Adversarially Robust Submodular Maximization under Knapsack Constraints

Dmitry Avdyukhin

Slobodan Mitrović

Grigory Yaroslavtsev

Samson Zhou

Submodular Functions

- ❖ Ground Set V (items, sets, vertices)
- ❖ Set function $f: 2^V \rightarrow \mathbb{R}$ with diminishing returns property

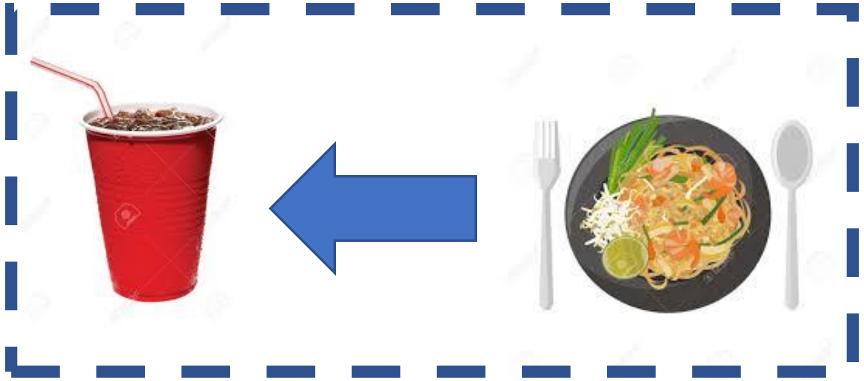
Oracle access to f.
Given a subset $S \subseteq V$
returns $f(S)$

$\forall A \subseteq B \subseteq V, e \notin B$

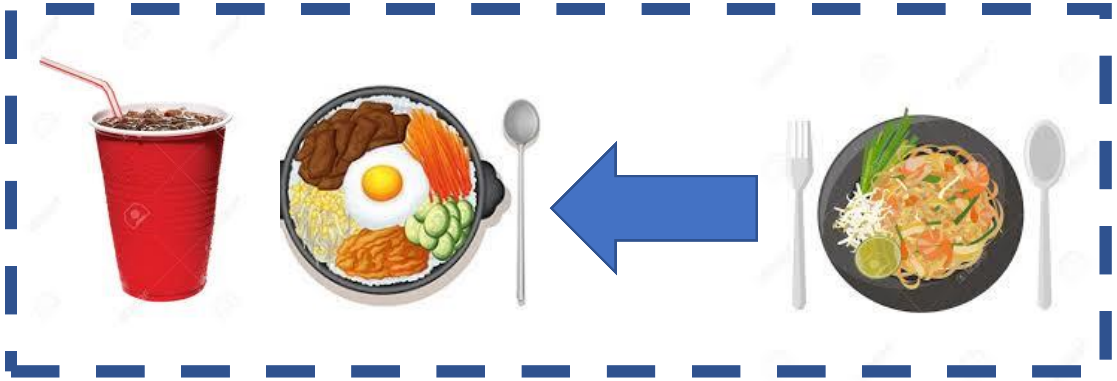
$$f(A \cup \{e\}) - f(A) \quad \geq \quad f(B \cup \{e\}) - f(B)$$

Submodular Functions

$V = \{$  $\}$



A



B

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

Applications of Submodular Functions

- Viral marketing [Kempe et al., 2003]
- Feature selection [Krause & Guestrin, 2005]
- Clustering [Narasimhan & Bilmes, 2005]
- Search result diversification [Agrawal et al., 2009]
- Recommender systems [El-Arini & Guestrin, 2011]
- Active learning [Golovin & Krause, 2011]
- Document summarization [Lin & Bilmes, 2011]
- Data subset selection [Wei, Iyer & Bilmes, 2015]
- etc

