

Approximate Modularity Revisited

Inbal Talgam-Cohen → Technion

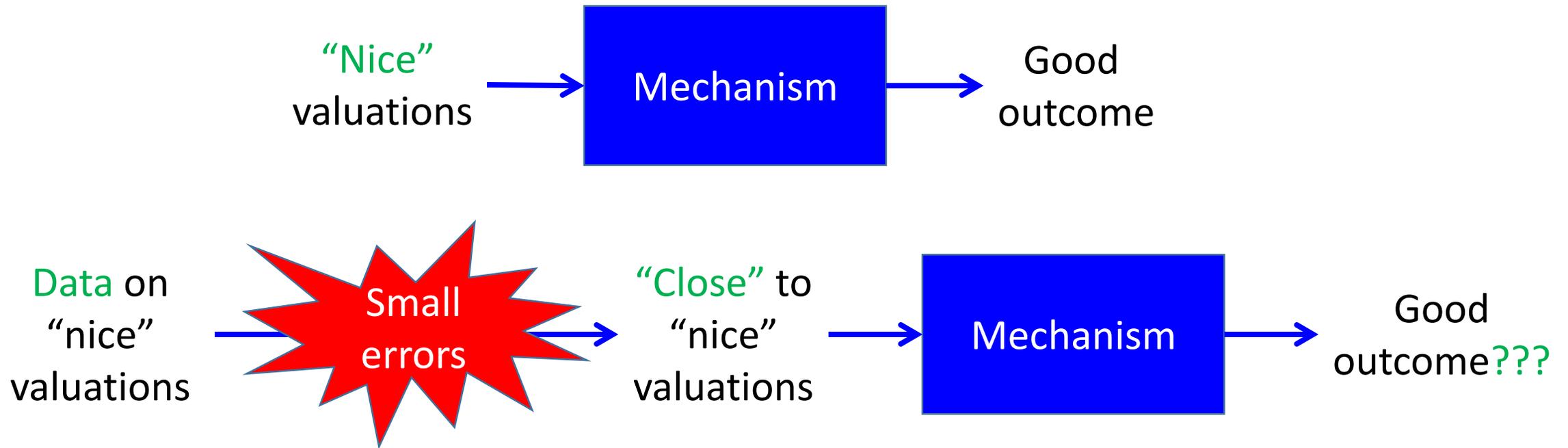
Joint work with Uriel Feige, Michal Feldman

STOC'17, HALG'18

*Part of this work was done at MSR Herzliya

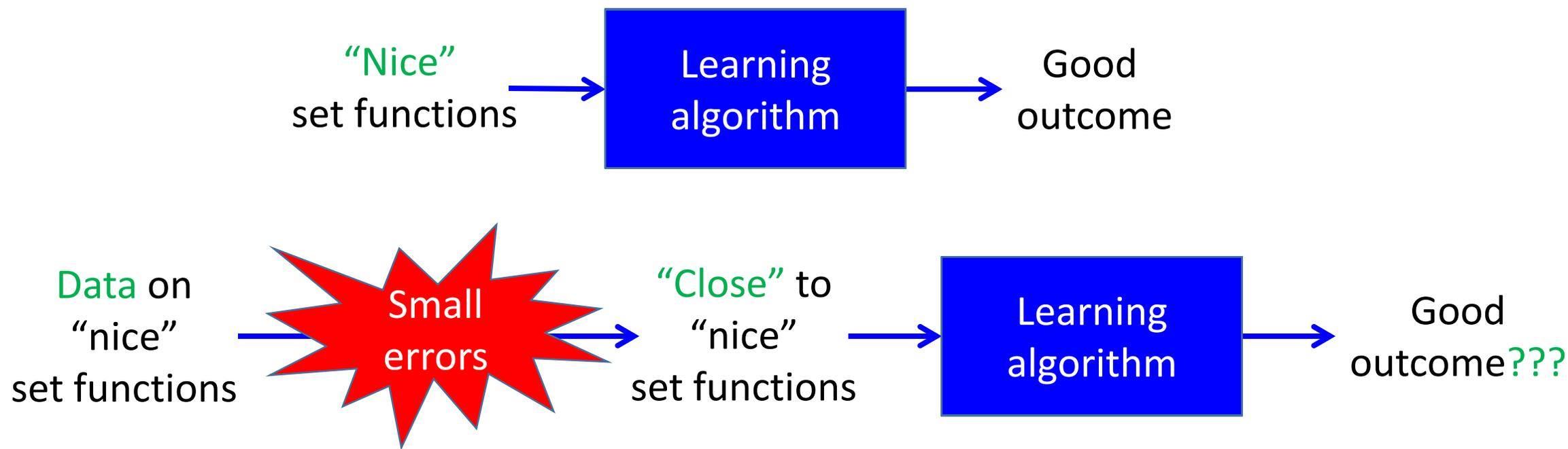
Motivation: Robustness/stability

Mechanism and market design as a “data-driven enterprise”.



