Submodular Functions, Optimization, and Applications to Machine Learning

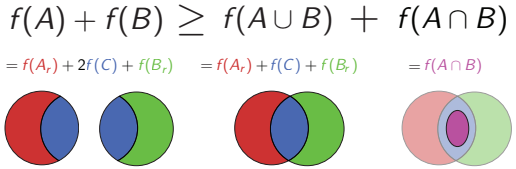
— Fall Quarter, Lecture 1 —

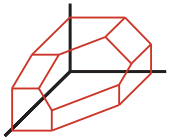
http://www.ee.washington.edu/people/faculty/bilmes/classes/ee563_spring_2018/

Prof. Jeff Bilmes

University of Washington, Seattle
Department of Electrical Engineering
http://melodi.ee.washington.edu/~bilmes

Sep 30, 2020





Announcements

- Welcome to: Submodular Functions, Optimization, and Applications to Machine Learning, EE563.
- Class: An introduction to submodular functions including methods for their optimization, and how they have been (and can be) applied in many application domains.
- Weekly Virtual Office Hours: Mondays, 10:00-11:00pm, via zoom (link posted on canvas).
- EEB 042, class web page is at our web page (`http://www.ee.washington.edu/people/faculty/bilmes/classes/ee563_spring_2018/`).
- Use our discussion board (`https://canvas.uw.edu/courses/1397085/discussion_topics`) for all questions, comments, so that all will benefit from them being answered.

Rough Class Outline

- Introduction to submodular functions: definitions, real-world and contrived examples, properties, operations that preserve submodularity, inequalities, variants and special submodular functions, and computational properties. Gain intuition, when is submodularity and supermodularity useful?
- A good model for cooperation, complexity, and irregularity. Also, a model for diversity, coverage, and information.
- Applications in data science, computer vision, tractable substructures in constraint satisfaction/SAT and graphical models, game theory, social networks, economics, information theory, structured convex norms, natural language processing, genomics/proteomics, sensor networks, summarization, probabilistic inference, and other areas of machine learning.

Rough Class Outline II

- Theory of matroids
- Polyhedral properties, polymatroids generalize matroids.
- Continuous Extensions: the Lovász extension of submodular functions, the Choquet integral, and convex and concave extensions, multi-linear extension.
- Submodular maximization algorithms under constraints, submodular cover problems, greedy algorithms, approximation guarantees.
- Streaming submodular maximization algorithms, regret bounds.
- Submodular minimization algorithms, numerical and combinatorial algorithms, computational properties. Submodular minimization under constraints.
- Lattices, Submodularity on lattices, DR submodularity, multi-argument submodularity

Useful Books, Classic Readings

- Fujishige, "Submodular Functions and Optimization", 2005
- Narayanan, "Submodular Functions and Electrical Networks", 1997
- Welsh, "Matroid Theory", 1975.
- Oxley, "Matroid Theory", 1992 (and 2011).
- Lawler, "Combinatorial Optimization: Networks and Matroids", 1976.
- Schrijver, "Combinatorial Optimization", 2003
- Gruenbaum, "Convex Polytopes, 2nd Ed", 2003.
- Additional readings will be announced on the slides.

Classic Readings:

- Jack Edmonds's paper "Submodular Functions, Matroids, and Certain Polyhedra" from 1970.
- Nemhauser, Wolsey, Fisher, "A Analysis of Approximations for Maximizing Submodular Set Functions-I", 1978
- Lovász's paper, "Submodular functions and convexity", from 1983.

Online material (most with an ML slant)

- Previous video version of this class http://j.ee.washington.edu/~bilmes/classes/ee596a_fall_2014/.
- Stefanie Jegelka & Andreas Krause's 2013 ICML tutorial http://techtalks.tv/talks/submodularity-in-machine-learning-new-directions-part-i/58125/
- NIPS, 2013 tutorial on submodularity http://melodi.ee.washington.edu/~bilmes/pgs/b2hd-bilmes2013-nips-tutorial.html and http://youtu.be/c4rBof38nKQ
- Andreas Krause's web page http://submodularity.org.
- Francis Bach's 2013 text. http://hal.archives-ouvertes.fr/docs/00/87/06/09/PDF/submodular_fot_revised_hal.pdf
- Tom McCormick's overview paper on submodular minimization http://people.commerce.ubc.ca/faculty/mccormick/sfmchap8a.pdf
- Georgia Tech's 2012 workshop on submodularity: http://www.arc.gatech.edu/events/arc-submodularity-workshop

Facts about the class

- Prerequisites: ideally knowledge in probability, statistics, convex optimization, and combinatorial optimization these will be reviewed as necessary. The course is open to students in all UW departments. Any questions, please contact me.
- Text: We will be drawing from the book by Satoru Fujishige entitled "Submodular Functions and Optimization" 2nd Edition, 2005, but we will also be reading handouts and research papers that will be posted here on this web page, especially for some of the application areas.
- Grades and Assignments: Grades will be based on a combination of a final project (40%), homeworks (60%). There will be between 3-5 homeworks during the quarter.
- Final project: The final project will consist of a 4-page paper (conference style) and a final short project presentation. The project must involve using/dealing mathematically with submodularity in some way or another, and might involve a contest!

Facts about the class

- Homework must be submitted electronically using our assignment dropbox (`https://canvas.uw.edu/courses/1397085/assignments`). PDF submissions only please. Photos of neatly hand written solutions, combined into one PDF, are fine
- Lecture slides - are being updated and improved this quarter. They will likely appear on the web page the night before, and the final version will appear just before class.
- Slides from previous version of this class are at `http://www.ee.washington.edu/people/faculty/bilmes/classes/ee596b_spring_2016/`.

Cumulative Outstanding Reading

- Read chapter 1 from Fujishige's book.

Class Road Map - EE563 (approximate goal)

- L1(3/26): Motivation, Applications, & Basic Definitions,
- L2(3/28): Machine Learning Apps (diversity, complexity, parameter, learning target, surrogate).
- L3(4/2): Info theory exs, more apps, definitions, graph/combinatorial examples
- L4(4/4): Graph and Combinatorial Examples, Matrix Rank, Examples and Properties, visualizations
- L5(4/9): More Examples/Properties/ Other Submodular Defs., Independence,
- L6(4/11): Matroids, Matroid Examples, Matroid Rank, Partition/Laminar Matroids
- L7(4/16): Laminar Matroids, System of Distinct Reps, Transversals, Transversal Matroid, Matroid Representation, Dual Matroids
- L8(4/18): Dual Matroids, Other Matroid Properties, Combinatorial Geometries, Matroids and Greedy.
- L9(4/23): Polyhedra, Matroid Polytopes, Matroids $\rightarrow$ Polymatroids
- L10(4/29): Matroids $\rightarrow$ Polymatroids, Polymatroids, Polymatroids and Greedy,
- L11(4/30): Polymatroids, Polymatroids and Greedy
- L12(5/2): Polymatroids and Greedy, Extreme Points, Cardinality Constrained Maximization
- L13(5/7): Constrained Submodular Maximization
- L14(5/9): Submodular Max w. Other Constraints, Cont. Extensions, Lovasz Extension
- L15(5/14): Cont. Extensions, Lovasz Extension, Choquet Integration, Properties
- L16(5/16): More Lovasz extension, Choquet, defs/props, examples, multiliear extension
- L17(5/21): Finish L.E., Multilinear Extension, Submodular Max/polyhedral approaches, Most Violated inequality, Still More on Matroids, Closure/Sat
- L–(5/28): Memorial Day (holiday)
- L18(5/30): Closure/Sat, Fund. Circuit/Dep
- L19(6/6): Fund. Circuit/Dep, Min-Norm Point Definitions, Review & Support for Min-Norm, Proof that min-norm gives optimal, Computing Min-Norm Vector for B_f maximization.

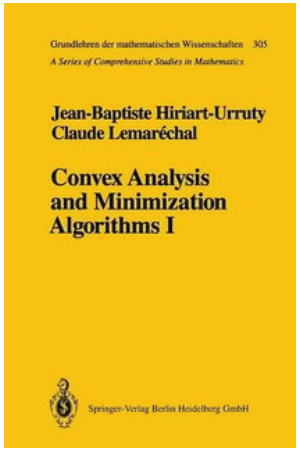
Last day of instruction, June 1st. Finals Week: June 2-8, 2018.

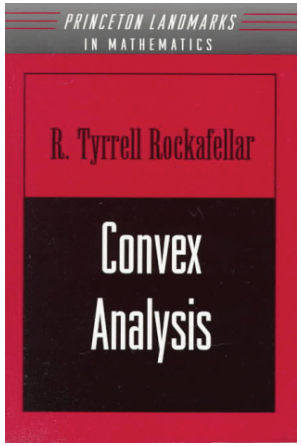
Class Road Map - EE563

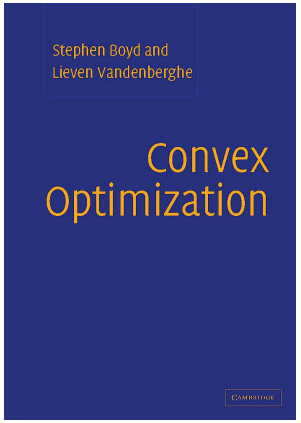
- L1(9/30): Motivation, Applications, Definitions, Properties
- L2(10/5):
- L3(10/7):
- L4(10/12):
- L5(10/14):
- L6(10/19):
- L7(10/21):
- L8(10/26):
- L9(10/28):
- L10(11/2):
- L11(11/4):
- L12(11/9):
- L–(11/11): Veterans Day, Holiday
- L13(11/16):
- L14(11/18):
- L15(11/23):
- L16(11/25):
- L17(11/30):
- L18(12/2):
- L19(12/7):
- L20(12/9): maximization.

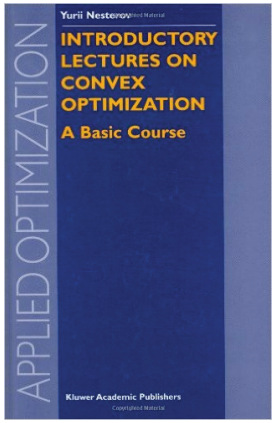
Last day of instruction, Fri. Dec 11th. Finals Week: Dec 12-18, 2020

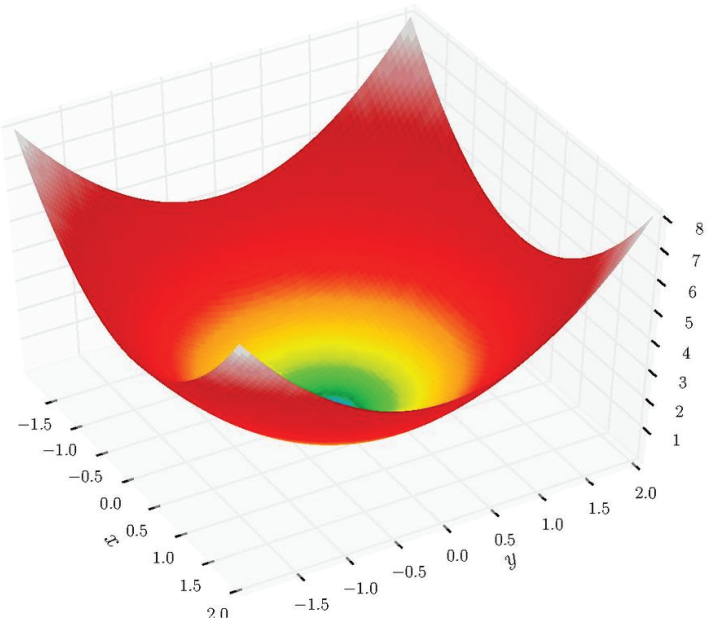
Convex Analysis in Machine Learning

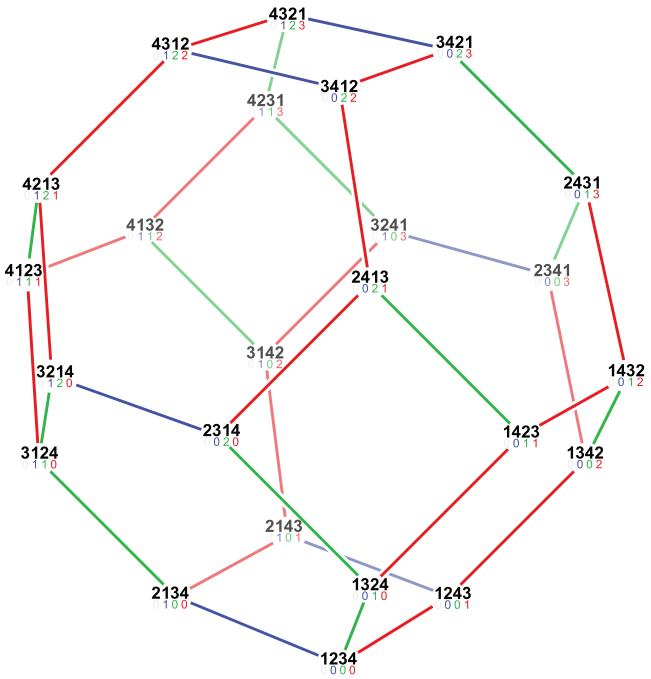












Successful Convexity in Machine Learning

- Linear and logistic regression, surrogate loss functions.
- Convex sparse regularizers (such as the ℓ_p family and nuclear norms).
- PSD matrices (i.e., positive semidefinite cone) and Gaussian densities.
- Optimizing non-linear and even non-convex classification/regression methods such as support-vector (SVMs) and kernel machines via convex optimization.
- Maximum entropy estimation
- The expectation-maximization (EM) algorithm.
- Ideas/techniques/insight for non-convex methods, convex minimization useful even for non-convex problems, such as Deep Neural Networks (DNNs). Convex analysis for non-convex problems.

A Convexity Limitation: Discrete Problems

Many Machine Learning problems are **inherently discrete:**

- Active learning/label selection.
- MAP & diverse k-best discrete probabilistic inference
- Data Science: data partitioning, clustering, summarization.
- Sparse modeling, compressed sensing, low-rank approximation.
- Probabilistic models: structure learning in graphical models and neural networks. Non-graphical global potentials.
- Variable and feature selection; dictionary selection.
- Deep neural network structure AutoML/NAS/
- Natural language processing (NLP): words, phrases, sentences, paragraphs, n-grams, syntax trees, graphs, semantic structures.
- Social choice and voting theory, social networks, viral marketing,
- (Multi-label) image segmentation in computer vision.
- Proteomics: selecting peptides, proteins, drug trial participants
- Genomics: cell-type or assay selection, genomic summarization
- Social networks, influence, viral marketing, information cascades, diffusion networks

Classic Discrete Optimization Approach

General Integer Programming (e.g., Integer Linear Programming (ILP), Integer Quadratic Programming (IQP), etc). Useful in:

- **Operations Research/Industrial Engineering**: facility and factory location, packing and covering.
- **Sensor placement** where to optimally place sensors?
- **Information:** Information theory, sets of random variables.
- **Geometry**: Polytopes and polyhedra
- **Mathematics:** e.g., Monge matrices, efficient dynamic programming, Birkhoff lattice theory
- **Combinatorial Problems**: e.g., sets, graphs, graph cuts, max k coverage, packings, coverings, partitions, paths, flows, matchings, colorings,
- **Algorithms**: Algorithms, and time/space complexity
- **Economics**: markets, economies of scale, mathematics of supply & demand

But, general ILP case can ignore useful and natural structures common to many problems.

Attractions of Convex Functions

Why do we like Convex Functions? (Quoting Lovász 1983):

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

Attractions of Convex Functions

Why do we like Convex Functions? (Quoting Lovász 1983):

1. *Convex functions occur in many mathematical models in economy, engineering, and other sciences. Convexity is a very natural property of various functions and domains occurring in such models; quite often the only non-trivial property which can be stated in general.*
2. *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.*

Attractions of Convex Functions

Why do we like Convex Functions? (Quoting Lovász 1983):

1. *Convex functions occur in many mathematical models in economy, engineering, and other sciences. Convexity is a very natural property of various functions and domains occurring in such models; quite often the only non-trivial property which can be stated in general.*
2. *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.*
3. *Convex functions and domains exhibit sufficient structure so that a mathematically beautiful and practically useful theory can be developed.*

Attractions of Convex Functions

Why do we like Convex Functions? (Quoting Lovász 1983):

1. *Convex functions occur in many mathematical models in economy, engineering, and other sciences. Convexity is a very natural property of various functions and domains occurring in such models; quite often the only non-trivial property which can be stated in general.*
2. *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.*
3. *Convex functions and domains exhibit sufficient structure so that a mathematically beautiful and practically useful theory can be developed.*
4. *There are theoretically and practically (reasonably) efficient methods to find the minimum of a convex function.*

Attractions of Submodular Functions

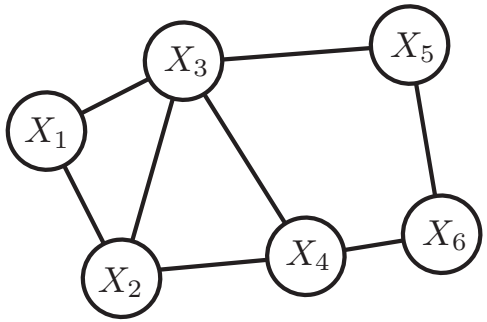
- In this course, we wish to demonstrate that submodular and supermodular functions also possess attractions of these four sorts as well.

Graphical Models and Decomposition

- Let $\mathcal{B}$ be the set of cliques of a graph G. A graphical model prescribes how to write functions $f : \{0,1\}^n \to \mathbb{R}$. Let $x \in \{0,1\}^n$

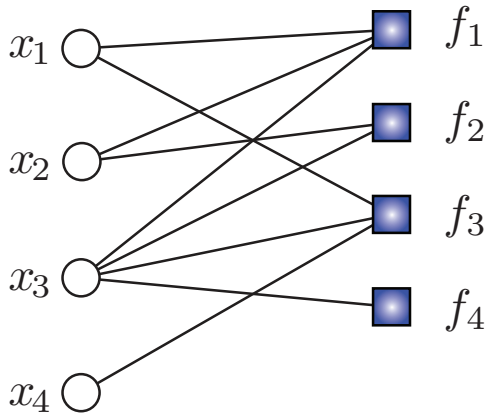
$$f(x) = \sum_{B \in \mathcal{B}} f_B(x_B) \tag{1.1}$$

Example: Undirected Graphs



$$\begin{aligned} f(x_{1:6}) &= f(x_1, x_2, x_3) + f(x_2, x_3, x_4) \\ &\quad + f(x_3, x_5) + f(x_5, x_6) + f(x_4, x_6) \\ f(x_{1:6}) &= f(x_1, x_2) + f(x_2, x_3) + f(x_3, x_1) \\ &\quad + f(x_2, x_3) + f(x_3, x_4) + f(x_4, x_2) \\ &\quad + f(x_3, x_5) + f(x_5, x_6) + f(x_4, x_6) \end{aligned}$$

Example: Factor/Hyper Graphs



$$\begin{aligned} f(x_{1:4}) &= f_1(x_1, x_2, x_3) + f_2(x_2, x_3) \\ &= f_3(x_1, x_3, x_4) + f_4(x_3) \end{aligned}$$

Graphical Models/Decomposition: Real-Object Example

- How to valuate a set of items?

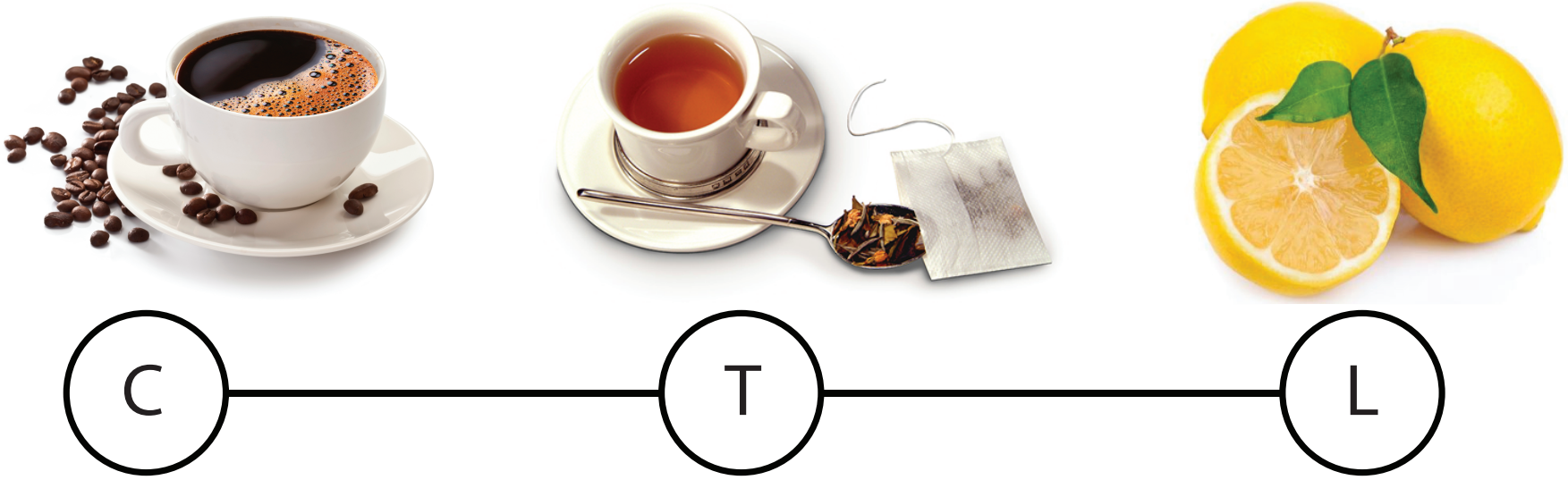
Graphical Models/Decomposition: Real-Object Example

- How to valuate a set of items?
- Let C, T, and L be binary variables indicating the presence or absence of items, and we wish to compute value(C, T, L).

value(C)
value(C, T)

Graphical Models/Decomposition: Real-Object Example

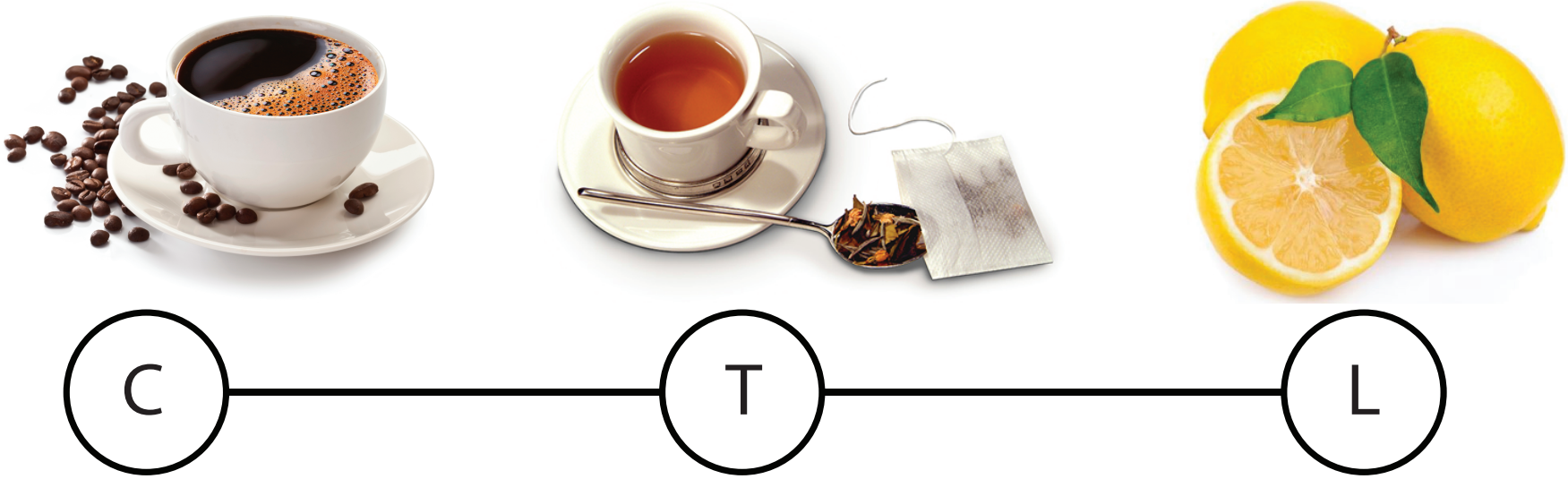
- How to valuate a set of items?
- Let C, T, and L be binary variables indicating the presence or absence of items, and we wish to compute value(C, T, L).
- Example: Value of Coffee (C), Tea (T), and Lemon (L).



