Near-optimal asymmetric binary matrix partitions

Alexandros Voudouris

joint work with

Fidaa Abed and Ioannis Caragiannis

University of Patras and CTI "Diophantus"

Take-it-or-leave-it sales

- A seller with m items for sale
- n potential buyers
- Buyers have values for the items

Take-it-or-leave-it sales

- A seller with m items for sale
- n potential buyers
- Buyers have values for the items

Take-it-or-leave-it sales

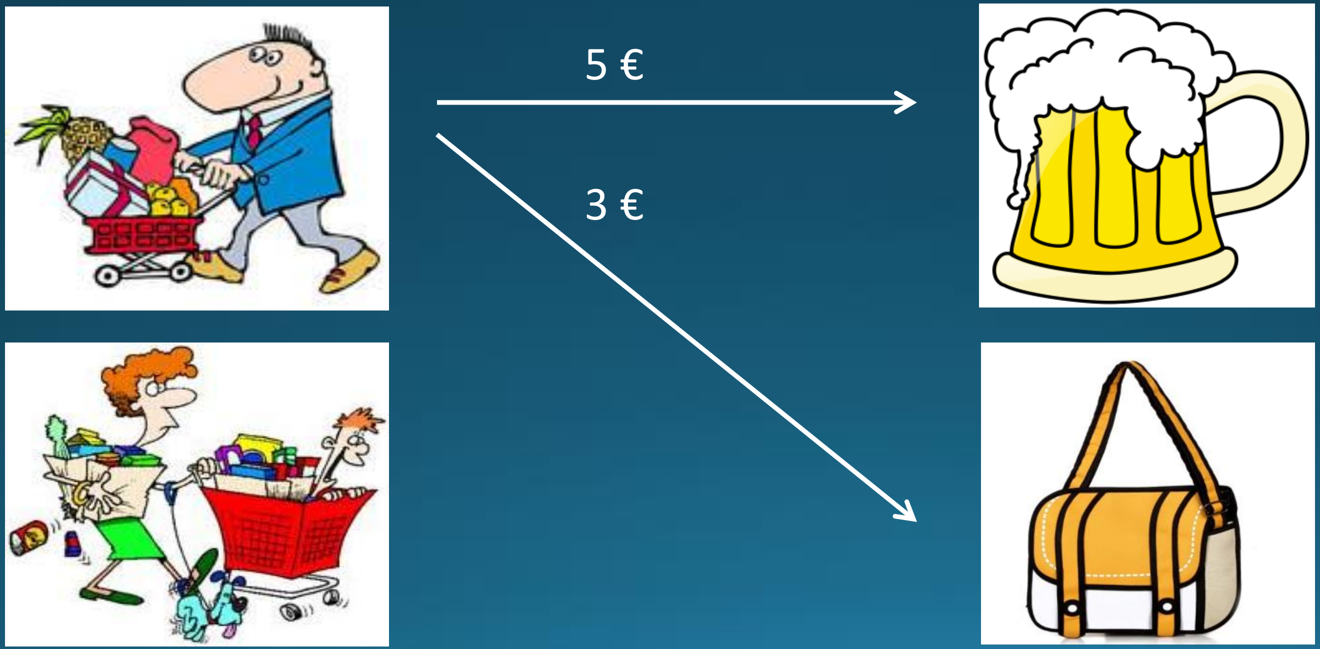
- A seller with m items for sale
- n potential buyers
- Buyers have values for the items


Take-it-or-leave-it sales

- A seller with m items for sale
- n potential buyers
- Buyers have values for the items

