Fractional 0-1 programming and Submodularity

Shaoning Han

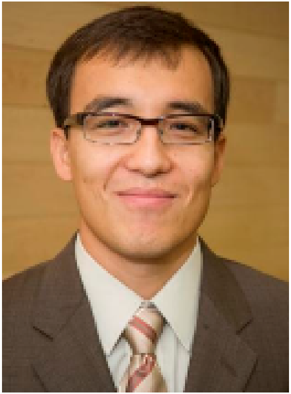
Department of Industrial & Systems Engineering
University of Southern California

shaoning@usc.edu

October, 2021

Collaborators

Andres Gomez
University of Southern California

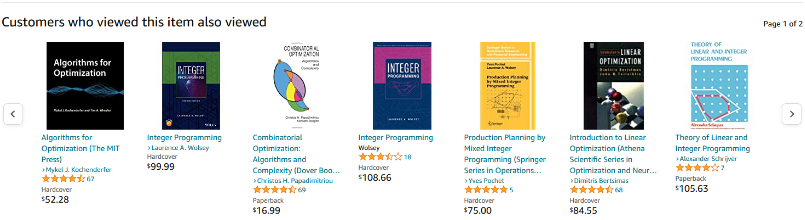


Oleg Prokopyev
University of Pittsburgh

One motivating example

Assortment optimization

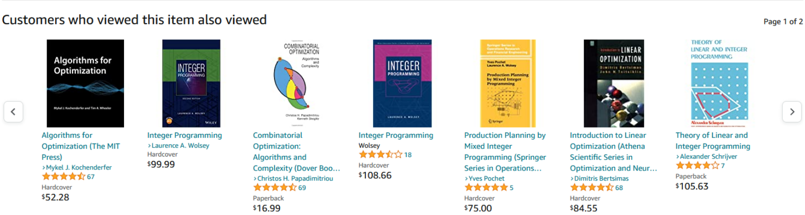
Search result of “Integer programming”



One motivating example

Assortment optimization

Search result of "Integer programming"



How does a company decide which products to display?

Motivating example: assortment optimization problem

Goal: choose an assortment of products to maximize profits under the MMNL model

- $[n]$: set of products offered to customers
- $[m]$: set of market segments
- v: preference weights
- r: revenue rates
- x: $x_i = 1$ iff $i \in S$

Motivating example: assortment optimization problem

Goal: choose an assortment of products to maximize profits under the MMNL model

$$q(i, S; v) = \frac{v_i}{v_0 + \sum_{j \in S} v_j}$$

- $[n]$: set of products offered to customers
- $[m]$: set of market segments
- v: preference weights
- r: revenue rates
- x: $x_i = 1$ iff $i \in S$

Motivating example: assortment optimization problem

Goal: choose an assortment of products to maximize profits under the MMNL model

$$r_i q(i, S; v) = \frac{r_i v_i}{v_0 + \sum_{j \in S} v_j}$$

- $[n]$: set of products offered to customers
- $[m]$: set of market segments
- v: preference weights
- r: revenue rates
- x: $x_i = 1$ iff $i \in S$

Motivating example: assortment optimization problem

Goal: choose an assortment of products to maximize profits under the MMNL model

$$r(S; v) = \sum_{i \in S} r_i q(i, S; v) = \frac{\sum_{i \in S} r_i v_i}{v_0 + \sum_{j \in S} v_j}$$

- $[n]$: set of products offered to customers
- $[m]$: set of market segments
- v: preference weights
- r: revenue rates
- x: $x_i = 1$ iff $i \in S$

Motivating example: assortment optimization problem

Goal: choose an assortment of products to maximize profits under the MMNL model

$$\mathbb{E}_v[r(S;v)] \iff$$

$$\sum_{k\in[m]} p_k r(S;v^k) = \sum_{k\in[m]} p_k \sum_{i\in S} r_i q(i,S;v^k) = \sum_{k\in[m]} p_k \frac{\sum_{i\in S} r_i v_{ik}}{v_{0k} + \sum_{j\in S} v_{jk}}$$

