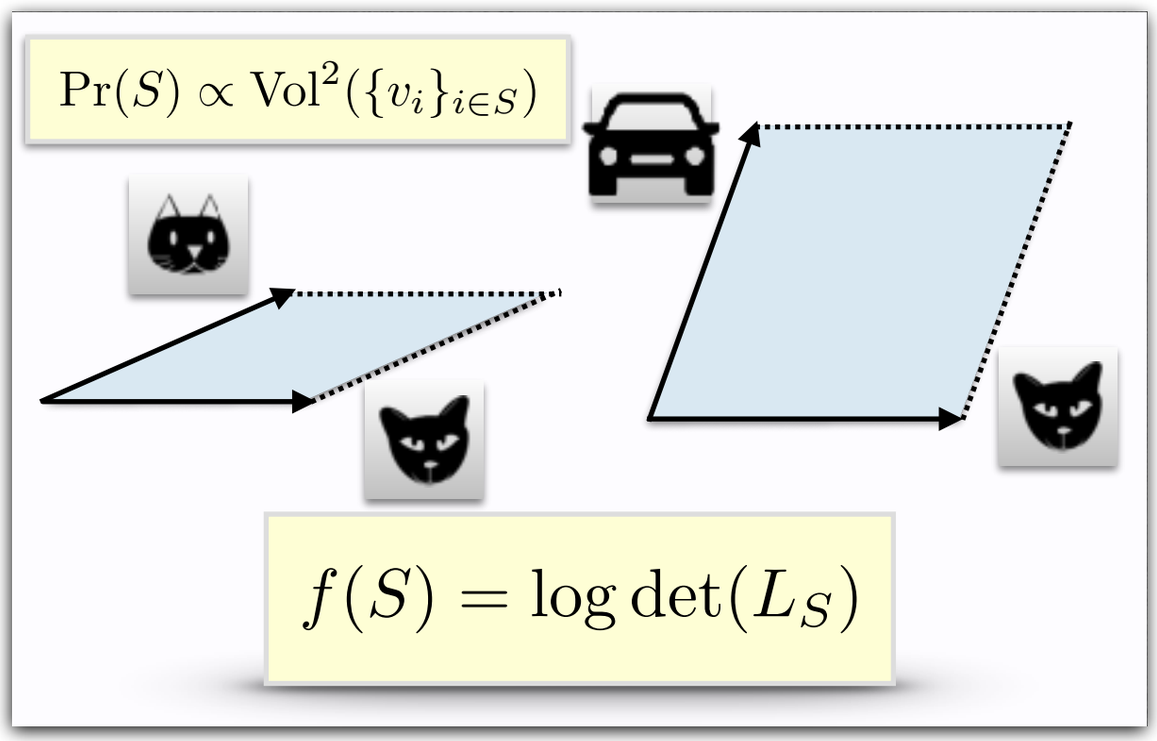

Y_1
Y_2
Y_3
Y_4
Y_5
$f(S) \doteq H(\boldsymbol{Y}) - H(\boldsymbol{Y} \mid \boldsymbol{X}_S) = I(\boldsymbol{Y}; \boldsymbol{X})$

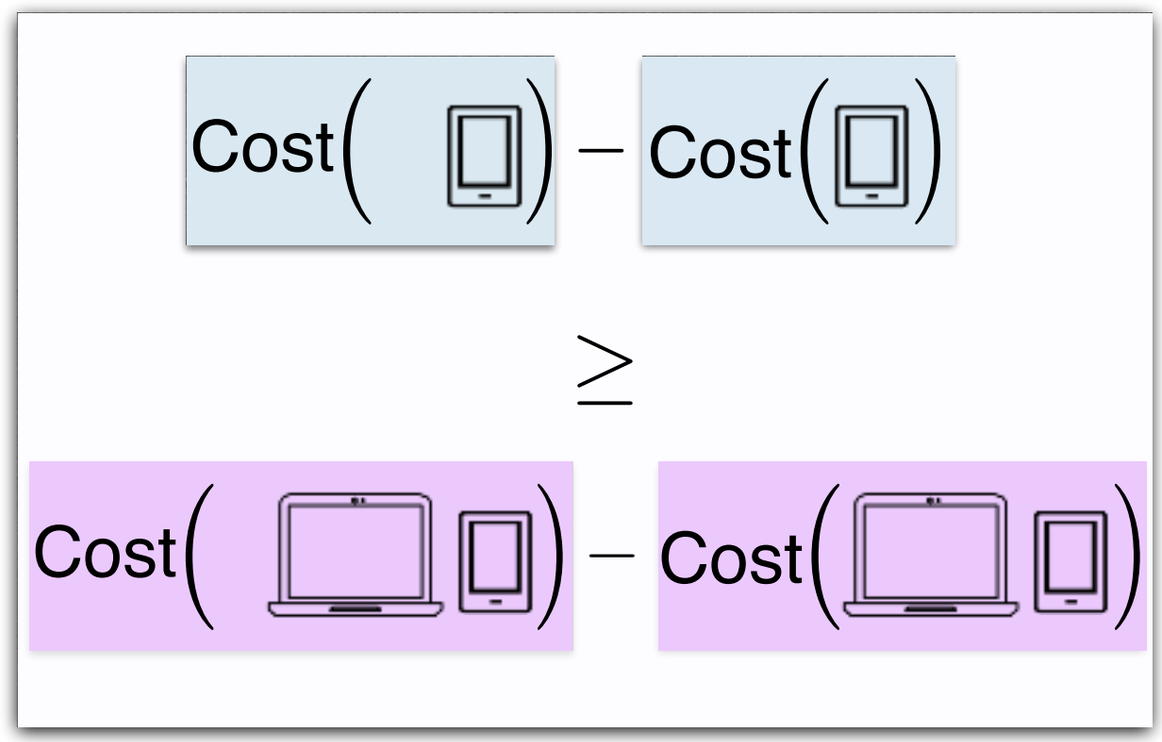

$\Pr(S) \propto \mathrm{Vol}^2(\{v_i\}_{i \in S})$
$f(S) = \log\det(L_S)$

Examples

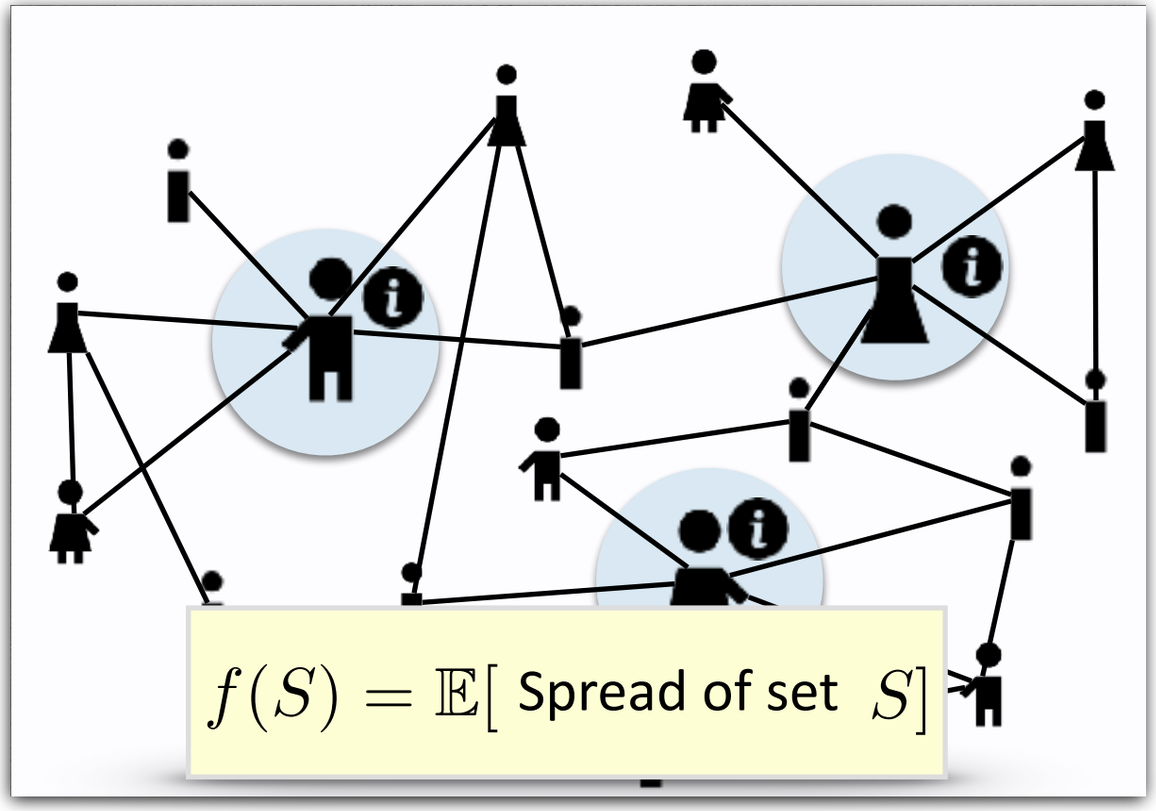

$f(S) = \mathbb{E}[$ Spread of set $S]$

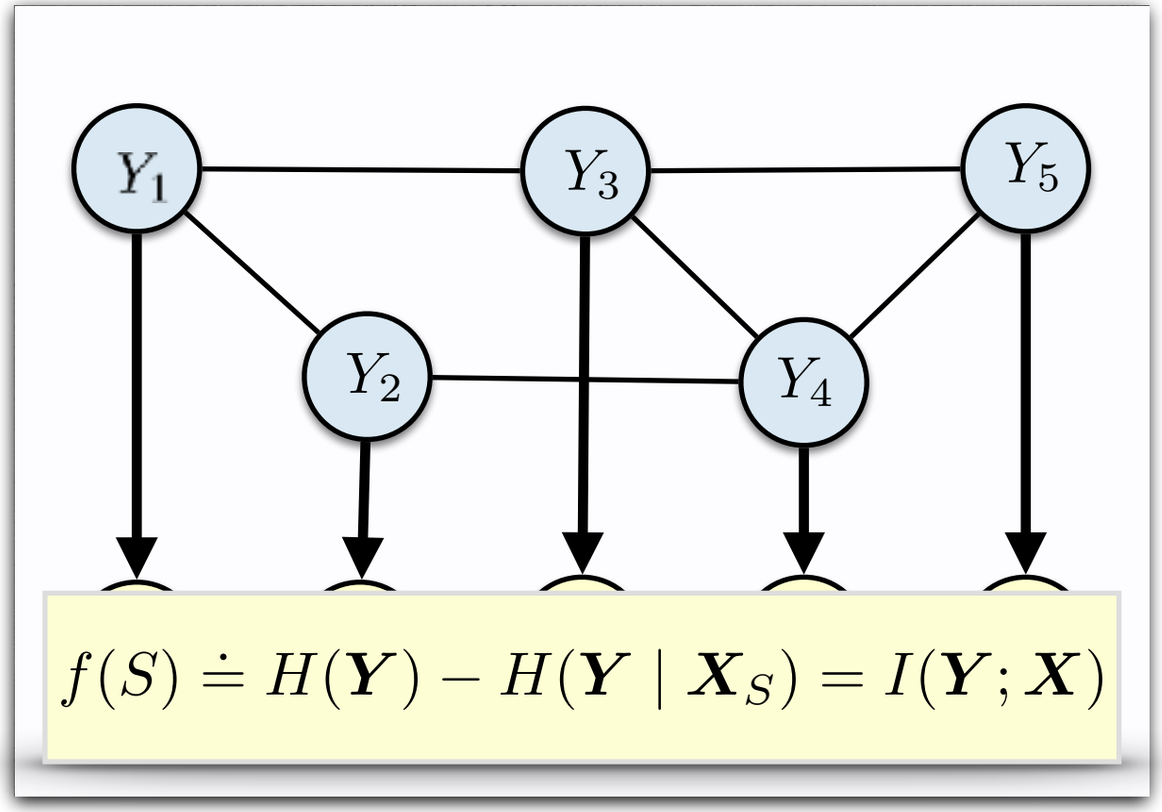

Y_1
Y_2
Y_3
Y_4
Y_5
$f(S) \doteq H(\boldsymbol{Y}) - H(\boldsymbol{Y} \mid \boldsymbol{X}_S) = I(\boldsymbol{Y}; \boldsymbol{X})$

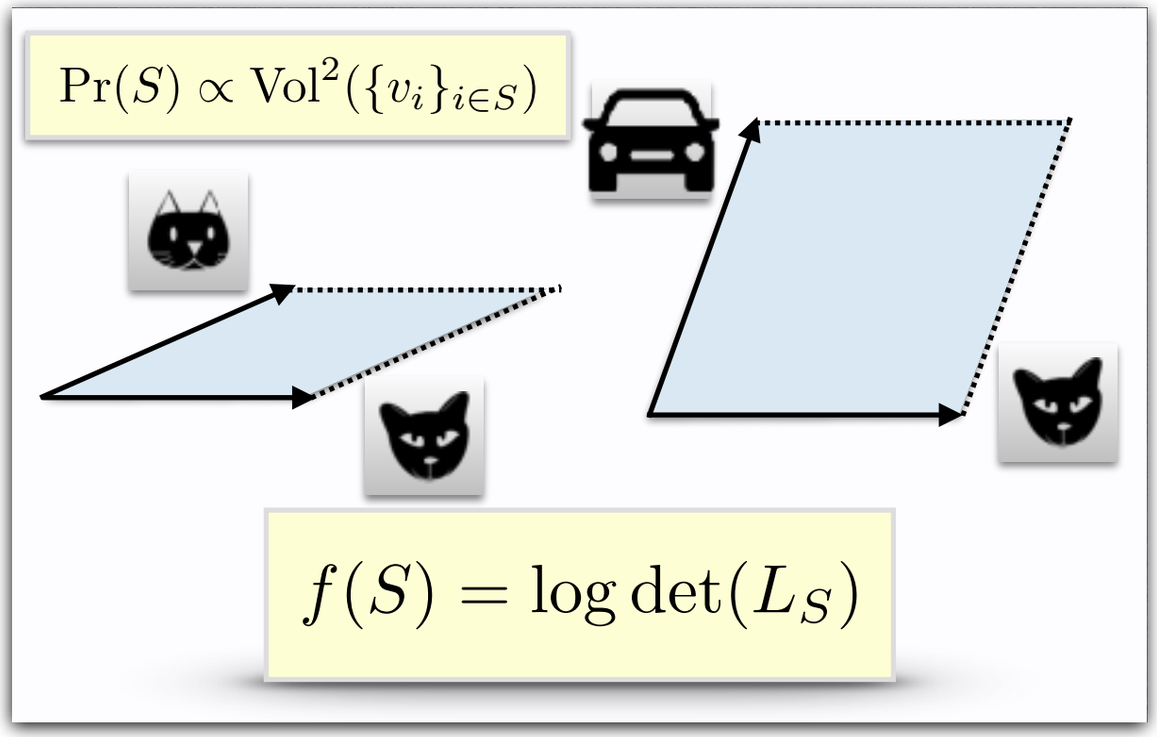

$\Pr(S) \propto \mathrm{Vol}^2(\{v_i\}_{i \in S})$
$f(S) = \log\det(L_S)$

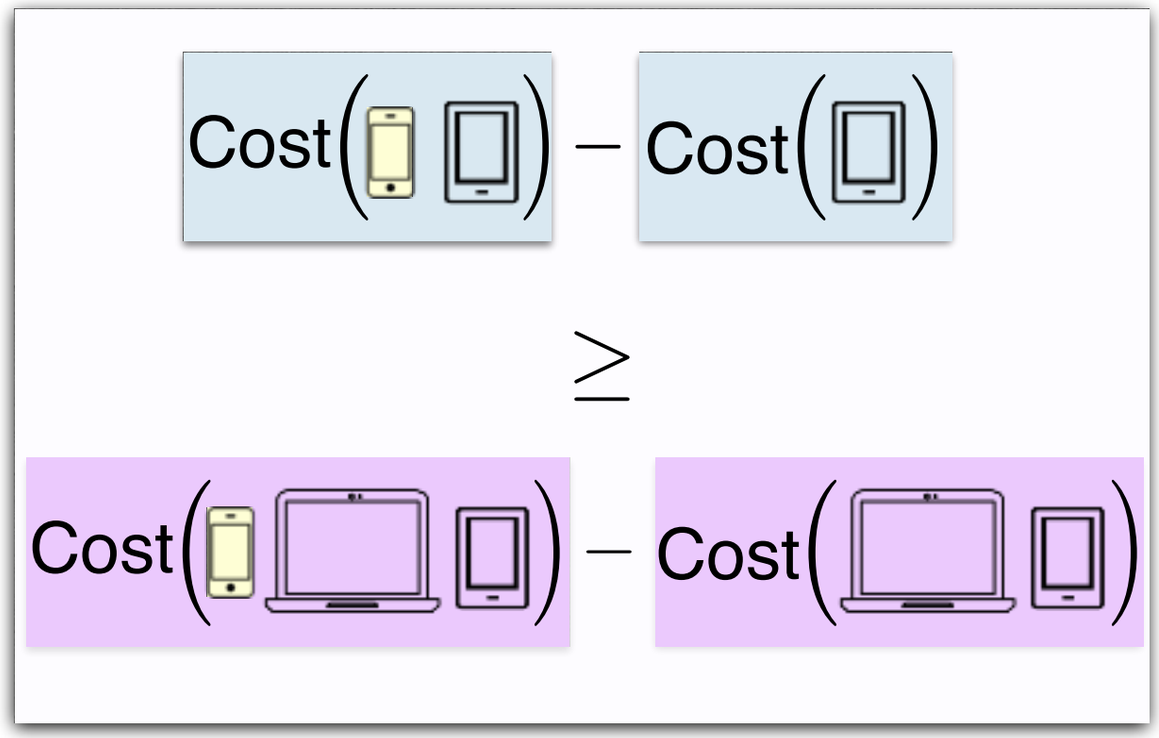

Cost( ) − Cost( )
≥
Cost( ) − Cost( )

Examples

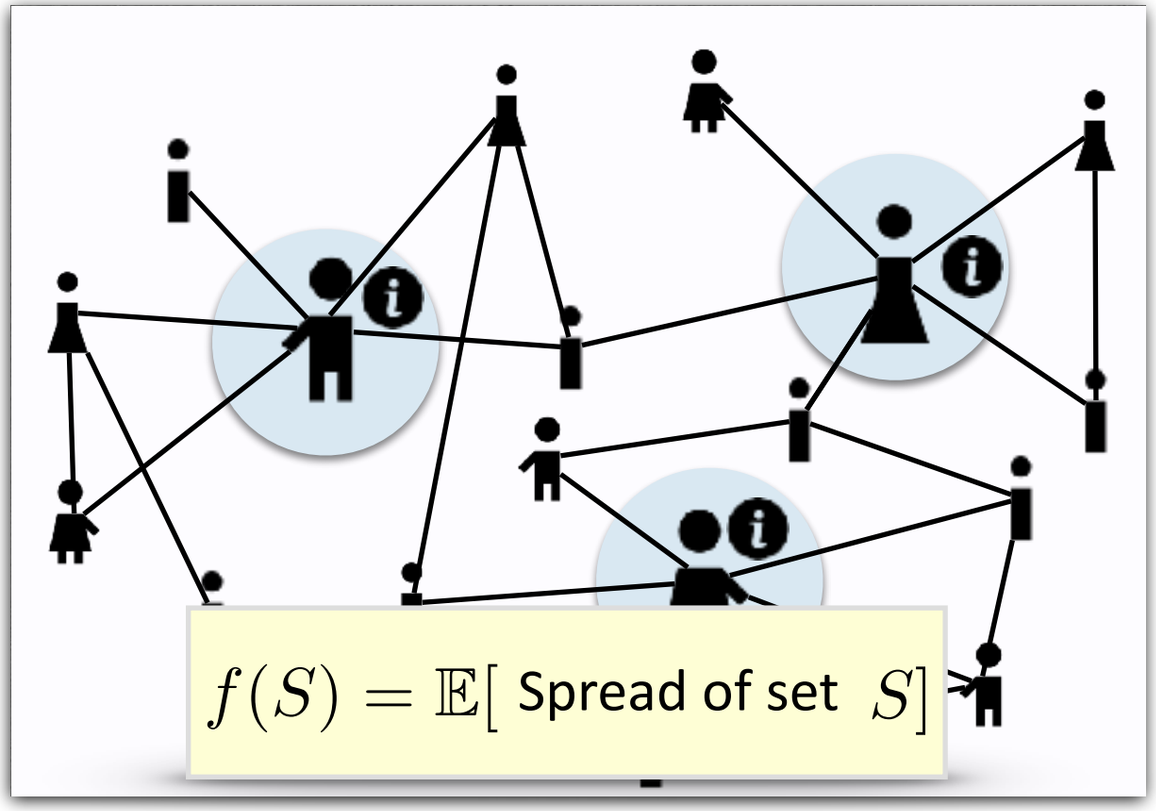

$f(S) = \mathbb{E}[$ Spread of set $S]$

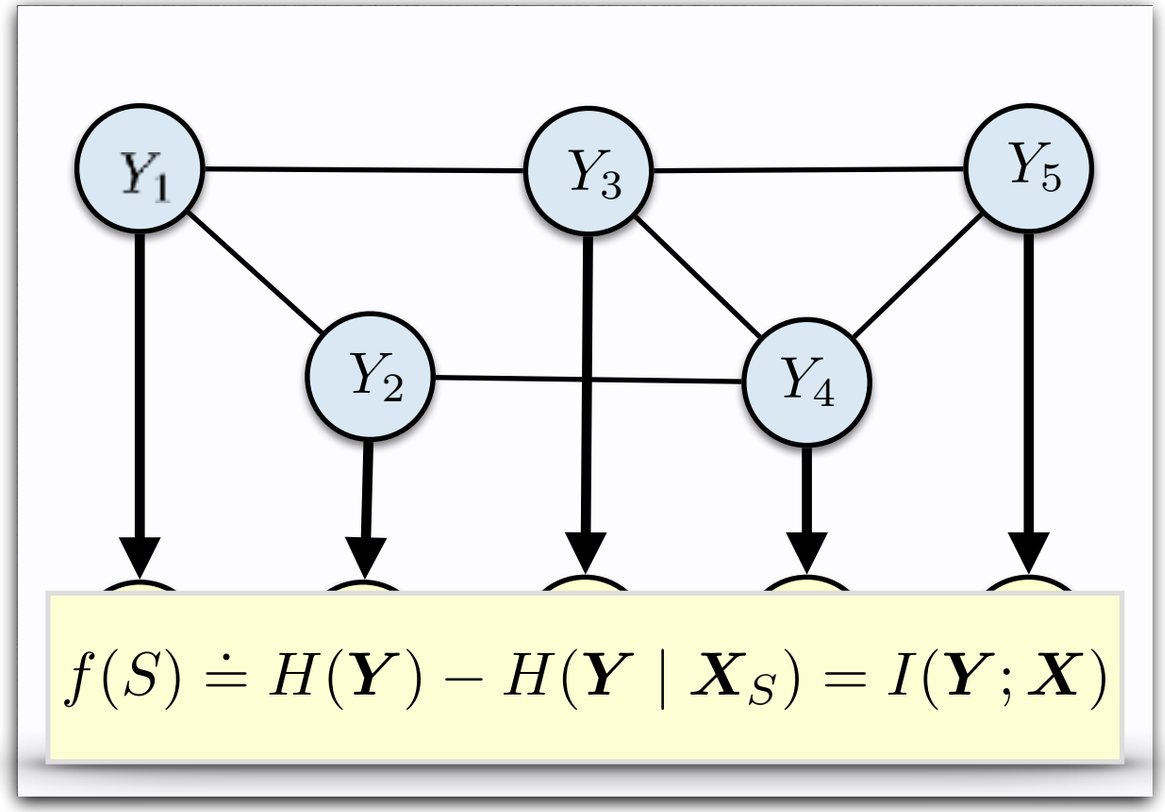

Y_1
Y_2
Y_3
Y_4
Y_5
$f(S) \doteq H(\boldsymbol{Y}) - H(\boldsymbol{Y} \mid \boldsymbol{X}_S) = I(\boldsymbol{Y}; \boldsymbol{X})$

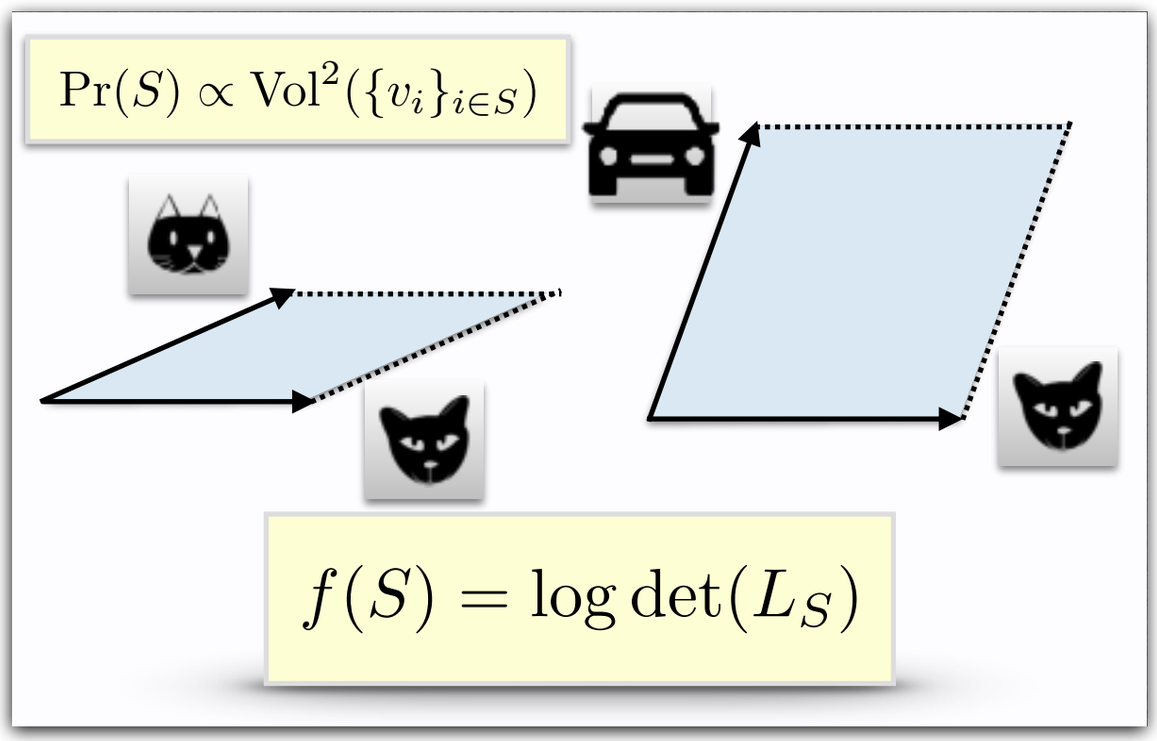

$\Pr(S) \propto \mathrm{Vol}^2(\{v_i\}_{i \in S})$
$f(S) = \log\det(L_S)$

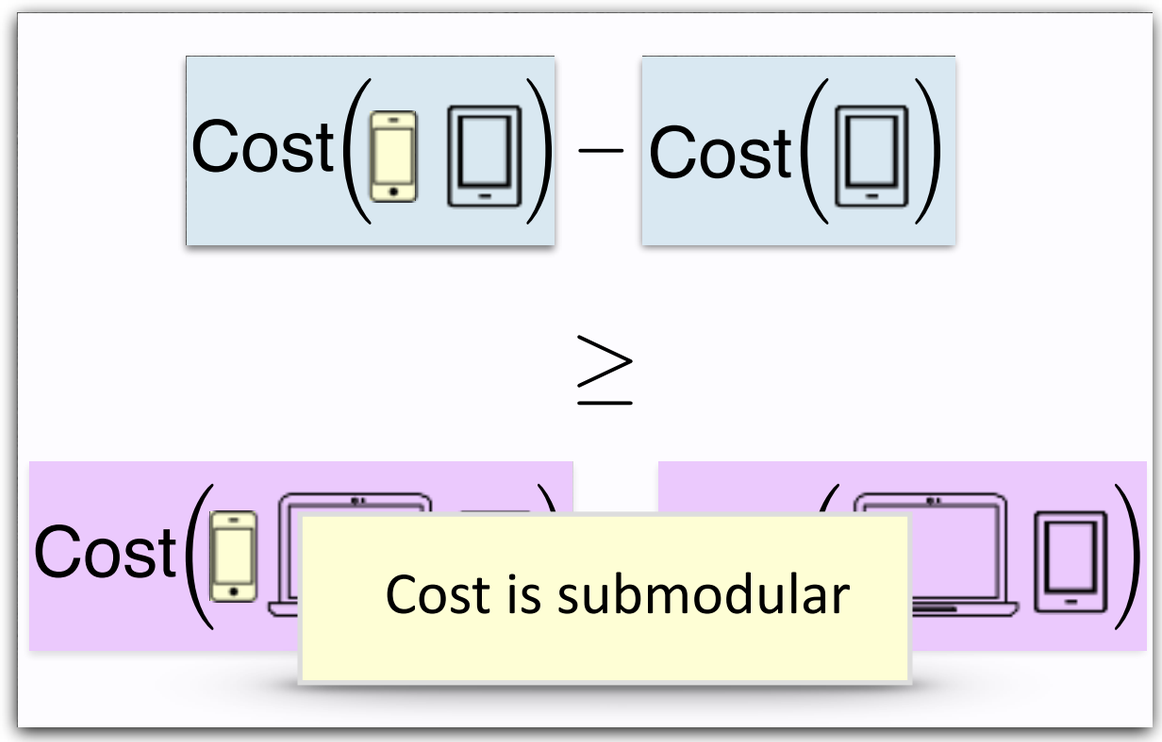

Cost
Cost
$\geq$
Cost
Cost

Examples

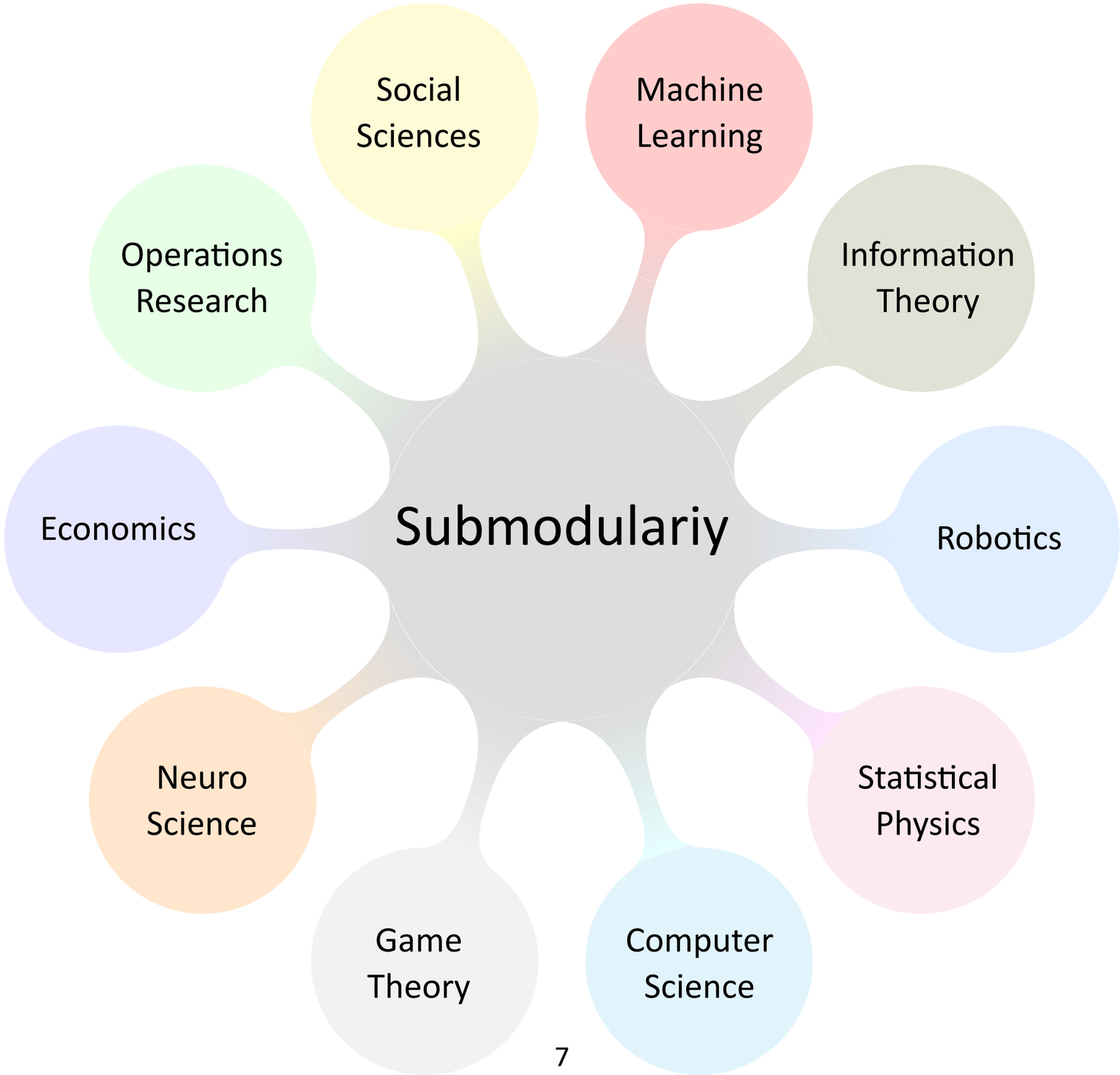

$f(S) = \mathbb{E}[$ Spread of set $S]$


Y_1
Y_2
Y_3
Y_4
Y_5
$f(S) \doteq H(\boldsymbol{Y}) - H(\boldsymbol{Y} \mid \boldsymbol{X}_S) = I(\boldsymbol{Y}; \boldsymbol{X})$


$\Pr(S) \propto \mathrm{Vol}^2(\{v_i\}_{i \in S})$
$f(S) = \log\det(L_S)$


Cost
Cost
$\geq$
Cost
Cost is submodular

Submodularity in Everyone's Backyard



Part One

$$S^* = \arg\max_{S \subseteq V, S \in \mathcal{I}} f(S)$$

Submodularity in Action: From Machine Learning to Signal Processing Applications

Ehsan Tohidi*, Rouhollah Amiri†, Mario Coutino‡, David Gesbert*,
Geert Leus‡, and Amin Karbasi§
*Communication Systems Department, EURECOM, Biot, France
†Electrical Engineering Department, Sharif University of Technology, Tehran, Iran
‡Faculty of Electrical Engineering, Mathematics and Computer Science, Delft University of Technology, Delft, The Netherlands
§Electrical Engineering and Computer Science Department, Yale University, New Haven, USA

Part One

$$S^* = \arg\max_{S \subseteq V, S \in \mathcal{I}} f(S)$$

Perfect Oracle

Discrete Methods

Imperfect Oracle

Continuous Methods

Part One

$$S^* = \arg\max_{S \subseteq V, S \in \mathcal{I}} f(S)$$

Perfect Oracle

Discrete Methods

Imperfect Oracle

Continuous Methods

Part One

$$S^* = \arg\max_{S \subseteq V, S \in \mathcal{I}} f(S)$$

Perfect Oracle

Discrete Methods

Imperfect Oracle

Continuous Methods

Centralized

Streaming

Distributed

Part One

$$S^* = \arg\max_{S \subseteq V, S \in \mathcal{I}} f(S)$$

Perfect Oracle

Discrete Methods

Imperfect Oracle

Continuous Methods

Centralized

Streaming

Distributed

Performance Measure

$$S^* = \arg\max_{S \subseteq V, S \in \mathcal{I}} f(S)$$

- **Perfect Oracle:** The function f can be evaluated exactly.