Motivation: Robustness/stability

Mechanism and market design as a “data-driven enterprise”.



High-level research agenda

What happens when set functions are only “close” to being “nice”?

Many sub-questions:

- **Notions** of closeness.
- Optimization – can good **approximation** ratios still be achieved?
- Mechanism design – do good **economic** properties continue to hold?
- **Learning** – can nice set functions be recovered?
- One **take-away**: Basic questions still open; interesting math involved

Set functions

Universe of n items.

Function $f: \{0,1\}^n \rightarrow \mathbb{R}$ assigns values $f(S)$ to sets of items.

Examples:

- Items for sale in a **combinatorial auction**.
Value of set: willingness of a bidder to **pay** for bundle of items.
- Items are **vertices** in an edge-weighted graph.
Value of set: total weight of edges in induced **cut**.

Representation of set functions

Explicit: 2^n entries.

Value queries: upon query S learn $f(S)$.

- In a combinatorial auction, one may ask a bidder how much she is willing to pay for the bundle.
- Given a set S of vertices, one can compute in polytime the total weight of edges in the cut (S, \bar{S}) .

Other types of queries (such as demand queries) have been studied.

Classes of “nice” set functions

Some classes of nice set functions with polynomial representations:

Additive $f(S) = \sum_{i \in S} f(i)$, and $f(\emptyset) = 0$.

Linear $f(S) = f(\emptyset) + \sum_{i \in S} f(i)$.

Capped-additive $f(S) = \min\{1, g(S)\}$ where g is additive.

Nice properties but not necessarily a polynomial representation:

Submodular $f(S) + f(T) \geq f(S \cap T) + f(S \cup T)$.

Subadditive $f(S) + f(T) \geq f(S \cup T)$, and $f(\emptyset) = 0$.

Optimization problems for “nice” set functions

Linear set functions:

- Many optimization problems are **easy**.

Submodular set functions (cuts, valuations with diminishing marginals):

- Maximization subject to constraints (e.g., a cardinality constraint) can be solved **approximately**.

General set functions:

- Many optimization problems are computationally **hard** to approximate.

Motivations for “close to” nice set functions

- Answers to value queries might not be exact due to **noise** in measurements.
- **Rounding** errors.
- Computing approximate values may be **cheaper** than computing exact values.
- The functions might not be exactly nice (e.g., valuation functions of bidders need not be exactly submodular).

Some related work

Functions on **continuous domains** (rather than discrete hypercube):

- Hyers [1941], ...

Data-driven/sample-based optimization:

- Bertsimas and Thiele [2014], Singer and Vondrak [2015], Hassidim and Singer [2016], Balkanski, Rubinstein, and Singer [2016], ...

Approximate submodularity/convexity/substitutes:

- Lehmann, Lehmann, and Nisan [2006], Das and Kempe [2011], Belloni, Liang, Narayanan, and Rakhlin [2015], Roughgarden, T.C., and Vondrak [2016], ...

Learning submodular functions:

- Balcan and Harvey [2011], ...

...

Our Results

We follow-up on Chierichetti et al. [FOCS'15]

They suggested to start with an “easy” case: a function f close to **linear**.

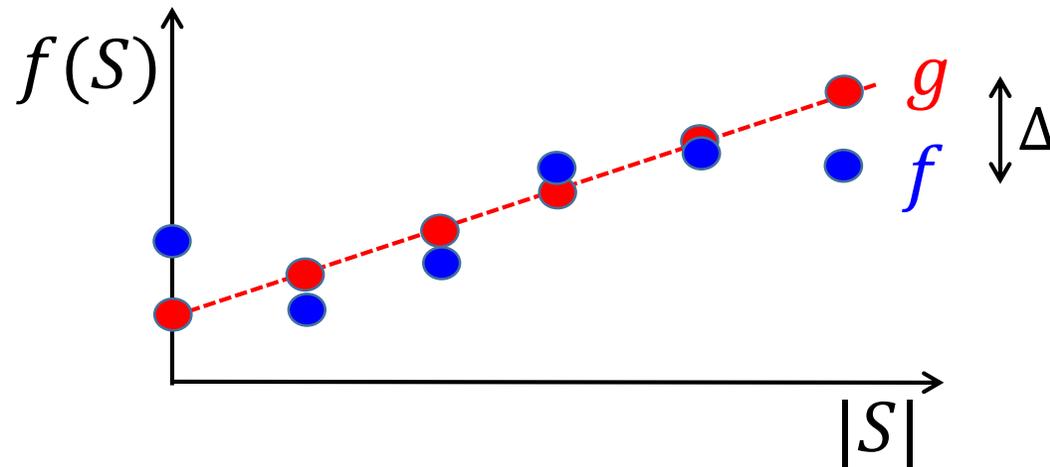
Posed 2 questions:

- To what extent are different measures of closeness to linearity **related** to each other?
 - being **pointwise** close to a linear function,
 - nearly satisfying **properties** of linear functions.
- How to **learn** a linear function h that is close to f ?