- $[n]$: set of products offered to customers
- $[m]$: set of market segments
- v: preference weights
- r: revenue rates
- x: $x_i = 1$ iff $i \in S$

Motivating example: assortment optimization problem

Goal: choose an assortment of products to maximize profits under the MMNL model

$$\max_{S \in \mathcal{F}} \mathbb{E}_v [r(S; v)] \iff$$

$$\max_{S \in \mathcal{F}} \sum_{k \in [m]} p_k r(S; v^k) = \sum_{k \in [m]} p_k \sum_{i \in S} r_i q(i, S; v^k) = \sum_{k \in [m]} p_k \frac{\sum_{i \in S} r_i v_{ik}}{v_{0k} + \sum_{j \in S} v_{jk}}$$

- $[n]$: set of products offered to customers
- $[m]$: set of market segments
- v: preference weights
- r: revenue rates
- x: $x_i = 1$ iff $i \in S$

Motivating example: assortment optimization problem

Goal: choose an assortment of products to maximize profits under the MMNL model

$$\max_{S \in \mathcal{F}} \mathbb{E}_v[r(S;v)] \iff$$

$$\max_{S \in \mathcal{F}} \sum_{k \in [m]} p_k r(S;v^k) = \sum_{k \in [m]} p_k \sum_{i \in S} r_i q(i,S;v^k) = \sum_{k \in [m]} p_k \frac{\sum_{i \in S} r_i v_{ik}}{v_{0k} + \sum_{j \in S} v_{jk}}$$

$$= \max_{x \in \mathcal{F}} \sum_{k \in [m]} \frac{\sum_{i \in [n]} r_i v_{ik} x_i}{v_{0k} + \sum_{j \in [n]} v_{jk} x_j}$$

- $[n]$: set of products offered to customers
- $[m]$: set of market segments
- v: preference weights
- r: revenue rates
- x: $x_i = 1$ iff $i \in S$

Introduction

Multiple-ratio fractional 0-1 program

$$\max_{x \in \mathcal{F}} \sum_{k \in [m]} \frac{\sum_{i \in [n]} a_{ki} x_i}{b_{k0} + \sum_{i \in [n]} b_{ki} x_i} \tag{1}$$

where $a > 0, b > 0$ and $\mathcal{F} \subseteq \{0,1\}^n$ is the feasible region

Introduction

Multiple-ratio fractional 0-1 program

$$\max_{x\in\mathcal{F}} \sum_{k\in[m]} \frac{\sum_{i\in[n]} a_{ki}x_i}{b_{k0} + \sum_{i\in[n]} b_{ki}x_i} \tag{1}$$

where $a > 0, b > 0$ and $\mathcal{F} \subseteq \{0,1\}^n$ is the feasible region

- Assortment optimization problem (Méndez-Díaz et al. 2014)

Introduction

Multiple-ratio fractional 0-1 program

$$\max_{x \in \mathcal{F}} \sum_{k \in [m]} \frac{\sum_{i \in [n]} a_{ki} x_i}{b_{k0} + \sum_{i \in [n]} b_{ki} x_i} \tag{1}$$

where $a > 0, b > 0$ and $\mathcal{F} \subseteq \{0,1\}^n$ is the feasible region

- Assortment optimization problem (Méndez-Díaz et al. 2014)
- Facility location problem (Tawarmalani et al. 2002)

Introduction

Multiple-ratio fractional 0-1 program

$$\max_{x \in \mathcal{F}} \sum_{k \in [m]} \frac{\sum_{i \in [n]} a_{ki} x_i}{b_{k0} + \sum_{i \in [n]} b_{ki} x_i} \tag{1}$$

where $a > 0, b > 0$ and $\mathcal{F} \subseteq \{0, 1\}^n$ is the feasible region

- Assortment optimization problem (Méndez-Díaz et al. 2014)
- Facility location problem (Tawarmalani et al. 2002)
- Minimum fractional spanning tree problem (Ursulenko et al. 2013)
- ...