$$\text{value}(C, T, L) = \text{value}(C, T) + \text{value}(T, L) \tag{1.2}$$

Graphical Decomposition Limitation: Manner of Interaction

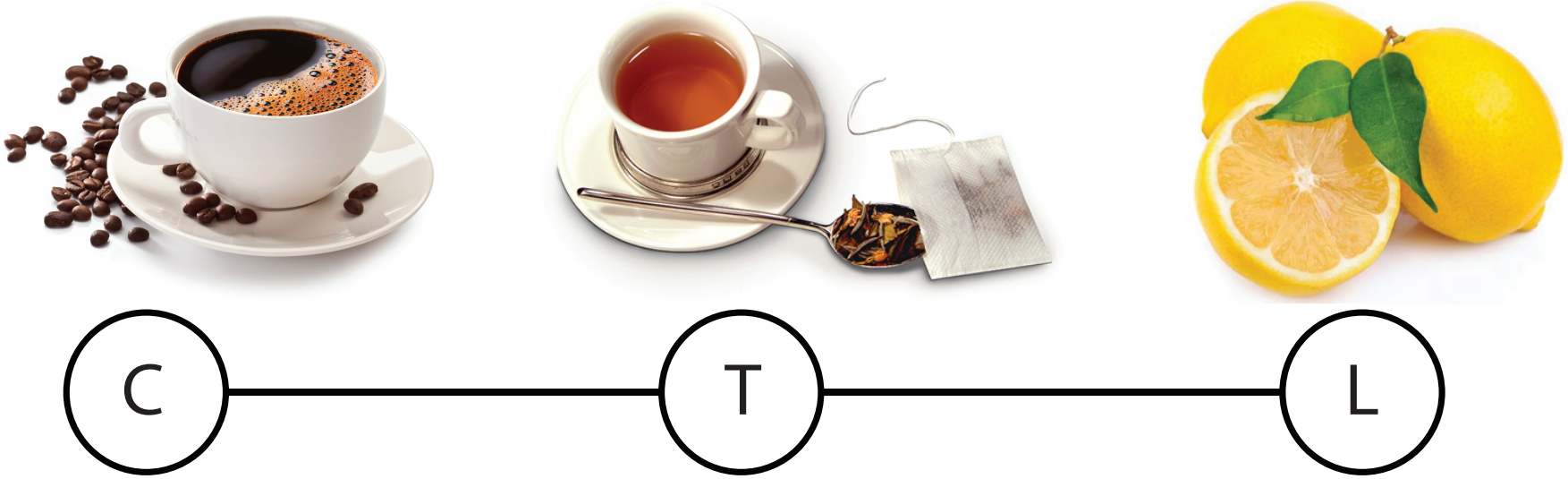
- Value of Coffee (C), Tea (T), and Lemon (L).



$$\text{value}(C, T, L) = \text{value}(C, T) + \text{value}(T, L) \tag{1.3}$$

Graphical Decomposition Limitation: Manner of Interaction

- Value of Coffee (C), Tea (T), and Lemon (L).



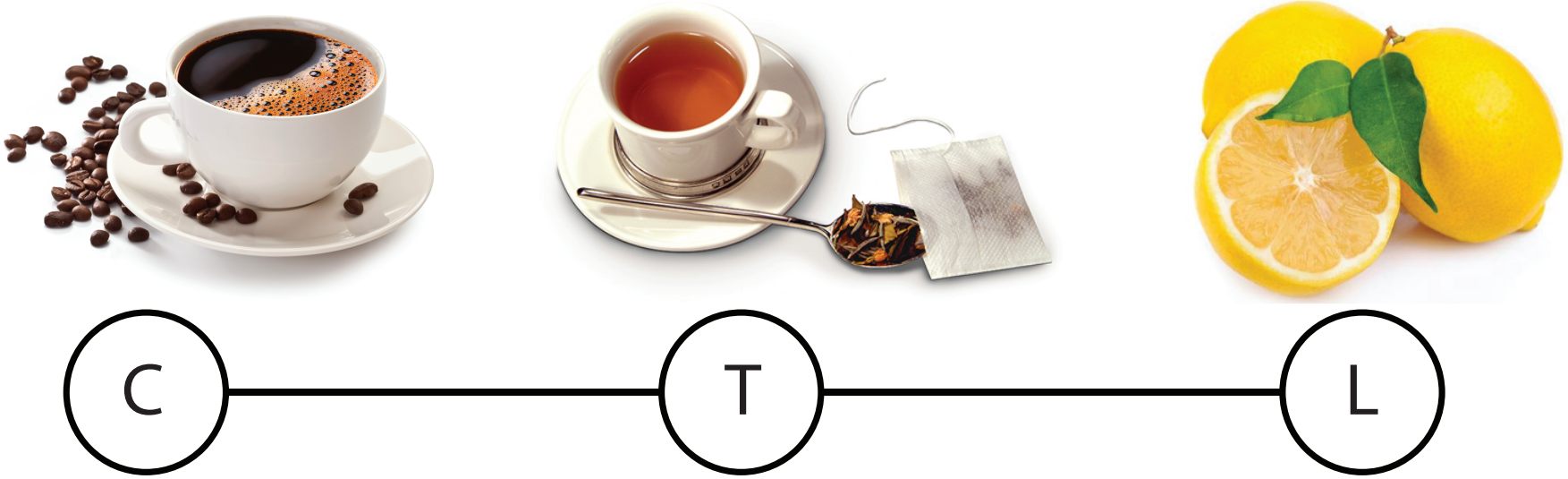
$$\text{value}(C, T, L) = \text{value}(C, T) + \text{value}(T, L) \tag{1.3}$$

- Coffee and Tea are "substitutive"

$$\text{value}(C, T) \leq \text{value}(C) + \text{value}(T) \tag{1.4}$$

Graphical Decomposition Limitation: Manner of Interaction

- Value of Coffee (C), Tea (T), and Lemon (L).



$$\text{value}(C, T, L) = \text{value}(C, T) + \text{value}(T, L) \tag{1.3}$$

- Coffee and Tea are "substitutive"

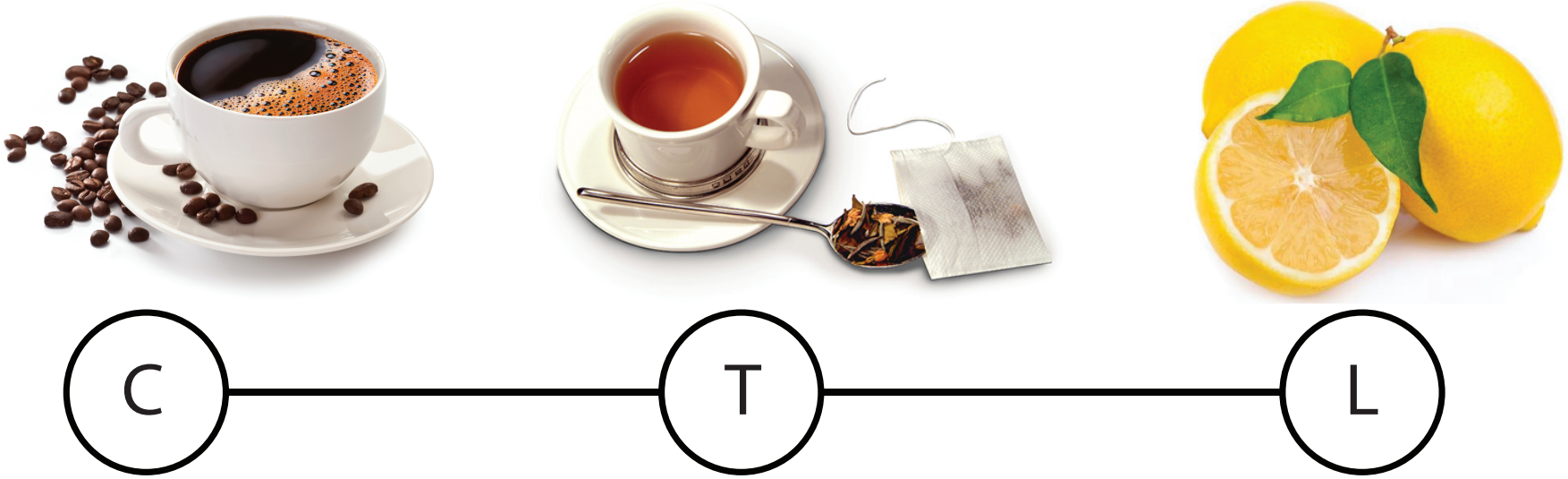
$$\text{value}(C, T) \leq \text{value}(C) + \text{value}(T) \tag{1.4}$$

- Tea and Lemon are "complementary"

$$\text{value}(T, L) \geq \text{value}(T) + \text{value}(L) \tag{1.5}$$

Graphical Decomposition Limitation: Manner of Interaction

- Value of Coffee (C), Tea (T), and Lemon (L).



$$\text{value}(C, T, L) = \text{value}(C, T) + \text{value}(T, L) \tag{1.3}$$

- Coffee and Tea are "substitutive"

$$\text{value}(C, T) \leq \text{value}(C) + \text{value}(T) \tag{1.4}$$

- Tea and Lemon are "complementary"

$$\text{value}(T, L) \geq \text{value}(T) + \text{value}(L) \tag{1.5}$$

- These are distinct non-graphically expressed manners of interaction!

Options for Cost Models via Graphical Decomposition

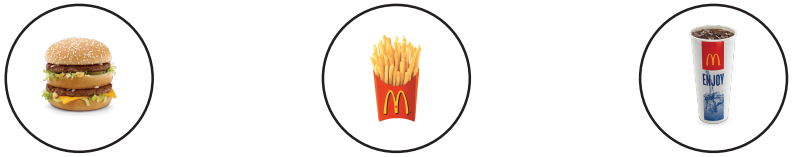
- Three items. Hamburger (H), Fries (F), Soda (S)

Options for Cost Models via Graphical Decomposition

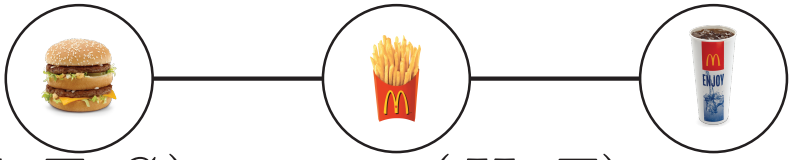
- Three items. Hamburger (H), Fries (F), Soda (S)



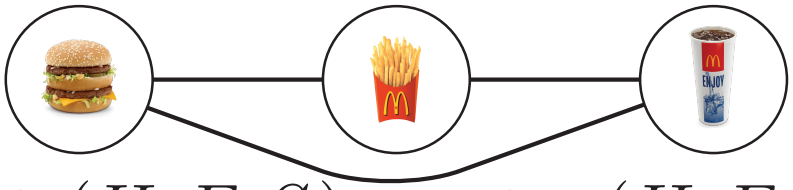
- Some graphical model options for costs(H, F, S):



$$\text{costs}(H, F, S) = \text{cst}_{\text{h}}(H) + \text{cst}_{\text{f}}(F) + \text{cst}_{\text{c}}(S)$$



$$\text{costs}(H, F, S) = \text{cst}_{\text{hf}}(H, F) + \text{cst}_{\text{fc}}(F, S)$$



$$\text{costs}(H, F, S) = \text{cst}_{\text{hfc}}(H, F, S)$$

Decompositions via Manner of Interaction

- costs(H, F, S) of Hamburger (H), Fries (F), Soda (S)



Consider components of cost: consumer-costs (ccs) and health-costs (hcs), each of which is ternary.

$$\text{costs}(H, F, S) = \text{ccs}(H, F, S) + \text{hcs}(H, F, S) \tag{1.6}$$

Decompositions via Manner of Interaction

- costs(H, F, S) of Hamburger (H), Fries (F), Soda (S)



Consider components of cost: consumer-costs (ccs) and health-costs (hcs), each of which is ternary.

$$\text{costs}(H, F, S) = \text{ccs}(H, F, S) + \text{hcs}(H, F, S) \tag{1.6}$$

- Consumer costs

$$\underbrace{\text{ccs}\Big(\text{🍔}\Big) - \text{ccs}\Big(\text{🍔}\Big)}_{\text{incremental cost of a coke on top of a hamburger}} \geq \underbrace{\text{ccs}\Big(\text{🍟}\Big) - \text{ccs}\Big(\text{🍟}\Big)}_{\text{incremental costs of a coke on top of both hamburger and fries.}}$$

Decompositions via Manner of Interaction

- costs(H, F, S) of Hamburger (H), Fries (F), Soda (S)



Consider components of cost: consumer-costs (ccs) and health-costs (hcs), each of which is ternary.

$$\text{costs}(H, F, S) = \text{ccs}(H, F, S) + \text{hcs}(H, F, S) \tag{1.6}$$

- Consumer costs

ccs() — ccs() $\geq$ ccs() — ccs()

- Health costs

hcs() — hcs() $\leq$ hcs() — hcs()

Decompositions via Manner of Interaction

- costs(H, F, S) of Hamburger (H), Fries (F), Soda (S)



Consider components of cost: consumer-costs (ccs) and health-costs (hcs), each of which is ternary.

$$\text{costs}(H, F, S) = \text{ccs}(H, F, S) + \text{hcs}(H, F, S) \tag{1.6}$$

- Consumer costs

ccs() — ccs() $\geq$ ccs() — ccs()

- Health costs

hcs() — hcs() $\leq$ hcs() — hcs()

- In both cases, graphical-only decompositions fail!

Sets and set functions $f : 2^V \to \mathbb{R}$

We are given a finite "ground" set V of objects, $2^V \triangleq \{A : A \subseteq V\}$

$V =$ 

Also given a set function $f : 2^V \to \mathbb{R}$ that valuates subsets $A \subseteq V$.

Ex: $f(V) = 6$

Sets and set functions $f : 2^V \to \mathbb{R}$

Subset $A \subseteq V$ of objects:

$A = \{$  $\}$

Also given a set function $f : 2^V \to \mathbb{R}$ that valuates subsets $A \subseteq V$.
Ex: $f(A) = 1$

Sets and set functions $f : 2^V \to \mathbb{R}$

Subset $B \subseteq V$ of objects:

$B =$ 

Also given a set function $f : 2^V \to \mathbb{R}$ that valuates subsets $A \subseteq V$.
Ex: $f(B) = 6$

Set functions are pseudo-Boolean functions

- Any set $A \subseteq V$ can be represented as a binary vector $x \in \{0,1\}^V$ (a "bit vector" representation of a set).

Set functions are pseudo-Boolean functions

- Any set $A \subseteq V$ can be represented as a binary vector $x \in \{0,1\}^V$ (a "bit vector" representation of a set).
- The characteristic vector $\mathbf{1}_A \in \{0,1\}^V$ of a set A is defined one where element $v \in V$ has value:

$$\mathbf{1}_A(v) = \begin{cases} 1 & \text{if } v \in A \\ 0 & else \end{cases} \tag{1.7}$$

$1_A \in \mathbb{R}^V$

$f(A)$ $f(1_A)$

Set functions are pseudo-Boolean functions

- Any set $A \subseteq V$ can be represented as a binary vector $x \in \{0,1\}^V$ (a "bit vector" representation of a set).
- The characteristic vector $\mathbf{1}_A \in \{0,1\}^V$ of a set A is defined one where element $v \in V$ has value:

$$\mathbf{1}_A(v) = \begin{cases} 1 & \text{if } v \in A \\ 0 & else \end{cases} \tag{1.7}$$

- Useful to be able to quickly map between $X = X(\mathbf{1}_X)$ and $x(X) \triangleq \mathbf{1}_X$.

Set functions are pseudo-Boolean functions

- Any set $A \subseteq V$ can be represented as a binary vector $x \in \{0,1\}^V$ (a "bit vector" representation of a set).
- The characteristic vector $\mathbf{1}_A \in \{0,1\}^V$ of a set A is defined one where element $v \in V$ has value:

$$\mathbf{1}_A(v) = \begin{cases} 1 & \text{if } v \in A \\ 0 & else \end{cases} \tag{1.7}$$

- Useful to be able to quickly map between $X = X(\mathbf{1}_X)$ and $x(X) \triangleq \mathbf{1}_X$.
- $f : \{0,1\}^V \to \{0,1\}$ are known as Boolean function.

Set functions are pseudo-Boolean functions

- Any set $A \subseteq V$ can be represented as a binary vector $x \in \{0,1\}^V$ (a "bit vector" representation of a set).
- The characteristic vector $\mathbf{1}_A \in \{0,1\}^V$ of a set A is defined one where element $v \in V$ has value:

$$\mathbf{1}_A(v) = \begin{cases} 1 & \text{if } v \in A \\ 0 & else \end{cases} \tag{1.7}$$

- Useful to be able to quickly map between $X = X(\mathbf{1}_X)$ and $x(X) \triangleq \mathbf{1}_X$.
- $f : \{0,1\}^V \to \{0,1\}$ are known as Boolean function.
- $f : \{0,1\}^V \to \mathbb{R}$ is a pseudo-Boolean function (submodular functions are a special case).

Two Equivalent Submodular Definitions

Definition 1.3.1 (submodular concave)

A function $f : 2^V \to \mathbb{R}$ is submodular if for any $A, B \subseteq V$, we have that:

$$f(A) + f(B) \geq f(A \cup B) + f(A \cap B) \tag{1.8}$$

An alternate and (as we will soon see) equivalent definition is:

Definition 1.3.2 (diminishing returns)

A function $f : 2^V \to \mathbb{R}$ is submodular if for any $A \subseteq B \subset V$, and $v \in V \setminus B$, we have that:

$f(A+v) + f(B) \geq f(B+v) + f(A)$

$$f(A \cup \{v\}) - f(A) \geq f(B \cup \{v\}) - f(B) \tag{1.9}$$

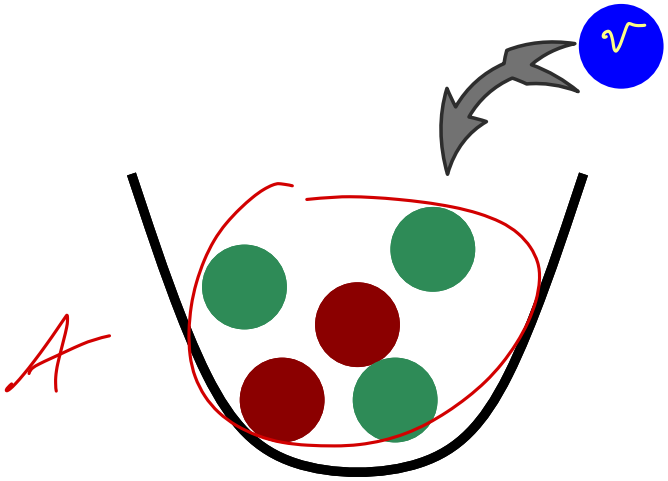
The incremental "value", "gain", or "cost" of v decreases (diminishes) as the context in which v is considered grows from A to B.

Example Submodular: Number of Colors of Balls in Urns

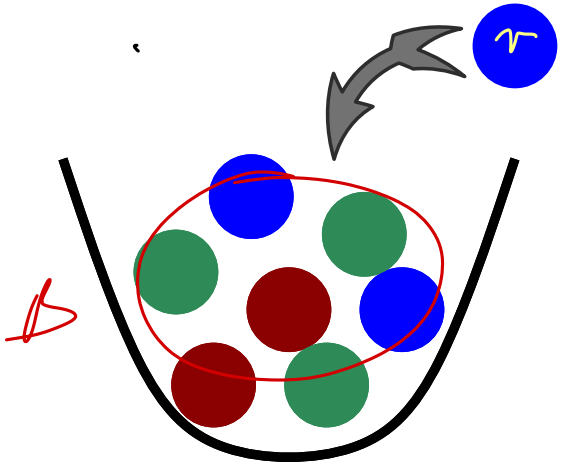
- Consider an urn containing colored balls. Given a set S of balls, $f(S)$ counts the number of distinct colors in S.

Example Submodular: Number of Colors of Balls in Urns

- Consider an urn containing colored balls. Given a set S of balls, $f(S)$ counts the number of distinct colors in S.



Initial value: 2 (colors in urn).
New value with added blue ball: 3



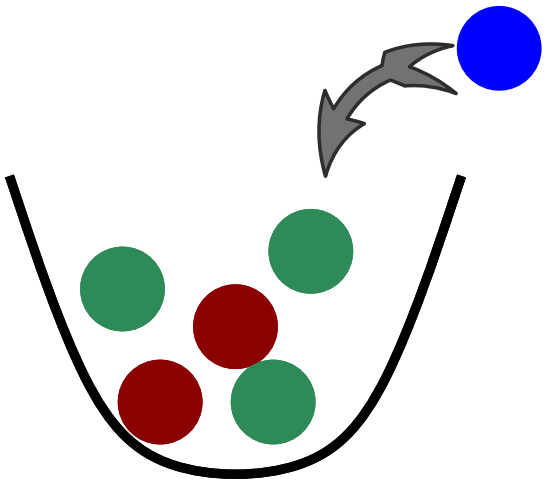
Initial value: 3 (colors in urn).
New value with added blue ball: 3

$A \subseteq B$

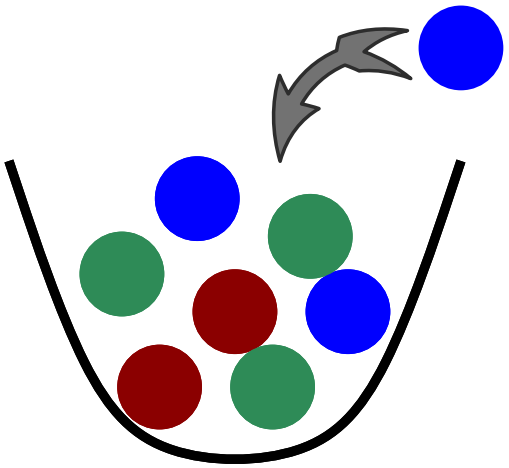
$1 = f(A + v) - f(A) > f(B + v) - f(B) = 0$

Example Submodular: Number of Colors of Balls in Urns

- Consider an urn containing colored balls. Given a set S of balls, $f(S)$ counts the number of distinct colors in S.



Initial value: 2 (colors in urn).
New value with added blue ball: 3

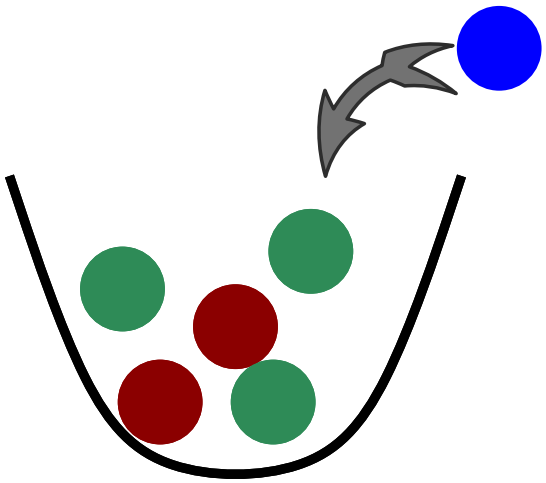


Initial value: 3 (colors in urn).
New value with added blue ball: 3

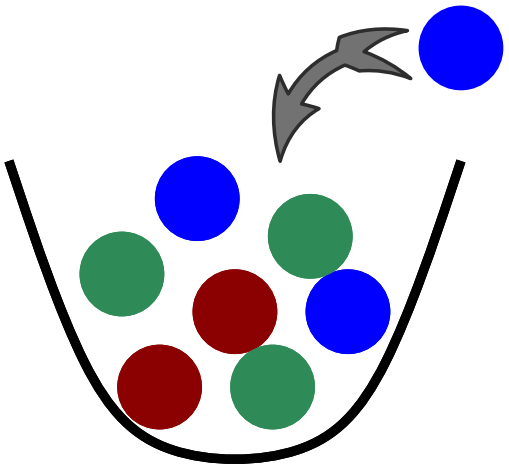
- Submodularity: Incremental Value of Object Diminishes in a Larger Context (diminishing returns).