Performance Measure

$$S^* = \arg\max_{S \subseteq V, S \in \mathcal{I}} f(S)$$

- **Perfect Oracle:** The function f can be evaluated exactly.

Performance Measures

- **Query Complexity:** How many times we evaluate the function f?
- **Approximation Guarantee:** What fraction of the optimum value we obtain?

$$f(S_{\text{Alg}}) \geq \alpha \cdot f(S^*)$$

Performance Measure

$$S^* = \arg\max_{S \subseteq V, S \in \mathcal{I}} f(S)$$

- **Perfect Oracle:** The function f can be evaluated exactly.

Performance Measures

- **Query Complexity:** How many times we evaluate the function f?
- **Approximation Guarantee:** What fraction of the optimum value we obtain?

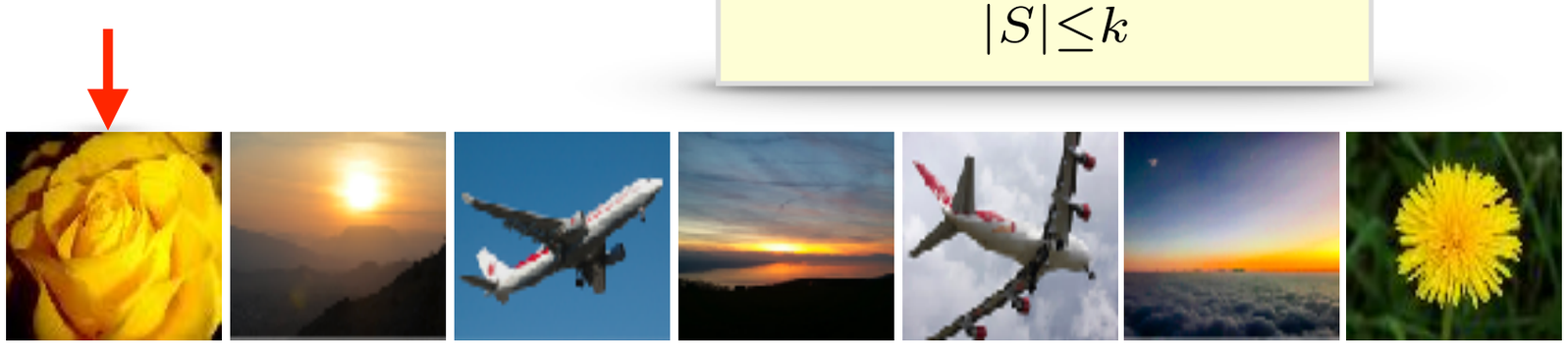
$$f(S_{\text{Alg}}) \geq \alpha \cdot f(S^*)$$

- **Assumption: the function f is non-negative.**

Greedy Algorithm

- Consider the following problem where f is monotone submodular:

$$S^* = \arg\max_{|S| \leq k} f(S)$$



$\Delta f = \{$ $\}$

$f(\{$ $\})$

Initialize: $S \leftarrow \varnothing$

for $i = 1$ **to** k **do**

$e^* \leftarrow \arg\max_{e \in V \setminus S} f(S \cup \{e\})$

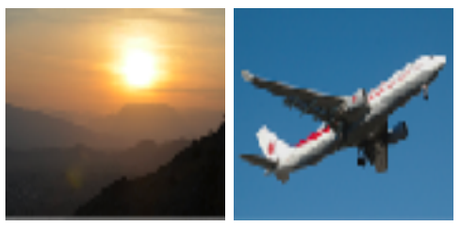
$S \leftarrow S \cup \{e^*\}$

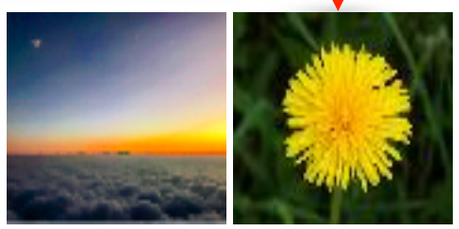
Greedy

Greedy Algorithm

- Consider the following problem where f is monotone submodular:

$$S^* = \arg\max_{|S| \leq k} f(S)$$





$\Delta f = \{$ $\}$

Initialize: $S \leftarrow \varnothing$

for $i = 1$ **to** k **do**

$e^* \leftarrow \arg\max_{e \in V \setminus S} f(S \cup \{e\})$

$S \leftarrow S \cup \{e^*\}$

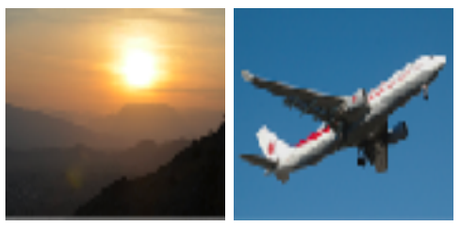
Greedy

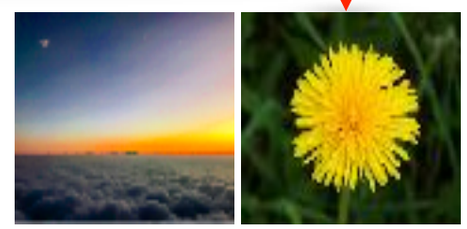
$f(\{$ $\})$

Greedy Algorithm

- Consider the following problem where f is monotone submodular:

$$S^* = \arg\max_{|S| \leq k} f(S)$$





Initialize: $S \leftarrow \varnothing$

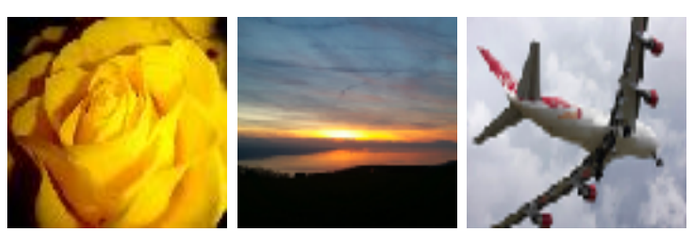
for $i = 1$ **to** k **do**

$e^* \leftarrow \arg\max_{e \in V \setminus S} f(S \cup \{e\})$

$S \leftarrow S \cup \{e^*\}$

Greedy

$\Delta f = \{$ $\}$

$f(\{$  $\})$

[Nemhauser, Wolsey, Fisher]

For monotone submodular functions the Greedy algorithm gives a constant factor approximation under a cardinality constraint

$$f(S_{\text{Greedy}}) \geq (1 - 1/e) \cdot \text{OPT}$$

"An Analysis of Approximations for Maximizing Submodular Set Functions—I", 1978

How Fast Greedy Can Be?

- For a ground set V of size n, standard greedy algorithm needs $O(nk)$ function evaluations to find a summary of size k.
- In many data intensive applications, evaluating f is expensive and running the standard greedy algorithm is infeasible.

$$S^* = \arg\max_{|S| \leq k} f(S)$$

Is it possible to have an algorithm that "**does not depend on k**" at all and **scales linearly** with the data size n?

How Fast Greedy Can Be?

- For a ground set V of size n, standard greedy algorithm needs $O(nk)$ function evaluations to find a summary of size k.
- In many data intensive applications, evaluating f is expensive and running the standard greedy algorithm is infeasible.

$$S^* = \arg\max_{|S| \leq k} f(S)$$

Is it possible to have an algorithm that "**does not depend on k**" at all and **scales linearly** with the data size n?

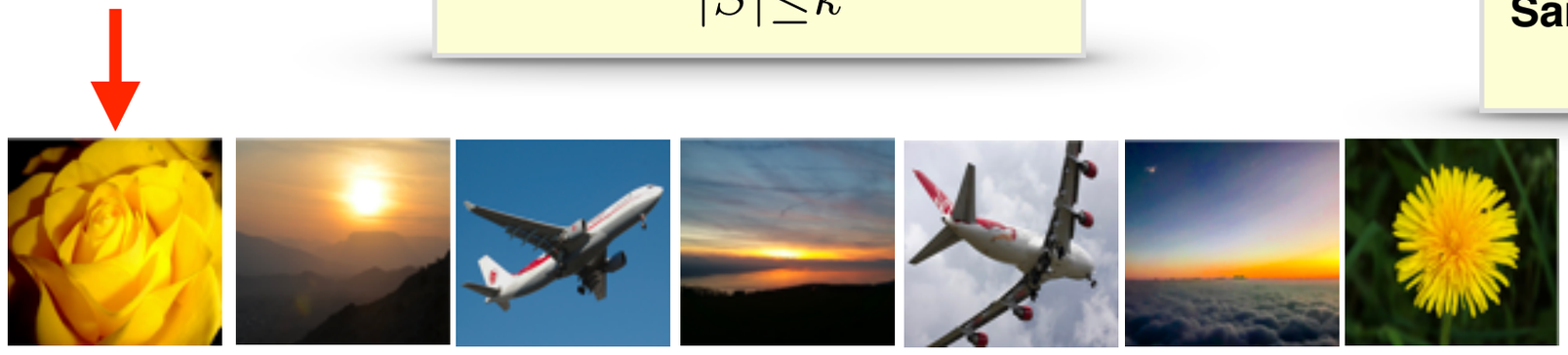
"Accelerated greedy algorithms for maximizing submodular set functions", Minoux, 1978

SubSample-Greedy Algorithm

- Consider the following problem where f is monotone submodular:

$$S^* = \arg\max_{|S| \leq k} f(S)$$

Sample n/k points per iteration



$\Delta f = \{ \qquad \}$

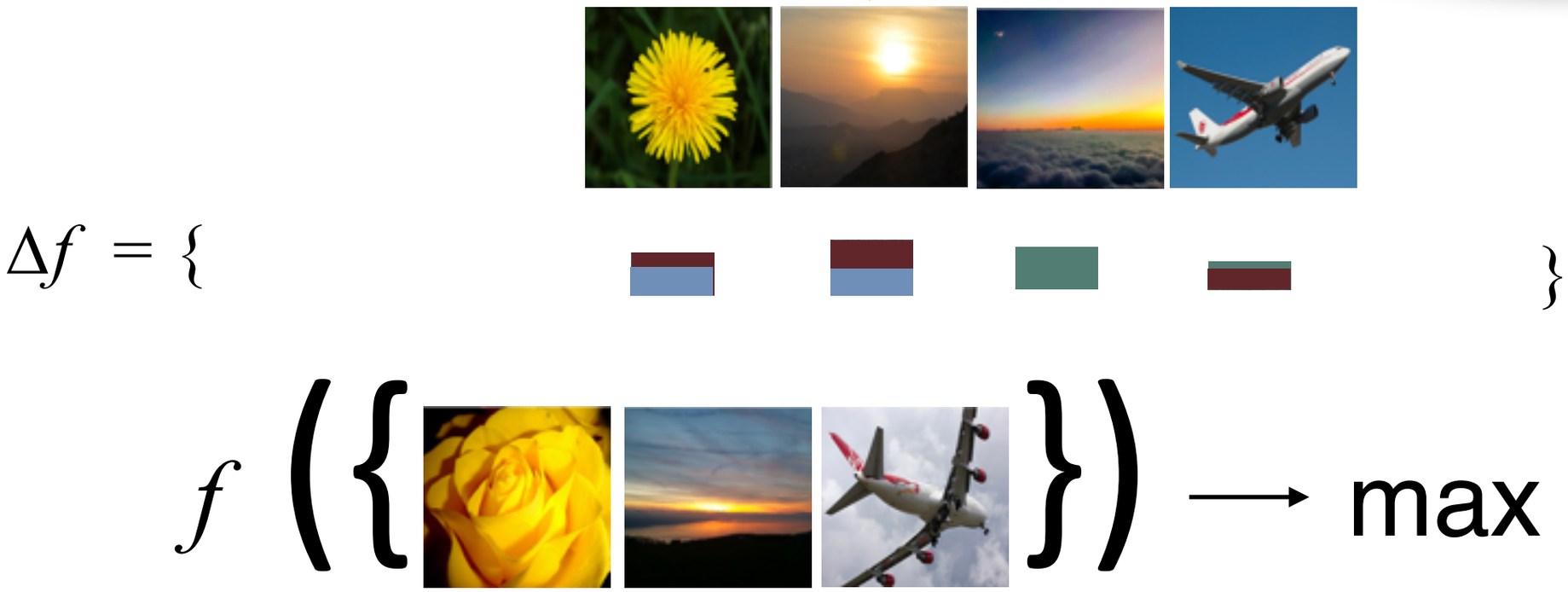
$f(\{ \qquad \}) \longrightarrow$ max

SubSample-Greedy Algorithm

- Consider the following problem where f is monotone submodular:

$$S^* = \arg\max_{|S| \leq k} f(S)$$

Sample n/k points per iteration

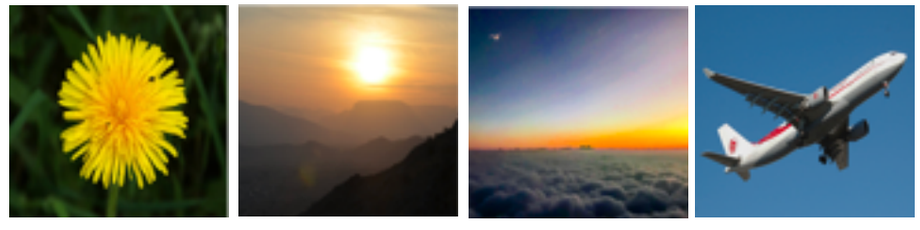


SubSample-Greedy Algorithm

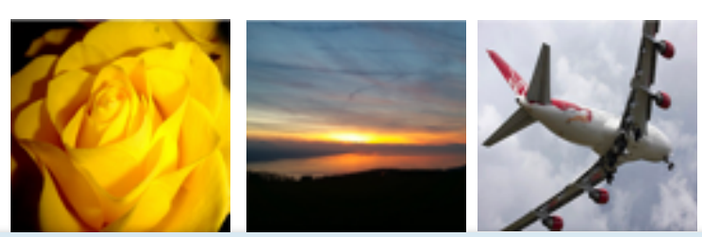
- Consider the following problem where f is monotone submodular:

$$S^* = \arg\max_{|S| \leq k} f(S)$$

Sample n/k points per iteration



$\Delta f = \{$ $\}$

$f\left(\{ \right.$  $\left. \}\right) \longrightarrow$ max

[Buchbinder, Feldman, Naor, Schwartz] [Mitrovic, Bun, Krause, Karbasi]

For monotonic submodular functions, SubSample-Greedy gives a constant factor approximation using only n function evaluations

$$\mathbb{E}[f(S_{\text{SubSample-Greedy}})] \geq (1 - 1/e)^2 \cdot \text{OPT}$$

"Submodular Maximization With Cardinality Constraints", 2014

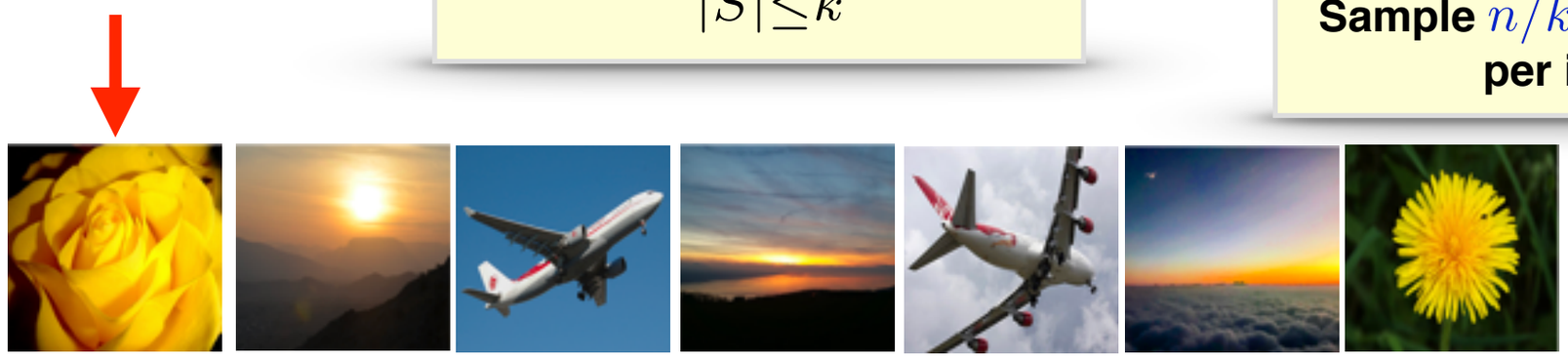
"Differentially Private Submodular Maximization: Data Summarization in Disguise", ICML'17

Stochastic-Greedy Algorithm

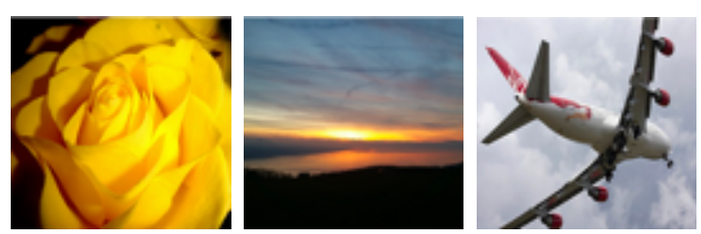
- Consider the following problem where f is monotone submodular:

$$S^* = \arg\max_{|S| \leq k} f(S)$$

Sample $n/k \log(1/\varepsilon)$ points per iteration



$\Delta f = \{$ $\}$

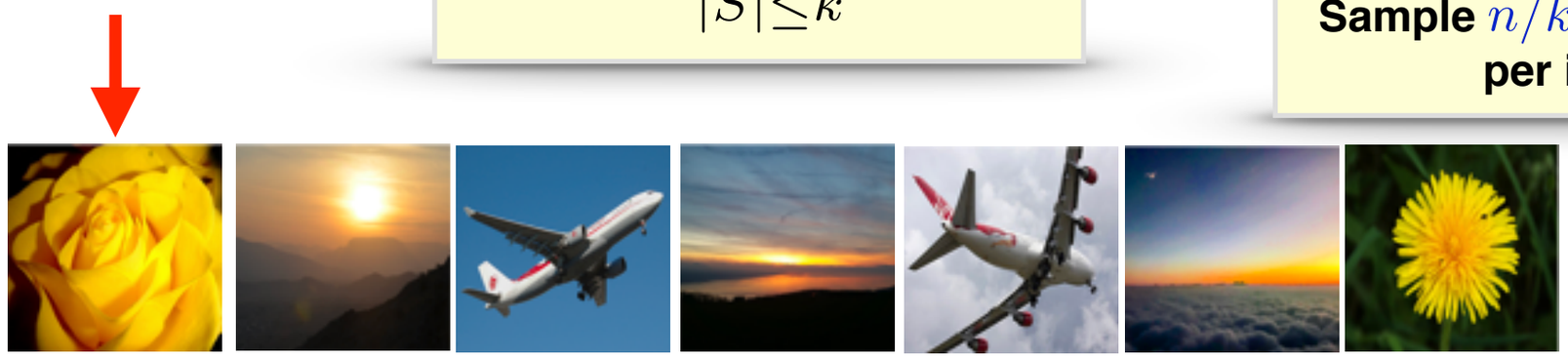
$f\left(\{\right.$  $\left.\}\right) \longrightarrow$ max

Stochastic-Greedy Algorithm

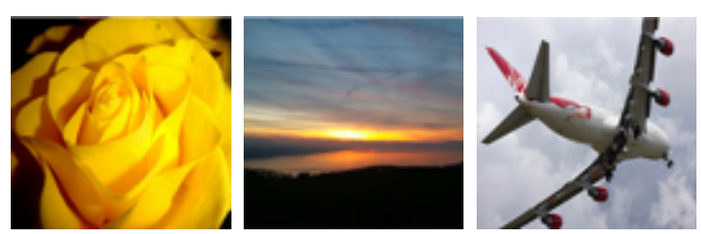
- Consider the following problem where f is monotone submodular:

$$S^* = \arg\max_{|S| \leq k} f(S)$$

Sample $n/k \log(1/\varepsilon)$ points per iteration



$\Delta f = \{$ $\}$

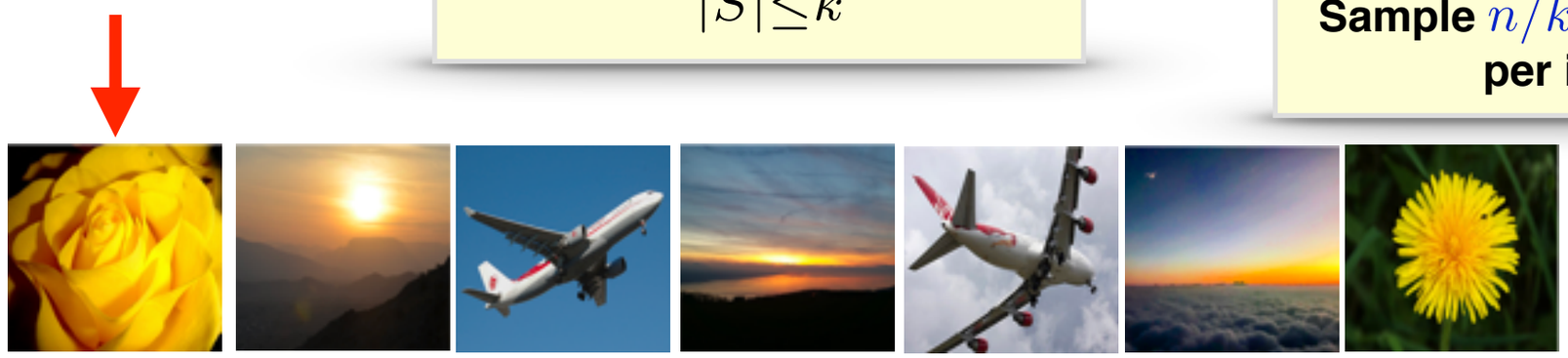
$f\left(\{\right.$  $\left.\}\right) \longrightarrow$ max

Stochastic-Greedy Algorithm

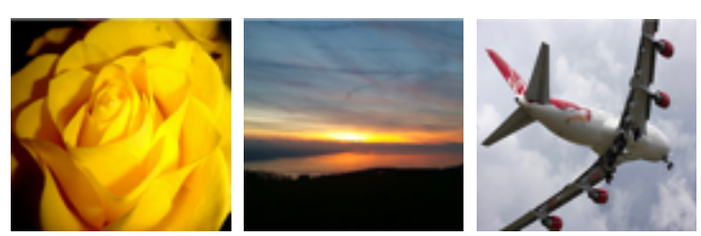
- Consider the following problem where f is monotone submodular:

$$S^* = \arg\max_{|S| \leq k} f(S)$$

Sample $n/k \log(1/\varepsilon)$ points per iteration



$\Delta f = \{$ $\}$

$f(\{$  $\}) \longrightarrow$ max

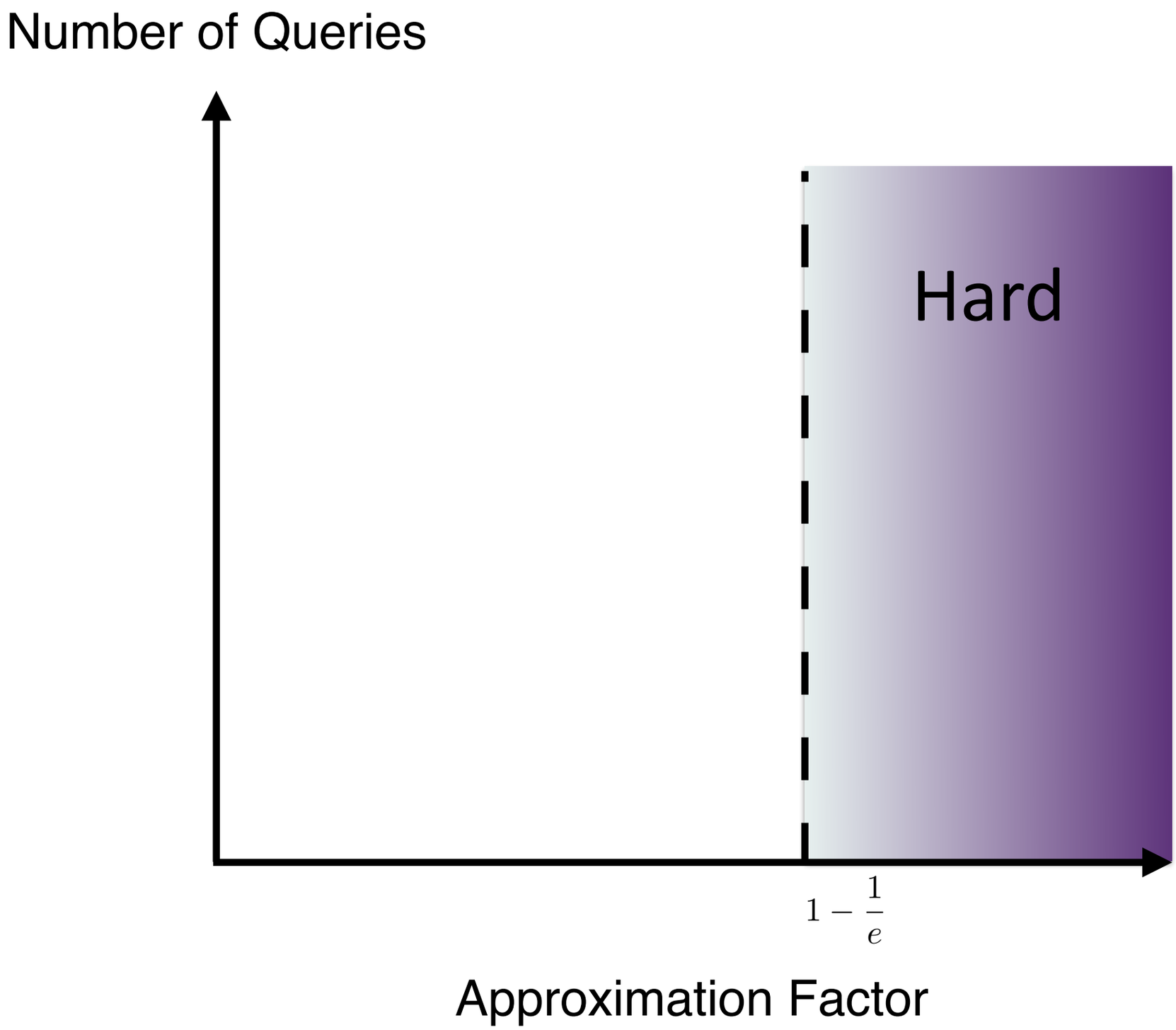
[Mirzasoleiman, Badanidiyuru, Karbasi, Vondrak, Krause]

For monotone submodular functions, Stochastic-Greedy gives constant factor approximation using only $n \log(1/\varepsilon)$ function evaluations

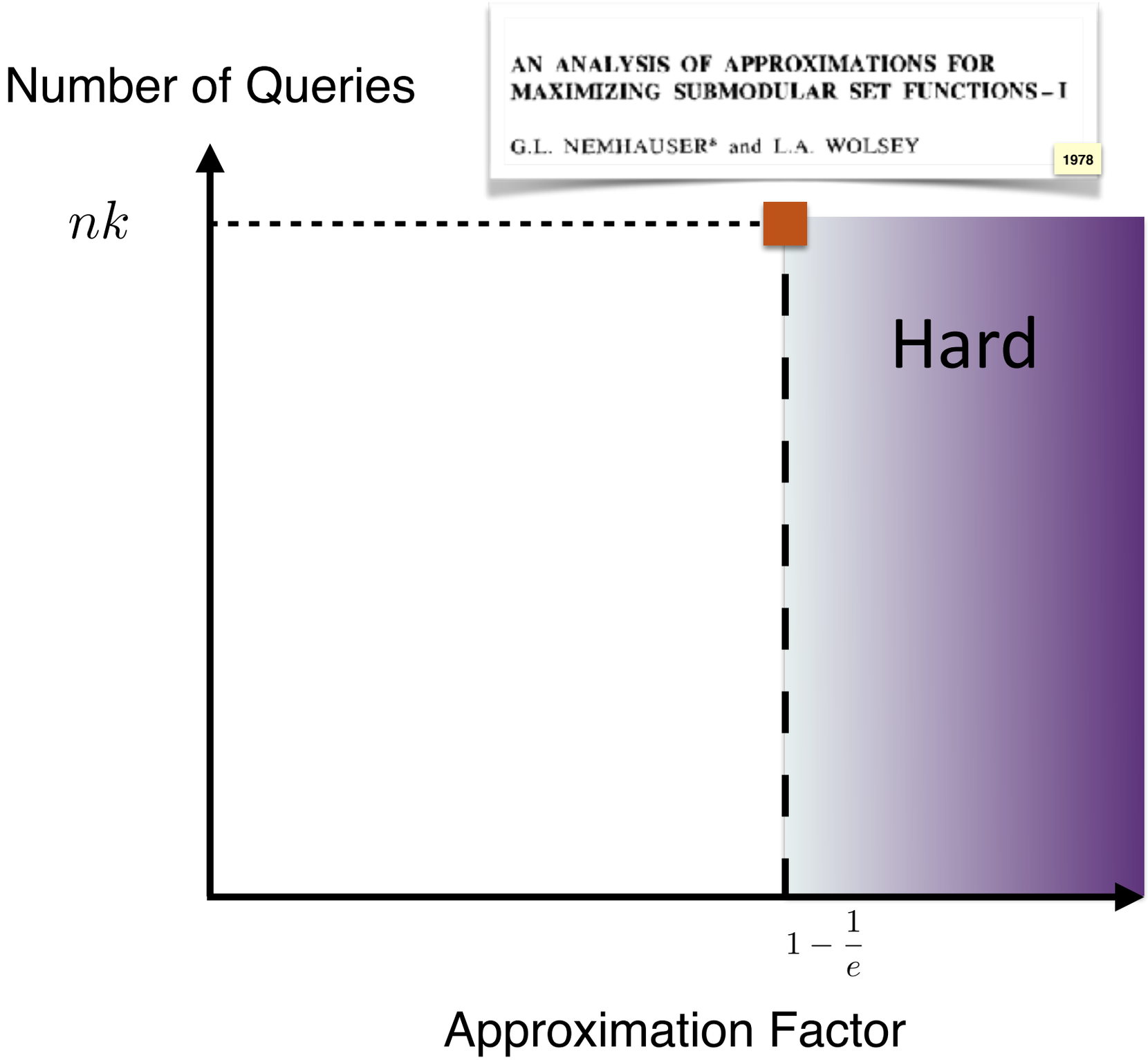
$$\mathbb{E}[f(S_{\text{Stochastic-Greedy}})] \geq (1 - 1/e - \varepsilon) \cdot \text{OPT}$$