Clustering









Clustering



$$C(S) = \frac{1}{|V|} \sum_{e \in V} \min_{v \in S} d(e, v)$$

$$f(S) = C(\{e_0\}) - C(S \cup \{e_o\})$$

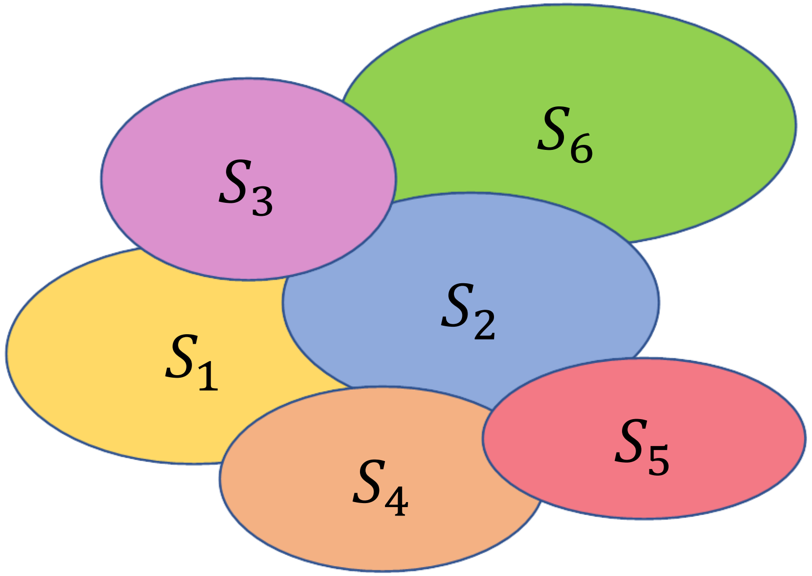


Coverage

- $E = \{e_1, e_2, \ldots, e_n\}$
- $V \subseteq 2^E$

$S = \{s_{i_1}, s_{i_2}, \ldots, s_{i_k}\} \in V$

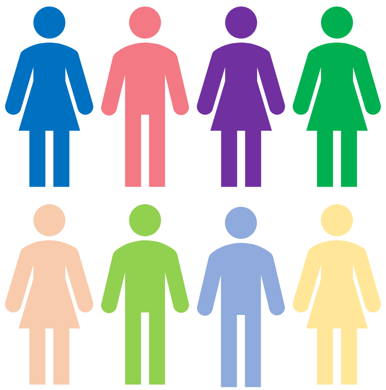
$f(S) = |\cup_{s_i \in S} s_i|$

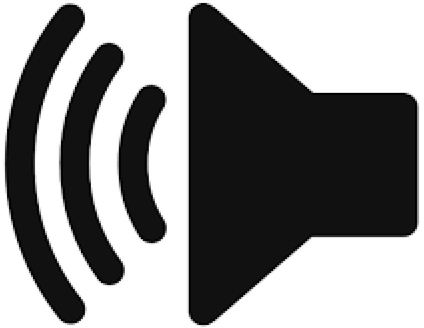


f is a submodular function

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

Viral Marketing





$$S^* = \arg \max_{c(S) \leq K} f(S)$$

- $E = \{p_1, p_2, \dots, p_n\}$
- $V \subseteq 2^E$

$$S = \{s_{i_1}, s_{i_2}, \dots, s_{i_k}\} \in V$$

$$c(s_i) \geq 0$$

First Objective

Submodular maximization under cardinality constraint

f is submodular, monotone, and $f(\emptyset) = 0$

Extract ***small*** representative subset out of a big dataset

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

Solving this problem exactly is NP-hard

Greedy [Nemhauser, Wolsey, Fisher, '78]

$V = \{$  $\}$

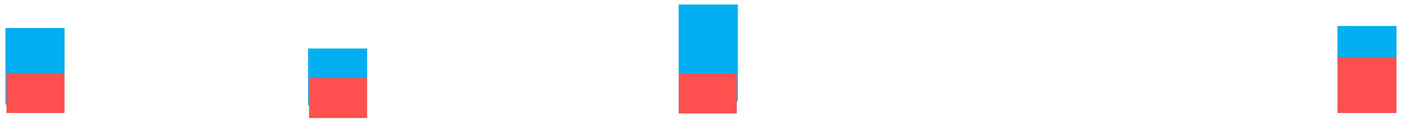
Marginal gain: 

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

$f($  $)$

Greedy [Nemhauser, Wolsey, Fisher, ‘78]

$V = \{$  $\}$

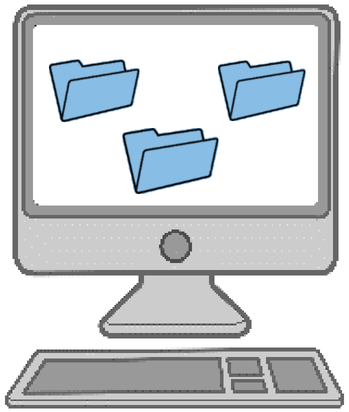
Marginal gain: 

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

$f($  $)$

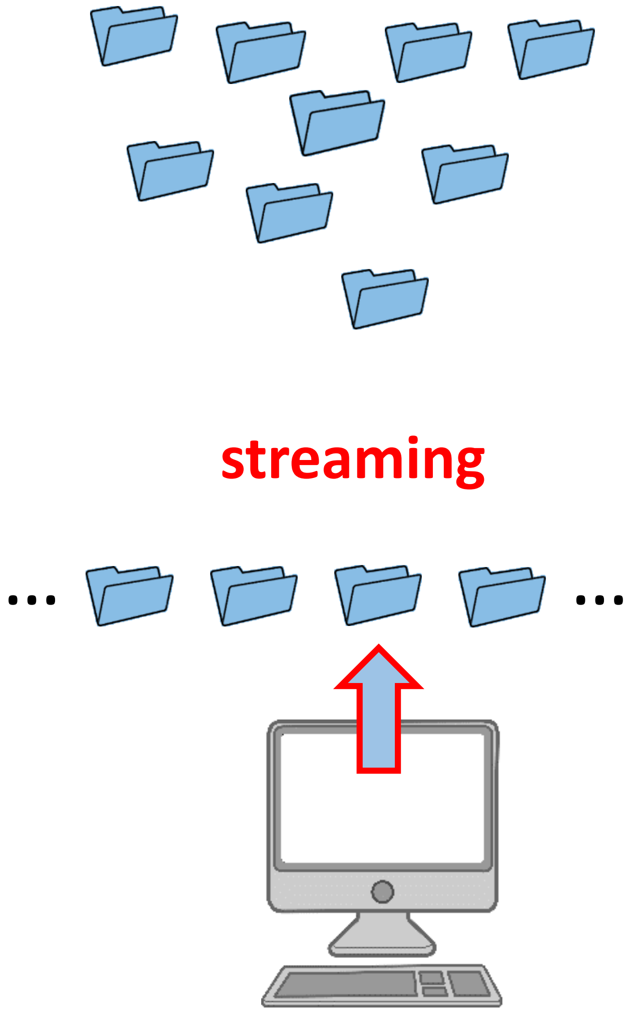
$\geq (1 - 1/e)\mathrm{OPT}$

Traditional

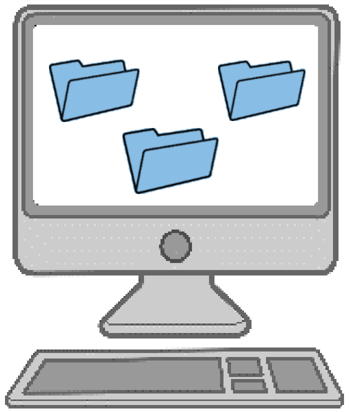


Algorithms performed sequentially.