Example Submodular: Number of Colors of Balls in Urns

- Consider an urn containing colored balls. Given a set S of balls, $f(S)$ counts the number of distinct colors in S.



Initial value: 2 (colors in urn).
New value with added blue ball: 3



Initial value: 3 (colors in urn).
New value with added blue ball: 3

- Submodularity: Incremental Value of Object Diminishes in a Larger Context (diminishing returns).
- Thus, f is submodular.

Two Equivalent Supermodular Definitions

Definition 1.3.3 (supermodular)

A function $f : 2^V \to \mathbb{R}$ is supermodular if for any $A, B \subseteq V$, we have that:

$$f(A) + f(B) \leq f(A \cup B) + f(A \cap B) \tag{1.10}$$

Definition 1.3.4 (supermodular (improving returns))

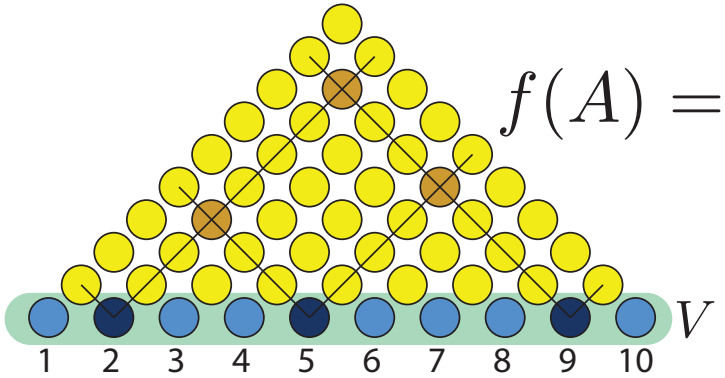
A function $f : 2^V \to \mathbb{R}$ is supermodular if for any $A \subseteq B \subset V$, and $v \in V \setminus B$, we have that:

$$f(A \cup \{v\}) - f(A) \leq f(B \cup \{v\}) - f(B) \tag{1.11}$$

- Incremental "value", "gain", or "cost" of v increases (improves) as the context in which v is considered grows from A to B.
- A function f is submodular iff $-f$ is supermodular.
- Additive If f both submodular and supermodular, then f is said to be **modular**, and $f(A) = c + \sum_{a \in A} \bar{f}(a)$ for some $\bar{f}$ (often $c = 0$).

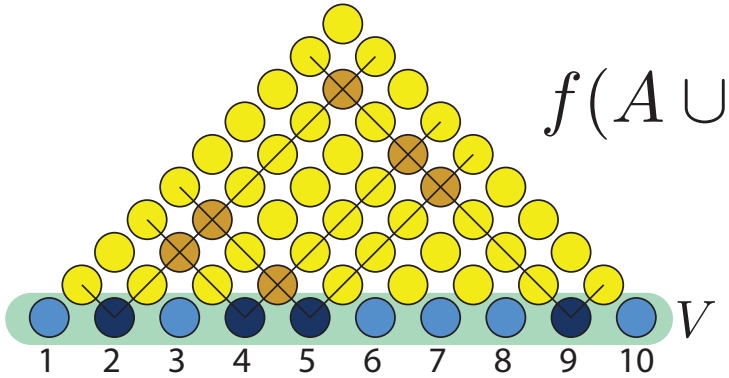
Example Supermodular: Number of Balls with Two Lines

Given ball pyramid, bottom row V is size $n = |V|$. For subset $S \subseteq V$ of bottom-row balls, draw 45° and 135° diagonal lines from each $s \in S$. Let $f(S)$ be number of non-bottom-row balls with two lines $\Rightarrow f(S)$ is supermodular.



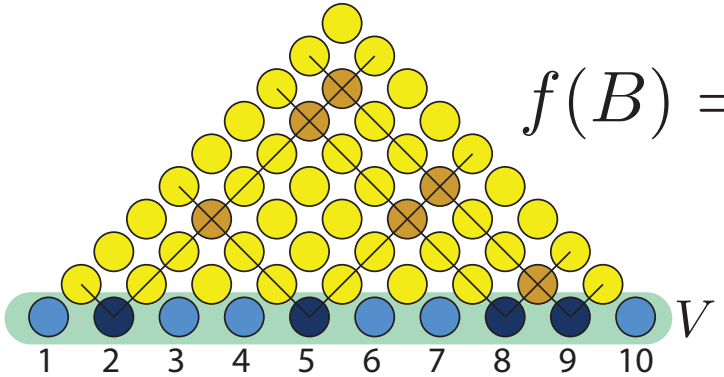
$f(A) = 3$

$A = \{2, 5, 9\}$



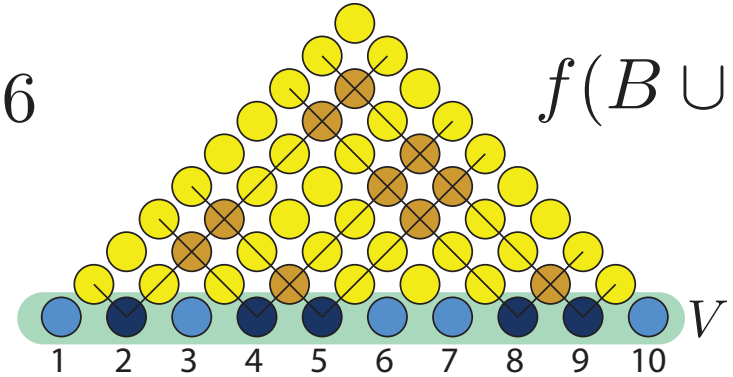
$f(A \cup \{4\}) = 6$

$A \cup \{4\} = \{2, 4, 5, 9\}$



$f(B) = 6$

$B = \{2, 5, 8, 9\}$



$f(B \cup \{4\}) = 10$

$B \cup \{4\} = \{2, 4, 5, 8, 9\}$

Submodular-Supermodular Decomposition

- As an alternative to graphical decomposition, we can decompose a function without resorting sums of local terms.

Submodular-Supermodular Decomposition

- As an alternative to graphical decomposition, we can decompose a function without resorting sums of local terms.

Theorem 1.3.5 (Additive Decomposition)

Let $h : 2^V \to \mathbb{R}$ be **any** *set function. Then there exists a submodular function $f : 2^V \to \mathbb{R}$ and a supermodular function $g : 2^V \to \mathbb{R}$ such that h may be additively decomposed as follows: For all $A \subseteq V$,*

$$h(A) = f(A) + g(A) \tag{1.12}$$

Submodular-Supermodular Decomposition

- As an alternative to graphical decomposition, we can decompose a function without resorting sums of local terms.

Theorem 1.3.5 (Additive Decomposition)

Let $h : 2^V \to \mathbb{R}$ be **any** *set function. Then there exists a submodular function $f : 2^V \to \mathbb{R}$ and a supermodular function $g : 2^V \to \mathbb{R}$ such that h may be additively decomposed as follows: For all $A \subseteq V$,*

$$h(A) = f(A) + g(A) \tag{1.12}$$

- For many applications (as we will see), either the submodular or supermodular component is naturally zero.

Submodular-Supermodular Decomposition

- As an alternative to graphical decomposition, we can decompose a function without resorting sums of local terms.

Theorem 1.3.5 (Additive Decomposition)

Let $h : 2^V \to \mathbb{R}$ be **any** *set function. Then there exists a submodular function $f : 2^V \to \mathbb{R}$ and a supermodular function $g : 2^V \to \mathbb{R}$ such that h may be additively decomposed as follows: For all $A \subseteq V$,*

$$h(A) = f(A) + g(A) \tag{1.12}$$

- For many applications (as we will see), either the submodular or supermodular component is naturally zero.
- Sometimes more natural than a graphical decomposition.

Submodular-Supermodular Decomposition

- As an alternative to graphical decomposition, we can decompose a function without resorting sums of local terms.

Theorem 1.3.5 (Additive Decomposition)

Let $h : 2^V \to \mathbb{R}$ be **any** *set function. Then there exists a submodular function $f : 2^V \to \mathbb{R}$ and a supermodular function $g : 2^V \to \mathbb{R}$ such that h may be additively decomposed as follows: For all $A \subseteq V$,*

a manner of interaction based decomposition.

$$h(A) = f(A) + g(A) \tag{1.12}$$

- For many applications (as we will see), either the submodular or supermodular component is naturally zero.
- Sometimes more natural than a graphical decomposition.
- Sometimes $h(A)$ has structure in terms of submodular functions but is non additively decomposed (one example is $h(A) = f(A)/g(A)$).

Discrete Optimization

- Unconstrained minimization and maximization:

$$\min_{X \subseteq V} f(X) \tag{1.13}$$

$$\max_{X \subseteq V} f(X) \tag{1.14}$$

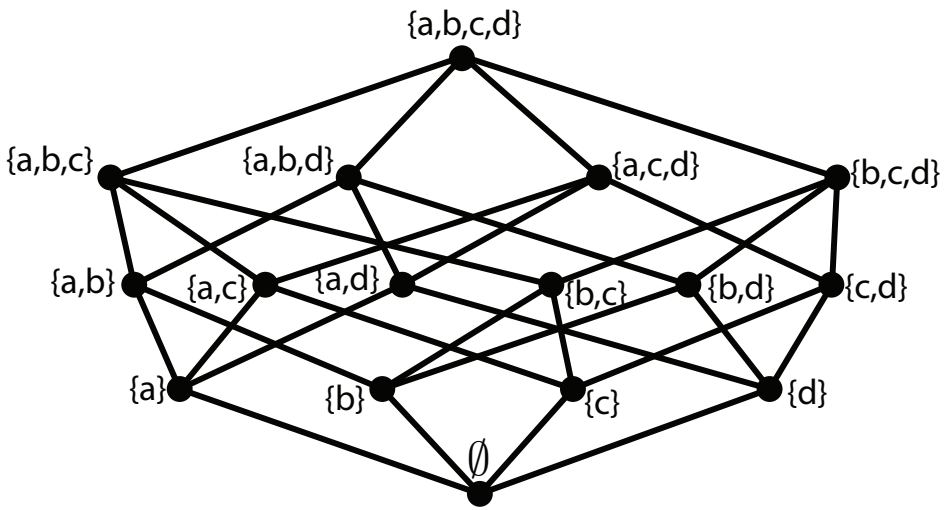
Discrete Optimization

- Unconstrained minimization and maximization:

$$\min_{X \subseteq V} f(X) \tag{1.13}$$

$$\max_{X \subseteq V} f(X) \tag{1.14}$$

- Knowing nothing about f, need 2^n queries for any quality assurance on candidate solution. Otherwise, solution can be unboundedly poor!!



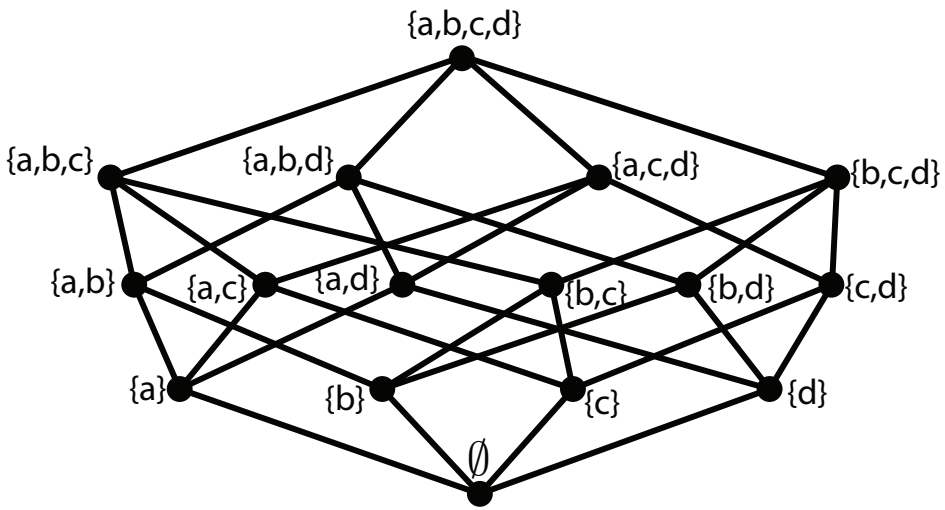
Discrete Optimization

- Unconstrained minimization and maximization:

$$\min_{X \subseteq V} f(X) \tag{1.13}$$

$$\max_{X \subseteq V} f(X) \tag{1.14}$$

- Knowing nothing about f, need 2^n queries for any quality assurance on candidate solution. Otherwise, solution can be unboundedly poor!!



- Alternatively, we may partition V into (necessarily disjoint) blocks $\{V_1, V_2, \ldots\}$ that collectively are good in some way.

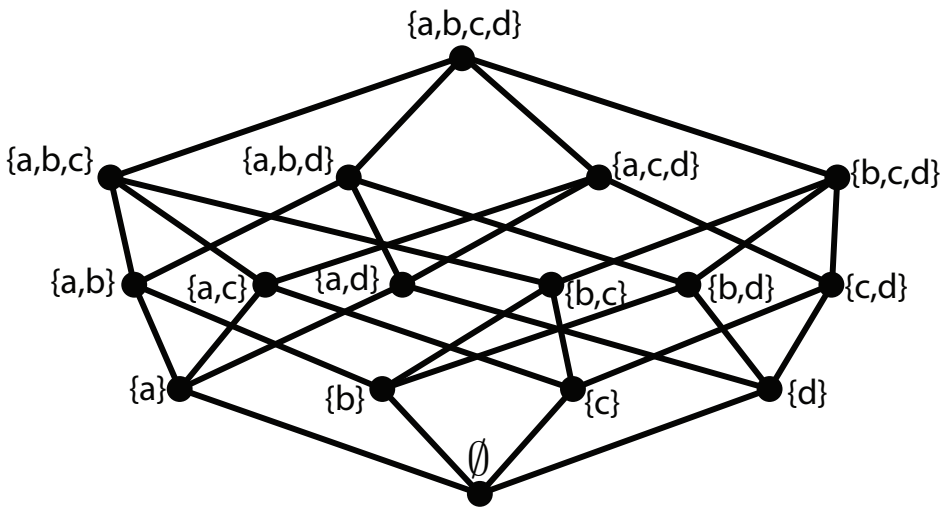
Discrete Optimization

- Unconstrained minimization and maximization:

$$\min_{X \subseteq V} f(X) \tag{1.13}$$

$$\max_{X \subseteq V} f(X) \tag{1.14}$$

- Knowing nothing about f, need 2^n queries for any quality assurance on candidate solution. Otherwise, solution can be unboundedly poor!!



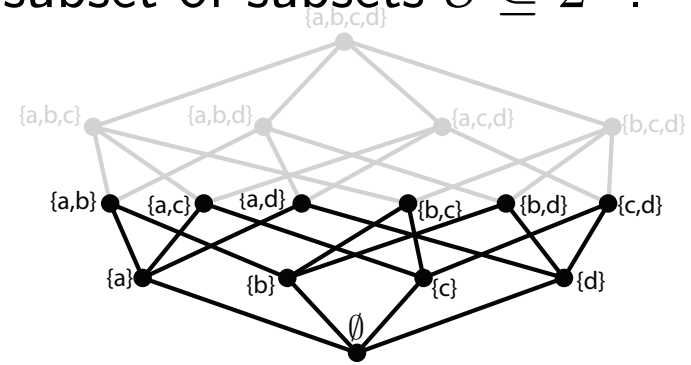
- Alternatively, we may partition V into (necessarily disjoint) blocks $\{V_1, V_2, \ldots\}$ that collectively are good in some way.
- When f is submodular, however, Eq. (1.13) is polytime, and Eq. (1.14) is constant-factor approximable. Partitionings are also approximable!

Constrained Discrete Optimization

- Constrained case: interested only in a subset of subsets $\mathcal{S} \subseteq 2^V$.

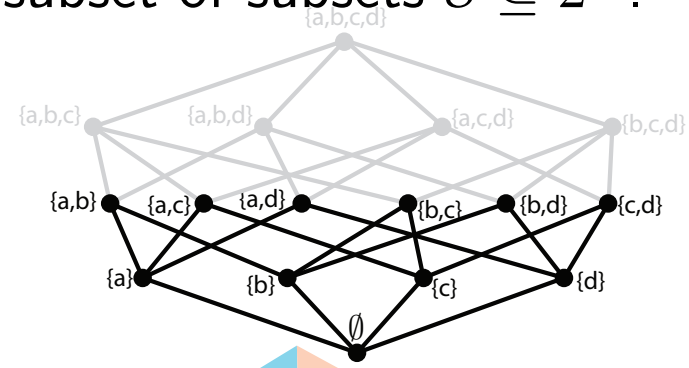
Constrained Discrete Optimization

- Constrained case: interested only in a subset of subsets $\mathcal{S} \subseteq 2^V$.
- Ex: Bounded size $\mathcal{S} = \{S \subseteq V : |S| \leq k\}$, or given cost vector w and budget, bounded cost $\left\{S \subseteq V : \sum_{s \in S} w(s) \leq b\right\}$.

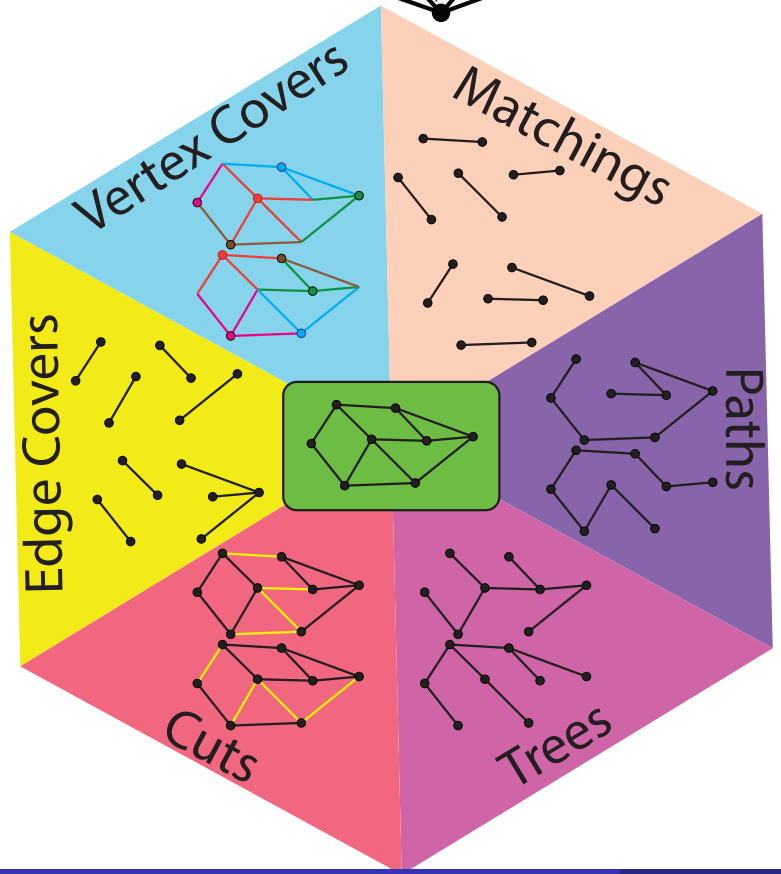


Constrained Discrete Optimization

- Constrained case: interested only in a subset of subsets $\mathcal{S} \subseteq 2^V$.
- Ex: Bounded size $\mathcal{S} = \{S \subseteq V : |S| \leq k\}$, or given cost vector w and budget, bounded cost $\{S \subseteq V : \sum_{s \in S} w(s) \leq b\}$.

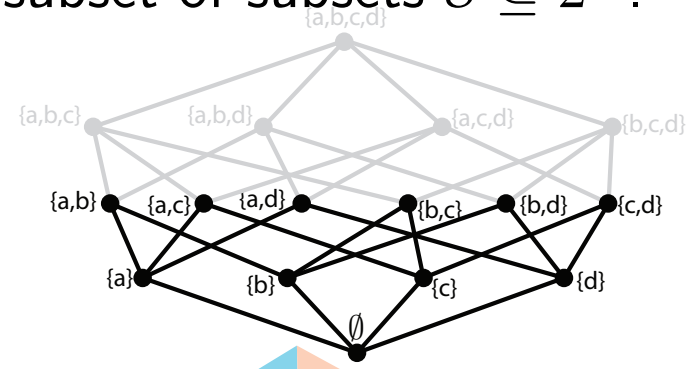


- Ex: feasible sets $\mathcal{S}$ as combinatorial objects

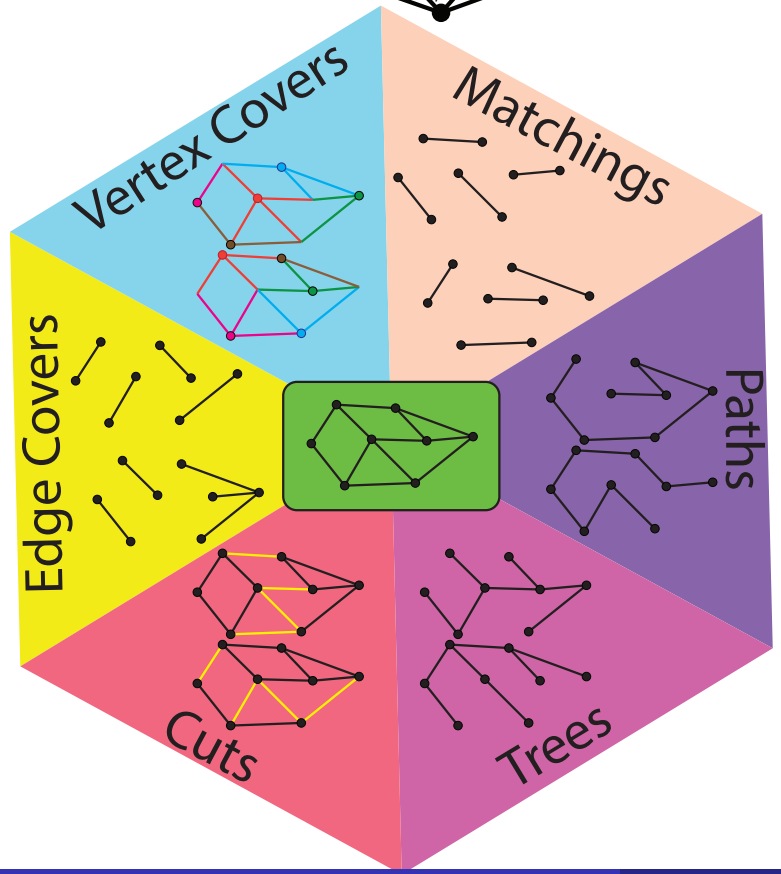


Constrained Discrete Optimization

- Constrained case: interested only in a subset of subsets $\mathcal{S} \subseteq 2^V$.
- Ex: Bounded size $\mathcal{S} = \{S \subseteq V : |S| \leq k\}$, or given cost vector w and budget, bounded cost $\{S \subseteq V : \sum_{s \in S} w(s) \leq b\}$.