"Lazier Than Lazy Greedy", 2015

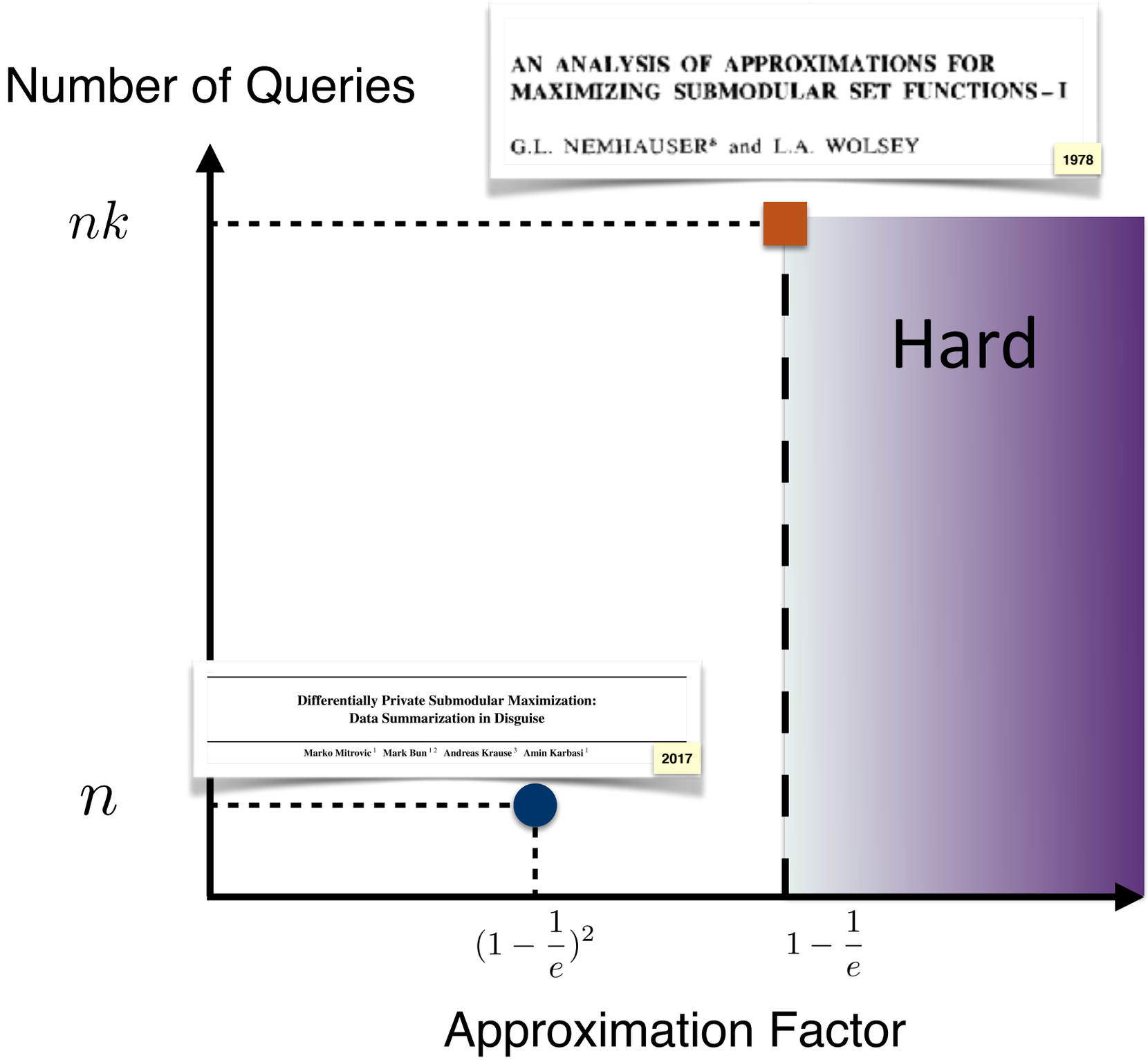
Fast Greedy Algorithms



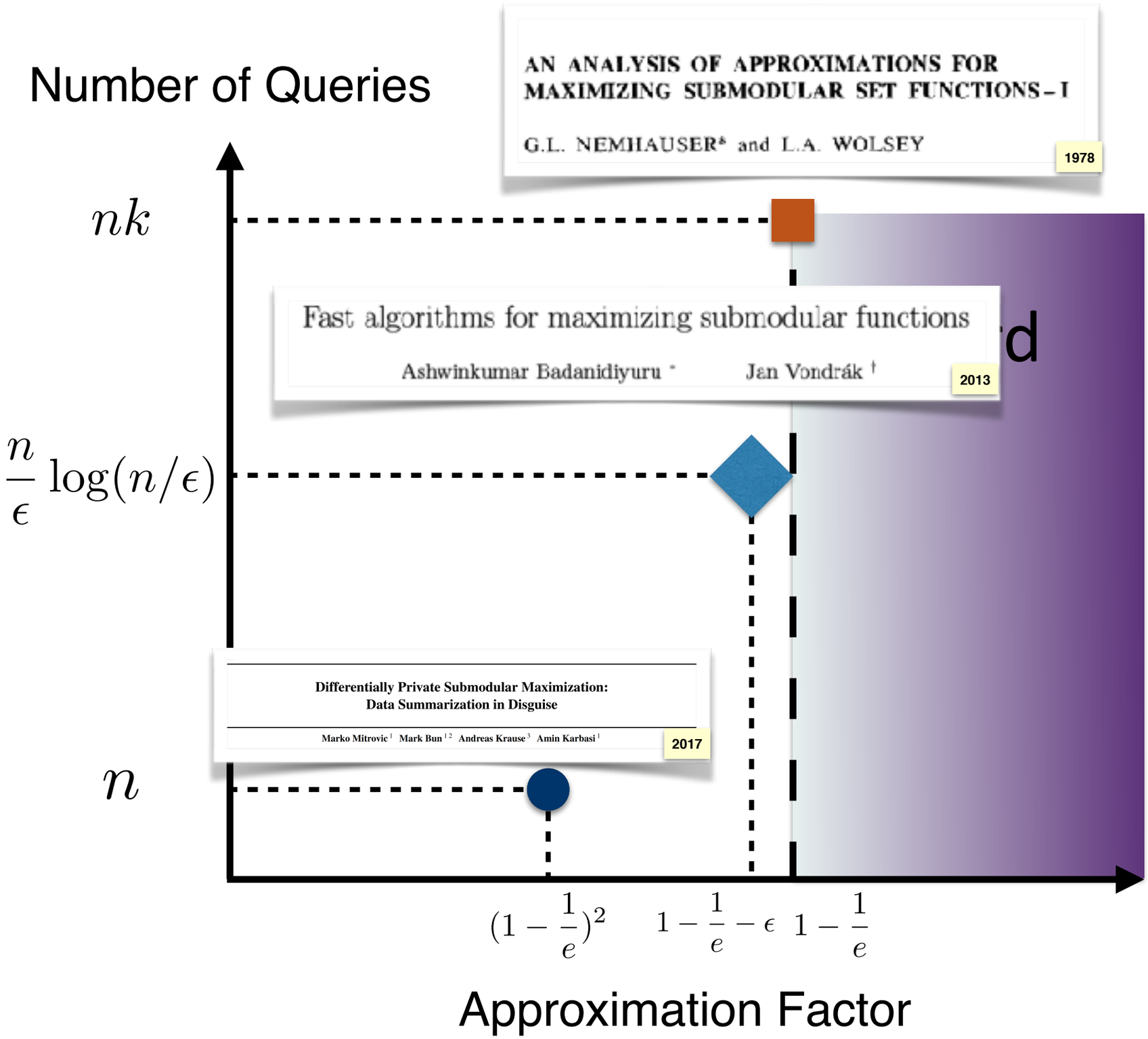
Fast Greedy Algorithms



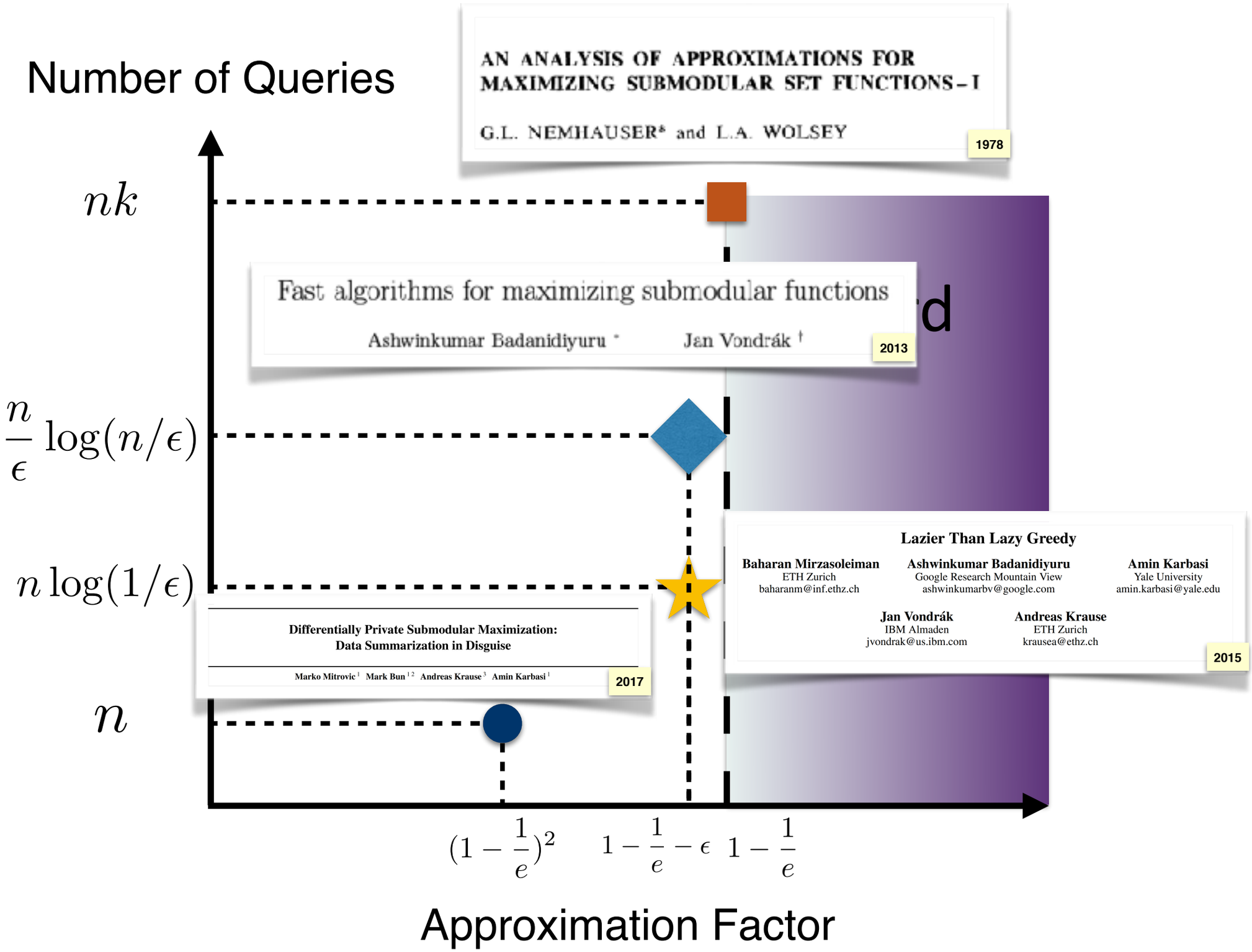
Fast Greedy Algorithms



Fast Greedy Algorithms



Fast Greedy Algorithms



Open Question

Can we develop a **deterministic** algorithm

- Query Complexity: $O(n)$
- Approximation Guarantee: $1 - \frac{1}{e}$

- Related work:

Open Question

Can we develop a **deterministic** algorithm

- Query Complexity: $O(n)$
- Approximation Guarantee: $1 - \frac{1}{e}$

- Related work:

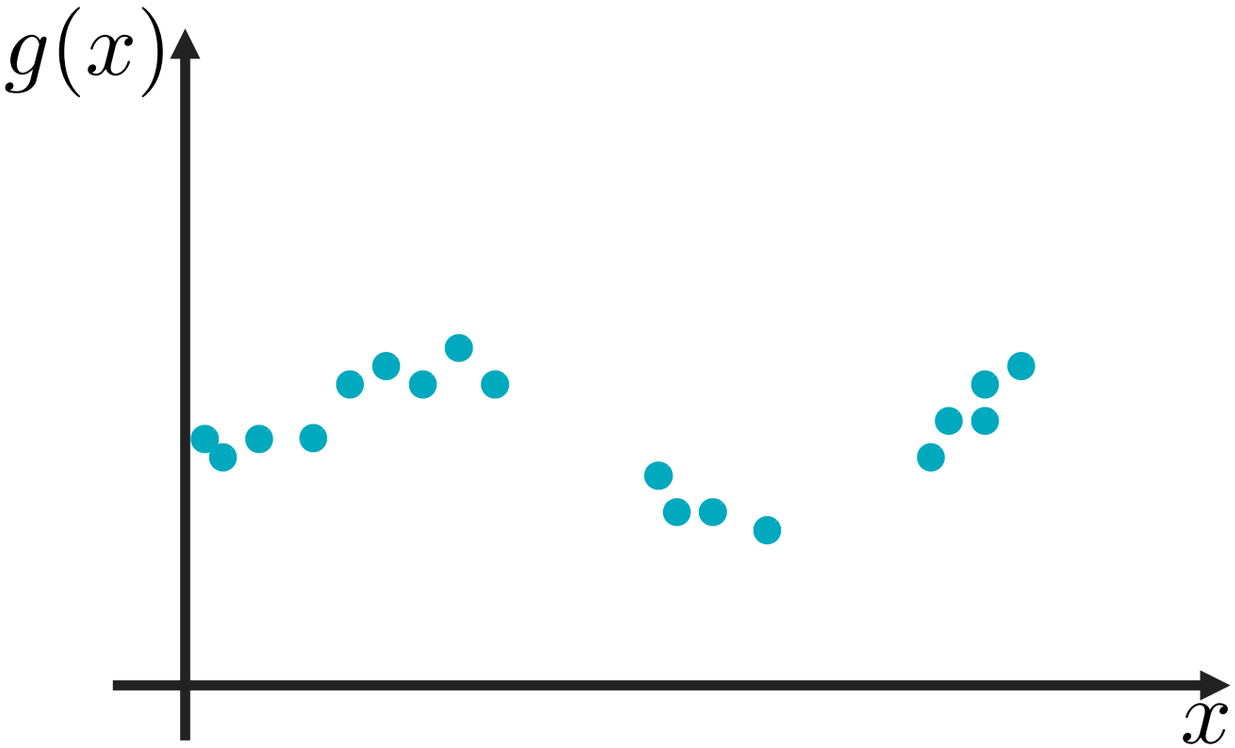
"Robust Guarantees for Stochastic Greedy Algorithms", Hassidim, Singer, 2017

"Accelerated Greedy Algorithms for Maximizing Submodular Set Functions", Minoux, 1978

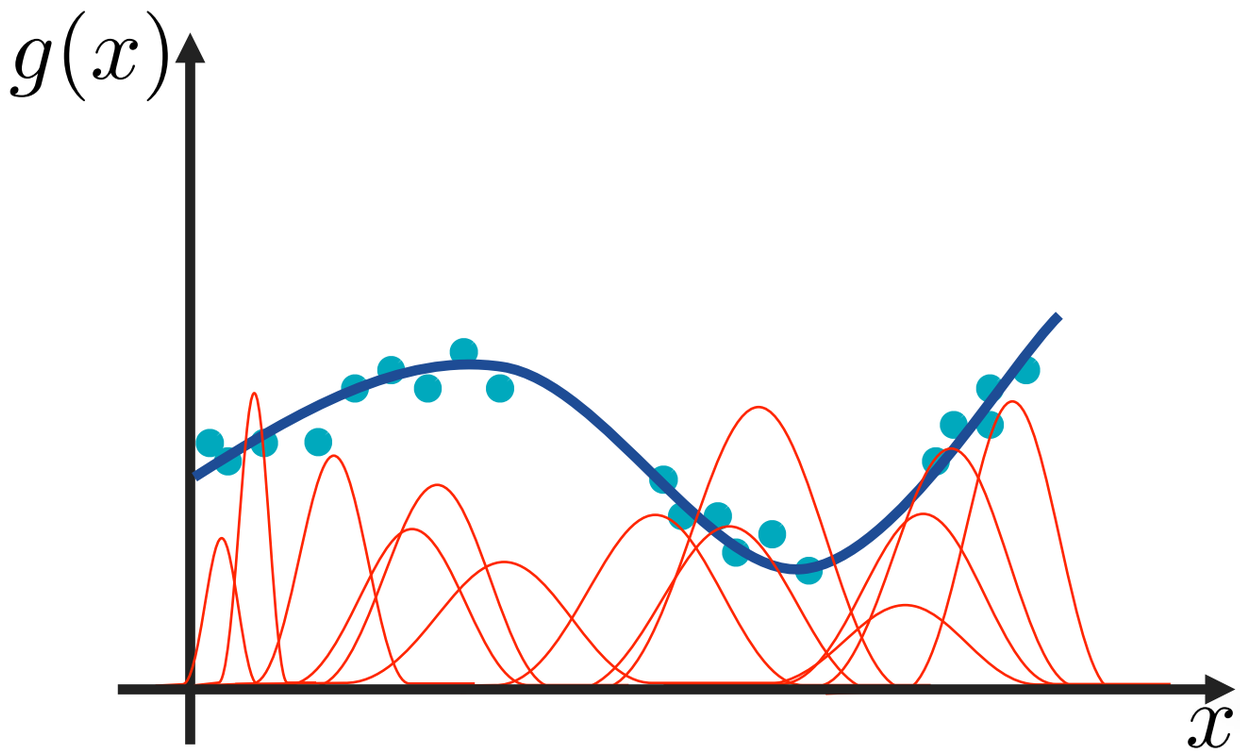
"Fast Multi-Stage Submodular Maximization", Wei, Iyer, Bilmes, 2015

"Efficient Algorithms and Lower Bound for Submodular Maximization", Li, Shroff, 2020

Nonparametric Regression

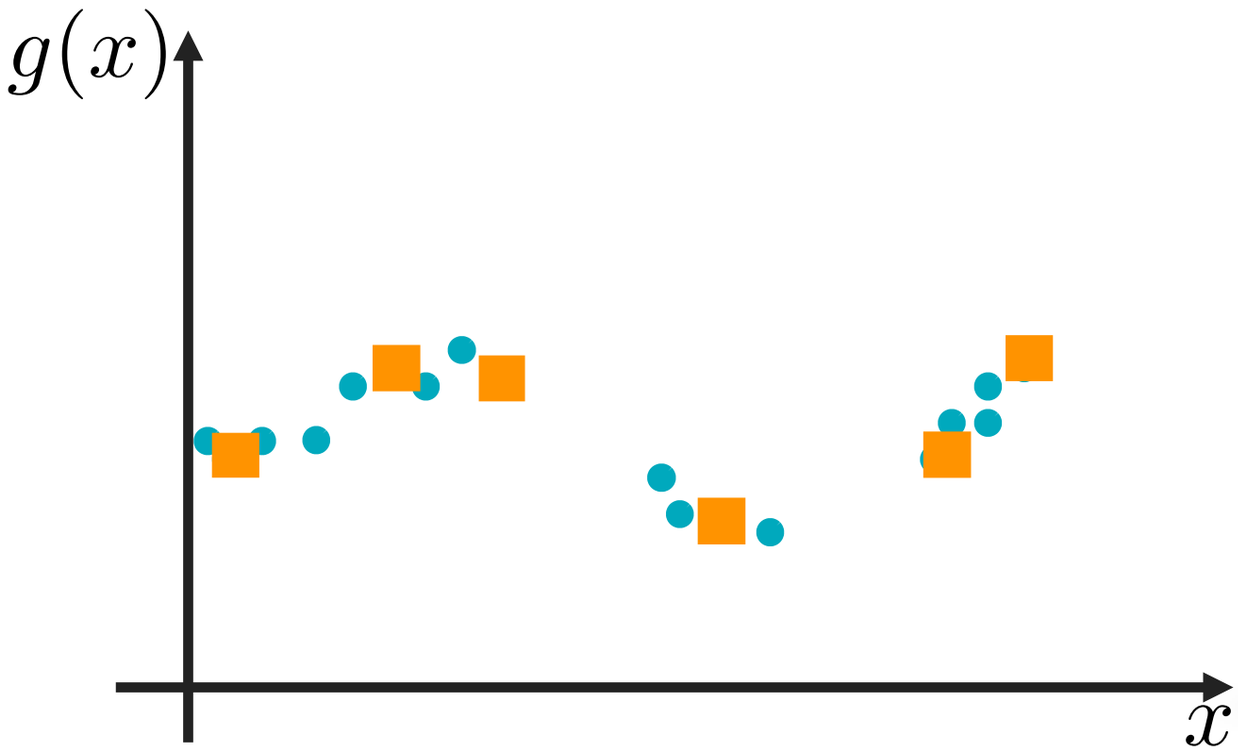


Nonparametric Regression



$$g_V(x) = \sum_{x' \in V} \alpha_{x'} k(x', x)$$

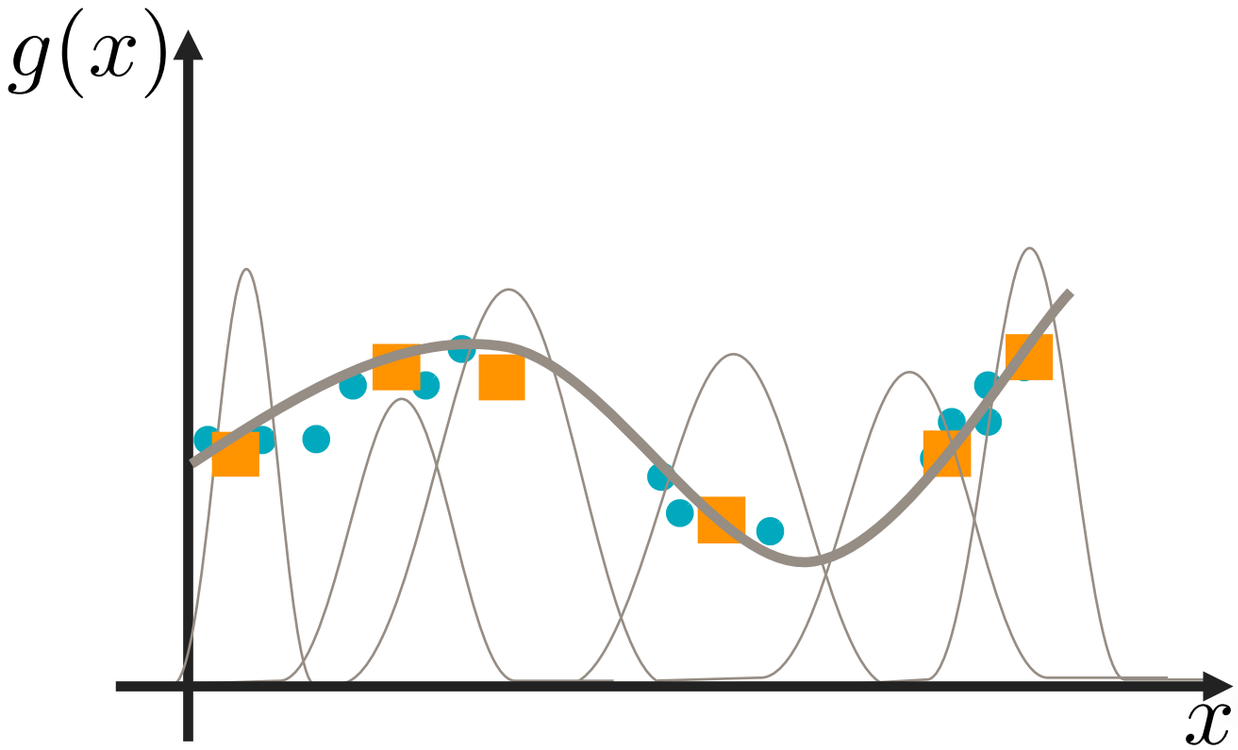
Nonparametric Regression



$$g_V(x) = \sum_{x' \in V} \alpha_{x'} k(x', x)$$

$$S \subseteq V, |S| \ll |V|$$

Nonparametric Regression

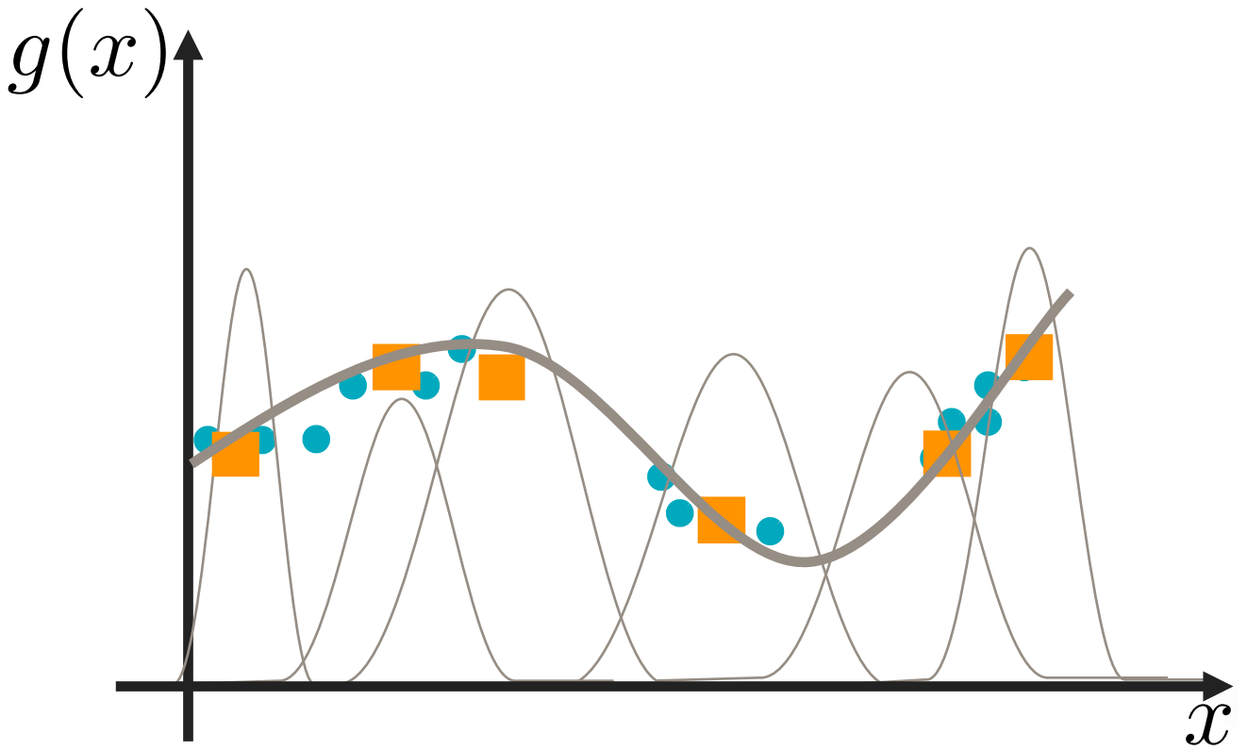


$$g_V(x) = \sum_{x' \in V} \alpha_{x'} k(x', x)$$

$$S \subseteq V, |S| \ll |V|$$

$$\approx g_S(x) = \sum_{x' \in S} \hat{\alpha}_{x'} k(x', x)$$

Nonparametric Regression



$$g_V(x) = \sum_{x' \in V} \alpha_{x'} k(x', x)$$

$$S \subseteq V, |S| \ll |V|$$

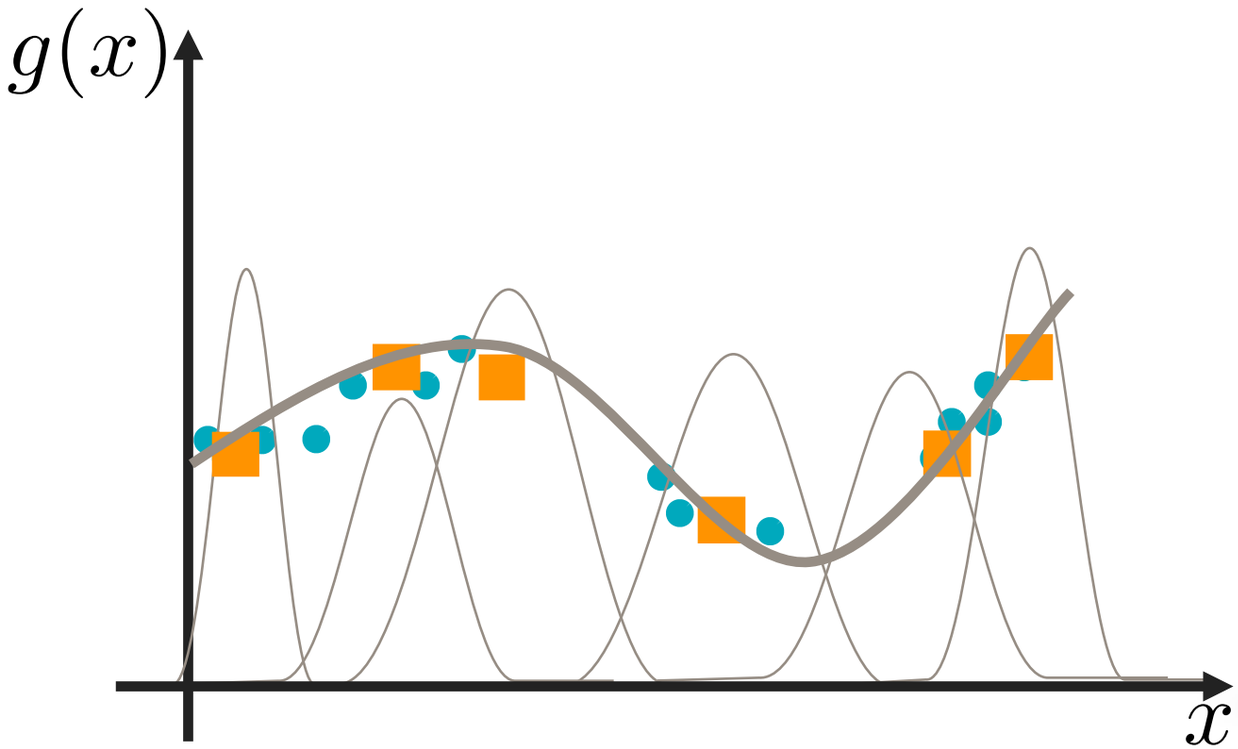
$$\approx g_S(x) = \sum_{x' \in S} \hat{\alpha}_{x'} k(x', x)$$

IVM: Pick a set to maximally capture variation

$$f(S) = \log\det(\mathbf{I} + \mathbf{K}_S)$$

$$\mathbf{K}_S = \begin{bmatrix} k(x_1, x_1) & \dots & k(x_1, x_{|S|}) \\ \vdots & & \vdots \\ k(x_{|S|}, x_1) & \dots & k(x_{|S|}, x_{|S|}) \end{bmatrix}$$

Nonparametric Regression



$$g_V(x) = \sum_{x' \in V} \alpha_{x'} k(x', x)$$

$$S \subseteq V, |S| \ll |V|$$

$$\approx g_S(x) = \sum_{x' \in S} \hat{\alpha}_{x'} k(x', x)$$

IVM: Pick a set to maximally capture variation

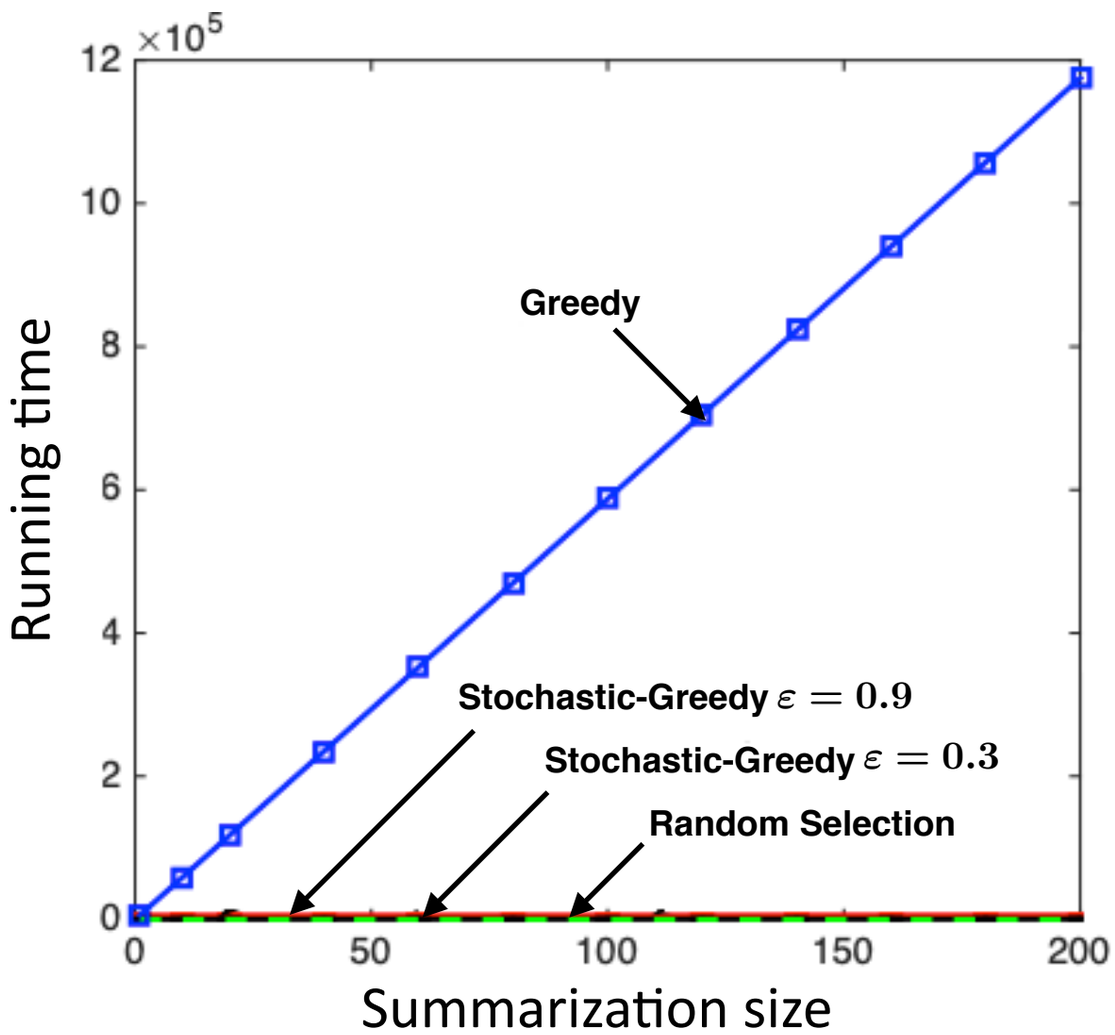
$$f(S) = \log\det(\mathbf{I} + \mathbf{K}_S)$$