Modern

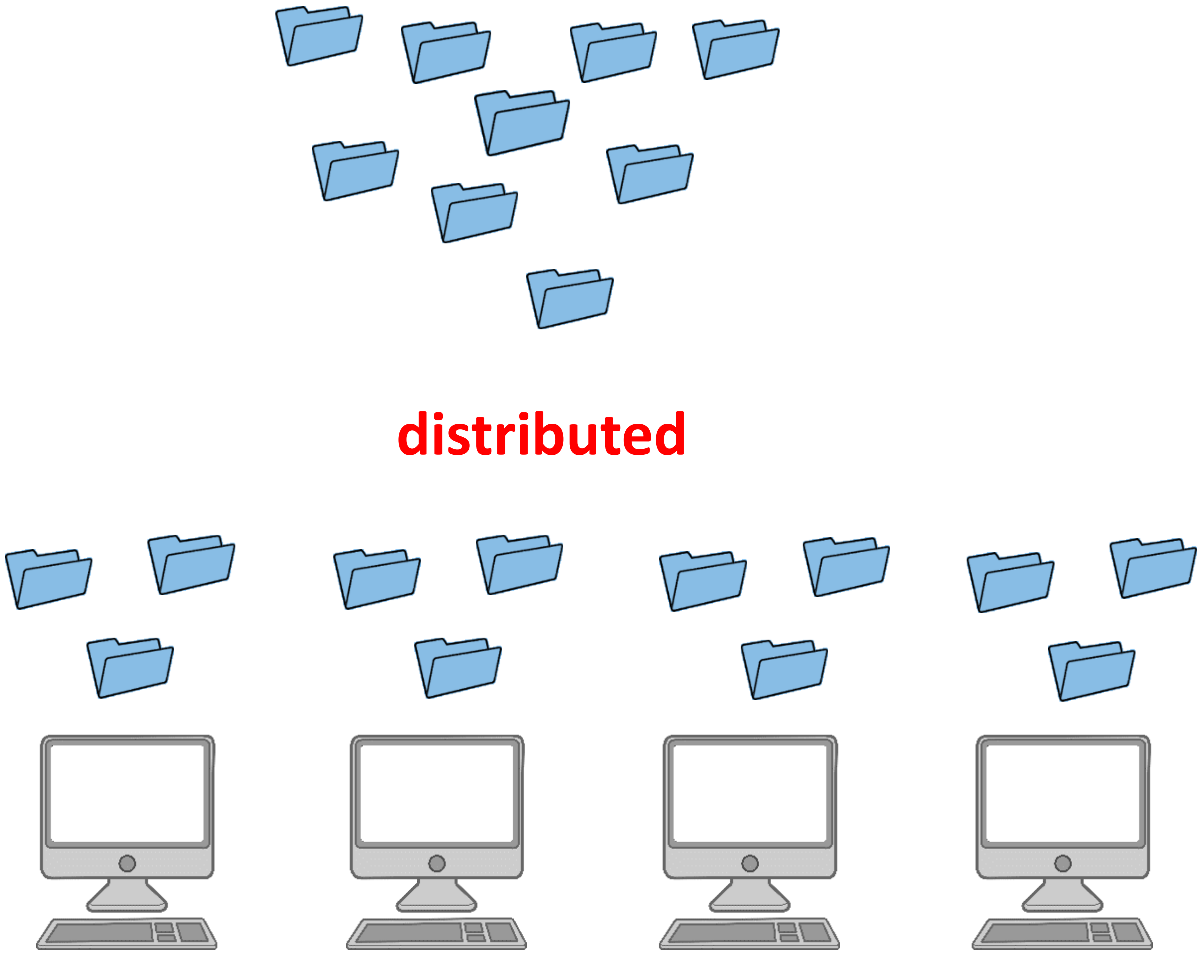


Traditional



Algorithms performed sequentially.

Modern



Thresholding [Badanidiyuru, Mirzasoleiman, Karbasi, Krause, '14]

Stream: 

$$\geq^{?} \frac{\mathrm{OPT}}{2k}$$

$f($  $)$

Thresholding [Badanidiyuru, Mirzasoleiman, Karbasi, Krause, '14]

Stream: 

$$\geq^{?} \frac{\mathrm{OPT}}{2k}$$

$f($  $)$

Thresholding [Badanidiyuru, Mirzasoleiman, Karbasi, Krause, ‘14]

Stream:

$\geq^{?} \frac{\text{OPT}}{2k}$

$f($  $)$

Thresholding [Badanidiyuru, Mirzasoleiman, Karbasi, Krause, '14]

Stream:

$$\geq^{?} \frac{\text{OPT}}{2k}$$

$f($  $)$

Thresholding [Badanidiyuru, Mirzasoleiman, Karbasi, Krause, '14]

Stream: 

$\text{OPT} \in \{(1+\epsilon)^i \mid i \in \boldsymbol{N}\}$

$\geq^{?} \frac{\text{OPT}}{2k}$

$f($  $) \geq \frac{1}{2}\,\text{OPT}$

Second Objective

Submodular maximization under knapsack constraint

f is submodular, monotone, and $f(\emptyset) = 0$

Extract ***small*** representative subset out of a big dataset

$$S^* = \arg \max_{c(S) \leq K} f(S)$$

Solving this problem exactly is NP-hard

Thresholding in Review

- Key concept: marginal gain $f(e \mid S) = f(e \cup S) - f(S)$
- If marginal gain exceeds threshold, add item to S.
- Else, discard item.