Definition: Δ -linear set functions

Linear set function: $g(S) = g(\emptyset) + \sum_{i \in S} g(i)$.

Given $\Delta \geq 0$, a set function f is Δ -linear if there is a linear set function g such that $|f(S) - g(S)| \leq \Delta$ pointwise for every set S .



Definition: ε -modular set functions

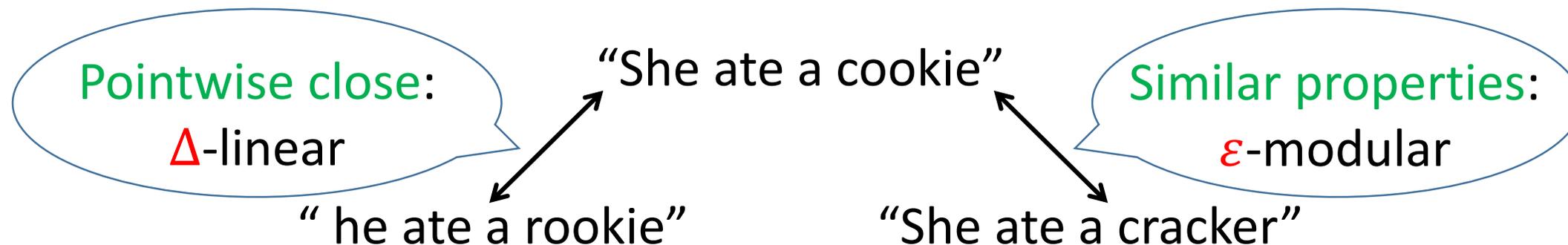
A set function is **linear** if and only if it is **modular**:

$$f(S) + f(T) = f(S \cap T) + f(S \cup T) \text{ for all sets } S \text{ and } T.$$

A set function is **ε -modular** if

$$|f(S) + f(T) - f(S \cap T) - f(S \cup T)| \leq \varepsilon \text{ for all sets } S \text{ and } T.$$

“Syntactic” versus “semantic” closeness



Relating the closeness notions

Recall a set function is ε -modular if

$$|f(S) + f(T) - f(S \cap T) - f(S \cup T)| \leq \varepsilon \text{ for all sets } S \text{ and } T.$$

Every Δ -linear function is ε -modular for $\varepsilon \leq 4\Delta$.

Is it true that every ε -modular function is Δ -linear for $\Delta \leq O(\varepsilon)$?

Δ -linear: pointwise close to linear up to $\pm\Delta$
 ε -modular: satisfies modularity eqs. up to $\pm\varepsilon$

Results for ε -modularity

Assaf Naor directed us to [previous work](#) from a different community [Kalton and Roberts; 1983] showing ε -modular is Δ -linear for:

- $\Delta \leq 44.5\varepsilon$

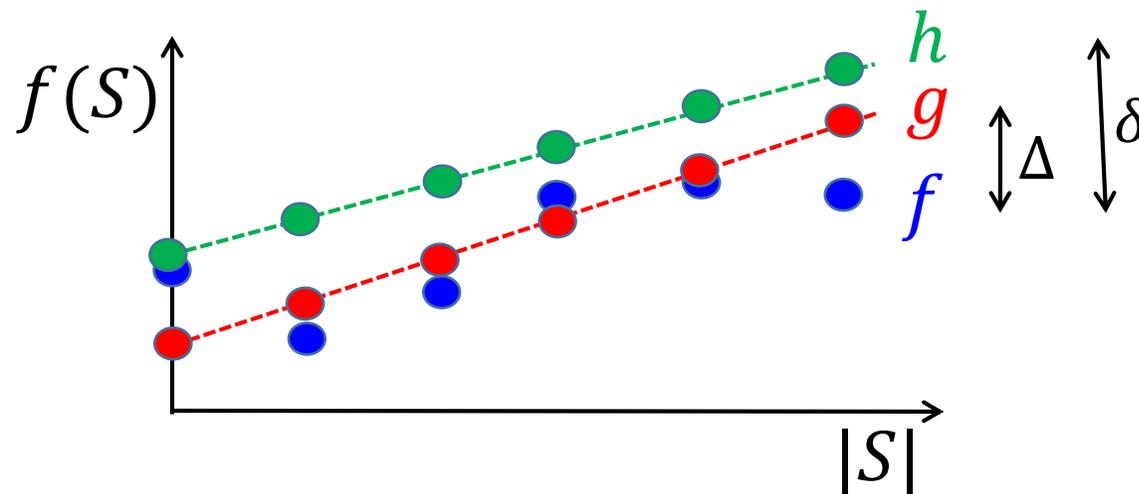
We adapt techniques from [KR'83] and [Chierichetti+'15] to show:

- $\Delta < 13\varepsilon$, with better bounds for special classes of set functions.
 - E.g., $\Delta \leq \frac{1}{2}\varepsilon$ for [symmetric](#) functions.
- $\Delta \geq \varepsilon$ for some set function (with $n = 70$).

Learning Δ -linear set functions

Suppose we are given value query access to a Δ -linear function f . Using polynomially many value queries, output a linear function h satisfying $|f(S) - h(S)| \leq \delta$ for every set S .

How small can we make δ as a function of Δ and n ?



Results for learning Δ -linear set functions

[Chierichetti et al.]:

- A **randomized** algorithm making $O(n^2 \log n)$ nonadaptive queries and achieving $\delta \leq O(\Delta\sqrt{n})$ w.h.p.

Our result [similar technique as Dworj-Yekhanin'08]:

- A **deterministic** algorithm making $O(n)$ nonadaptive queries and achieving $\delta \leq O(\Delta\sqrt{n})$.

Lower bound [Chierichetti et al.]: $\delta \geq \Omega(\Delta \sqrt{\frac{n}{\log n}})$ even with adaptivity.

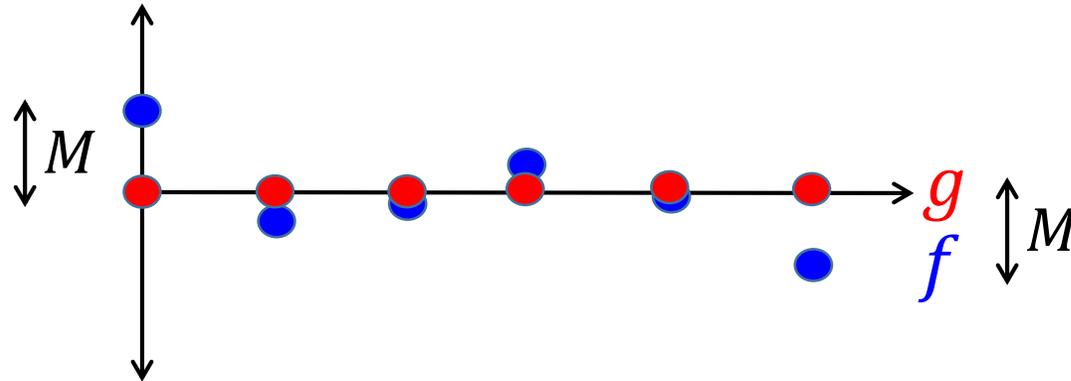
→ Algorithm useful when f is very close to linear (e.g. rounding errors).

Proof Overview

What we want to show

Δ -linear: pointwise close to linear up to $\pm\Delta$
 ϵ -modular: satisfies modularity eqs. up to $\pm\epsilon$
Theorem: $\Delta < 13\epsilon$

W.l.o.g. f is **1**-modular and has $g = 0$ as its closest linear function.



Goal: Bound **extreme value** $M (= \Delta)$.

Main proof steps

1. Define a way to rearrange a collection \mathcal{S} of sets: “split-and-merge”
2. Split-and-merge $\mathcal{S} \subseteq \{S \mid f(S) = M\}$ to bound M using 1-modularity
3. Wait! Can we split-and-merge \mathcal{S} ?
 - Yes if \mathcal{S} is sparse by existence of bipartite expander graphs
4. Construct sparse \mathcal{S}
5. Optimize construction to improve bound on M

Main proof steps

1. Define a way to rearrange a collection \mathcal{S} of sets: “split-and-merge”
2. Split-and-merge $\mathcal{S} \subseteq \{S \mid f(S) = M\}$ to bound M using 1-modularity
3. Wait! Can we split-and-merge \mathcal{S} ?
 - Yes if \mathcal{S} is sparse by existence of bipartite expander graphs
4. Construct sparse \mathcal{S}
5. Optimize construction to improve bound on M

Step 1: (r, θ) -split-and-merge

The wedding planner's problem:

- k invited families, only θk tables ($\theta < 1$).
- Each person has trait: “funny”, “shy”, ...
 - Traits do not repeat within family.
- $r k$ seating cards ($r > 1$) direct family members to their table.