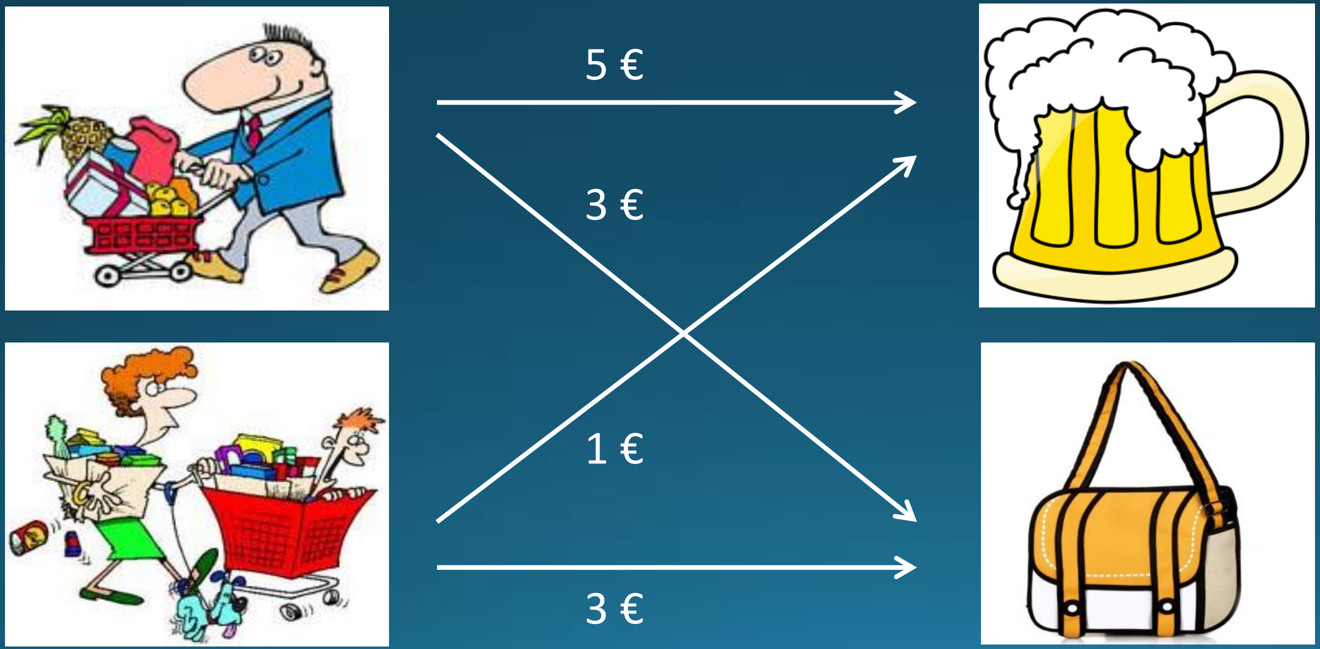
Take-it-or-leave-it sales

- A seller with m items for sale
- n potential buyers
- Buyers have values for the items



Take-it-or-leave-it sales

- A seller with m items for sale
- n potential buyers
- Buyers have values for the items



Take-it-or-leave-it sales

- A seller with m items for sale
- n potential buyers
- Buyers have values for the items

		
	5€	3€
	1€	3€

Take-it-or-leave-it sales

- Nature selects an item at random according to a prior distribution
- The seller wants to sell this item and maximize his *revenue*
- He offers the item to the buyer that values it the most

		
	**5€**	**3€**
	**1€**	**3€**

Take-it-or-leave-it sales

- Nature selects an item at random according to a prior distribution
- The seller wants to sell this item and maximize his *revenue*
- He offers the item to the buyer that values it the most

		
	**5€**	**3€**
	**1€**	**3€**

Take-it-or-leave-it sales

- Nature selects an item at random according to a prior distribution
- The seller wants to sell this item and maximize his *revenue*
- He offers the item to the buyer that values it the most

		
	5€	3€
	1€	3€

Take-it-or-leave-it sales

- Nature selects an item at random according to a prior distribution
- The seller wants to sell this item and maximize his *revenue*
- He offers the item to the buyer that values it the most

		
	5€	3€
	1€	3€

Take-it-or-leave-it sales

- What if there are items for which no buyer has any value for?

Take-it-or-leave-it sales

- What if there are items for which no buyer has any value for?

			
	5€	3€	0€
	1€	3€	0€

Take-it-or-leave-it sales

- What if there are items for which no buyer has any value for?

			
	5€	3€	0€
	1€	3€	0€

- Is it possible to sell such items at a price higher than 0?