- $f(\mathrm{OPT} \cup S) \geq f(\mathrm{OPT}), \quad |\mathrm{OPT} \cup S| \leq 2k$

- What about for knapsack constraints?
- Could have item with good marginal gain, but really large size

Knapsack Optimization

Stream:

Knapsack Optimization

- Key concept: marginal density $\rho(e \mid S) = \frac{f(e \cup S) - f(S)}{c(e)}$
- If marginal density exceeds threshold, add item to S.
- Else, discard density. Does it work?

Knapsack Optimization

- ALG 1:
  - If marginal density exceeds threshold, add item to S.
  - Else, discard density.
- ALG 2:
  - Keep “best” element
- ALG: Return max(ALG 1, ALG 2)

Adversarial Robust Submodular Optimization

NETFLIX

Adversarial Robust Submodular Optimization

NETFLIX

Adversarial Robust Submodular Optimization

Adversarial Robust Submodular Optimization

NETFLIX

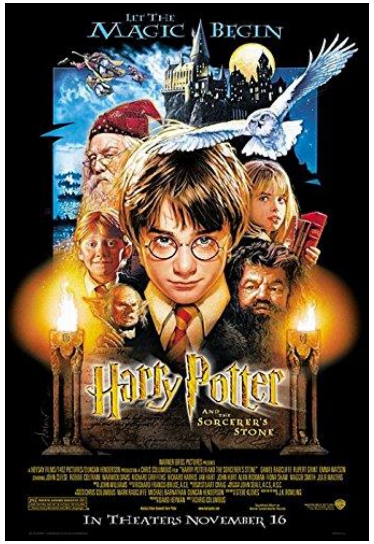
Adversarial Robust Submodular Optimization

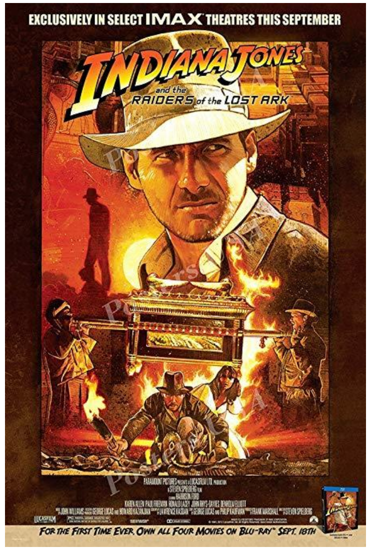
NETFLIX

Adversarial Robust Submodular Optimization

Adversarial Robust Submodular Optimization

NETFLIX





Adversarial Robust Submodular Optimization

- $V = \{e_1, e_2, \dots, e_n\}$
- Monotone submodular function f
- $c(e_i) \geq 0$

- See all the data and make summary Z.
- Given set E that is removed from V.

- Goal: $S^* = \arg \max_{c(S) \leq K, S \cap E = \emptyset} f(S)$

Results

- Streaming algorithm for single knapsack, robust to the removal of m items.
- Better streaming algorithm for single knapsack, robust to the removal of size M.
- Streaming algorithm for multiple knapsack, robust to the removal of m items.
- Distributed algorithm for multiple knapsack, robust to the removal of m items.
- Size of our summaries are almost optimal.

Approach

Stream: 

- ❖ Algorithm to produce summary S
- ❖ $Z = S \backslash E.$
- ❖ Run Greedy on Z

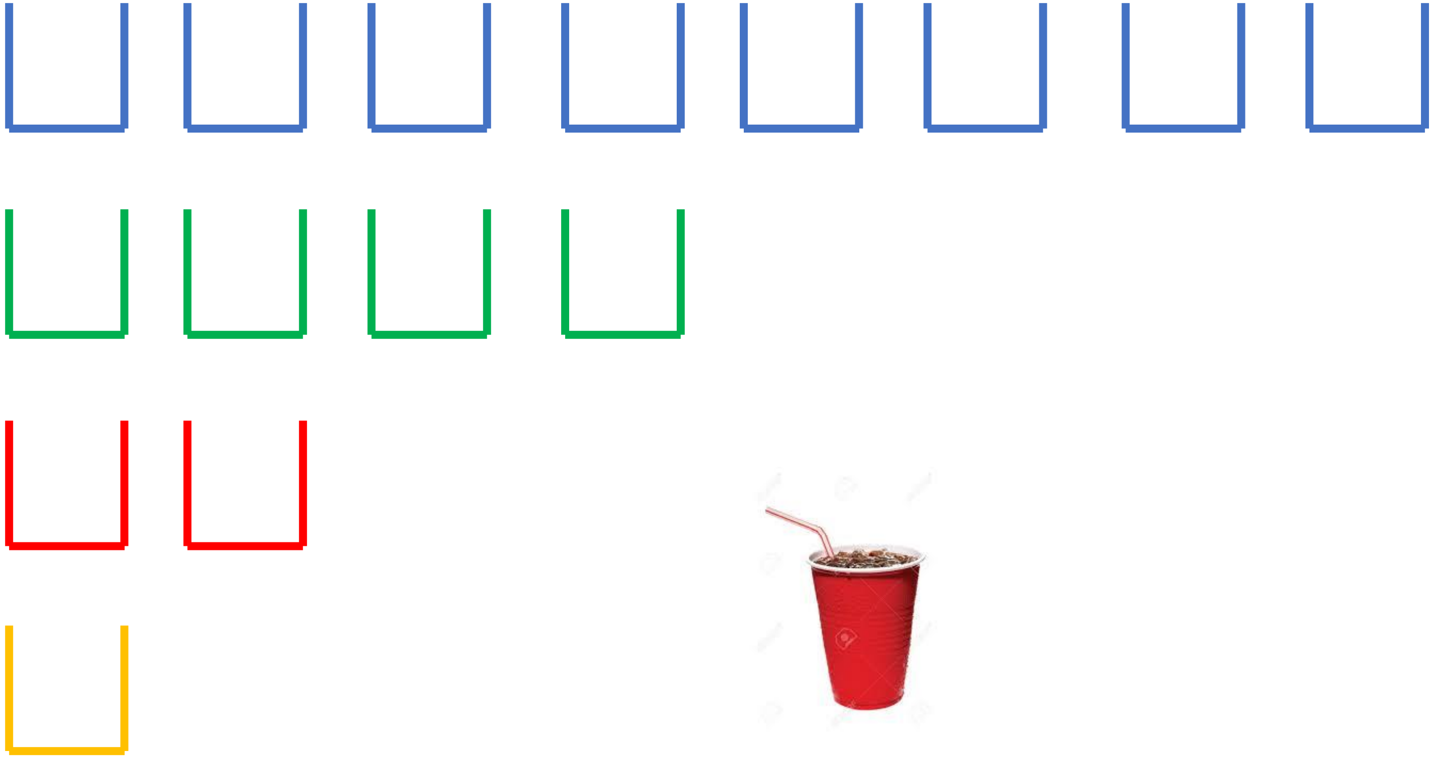
Partitions and Buckets Data Structure [Bogunovic, Mitrovic, Scarlett, and Cevher '17]