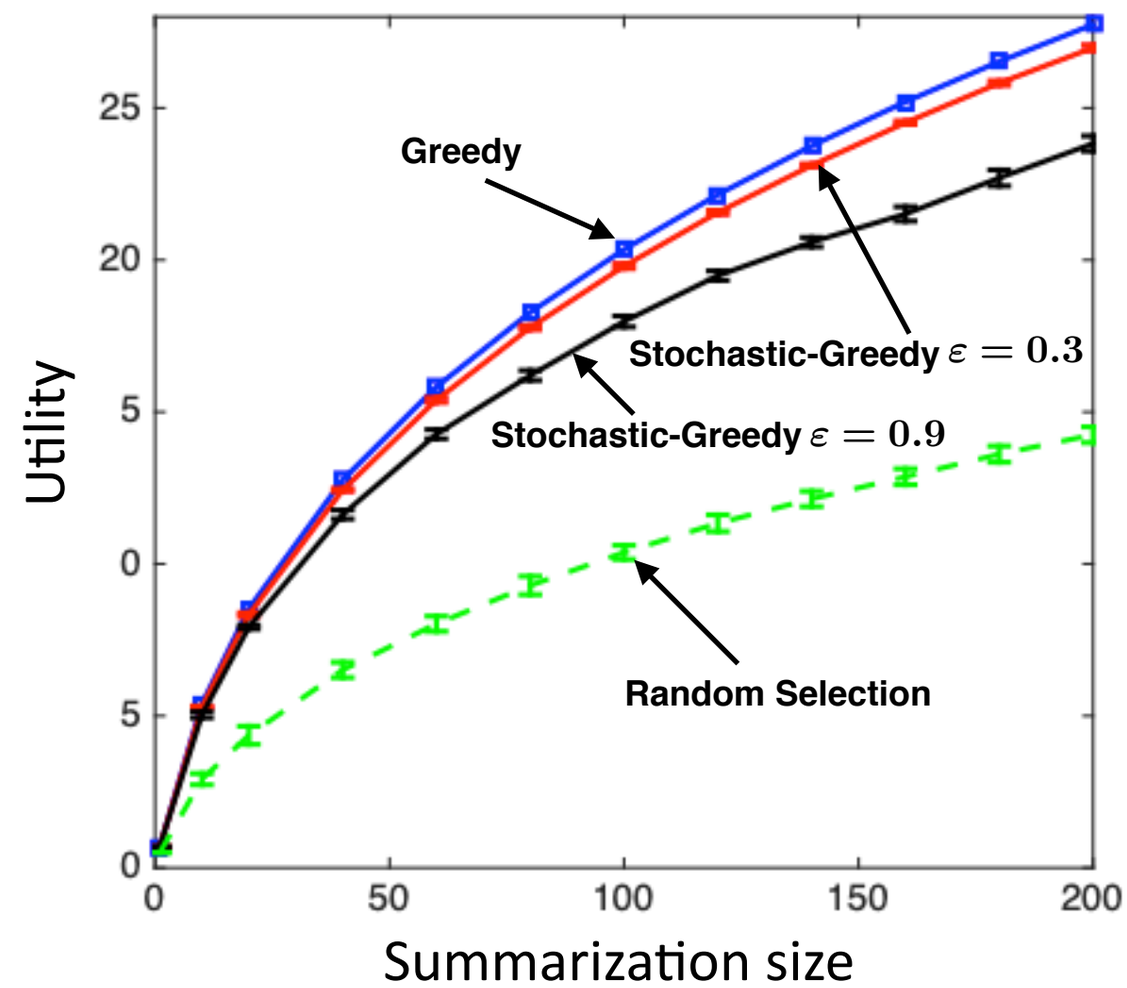
Monotone submodular

$$\mathbf{K}_S = \begin{bmatrix} k(x_1, x_1) & \dots & k(x_1, x_{|S|}) \\ \vdots & & \vdots \\ k(x_{|S|}, x_1) & \dots & k(x_{|S|}, x_{|S|}) \end{bmatrix}$$

Nonparametric Regression

- **Parkinsons Telemonitoring** dataset: 5,875 bio-medical voice measurements with 22 attributes from people with early-stage Parkinson's disease





Classic Greedy is too expensive

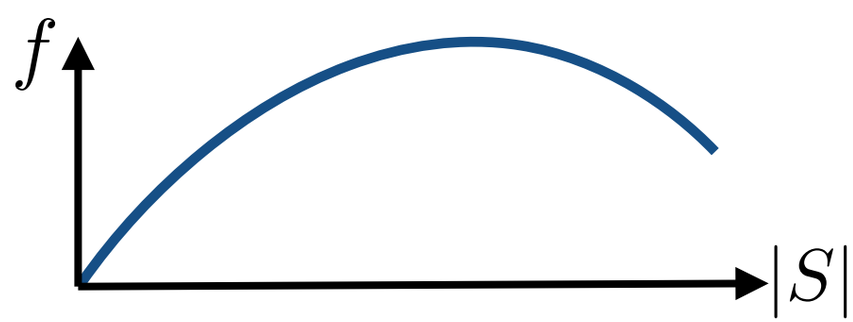
Non-Monotone Submodular Maximization

- Consider the following problem where f is submodular:

$$S^* = \arg\max_{S \in \mathcal{I}} f(S)$$

constraints

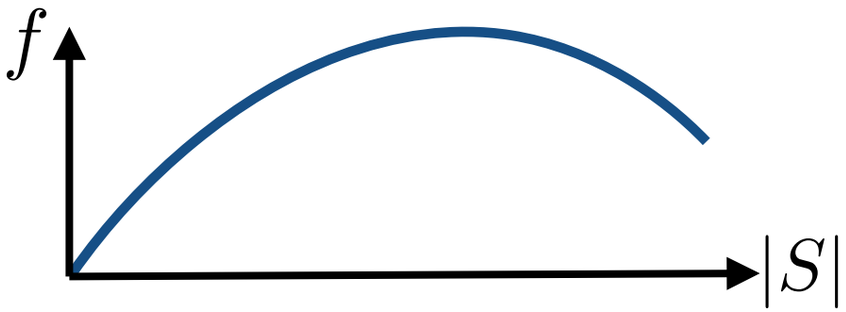
- When the utility functions f is non-monotone submodular



Unconstrained Submodular Maximization

- When the utility functions f is non-monotone submodular

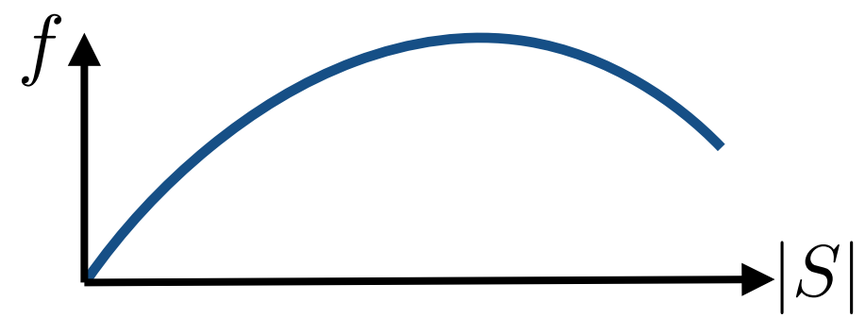
$$S^* = \arg\max_{S \subseteq V} f(S)$$



Unconstrained Submodular Maximization

- When the utility functions f is non-monotone submodular

$$S^* = \arg\max_{S \subseteq V} f(S)$$



- Greedy fails

$$V = \{u_1, u_2, \ldots, u_n\}$$

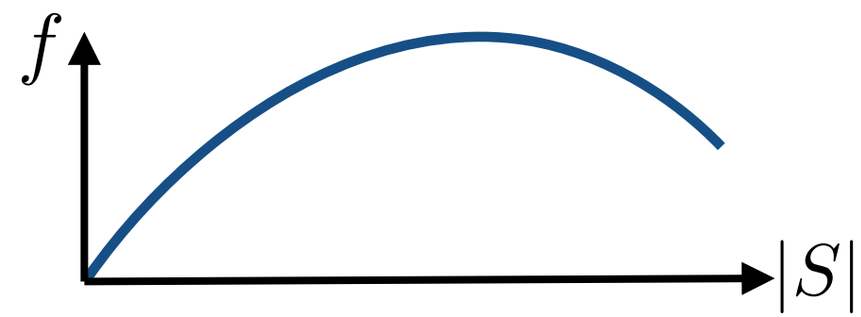
$$f(A) = \begin{cases} 1 + \epsilon & \text{if } u_1 \in A, \\ |A| & \text{if } u_1 \notin A. \end{cases}$$

"Submodular Functions Maximization Problems", Buchbinder, Feldman. 2017

Unconstrained Submodular Maximization

- When the utility functions f is non-monotone submodular

$$S^* = \arg\max_{S \subseteq V} f(S)$$



- Greedy fails

$$V = \{u_1, u_2, \ldots, u_n\}$$

Greedy

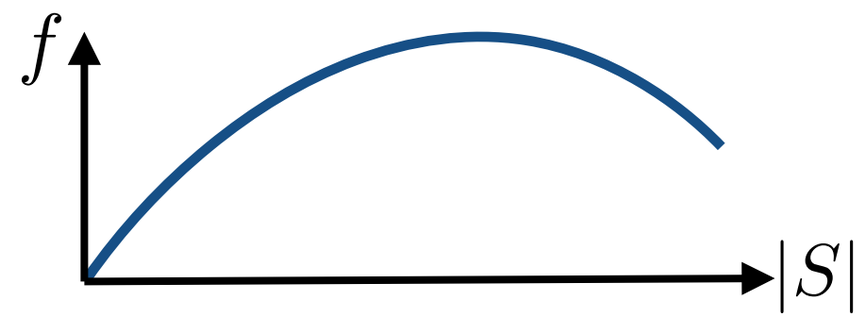
$$f(A) = \begin{cases} 1 + \epsilon & \text{if } u_1 \in A, \\ |A| & \text{if } u_1 \notin A. \end{cases}$$

"Submodular Functions Maximization Problems", Buchbinder, Feldman. 2017

Unconstrained Submodular Maximization

- When the utility functions f is non-monotone submodular

$$S^* = \arg\max_{S \subseteq V} f(S)$$



- Greedy fails

$$V = \{u_1, u_2, \ldots, u_n\}$$

$$f(A) = \begin{cases} 1 + \epsilon & \text{if } u_1 \in A, \\ |A| & \text{if } u_1 \notin A. \end{cases}$$

Greedy

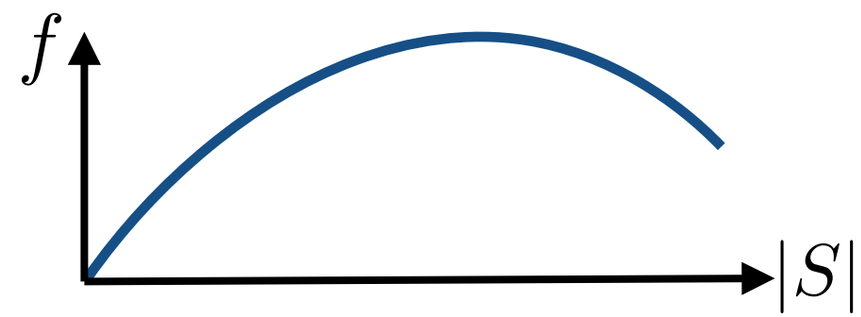
Optimum

"Submodular Functions Maximization Problems", Buchbinder, Feldman. 2017

Unconstrained Submodular Maximization

- When the utility functions f is non-monotone submodular

$$S^* = \arg\max_{S \subseteq V} f(S)$$



- Greedy fails

$$V = \{u_1, u_2, \ldots, u_n\}$$

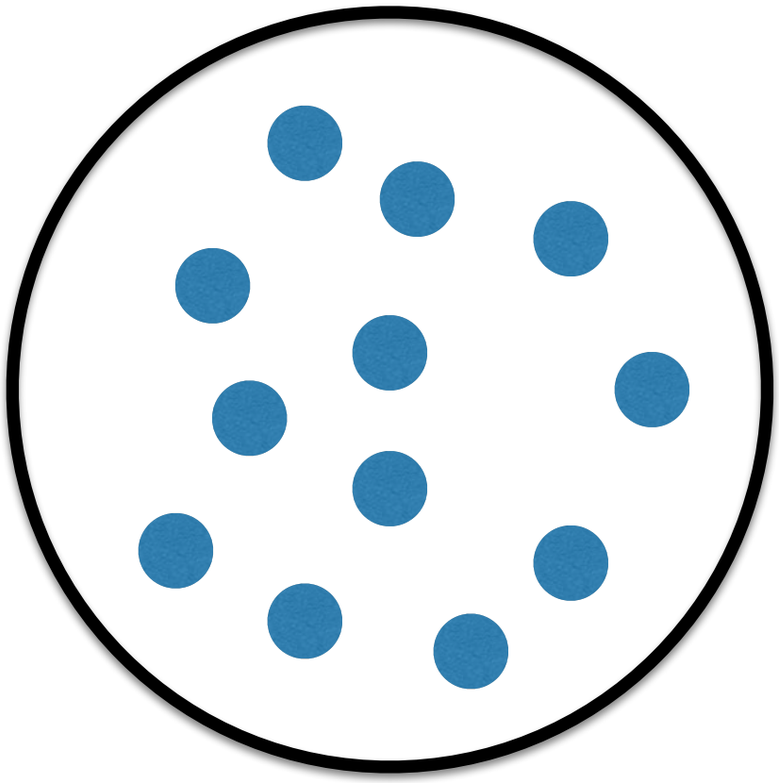
$$f(A) = \begin{cases} 1+\epsilon & \text{if } u_1 \in A, \\ |A| & \text{if } u_1 \notin A. \end{cases}$$

$$\frac{\text{Greedy}}{\text{Optimum}} = O\left(\frac{1+\epsilon}{n}\right)$$

"Submodular Functions Maximization Problems", Buchbinder, Feldman. 2017

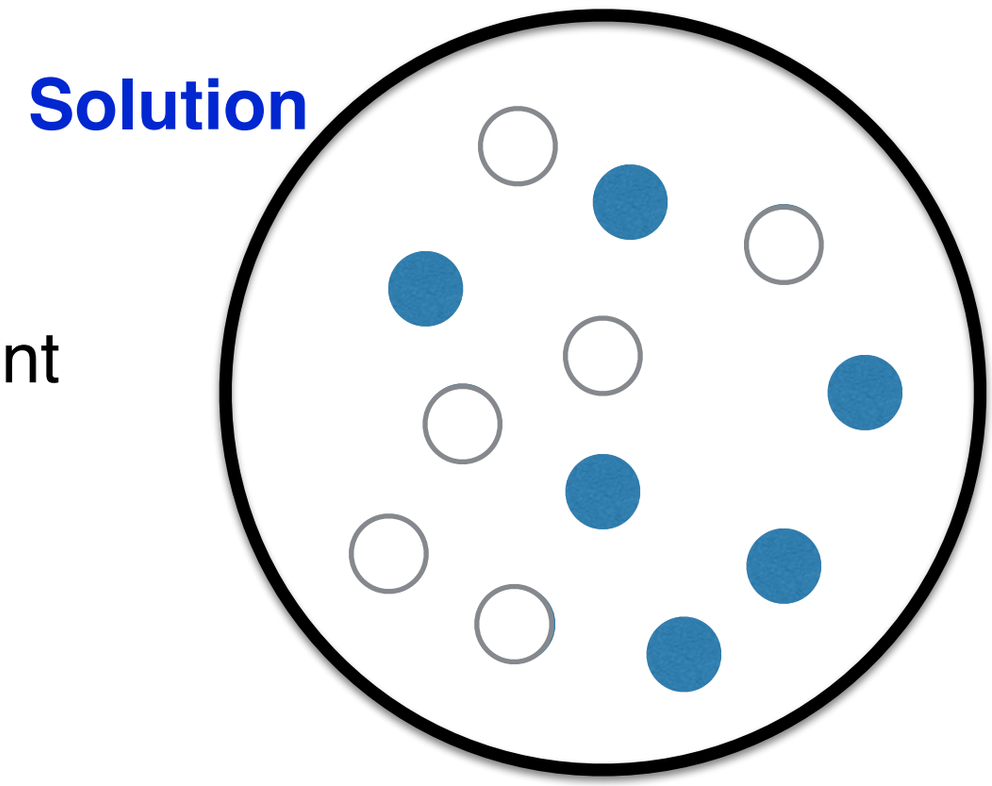
Unconstrained Submodular Maximization (Random)

- **Random Selection:** Choose each element independently with probability 1/2

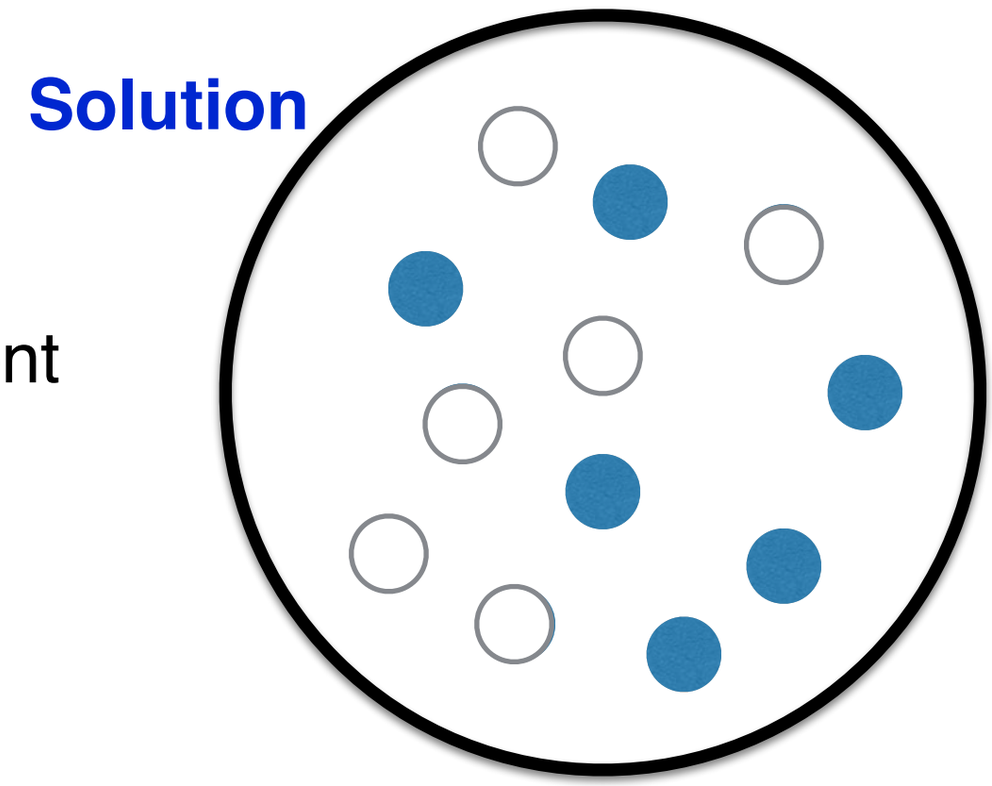


Unconstrained Submodular Maximization (Random)

- **Random Selection:** Choose each element independently with probability 1/2



Unconstrained Submodular Maximization (Random)



- **Random Selection:** Choose each element independently with probability 1/2

[Feige, Mirrokni, Vondrak]

In expectation, RandomSelection achieves

$$\mathbb{E}[f(S_{\text{Random}})] \geq \frac{1}{4} f(S^*)$$

"Maximizing Non-Monotone Submodular Functions", 2011

Double Greedy (Deterministic)

- S_1 starts as the empty set and S_2 starts with all elements.
- For each element, either add it to S_1 or remove it from S_2, depending on which action has the higher marginal gain.

"A Tight Linear Time (1/2)-Approximation For Unconstrained Submodular Maximization", 2015

$$S_1 = \{ \quad \}$$
$$S_2 = \{ v_1, \ v_2, \ v_3 \}$$

Double Greedy (Deterministic)

- S_1 starts as the empty set and S_2 starts with all elements.
- For each element, either add it to S_1 or remove it from S_2, depending on which action has the higher marginal gain.

"A Tight Linear Time (1/2)-Approximation For Unconstrained Submodular Maximization", 2015

Current element: v_1

$f(S_1 \cup \{v_1\}) - f(S_1) =$

$f(S_2 \setminus \{v_1\}) - f(S_2) =$

$$S_1 = \{ \quad \}$$

$$S_2 = \{ v_1, \ v_2, \ v_3 \}$$

Double Greedy (Deterministic)

- S_1 starts as the empty set and S_2 starts with all elements.
- For each element, either add it to S_1 or remove it from S_2, depending on which action has the higher marginal gain.

"A Tight Linear Time (1/2)-Approximation For Unconstrained Submodular Maximization", 2015

Current element: v_1

$f(S_1 \cup \{v_1\}) - f(S_1) =$

$f(S_2 \setminus \{v_1\}) - f(S_2) =$

Add v_1 to S_1

$$S_1 = \{ \quad \}$$

$$S_2 = \{ v_1, \ v_2, \ v_3 \}$$

Double Greedy (Deterministic)

- S_1 starts as the empty set and S_2 starts with all elements.
- For each element, either add it to S_1 or remove it from S_2, depending on which action has the higher marginal gain.

"A Tight Linear Time (1/2)-Approximation For Unconstrained Submodular Maximization", 2015

Current element: v_1

$f(S_1 \cup \{v_1\}) - f(S_1) =$

$f(S_2 \setminus \{v_1\}) - f(S_2) =$

Add v_1 to S_1

$$S_1 = \{ v_1 \}$$

$$S_2 = \{ v_1, v_2, v_3 \}$$

Double Greedy (Deterministic)

- S_1 starts as the empty set and S_2 starts with all elements.
- For each element, either add it to S_1 or remove it from S_2, depending on which action has the higher marginal gain.

Current element: v_2

$f(S_1 \cup \{v_2\}) - f(S_1) =$

$f(S_2 \setminus \{v_2\}) - f(S_2) =$

$$S_1 = \{ v_1 \quad \}$$

$$S_2 = \{ v_1, \ v_2, \ v_3 \}$$

Double Greedy (Deterministic)

- S_1 starts as the empty set and S_2 starts with all elements.
- For each element, either add it to S_1 or remove it from S_2, depending on which action has the higher marginal gain.

Current element: v_2

$f(S_1 \cup \{v_2\}) - f(S_1) =$ $f(S_2 \setminus \{v_2\}) - f(S_2) =$

$$S_1 = \{ v_1 \quad \}$$

$$S_2 = \{ v_1, \ v_2, \ v_3 \}$$

Double Greedy (Deterministic)

- S_1 starts as the empty set and S_2 starts with all elements.
- For each element, either add it to S_1 or remove it from S_2, depending on which action has the higher marginal gain.

Current element: v_2

$f(S_1 \cup \{v_2\}) - f(S_1) =$

$f(S_2 \setminus \{v_2\}) - f(S_2) =$

$$S_1 = \{ v_1 \quad \}$$

$$S_2 = \{ v_1, \quad v_3 \}$$

Remove v_2 from S_2

Double Greedy (Deterministic)

- S_1 starts as the empty set and S_2 starts with all elements.
- For each element, either add it to S_1 or remove it from S_2, depending on which action has the higher marginal gain.

Current element: v_3

$f(S_1 \cup \{v_3\}) - f(S_1) =$

$f(S_2 \setminus \{v_3\}) - f(S_2) =$

$$S_1 = \{ v_1 \quad \}$$

$$S_2 = \{ v_1, \quad v_3 \}$$

Double Greedy (Deterministic)

- S_1 starts as the empty set and S_2 starts with all elements.
- For each element, either add it to S_1 or remove it from S_2, depending on which action has the higher marginal gain.

Current element: v_3

$f(S_1 \cup \{v_3\}) - f(S_1) =$ ■ $\qquad$ $f(S_2 \setminus \{v_3\}) - f(S_2) =$ ■

$$S_1 = \{ v_1 \qquad \}$$

$$S_2 = \{ v_1, \qquad v_3 \}$$

Double Greedy (Deterministic)

- S_1 starts as the empty set and S_2 starts with all elements.
- For each element, either add it to S_1 or remove it from S_2, depending on which action has the higher marginal gain.

Current element: v_3

$f(S_1 \cup \{v_3\}) - f(S_1) =$

$f(S_2 \setminus \{v_3\}) - f(S_2) =$

$$S_1 = \{ v_1 \quad \}$$

$$S_2 = \{ v_1 \quad \}$$

Remove v_3 from S_2

Double Greedy (Deterministic)

- S_1 starts as the empty set and S_2 starts with all elements.
- For each element, either add it to S_1 or remove it from S_2, depending on which action has the higher marginal gain.

Solution

$$S_1 = \{ v_1 \quad \}$$
$$S_2 = \{ v_1 \quad \}$$

Double Greedy (Deterministic)

- S_1 starts as the empty set and S_2 starts with all elements.
- For each element, either add it to S_1 or remove it from S_2, depending on which action has the higher marginal gain.

[Buchbinder, Feldman, Naor, Schwartz]

Double Greedy achieves a (1/3)-approximation ratio and runs in $O(n)$ time:

$$f(S_1) = f(S_2) \geq f(OPT)/3$$

"A Tight Linear Time (1/2)-Approximation For Unconstrained Submodular Maximization", 2015

Double Greedy (Randomized)

- S_1 starts as the empty set and S_2 starts with all elements.
- For each element, either add it to S_1 or remove it from S_2, with probability proportional to the marginal gain.

$$S_1 = \{ \qquad \}$$
$$S_2 = \{ v_1, \ v_2, \ v_3 \}$$

Double Greedy (Randomized)

- S_1 starts as the empty set and S_2 starts with all elements.
- For each element, either add it to S_1 or remove it from S_2, with probability proportional to the marginal gain.

Current element: v_1

$f(S_1 \cup \{v_1\}) - f(S_1) =$ $f(S_2 \setminus \{v_1\}) - f(S_2) =$

$$S_1 = \{ \quad \}$$

$$S_2 = \{ v_1, \ v_2, \ v_3 \}$$

Double Greedy (Randomized)

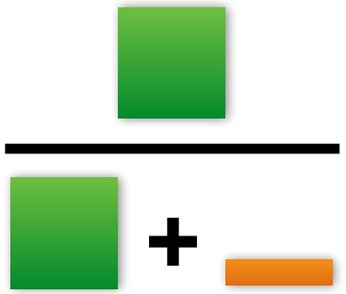
- S_1 starts as the empty set and S_2 starts with all elements.
- For each element, either add it to S_1 or remove it from S_2, with probability proportional to the marginal gain.

Current element: v_1

$f(S_1 \cup \{v_1\}) - f(S_1) =$

$f(S_2 \setminus \{v_1\}) - f(S_2) =$

Add v_1 to S_1 with probability



$S_1 = \{ \quad \}$

$S_2 = \{ v_1, v_2, v_3 \}$

Double Greedy (Randomized)

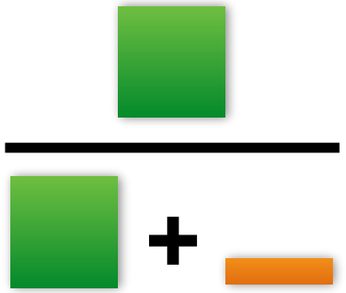
- S_1 starts as the empty set and S_2 starts with all elements.
- For each element, either add it to S_1 or remove it from S_2, with probability proportional to the marginal gain.

Current element: v_1

$f(S_1 \cup \{v_1\}) - f(S_1) =$

$f(S_2 \setminus \{v_1\}) - f(S_2) =$

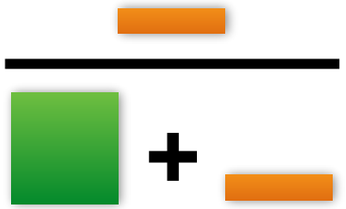
Add v_1 to S_1 with probability



$S_1 = \{ \quad \}$

$S_2 = \{ v_1, v_2, v_3 \}$

Remove v_1 from S_2 with probability



Double Greedy (Randomized)

- S_1 starts as the empty set and S_2 starts with all elements.
- For each element, either add it to S_1 or remove it from S_2, with probability proportional to the marginal gain.

Current element: v_1

$f(S_1 \cup \{v_1\}) - f(S_1) =$

$f(S_2 \setminus \{v_1\}) - f(S_2) =$

Add v_1 to S_1 with probability

Remove v_1 from S_2 with probability

$$S_1 = \{ v_1 \}$$

$$S_2 = \{ v_1, v_2, v_3 \}$$

Double Greedy (Randomized)

- S_1 starts as the empty set and S_2 starts with all elements.
- For each element, either add it to S_1 or remove it from S_2, with probability proportional to the marginal gain.

Current element: v_2

$f(S_1 \cup \{v_2\}) - f(S_1) =$

$f(S_2 \setminus \{v_2\}) - f(S_2) =$

$$S_1 = \{ v_1 \quad \}$$

$$S_2 = \{ v_1, \ v_2, \ v_3 \}$$

Double Greedy (Randomized)

- S_1 starts as the empty set and S_2 starts with all elements.
- For each element, either add it to S_1 or remove it from S_2, with probability proportional to the marginal gain.

Current element: v_2

$f(S_1 \cup \{v_2\}) - f(S_1) =$ $f(S_2 \setminus \{v_2\}) - f(S_2) =$

$$S_1 = \{ v_1 \quad \}$$

$$S_2 = \{ v_1, \ v_2, \ v_3 \}$$

Double Greedy (Randomized)

- S_1 starts as the empty set and S_2 starts with all elements.
- For each element, either add it to S_1 or remove it from S_2, with probability proportional to the marginal gain.

Current element: v_2

$f(S_1 \cup \{v_2\}) - f(S_1) =$

$f(S_2 \setminus \{v_2\}) - f(S_2) =$

Add v_2 to S_1 with probability

Remove v_2 from S_2 with probability

$$S_1 = \{ v_1 \}$$

$$S_2 = \{ v_1, v_2, v_3 \}$$

Double Greedy (Randomized)

- S_1 starts as the empty set and S_2 starts with all elements.
- For each element, either add it to S_1 or remove it from S_2, with probability proportional to the marginal gain.

Current element: v_2

$f(S_1 \cup \{v_2\}) - f(S_1) =$

$f(S_2 \setminus \{v_2\}) - f(S_2) =$

Add v_2 to S_1 with probability

$$\frac{\quad}{\quad + \quad}$$

$$S_1 = \{ v_1, \ v_2 \}$$

$$S_2 = \{ v_1, \ v_2, \ v_3 \}$$

Remove v_2 from S_2 with probability

$$\frac{\quad}{\quad + \quad}$$

Double Greedy (Randomized)

- S_1 starts as the empty set and S_2 starts with all elements.
- For each element, either add it to S_1 or remove it from S_2, with probability proportional to the marginal gain.

Current element: v_3

$f(S_1 \cup \{v_3\}) - f(S_1) =$

$f(S_2 \setminus \{v_3\}) - f(S_2) =$

$$S_1 = \{ v_1, \ v_2 \quad \}$$

$$S_2 = \{ v_1, \ v_2, \ v_3 \}$$

Double Greedy (Randomized)

- S_1 starts as the empty set and S_2 starts with all elements.
- For each element, either add it to S_1 or remove it from S_2, with probability proportional to the marginal gain.

Current element: v_3

$f(S_1 \cup \{v_3\}) - f(S_1) =$ ■

$f(S_2 \setminus \{v_3\}) - f(S_2) =$ ■

If one of the actions has negative marginal gain...

$$S_1 = \{ v_1, \ v_2 \quad \}$$

Deterministically take the other action.

$$S_2 = \{ v_1, \ v_2 \quad \}$$

Remove v_3 from S_2

Double Greedy (Randomized)

- S_1 starts as the empty set and S_2 starts with all elements.
- For each element, either add it to S_1 or remove it from S_2, with probability proportional to the marginal gain.

Solution

$$S_1 = \{ v_1, \ v_2 \quad \}$$
$$S_2 = \{ v_1, \ v_2 \quad \}$$

Double Greedy

- S_1 starts as the empty set and S_2 starts with all elements.
- For each element, either add it to S_1 or remove it from S_2, depending on which action has the higher marginal gain.