Introduction

One-to-one correspondence

$$x = \mathbb{I}_S := \begin{cases} 1 & \text{if } i \in S \\ 0 & \text{o.w.} \end{cases} \leftrightarrow S = \{i : x_i = 1\}$$

Introduction

One-to-one correspondence

$$x = \mathbb{I}_S := \begin{cases} 1 & \text{if } i \in S \\ 0 & \text{o.w.} \end{cases} \leftrightarrow S = \{i : x_i = 1\}$$

Set-function optimization problem

$$\max_{S \in \mathcal{F}} \sum_{k \in [m]} \frac{\sum_{i \in S} a_{ki}}{b_{k0} + \sum_{i \in S} b_{ki}}, \tag{2}$$

where $a > 0, b > 0$ and $\mathcal{F} \subseteq [n] := \{1, 2, \ldots, n\}$

Introduction

One-to-one correspondence

$$x = \mathbb{I}_S := \begin{cases} 1 & \text{if } i \in S \\ 0 & \text{o.w.} \end{cases} \leftrightarrow S = \{i : x_i = 1\}$$

Set-function optimization problem

$$\max_{S \in \mathcal{F}} \sum_{k \in [m]} \frac{\sum_{i \in S} a_{ki}}{b_{k0} + \sum_{i \in S} b_{ki}}, \tag{2}$$

where $a > 0, b > 0$ and $\mathcal{F} \subseteq [n] := \{1, 2, \ldots, n\}$

Assumption: $\mathcal{F}$ is *downward closed*, i.e. $S \in \mathcal{F} \Rightarrow T \in \mathcal{F} \; \forall \, T \subseteq S$

Introduction

One-to-one correspondence

$$x = \mathbb{I}_S := \begin{cases} 1 & \text{if } i \in S \\ 0 & \text{o.w.} \end{cases} \leftrightarrow S = \{i : x_i = 1\}$$

Set-function optimization problem

$$\max_{S \in \mathcal{F}} \sum_{k \in [m]} \frac{\sum_{i \in S} a_{ki}}{b_{k0} + \sum_{i \in S} b_{ki}}, \tag{2}$$

where $a > 0, b > 0$ and $\mathcal{F} \subseteq [n] := \{1, 2, \ldots, n\}$

Assumption: $\mathcal{F}$ is *downward closed*, i.e. $S \in \mathcal{F} \Rightarrow T \in \mathcal{F} \; \forall \, T \subseteq S$

- Unconstrained problem: $\mathcal{F} = 2^{[n]}$

Introduction

One-to-one correspondence

$$x = \mathbb{I}_S := \begin{cases} 1 & \text{if } i \in S \\ 0 & \text{o.w.} \end{cases} \leftrightarrow S = \{i : x_i = 1\}$$

Set-function optimization problem

$$\max_{S \in \mathcal{F}} \sum_{k \in [m]} \frac{\sum_{i \in S} a_{ki}}{b_{k0} + \sum_{i \in S} b_{ki}}, \tag{2}$$

where $a > 0, b > 0$ and $\mathcal{F} \subseteq [n] := \{1, 2, \ldots, n\}$

Assumption: $\mathcal{F}$ is *downward closed*, i.e. $S \in \mathcal{F} \Rightarrow T \in \mathcal{F}\ \forall\ T \subseteq S$

- Unconstrained problem: $\mathcal{F} = 2^{[n]}$
- Cardinality constraint: $\mathcal{F} = \{S \subseteq [n] : |S| \leq p\}$

Introduction

One-to-one correspondence

$$x = \mathbb{I}_S := \begin{cases} 1 & \text{if } i \in S \\ 0 & \text{o.w.} \end{cases} \leftrightarrow S = \{i : x_i = 1\}$$