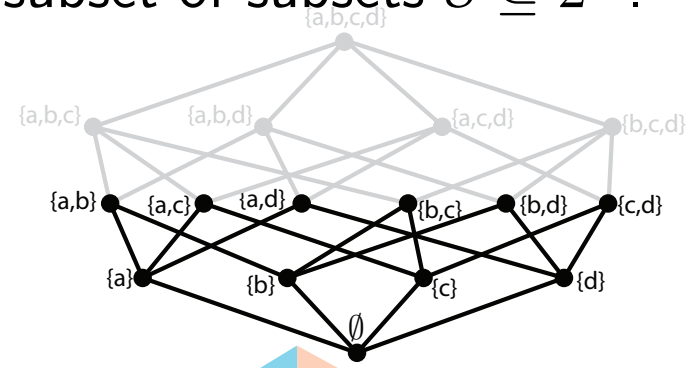
- Ex: feasible sets $\mathcal{S}$ as combinatorial objects



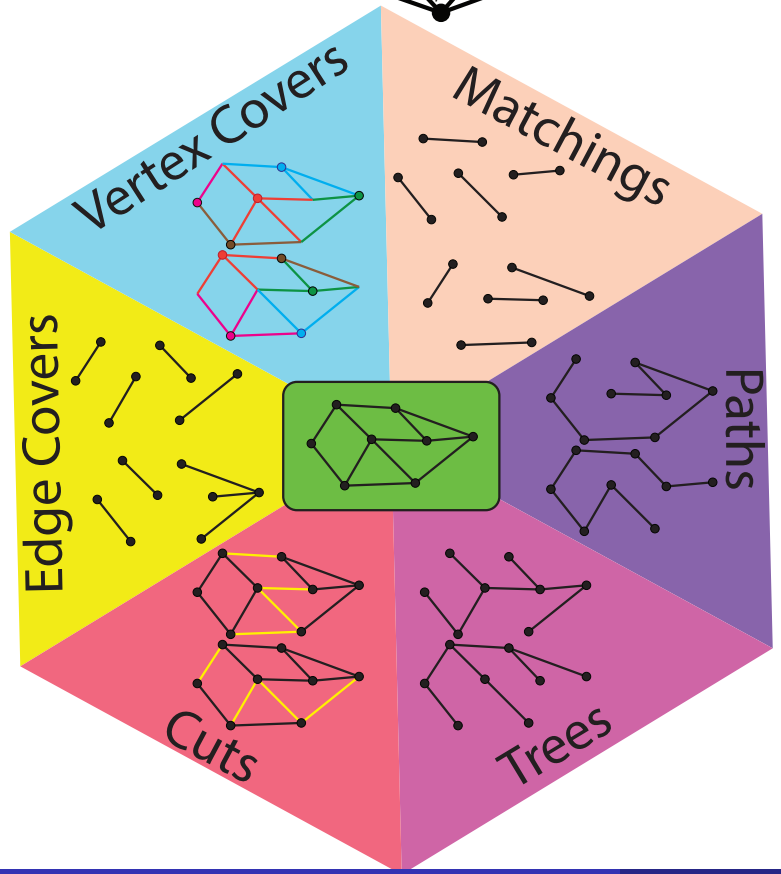
- Ex: feasible sets $\mathcal{S}$ as matroids.

Constrained Discrete Optimization

- Constrained case: interested only in a subset of subsets $\mathcal{S} \subseteq 2^V$.
- Ex: Bounded size $\mathcal{S} = \{S \subseteq V : |S| \leq k\}$, or given cost vector w and budget, bounded cost $\{S \subseteq V : \sum_{s \in S} w(s) \leq b\}$.



- Ex: feasible sets $\mathcal{S}$ as combinatorial objects



- Ex: feasible sets $\mathcal{S}$ as matroids.
- Ex: feasible sets $\mathcal{S}$ as sub-level sets of g, $\mathcal{S} = \{S \subseteq V : g(S) \leq \alpha\}$, sup-level sets $\mathcal{S} = \{S \subseteq V : g(S) \geq \alpha\}$

Constrained Discrete Optimization

- Constrained discrete optimization problems:

$$\begin{array}{ll} \text{maximize} & f(S) \\ \text{subject to} & S \in \mathcal{S} \end{array} \quad (1.15) \qquad \begin{array}{ll} \text{minimize} & f(S) \\ \text{subject to} & S \in \mathcal{S} \end{array} \quad (1.16)$$

where $\mathcal{S} \subseteq 2^V$ is the feasible set of sets.

Constrained Discrete Optimization

- Constrained discrete optimization problems:

$$\begin{array}{ll} \text{maximize} & f(S) \\ \text{subject to} & S \in \mathcal{S} \end{array} \quad (1.15) \qquad \begin{array}{ll} \text{minimize} & f(S) \\ \text{subject to} & S \in \mathcal{S} \end{array} \quad (1.16)$$

  where $\mathcal{S} \subseteq 2^V$ is the feasible set of sets.

- Fortunately, when f (and g) are submodular, these problems can often be solved with guarantees, often very efficiently!

Submodular and Supermodular Applications

- Algorithms: Algorithms can be developed that often are tractable (and as we will see many in this class).
- Applications: There are many seemingly different applications that are strongly related to submodularity.
- Submodularity and supermodularity is as common and natural for discrete problems in machine learning as is convexity/concavity for continuous problems.
- First, lets look at a few more very simple examples of submodular functions.

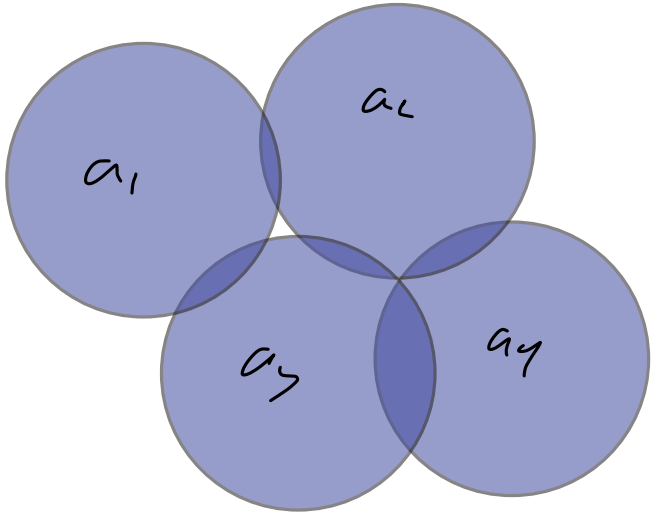
Continuous Set Cover

The area of the union of areas indexed by A

- Let V be a set of indices, and each $v \in V$ indexes a given fixed sub-area of some region in $\mathbb{R}^2$.
- Let area(v) be the area corresponding to item v.
- Let $f(S) = \bigcup_{s \in S}$ area(s) be the union of the areas indexed by elements in S.
- Then $f(S)$ is submodular, and corresponds to a continuous set cover function.

Continuous Set Cover

The area of the union of areas indexed by A — Example

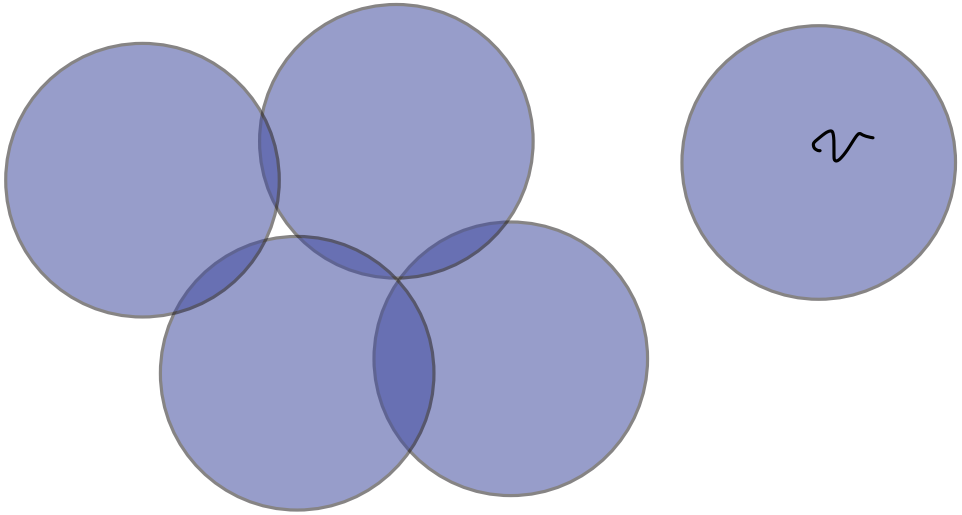


Union of areas of elements of A is given by:

$$f(A) = f(\{a_1, a_2, a_3, a_4\})$$

Continuous Set Cover

The area of the union of areas indexed by A — Example

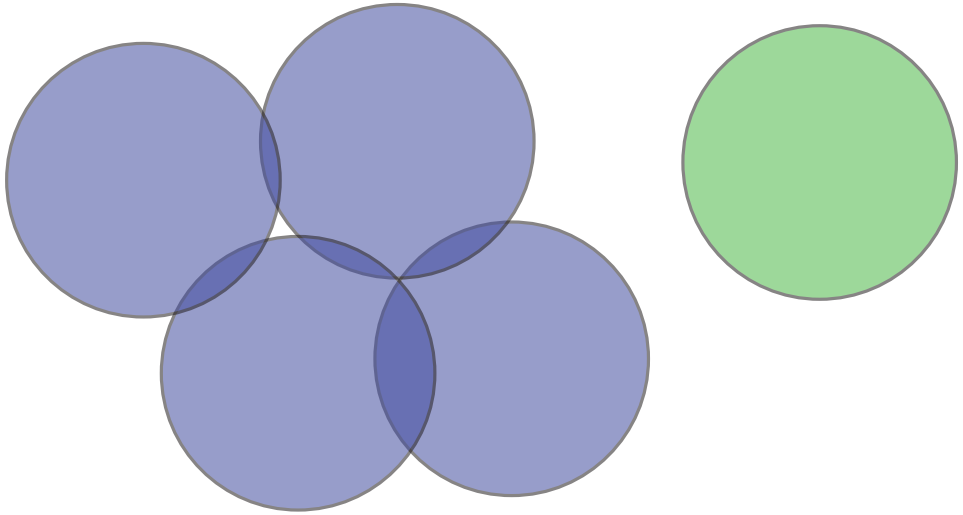


Area of A along with v:

$$f(A \cup \{v\}) = f(\{a_1, a_2, a_3, a_4\} \cup \{v\})$$

Continuous Set Cover

The area of the union of areas indexed by A — Example



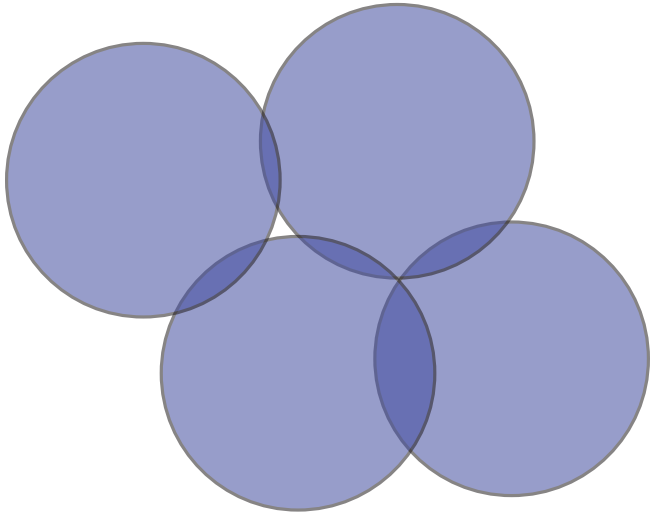
Gain (value) of v in context of A:

$$f(A \cup \{v\}) - f(A) = f(\{v\})$$

We get full value $f(\{v\})$ in this case since the area of v has no overlap with that of A.

Continuous Set Cover

The area of the union of areas indexed by A — Example

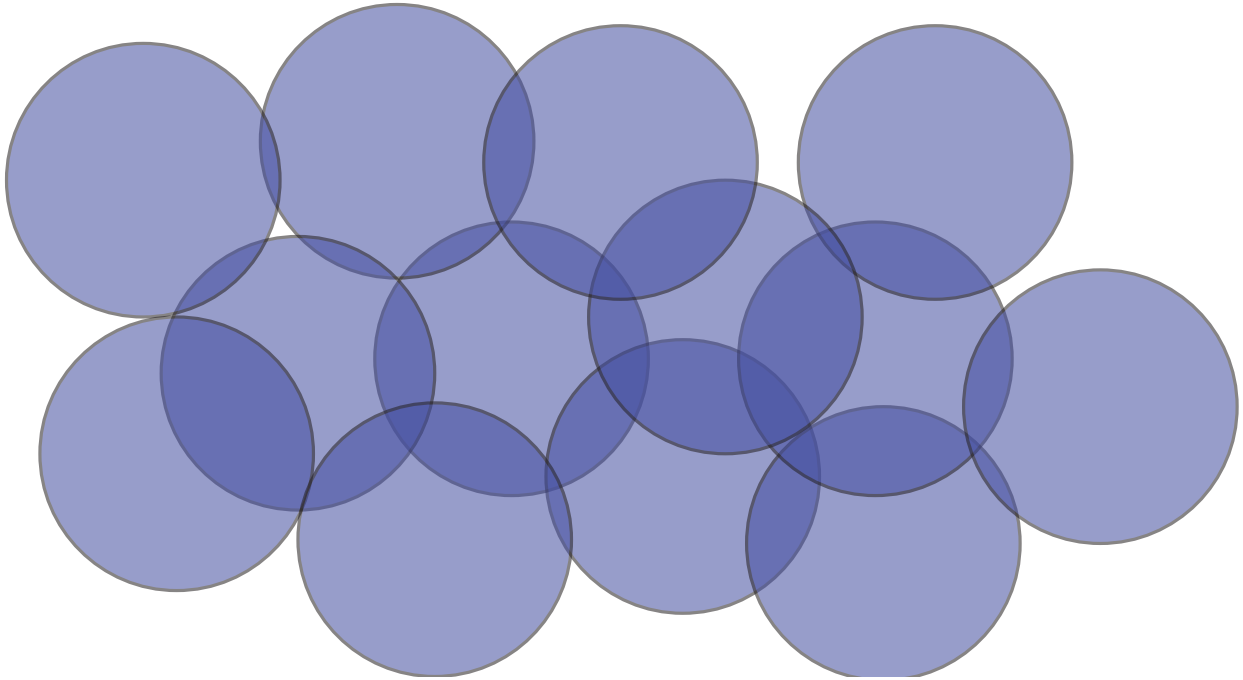


Area of A once again.

$$f(A) = f(\{a_1, a_2, a_3, a_4\})$$

Continuous Set Cover

The area of the union of areas indexed by A — Example

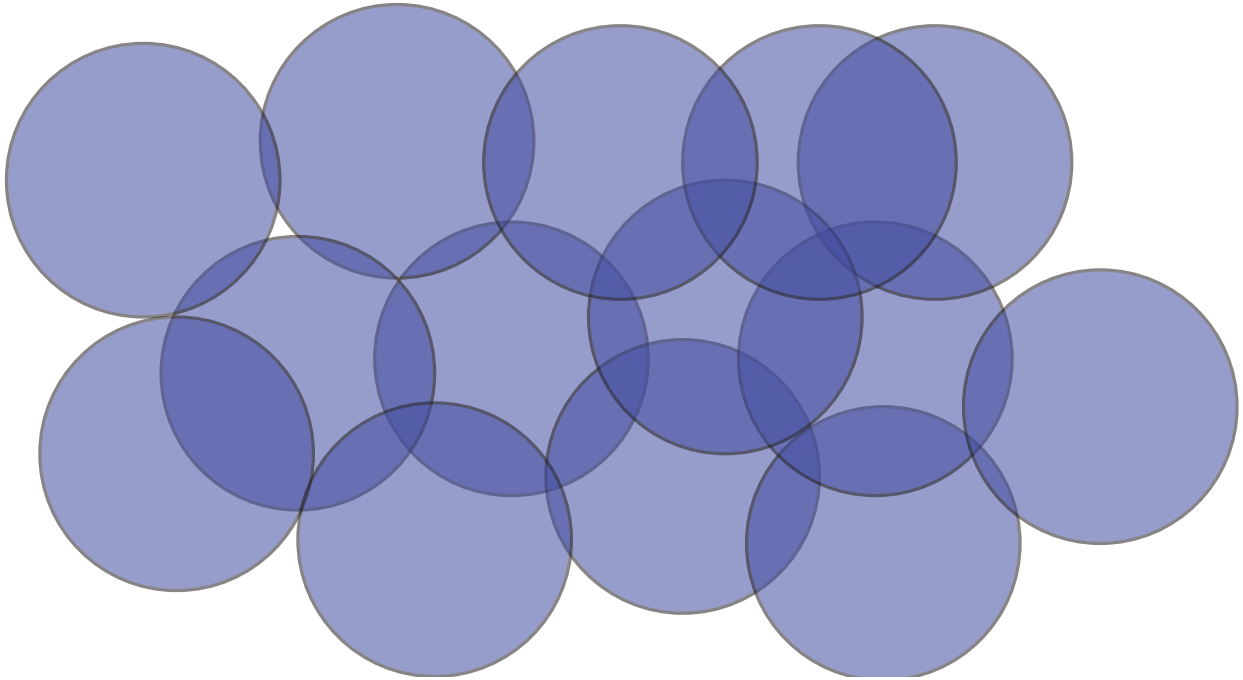


Union of areas of elements of $B \supset A$, where v is not included:

$$f(B) \text{ where } v \notin B \text{ and where } A \subseteq B$$

Continuous Set Cover

The area of the union of areas indexed by A — Example

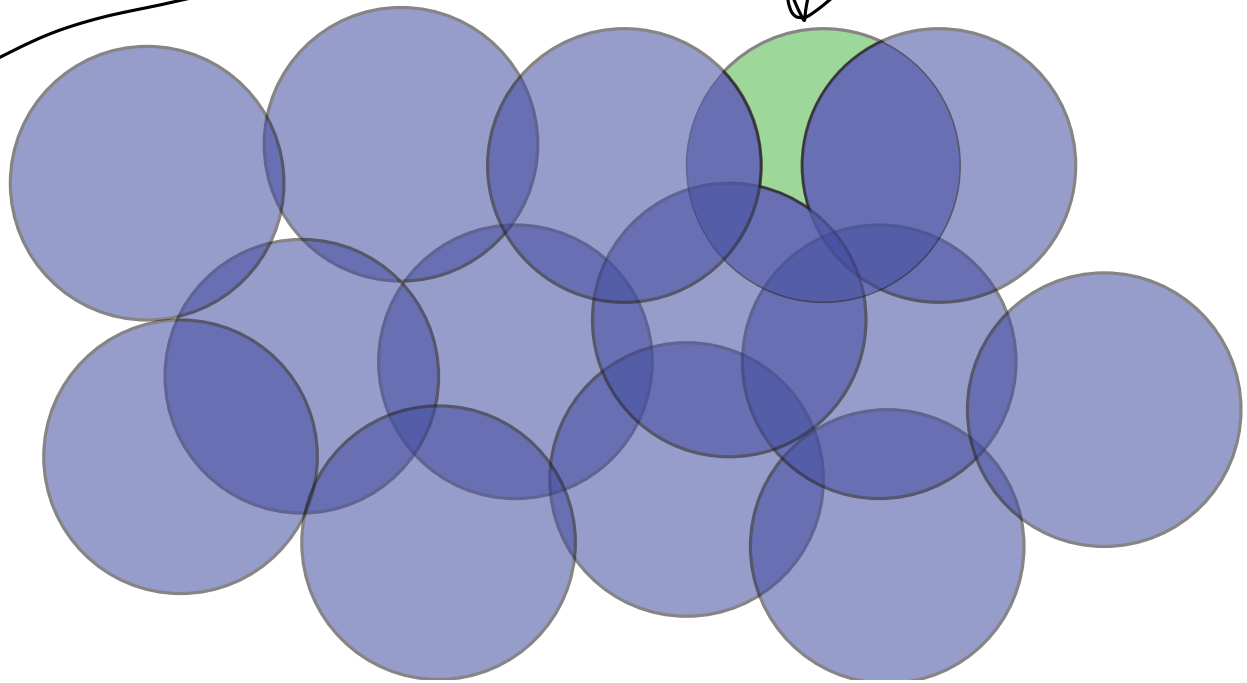


Area of B now also including v:

$$f(B \cup \{v\})$$

Continuous Set Cover

The area of the union of areas indexed by A — Example



Incremental value of v in the context of $B \supset A$.

$$f(B \cup \{v\}) - f(B) < f(\{v\}) = f(A \cup \{v\}) - f(A)$$

So benefit of v in the context of A is greater than the benefit of v in the context of $B \supseteq A$.

$$f(A) = \left| \bigcup_{a \in A} S_a \right|$$

Simple Consumer Costs

- Grocery store: finite set of items V that one can purchase.

Simple Consumer Costs

- Grocery store: finite set of items V that one can purchase.
- Each item $v \in V$ has a price $m(v)$.

Simple Consumer Costs


FUNNYRECEIPTS.com
TRADER JOE'S
[redacted]
Store [redacted]
OPEN 9:00AM TO 10:00PM DAILY

TJ'S PLAIN SOY MILK	1.69
EGGS BROWN	1.79
VEG TEMPEH ORGANIC 3 GRAIN	1.69
VEG SOY CHORIZO	1.99
PLAIN ORGANIC NONFAT YOGURT 32	2.99
LARGE BABY NON TAXABLE	1.99
GROCERY	0.49
3 @ 3 FOR 0.49	
SUBTOTAL	$12.63
TOTAL	$12.63

- Grocery store: finite set of items V that one can purchase.
- Each item $v \in V$ has a price $m(v)$.
- Basket of groceries $A \subseteq V$ costs:

$$m(A) = \sum_{a \in A} m(a), \tag{1.17}$$

the sum of individual item costs (no two-for-one discounts).

Simple Consumer Costs


FUNNYRECEIPTS.com
TRADER JOE'S
[redacted]
Store [redacted]
OPEN 9:00AM TO 10:00PM DAILY

TJ'S PLAIN SOY MILK	1.69
EGGS BROWN	1.79
VEG TEMPEH ORGANIC 3 GRAIN	1.69
VEG SOY CHORIZO	1.99
PLAIN ORGANIC NONFAT YOGURT 32	2.99
LARGE BABY NON TAXABLE	1.99
GROCERY	0.49
3 @ 3 FOR 0.49	
SUBTOTAL	$12.63
TOTAL	$12.63

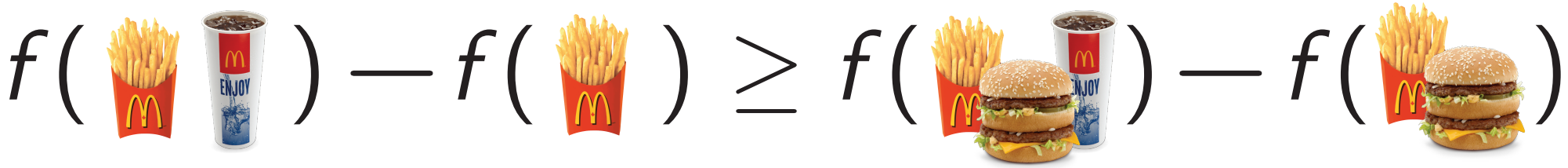
- Grocery store: finite set of items V that one can purchase.
- Each item $v \in V$ has a price $m(v)$.
- Basket of groceries $A \subseteq V$ costs:

$$m(A) = \sum_{a \in A} m(a), \tag{1.17}$$

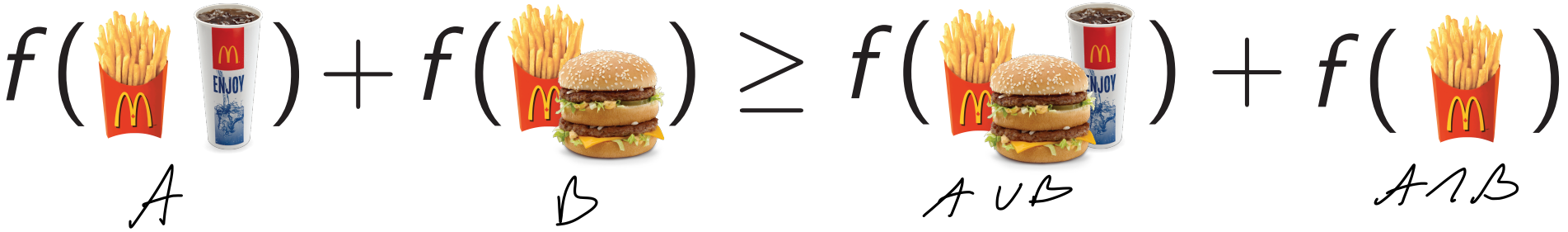
  the sum of individual item costs (no two-for-one discounts).
- This is known as a modular function.