An (r, θ) -split-and-merge of the k families exists if there are seating cards s.t. traits not repeated within the same table.

Smaller $r, \theta \rightarrow$ problem more constrained.

Example of (r, θ) -split-and-merge

$k = 3$ families, $rk = 6$ seating cards, $\theta k = 2$ tables

(1,2,3,4,7)
 

(1,2,5,6)


(3,4,5,7)


(1,2,7) (3,4)
 

(1,2) (5,6)
 

(3,4) (5,7)
 

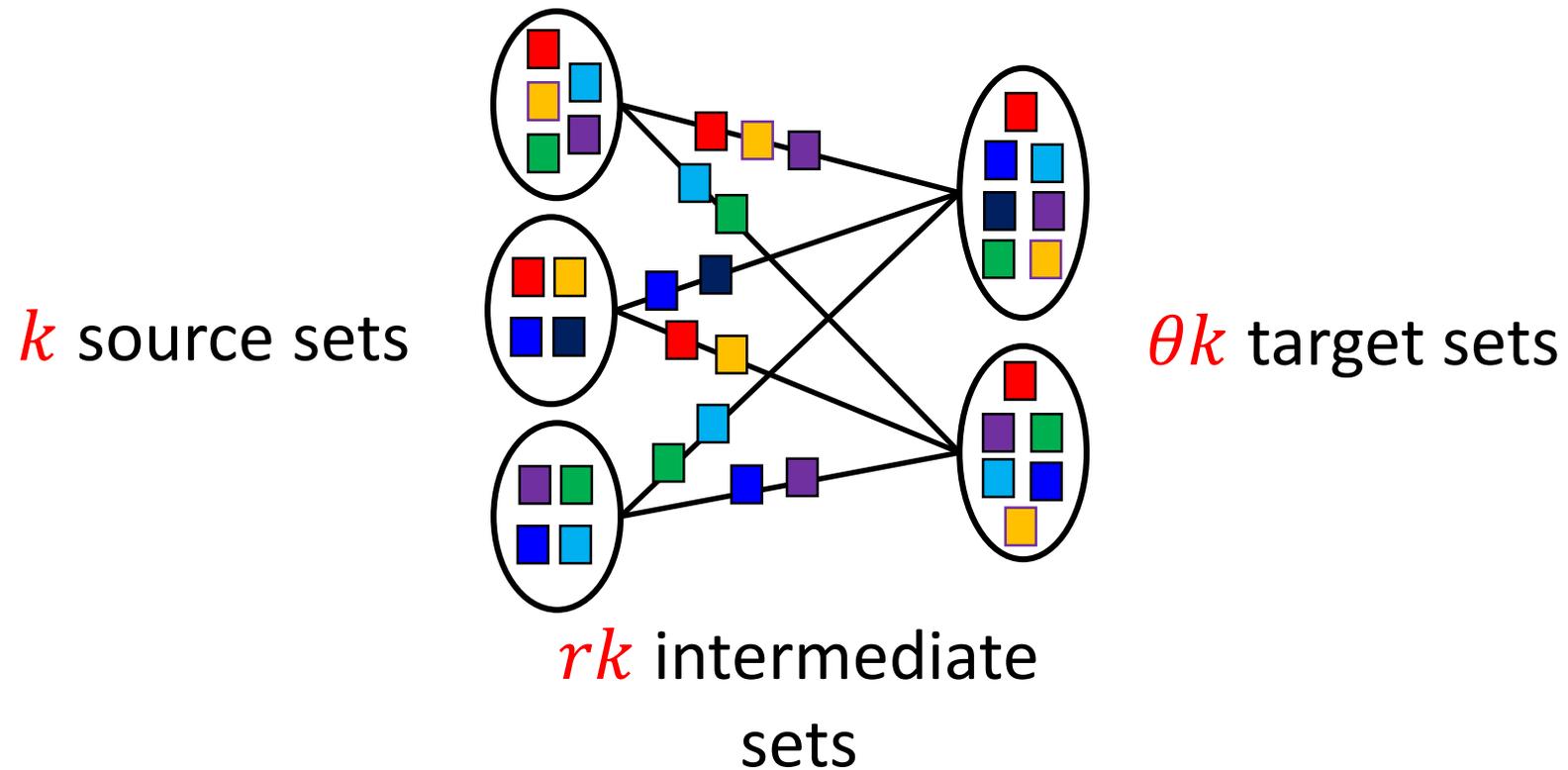
(1,2,7,5,6,3,4)