Set-function optimization problem

$$\max_{S \in \mathcal{F}} \sum_{k \in [m]} \frac{\sum_{i \in S} a_{ki}}{b_{k0} + \sum_{i \in S} b_{ki}}, \tag{2}$$

where $a > 0, b > 0$ and $\mathcal{F} \subseteq [n] := \{1, 2, \ldots, n\}$

Assumption: $\mathcal{F}$ is *downward closed*, i.e. $S \in \mathcal{F} \Rightarrow T \in \mathcal{F} \; \forall \, T \subseteq S$

- Unconstrained problem: $\mathcal{F} = 2^{[n]}$
- Cardinality constraint: $\mathcal{F} = \{S \subseteq [n] : |S| \leq p\}$
- Capacity constraint: $\mathcal{F} = \{S \subseteq [n] : \sum_{i \in S} w_i \leq c\}$, $w > 0, c > 0$

Complexity

How hard is it?

$$\max_{x \in \mathcal{F}} \sum_{k \in [m]} \frac{\sum_{i \in [n]} a_{ki} x_i}{b_{k0} + \sum_{i \in [n]} b_{ki} x_i}$$

Complexity

How hard is it?

$$\max_{x\in\mathcal{F}} \sum_{k\in[m]} \frac{\sum_{i\in[n]} a_{ki}x_i}{b_{k0} + \sum_{i\in[n]} b_{ki}x_i}$$

- Single-ratio + conv($\mathcal{F}$) $\Rightarrow$ polynomially solvable (Megiddo et al. 1979)

Complexity

How hard is it?

$$\max_{x \in \mathcal{F}} \sum_{k \in [m]} \frac{\sum_{i \in [n]} a_{ki} x_i}{b_{k0} + \sum_{i \in [n]} b_{ki} x_i}$$

- Single-ratio + conv($\mathcal{F}$) $\Rightarrow$ polynomially solvable (Megiddo et al. 1979)
- General single-ratio + $\mathcal{F} \Rightarrow \mathcal{NP}$-hard as generalization of linear IP

Complexity

How hard is it?

$$\max_{x \in \mathcal{F}} \sum_{k \in [m]} \frac{\sum_{i \in [n]} a_{ki} x_i}{b_{k0} + \sum_{i \in [n]} b_{ki} x_i}$$

- Single-ratio + conv($\mathcal{F}$) $\Rightarrow$ polynomially solvable (Megiddo et al. 1979)
- General single-ratio + $\mathcal{F} \Rightarrow \mathcal{NP}$-hard as generalization of linear IP
- Unconstrained single-ratio + modular $\Rightarrow \mathcal{NP}$-hard (Hansen et al. 1991)

Complexity

How hard is it?

$$\max_{x \in \mathcal{F}} \sum_{k \in [m]} \frac{\sum_{i \in [n]} a_{ki} x_i}{b_{k0} + \sum_{i \in [n]} b_{ki} x_i}$$

- Single-ratio + conv($\mathcal{F}$) $\Rightarrow$ polynomially solvable (Megiddo et al. 1979)
- General single-ratio + $\mathcal{F} \Rightarrow \mathcal{NP}$-hard as generalization of linear IP
- Unconstrained single-ratio + modular $\Rightarrow \mathcal{NP}$-hard (Hansen et al. 1991)
- Unconstrained multi-ratio $\Rightarrow$ no algorithm with a factor better than $\mathcal{O}(1/m^{1-\delta})\ \forall \delta > 0$. (Rusmevichientong et al. 2014)

Complexity

How hard is it?

$$\max_{x \in \mathcal{F}} \sum_{k \in [m]} \frac{\sum_{i \in [n]} a_{ki} x_i}{b_{k0} + \sum_{i \in [n]} b_{ki} x_i}$$

- Single-ratio + conv($\mathcal{F}$) $\Rightarrow$ polynomially solvable (Megiddo et al. 1979)
- General single-ratio + $\mathcal{F} \Rightarrow \mathcal{NP}$-hard as generalization of linear IP
- Unconstrained single-ratio + modular $\Rightarrow \mathcal{NP}$-hard (Hansen et al. 1991)
- Unconstrained multi-ratio $\Rightarrow$ no algorithm with a factor better than $\mathcal{O}(1/m^{1-\delta})\ \forall \delta > 0$. (Rusmevichientong et al. 2014)

Very challenging!