[Buchbinder, Feldman, Naor, Schwartz]

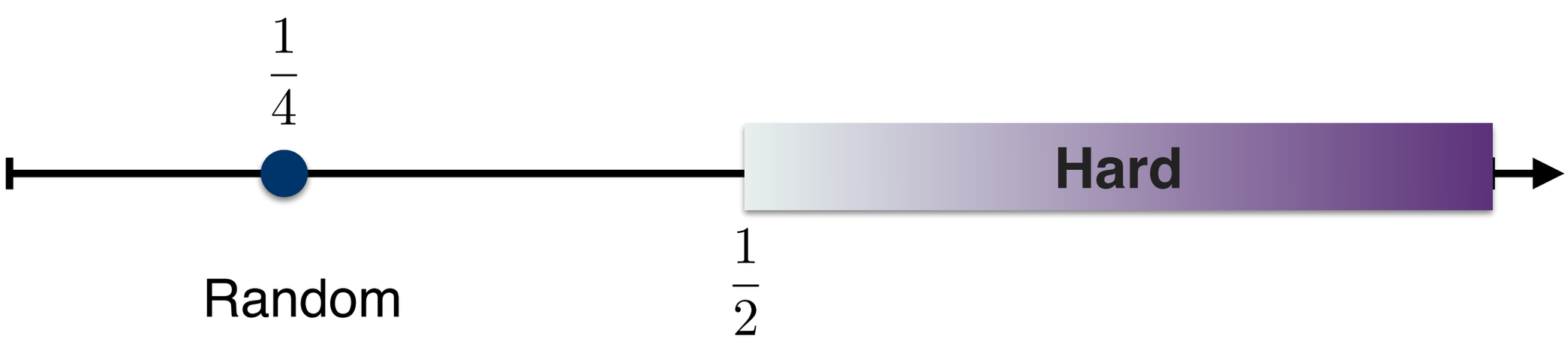
In expectation, Randomized Double Greedy achieves a (1/2)-approximation ratio and runs in $O(n)$ time:

$$\mathbb{E}\big[f(S_1)\big] = \mathbb{E}\big[f(S_2)\big] \geq f(OPT)/2$$

"A Tight Linear Time (1/2)-Approximation For Unconstrained Submodular Maximization", 2015

Unconstrained Submodular Maximization

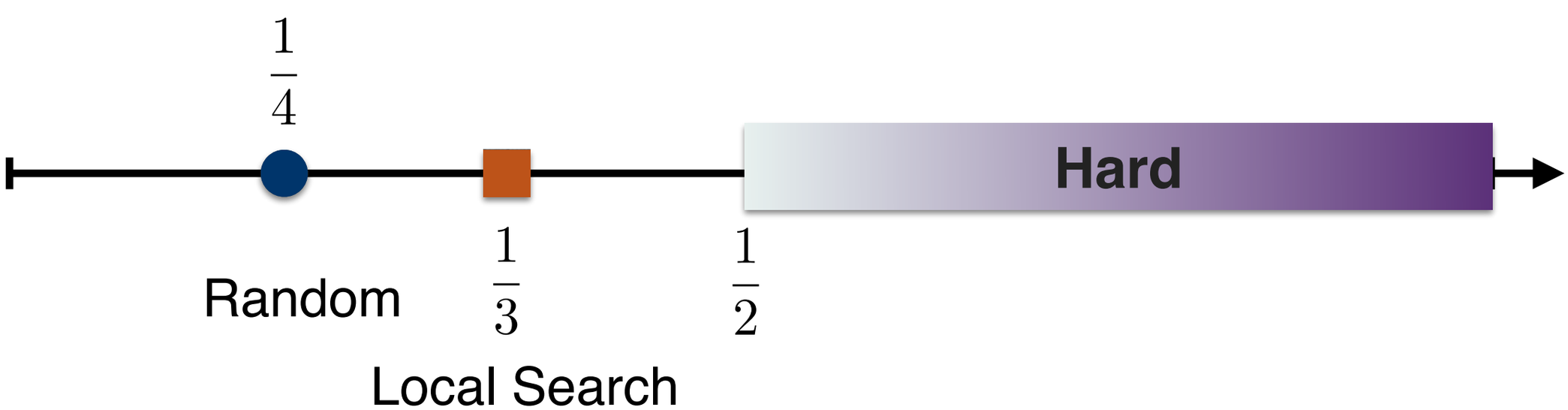
Unconstrained Submodular Maximization



Maximizing non-monotone submodular functions

Uriel Feige* Vahab S. Mirrokni* Jan Vondrák†

Unconstrained Submodular Maximization



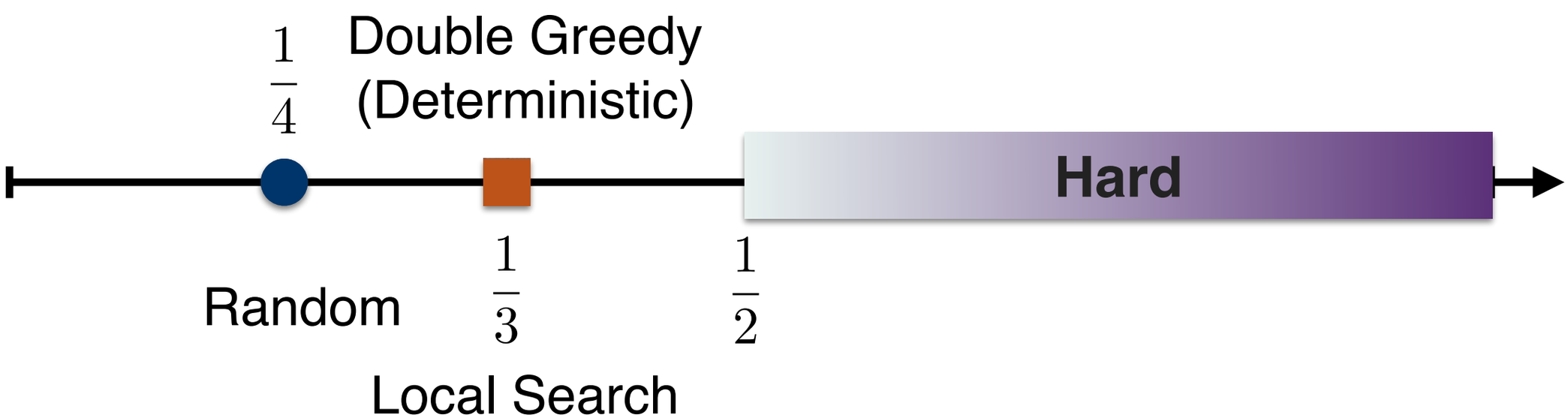
Maximizing non-monotone submodular functions

Uriel Feige* Vahab S. Mirrokni* Jan Vondrák†

Unconstrained Submodular Maximization

A TIGHT LINEAR TIME (1/2)-APPROXIMATION FOR UNCONSTRAINED SUBMODULAR MAXIMIZATION*

NIV BUCHBINDER†, MORAN FELDMAN‡, JOSEPH (SEFFI) NAOR§, AND ROY SCHWARTZ¶



Maximizing non-monotone submodular functions

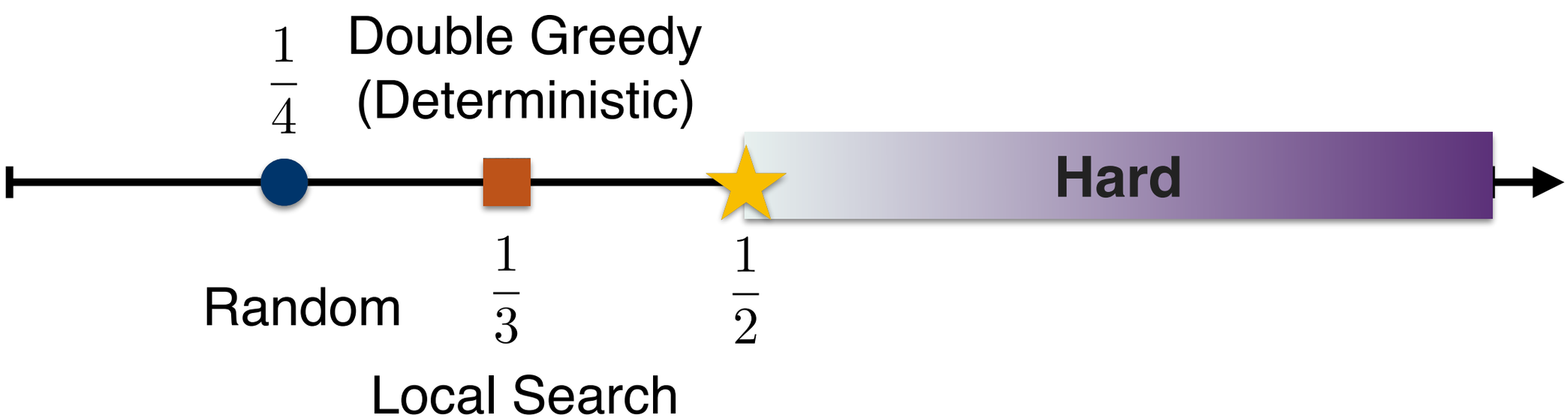
Uriel Feige* Vahab S. Mirrokni* Jan Vondrák†

Unconstrained Submodular Maximization

A TIGHT LINEAR TIME (1/2)-APPROXIMATION FOR UNCONSTRAINED SUBMODULAR MAXIMIZATION*

NIV BUCHBINDER†, MORAN FELDMAN‡, JOSEPH (SEFFI) NAOR§, AND ROY SCHWARTZ¶

Double Greedy
(Random)



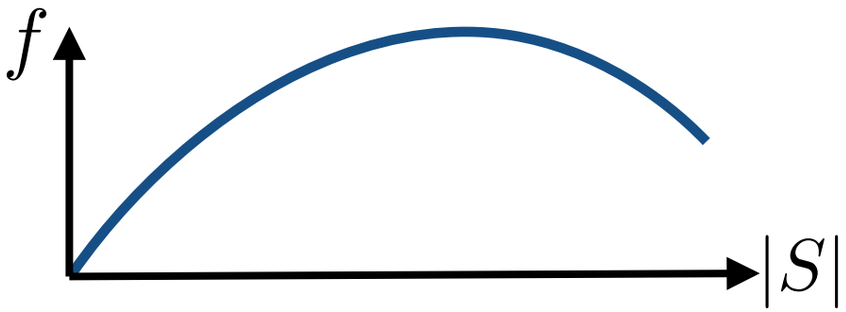
Maximizing non-monotone submodular functions

Uriel Feige* Vahab S. Mirrokni* Jan Vondrák†

Cardinality Constrained Submodular Maximization

- When the utility function f is non-monotone submodular

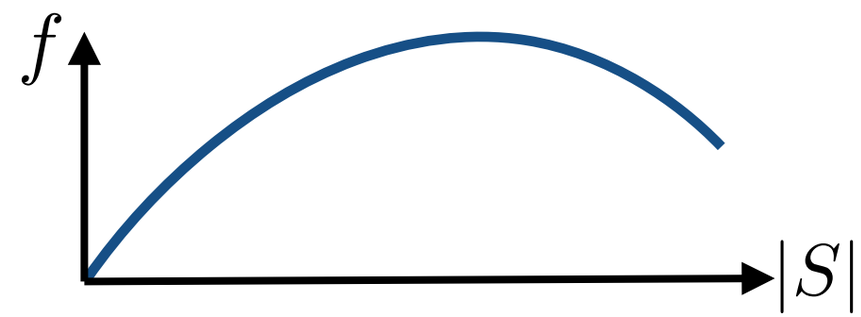
$$S^* = \arg\max_{|S| \leq k} f(S)$$



Cardinality Constrained Submodular Maximization

- When the utility function f is non-monotone submodular

$$S^* = \arg\max_{|S| \leq k} f(S)$$



- Greedy fails

$$V = \{u_1, u_2, \ldots, u_k, \ldots u_n\}$$

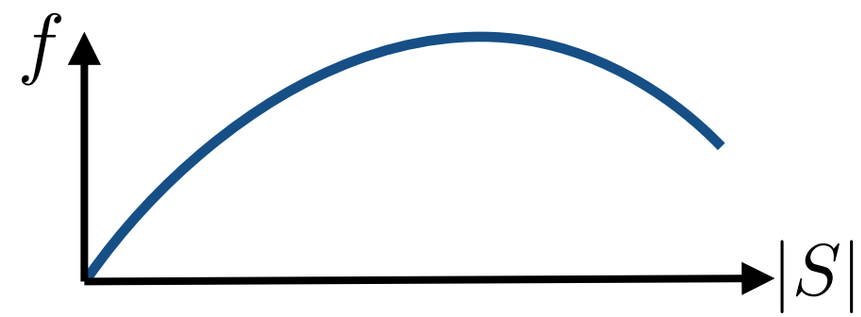
$$f(A) = \begin{cases} 1 + \epsilon & \text{if } u_1 \in A, \\ |A| & \text{if } u_1 \notin A. \end{cases}$$

"Submodular Functions Maximization Problems", Buchbinder, Feldman. 2017

Cardinality Constrained Submodular Maximization

- When the utility function f is non-monotone submodular

$$S^* = \arg\max_{|S| \leq k} f(S)$$



- Greedy fails

$$V = \{u_1, u_2, \ldots, u_k, \ldots u_n\}$$

Greedy

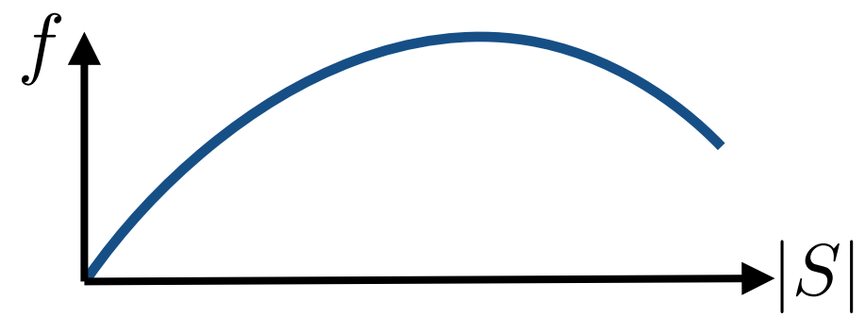
$$f(A) = \begin{cases} 1 + \epsilon & \text{if } u_1 \in A, \\ |A| & \text{if } u_1 \notin A. \end{cases}$$

"Submodular Functions Maximization Problems", Buchbinder, Feldman. 2017

Cardinality Constrained Submodular Maximization

- When the utility function f is non-monotone submodular

$$S^* = \arg\max_{|S| \leq k} f(S)$$



- Greedy fails

$$V = \{u_1, u_2, \ldots, u_k, \ldots u_n\}$$

$$f(A) = \begin{cases} 1 + \epsilon & \text{if } u_1 \in A, \\ |A| & \text{if } u_1 \notin A. \end{cases}$$

Greedy

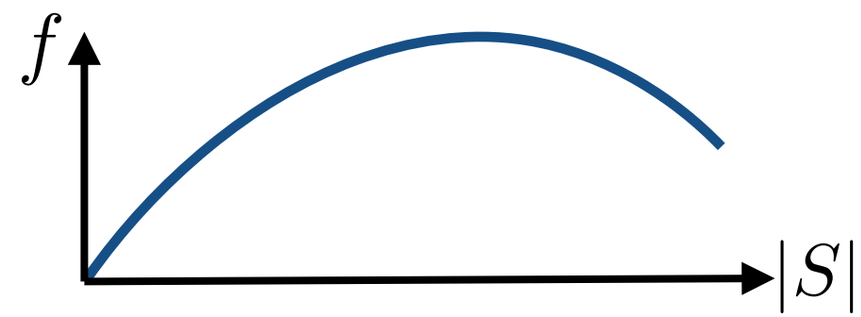
Optimum

"Submodular Functions Maximization Problems", Buchbinder, Feldman. 2017

Cardinality Constrained Submodular Maximization

- When the utility function f is non-monotone submodular

$$S^* = \arg\max_{|S| \leq k} f(S)$$



- Greedy fails

$$V = \{u_1, u_2, \ldots, u_k, \ldots u_n\}$$

$$f(A) = \begin{cases} 1+\epsilon & \text{if } u_1 \in A, \\ |A| & \text{if } u_1 \notin A. \end{cases}$$

$$\frac{\text{Greedy}}{\text{Optimum}} = O\left(\frac{1+\epsilon}{k}\right)$$

"Submodular Functions Maximization Problems", Buchbinder, Feldman. 2017

Random Greedy

- For k iterations:
    1. Identify the top k items with the highest marginal gain.
    2. Select one uniformly at random and add it to S.

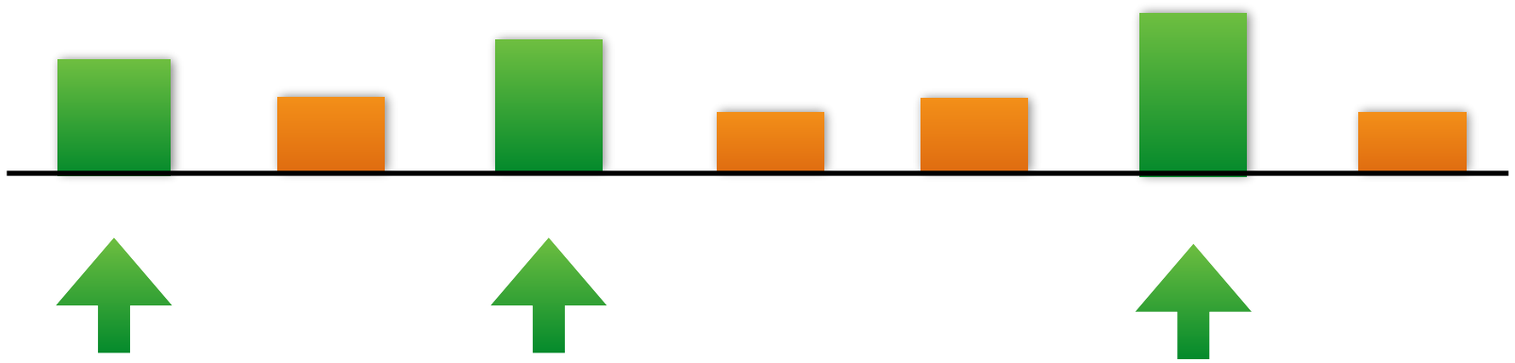
Random Greedy

- For k iterations:

  1. Identify the top k items with the highest marginal gain.
  2. Select one uniformly at random and add it to S.

Iteration 1 of 3

Marginal gains:



Top 3 marginal gains

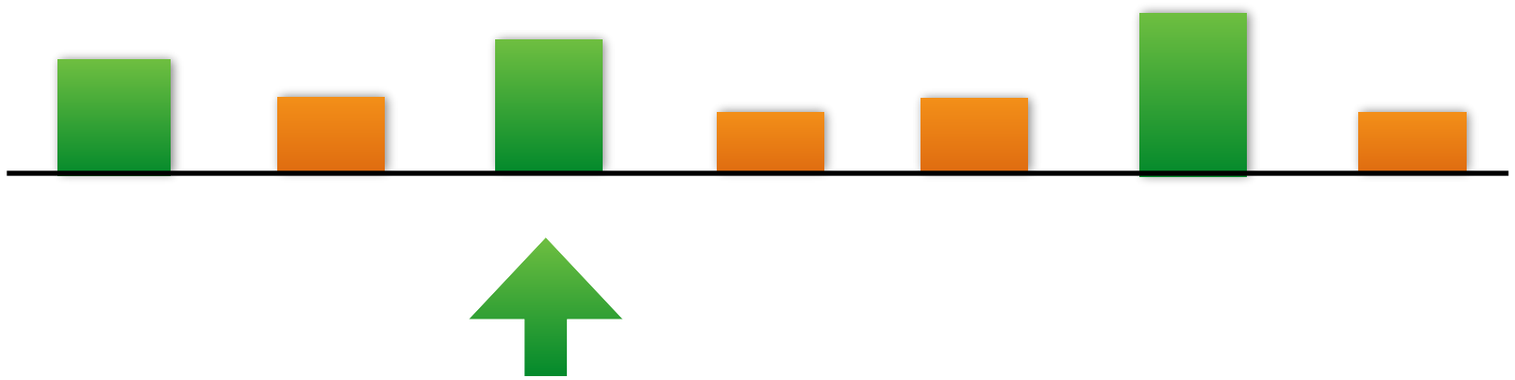
$$S = \{ \qquad \}$$

Random Greedy

- For k iterations:
  1. Identify the top k items with the highest marginal gain.
  2. Select one uniformly at random and add it to S.

Iteration 1 of 3

Marginal gains:



Select one uniformly at random

$S = \{$ $\}$

Random Greedy

- For k iterations:

  1. Identify the top k items with the highest marginal gain.
  2. Select one uniformly at random and add it to S.

Iteration 2 of 3

Marginal gains:

$S = \{$ $\}$

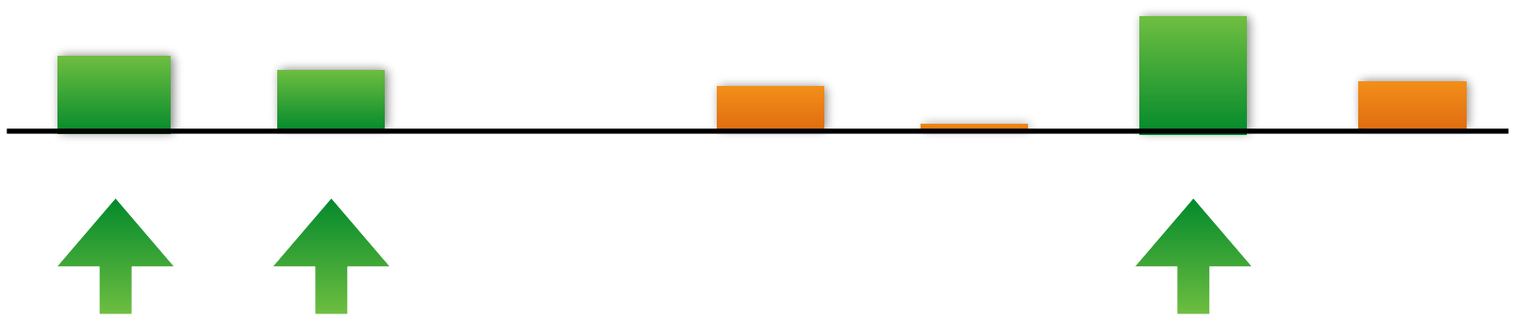
Random Greedy

- For k iterations:

  1. Identify the top k items with the highest marginal gain.
  2. Select one uniformly at random and add it to S.

Iteration 2 of 3

Marginal gains:



Top 3 marginal gains

$S = \{$ $\}$

Random Greedy

- For k iterations:

  1. Identify the top k items with the highest marginal gain.
  2. Select one uniformly at random and add it to S.

Iteration 2 of 3

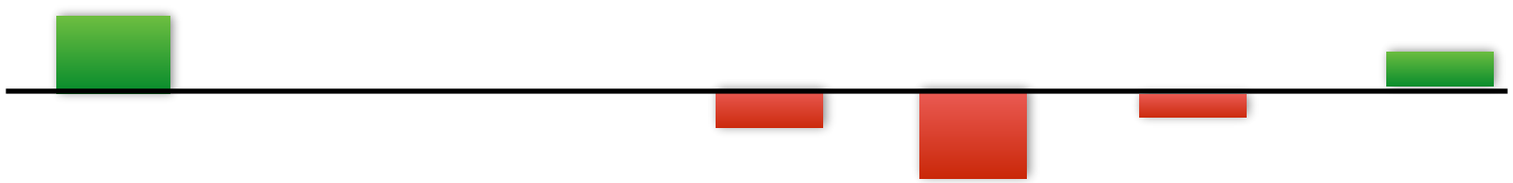
Random Greedy

- For k iterations:

  1. Identify the top k items with the highest marginal gain.
  2. Select one uniformly at random and add it to S.

Iteration 3 of 3

Marginal gains:



$S = \{$, $\}$

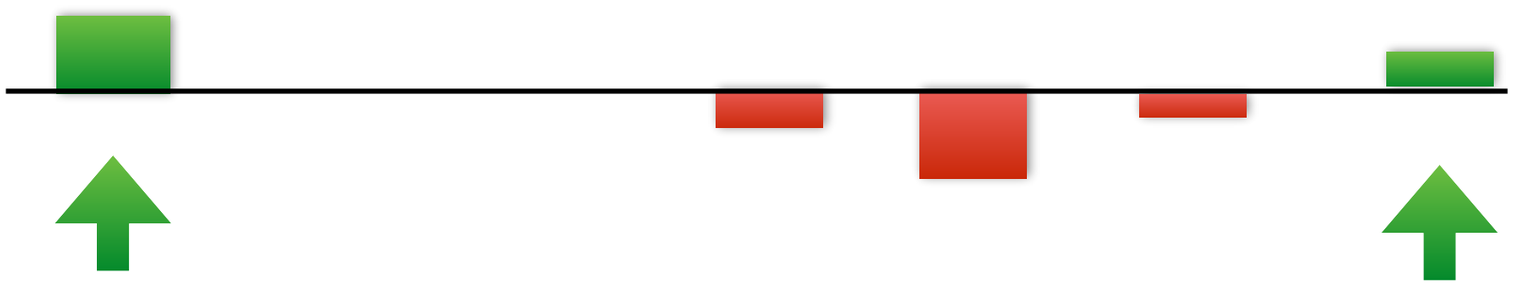
Random Greedy

- For k iterations:

  1. Identify the top k items with the highest marginal gain.
  2. Select one uniformly at random and add it to S.

Iteration 3 of 3

Marginal gains:



Note: we only consider items with positive marginal gain

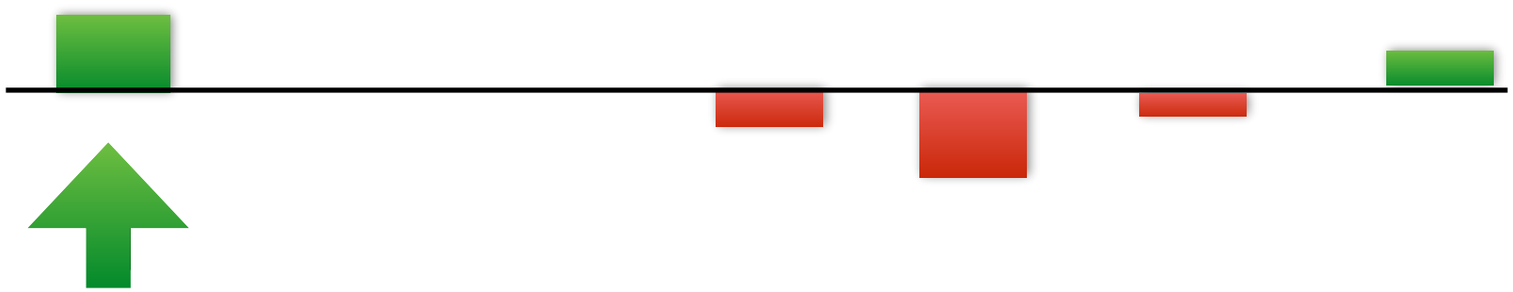
$S = \{$, $\}$

Random Greedy

- For k iterations:
  1. Identify the top k items with the highest marginal gain.
  2. Select one uniformly at random and add it to S.

Iteration 3 of 3

Marginal gains:



Select one uniformly at random

$S = \{$, , $\}$

Random Greedy

- For k iterations:
    1. Identify the top k items with the highest marginal gain.
    2. Select one uniformly at random and add it to S.

Solution

$S = \{$, , $\}$

Random Greedy

- For k iterations:
  1. Identify the top k items with the highest marginal gain.
  2. Select one uniformly at random and add it to S.

[Buchbinder, Feldman, Naor, Schwartz]

If f is monotone submodular then Random Greedy achieves a $(1 - \frac{1}{e})$-approximation ratio:

$$\mathbb{E}[f(S)] \geq \left(1 - \frac{1}{e}\right) \mathrm{OPT}$$

If f is non-monotone submodular then Random Greedy achieves a $(\frac{1}{e})$-approximation ratio:

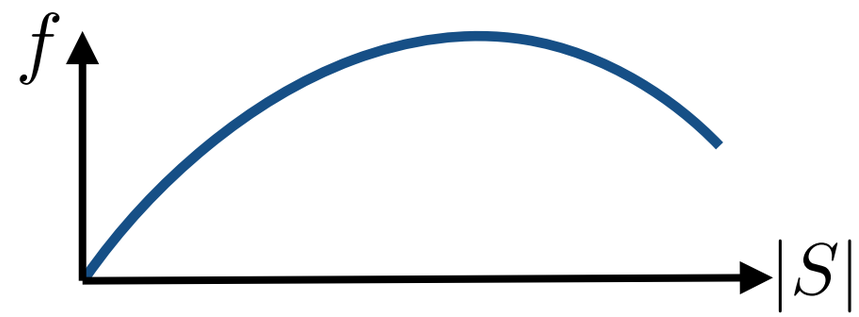
$$\mathbb{E}[f(S)] \geq \left(\frac{1}{e}\right) \mathrm{OPT}$$

"Submodular Maximization With Cardinality Constraints", 2014

Open Question: Cardinality Constrained Submodular Maximization

- When the utility function f is non-monotone submodular

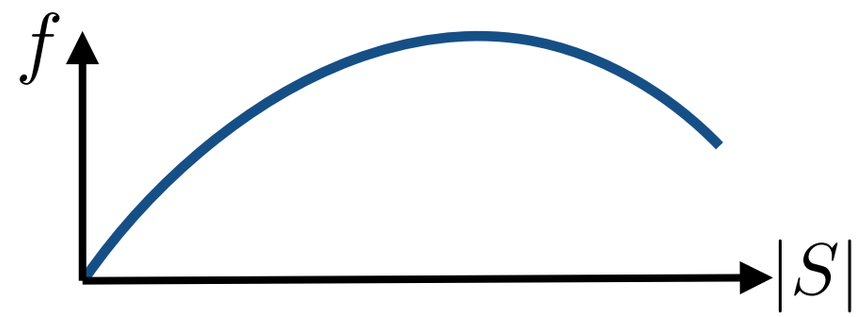
$$S^* = \arg\max_{|S| \leq k} f(S)$$



Open Question: Cardinality Constrained Submodular Maximization

- When the utility function f is non-monotone submodular

$$S^* = \arg\max_{|S| \le k} f(S)$$



Constrained Submodular Maximization via a Non-symmetric Technique

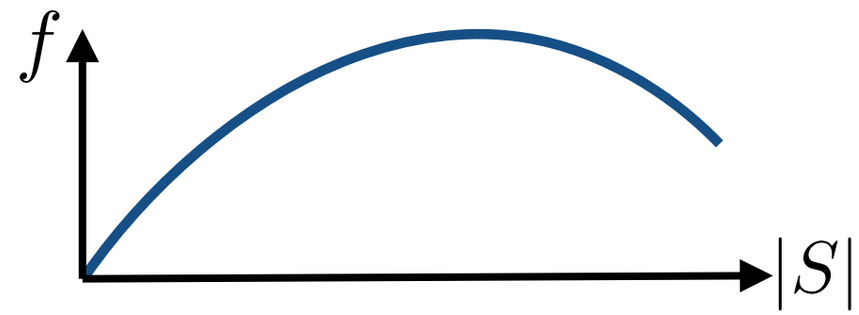
Niv Buchbinder* Moran Feldman†

2016

Open Question: Cardinality Constrained Submodular Maximization

- When the utility function f is non-monotone submodular

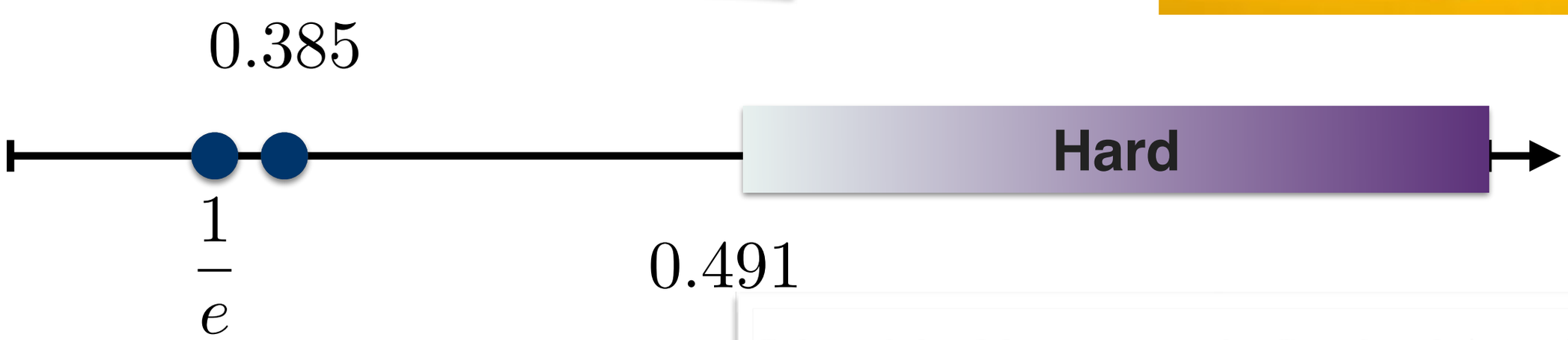
$$S^* = \arg\max_{|S| \le k} f(S)$$



Constrained Submodular Maximization via a Non-symmetric Technique

Niv Buchbinder* Moran Feldman†

2016



Submodular Maximization by Simulated Annealing

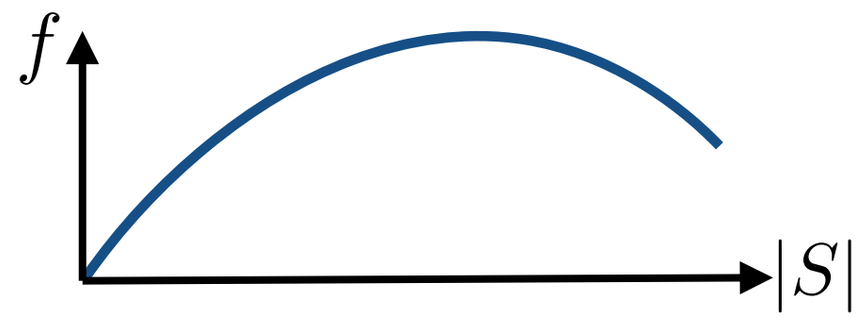
Shayan Oveis Gharan* Jan Vondrák†

2010

Open Question: Cardinality Constrained Submodular Maximization

- When the utility function f is non-monotone submodular

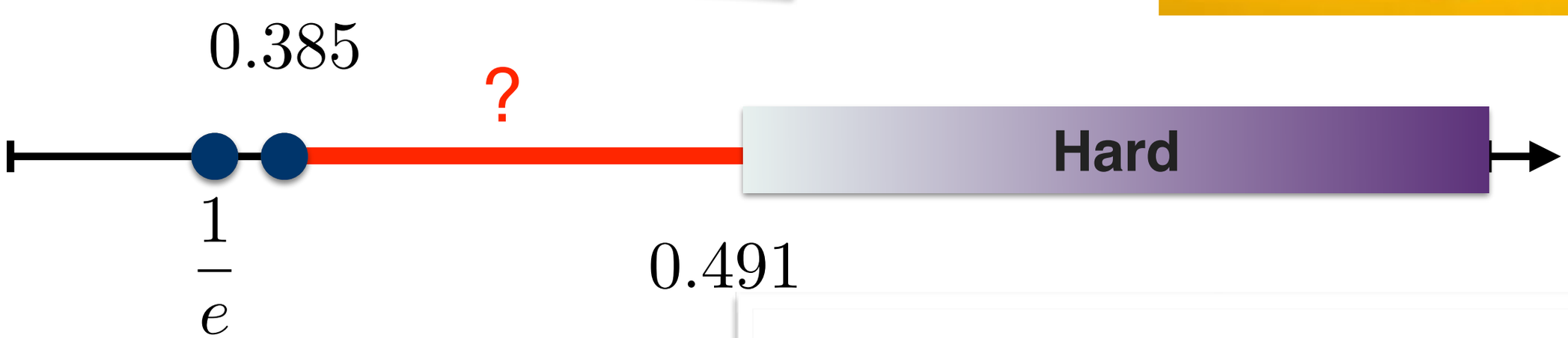
$$S^* = \arg\max_{|S| \le k} f(S)$$



Constrained Submodular Maximization via a Non-symmetric Technique

Niv Buchbinder* Moran Feldman†

2016



Submodular Maximization by Simulated Annealing

Shayan Oveis Gharan* Jan Vondrák†

2010

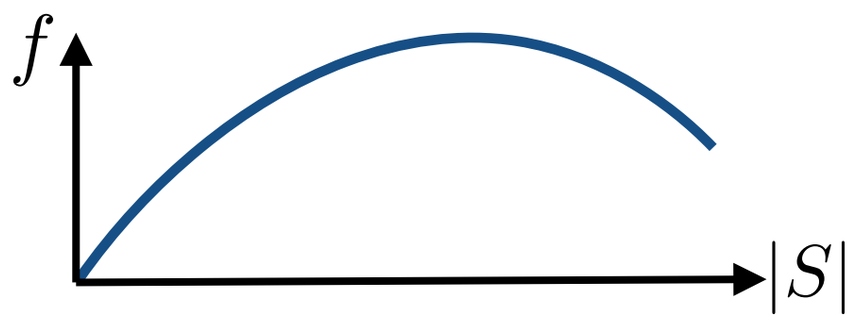
Constrained Non-Monotone Submodular Maximization