Take-it-or-leave-it sales

- Solution: *bundle* items together

- *Symmetric* bundling: all bundles are the same for all buyers
- *Asymmetric* bundling: different bundles for each buyer

Take-it-or-leave-it sales

- Solution: *bundle* items together

Buyers don't know the actual realization

- *Symmetric* bundling: all bundles are the same for all buyers
- *Asymmetric* bundling: different bundles for each buyer

Take-it-or-leave-it sales

- Solution: *bundle* items together

Buyers don't know
the actual realization

- *Symmetric* bundling: all bundles are the same for all buyers
- *Asymmetric* bundling: different bundles for each buyer

Information asymmetry

- *The seller has more info than the buyers*
- Akerlof (1970): the market for lemons
  - Nobel prize in Economics (2001)
- Crawford and Sobel (1982): can the seller exploit this info?
- Milgrom and Weber (1982): the linkage principle
- Ghosh et al. (2007): clustering schemes + second price auction
- Emek et al. (2012): complexity

Asymmetric binary matrix partition (ABMP)

- $A = n \times m$ matrix such that $A_{ij} = \{0,1\}$
- p = probability distribution over the columns of A
- B = *partition scheme* (one partition per row – B_i for row i)
- B_i = contains disjoint sets of columns (*bundles*)

Asymmetric binary matrix partition (ABMP)

- A = n x m matrix such that A_{ij} = {0,1}
- p = probability distribution over the columns of A
- B = *partition scheme* (one partition per row – B_i for row i)
- B_i = contains disjoint sets of columns (*bundles*)

A =

0	0	0	1
1	0	0	0
0	0	1	0
1	0	1	1

Asymmetric binary matrix partition (ABMP)

- $A = n \times m$ matrix such that $A_{ij} = \{0,1\}$
- p = probability distribution over the columns of A
- B = *partition scheme* (one partition per row – B_i for row i)
- B_i = contains disjoint sets of columns (*bundles*)

$$A = \begin{array}{cccc} 10\% & 20\% & 25\% & 45\% \\ \end{array}$$

10%	20%	25%	45%
0	0	0	1
1	0	0	0
0	0	1	0
1	0	1	1

Asymmetric binary matrix partition (ABMP)

- A = n x m matrix such that A_{ij} = {0,1}
- p = probability distribution over the columns of A
- B = *partition scheme* (one partition per row – B_i for row i)
- B_i = contains disjoint sets of columns (*bundles*)

$$A = \begin{array}{cccc} \mathbf{10\%} & \mathbf{20\%} & \mathbf{25\%} & \mathbf{45\%} \\ \left[\begin{array}{c} 0 \\ 1 \\ 0 \\ 1 \end{array}\right. & \begin{array}{c} 0 \\ 0 \\ 0 \\ 0 \end{array} & \begin{array}{c} 0 \\ 0 \\ 1 \\ 1 \end{array} & \left.\begin{array}{c} 1 \\ 0 \\ 0 \\ 1 \end{array}\right] \end{array}$$

Asymmetric binary matrix partition (ABMP)

- A = n x m matrix such that A_{ij} = {0,1}
- p = probability distribution over the columns of A
- B = *partition scheme* (one partition per row – B_i for row i)
- B_i = contains disjoint sets of columns (*bundles*)

$$A = \begin{array}{cccc} 10\% & 20\% & 25\% & 45\% \\ \hline 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 1 \end{array}$$

Asymmetric binary matrix partition (ABMP)

- A^B = *smooth matrix* induced by applying B to A
- $A_{ij}{}^B$ = *smooth value* of element A_{ij}

$$A = \begin{array}{cccc} \mathbf{10\%} & \mathbf{20\%} & \mathbf{25\%} & \mathbf{45\%} \\ \hline 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 1 \end{array}$$

Asymmetric binary matrix partition (ABMP)

- A^B = *smooth matrix* induced by applying B to A
- $A_{ij}{}^B$ = *smooth value* of element A_{ij}

$$A_{ij}^B = \frac{\sum_{\ell \in B_{ik}} p_\ell \cdot A_{i\ell}}{\sum_{\ell \in B_{ik}} p_\ell}$$

$$A = \begin{array}{cccc} 10\% & 20\% & 25\% & 45\% \\ \hline 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 1 \end{array}$$

Asymmetric binary matrix partition (ABMP)

- A^B = *smooth matrix* induced by applying B to A
- $A_{ij}{}^B$ = *smooth value* of element A_{ij}

$$A^B_{ij} = \frac{\sum_{\ell \in B_{ik}} p_\ell \cdot A_{i\ell}}{\sum_{\ell \in B_{ik}} p_\ell}$$

$$A = \begin{array}{cccc} 10\% & 20\% & 25\% & 45\% \\ \hline 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 1 \end{array}$$

$$A^B_{41} = \frac{10\% \times 1 + 20\% \times 0}{10\% + 20\%} = 0.33$$

$$A^B_{23} = \frac{10\% \times 1 + 20\% \times 0 + 25\% \times 0}{10\% + 20\% + 25\%} = 0.18$$