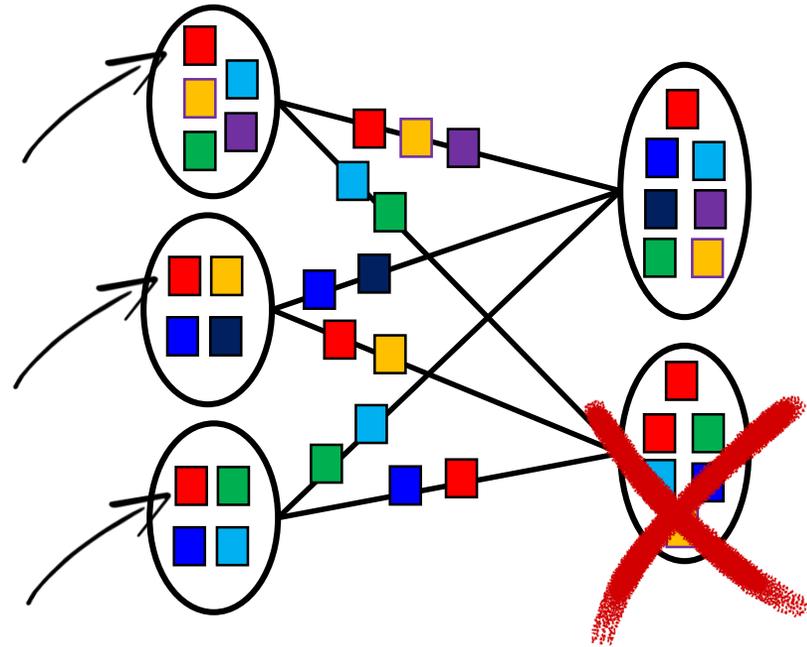
(3,4,1,2,5,7)



Split-and-merge as a bipartite graph



Split-and-merge not always possible



Problem: Source sets not sparse.

Main proof steps

1. Define a way to rearrange a collection \mathcal{S} of sets: “split-and-merge”
2. Split-and-merge $\mathcal{S} \subseteq \{S \mid f(S) = M\}$ to bound M using 1-modularity
3. Wait! Can we split-and-merge \mathcal{S} ?
 - Yes if \mathcal{S} is sparse by existence of bipartite expander graphs
4. Construct sparse \mathcal{S}
5. Optimize construction to improve bound on M

1-modularity implication

Let S be disjoint union of r sets I_1, \dots, I_r .

By 1-modularity of f :

$$\sum_{j \leq r} f(I_j) = f(S) \pm (r - 1).$$

Step 2: Applying split-and-merge to bound M

Idea: Use **intermediate** sets to relate values of **source**, **target** sets

Source set S is split to **intermediate** sets I_1, \dots, I_r :

$$\sum_{j \leq r} f(I_j) \geq f(S) - r + 1. (*)$$

Target set T is merged from **intermediate** sets $I_1, \dots, I_{r/\theta}$:

$$\sum_{j \leq r/\theta} f(I_j) \leq f(T) + r/\theta - 1. (**)$$

Summing up (*), (**) over all **source** and **target** sets:

$$\sum_{j \leq k} f(S_j) - rk + k \leq \sum_{j \leq \theta k} f(T_j) + rk - \theta k.$$

Step 2: Applying split-and-merge to bound M

Applying to source sets of value $f(S_j) = M$:

$$\rightarrow kM - rk + k \leq \theta kM + rk - \theta k \rightarrow M \leq \frac{2r - 1 - \theta}{1 - \theta}.$$

Main proof steps

1. Define a way to rearrange a collection \mathcal{S} of sets: “split-and-merge”
2. Split-and-merge $\mathcal{S} \subseteq \{S \mid f(S) = M\}$ to bound M using 1-modularity
3. Wait! Can we split-and-merge \mathcal{S} ?
 - Yes if \mathcal{S} is sparse by existence of bipartite expander graphs
4. Construct sparse \mathcal{S}
5. Optimize construction to improve bound on M