- Consider the following problem where f is submodular:

$$S^* = \arg\max_{S \in \mathcal{I}} f(S)$$

constraints

- When the utility functions f is non-monotone submodular



- and constraints imposed by the application

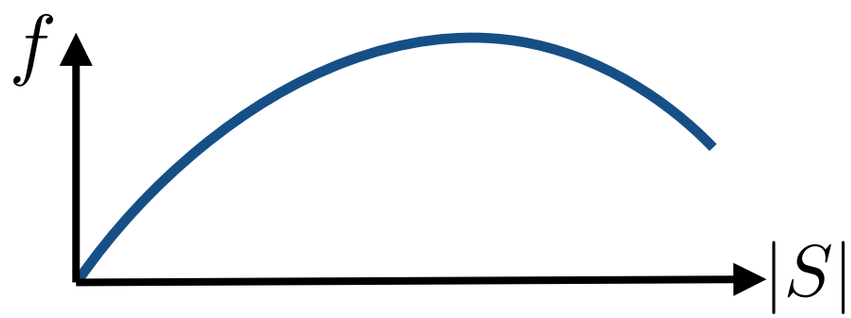
Constrained Non-Monotone Submodular Maximization

- Consider the following problem where f is submodular:

$$S^* = \arg\max_{S \in \mathcal{I}} f(S)$$

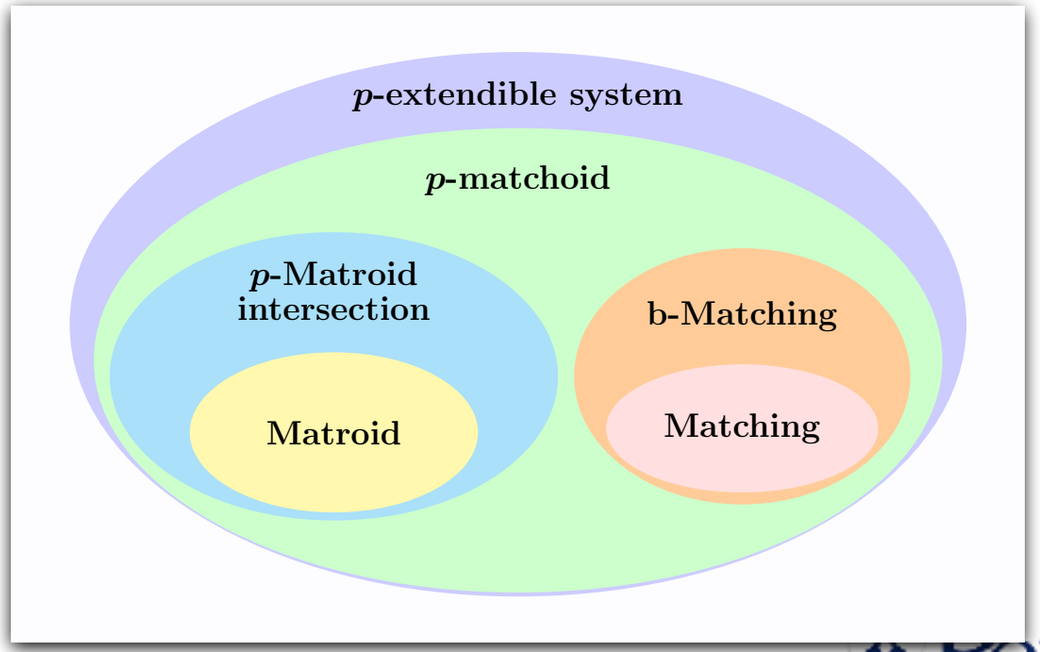
constraints

- When the utility functions f is non-monotone submodular



- and constraints imposed by the application

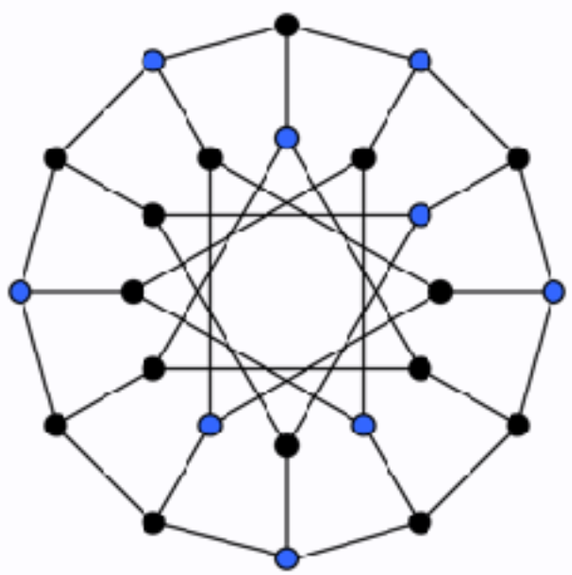
p-extendible system: adding an element e to a feasible set B requires the removal of at most p elements in order to keep the resulting set feasible.



Examples

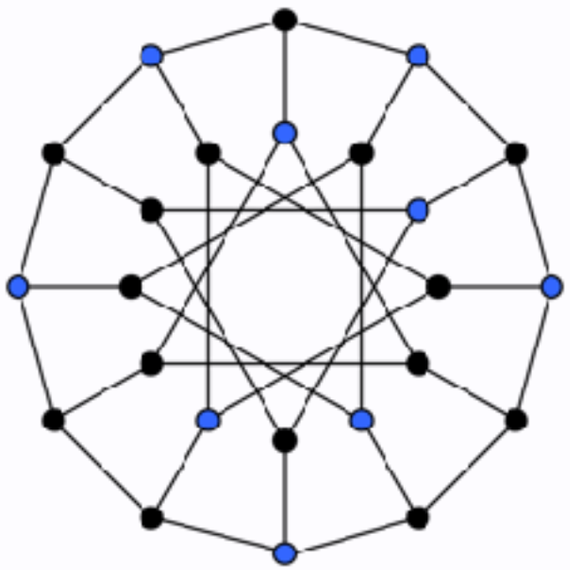
Examples

Independent Set $p = d_{\max}$



Examples

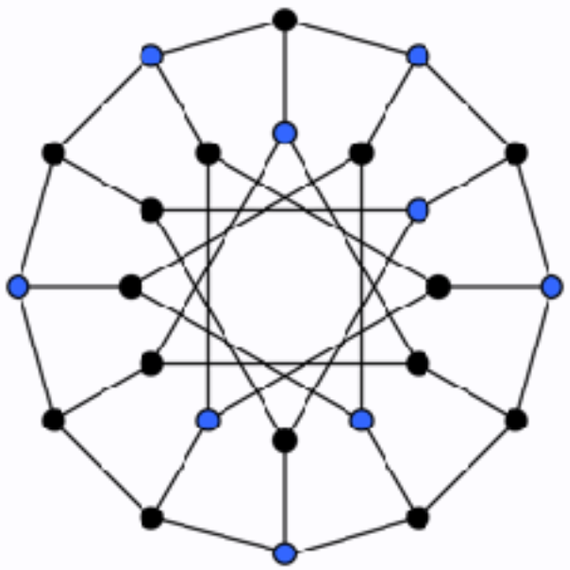
Independent Set $p = d_{\max}$



Matroid $p = 1$

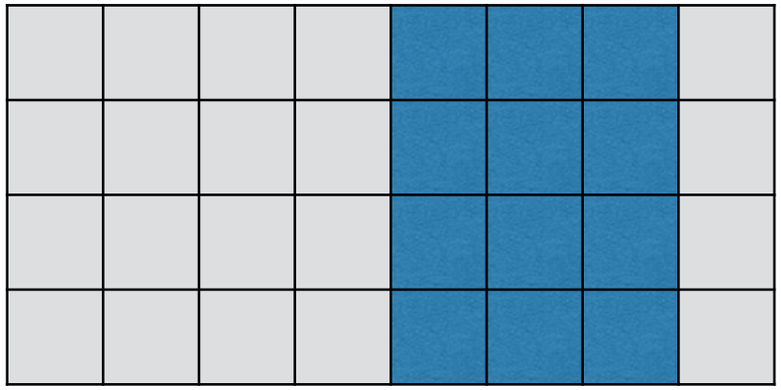
Examples

Independent Set $p = d_{\max}$



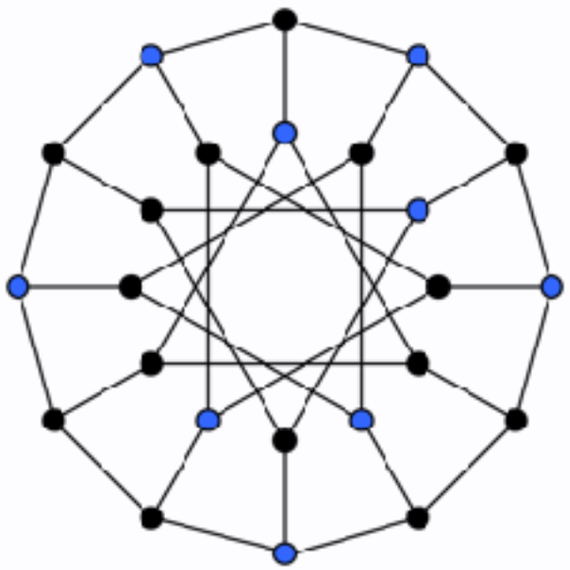
Matroid $p = 1$

$\forall A \subset B \text{ if } B \in \mathcal{I} \Rightarrow A \in \mathcal{I}$



Examples

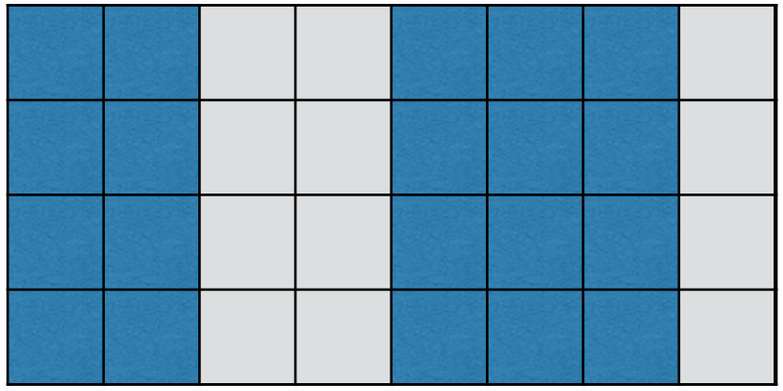
Independent Set $p = d_{\max}$



Matroid $p = 1$

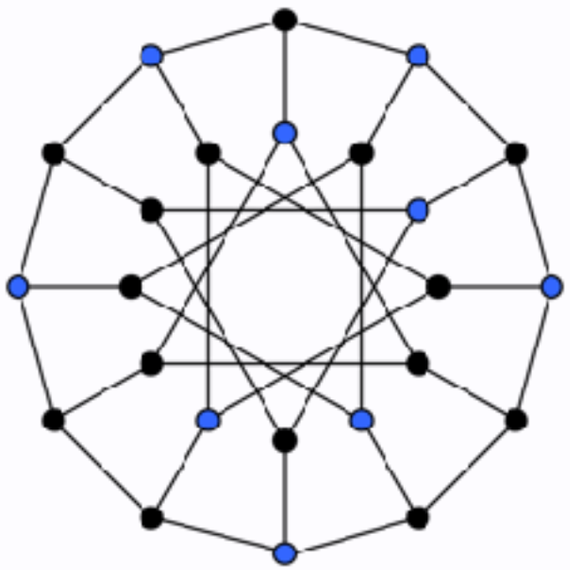
$\forall A \subset B$ if $B \in \mathcal{I} \Rightarrow A \in \mathcal{I}$

$A, B \in \mathcal{I}$ & $|A| < |B| \Rightarrow \exists v \in B$ s.t. $A \cup \{v\} \in \mathcal{I}$



Examples

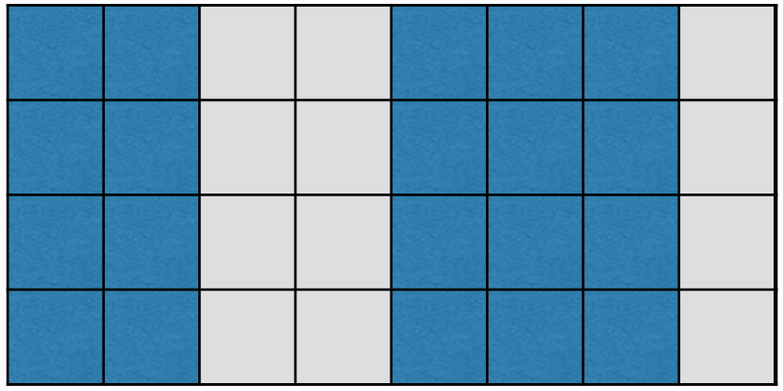
Independent Set $p = d_{\max}$



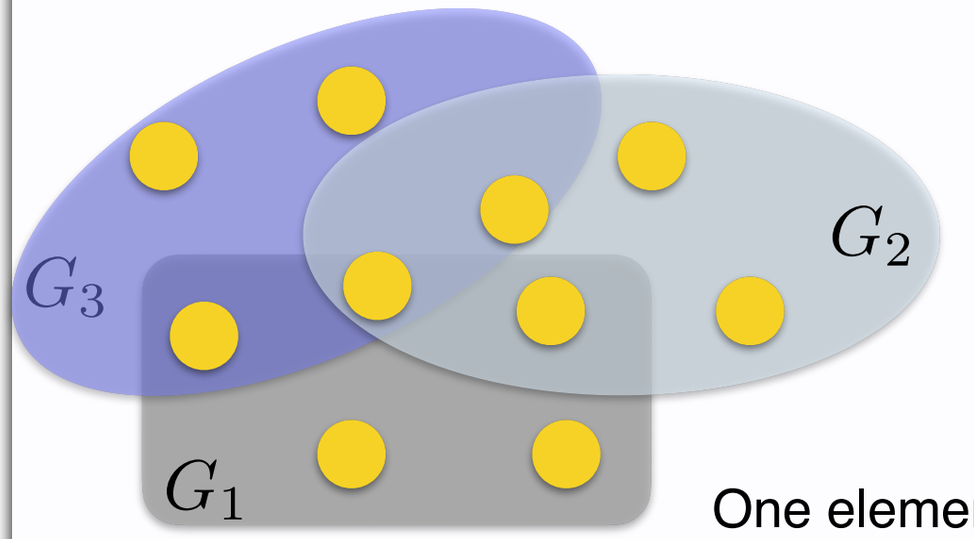
Matroid $p = 1$

$\forall A \subset B$ if $B \in \mathcal{I} \Rightarrow A \in \mathcal{I}$

$A, B \in \mathcal{I}$ & $|A| < |B| \Rightarrow \exists v \in B$ s.t. $A \cup \{v\} \in \mathcal{I}$



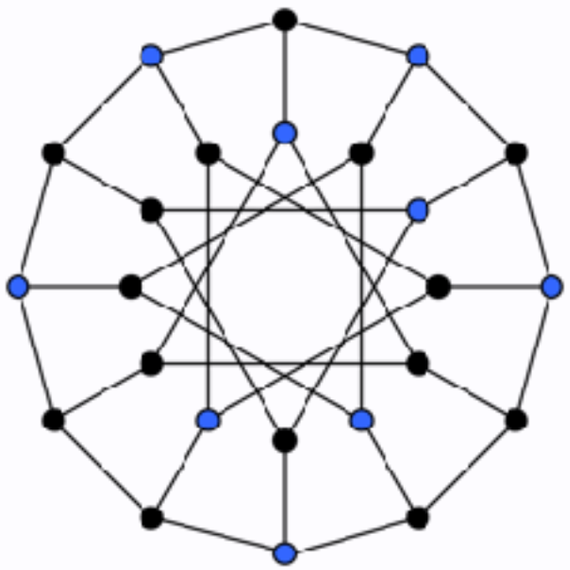
Intersection of k matroids $p = k$



One element from each group

Examples

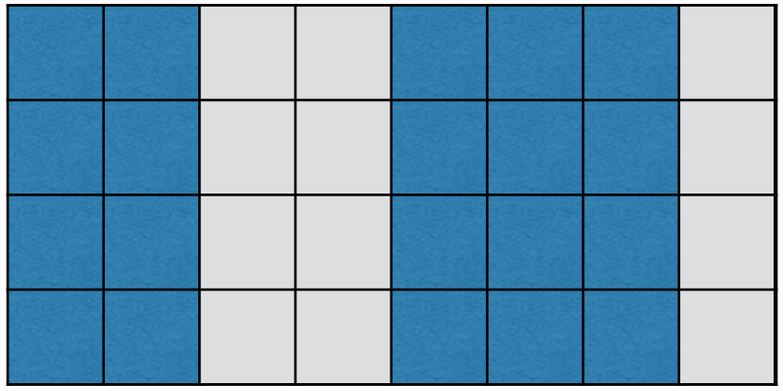
Independent Set $p = d_{\max}$



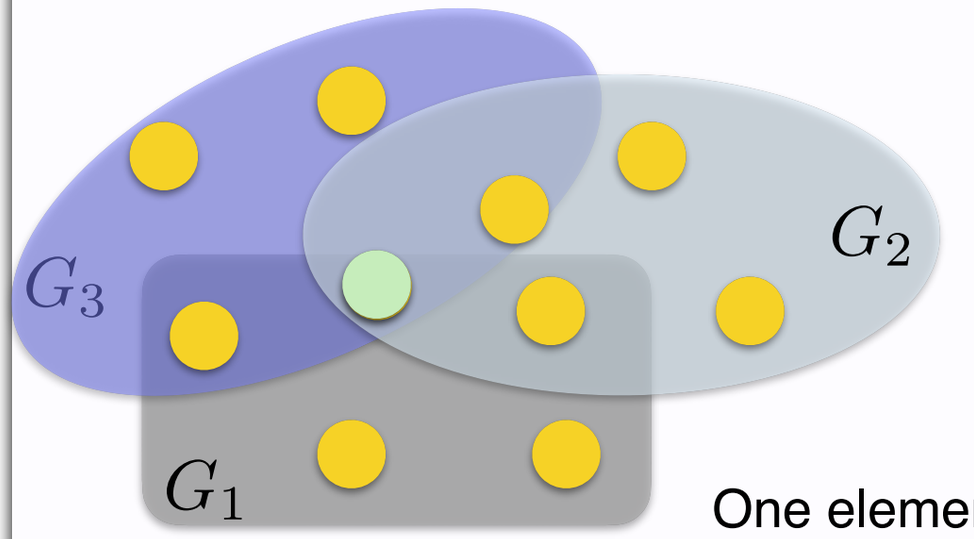
Matroid $p = 1$

$\forall A \subset B$ if $B \in \mathcal{I} \Rightarrow A \in \mathcal{I}$

$A, B \in \mathcal{I}$ & $|A| < |B| \Rightarrow \exists v \in B$ s.t. $A \cup \{v\} \in \mathcal{I}$



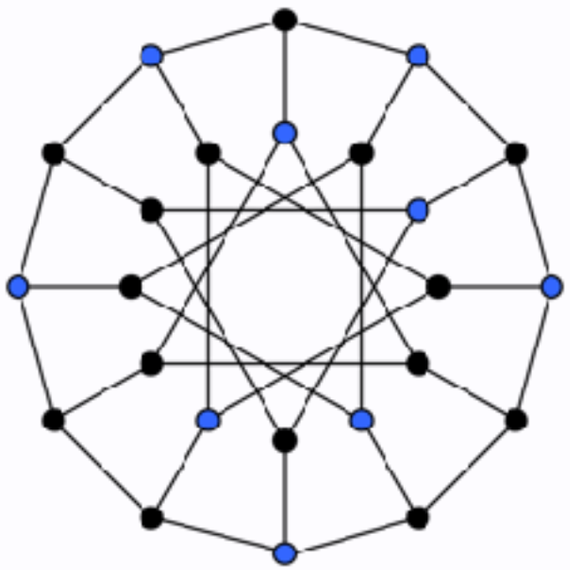
Intersection of k matroids $p = k$



One element from each group

Examples

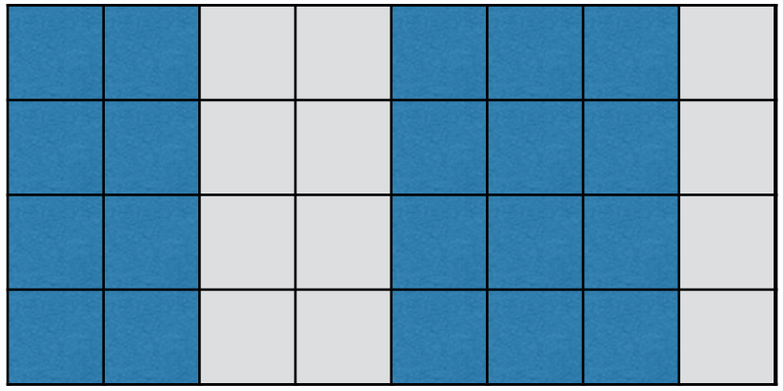
Independent Set $p = d_{\max}$



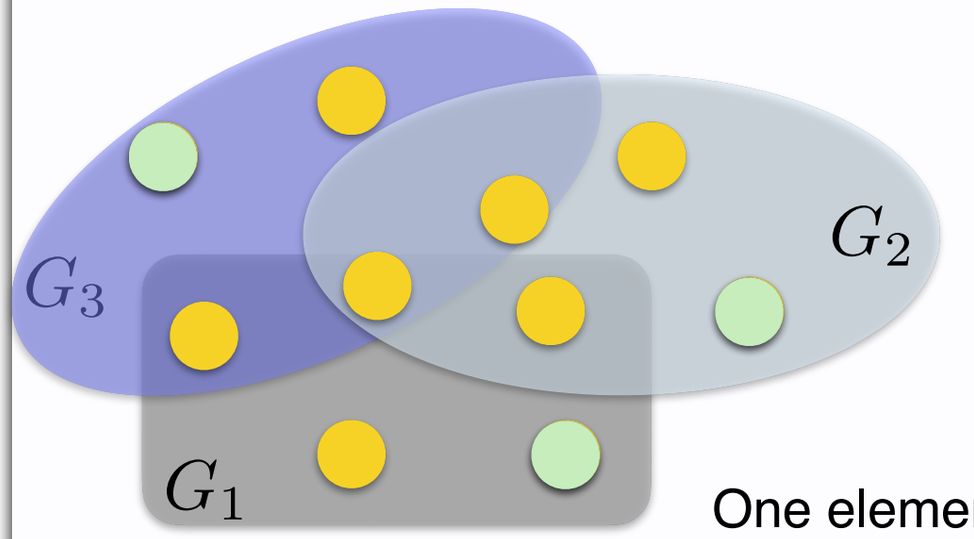
Matroid $p = 1$

$\forall A \subset B$ if $B \in \mathcal{I} \Rightarrow A \in \mathcal{I}$

$A, B \in \mathcal{I}$ & $|A| < |B| \Rightarrow \exists v \in B$ s.t. $A \cup \{v\} \in \mathcal{I}$



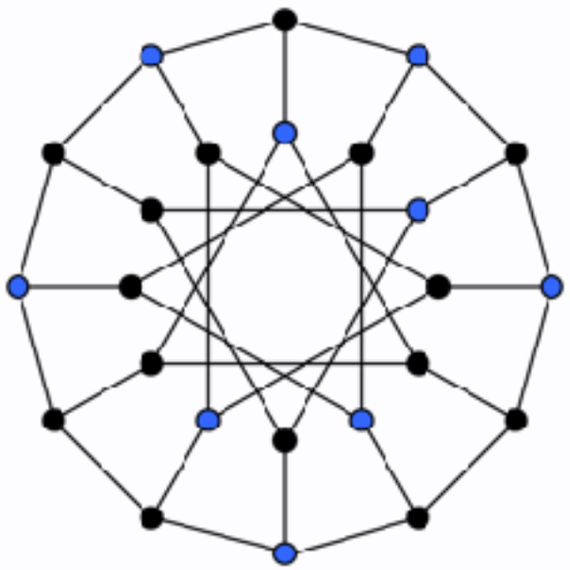
Intersection of k matroids $p = k$



One element from each group

Examples

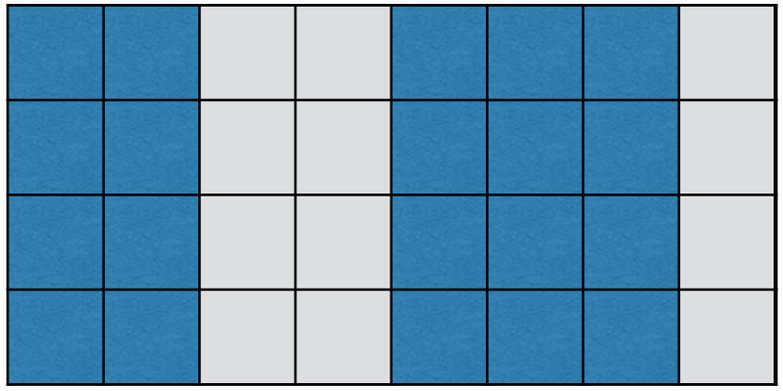
Independent Set $p = d_{\max}$



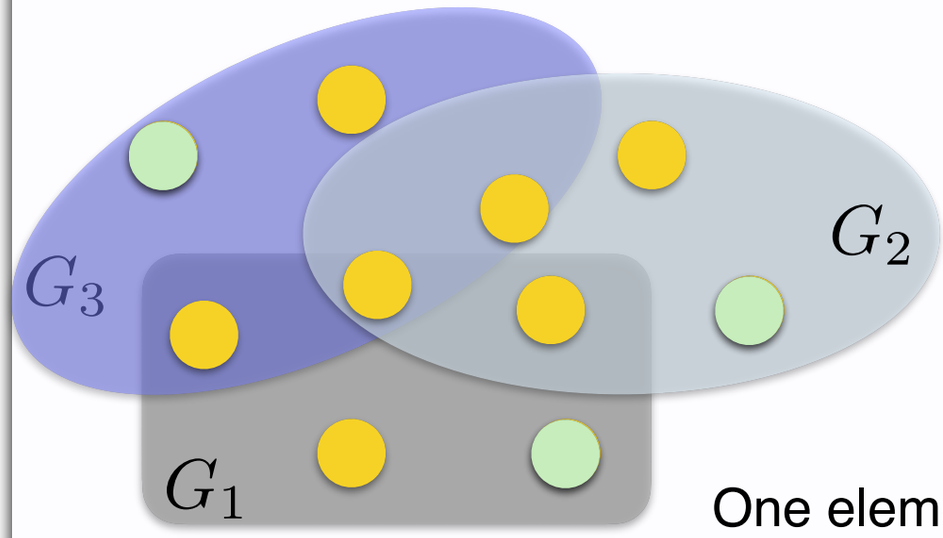
Matroid $p = 1$

$\forall A \subset B$ if $B \in \mathcal{I} \Rightarrow A \in \mathcal{I}$

$A, B \in \mathcal{I}$ & $|A| < |B| \Rightarrow \exists v \in B$ s.t. $A \cup \{v\} \in \mathcal{I}$



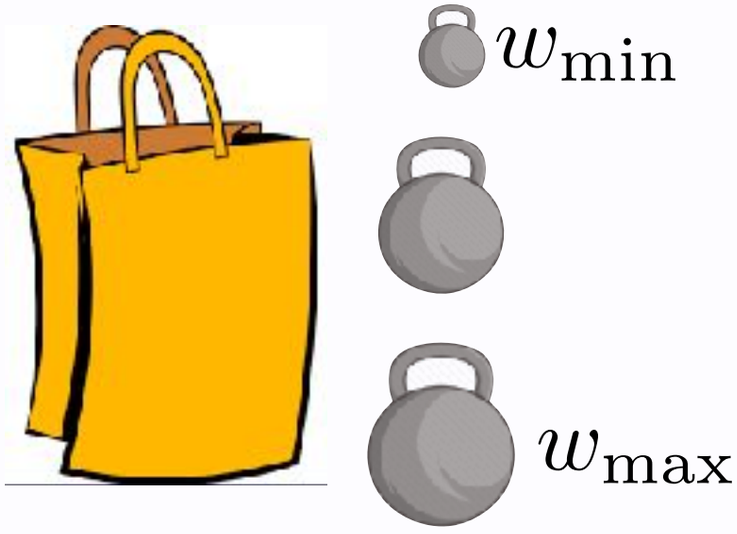
Intersection of k matroids $p = k$



One element from each group

Knapsack

$$p = \left\lceil \frac{w_{\max}}{w_{\min}} \right\rceil$$

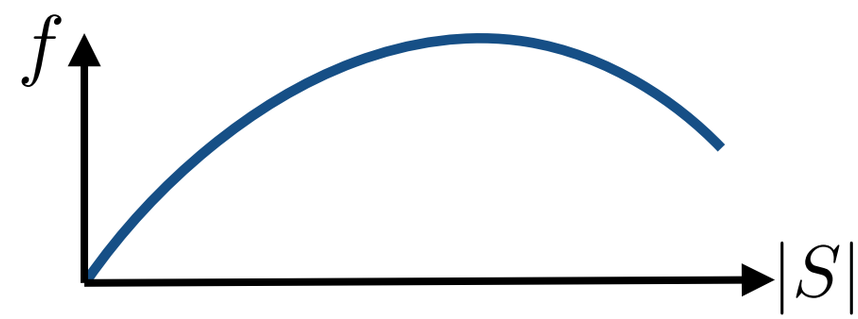


Constrained Non-Monotone Submodular Maximization

- Consider the following problem where f is submodular:

$$S^* = \arg\max_{S \in \mathcal{I}} f(S)$$

constraints

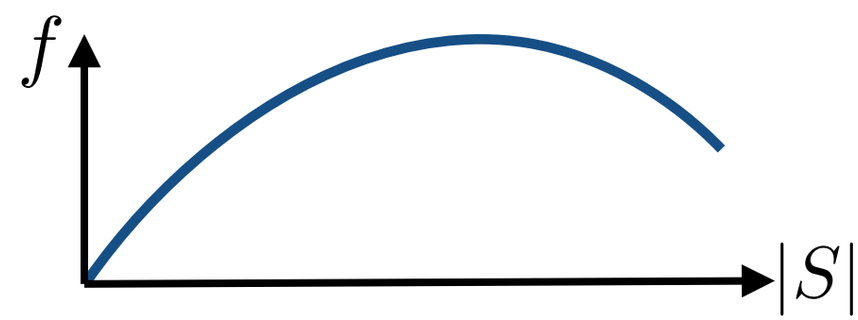


Constrained Non-Monotone Submodular Maximization

- Consider the following problem where f is submodular:

$$S^* = \arg\max_{S \in \mathcal{I}} f(S)$$

constraints



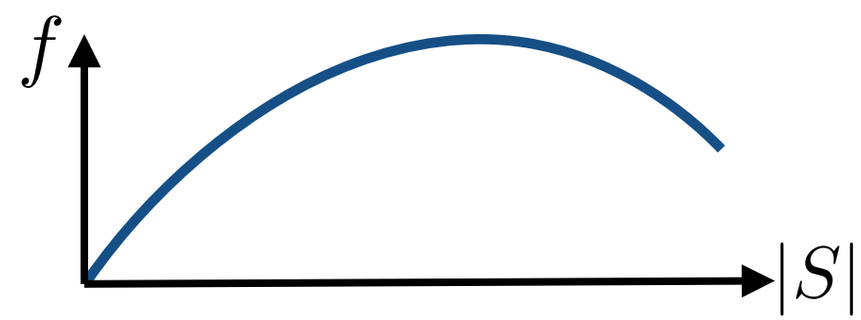
- Two approaches:
  1. Repeat the greedy **multiple** times and update the ground set each time
  2. Subsample the ground set and run the greedy **once**.