Submodularity

Definition

A set function f is *submodular* if it exhibits diminishing returns, i.e. for all $S \subseteq T$

$$f(S \cup \{i\}) - f(S) \geq f(T \cup \{i\}) - f(T).$$

Submodularity

Definition

A set function f is *submodular* if it exhibits diminishing returns, i.e. for all $S \subseteq T$

$$f(S \cup \{i\}) - f(S) \geq f(T \cup \{i\}) - f(T).$$

Goal: understand when a $h(x) := \frac{\sum_{i \in [n]} a_i x_i}{b_0 + \sum_{i \in [n]} b_i x_i}$ is submodular

Submodularity

Definition

A set function f is *submodular* if it exhibits diminishing returns, i.e. for all $S \subseteq T$

$$f(S \cup \{i\}) - f(S) \geq f(T \cup \{i\}) - f(T).$$

Goal: understand when a $h(x) := \frac{\sum_{i \in [n]} a_i x_i}{b_0 + \sum_{i \in [n]} b_i x_i}$ is submodular

When $a_i / b_i = r \, \forall i$, i.e. $a_i = r b_i$

Submodularity

Definition

A set function f is *submodular* if it exhibits diminishing returns, i.e. for all $S \subseteq T$

$$f(S \cup \{i\}) - f(S) \geq f(T \cup \{i\}) - f(T).$$

Goal: understand when a $h(x) := \frac{\sum_{i \in [n]} a_i x_i}{b_0 + \sum_{i \in [n]} b_i x_i}$ is submodular

When $a_i / b_i = r \; \forall i$, i.e. $a_i = r b_i$

$$h(x) = r \frac{\sum_{i \in [n]} b_i x_i}{b_0 + \sum_{i \in [n]} b_i x_i}$$

Submodularity

Definition

A set function f is *submodular* if it exhibits diminishing returns, i.e. for all $S \subseteq T$

$$f(S \cup \{i\}) - f(S) \geq f(T \cup \{i\}) - f(T).$$

Goal: understand when a $h(x) := \frac{\sum_{i\in[n]} a_i x_i}{b_0 + \sum_{i\in[n]} b_i x_i}$ is submodular

When $a_i / b_i = r \, \forall \, i$, i.e. $a_i = r b_i$

$$h(x) = r \frac{\sum_{i\in[n]} b_i x_i}{b_0 + \sum_{i\in[n]} b_i x_i} = rg\left(\sum_{i\in[n]} b_i x_i\right),$$

Submodularity

Definition

A set function f is *submodular* if it exhibits diminishing returns, i.e. for all $S \subseteq T$

$$f(S \cup \{i\}) - f(S) \geq f(T \cup \{i\}) - f(T).$$

Goal: understand when a $h(x) := \frac{\sum_{i \in [n]} a_i x_i}{b_0 + \sum_{i \in [n]} b_i x_i}$ is submodular

When $a_i / b_i = r \; \forall \, i$, i.e. $a_i = r b_i$

$$h(x) = r \frac{\sum_{i \in [n]} b_i x_i}{b_0 + \sum_{i \in [n]} b_i x_i} = rg\left(\sum_{i \in [n]} b_i x_i\right),$$

where $g(t) = t/(b_0 + t)$ is concave; see, e.g. Benati and Hansen (2002)

Submodularity characterization of a single ratio

In general:

Theorem (Han et al. (2020))

Function $h(\cdot)$ is submodular over $\mathcal{F}$ if and only if

$$h(S \cup \{i\}) + h(S \cup \{j\}) \leq \frac{a_i}{b_i} + \frac{a_j}{b_j}$$

for all $S \subseteq N$, and $i, j \notin S$ with $i \neq j$ such that $S \cup \{i\} \cup \{j\} \in \mathcal{F}$.