Discounted Consumer Costs (as we saw earlier)

- Let f be the cost of purchasing a set of items (consumer cost). For example, $V = \{$"coke", "fries", "hamburger"$\}$ and $f(A)$ measures the cost of any subset $A \subseteq V$.We get diminishing returns:



- Simply rearranging terms, we get the other definition of submodularity:



- Typical: additional cost of a coke is free only if you add it to a fries and hamburger order.

Shared Fixed Costs (interacting costs)

- Costs often interact in the real world.

Shared Fixed Costs (interacting costs)

- Costs often interact in the real world.
- Ex: Let $V = \{v_1, v_2\}$ be a set of actions with:

v_1 = "buy milk at the store" v_2 = "buy honey at the store"

Shared Fixed Costs (interacting costs)

- Costs often interact in the real world.
- Ex: Let $V = \{v_1, v_2\}$ be a set of actions with:

v_1 = "buy milk at the store" v_2 = "buy honey at the store"

- For $A \subseteq V$, let $f(A)$ be the consumer cost of set of items A.

Shared Fixed Costs (interacting costs)

- Costs often interact in the real world.
- Ex: Let $V = \{v_1, v_2\}$ be a set of actions with:

v_1 = "buy milk at the store" v_2 = "buy honey at the store"

- For $A \subseteq V$, let $f(A)$ be the consumer cost of set of items A.
- $f(\{v_1\})$ = cost to drive to and from store c_d, and cost to purchase milk c_m, so $f(\{v_1\}) = c_d + c_m$.

Shared Fixed Costs (interacting costs)

- Costs often interact in the real world.
- Ex: Let $V = \{v_1, v_2\}$ be a set of actions with:

v_1 = "buy milk at the store" v_2 = "buy honey at the store"

- For $A \subseteq V$, let $f(A)$ be the consumer cost of set of items A.
- $f(\{v_1\})$ = cost to drive to and from store c_d, <u>and</u> cost to purchase milk c_m, so $f(\{v_1\}) = c_d + c_m$.
- $f(\{v_2\})$ = cost to drive to and from store c_d, <u>and</u> cost to purchase honey c_h, so $f(\{v_2\}) = c_d + c_h$.

Shared Fixed Costs (interacting costs)

- Costs often interact in the real world.
- Ex: Let $V = \{v_1, v_2\}$ be a set of actions with:

v_1 = "buy milk at the store" v_2 = "buy honey at the store"

- For $A \subseteq V$, let $f(A)$ be the consumer cost of set of items A.
- $f(\{v_1\})$ = cost to drive to and from store c_d, <u>and</u> cost to purchase milk c_m, so $f(\{v_1\}) = c_d + c_m$.
- $f(\{v_2\})$ = cost to drive to and from store c_d, <u>and</u> cost to purchase honey c_h, so $f(\{v_2\}) = c_d + c_h$.
- But $f(\{v_1, v_2\}) = c_d + c_m + c_h < 2c_d + c_m + c_h$ since c_d (driving) is a shared fixed cost.

Shared Fixed Costs (interacting costs)

- Costs often interact in the real world.
- Ex: Let $V = \{v_1, v_2\}$ be a set of actions with:

  v_1 = "buy milk at the store" v_2 = "buy honey at the store"

- For $A \subseteq V$, let $f(A)$ be the consumer cost of set of items A.
- $f(\{v_1\})$ = cost to drive to and from store c_d, and cost to purchase milk c_m, so $f(\{v_1\}) = c_d + c_m$.
- $f(\{v_2\})$ = cost to drive to and from store c_d, and cost to purchase honey c_h, so $f(\{v_2\}) = c_d + c_h$.
- But $f(\{v_1, v_2\}) = c_d + c_m + c_h < 2c_d + c_m + c_h$ since c_d (driving) is a shared fixed cost.
- Shared fixed costs are submodular: $f(v_1) + f(v_2) \geq f(v_1, v_2) + f(\emptyset)$

Markets: Supply Side Economies of scale

- Economies of Scale: Many goods and services can be produced at a much lower per-unit cost only if they are produced in very large quantities.
- The profit margin for producing a unit of goods is improved as more of those goods are created.
- If you already make a good, making a similar good is easier than if you start from scratch (e.g., Apple making both iPod and iPhone).
- An argument in favor of free trade is that it opens up larger markets for firms (especially in otherwise small markets), thereby enabling better economies of scale, and hence greater efficiency (lower costs and resources per unit of good produced).

Supply Side Economies of Scale

- What is a good model of the cost of manufacturing a set of items?

Supply Side Economies of Scale

- What is a good model of the cost of manufacturing a set of items?
- Let V be a set of possible items to manufacture, and let $f(S)$ for $S \subseteq V$ be the manufacture costs of items in the subset S.

Supply Side Economies of Scale

- What is a good model of the cost of manufacturing a set of items?
- Let V be a set of possible items to manufacture, and let $f(S)$ for $S \subseteq V$ be the manufacture costs of items in the subset S.
- Ex: V might be paint colors to produce: green, red, blue, yellow, white, etc.

Supply Side Economies of Scale

- What is a good model of the cost of manufacturing a set of items?
- Let V be a set of possible items to manufacture, and let $f(S)$ for $S \subseteq V$ be the manufacture costs of items in the subset S.
- Ex: V might be paint colors to produce: green, red, blue, yellow, white, etc.
- Producing green when you are already producing yellow and blue is probably cheaper than if you were only producing some other colors.

$$f(\text{green}, \text{blue}, \text{yellow}) - f(\text{blue}, \text{yellow}) <= f(\text{green}, \text{blue}) - f(\text{blue})$$

Supply Side Economies of Scale

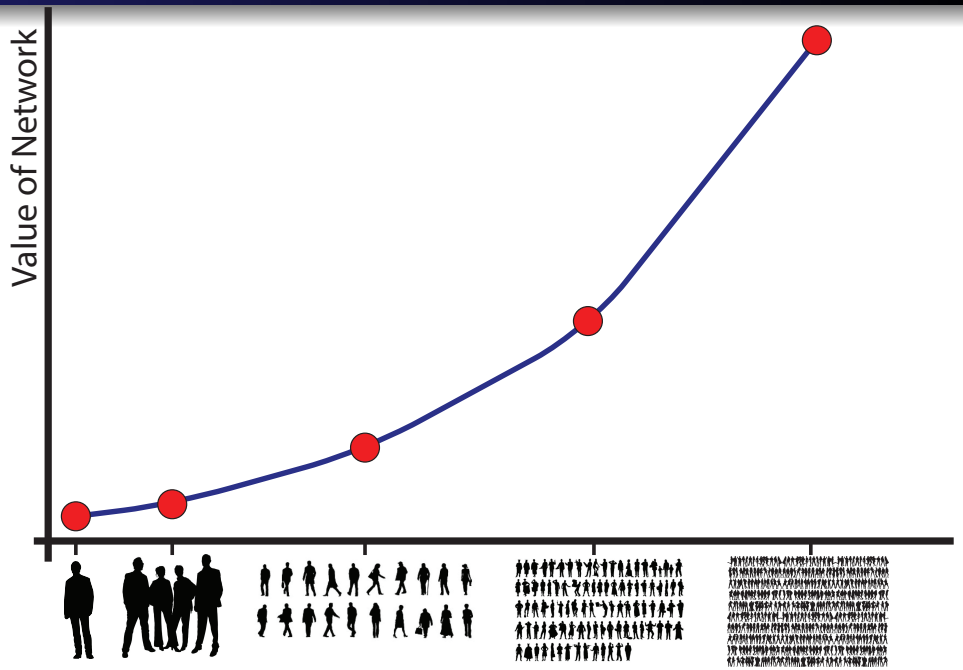
- What is a good model of the cost of manufacturing a set of items?
- Let V be a set of possible items to manufacture, and let $f(S)$ for $S \subseteq V$ be the manufacture costs of items in the subset S.
- Ex: V might be paint colors to produce: green, red, blue, yellow, white, etc.
- Producing green when you are already producing yellow and blue is probably cheaper than if you were only producing some other colors.

$$f(\text{green}, \text{blue}, \text{yellow}) - f(\text{blue}, \text{yellow}) <= f(\text{green}, \text{blue}) - f(\text{blue})$$

- So diminishing returns (a submodular function) would be a good model.

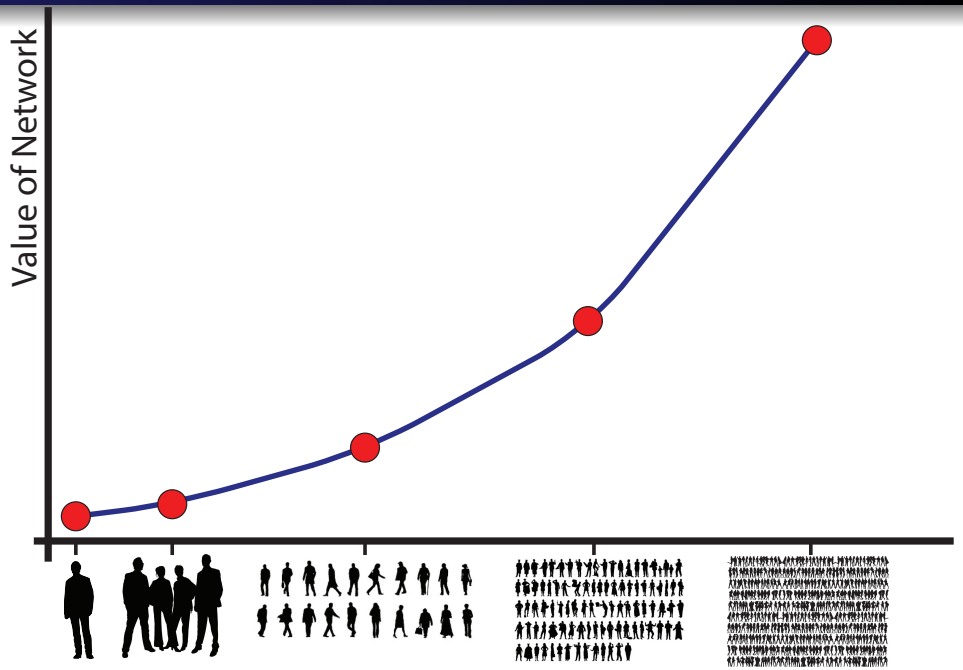
Demand side Economies of Scale: Network Externalities

- Value of a network to a user depends on the number of other users in that network. External use benefits internal use.



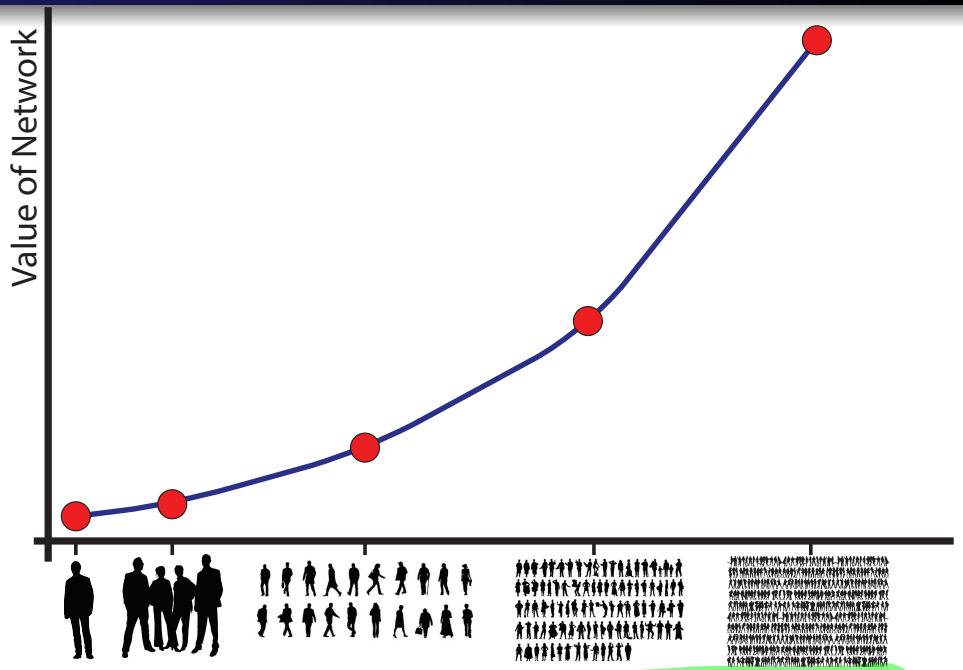
Demand side Economies of Scale: Network Externalities

- Value of a network to a user depends on the number of other users in that network. External use benefits internal use.
- Consumers derive positive incremental value when size of the market for that good increases.



Demand side Economies of Scale: Network Externalities

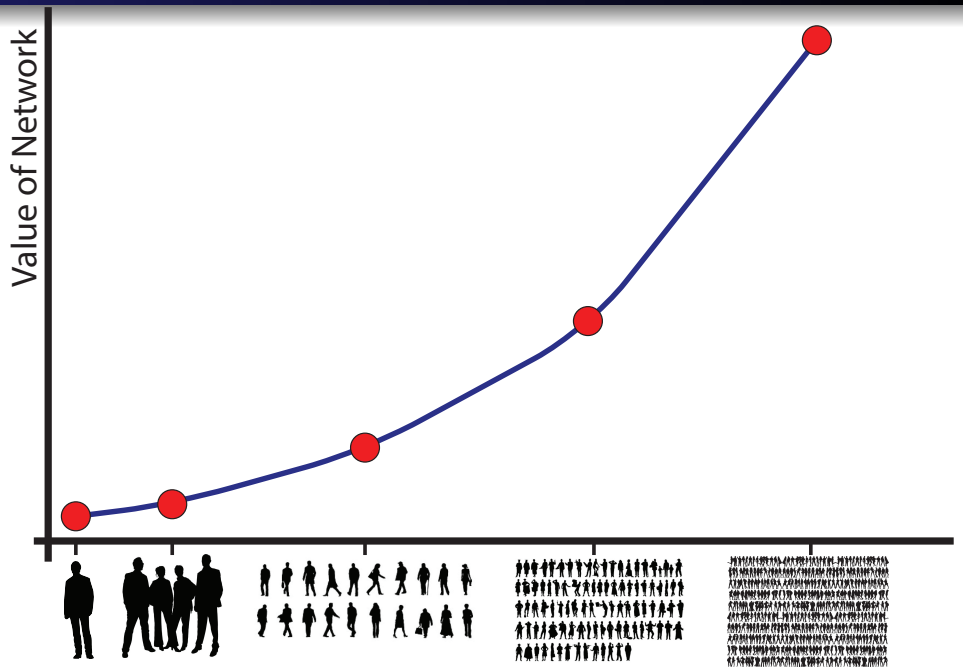
- Value of a network to a user depends on the number of other users in that network. External use benefits internal use.
- Consumers derive positive incremental value when size of the market for that good increases.



- Called network externalities (Katz & Shapiro 1986), or network effects and is a form of demand-side economies of scale

Demand side Economies of Scale: Network Externalities

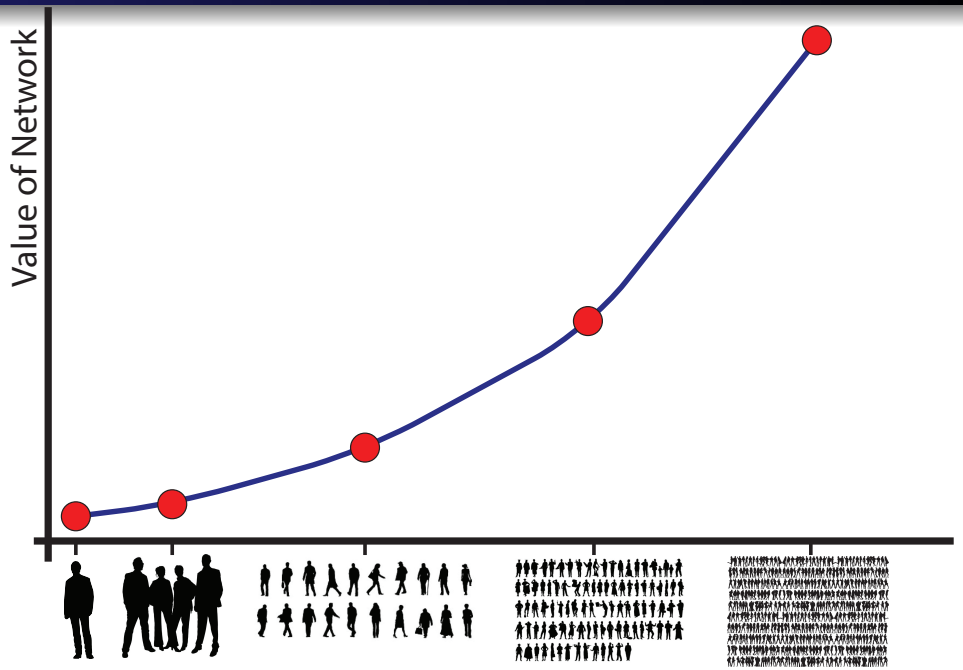
- Value of a network to a user depends on the number of other users in that network. External use benefits internal use.
- Consumers derive positive incremental value when size of the market for that good increases.



- Called network externalities (Katz & Shapiro 1986), or network effects and is a form of demand-side economies of scale
- Ex: durable goods (e.g., a car or phone), software (Facebook, smartphone apps), and technology-specific human capital investment (e.g., education in a skill), benefit depends on total user base.

Demand side Economies of Scale: Network Externalities

- Value of a network to a user depends on the number of other users in that network. External use benefits internal use.
- Consumers derive positive incremental value when size of the market for that good increases.



- Called network externalities (Katz & Shapiro 1986), or network effects and is a form of demand-side economies of scale
- Ex: durable goods (e.g., a car or phone), software (Facebook, smartphone apps), and technology-specific human capital investment (e.g., education in a skill), benefit depends on total user base.
- Let V be a set of goods, A a subset and $v \notin A$. Incremental gain of good $f(A+v) - f(A)$ gets larger as size of market A grows. This is known as a supermodular function.

Why is education a network externality?

- As more people become educated in a skill, more people innovate in that area.

Why is education a network externality?

- As more people become educated in a skill, more people innovate in that area.
- This helps to create more and better products.

Why is education a network externality?

- As more people become educated in a skill, more people innovate in that area.
- This helps to create more and better products.
- This creates demand for those products.

Why is education a network externality?

- As more people become educated in a skill, more people innovate in that area.
- This helps to create more and better products.
- This creates demand for those products.
- This thus creates more demand for more people who are educated in that skill and can produce, maintain, and improve those products.

Why is education a network externality?

- As more people become educated in a skill, more people innovate in that area.
- This helps to create more and better products.
- This creates demand for those products.
- This thus creates more demand for more people who are educated in that skill and can produce, maintain, and improve those products.
- Education itself, thus, creates a market for what you are being educated in.

Submodularity's utility in ML

- A model of a physical process:
  - When maximizing, submodularity naturally models: diversity, coverage, span, and information.
  - When minimizing, submodularity naturally models: cooperative costs, complexity, roughness, and irregularity.
  - vice-versa for supermodularity.
- A submodular function can act as a parameter for a machine learning strategy (active/semi-supervised learning, discrete divergence, structured sparse convex norms for use in regularization).
- Itself, as an object or function to learn, based on data.
- A surrogate or relaxation strategy for optimization or analysis
  - An alternate to factorization, decomposition, or sum-product based simplification (as one typically finds in a graphical model). I.e., a means towards tractable surrogates for graphical models.
  - Also, we can "relax" a problem to a submodular one where it can be efficiently solved and offer a bounded quality solution.
  - Non-submodular problems can be analyzed via submodularity.

Many different functions are submodular!

- We will see many applications of submodularity in machine learning.
- On next set of slides, we will state (without proof, for now) that many of the functions are submodular (or supermodular).
- In subsequent lectures, we will start showing how to prove submodularity.

Functions to Measure Diversity

Diversity is good, especially when it is high

- Quantitative measurement diversity in data science and ML. Goal of diversity: ensure small set properly represents the large.

Functions to Measure Diversity

Diversity is good, especially when it is high

- Quantitative measurement diversity in data science and ML. Goal of diversity: ensure small set properly represents the large.
- Web search: given ambiguous search term (e.g., "jaguar") with no other information, one wants articles more than just about cars.
  - Try google searching for words (e.g., "break") with many meanings (`http://muse.dillfrog.com/lists/ambiguous`), how well does google's diversity measure do?

Functions to Measure Diversity

Diversity is good, especially when it is high

- Quantitative measurement diversity in data science and ML. Goal of diversity: ensure small set properly represents the large.
- Web search: given ambiguous search term (e.g., "jaguar") with no other information, one wants articles more than just about cars.
  - Try google searching for words (e.g., "break") with many meanings (`http://muse.dillfrog.com/lists/ambiguous`), how well does google's diversity measure do?
  - Overall goal: user quickly finds informative, concise, accurate, relevant, comprehensive information $\Rightarrow$ diversity

Functions to Measure Diversity

Diversity is good, especially when it is high

- Quantitative measurement diversity in data science and ML. Goal of diversity: ensure small set properly represents the large.
- Web search: given ambiguous search term (e.g., "jaguar") with no other information, one wants articles more than just about cars.
  - Try google searching for words (e.g., "break") with many meanings (`http://muse.dillfrog.com/lists/ambiguous`), how well does google's diversity measure do?
  - Overall goal: user quickly finds informative, concise, accurate, relevant, comprehensive information $\Rightarrow$ diversity
- Given a set V of items, how do we choose a subset $S \subseteq V$ that is as diverse as possible, with perhaps constraints on S such as its size? Answer: submodular maximization.

Functions to Measure Diversity

Diversity is good, especially when it is high

- Quantitative measurement diversity in data science and ML. Goal of diversity: ensure small set properly represents the large.
- Web search: given ambiguous search term (e.g., "jaguar") with no other information, one wants articles more than just about cars.
  - Try google searching for words (e.g., "break") with many meanings (`http://muse.dillfrog.com/lists/ambiguous`), how well does google's diversity measure do?
  - Overall goal: user quickly finds informative, concise, accurate, relevant, comprehensive information $\Rightarrow$ diversity
- Given a set V of items, how do we choose a subset $S \subseteq V$ that is as diverse as possible, with perhaps constraints on S such as its size? Answer: submodular maximization.
- How do we choose the smallest set S that maintains a given degree of diversity? Constrained minimization (i.e., $\min |A|$ s.t. $f(A) \geq \alpha$).

Functions to Measure Diversity

Diversity is good, especially when it is high

- Quantitative measurement diversity in data science and ML. Goal of diversity: ensure small set properly represents the large.
- Web search: given ambiguous search term (e.g., "jaguar") with no other information, one wants articles more than just about cars.
  - Try google searching for words (e.g., "break") with many meanings (`http://muse.dillfrog.com/lists/ambiguous`), how well does google's diversity measure do?
  - Overall goal: user quickly finds informative, concise, accurate, relevant, comprehensive information $\Rightarrow$ <u>diversity</u>
- Given a set V of items, how do we choose a subset $S \subseteq V$ that is as diverse as possible, with perhaps constraints on S such as its size? Answer: submodular maximization.
- How do we choose the smallest set S that maintains a given degree of diversity? Constrained minimization (i.e., $\min |A|$ s.t. $f(A) \geq \alpha$).
- Random sample has probability of poorly representing normally underrepresented groups.