Step 3: α -sparsity ensures split-and-merge

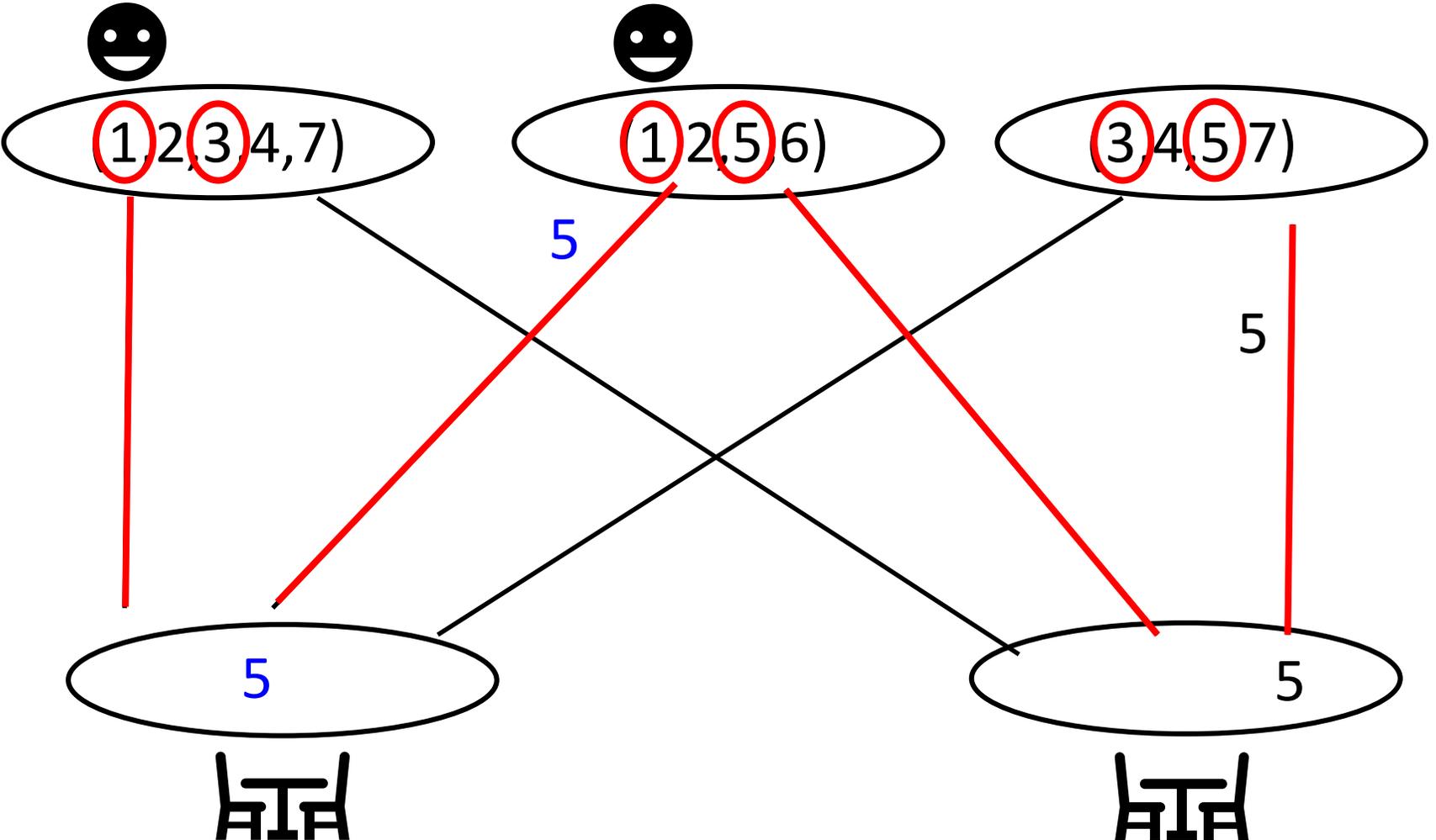
Let $\alpha < 1$.

A collection of k sets is α -sparse if each item appears in $\leq \alpha k$ sets.

Lemma [Kalton and Roberts, 1983]:

Every α -sparse collection has an (r, θ) -split-and-merge, where r, θ depend on parameters of bipartite expander graphs with α -expansion.