Asymmetric binary matrix partition (ABMP)

- A^B = *smooth matrix* induced by applying B to A
- A_{ij}^B = *smooth value* of element A_{ij}

$$A_{ij}^B = \frac{\sum_{\ell \in B_{ik}} p_\ell \cdot A_{i\ell}}{\sum_{\ell \in B_{ik}} p_\ell}$$

A =

10%	20%	25%	45%
0	0	0	1
1	0	0	0
0	0	1	0
1	0	1	1

10%	20%	25%	45%
0	0.5	0.5	0.5
0.18	0.18	0.18	0
0.25	0.25	0.25	0.25
0.33	0.33	1	1

= A^B

Asymmetric binary matrix partition (ABMP)

- $v^B(A,p)$ = *partition value* of B

$$A = \begin{array}{cccc} 10\% & 20\% & 25\% & 45\% \\ \hline 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 1 \end{array}$$

$$\begin{array}{cccc} 10\% & 20\% & 25\% & 45\% \\ \hline 0 & 0.5 & 0.5 & 0.5 \\ 0.18 & 0.18 & 0.18 & 0 \\ 0.25 & 0.25 & 0.25 & 0.25 \\ 0.33 & 0.33 & 1 & 1 \end{array} = A^B$$

Asymmetric binary matrix partition (ABMP)

- $v^B(A,p)$ = *partition value* of B

$$v^B(A,p) = \sum_{j \in [m]} p_j \cdot \max_i A^B_{ij}$$

$$A = \begin{array}{cccc} 10\% & 20\% & 25\% & 45\% \\ \end{array}$$

10%	20%	25%	45%
0	0	0	1
1	0	0	0
0	0	1	0
1	0	1	1

10%	20%	25%	45%
0	0.5	0.5	0.5
0.18	0.18	0.18	0
0.25	0.25	0.25	0.25
0.33	0.33	1	1

$= A^B$

Asymmetric binary matrix partition (ABMP)

- $v^B(A,p)$ = *partition value* of *B*

$$v^B(A,p) = \sum_{j \in [m]} p_j \cdot \max_i A^B_{ij}$$

$$A = \begin{array}{cccc} 10\% & 20\% & 25\% & 45\% \\ \end{array}$$

10%	20%	25%	45%
0	0	0	1
1	0	0	0
0	0	1	0
1	0	1	1

10%	20%	25%	45%
0	0.5	0.5	0.5
0.18	0.18	0.18	0
0.25	0.25	0.25	0.25
0.33	0.33	1	1

$= A^B$

Asymmetric binary matrix partition (ABMP)

- $v^B(A,p)$ = *partition value* of B

$$v^B(A,p) = \sum_{j \in [m]} p_j \cdot \max_i A^B_{ij}$$

$$v^B(A,p) = (10\% \times 0.33) + (20\% \times 0.5) + (25\% \times 1) + (45\% \times 1) = 0.83$$

$A =$

10%	20%	25%	45%
0	0	0	1
1	0	0	0
0	0	1	0
1	0	1	1

10%	20%	25%	45%
0	0.5	0.5	0.5
0.18	0.18	0.18	0
0.25	0.25	0.25	0.25
0.33	0.33	1	1

$= A^B$

Asymmetric binary matrix partition (ABMP)

The problem

Compute a partition scheme with maximum partition value

- Introduced by Alon et al. (2013)
- APX–hard (*this is really interesting!*)
- 0.563–approximation for uniform distribution
- 1/13–approximation for non-uniform distributions

Interlude: approximation algorithms

- Consider an NP-hard maximization problem with some objective function

A poly-time algorithm is $1/\rho$–approximate if and only if

$$\mathbf{ALG \geq 1/\rho \cdot OPT}$$

where ALG is the objective value of the solution returned by the algorithm and OPT is the maximum objective value

A problem is hard to approximate (APX-hard) within a factor of $1/\rho$ iff it is NP-hard to design a $1/\rho$–approximation algorithm for it

Our results

- Improved approximation algorithms for both cases

Theorem
There is a 9/10–approximate algorithm for the uniform case

Theorem
There is a (1-1/e)–approximate algorithm for non-uniform case distributions