Constrained Non-Monotone Submodular Maximization

- Consider the following problem where f is submodular:

$$S^* = \arg\max_{S \in \mathcal{I}} f(S)$$

constraints



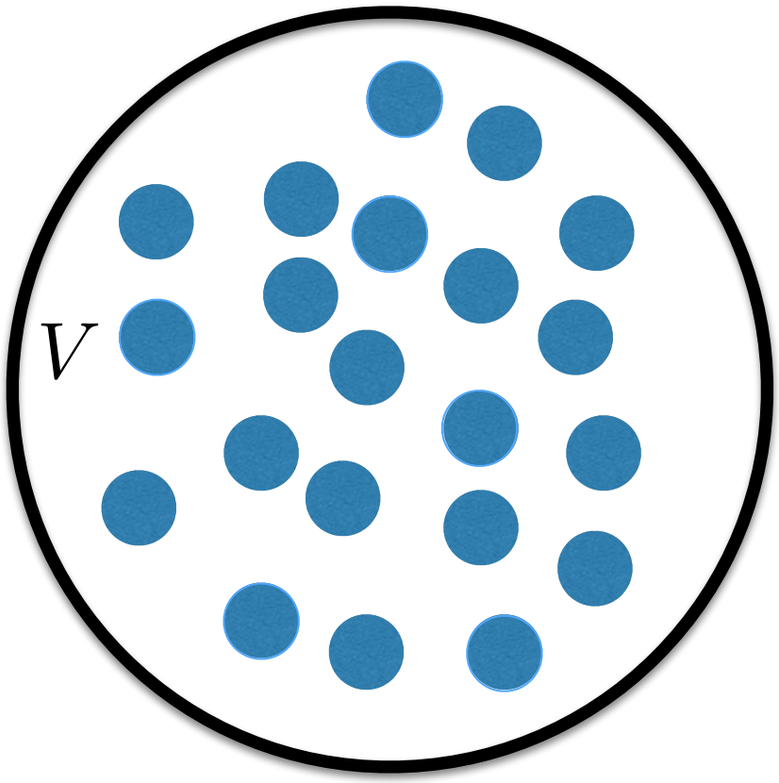
- Two approaches:
  1. Repeat the greedy **multiple** times and update the ground set each time
  2. Subsample the ground set and run the greedy **once**.

Greedy Algorithm

$S \rightarrow \emptyset$.
while $\exists e \in V$ st. $S \cup \{e\} \in \mathcal{I}$, $f(S \cup \{e\}) > f(S)$ **do**
  $S \leftarrow S \cup \arg\max_{\substack{e \in V \\ S \cup \{e\} \in \mathcal{I}}} f(S \cup \{e\})$
end while
return S

Repeated Greedy

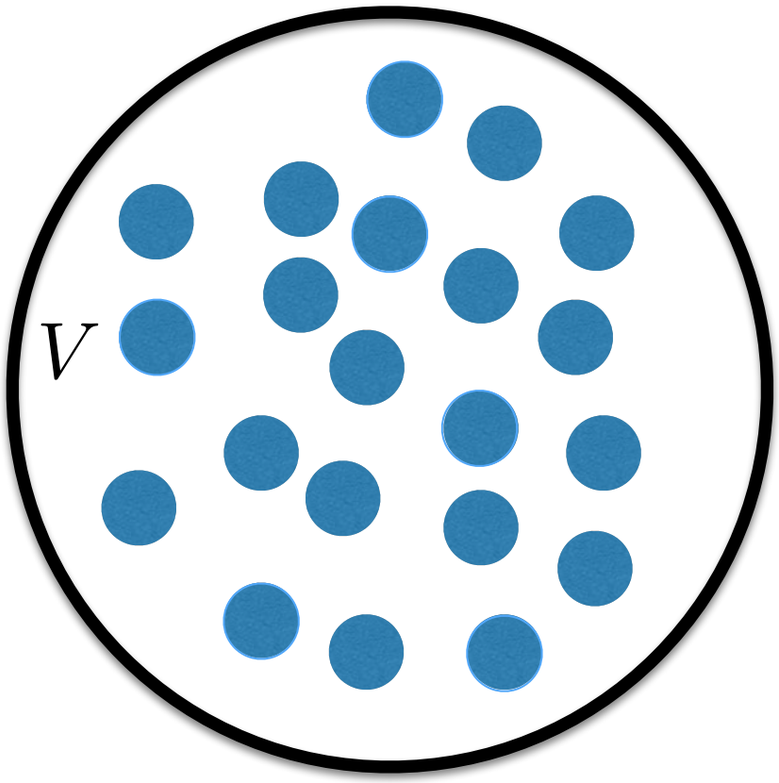
- Ground set V and p-extendible system $\mathcal{I}$



Repeated Greedy

- Ground set V and p-extendible system $\mathcal{I}$

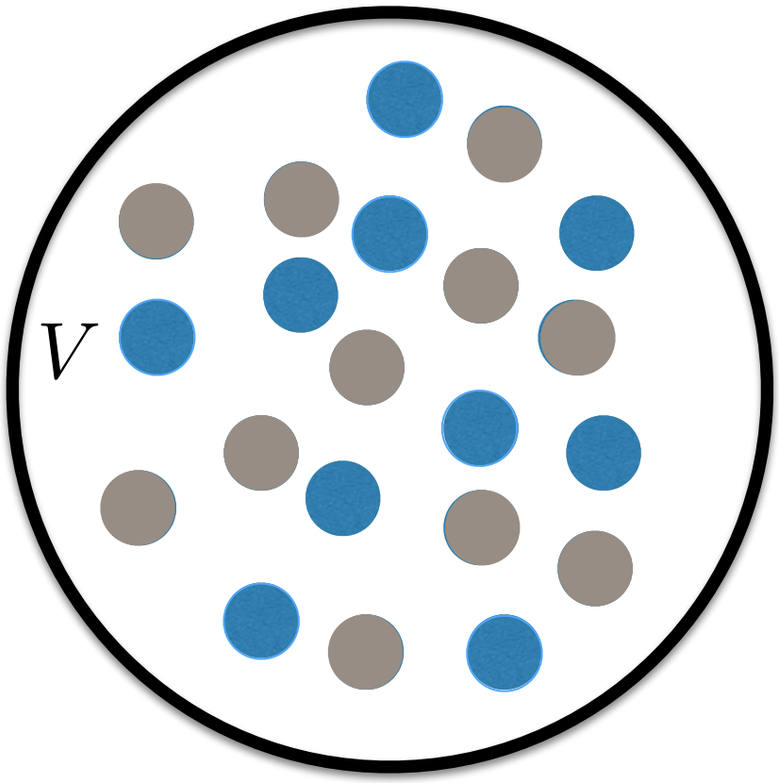
- Construct S_i by running greedy on V
- Construct $S_i^{'}$ by running double greedy on S_i
- Remove S_i from the ground set V



Repeated Greedy

- Ground set V and p-extendible system $\mathcal{I}$

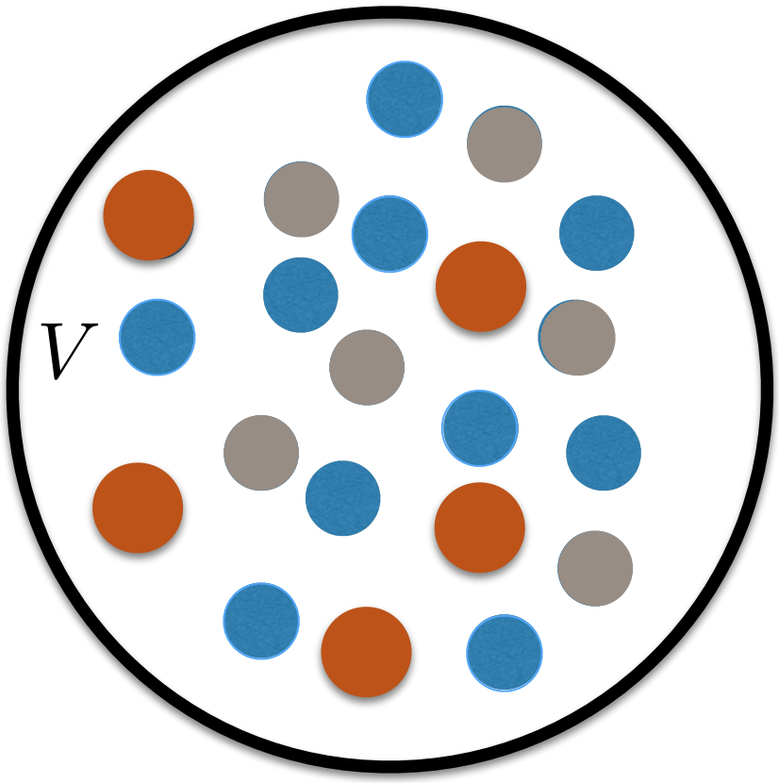
- Construct S_i by running greedy on V
- Construct $S_i^{'}$ by running double greedy on S_i
- Remove S_i from the ground set V



Repeated Greedy

- Ground set V and p-extendible system $\mathcal{I}$

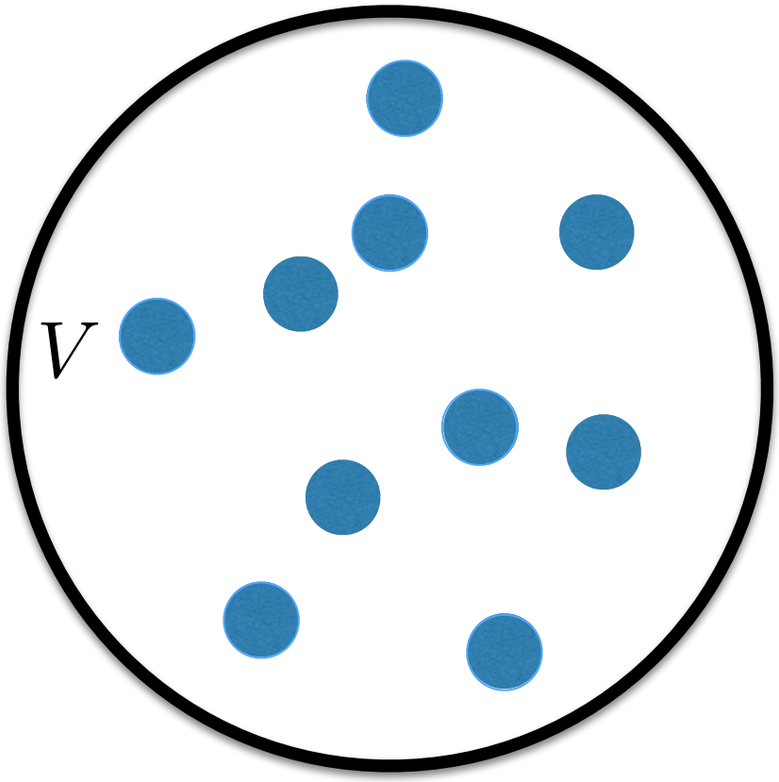
- Construct S_i by running greedy on V
- Construct $S_i^{'}$ by running double greedy on S_i
- Remove S_i from the ground set V

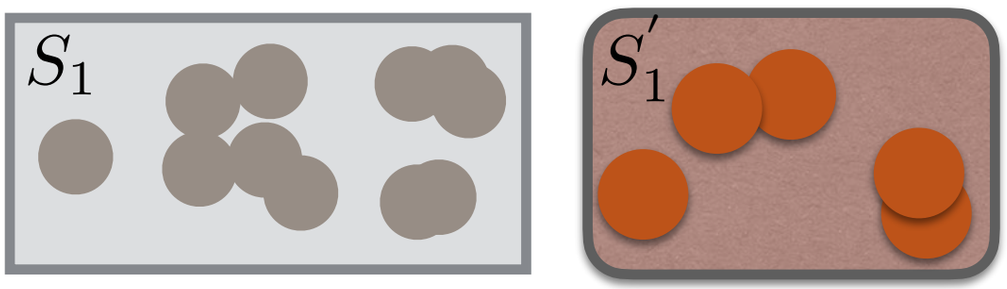


Repeated Greedy

- Ground set V and p-extendible system $\mathcal{I}$

- Construct S_i by running greedy on V
- Construct $S_i^{'}$ by running double greedy on S_i
- Remove S_i from the ground set V

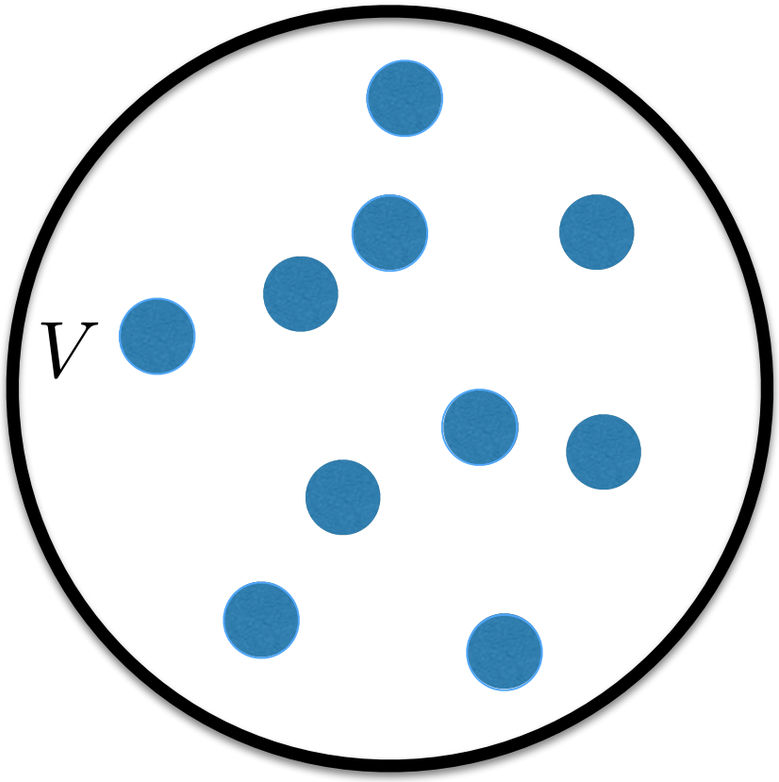




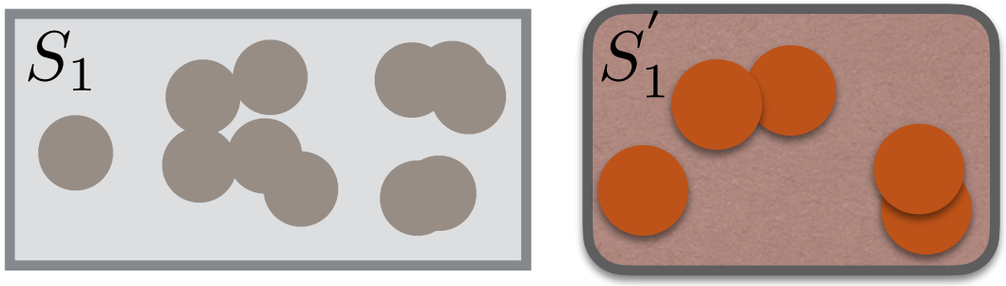
Repeated Greedy

- Ground set V and p-extendible system $\mathcal{I}$

- Construct S_i by running greedy on V
- Construct $S_i^{'}$ by running double greedy on S_i
- Remove S_i from the ground set V



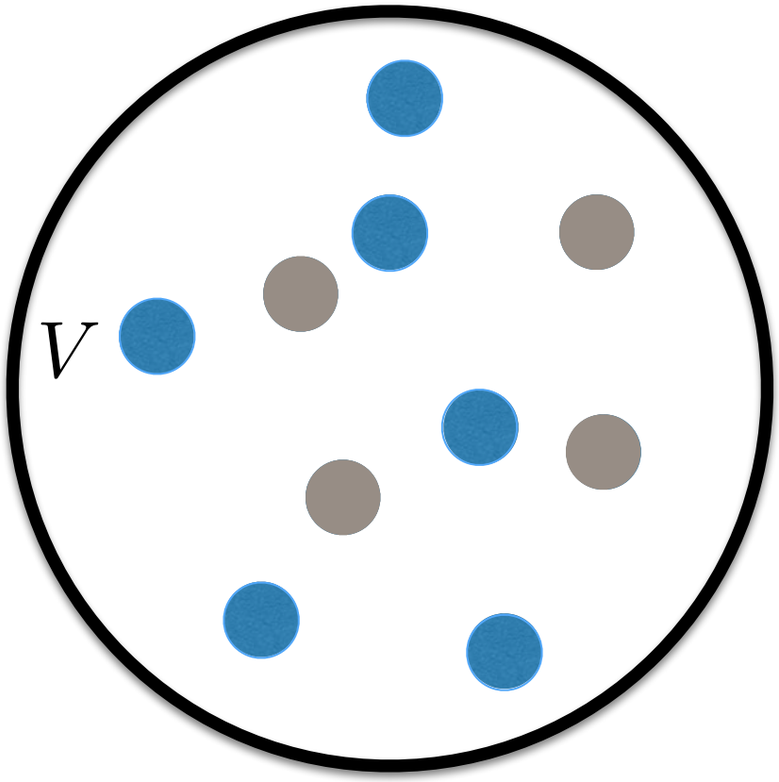
- Repeat the block L times



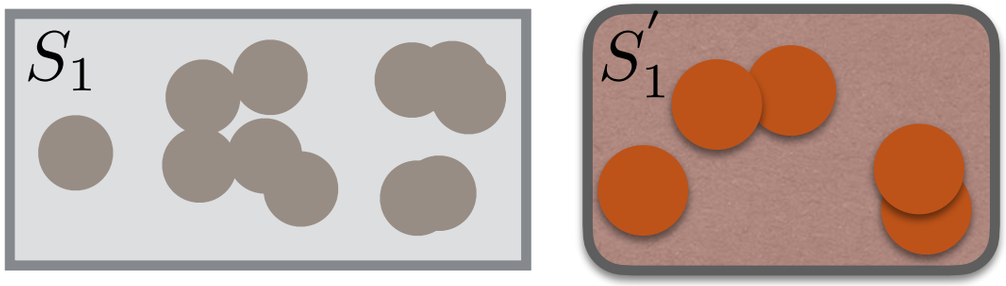
Repeated Greedy

- Ground set V and p-extendible system $\mathcal{I}$

- Construct S_i by running greedy on V
- Construct $S_i^{'}$ by running double greedy on S_i
- Remove S_i from the ground set V



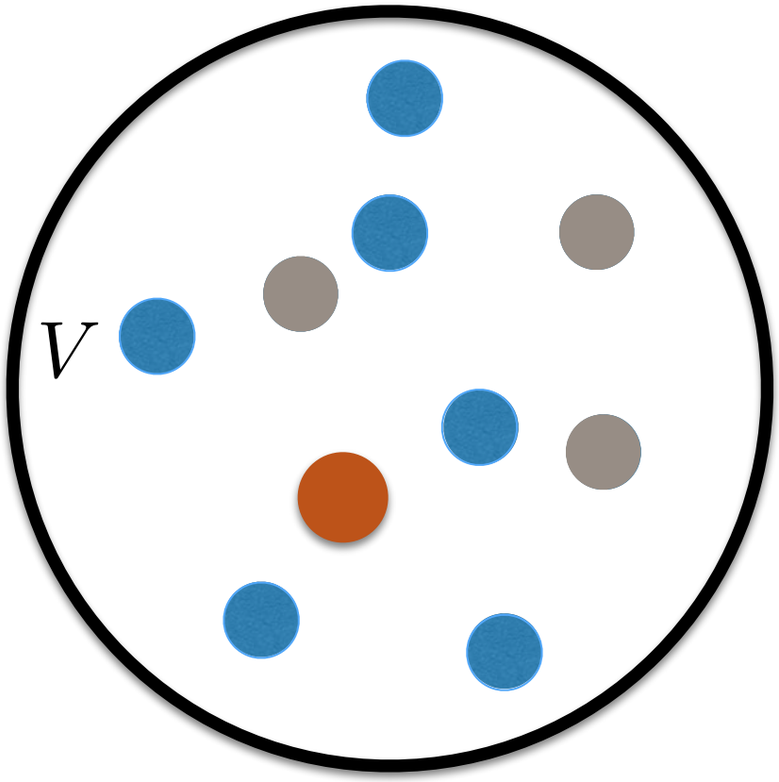
- Repeat the block L times



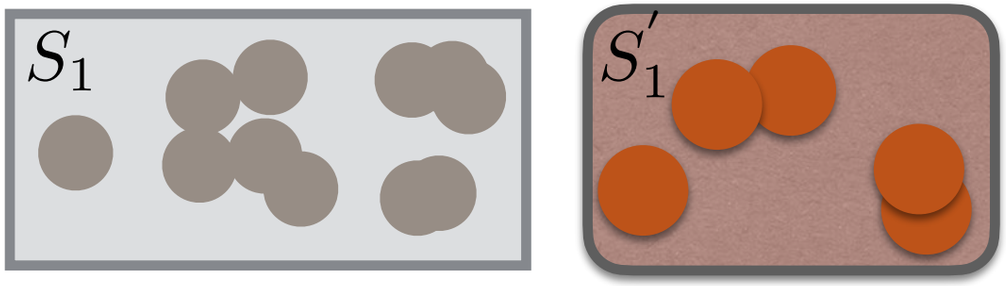
Repeated Greedy

- Ground set V and p-extendible system $\mathcal{I}$

- Construct S_i by running greedy on V
- Construct $S_i^{'}$ by running double greedy on S_i
- Remove S_i from the ground set V



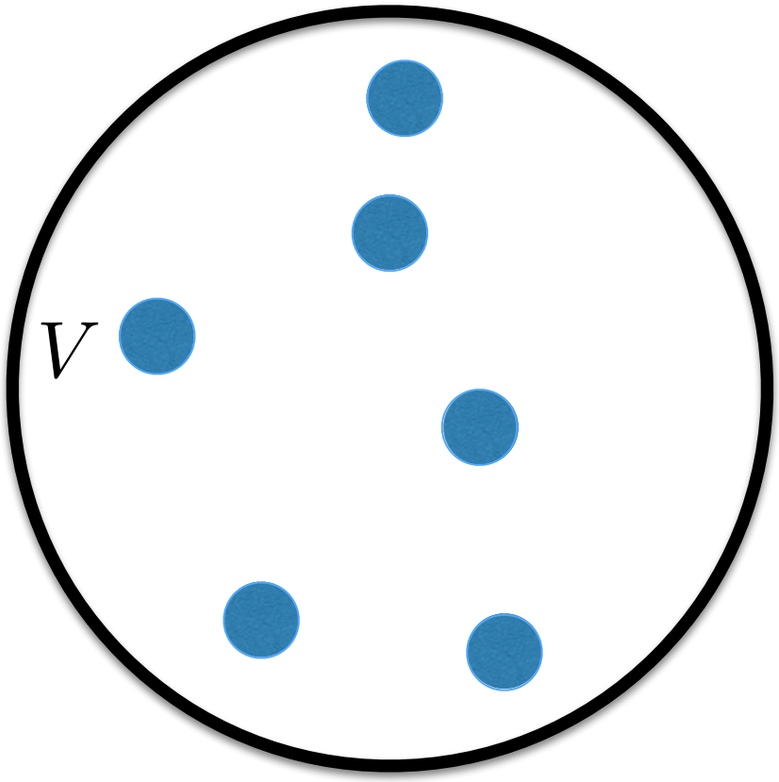
- Repeat the block L times



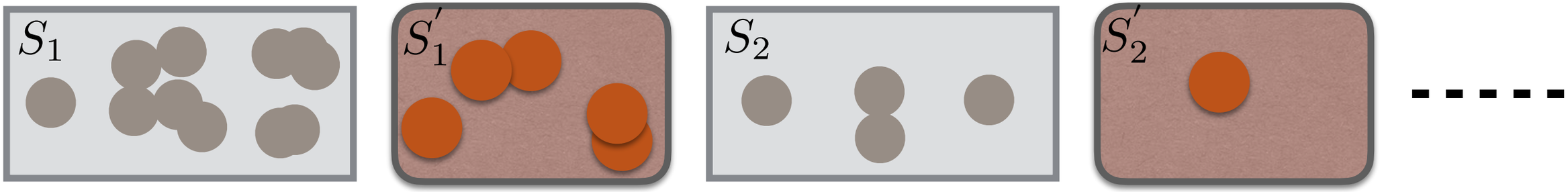
Repeated Greedy

- Ground set V and p-extendible system $\mathcal{I}$

- Construct S_i by running greedy on V
- Construct $S_i^{'}$ by running double greedy on S_i
- Remove S_i from the ground set V



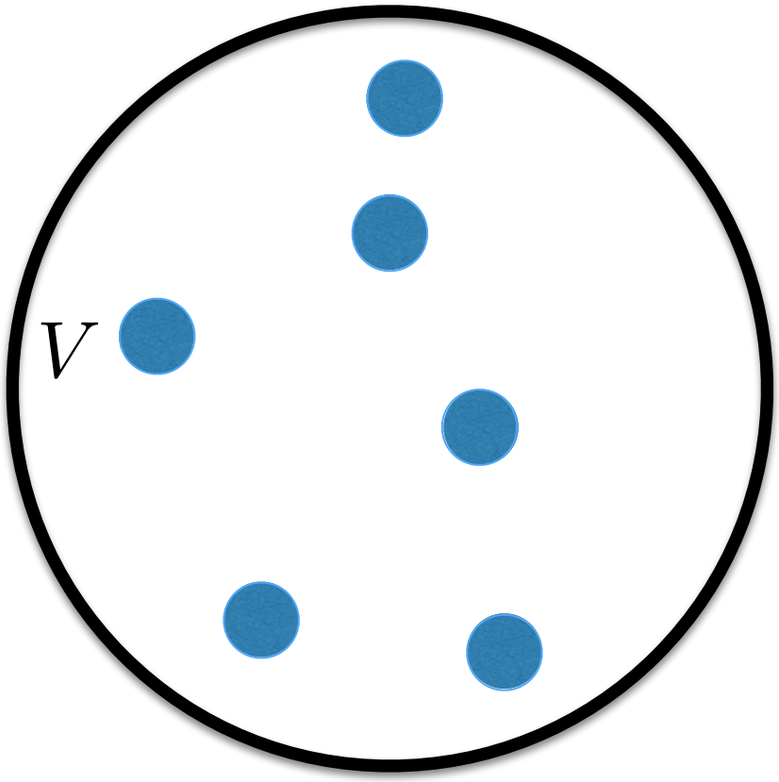
- Repeat the block L times



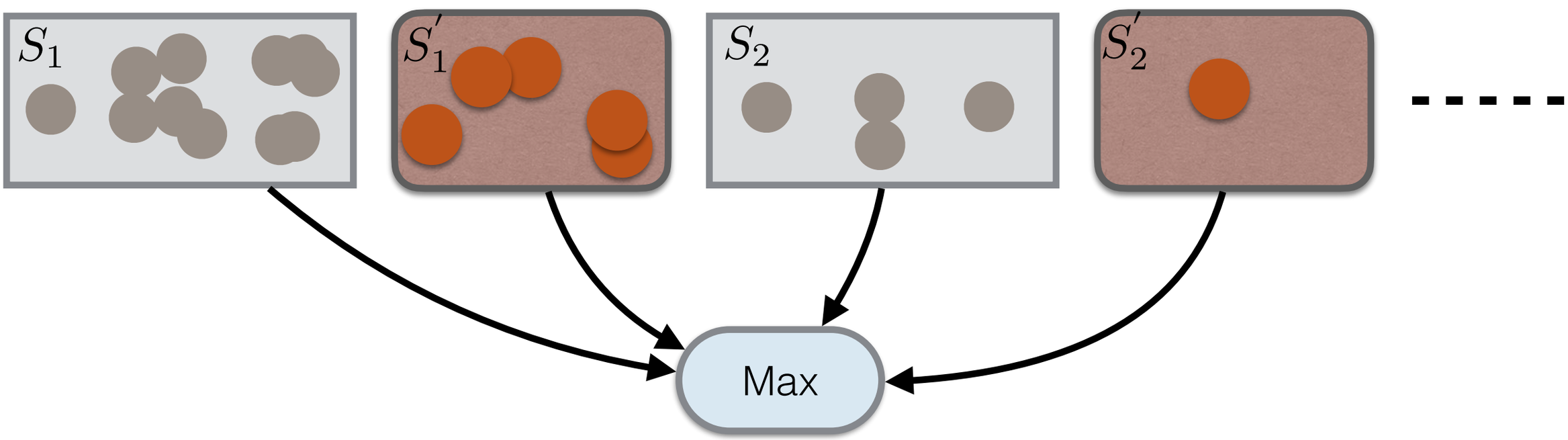
Repeated Greedy

- Ground set V and p-extendible system $\mathcal{I}$

- Construct S_i by running greedy on V
- Construct $S_i^{'}$ by running double greedy on S_i
- Remove S_i from the ground set V



- Repeat the block L times

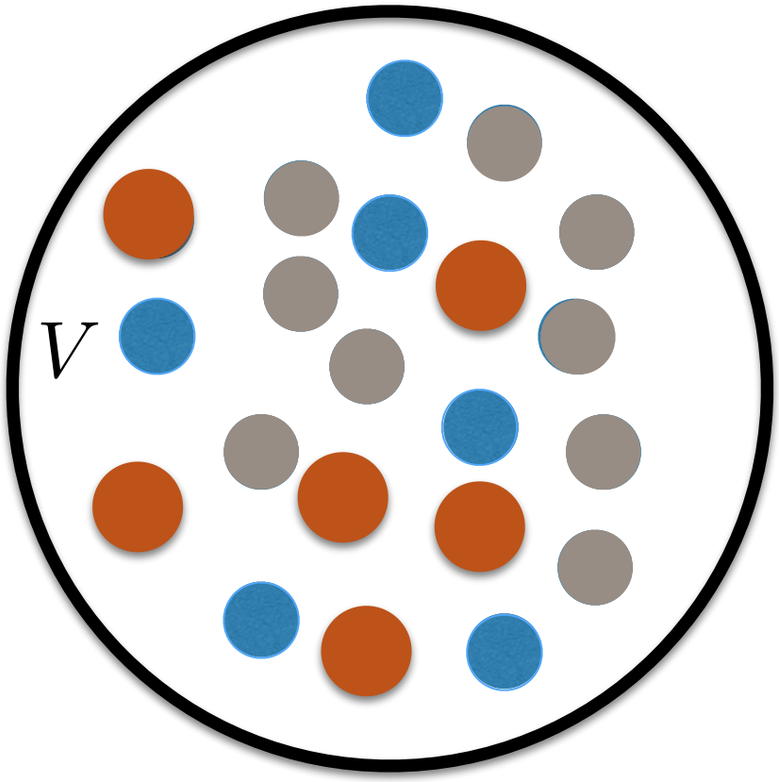


"Constrained Non-Monotone Submodular Maximization: Offline and Secretary Algorithms", Gupta et al, 2010

Repeated Greedy

- Ground set V and p-extendible system $\mathcal{I}$

- Construct S_i by running greedy on V
- Construct $S_i^{'}$ by running double greedy on S_i
- Remove S_i from the ground set V



- Repeat the block L times

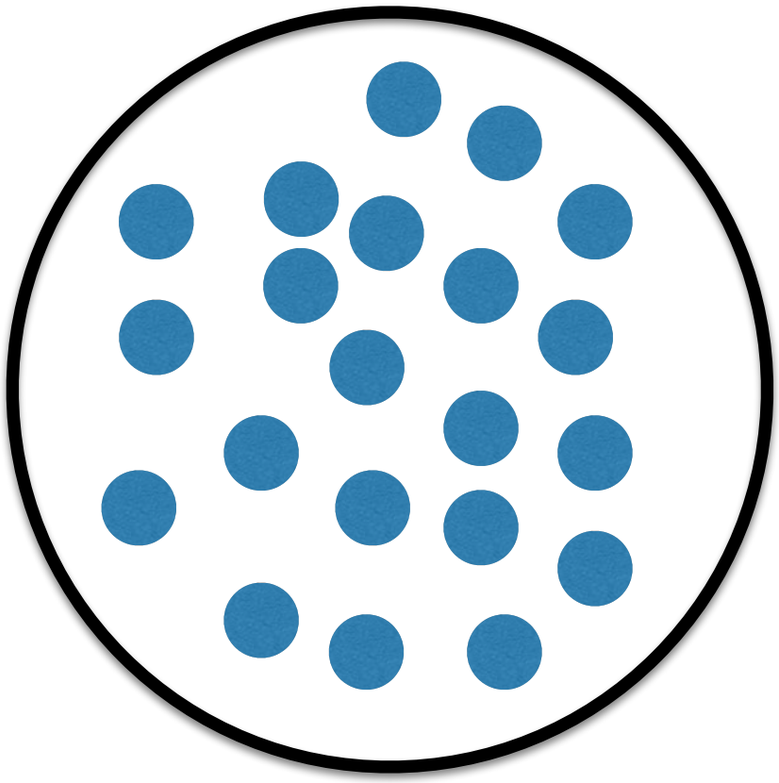
[Gupta, Roth, Schoenebeck, Talwar]

Let $L = p + 1$. Then, Repeated Greedy achieves

$$f(S_{\text{Repeated-Greedy}}) \geq \frac{1}{3(p + 2 + 1/p)} \text{OPT}$$

"Constrained Non-Monotone Submodular Maximization: Offline and Secretary Algorithms", 2010