Submodularity characterization of a single ratio

In general:

Theorem (Han et al. (2020))

Function $h(\cdot)$ is submodular over $\mathcal{F}$ if and only if

$$h(S \cup \{i\}) + h(S \cup \{j\}) \leq \frac{a_i}{b_i} + \frac{a_j}{b_j}$$

for all $S \subseteq N$, and $i, j \notin S$ with $i \neq j$ such that $S \cup \{i\} \cup \{j\} \in \mathcal{F}$.

In the context of assortment optimization:

Proposition (Han et al. (2020))

If

$$\frac{r_{\max} - r_{\min}}{r_{\max}} \leq \min_{S \in \mathcal{F}} \frac{\mathbb{P}\{\text{customer leave with no purchase}; S\}}{\mathbb{P}\{\text{customer make a purchase}; S\}},$$

where $r_{\max} = \max_i a_i/b_i$, $r_{\min} = \min_i a_i/b_i$, then $h(x)$ is submodular.

Monotonicity and submodularity

Definition

A set function h is monotone nondecreasing if $h(S) \leq h(S \cup \{i\})$ for all S and i.

Monotonicity and submodularity

Definition

A set function h is monotone nondecreasing if $h(S) \leq h(S \cup \{i\})$ for all S and i.

Monotonicity $\Rightarrow$ submodularity:

Monotonicity and submodularity

Definition

A set function h is monotone nondecreasing if $h(S) \leq h(S \cup \{i\})$ for all S and i.

Monotonicity $\Rightarrow$ submodularity:

$$h(S) \leq h(S \cup \{i\})$$

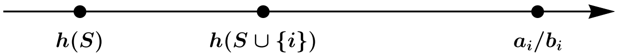
Monotonicity and submodularity

Definition

A set function h is monotone nondecreasing if $h(S) \le h(S \cup \{i\})$ for all S and i.

Monotonicity $\Rightarrow$ submodularity:

$$h(S) \le h(S \cup \{i\}) \Leftrightarrow h(S \cup \{i\}) \le a_i/b_i$$



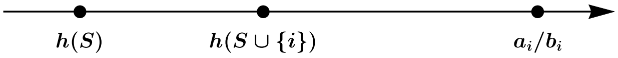
Definition

A set function h is monotone nondecreasing if $h(S) \le h(S \cup \{i\})$ for all S and i.

Monotonicity $\Rightarrow$ submodularity:

$$h(S) \le h(S \cup \{i\}) \Leftrightarrow h(S \cup \{i\}) \le a_i/b_i$$

$$\Rightarrow \quad h(S \cup \{i\}) + h(S \cup \{j\}) \le \frac{a_i}{b_i} + \frac{a_j}{b_j}.$$



Monotonicity and submodularity

Does submodularity $\Rightarrow$ monotonicity?

Monotonicity and submodularity

Does submodularity ⇒ monotonicity?

Example: Consider

$$h(x) = \frac{3x_1 + 2x_2 + x_3}{2 + x_1 + x_2 + x_3},$$

which is submodular.

Monotonicity and submodularity

Does submodularity $\Rightarrow$ monotonicity?

Example: Consider

$$h(x) = \frac{3x_1 + 2x_2 + x_3}{2 + x_1 + x_2 + x_3},$$

which is submodular. However,

$$h(\{3\}) = \frac{1}{3} < h(\{1,2,3\}) = \frac{6}{5} < h(\{1,2\}) = \frac{5}{4}$$

$\Rightarrow$ monotonicity fails to hold.

Monotonicity and submodularity

Does submodularity $\Rightarrow$ monotonicity?

Example: Consider

$$h(x) = \frac{3x_1 + 2x_2 + x_3}{2 + x_1 + x_2 + x_3},$$

which is submodular. However,

$$h(\{3\}) = \frac{1}{3} < h(\{1,2,3\}) = \frac{6}{5} < h(\{1,2\}) = \frac{5}{4}$$

$\Rightarrow$ monotonicity fails to hold.