Partition 3 τ

Partition 2 $\tau/2$

Partition 1 $\tau/4$

Partition 0 $\tau/8$



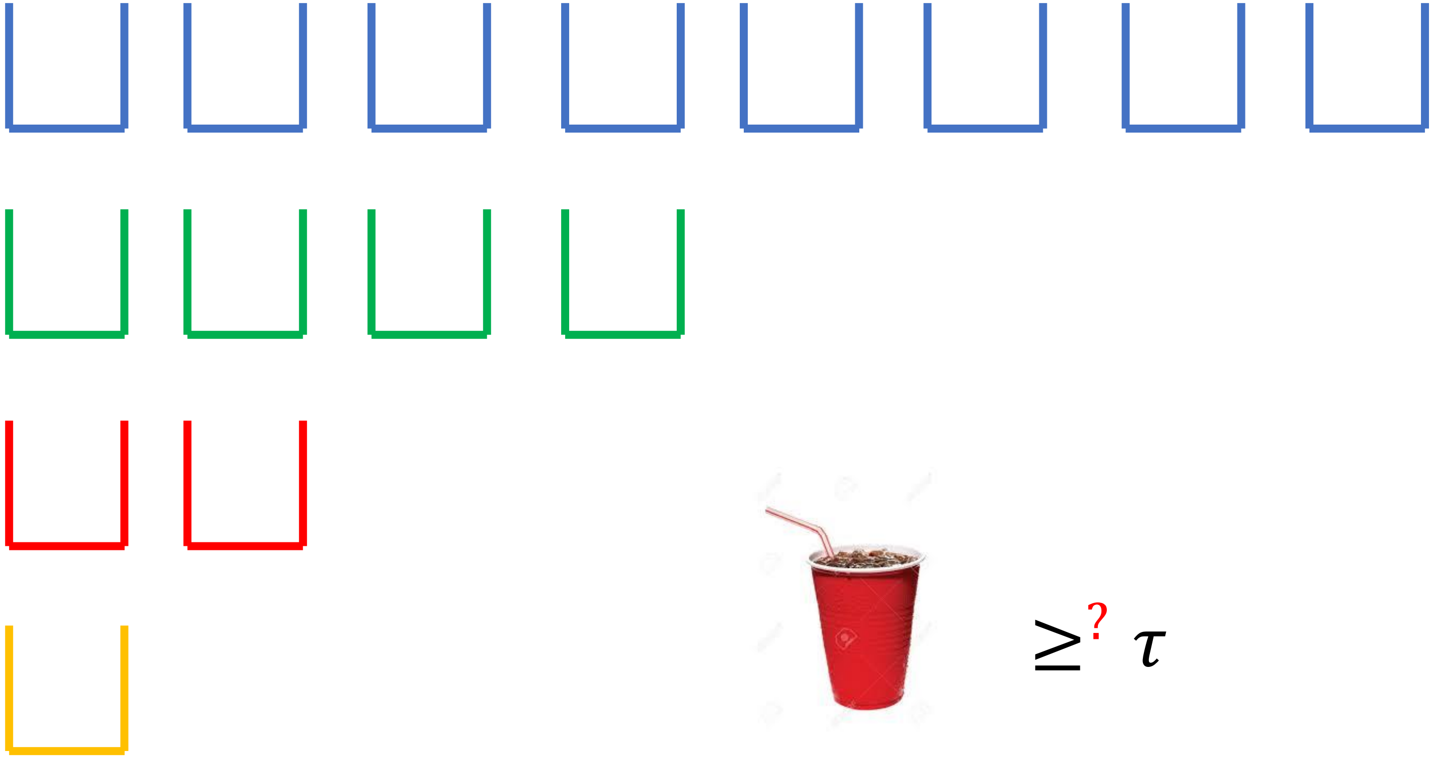
Partitions and Buckets Data Structure [Bogunovic, Mitrovic, Scarlett, and Cevher '17]