Image Summarization

10×10 image collection:

3 good summaries (diverse):

3 ok summaries:

3 poor summaries (redundant):

More Generally: Information and Summarization

- Let V be a set of (information containing) elements (e.g., words, sentences, documents, images, web pages, blogs, sensor readings, etc.).
- Each $v \in V$ is one element. Total amount of information in V is measure by a function $f(V)$, and any given subset $S \subseteq V$ measures the amount of information in S, given by $f(S)$.
- How informative is v in different sized contexts?
- A submodular function is likely a good model of information. Why?

$A \subseteq V$

$$\frac{f(v)}{f(v)} - \frac{f(A+v) - f(A)}{f(v)} = 1 - \frac{f(A+v) - f(A)}{f(v)}$$

$= \text{Knowledge Of}(v; A)$

Suppose $A \subseteq B$, if f is an information function,

$\text{Knowledge Of}(v; A) \leq \text{Knowledge Of}(v; B)$

true iff f is submodular

Variable/Feature Selection in Classification/Regression

- Let Y be a random variable we wish to accurately predict based on at most $n = |V|$ observed measurement variables $(X_1, X_2, \ldots, X_n) = X_V$ in a probability model $\Pr(Y, X_1, X_2, \ldots, X_n)$.

Variable/Feature Selection in Classification/Regression

- Let Y be a random variable we wish to accurately predict based on at most $n = |V|$ observed measurement variables $(X_1, X_2, \ldots, X_n) = X_V$ in a probability model $\Pr(Y, X_1, X_2, \ldots, X_n)$.
- Too costly to use all V variables. Goal: choose subset $A \subseteq V$ of variables within budget $|A| \leq k$. Predictions based on only $\Pr(y|x_A)$, hence subset A should retain accuracy.

Variable/Feature Selection in Classification/Regression

- Let Y be a random variable we wish to accurately predict based on at most $n = |V|$ observed measurement variables $(X_1, X_2, \ldots, X_n) = X_V$ in a probability model $\Pr(Y, X_1, X_2, \ldots, X_n)$.
- Too costly to use all V variables. Goal: choose subset $A \subseteq V$ of variables within budget $|A| \leq k$. Predictions based on only $\Pr(y|x_A)$, hence subset A should retain accuracy.
- The mutual information function $f(A) = I(Y; X_A)$ ("information gain") measures how well variables A can predicting Y (entropy reduction, reduction of uncertainty of Y).

Variable/Feature Selection in Classification/Regression

- Let Y be a random variable we wish to accurately predict based on at most $n = |V|$ observed measurement variables $(X_1, X_2, \ldots, X_n) = X_V$ in a probability model $\Pr(Y, X_1, X_2, \ldots, X_n)$.
- Too costly to use all V variables. Goal: choose subset $A \subseteq V$ of variables within budget $|A| \leq k$. Predictions based on only $\Pr(y|x_A)$, hence subset A should retain accuracy.
- The mutual information function $f(A) = I(Y; X_A)$ ("information gain") measures how well variables A can predicting Y (entropy reduction, reduction of uncertainty of Y).
- The mutual information function $f(A) = I(Y; X_A)$ is defined as:

$$I(Y; X_A) = \sum_{y, x_A} \Pr(y, x_A) \log \frac{\Pr(y, x_A)}{\Pr(y)\Pr(x_A)} = H(Y) - H(Y|X_A) \quad (1.18)$$

$$= H(X_A) - H(X_A|Y) = H(X_A) + H(Y) - H(X_A, Y) \quad (1.19)$$

Variable/Feature Selection in Classification/Regression

- Let Y be a random variable we wish to accurately predict based on at most $n = |V|$ observed measurement variables $(X_1, X_2, \ldots, X_n) = X_V$ in a probability model $\Pr(Y, X_1, X_2, \ldots, X_n)$.
- Too costly to use all V variables. Goal: choose subset $A \subseteq V$ of variables within budget $|A| \leq k$. Predictions based on only $\Pr(y|x_A)$, hence subset A should retain accuracy.
- The mutual information function $f(A) = I(Y; X_A)$ ("information gain") measures how well variables A can predicting Y (entropy reduction, reduction of uncertainty of Y).
- The mutual information function $f(A) = I(Y; X_A)$ is defined as:

$$I(Y; X_A) = \sum_{y, x_A} \Pr(y, x_A) \log \frac{\Pr(y, x_A)}{\Pr(y)\Pr(x_A)} = H(Y) - H(Y|X_A) \tag{1.18}$$

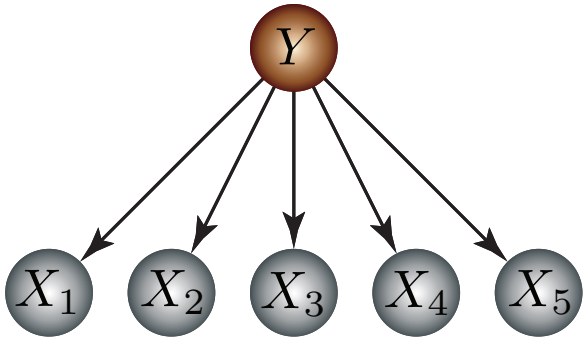
$$= H(X_A) - H(X_A|Y) = H(X_A) + H(Y) - H(X_A, Y) \tag{1.19}$$

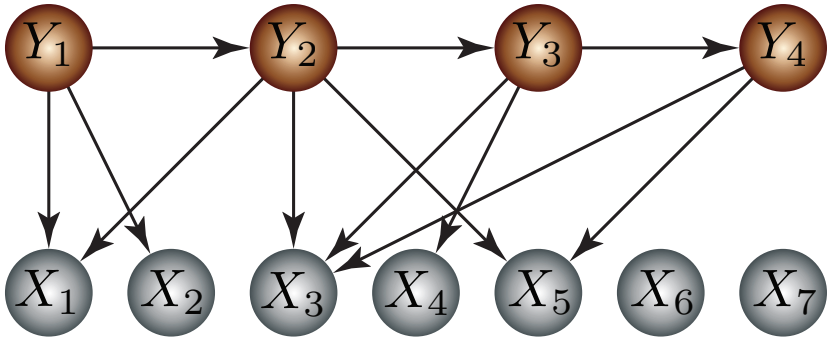
- Applicable in pattern recognition, also in sensor coverage problem, where Y is whatever question we wish to ask about environment.

Information Gain and Feature Selection

in Pattern Classification: Naïve Bayes

- Naïve Bayes property: $X_A \perp\!\!\!\perp X_B | Y$ for all A, B.

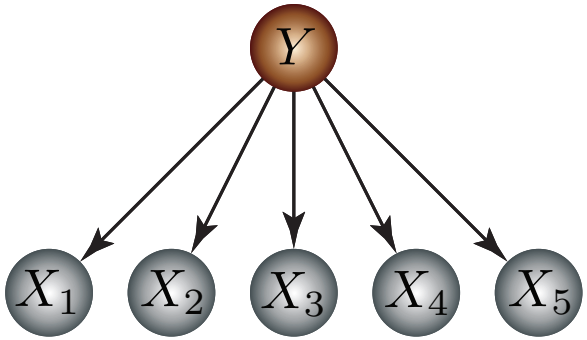


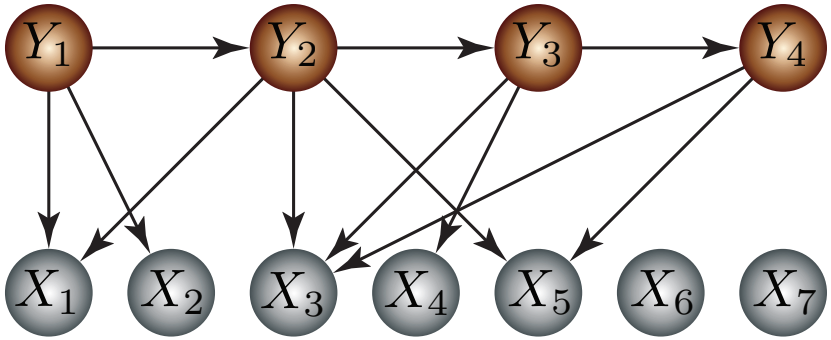


Information Gain and Feature Selection

in Pattern Classification: Naïve Bayes

- Naïve Bayes property: $X_A \perp\!\!\!\perp X_B | Y$ for all A, B.





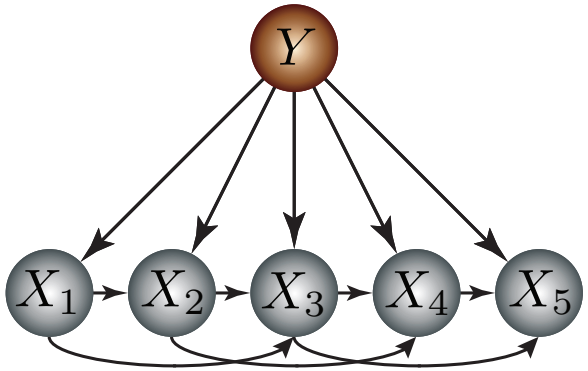
- When $X_A \perp\!\!\!\perp X_B | Y$ for all A, B (the Naïve Bayes assumption holds), then

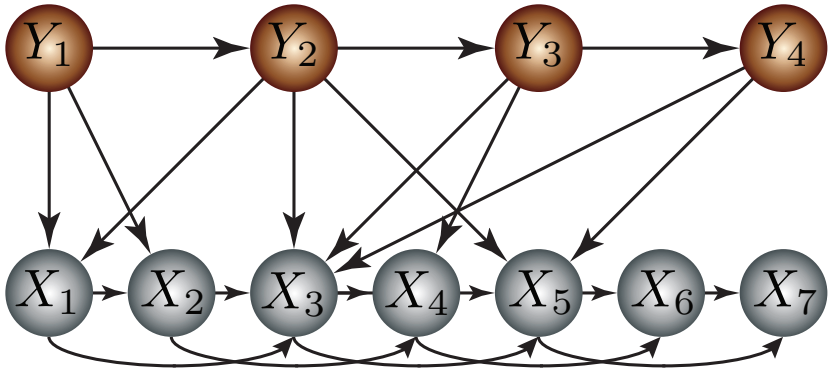
$$f(A) = I(Y; X_A) = H(X_A) - H(X_A|Y) = H(X_A) - \sum_{a \in A} H(X_a|Y) \tag{1.20}$$

is submodular (submodular minus modular).

Variable Selection in Pattern Classification

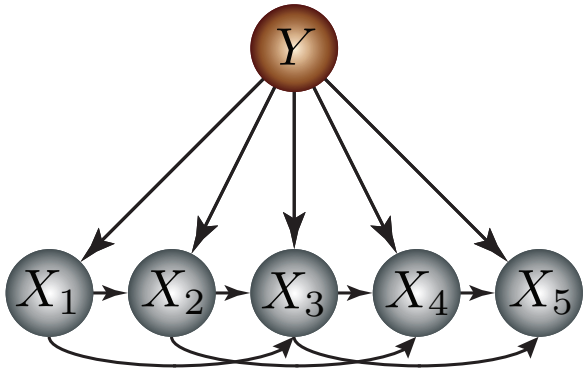
- Naïve Bayes property fails:

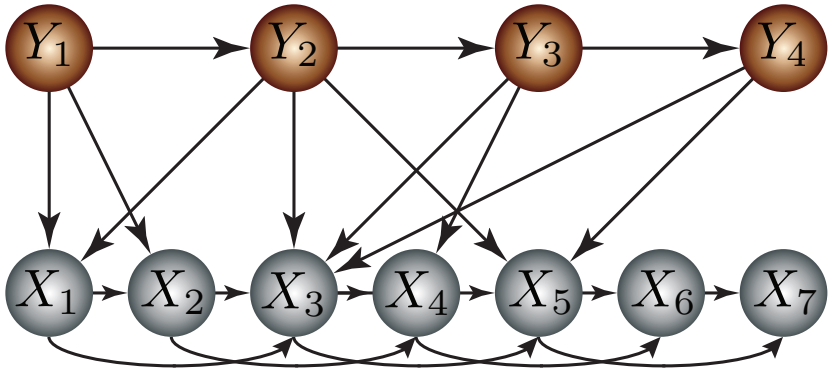




Variable Selection in Pattern Classification

- Naïve Bayes property fails:





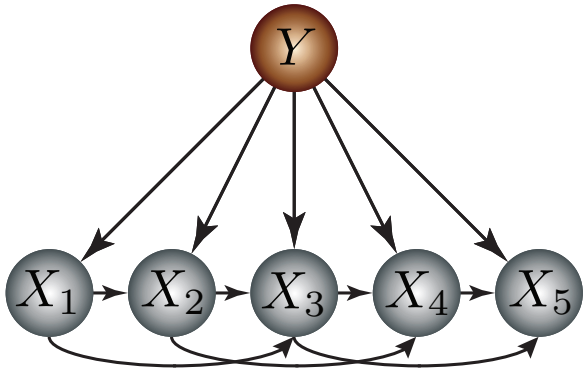
- $f(A)$ naturally expressed as a difference of two submodular functions

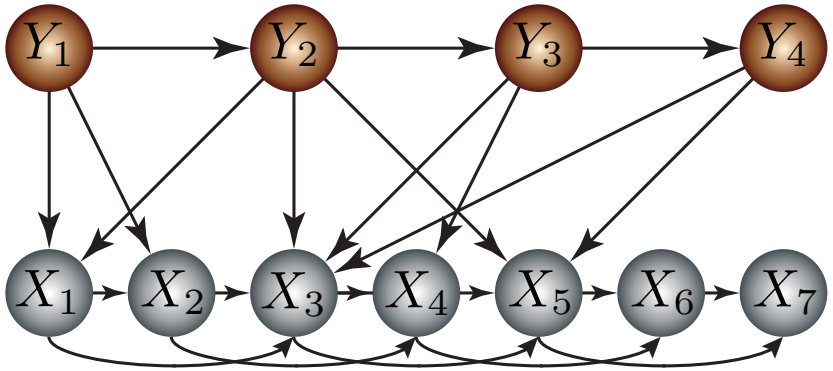
$$f(A) = I(Y; X_A) = H(X_A) - H(X_A|Y), \tag{1.21}$$

which is a DS (difference of submodular) function.

Variable Selection in Pattern Classification

- Naïve Bayes property fails:





- $f(A)$ naturally expressed as a difference of two submodular functions

$$f(A) = I(Y; X_A) = H(X_A) - H(X_A|Y), \tag{1.21}$$

which is a DS (difference of submodular) function.

- Alternatively, when Naïve Bayes assumption is false, we can make a submodular approximation (Peng-2005). E.g., functions of the form:

$$f(A) = \sum_{a \in A} I(X_a; Y) - \lambda \sum_{a, a' \in A} I(X_a; X_{a'}|Y) \tag{1.22}$$

where $\lambda \geq 0$ is a tradeoff constant.

Variable Selection: Linear Regression Case

- Next, let Z be continuous. Predictor is linear $\tilde{Z}_A = \sum_{i \in A} \alpha_i X_i$.

Variable Selection: Linear Regression Case

- Next, let Z be continuous. Predictor is linear $\tilde{Z}_A = \sum_{i \in A} \alpha_i X_i$.
- Error measure is the residual variance

$$R^2_{Z,A} = \frac{\mathsf{Var}(Z) - E[(Z - \tilde{Z}_A)^2]}{\mathsf{Var}(Z)} \tag{1.23}$$

Variable Selection: Linear Regression Case

- Next, let Z be continuous. Predictor is linear $\tilde{Z}_A = \sum_{i \in A} \alpha_i X_i$.
- Error measure is the residual variance

$$R^2_{Z,A} = \frac{\mathsf{Var}(Z) - E[(Z - \tilde{Z}_A)^2]}{\mathsf{Var}(Z)} \tag{1.23}$$

- $R^2_{Z,A}$'s minimizing parameters, for a given A, can be analytically computed. $R^2_{Z,A} = b_A{}^\intercal (C_A^{-1})^\intercal b_A$ whenever $\mathsf{Var}(Z) = 1$, where $b_i = \mathsf{Cov}(Z, X_i)$ and $C = E[(X - E[X])^\intercal (X - E[X])]$ is the covariance matrix.

Variable Selection: Linear Regression Case

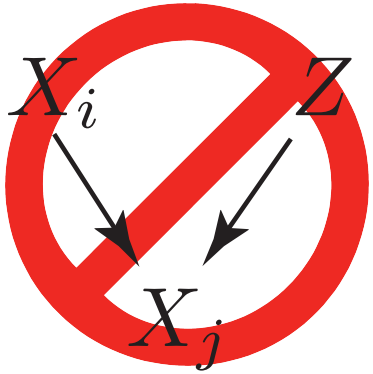
- Next, let Z be continuous. Predictor is linear $\tilde{Z}_A = \sum_{i \in A} \alpha_i X_i$.
- Error measure is the residual variance

$$R^2_{Z,A} = \frac{\mathsf{Var}(Z) - E[(Z - \tilde{Z}_A)^2]}{\mathsf{Var}(Z)} \tag{1.23}$$

- $R^2_{Z,A}$'s minimizing parameters, for a given A, can be analytically computed. $R^2_{Z,A} = b_A{}^\intercal (C_A^{-1})^\intercal b_A$ whenever $\mathsf{Var}(Z) = 1$, where $b_i = \mathsf{Cov}(Z, X_i)$ and $C = E[(X - E[X])^\intercal (X - E[X])]$ is the covariance matrix.
- When there are no "suppressor" variables (essentially, no v-structures that converge on X_j with parents X_i and Z), then

$$f(A) = R^2_{Z,A} = b_A{}^\intercal (C_A^{-1})^\intercal b_A \tag{1.24}$$

is a submodular function (so the greedy algorithm gives the $1 - 1/e$ guarantee). (Das&Kempe).



Data Subset Selection

- Suppose we are given a large data set $\mathcal{D} = \{x_i\}_{i=1}^n$ of n data items $V = \{v_1, v_2, \ldots, v_n\}$ and we wish to choose a subset $A \subset V$ of items that is good in some way (e.g., a summary).

Data Subset Selection

- Suppose we are given a large data set $\mathcal{D} = \{x_i\}_{i=1}^n$ of n data items $V = \{v_1, v_2, \ldots, v_n\}$ and we wish to choose a subset $A \subset V$ of items that is good in some way (e.g., a summary).
- Suppose moreover each data item $v \in V$ is described by a vector of non-negative scores for a set U of features (or "properties", or "concepts", etc.) of each data item.

Data Subset Selection

- Suppose we are given a large data set $\mathcal{D} = \{x_i\}_{i=1}^n$ of n data items $V = \{v_1, v_2, \ldots, v_n\}$ and we wish to choose a subset $A \subset V$ of items that is good in some way (e.g., a summary).
- Suppose moreover each data item $v \in V$ is described by a vector of non-negative scores for a set U of features (or "properties", or "concepts", etc.) of each data item.
- That is, for $u \in U$ and $v \in V$, let $m_u(v)$ represent the "degree of u-ness" possessed by data item v. Then $m_u \in \mathbb{R}_+^V$ for all $u \in U$.

Data Subset Selection

- Suppose we are given a large data set $\mathcal{D} = \{x_i\}_{i=1}^n$ of n data items $V = \{v_1, v_2, \ldots, v_n\}$ and we wish to choose a subset $A \subset V$ of items that is good in some way (e.g., a summary).
- Suppose moreover each data item $v \in V$ is described by a vector of non-negative scores for a set U of features (or "properties", or "concepts", etc.) of each data item.
- That is, for $u \in U$ and $v \in V$, let $m_u(v)$ represent the "degree of u-ness" possessed by data item v. Then $m_u \in \mathbb{R}_+^V$ for all $u \in U$.
- Example: U could be a set of colors, and for an image $v \in V$, $m_u(v)$ could represent the number of pixels that are of color u.

Data Subset Selection

- Suppose we are given a large data set $\mathcal{D} = \{x_i\}_{i=1}^n$ of n data items $V = \{v_1, v_2, \ldots, v_n\}$ and we wish to choose a subset $A \subset V$ of items that is good in some way (e.g., a summary).
- Suppose moreover each data item $v \in V$ is described by a vector of non-negative scores for a set U of features (or "properties", or "concepts", etc.) of each data item.
- That is, for $u \in U$ and $v \in V$, let $m_u(v)$ represent the "degree of u-ness" possessed by data item v. Then $m_u \in \mathbb{R}_+^V$ for all $u \in U$.
- Example: U could be a set of colors, and for an image $v \in V$, $m_u(v)$ could represent the number of pixels that are of color u.
- Example: U might be a set of textual features (e.g., ngrams), and $m_u(v)$ is the number of ngrams of type u in sentence v. E.g., if a document consists of the sentence

$$v = \text{"Whenever I go to New York City, I visit the New York City museum."}$$

then $m_{\text{'the'}}(v) = 1$ while $m_{\text{'New York City'}}(v) = 2$.

Data Subset Selection

- For $X \subseteq V$, define $m_u(X) = \sum_{x \in X} m_u(x)$, so $m_u(X)$ is a modular function representing the "degree of u-ness" in subset X.

Data Subset Selection

- For $X \subseteq V$, define $m_u(X) = \sum_{x \in X} m_u(x)$, so $m_u(X)$ is a modular function representing the "degree of u-ness" in subset X.
- Since $m_u(X)$ is modular, it does not have a diminishing returns property. I.e., as we add to X, the degree of u-ness grows additively.

Data Subset Selection

- For $X \subseteq V$, define $m_u(X) = \sum_{x \in X} m_u(x)$, so $m_u(X)$ is a modular function representing the "degree of u-ness" in subset X.
- Since $m_u(X)$ is modular, it does not have a diminishing returns property. I.e., as we add to X, the degree of u-ness grows additively.
- With g non-decreasing concave, $g(m_u(X))$ grows subadditively (if we add v to a context A with less u-ness, the u-ness benefit is more than if we add v to a context $B \supseteq A$ having more u-ness). That is

$$g(m_u(A+v)) - g(m_u(A)) \geq g(m_u(B+v)) - g(m_u(B)) \tag{1.25}$$

Data Subset Selection

- For $X \subseteq V$, define $m_u(X) = \sum_{x \in X} m_u(x)$, so $m_u(X)$ is a modular function representing the "degree of u-ness" in subset X.
- Since $m_u(X)$ is modular, it does not have a diminishing returns property. I.e., as we add to X, the degree of u-ness grows additively.
- With g non-decreasing concave, $g(m_u(X))$ grows subadditively (if we add v to a context A with less u-ness, the u-ness benefit is more than if we add v to a context $B \supseteq A$ having more u-ness). That is

$$g(m_u(A+v)) - g(m_u(A)) \geq g(m_u(B+v)) - g(m_u(B)) \tag{1.25}$$

- Consider the following class of feature functions $f : 2^V \to \mathbb{R}_+$

$$f(X) = \sum_{u \in U} \alpha_u g_u(m_u(X)) \tag{1.26}$$

where g_u is a non-decreasing concave, and $\alpha_u \geq 0$ is a feature importance weight. Thus, f is submodular.

Data Subset Selection

- For $X \subseteq V$, define $m_u(X) = \sum_{x \in X} m_u(x)$, so $m_u(X)$ is a modular function representing the "degree of u-ness" in subset X.
- Since $m_u(X)$ is modular, it does not have a diminishing returns property. I.e., as we add to X, the degree of u-ness grows additively.
- With g non-decreasing concave, $g(m_u(X))$ grows subadditively (if we add v to a context A with less u-ness, the u-ness benefit is more than if we add v to a context $B \supseteq A$ having more u-ness). That is

$$g(m_u(A+v)) - g(m_u(A)) \geq g(m_u(B+v)) - g(m_u(B)) \tag{1.25}$$

- Consider the following class of feature functions $f : 2^V \to \mathbb{R}_+$

$$f(X) = \sum_{u \in U} \alpha_u g_u(m_u(X)) \tag{1.26}$$

where g_u is a non-decreasing concave, and $\alpha_u \geq 0$ is a feature importance weight. Thus, f is submodular.
- $f(X)$ measures X's ability to represent set of features U as measured by $m_u(X)$, with diminishing returns function g, and importance weights α_u.