Proposition (Han et al. (2020))

If $h(x)$ is submodular over $\mathcal{F}$, then it is monotone nondecreasing over $\mathcal{F}_1 := \{S \in \mathcal{F} : n \in S\}$ and $\mathcal{F}_2 = \{S \in \mathcal{F} : n \notin S\}$, where $a_n/b_n = \min_{i\in[n]} a_i/b_i$.

Membership testing

Submodularity testing amounts to solving

$$\frac{a_i}{b_i} + \frac{a_j}{b_j} \geq t_{ij} := \max_{S \in \mathcal{F}} \; h(S \cup \{i\}; a, b) + h(S \cup \{j\}; a, b)$$

Membership testing

Submodularity testing amounts to solving

$$\frac{a_i}{b_i} + \frac{a_j}{b_j} \geq t_{ij} := \max_{S \in \mathcal{F}} \; h(S \cup \{i\}; a, b) + h(S \cup \{j\}; a, b)$$

Testing algorithm for $\mathcal{F} = 2^{[n]}$:

- Sort $a_1/b_1 \geq a_2/b_2 \geq \cdots \geq a_n/b_n$

Membership testing

Submodularity testing amounts to solving

$$\frac{a_i}{b_i} + \frac{a_j}{b_j} \geq t_{ij} := \max_{S \in \mathcal{F}} h(S \cup \{i\}; a, b) + h(S \cup \{j\}; a, b)$$

Testing algorithm for $\mathcal{F} = 2^{[n]}$:

- Sort $a_1/b_1 \geq a_2/b_2 \geq \cdots \geq a_n/b_n$
- Check if monotonicity holds over $\mathcal{F}_1 = \{S \in \mathcal{F} : n \in S\}$ and $\mathcal{F}_2 = \{S \in \mathcal{F} : n \notin S\}$

Membership testing

Submodularity testing amounts to solving

$$\frac{a_i}{b_i} + \frac{a_j}{b_j} \geq t_{ij} := \max_{S \in \mathcal{F}} h(S \cup \{i\}; a, b) + h(S \cup \{j\}; a, b)$$

Testing algorithm for $\mathcal{F} = 2^{[n]}$:

- Sort $a_1/b_1 \geq a_2/b_2 \geq \cdots \geq a_n/b_n$
- Check if monotonicity holds over $\mathcal{F}_1 = \{S \in \mathcal{F} : n \in S\}$ and $\mathcal{F}_2 = \{S \in \mathcal{F} : n \notin S\}$
- Check if $t_{in} \leq a_i/b_i + a_n/b_n$ holds for all $i \in [n-1]$

Implications in computations

How can we benefit from submodularity?

Implications in computations

How can we benefit from submodularity?

Submodular function maximization

Proposition (Nemhauser et al. (1978))

When the feasible region is given by a cardinality constraint,the greedy algorithm produces a solution with $(1 - e^{-1})$ *approx factor for* $\max_{S \in \mathcal{F}} f(S)$.

Implications in computations

How can we benefit from submodularity?

Submodular function maximization

Proposition (Nemhauser et al. (1978))

When the feasible region is given by a cardinality constraint,the greedy algorithm produces a solution with $(1 - e^{-1})$ approx factor for $\max_{S \in \mathcal{F}} f(S)$.

Decomposition and cutting plane methods

$$h(x) = \frac{a'x}{1 + b'x} = \left(h(x) + \alpha \frac{b'x}{1 + b'x} \right) - \left(\alpha \frac{b'x}{1 + b'x} \right).$$

- epigraph ⇔ Lovász extension
- hypograph ⇒ valid inequalities

Computational experiment

Atamtürk and Narayanan (2021) consider

$$\min \left\{ \frac{a'x}{1 + b'x} - \Omega s'x : x \in \{0, 1\}^n \right\}$$

Benchmark:

Computational experiment

Atamtürk and Narayanan (2021) consider

$$\min \left\{ \frac{a'x}{1 + b'x} - \Omega s'x : x \in \{0,1\}^n \right\}$$

Benchmark:

- Branch and Bound (B&B)

Computational experiment

Atamtürk and Narayanan (2021) consider

$$\min\left\{\frac{a'x}{1+b'x} - \Omega s'x : x \in \{0,1\}^n\right\}$$

Benchmark:

- Branch and Bound (B&B)
- B&B + cuts from submodular-supermodular decomposition

Computational results

λ	0.00	0.2	0.60	0.8	1.0
Gap(%)	1326.80	856.80	347.10	178.70	44.00
Cgap(%)	**90.80**	**61.50**	**21.50**	**9.60**	**0.00**
Time(s)	83.30	117.20	261.40	84.50	40.80
Ctime(s)	**44.10**	**88.90**	**71.10**	**12.30**	**0.00**
#Nodes	3.1E+04	3.6E+04	5.7E+04	3.2E+04	2.4E+04
#Cnodes	**1.6E+04**	**2.1E+04**	**2.2E+04**	**9.7E+03**	**0.0**
#Cuts	27.60	27.80	23.20	23.40	22.00

All computations are done with Gurobi version 9.0 on a Xeon workstation

Take home message

Take home message

- Characterization of submodularity of a single ratio

Take home message

- Characterization of submodularity of a single ratio
- Monotonicity $\Leftarrow\cdots\Longrightarrow$ submodularity

Take home message

- Characterization of submodularity of a single ratio
- Monotonicity $\dashleftarrow\!\Longrightarrow$ submodularity
- Membership testing

Take home message

- Characterization of submodularity of a single ratio
- Monotonicity $\dashleftarrow\!\Longrightarrow$ submodularity
- Membership testing

Our paper is available at: `https://arxiv.org/abs/2012.07235`

Take home message

- Characterization of submodularity of a single ratio
- Monotonicity $\Longleftarrow\!\!\!\!\Longrightarrow$ submodularity
- Membership testing

Our paper is available at: `https://arxiv.org/abs/2012.07235`

Thank You!

Reference

Atamtürk, A. and Narayanan, V. (2021). Submodular function minimization and polarity. *Mathematical Programming*, pages 1–11.

Benati, S. and Hansen, P. (2002). The maximum capture problem with random utilities: Problem formulation and algorithms. *European Journal of Operational Research*, 143(3):518–530.

Han, S., Gómez, A., and Prokopyev, O. A. (2020). Fractional 0-1 programming and submodularity. *arXiv preprint arXiv:2012.07235*.

Hansen, P., De Aragão, M. V. P., and Ribeiro, C. C. (1991). Hyperbolic 0–1 programming and query optimization in information retrieval. *Mathematical Programming*, 52(1-3):255–263.

Megiddo, N. et al. (1979). Combinatorial optimization with rational objective functions. *Mathematics of Operations Research*, 4(4):414–424.

Méndez-Díaz, I., Miranda-Bront, J. J., Vulcano, G., and Zabala, P. (2014). A branch-and-cut algorithm for the latent-class logit assortment problem. *Discrete Applied Mathematics*, 164:246–263.

Nemhauser, G. L., Wolsey, L. A., and Fisher, M. L. (1978). An analysis of approximations for maximizing submodular set functions—I. *Mathematical Programming*, 14(1):265–294.

Rusmevichientong, P., Shmoys, D., Tong, C., and Topaloglu, H. (2014). Assortment optimization under the multinomial logit model with random choice parameters. *Production and Operations Management*, 23(11):2023–2039.

Tawarmalani, M., Ahmed, S., and Sahinidis, N. V. (2002). Global optimization of 0-1 hyperbolic programs. *Journal of Global Optimization*, 24(4):385–416.

Ursulenko, O., Butenko, S., and Prokopyev, O. A. (2013). A global optimization algorithm for solving the minimum multiple ratio spanning tree problem. *Journal of Global Optimization*, 56(3):1029–1043.