Our results

- Improved approximation algorithms for both cases

<u>Theorem</u>
There is a 9/10–approximate algorithm for the uniform case

Combinatorial algorithm; analysis using LPs

<u>Theorem</u>
There is a (1-1/e)–approximate algorithm for non-uniform case distributions

Our results

- Improved approximation algorithms for both cases

Theorem
There is a 9/10–approximate algorithm for the uniform case

Combinatorial algorithm; analysis using LPs

Theorem
There is a (1-1/e)–approximate algorithm for non-uniform case distributions

Reduction to welfare maximization

The uniform case

Greedy algorithm
Cover phase: compute a full cover of the one-columns
Greedy phase:

- Consider the zero-columns one by one
- Add the next zero-column to the bundle that maximizes its marginal contribution

- Marginal contribution of a zero-column when it is added to a mixed bundle that consists of x zero-columns and y one-columns:

$$\Delta(x,y) = (x+1)\frac{y}{x+y+1} - x\frac{y}{x+y}$$

The uniform case

<u>Greedy algorithm</u>

Cover phase: compute a full cover of the one-columns

Greedy phase:

- Consider the zero-columns one by one
- Add the next zero-column to the bundle that maximizes its marginal contribution

25%	25%	25%	25%
0	1	1	0
1	0	0	0
0	1	0	0
1	1	1	0

- Marginal contribution of a zero-column when it is added to a mixed bundle that consists of *x* zero-columns and *y* one-columns:

$$\Delta(x, y) = (x+1)\frac{y}{x+y+1} - x\frac{y}{x+y}$$

The uniform case

Greedy algorithm

Cover phase: compute a full cover of the one-columns

Greedy phase:

- Consider the zero-columns one by one
- Add the next zero-column to the bundle that maximizes its marginal contribution

25%	25%	25%	25%
0	1	1	0
1	0	0	0
0	1	0	0
1	1	1	0

- Marginal contribution of a zero-column when it is added to a mixed bundle that consists of x zero-columns and y one-columns:

$$\Delta(x,y) = (x+1)\frac{y}{x+y+1} - x\frac{y}{x+y}$$

The uniform case

Greedy algorithm

Cover phase: compute a full cover of the one-columns

Greedy phase:

- Consider the zero-columns one by one
- Add the next zero-column to the bundle that maximizes its marginal contribution

25%	25%	25%	25%
0	1	1	0
1	0	0	0
0	1	0	0
1	1	1	0

- Marginal contribution of a zero-column when it is added to a mixed bundle that consists of x zero-columns and y one-columns:

$$\Delta(x,y) = (x+1)\frac{y}{x+y+1} - x\frac{y}{x+y}$$

The uniform case

<u>Greedy algorithm</u>
Cover phase: compute a full cover of the one-columns
Greedy phase:
- Consider the zero-columns one by one
- Add the next zero-column to the bundle that maximizes its marginal contribution

25%	25%	25%	25%
0	1	1	0
1	0	0	0
0	1	0	0
1	1	1	0

- Marginal contribution of a zero-column when it is added to a mixed bundle that consists of x zero-columns and y one-columns:

$$\Delta(x,y) = (x+1)\frac{y}{x+y+1} - x\frac{y}{x+y}$$

The uniform case

Greedy algorithm

Cover phase: compute a full cover of the one-columns

Greedy phase:

- Consider the zero-columns one by one
- Add the next zero-column to the bundle that maximizes its marginal contribution

25%	25%	25%	25%
0	1	1	0
1	0	0	0
0	1	0	0
1	1	1	0

- Marginal contribution of a zero-column when it is added to a mixed bundle that consists of x zero-columns and y one-columns:

$$\Delta(x, y) = (x+1)\frac{y}{x+y+1} - x\frac{y}{x+y}$$

The uniform case

Greedy algorithm

Cover phase: compute a full cover of the one-columns

Greedy phase:

- Consider the zero-columns one by one
- Add the next zero-column to the bundle that maximizes its marginal contribution

25%	25%	25%	25%
0	1	1	0
1	0	0	0
0	1	0	0
1	1	1	0

- Marginal contribution of a zero-column when it is added to a mixed bundle that consists of x zero-columns and y one-columns:

$$\Delta(x,y) = (x+1)\frac{y}{x+y+1} - x\frac{y}{x+y}$$

The uniform case

Theorem
There is an instance of uniform ABMP for which the greedy algorithm computes a partition scheme with value at most 9/10 of the optimal one