Data Subset Selection, KL-divergence

- Let $p = \{p_u\}_{u \in U}$ be a desired probability distribution over features (i.e., $\sum_u p_u = 1$ and $p_u \geq 0$ for all $u \in U$).

Data Subset Selection, KL-divergence

- Let $p = \{p_u\}_{u \in U}$ be a desired probability distribution over features (i.e., $\sum_u p_u = 1$ and $p_u \geq 0$ for all $u \in U$).
- Next, normalize the modular weights for each feature:

$$0 \leq \bar{m}_u(X) \triangleq \frac{m_u(X)}{\sum_{u' \in U} m_{u'}(X)} = \frac{m_u(X)}{m(X)} \leq 1 \tag{1.27}$$

where $m(X) \triangleq \sum_{u' \in U} m_{u'}(X)$.

Data Subset Selection, KL-divergence

- Let $p = \{p_u\}_{u \in U}$ be a desired probability distribution over features (i.e., $\sum_u p_u = 1$ and $p_u \geq 0$ for all $u \in U$).
- Next, normalize the modular weights for each feature:

$$0 \leq \bar{m}_u(X) \triangleq \frac{m_u(X)}{\sum_{u' \in U} m_{u'}(X)} = \frac{m_u(X)}{m(X)} \leq 1 \tag{1.27}$$

  where $m(X) \triangleq \sum_{u' \in U} m_{u'}(X)$.
- Then $\bar{m}_u(X)$ can also be seen as a distribution over features U since $\bar{m}_u(X) \geq 0$ and $\sum_{u \in U} \bar{m}_u(X) = 1$ for any $X \subseteq V$.

Data Subset Selection, KL-divergence

- Let $p = \{p_u\}_{u \in U}$ be a desired probability distribution over features (i.e., $\sum_u p_u = 1$ and $p_u \geq 0$ for all $u \in U$).
- Next, normalize the modular weights for each feature:

$$0 \leq \bar{m}_u(X) \triangleq \frac{m_u(X)}{\sum_{u' \in U} m_{u'}(X)} = \frac{m_u(X)}{m(X)} \leq 1 \tag{1.27}$$

where $m(X) \triangleq \sum_{u' \in U} m_{u'}(X)$.

- Then $\bar{m}_u(X)$ can also be seen as a distribution over features U since $\bar{m}_u(X) \geq 0$ and $\sum_{u \in U} \bar{m}_u(X) = 1$ for any $X \subseteq V$.
- Consider the KL-divergence between these two distributions:

$$D(p || \{\bar{m}_u(X)\}_{u \in U}) = \sum_{u \in U} p_u \log p_u - \sum_{u \in U} p_u \log(\bar{m}_u(X)) \tag{1.28}$$

$$= \sum_{u \in U} p_u \log p_u - \sum_{u \in U} p_u \log(m_u(X)) + \log(m(X))$$

$$= -H(p) + \log m(X) - \sum_{u \in U} p_u \log(m_u(X)) \tag{1.29}$$

Data Subset Selection, KL-divergence

- The objective once again, treating entropy $H(p)$ as a constant,

$$D(p||\{\bar{m}_u(X)\}) = \text{const.} + \log m(X) - \sum_{u \in U} p_u \log(m_u(X)) \quad (1.30)$$

Data Subset Selection, KL-divergence

- The objective once again, treating entropy $H(p)$ as a constant,

$$D(p||\{\bar{m}_u(X)\}) = \text{const.} + \log m(X) - \sum_{u \in U} p_u \log(m_u(X)) \quad (1.30)$$

- But seen as a function of X, both $\log m(X)$ and $\sum_{u \in U} p_u \log m_u(X)$ are submodular functions.

Data Subset Selection, KL-divergence

- The objective once again, treating entropy $H(p)$ as a constant,

$$D(p||\{\bar{m}_u(X)\}) = \text{const.} + \log m(X) - \sum_{u \in U} p_u \log(m_u(X)) \quad (1.30)$$

- But seen as a function of X, both $\log m(X)$ and $\sum_{u \in U} p_u \log m_u(X)$ are submodular functions.
- Hence the KL-divergence, seen as a function of X, i.e., $f(X) = D(p||\{\bar{m}_u(X)\})$ is quite naturally represented as a difference of submodular functions.

Data Subset Selection, KL-divergence

- The objective once again, treating entropy $H(p)$ as a constant,

$$D(p||\{\bar{m}_u(X)\}) = \text{const.} + \log m(X) - \sum_{u \in U} p_u \log(m_u(X)) \quad (1.30)$$

- But seen as a function of X, both $\log m(X)$ and $\sum_{u \in U} p_u \log m_u(X)$ are submodular functions.
- Hence the KL-divergence, seen as a function of X, i.e., $f(X) = D(p||\{\bar{m}_u(X)\})$ is quite naturally represented as a difference of submodular functions.
- Alternatively, if we define (Shinohara, 2014)

$$g(X) \triangleq \log m(X) - D(p||\{\bar{m}_u(X)\}) = \sum_{u \in U} p_u \log(m_u(X)) \quad (1.31)$$

we have a submodular function g that represents a combination of its quantity of X via its features (i.e., $\log m(X)$) and its feature distribution closeness to some distribution p (i.e., $D(p||\{\bar{m}_u(X)\})$).

Other examples as diversity models

- Sensor placement

Other examples as diversity models

- Sensor placement
- Social networks and influential nodes

Other examples as diversity models

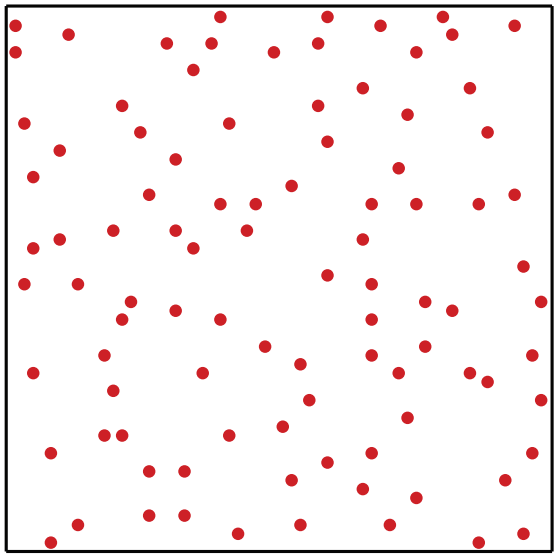
- Sensor placement
- Social networks and influential nodes
- Viral marketing, information cascades, diffusion networks

Determinantal Point Processes (DPPs)

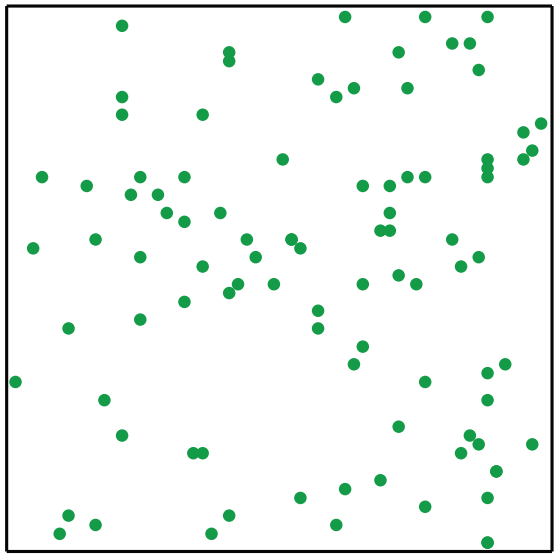
- Sometimes we wish not only to valuate subsets $A \subseteq V$ but to induce probability distributions over all subsets.

Determinantal Point Processes (DPPs)

- Sometimes we wish not only to valuate subsets $A \subseteq V$ but to induce probability distributions over all subsets.
- We may wish to prefer samples where elements of A are diverse (i.e., given a sample A, for $a, b \in A$, we prefer a and b to be different).



DPP

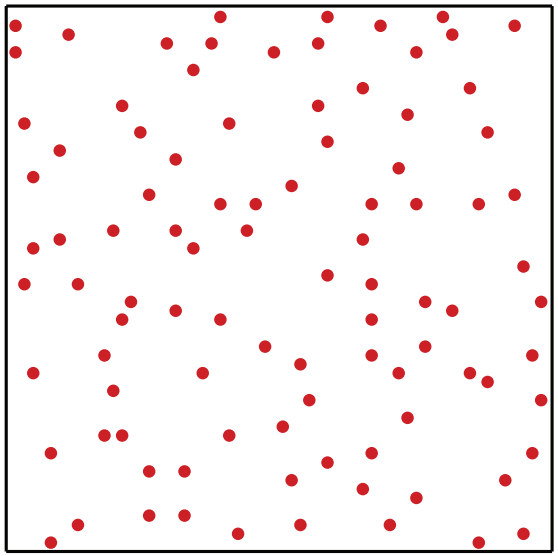


Independent

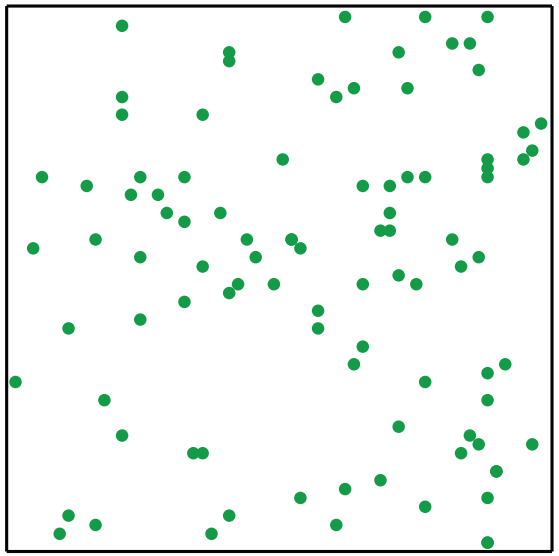
(Kulesza, Gillenwater, & Taskar, 2011)

Determinantal Point Processes (DPPs)

- Sometimes we wish not only to valuate subsets $A \subseteq V$ but to induce probability distributions over all subsets.
- We may wish to prefer samples where elements of A are diverse (i.e., given a sample A, for $a, b \in A$, we prefer a and b to be different).



DPP



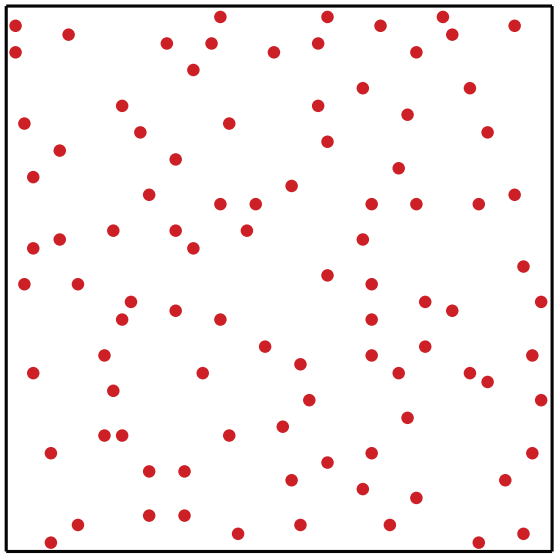
Independent

(Kulesza, Gillenwater, & Taskar, 2011)

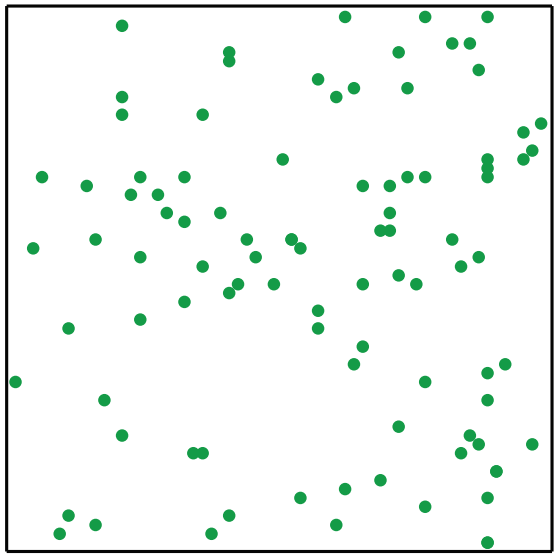
- A Determinantal point processes (DPPs) is a probability distribution over subsets A of V where the "energy" function is submodular.

Determinantal Point Processes (DPPs)

- Sometimes we wish not only to valuate subsets $A \subseteq V$ but to induce probability distributions over all subsets.
- We may wish to prefer samples where elements of A are diverse (i.e., given a sample A, for $a, b \in A$, we prefer a and b to be different).



DPP



Independent

(Kulesza, Gillenwater, & Taskar, 2011)

- A Determinantal point processes (DPPs) is a probability distribution over subsets A of V where the "energy" function is submodular.
- More "diverse" or "complex" samples are given higher probability.

DPPs and log-submodular probability distributions

- Given binary vectors $x, y \in \{0,1\}^V$, $y \leq x$ if $y(v) \leq x(v), \forall v \in V$.

DPPs and log-submodular probability distributions

- Given binary vectors $x, y \in \{0,1\}^V$, $y \leq x$ if $y(v) \leq x(v), \forall v \in V$.
- Given a positive-definite $n \times n$ matrix M, a subset $X \subseteq V$, let M_X be $|X| \times |X|$ principle submatrix, rows/columns specified by $X \subseteq V$.

DPPs and log-submodular probability distributions

- Given binary vectors $x, y \in \{0,1\}^V$, $y \leq x$ if $y(v) \leq x(v), \forall v \in V$.
- Given a positive-definite $n \times n$ matrix M, a subset $X \subseteq V$, let M_X be $|X| \times |X|$ principle submatrix, rows/columns specified by $X \subseteq V$.
- A Determinantal Point Process (DPP) is a distribution of the form:

$$\Pr(\mathbf{X} = x) = \frac{|M_{X(x)}|}{|M+I|} = \exp\left(\log\left(\frac{|M_{X(x)}|}{|M+I|}\right)\right) \propto \det(M_{X(x)}) \tag{1.32}$$

where I is $n \times n$ identity matrix, and $\mathbf{X} \in \{0,1\}^V$ is a random vector.

DPPs and log-submodular probability distributions

- Given binary vectors $x, y \in \{0,1\}^V$, $y \leq x$ if $y(v) \leq x(v), \forall v \in V$.
- Given a positive-definite $n \times n$ matrix M, a subset $X \subseteq V$, let M_X be $|X| \times |X|$ principle submatrix, rows/columns specified by $X \subseteq V$.
- A Determinantal Point Process (DPP) is a distribution of the form:

$$\Pr(\mathbf{X} = x) = \frac{|M_{X(x)}|}{|M+I|} = \exp\left(\log\left(\frac{|M_{X(x)}|}{|M+I|}\right)\right) \propto \det(M_{X(x)}) \tag{1.32}$$

  where I is $n \times n$ identity matrix, and $\mathbf{X} \in \{0,1\}^V$ is a random vector.
- Equivalently, defining K as $K = M(M+I)^{-1}$, we have:

$$\sum_{x \in \{0,1\}^V : x \geq y} \Pr(\mathbf{X} = x) = \Pr(\mathbf{X} \geq y) = \exp\left(\log\left(|K_{Y(y)}|\right)\right) \tag{1.33}$$

DPPs and log-submodular probability distributions

- Given binary vectors $x, y \in \{0,1\}^V$, $y \leq x$ if $y(v) \leq x(v), \forall v \in V$.
- Given a positive-definite $n \times n$ matrix M, a subset $X \subseteq V$, let M_X be $|X| \times |X|$ principle submatrix, rows/columns specified by $X \subseteq V$.
- A Determinantal Point Process (DPP) is a distribution of the form:

$$\Pr(\mathbf{X} = x) = \frac{|M_{X(x)}|}{|M+I|} = \exp\left(\log\left(\frac{|M_{X(x)}|}{|M+I|}\right)\right) \propto \det(M_{X(x)}) \tag{1.32}$$

  where I is $n \times n$ identity matrix, and $\mathbf{X} \in \{0,1\}^V$ is a random vector.
- Equivalently, defining K as $K = M(M+I)^{-1}$, we have:

$$\sum_{x \in \{0,1\}^V : x \geq y} \Pr(\mathbf{X} = x) = \Pr(\mathbf{X} \geq y) = \exp\left(\log\left(|K_{Y(y)}|\right)\right) \tag{1.33}$$

- Given positive definite matrix M, function $f : 2^V \to \mathbb{R}$ with $f(A) = \log |M_A|$ (the logdet function) is submodular.

DPPs and log-submodular probability distributions

- Given binary vectors $x, y \in \{0,1\}^V$, $y \leq x$ if $y(v) \leq x(v), \forall v \in V$.
- Given a positive-definite $n \times n$ matrix M, a subset $X \subseteq V$, let M_X be $|X| \times |X|$ principle submatrix, rows/columns specified by $X \subseteq V$.
- A Determinantal Point Process (DPP) is a distribution of the form:

$$\Pr(\mathbf{X} = x) = \frac{|M_{X(x)}|}{|M+I|} = \exp\left(\log\left(\frac{|M_{X(x)}|}{|M+I|}\right)\right) \propto \det(M_{X(x)}) \tag{1.32}$$

  where I is $n \times n$ identity matrix, and $\mathbf{X} \in \{0,1\}^V$ is a random vector.
- Equivalently, defining K as $K = M(M+I)^{-1}$, we have:

$$\sum_{x \in \{0,1\}^V : x \geq y} \Pr(\mathbf{X} = x) = \Pr(\mathbf{X} \geq y) = \exp\left(\log\left(|K_{Y(y)}|\right)\right) \tag{1.33}$$

- Given positive definite matrix M, function $f : 2^V \to \mathbb{R}$ with $f(A) = \log |M_A|$ (the logdet function) is submodular.
- Therefore, a DPP is a log-submodular probability distribution.

Graphical Models and fast MAP Inference

- Given distribution that factors w.r.t. a graph:

$$p(x) = \frac{1}{Z} \exp(-E(x)) \tag{1.34}$$

where $E(x) = \sum_{c \in \mathcal{C}} E_c(x_c)$ and $\mathcal{C}$ are cliques of graph $G = (V, \mathcal{E})$.

Graphical Models and fast MAP Inference

- Given distribution that factors w.r.t. a graph:

$$p(x) = \frac{1}{Z} \exp(-E(x)) \tag{1.34}$$

where $E(x) = \sum_{c \in \mathcal{C}} E_c(x_c)$ and $\mathcal{C}$ are cliques of graph $G = (V, \mathcal{E})$.

- MAP inference problem is important in ML: compute

$$x^* \in \operatorname*{argmax}_{x \in \{0,1\}^V} p(x) \tag{1.35}$$

Graphical Models and fast MAP Inference

- Given distribution that factors w.r.t. a graph:

$$p(x) = \frac{1}{Z} \exp(-E(x)) \tag{1.34}$$

where $E(x) = \sum_{c \in \mathcal{C}} E_c(x_c)$ and $\mathcal{C}$ are cliques of graph $G = (V, \mathcal{E})$.

- MAP inference problem is important in ML: compute

$$x^* \in \operatorname*{argmax}_{x \in \{0,1\}^V} p(x) \tag{1.35}$$

- Easy when G a tree, exponential in k (tree-width of G) in general.

Graphical Models and fast MAP Inference

- Given distribution that factors w.r.t. a graph:

$$p(x) = \frac{1}{Z}\exp(-E(x)) \tag{1.34}$$

where $E(x) = \sum_{c\in\mathcal{C}} E_c(x_c)$ and $\mathcal{C}$ are cliques of graph $G = (V, \mathcal{E})$.

- MAP inference problem is important in ML: compute

$$x^* \in \operatorname*{argmax}_{x\in\{0,1\}^V} p(x) \tag{1.35}$$

- Easy when G a tree, exponential in k (tree-width of G) in general.
- Even worse, NP-hard to find the tree-width.

Graphical Models and fast MAP Inference

- Given distribution that factors w.r.t. a graph:

$$p(x) = \frac{1}{Z}\exp(-E(x)) \tag{1.34}$$

where $E(x) = \sum_{c\in\mathcal{C}} E_c(x_c)$ and $\mathcal{C}$ are cliques of graph $G = (V, \mathcal{E})$.

- MAP inference problem is important in ML: compute

$$x^* \in \operatorname*{argmax}_{x\in\{0,1\}^V} p(x) \tag{1.35}$$

- Easy when G a tree, exponential in k (tree-width of G) in general.
- Even worse, NP-hard to find the tree-width.
- Tree-width can be large even when degree is small (e.g., regular grid graphs have low-degree but $\Omega(\sqrt{n})$ tree-width).

Graphical Models and fast MAP Inference

- Given distribution that factors w.r.t. a graph:

$$p(x) = \frac{1}{Z}\exp(-E(x)) \tag{1.34}$$

where $E(x) = \sum_{c\in\mathcal{C}} E_c(x_c)$ and $\mathcal{C}$ are cliques of graph $G = (V, \mathcal{E})$.

- MAP inference problem is important in ML: compute

$$x^* \in \operatorname*{argmax}_{x\in\{0,1\}^V} p(x) \tag{1.35}$$

- Easy when G a tree, exponential in k (tree-width of G) in general.
- Even worse, NP-hard to find the tree-width.
- Tree-width can be large even when degree is small (e.g., regular grid graphs have low-degree but $\Omega(\sqrt{n})$ tree-width).
- Many approximate inference strategies utilize additional factorization assumptions (e.g., mean-field, variational inference, expectation propagation, etc).

Graphical Models and fast MAP Inference

- Given distribution that factors w.r.t. a graph:

$$p(x) = \frac{1}{Z}\exp(-E(x)) \tag{1.34}$$

where $E(x) = \sum_{c \in \mathcal{C}} E_c(x_c)$ and $\mathcal{C}$ are cliques of graph $G = (V, \mathcal{E})$.

- MAP inference problem is important in ML: compute

$$x^* \in \operatorname*{argmax}_{x \in \{0,1\}^V} p(x) \tag{1.35}$$

- Easy when G a tree, exponential in k (tree-width of G) in general.
- Even worse, NP-hard to find the tree-width.
- Tree-width can be large even when degree is small (e.g., regular grid graphs have low-degree but $\Omega(\sqrt{n})$ tree-width).
- Many approximate inference strategies utilize additional factorization assumptions (e.g., mean-field, variational inference, expectation propagation, etc).
- Can we do exact MAP inference in polynomial time regardless of the tree-width, without even knowing the tree-width?

Order-two (edge) graphical models

- Given G let $p \in \mathcal{F}(G, \mathcal{M}^{(\mathsf{f})})$ such that we can write the global energy $E(x)$ as a sum of unary and pairwise potentials:

$$E(x) = \sum_{v \in V(G)} e_v(x_v) + \sum_{(i,j) \in E(G)} e_{ij}(x_i, x_j) \tag{1.36}$$

Order-two (edge) graphical models

- Given G let $p \in \mathcal{F}(G, \mathcal{M}^{(\mathsf{f})})$ such that we can write the global energy $E(x)$ as a sum of unary and pairwise potentials:

$$E(x) = \sum_{v \in V(G)} e_v(x_v) + \sum_{(i,j) \in E(G)} e_{ij}(x_i, x_j) \tag{1.36}$$

- $e_v(x_v)$ and $e_{ij}(x_i, x_j)$ are like local energy potentials.