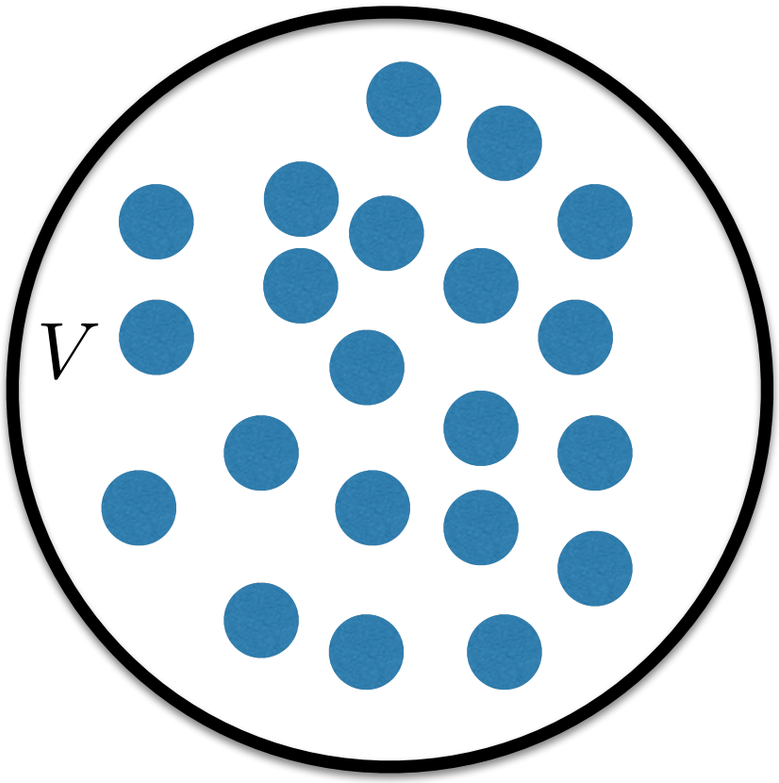
Sample Greedy: Constrained



Sub-Sampled set

Sample Greedy: Constrained

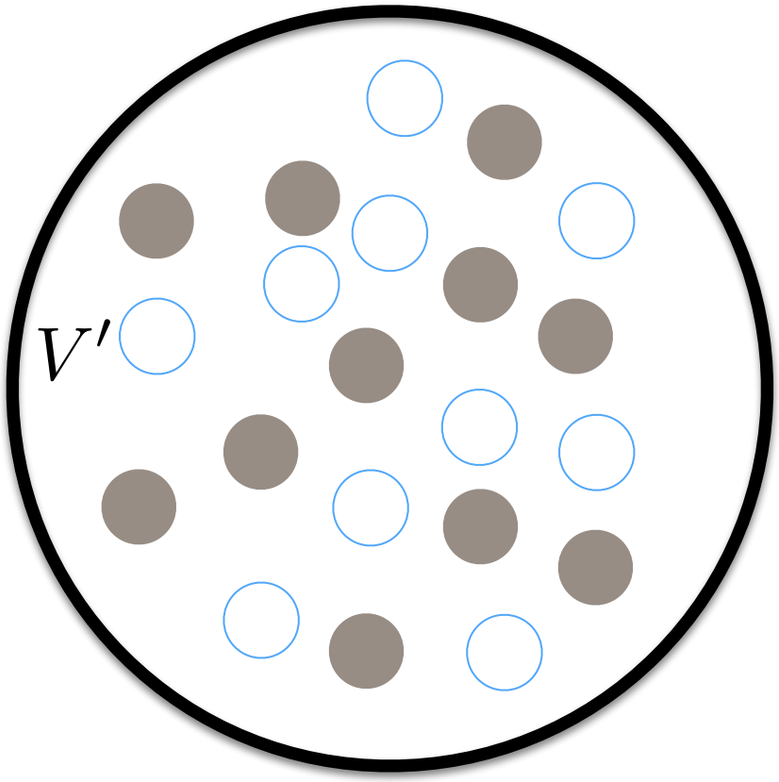
- Ground set V and p-extendible system $\mathcal{I}$



Sub-Sampled set

Sample Greedy: Constrained

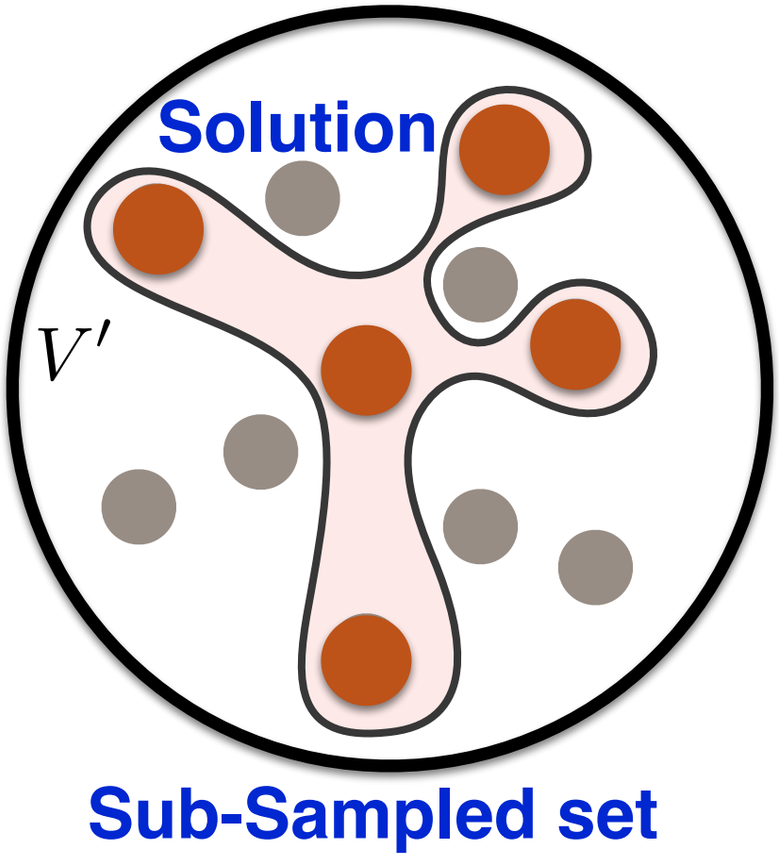
- Ground set V and p-extendible system $\mathcal{I}$
- Construct V' by randomly sampling each point with probability $1/(p+1)$



Sub-Sampled set

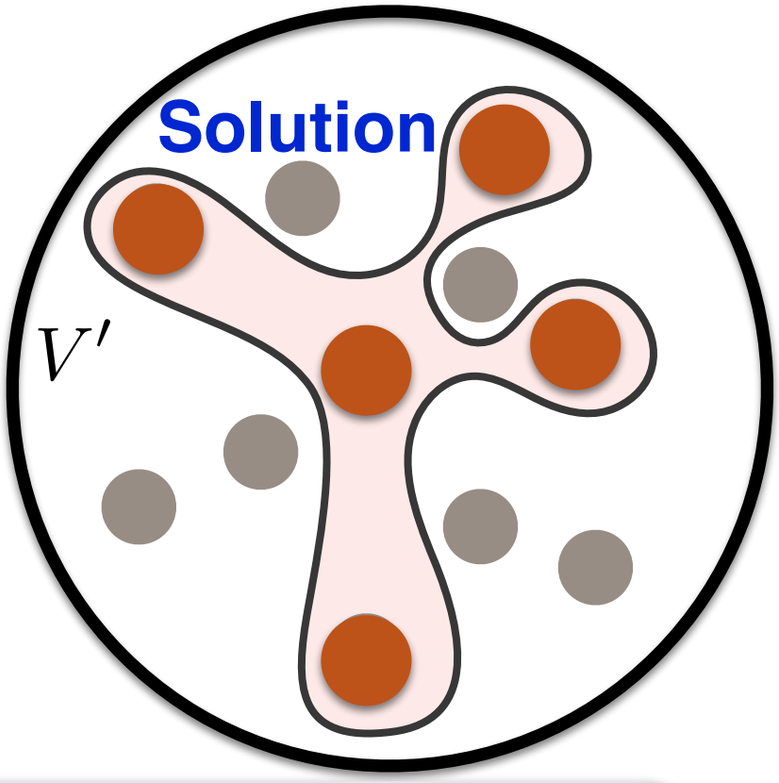
Sample Greedy: Constrained

- Ground set V and p-extendible system $\mathcal{I}$
- Construct V' by randomly sampling each point with probability $1/(p+1)$
- Produce $S_{\mathrm{SampleGreedy}}$ by running Greedy algorithm on the sub-sampled set V'



Sample Greedy: Constrained

- Ground set V and p-extendible system $\mathcal{I}$
- Construct V' by randomly sampling each point with probability $1/(p+1)$
- Produce $S_{\text{SampleGreedy}}$ by running Greedy algorithm on the sub-sampled set V'



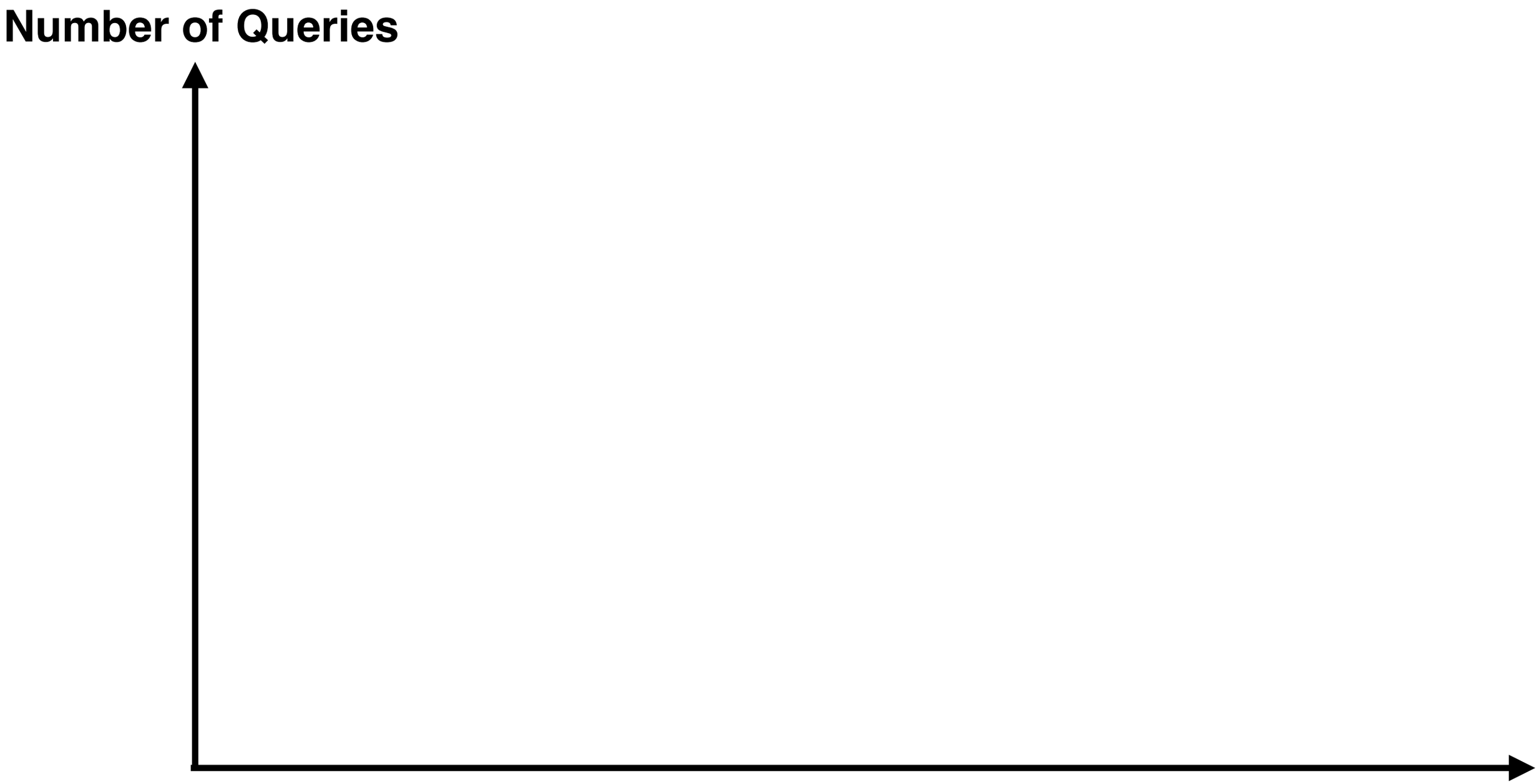
[Feldman, Harshaw, Karbasi]

In expectation, SampleGreedy achieves

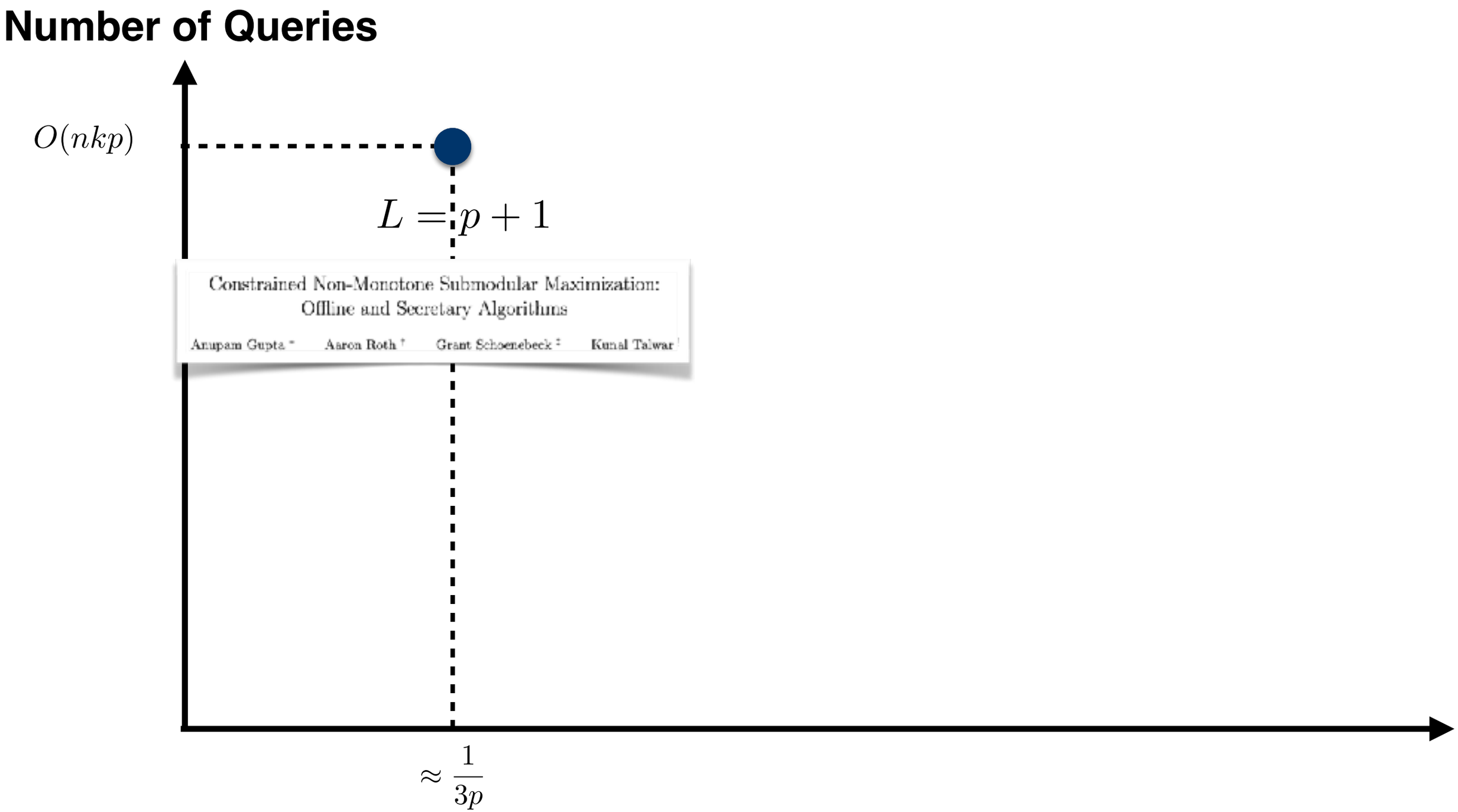
$$\mathbb{E}[f\,(S_{\text{SampleGreedy}})] \geq \frac{1}{p+2+o(1)} \cdot \text{OPT}$$

"Greed Is Good: Near-Optimal Submodular Maximization via Greedy Optimization".2017

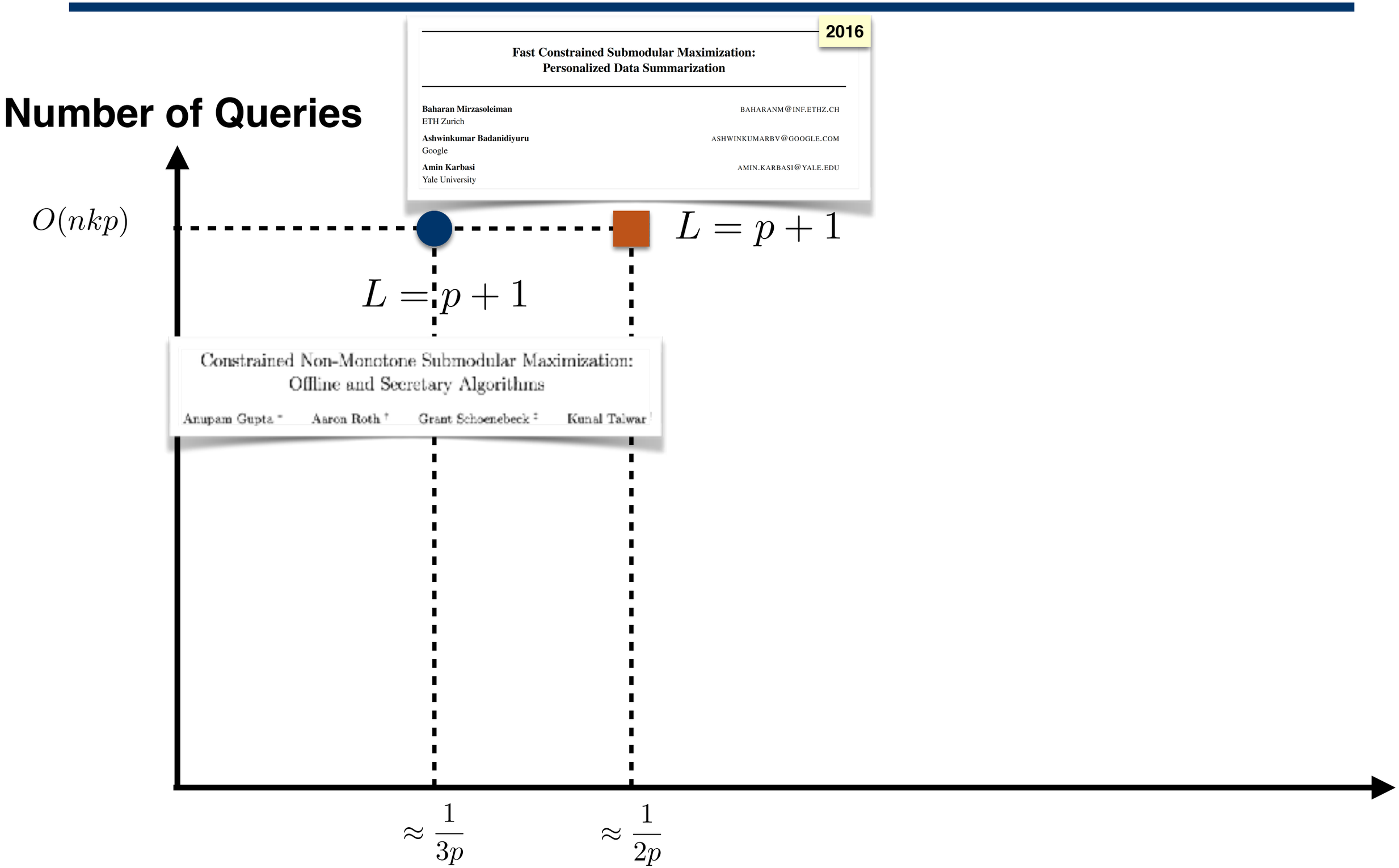
Constrained Submodular Maximization



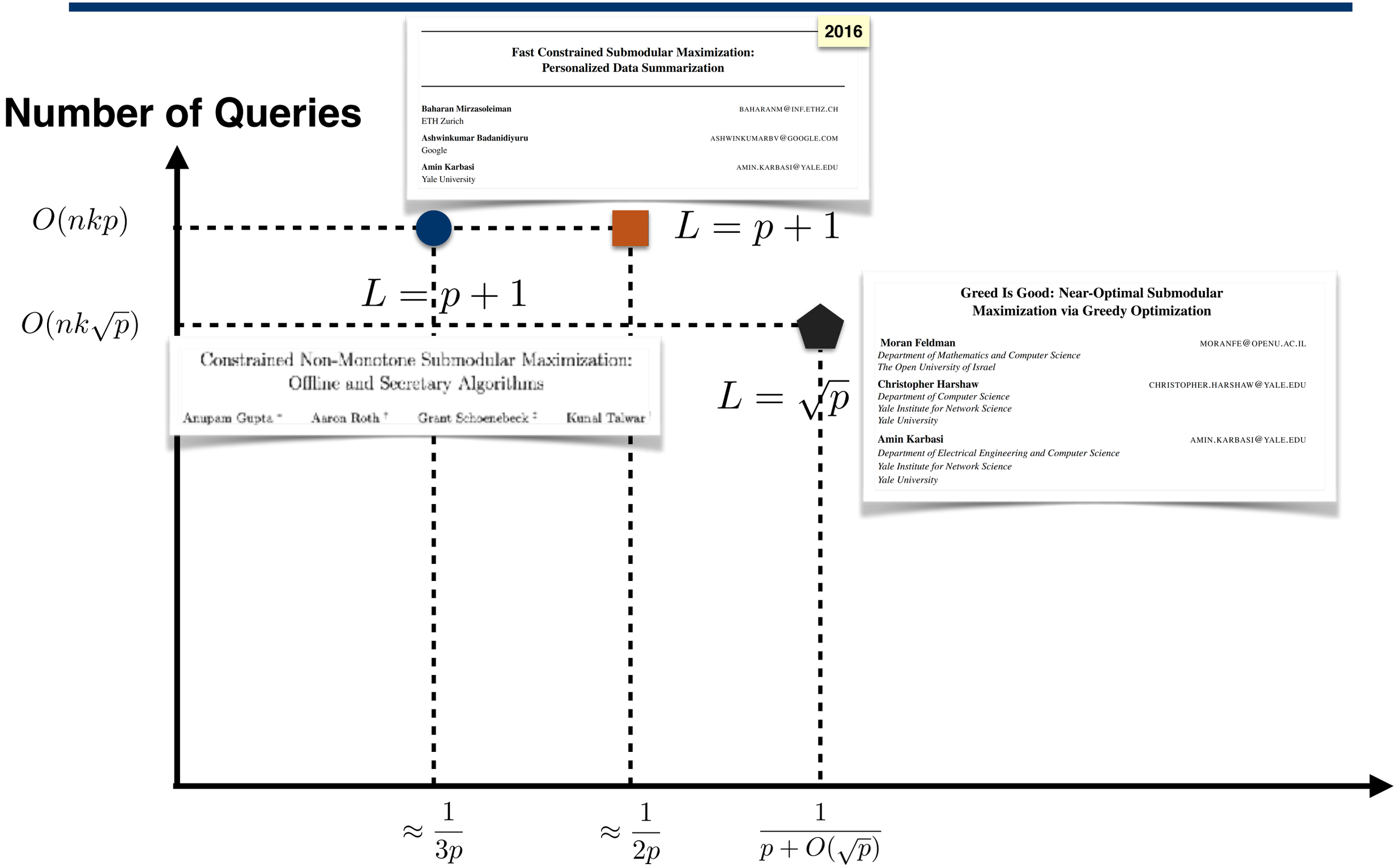
Constrained Submodular Maximization



Constrained Submodular Maximization



Constrained Submodular Maximization



2016

Fast Constrained Submodular Maximization: Personalized Data Summarization

Baharan Mirzasoleiman BAHARANM@INF.ETHZ.CH
ETH Zurich
Ashwinkumar Badanidiyuru ASHWINKUMARBV@GOOGLE.COM
Google
Amin Karbasi AMIN.KARBASI@YALE.EDU
Yale University

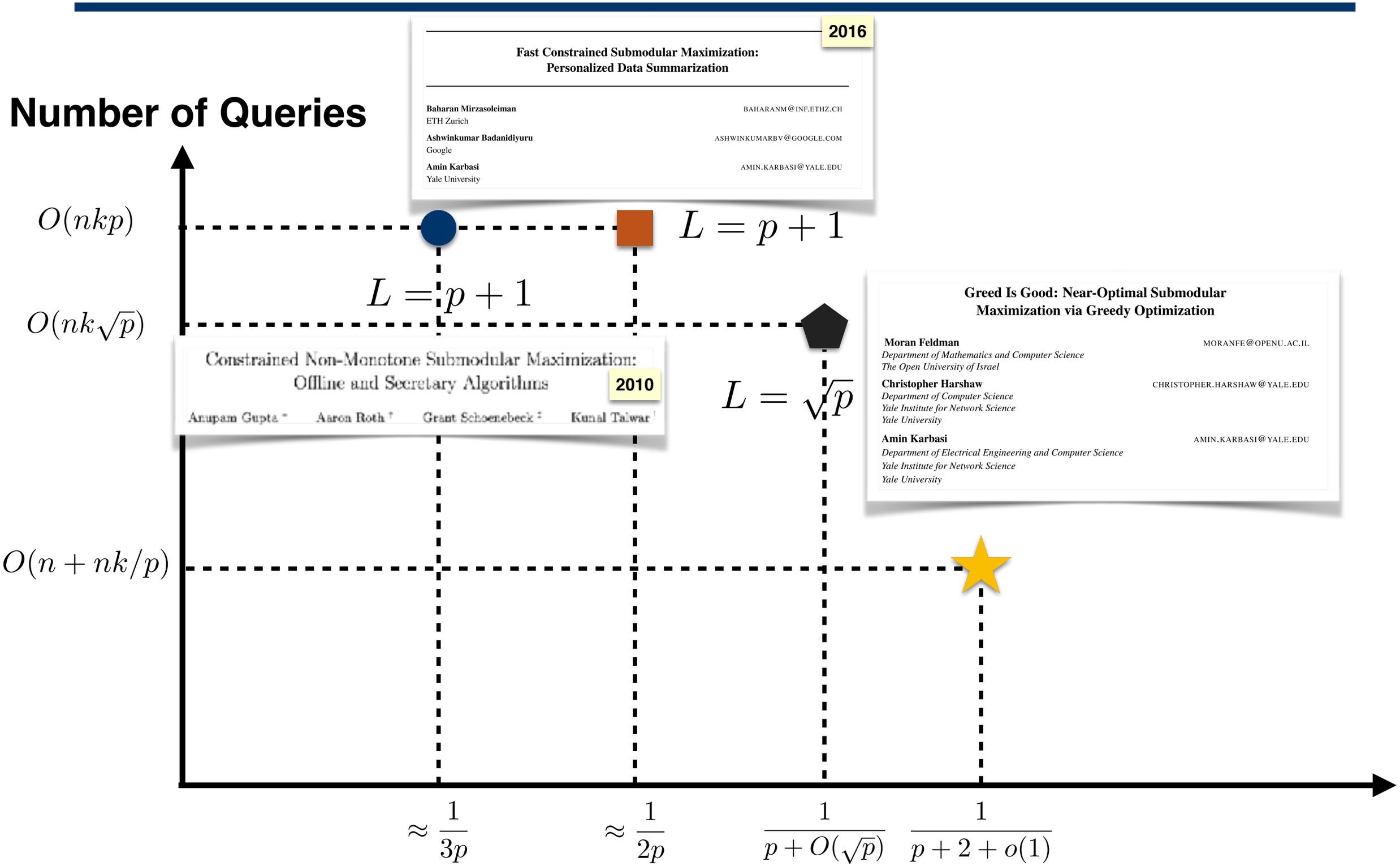
Constrained Non-Monotone Submodular Maximization: Offline and Secretary Algorithms

Anupam Gupta * Aaron Roth † Grant Schoenebeck ‡ Kunal Talwar

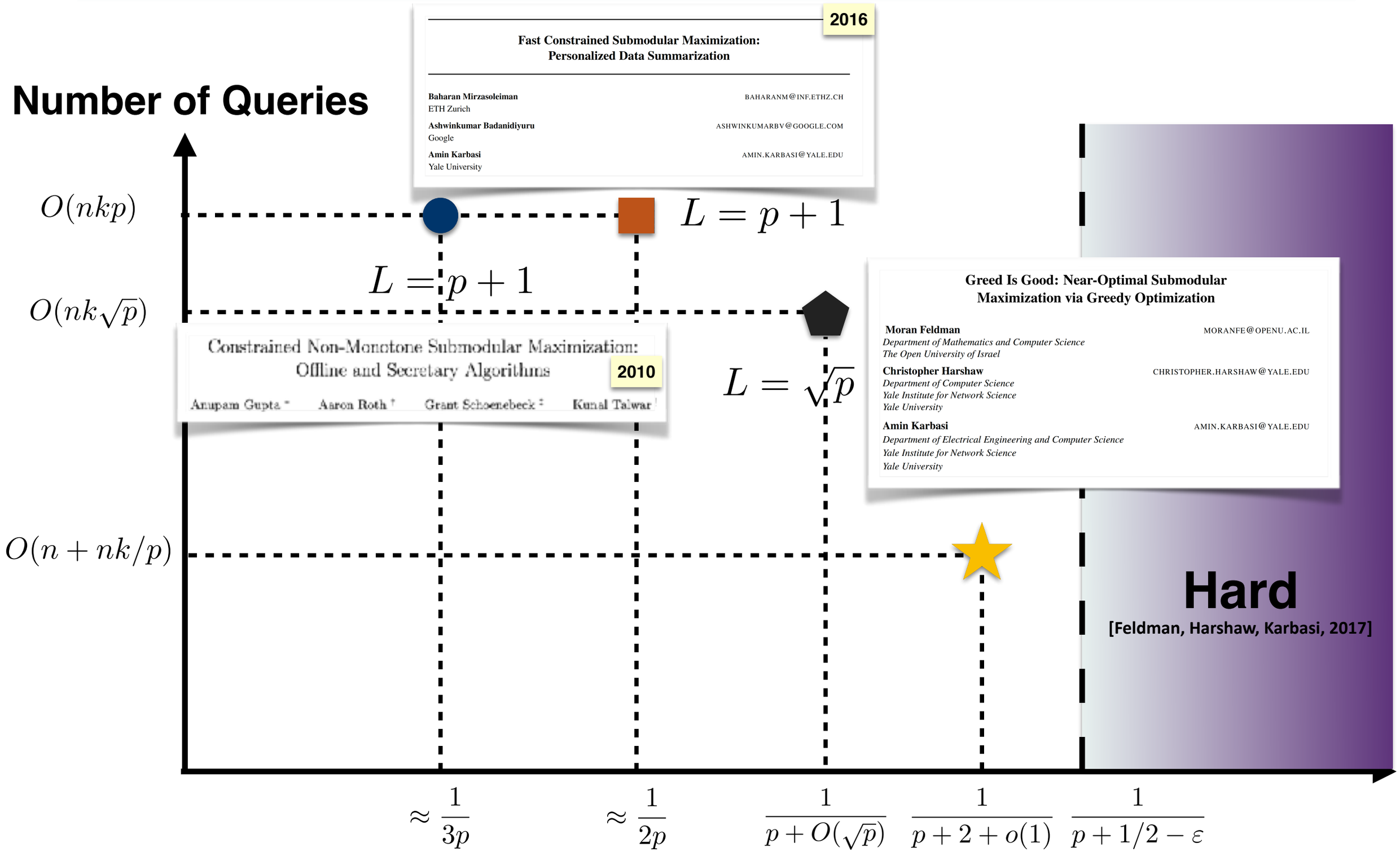
Greed Is Good: Near-Optimal Submodular Maximization via Greedy Optimization

Moran Feldman MORANFE@OPENU.AC.IL
Department of Mathematics and Computer Science
The Open University of Israel
Christopher Harshaw CHRISTOPHER.HARSHAW@YALE.EDU
Department of Computer Science
Yale Institute for Network Science
Yale University
Amin Karbasi AMIN.KARBASI@YALE.EDU
Department of Electrical Engineering and Computer Science
Yale Institute for Network Science
Yale University

Constrained Submodular Maximization



Constrained Submodular Maximization



Other Extensions

- Knapsack
- Weak Submodularity
- Sequence Submodularity
- Acceleration by reducing the ground set
- Interactive optimisation

Other Extensions

- Knapsack

  "A Nearly-linear Time Algorithm for Submodular Maximization with a Knapsack Constraint", Ene, Nguyen.2017

  "Fast algorithms for maximizing submodular functions", Badanidiyuru, Vondrák, 2014

  "Submodular Maximization Through Barrier Functions", Badanidiyuru, Karbasi, Kazemi, Vondrák, 2020

  "Submodular Optimization with Submodular Cover and Submodular Knapsack Constraints", Iyer, Bilmes 2020

- Weak Submodularity

  "Submodular meets Spectral: Greedy Algorithms for Subset Selection, Sparse Approximation and Dictionary Selection", Das, Kempe, 2015

  "Guarantees for Greedy Maximization of Non-submodular Functions with Applications", Bian, Buhmann, Krause, Tschiatschek, 2017

  "Weakly Submodular Maximization Beyond Cardinality Constraints: Does Randomization Help Greedy?", Chen, Feldman, Karbasi, 2018

- Sequence Submodularity

  "Selecting Sequences of Items via Submodular Maximization", Tschiatschek, Singla, Krause. 2017

  "Maximizing Sequence-Submodular Functions and its Application to Online Advertising", Alaei, Makhdoumi, Malekian. 2010

- Acceleration by reducing the ground set

  "Learning Sparse Combinatorial Representations via Two-stage Submodular Maximization", Balkanski, Mirzasoleiman, Krause, Singer'16

  "Probabilistic Submodular Maximization in Sub-Linear Time", Stan, Zadimoghaddam, Krause, Karbasi, 2017.

  "Submodular Meta-Learning", Adibi, Mokhtari, Hassani, 2017.

- Interactive optimisation

  "Adaptive Submodularity: Theory and Applications in Active Learning and Stochastic Optimization", Golovin, Krause, 2010.

  "Interactive Submodular Set Cover", Guillory, Bilmes, 2010.