Example

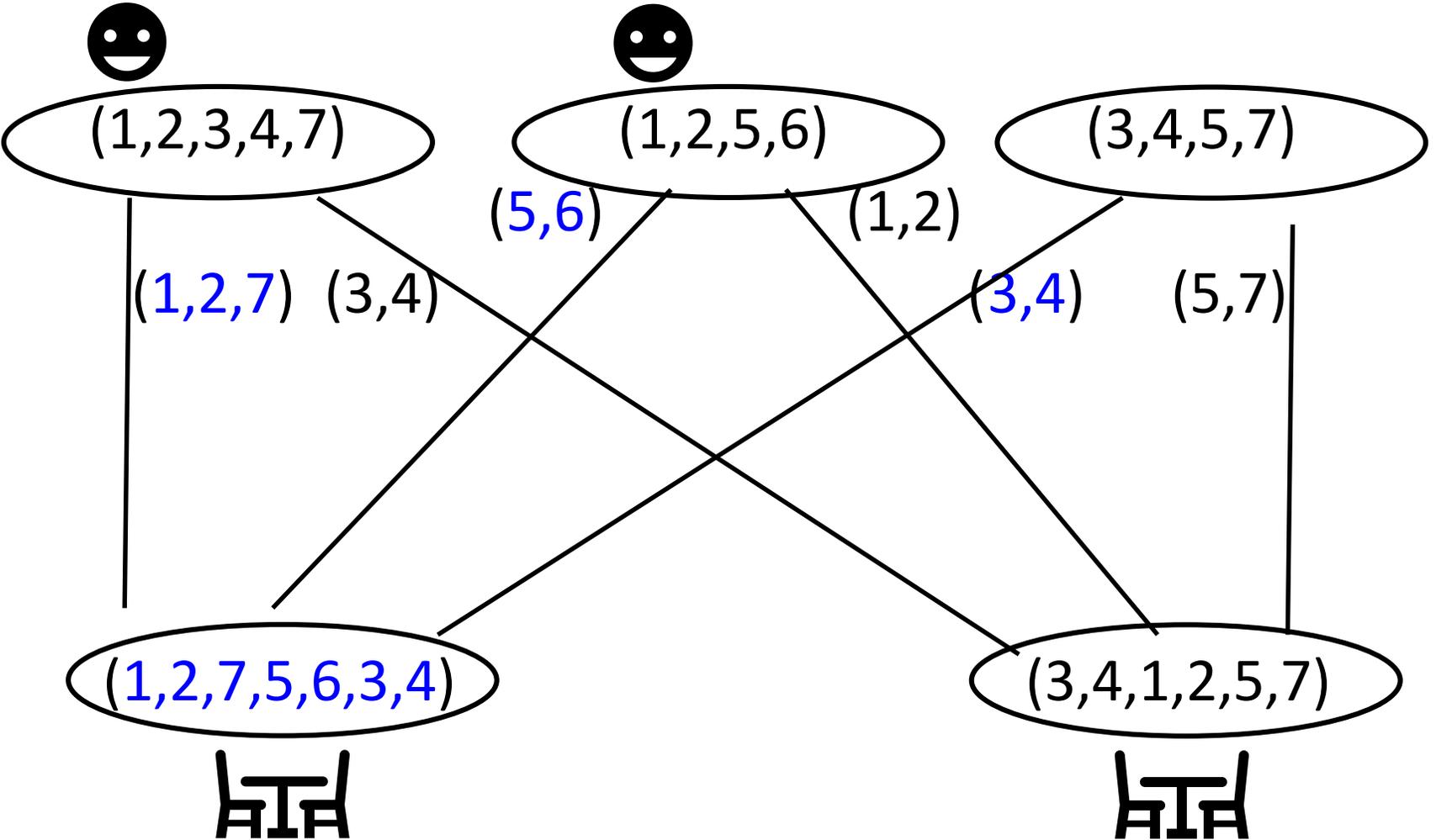


$\frac{2}{3}$ -sparse collection

Expander graph with rk edges:

Every set with $\leq \frac{2}{3}$ of the k top vertices has a **matching** to the θk bottom vertices.

Example



$\frac{2}{3}$ -sparse collection

Expander graph with rk edges:
 Every set with $\leq \frac{2}{3}$ of the k top vertices has a **matching** to the θk bottom vertices.

Main proof steps

1. Define a way to rearrange a collection \mathcal{S} of sets: “split-and-merge”
2. Split-and-merge $\mathcal{S} \subseteq \{S \mid f(S) = M\}$ to bound M using 1-modularity
3. Wait! Can we split-and-merge \mathcal{S} ?
 - Yes if \mathcal{S} is sparse by existence of bipartite expander graphs
4. Construct sparse \mathcal{S}
5. Optimize construction to improve bound on M

Step 4: α -sparse collection with value $\approx M$

Lemma: Every 1-modular function f whose closest linear function is 0 has a $\frac{1}{2}$ -sparse collection with average value $M - d$ s.t. deficit $d \leq 0.5$.

Shown by formulating an LP for closest linear function to f [Chierichetti et al, 2015], and constructing the collection from the dual distributions.

Δ -linear: pointwise close to linear up to $\pm\Delta$
 ϵ -modular: satisfies modularity eqs. up to $\pm\epsilon$

Summary

Relating two notions of “close to” linear functions:

- Every ϵ -modular set function is Δ -linear for $\Delta \leq O(\epsilon)$.
- Current bounds are $\epsilon \leq \Delta < 13\epsilon$.
- Proof based on expander graphs.

Learning (really) close to linear functions

- A linear function h that is $O(\Delta\sqrt{n})$ -close to a Δ -linear function f can be learned by making $O(n)$ value queries non-adaptively.
- Nearly best possible.