Order-two (edge) graphical models

- Given G let $p \in \mathcal{F}(G, \mathcal{M}^{(\mathsf{f})})$ such that we can write the global energy $E(x)$ as a sum of unary and pairwise potentials:

$$E(x) = \sum_{v \in V(G)} e_v(x_v) + \sum_{(i,j) \in E(G)} e_{ij}(x_i, x_j) \tag{1.36}$$

- $e_v(x_v)$ and $e_{ij}(x_i, x_j)$ are like local energy potentials.
- Since $\log p(x) = -E(x) + \text{const.}$, the smaller $e_v(x_v)$ or $e_{ij}(x_i, x_j)$ become, the higher the probability becomes.

Order-two (edge) graphical models

- Given G let $p \in \mathcal{F}(G, \mathcal{M}^{(\mathsf{f})})$ such that we can write the global energy $E(x)$ as a sum of unary and pairwise potentials:

$$E(x) = \sum_{v \in V(G)} e_v(x_v) + \sum_{(i,j) \in E(G)} e_{ij}(x_i, x_j) \tag{1.36}$$

- $e_v(x_v)$ and $e_{ij}(x_i, x_j)$ are like local energy potentials.
- Since $\log p(x) = -E(x) + \text{const.}$, the smaller $e_v(x_v)$ or $e_{ij}(x_i, x_j)$ become, the higher the probability becomes.
- Further, say that $\mathsf{D}_{X_v} = \{0, 1\}$ (binary), so we have binary random vectors distributed according to $p(x)$.

Order-two (edge) graphical models

- Given G let $p \in \mathcal{F}(G, \mathcal{M}^{(\mathsf{f})})$ such that we can write the global energy $E(x)$ as a sum of unary and pairwise potentials:

$$E(x) = \sum_{v \in V(G)} e_v(x_v) + \sum_{(i,j) \in E(G)} e_{ij}(x_i, x_j) \tag{1.36}$$

- $e_v(x_v)$ and $e_{ij}(x_i, x_j)$ are like local energy potentials.
- Since $\log p(x) = -E(x) + \text{const.}$, the smaller $e_v(x_v)$ or $e_{ij}(x_i, x_j)$ become, the higher the probability becomes.
- Further, say that $\mathsf{D}_{X_v} = \{0, 1\}$ (binary), so we have binary random vectors distributed according to $p(x)$.
- Thus, $x \in \{0, 1\}^V$, and finding MPE solution is setting some of the variables to 0 and some to 1, i.e.,

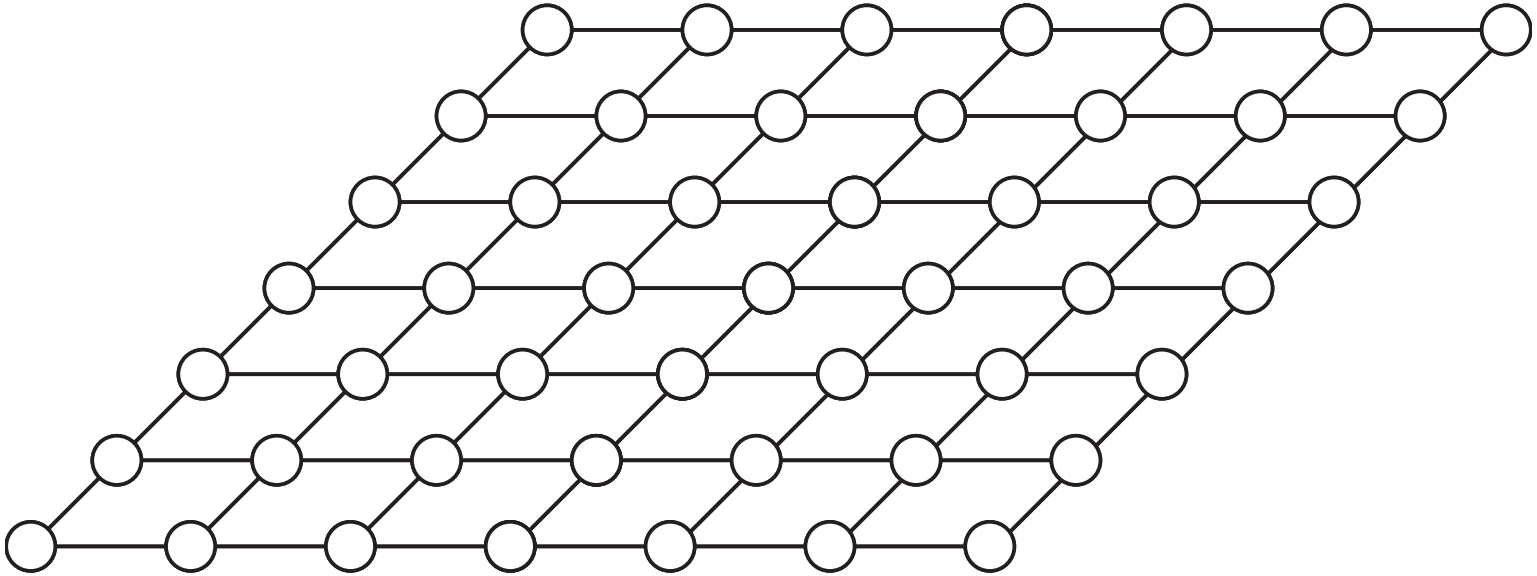
$$\min_{x \in \{0,1\}^V} E(x) \tag{1.37}$$

MRF example

Markov random field

$$\log p(x) \propto \sum_{v \in V(G)} e_v(x_v) + \sum_{(i,j) \in E(G)} e_{ij}(x_i, x_j) \tag{1.38}$$

When G is a 2D grid graph, we have

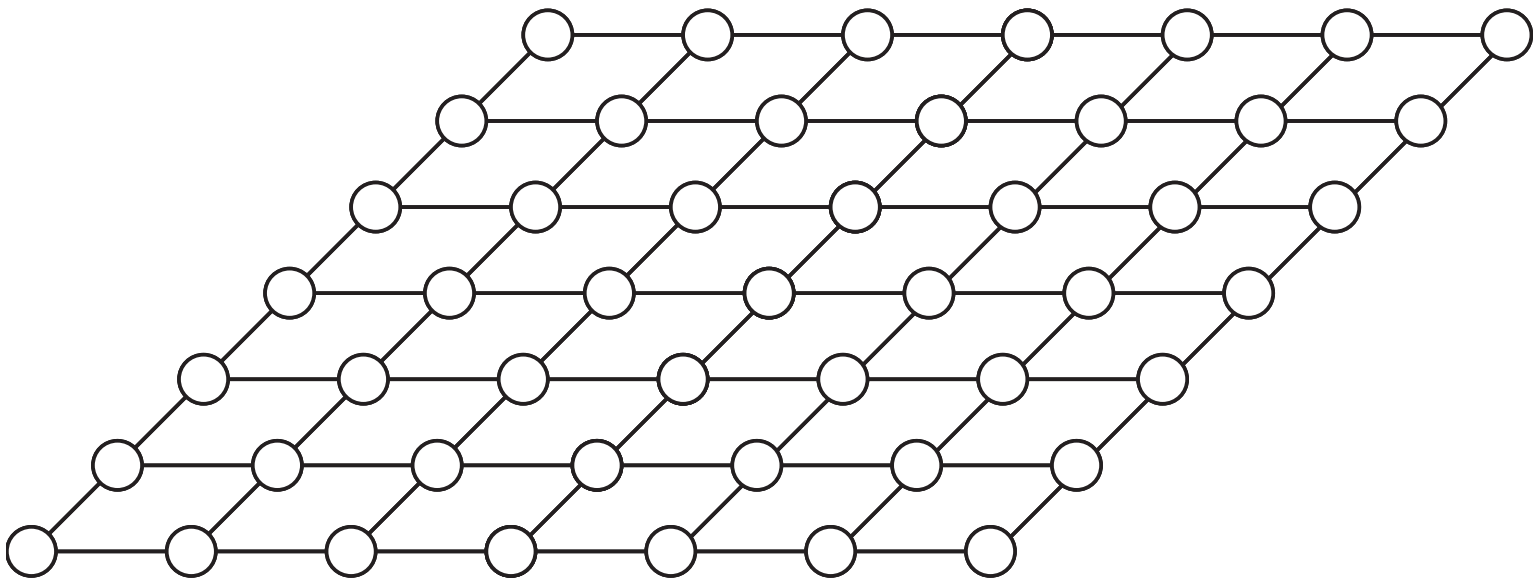


Create an auxiliary graph

- We can create auxiliary graph G_a that involves two new "terminal" nodes s and t and all of the original "non-terminal" nodes $v \in V(G)$.
- The non-terminal nodes represent the original random variables $x_v, v \in V$.
- Starting with the original grid-graph amongst the vertices $v \in V$, we connect each of s and t to all of the original nodes.
- I.e., we form $G_a = (V \cup \{s, t\}, E + \cup_{v \in V}((s, v) \cup (v, t)))$.

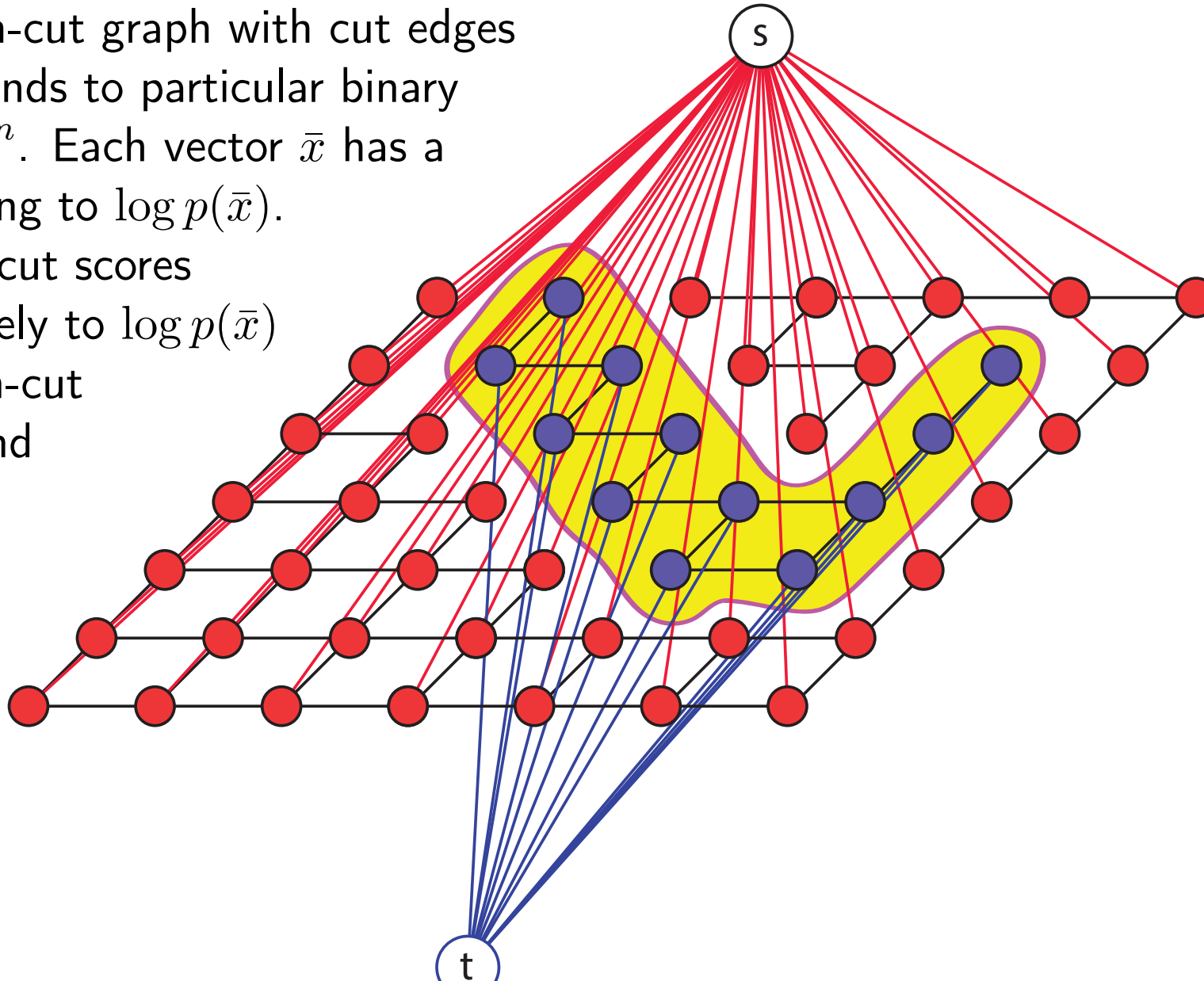
Transformation from graphical model to auxiliary graph

Original 2D-grid graphical model G and energy function $E(x) = \sum_{v \in V(G)} e_v(x_v) + \sum_{(i,j) \in E(G)} e_{ij}(x_i, x_j)$ needing to be minimized over $x \in \{0,1\}^V$. Recall, tree-width is $O(\sqrt{|V|})$.



Transformation from graphical model to auxiliary graph

Augmented graph-cut graph with cut edges removed corresponds to particular binary vector $\bar{x} \in \{0,1\}^n$. Each vector $\bar{x}$ has a score corresponding to $\log p(\bar{x})$. When can graph cut scores correspond precisely to $\log p(\bar{x})$ in a way that min-cut algorithms can find minimum of energy $E(x)$?



Setting of the weights in the auxiliary cut graph

- Any graph cut corresponds to a vector $\bar{x} \in \{0,1\}^n$.
- If weights of all edges, except those involving terminals s and t, are non-negative, graph cut computable in polynomial time via max-flow (many algorithms, e.g., Edmonds&Karp $O(nm^2)$ or $O(n^2 m \log(nC))$; Goldberg&Tarjan $O(nm \log(n^2/m))$, see Schrijver, page 161).
- If weights are set correctly in the cut graph, and if edge functions e_{ij} satisfy certain properties, then graph-cut score corresponding to $\bar{x}$ can be made equivalent to $E(x) = \log p(\bar{x}) +$ const..
- Hence, poly time graph cut, can find the optimal MPE assignment, regardless of the graphical model's tree-width!
- In general, finding MPE is an NP-hard optimization problem.

Submodular potentials

submodularity is what allows graph cut to find exact solution

- Edge functions must be submodular (in the binary case, equivalent to "associative", "attractive", "regular", "Potts", or "ferromagnetic"): for all $(i, j) \in E(G)$, must have:

$$e_{ij}(0,1) + e_{ij}(1,0) \geq e_{ij}(1,1) + e_{ij}(0,0) \tag{1.46}$$

Submodular potentials

submodularity is what allows graph cut to find exact solution

- Edge functions must be submodular (in the binary case, equivalent to "associative", "attractive", "regular", "Potts", or "ferromagnetic"): for all $(i, j) \in E(G)$, must have:

$$e_{ij}(0,1) + e_{ij}(1,0) \geq e_{ij}(1,1) + e_{ij}(0,0) \tag{1.46}$$

- This means: on average, preservation is preferred over change.

Submodular potentials

submodularity is what allows graph cut to find exact solution

- Edge functions must be submodular (in the binary case, equivalent to "associative", "attractive", "regular", "Potts", or "ferromagnetic"): for all $(i, j) \in E(G)$, must have:

$$e_{ij}(0,1) + e_{ij}(1,0) \geq e_{ij}(1,1) + e_{ij}(0,0) \tag{1.46}$$

- This means: on average, preservation is preferred over change.
- As a set function, this is the same as:

$$f(X) = \sum_{\{i,j\} \in \mathcal{E}(G)} f_{i,j}(X \cap \{i,j\}) \tag{1.47}$$

which is submodular if each of the $f_{i,j}$'s are submodular!

Submodular potentials

submodularity is what allows graph cut to find exact solution

- Edge functions must be submodular (in the binary case, equivalent to "associative", "attractive", "regular", "Potts", or "ferromagnetic"): for all $(i, j) \in E(G)$, must have:

$$e_{ij}(0,1) + e_{ij}(1,0) \geq e_{ij}(1,1) + e_{ij}(0,0) \tag{1.46}$$

- This means: on average, preservation is preferred over change.
- As a set function, this is the same as:

$$f(X) = \sum_{\{i,j\} \in \mathcal{E}(G)} f_{i,j}(X \cap \{i,j\}) \tag{1.47}$$

  which is submodular if each of the $f_{i,j}$'s are submodular!
- A special case of more general submodular functions – unconstrained submodular function minimization is solvable in polytime.

Submodular Generalized Dependence

- there is a notion of "independence" , i.e., $A \perp\!\!\!\perp B$:

$$f(A \cup B) = f(A) + f(B), \tag{1.48}$$

Submodular Generalized Dependence

- there is a notion of "independence" , i.e., $A \perp\!\!\!\perp B$:

$$f(A \cup B) = f(A) + f(B), \tag{1.48}$$

- and a notion of "conditional independence" , i.e., $A \perp\!\!\!\perp B | C$:

$$f(A \cup B \cup C) + f(C) = f(A \cup C) + f(B \cup C) \tag{1.49}$$

Submodular Generalized Dependence

- there is a notion of "independence" , i.e., $A \perp\!\!\!\perp B$:

$$f(A \cup B) = f(A) + f(B), \tag{1.48}$$

- and a notion of "conditional independence" , i.e., $A \perp\!\!\!\perp B | C$:

$$f(A \cup B \cup C) + f(C) = f(A \cup C) + f(B \cup C) \tag{1.49}$$

- and a notion of "dependence" (conditioning reduces valuation):

$$f(A|B) \triangleq f(A \cup B) - f(B) < f(A), \tag{1.50}$$

Submodular Generalized Dependence

- there is a notion of "independence" , i.e., $A \perp\!\!\!\perp B$:

$$f(A \cup B) = f(A) + f(B), \tag{1.48}$$

- and a notion of "conditional independence" , i.e., $A \perp\!\!\!\perp B | C$:

$$f(A \cup B \cup C) + f(C) = f(A \cup C) + f(B \cup C) \tag{1.49}$$

- and a notion of "dependence" (conditioning reduces valuation):

$$f(A|B) \triangleq f(A \cup B) - f(B) < f(A), \tag{1.50}$$

- and a notion of "conditional mutual information"

$$I_f(A; B|C) \triangleq f(A \cup C) + f(B \cup C) - f(A \cup B \cup C) - f(C) \geq 0$$

Submodular Generalized Dependence

- there is a notion of "independence" , i.e., $A \perp\!\!\!\perp B$:

$$f(A \cup B) = f(A) + f(B), \tag{1.48}$$

- and a notion of "conditional independence" , i.e., $A \perp\!\!\!\perp B | C$:

$$f(A \cup B \cup C) + f(C) = f(A \cup C) + f(B \cup C) \tag{1.49}$$

- and a notion of "dependence" (conditioning reduces valuation):

$$f(A|B) \triangleq f(A \cup B) - f(B) < f(A), \tag{1.50}$$

- and a notion of "conditional mutual information"

$$I_f(A; B|C) \triangleq f(A \cup C) + f(B \cup C) - f(A \cup B \cup C) - f(C) \geq 0$$

- and two notions of "information amongst a collection of sets":

$$I_f(S_1; S_2; \ldots; S_k) = \sum_{i=1}^{k} f(S_k) - f(S_1 \cup S_2 \cup \cdots \cup S_k) \tag{1.51}$$

$$I'_f(S_1; S_2; \ldots; S_k) = \sum_{A \subseteq \{1,2,\ldots,k\}} (-1)^{|A|+1} f(\bigcup_{j \in A} S_j) \tag{1.52}$$

Submodular Parameterized Clustering

- Given a submodular function $f : 2^V \to \mathbb{R}$, form the combinatorial dependence function $I_f(A; B) = f(A) + f(B) - f(A \cup B)$.

Submodular Parameterized Clustering

- Given a submodular function $f : 2^V \to \mathbb{R}$, form the combinatorial dependence function $I_f(A; B) = f(A) + f(B) - f(A \cup B)$.
- Consider clustering algorithm: First find partition $A_1^* \in \operatorname{argmin}_{A \subseteq V} I_f(A; V \setminus A)$ and $A_2^* = V \setminus A_1^*$.

Submodular Parameterized Clustering

- Given a submodular function $f : 2^V \to \mathbb{R}$, form the combinatorial dependence function $I_f(A; B) = f(A) + f(B) - f(A \cup B)$.
- Consider clustering algorithm: First find partition $A_1^* \in \operatorname{argmin}_{A \subseteq V} I_f(A; V \setminus A)$ and $A_2^* = V \setminus A_1^*$.
- Then partition the partitions: $A_{11}^* \in \operatorname{argmin}_{A \subseteq A_1^*} I_f(A; A_1^* \setminus A)$, $A_{12}^* = A_1^* \setminus A_{11}^*$, and $A_{21}^* \in \operatorname{argmin}_{A \subseteq A_2^*} I_f(A; A_2^* \setminus A)$, etc.

Submodular Parameterized Clustering

- Given a submodular function $f : 2^V \to \mathbb{R}$, form the combinatorial dependence function $I_f(A; B) = f(A) + f(B) - f(A \cup B)$.
- Consider clustering algorithm: First find partition $A_1^* \in \operatorname{argmin}_{A \subseteq V} I_f(A; V \setminus A)$ and $A_2^* = V \setminus A_1^*$.
- Then partition the partitions: $A_{11}^* \in \operatorname{argmin}_{A \subseteq A_1^*} I_f(A; A_1^* \setminus A)$, $A_{12}^* = A_1^* \setminus A_{11}^*$, and $A_{21}^* \in \operatorname{argmin}_{A \subseteq A_2^*} I_f(A; A_2^* \setminus A)$, etc.
- Recursively partition the partitions, we end up with a partition $V = V_1 \cup V_2 \cup \cdots \cup V_k$ that clusters the data.

Submodular Parameterized Clustering

- Given a submodular function $f : 2^V \to \mathbb{R}$, form the combinatorial dependence function $I_f(A; B) = f(A) + f(B) - f(A \cup B)$.
- Consider clustering algorithm: First find partition $A_1^* \in \operatorname{argmin}_{A \subseteq V} I_f(A; V \setminus A)$ and $A_2^* = V \setminus A_1^*$.
- Then partition the partitions: $A_{11}^* \in \operatorname{argmin}_{A \subseteq A_1^*} I_f(A; A_1^* \setminus A)$, $A_{12}^* = A_1^* \setminus A_{11}^*$, and $A_{21}^* \in \operatorname{argmin}_{A \subseteq A_2^*} I_f(A; A_2^* \setminus A)$, etc.
- Recursively partition the partitions, we end up with a partition $V = V_1 \cup V_2 \cup \cdots \cup V_k$ that clusters the data.
- Each minimization can be done using Queyranne's algorithm (alternatively can construct a Gomory-Hu tree). This gives a partition no worse than factor 2 away from optimal partition. (Narasimhan&Bilmes, 2007).

Submodular Parameterized Clustering

- Given a submodular function $f : 2^V \to \mathbb{R}$, form the combinatorial dependence function $I_f(A; B) = f(A) + f(B) - f(A \cup B)$.
- Consider clustering algorithm: First find partition $A_1^* \in \operatorname{argmin}_{A \subseteq V} I_f(A; V \setminus A)$ and $A_2^* = V \setminus A_1^*$.
- Then partition the partitions: $A_{11}^* \in \operatorname{argmin}_{A \subseteq A_1^*} I_f(A; A_1^* \setminus A)$, $A_{12}^* = A_1^* \setminus A_{11}^*$, and $A_{21}^* \in \operatorname{argmin}_{A \subseteq A_2^*} I_f(A; A_2^* \setminus A)$, etc.
- Recursively partition the partitions, we end up with a partition $V = V_1 \cup V_2 \cup \cdots \cup V_k$ that clusters the data.
- Each minimization can be done using Queyranne's algorithm (alternatively can construct a Gomory-Hu tree). This gives a partition no worse than factor 2 away from optimal partition. (Narasimhan&Bilmes, 2007).
- Hence, family of clustering algorithms parameterized by f.