The uniform case

Theorem
There is an instance of uniform ABMP for which the greedy algorithm computes a partition scheme with value at most 9/10 of the optimal one

25%	25%	25%	25%
1	0	0	0
0	1	0	0
1	1	0	0
1	1	0	0

The uniform case

25%	25%	25%	25%
1	0	0	0
0	1	0	0
1	1	0	0
1	1	0	0

Theorem

There is an instance of uniform ABMP for which the greedy algorithm computes a partition scheme with value at most 9/10 of the optimal one

Optimal solution

Greedy algorithm

The uniform case

Theorem
There is an instance of uniform ABMP for which the greedy algorithm computes a partition scheme with value at most 9/10 of the optimal one

25%	25%	25%	25%
1	0	0	0
0	1	0	0
1	1	0	0
1	1	0	0

Optimal solution

25%	25%	25%	25%
1	0	0	0
0	1	0	0
1	1	0	0
1	1	0	0

$$\mathrm{OPT} = \frac{5}{6}$$

Greedy algorithm

The uniform case

Theorem
There is an instance of uniform ABMP for which the greedy algorithm computes a partition scheme with value at most 9/10 of the optimal one

25%	25%	25%	25%
1	0	0	0
0	1	0	0
1	1	0	0
1	1	0	0

Optimal solution

25%	25%	25%	25%
1	0	0	0
0	1	0	0
1	1	0	0
1	1	0	0

$$\mathrm{OPT} = \frac{5}{6}$$

Greedy algorithm

25%	25%	25%	25%
1	0	0	0
0	1	0	0
1	1	0	0
1	1	0	0

$$\mathrm{ALG} = \frac{3}{4}$$

The uniform case

Theorem
There is an instance of uniform ABMP for which the greedy algorithm computes a partition scheme with value at most 9/10 of the optimal one

25%	25%	25%	25%
1	0	0	0
0	1	0	0
1	1	0	0
1	1	0	0

Optimal solution

25%	25%	25%	25%
1	0	0	0
0	1	0	0
1	1	0	0
1	1	0	0

$$\mathrm{OPT} = \frac{5}{6}$$

Greedy algorithm

25%	25%	25%	25%
1	0	0	0
0	1	0	0
1	1	0	0
1	1	0	0

$$\mathrm{ALG} = \frac{3}{4}$$

$$\frac{\mathrm{ALG}}{\mathrm{OPT}} = \frac{9}{10}$$

Non-uniform distributions

- Partition scheme = a collection of disjoint sets S_i (one per row) such that S_i contains the columns that achieve maximum smooth value at row i
- Partition value = total contribution of all rows (through the columns S_i)

Non-uniform distributions

- Partition scheme = a collection of disjoint sets S_i (one per row) such that S_i contains the columns that achieve maximum smooth value at row i
- Partition value = total contribution of all rows (through the columns S_i)

10%	20%	25%	45%
0	0	0	1
1	0	0	0
0	0	1	0
1	0	1	1

10%	20%	25%	45%
0	0.5	0.5	0.5
0.18	0.18	0.18	0
0.25	0.25	0.25	0.25
0.33	0.33	1	1

Non-uniform distributions

- Partition scheme = a collection of disjoint sets S_i (one per row) such that S_i contains the columns that achieve maximum smooth value at row i
- Partition value = total contribution of all rows (through the columns S_i)

$$S_1 = \{2\} \qquad S_2 = \emptyset \qquad S_3 = \emptyset \qquad S_4 = \{1, 3, 4\}$$

10%	20%	25%	45%
0	0	0	1
1	0	0	0
0	0	1	0
1	0	1	1

10%	20%	25%	45%
0	0.5	0.5	0.5
0.18	0.18	0.18	0
0.25	0.25	0.25	0.25
0.33	0.33	1	1

Non-uniform distributions

Observation
The problem is equivalent to *welfare maximization* where (agents = rows) and (bundles of items = sets of columns)

Lemma
The contribution of each row is a submodular function

- The smooth greedy algorithm of Vondrák (2008) is (1-1/e)–approximate for submodular welfare maximization

Open problems

- Better approximation algorithms
- The general case: non-binary values
    - Alon et al.(2013) proved logarithmic approximation guarantees, can we do better?

Open problems

- Better approximation algorithms
- The general case: non-binary values
  - Alon et al.(2013) proved logarithmic approximation guarantees, can we do better?

Thank you!