Partition 3 τ

Partition 2 $\tau/2$

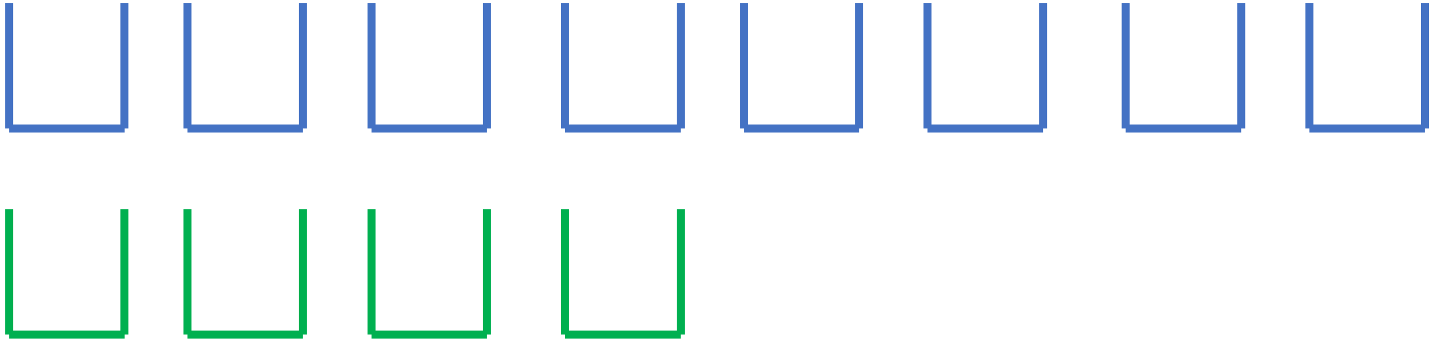
Partition 1 $\tau/4$

Partition 0 $\tau/8$



$\overset{?}{\geq} \tau$

Partitions and Buckets Data Structure [Bogunovic, Mitrovic, Scarlett, and Cevher '17]



Partition 3 τ

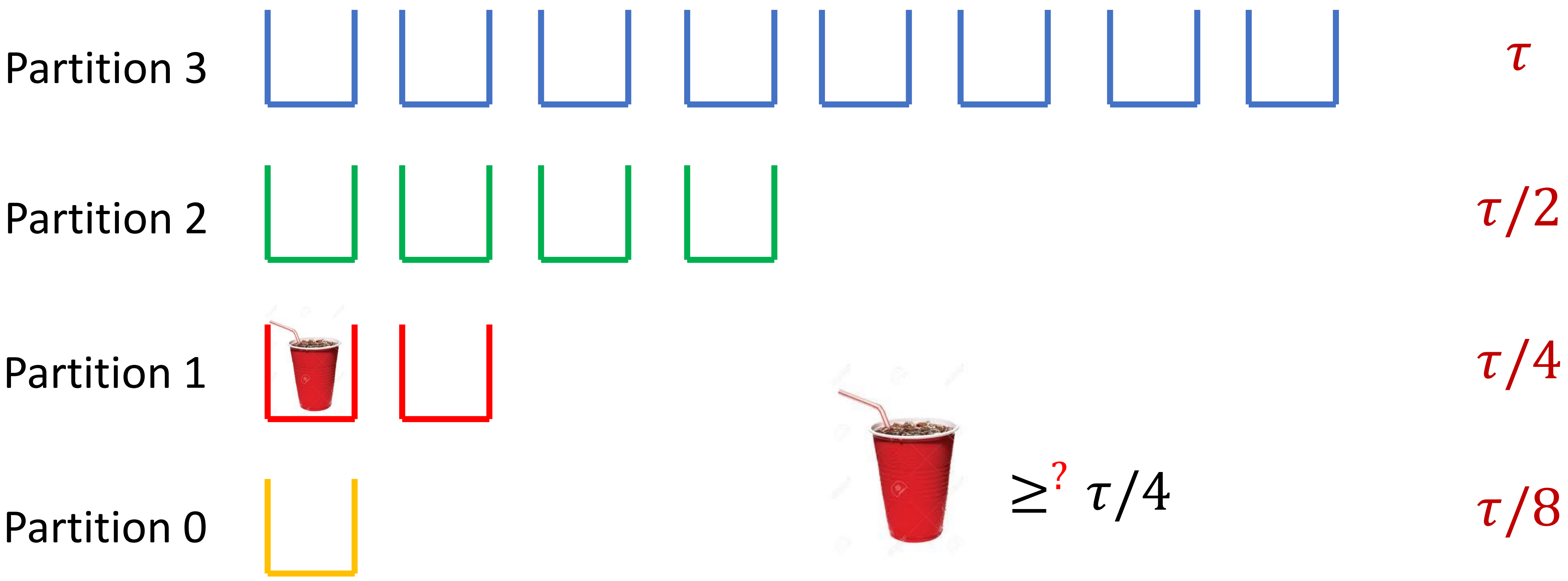
Partition 2 $\tau/2$

Partition 1 $\tau/4$

Partition 0 $\tau/8$

$\overset{?}{\geq} \tau/2$

Partitions and Buckets Data Structure [Bogunovic, Mitrovic, Scarlett, and Cevher '17]



Partitions and Buckets Data Structure [Bogunovic, Mitrovic, Scarlett, and Cevher '17]

- Items in high partitions are more valuable
- Bad approximation if they are deleted, so we need more buckets

- Items in low partitions are not as valuable
- Still have good approximation if many buckets are full

- If many items are deleted from high partitions, but buckets in low partitions are not full, must still have captured "good" items

Towards Knapsack Constraints

- Initial idea: replace marginal gain with marginal density

Partition 0

- Problem: big items can't fit
- Hotfix: double the size of each bucket

Towards Knapsack Constraints

- Problem: number of buckets is based on the threshold, not size

Partition 0

Remains good approximation

Partition 0

Bad approximation!

Towards Knapsack Constraints

- Problem: number of buckets is based on the threshold, not size
- Main idea: *Dynamic* bucketing scheme

- Each time element is added, allocate space proportional to its size

Partition 0

- Cap total number of items

Towards Knapsack Constraints

- Items in high partitions are more valuable
- Bad approximation if they are deleted, so we need more buckets

- Items in low partitions are not as valuable unless they are large
- Large items allocate more buckets
- Still have good approximation if many buckets are full

- If many items are deleted from high partitions, but buckets in low partitions are not full, must still have captured “good” items

ARMSM: Multiple Knapsacks

- $V = \{e_1, e_2, \ldots, e_n\}$
- Monotone submodular function f
- $c_1(e_i) \geq 0,\ c_2(e_i) \geq 0,\ \ldots,\ c_d(e_i) \geq 0$

- See all the data and make summary Z.
- Given set E that is removed from V.

- Goal: $S^* = \arg \max_{c_1(S) \leq b_1, \ldots c_d(S) \leq b_d, S \cap E = \emptyset} f(S)$

Normalization

- Rescale each row i in cost matrix by b_1/b_i so that all knapsack constraints are $K \coloneqq b_1$.
- Rescale all entries in cost matrix and constraint vector by minimum entry so that all costs are at least 1.

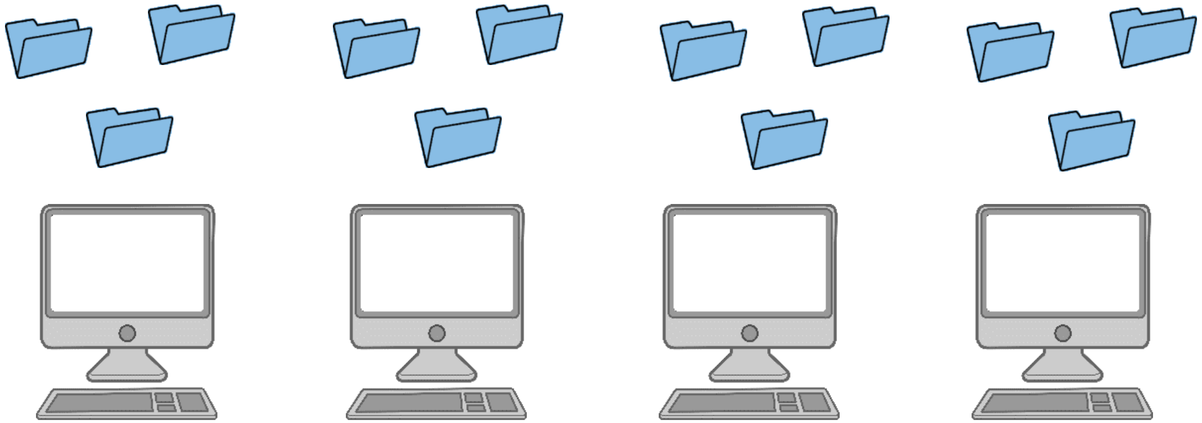
ARMSM: Multiple Knapsacks

- Recall algorithm: partitions and buckets, add item if marginal density exceeds threshold.
- What is the marginal density here?

- Marginal gain divided by the *largest* cost (across all knapsacks).
- Lose a factor of ~ $\frac{1}{2d}$ in the approximation guarantee.

Distributed Algorithm

- Send partition and buckets data structure and data across multiple machines
- With high probability, “bad” cases will be split across multiple machines



Results

- First constant factor approximation algorithms for submodular maximization robust to a number of removals

Model	Removal	Approximation	Constraint	Remarks
Streaming	m items	$O(1)$	Single knapsack	Nearly optimal summary size
Streaming	M space	$O(1)$	Single knapsack	Better guarantees, nearly optimal summary size *and* algorithm space
Streaming	m items	$O\left(\frac{1}{d}\right)$	d knapsacks	Nearly optimal summary size
Distributed	m items	$O\left(\frac{1}{d}\right)$	d knapsacks	2 rounds of communication

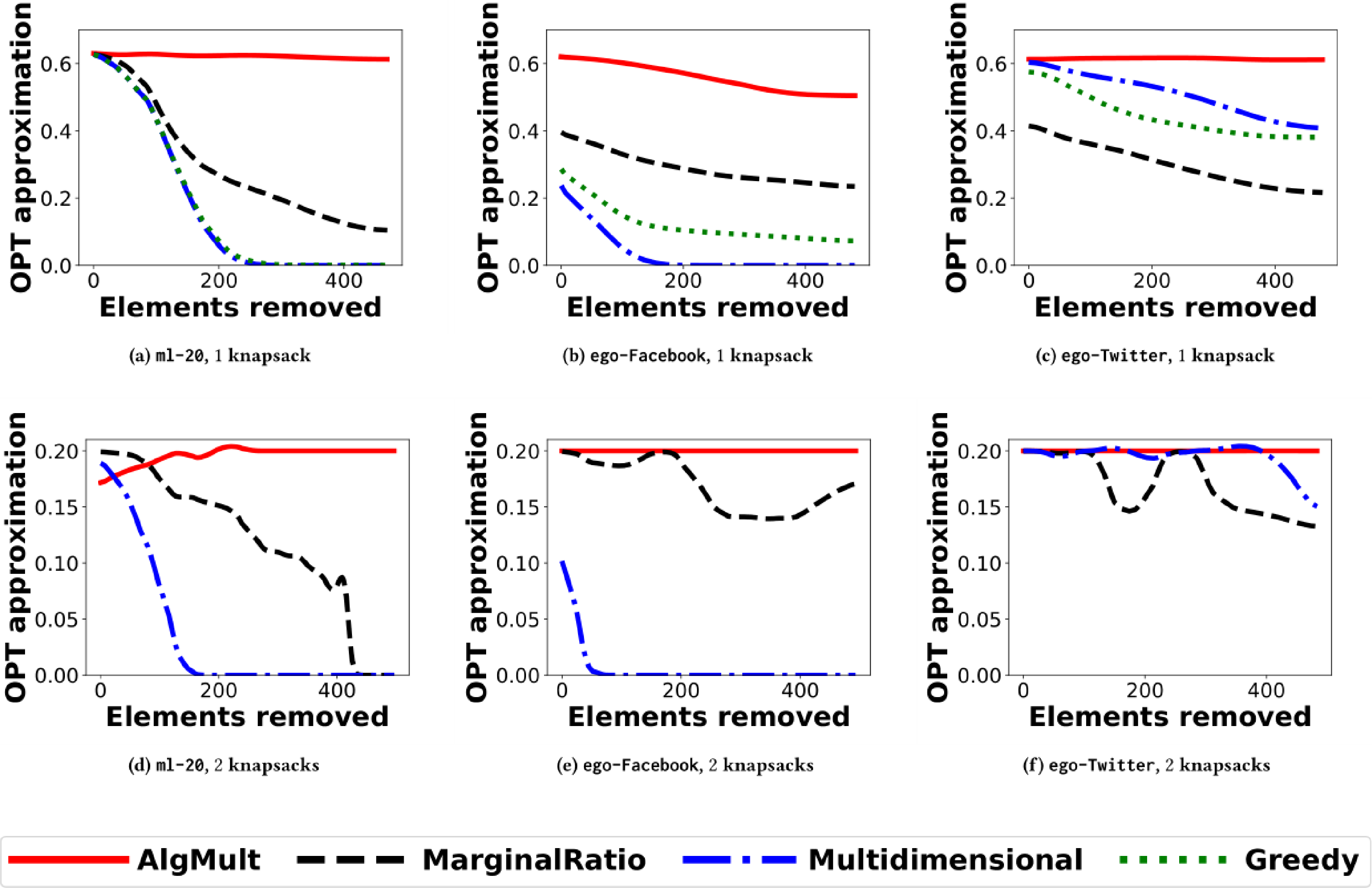
Empirical Evaluations

- Social network graphs from Facebook (4K vertices, 81K edges) and Twitter (88K vertices, 1.8M edges) collected by the Stanford Network Analysis Project (SNAP).
- Dominating set

- MovieLens (27K movies, 200K ratings)
- Coverage

- Baselines: Offline Greedy, “Robustified” versions of streaming algorithms

Empirical Evaluations



(a) ml-20, 1 knapsack
(b) ego-Facebook, 1 knapsack
(c) ego-Twitter, 1 knapsack
(d) ml-20, 2 knapsacks
(e) ego-Facebook, 2 knapsacks
(f) ego-Twitter, 2 knapsacks

Empirical Evaluations

	ml-20, 1 knapsack	fb, 1 knapsack	twitter, 1 knapsack	ml-20, 2 knapsacks	fb, 2 knapsacks	twitter, 2 knapsacks
AlgMult	641	378	401	1350	2745	4208
MarginalRatio	641	377	402	1350	2745	4209
Multidimensional	87	18	435	72	22	4221
Greedy	647	393	493	-	-	-

Table 1: Sizes of robust summaries produced by the algorithms ($K = 10$).

Related Questions?

- ❖ Non-monotone robust submodular maximization
- ❖ Other constraints
- ❖ Better approximation guarantee
- ❖ Streaming algorithms with less space

Questions?


рахмат
danke
謝謝
ngiyabonga
teşekkür ederim
Баярлалаа
mersi
kia ora
barka
welalin
tack
spas
mahalo
tapadh leat
спасибо
faafetai lava
vinaka
спасибі
blagodaram
dank je
misaotra
matondo
paldies
grazzi
kiitos
dankie
nanni
nandri
dhanyavad
thank you
gracias
хвала
asante
manana
obrigada
tenki
murakoze
bedankt
bayarlalaa
gràcie
hvala
maururu
köszönöm
akun
dankon
aćiū
djiere dieuf
tau
mochchakkeram
дякую
mamnun
chokrane
enkosi
dziękuję
chnorakaloutioun
gratias ago
gràcies
sulpáy
taiku
go raibh maith agat
sobodi
dĕkuji
obrigado
sukriya
kop khun krap
arigatô
takk
dakujem
trugarez
mèsi
didi madloba
sagolun
najis tuke
ありがとう
tanemirt
grazie
shukriya
mercé
мерси
kam sah hamnida
rahmat
terima kasih
rahmet
diolch
dhanyavadagalu
তোমাকে ধন্যবাদ
감사합니다
xièxie
ευχαριστώ
merci