Representation, Approximation and Learning of Submodular Functions Using Low-Rank Decision Trees

Vitaly Feldman
IBM Research

Pravesh Kothari*
UT Austin

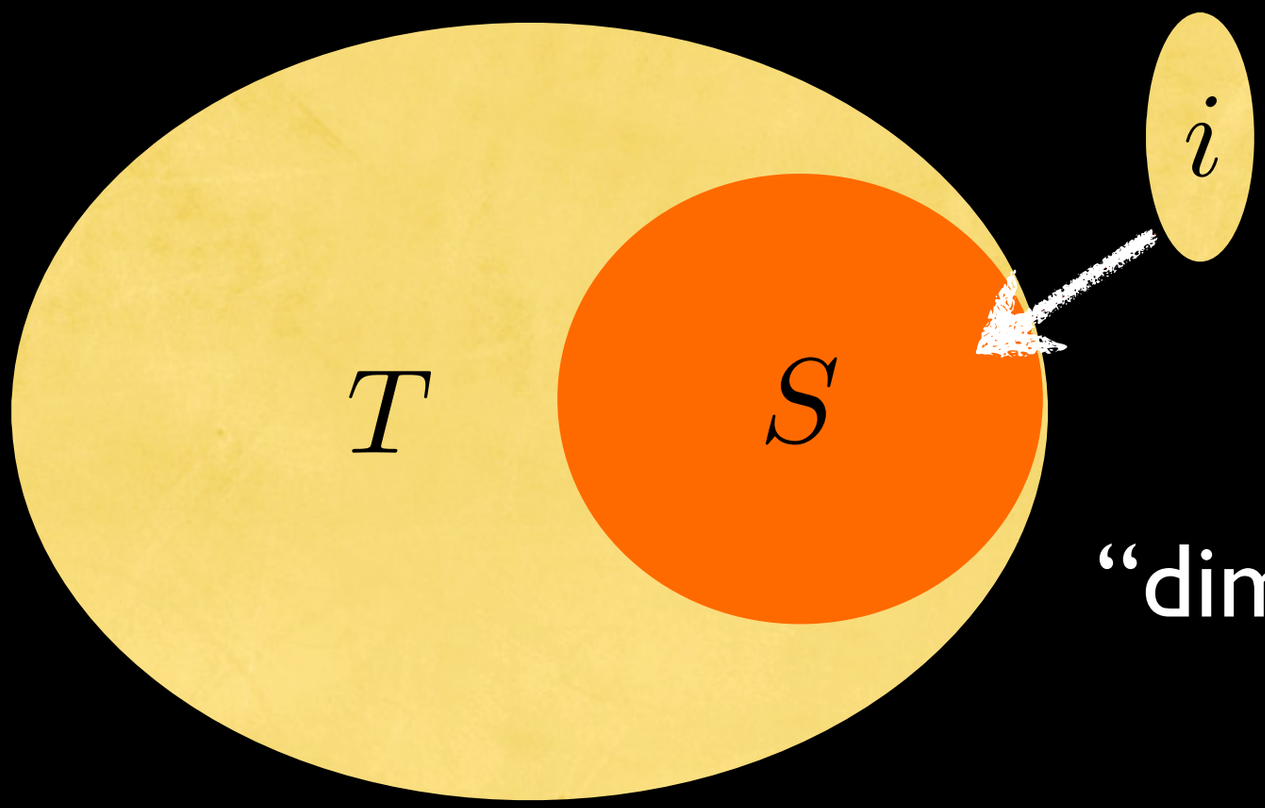
Jan Vondrák
IBM Research

*work done at IBM Research

Submodular Functions

A function $f : 2^{[n]} \to \mathbb{R}$ is *submodular* if:

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



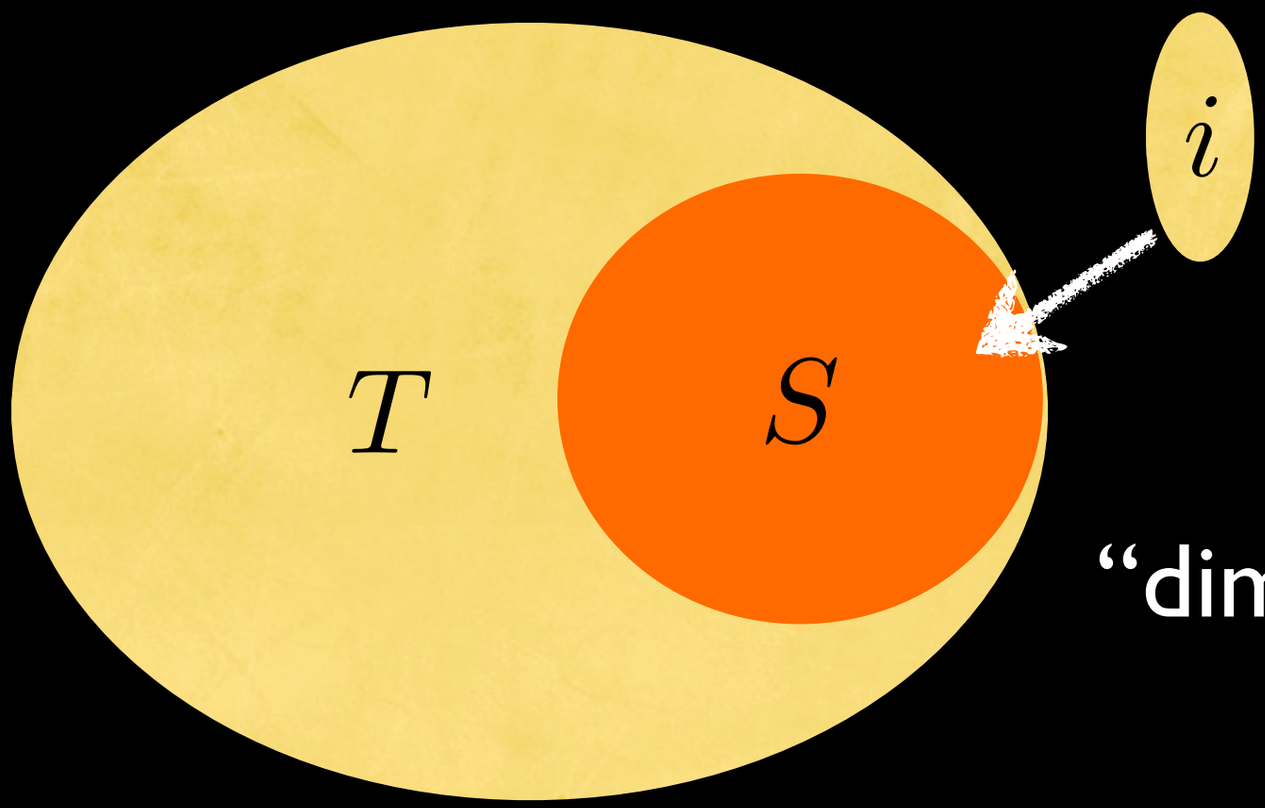
"diminishing marginal returns"

Submodular Functions

$$\{0,1\}^n$$

A function $f : 2^{[n]} \to \mathbb{R}$ is *submodular* if:

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



"diminishing marginal returns"

Submodular Functions

✦Combinatorial Optimization

"discrete analog of convex functions"

- graph cuts Goemans-Williamson-95, Queryanne-95, Fleisher et. al.-01
- rank functions of matroids Edmonds-70, Frank-97
- set covering, Feige-98,
- plant location, sensor placement

Cornuejols et. al.-77,Krause-05, Guestrin-06, -08, Krause-Guestrin-10

✦Algorithmic Game Theory\Economics

- utilities of agents -"diminishing returns"

Lehmann-Nisan-06, Dobzinski et. al.-05,Vondrák-08,
Papadimitriou et. al.-08, Dughmi et. al.-11

Learning Submodular Functions

Balcan-Harvey-11

Motivation: learn and predict

- ✦Pricing\Utility functions.
- ✦Demands of agents.
- ✦Advertisements.

PMAC: Probably *Mostly* Approximately Correct.

*non-negative submodular functions only.

Learning Submodular Functions

Balcan-Harvey-11: PMAC Model

"Probably *Mostly* Approximately Correct"

Learner sees random examples from
the target submodular function f.

Must return a hypothesis h such that

$$\Pr[f(x) \leq h(x) \leq (1+\alpha) f(x)] \geq 1 - \epsilon$$

Learning Submodular Functions

Balcan-Harvey-11: PMAC Model

✓ Arbitrary Distributions: $O(\sqrt{n})$-multiplicative approx. w.p. 1-ϵ in poly time for all submodular functions.

✓ *Lower Bound:* $\Omega(\sqrt[3]{n})$*-multiplicative factor for any poly-time algorithm.*

✓ *Product Distributions:* $\log(1/\epsilon)$-multiplicative approx. w. p. 1-ϵ in poly time for *1-Lipschitz* submodular functions with min-value 1.

Learning Submodular Functions

Gupta-Hardt-Roth-Ullman-11

Learner can make value queries on the target submodular function f^*.

Must return a hypothesis h such that

$$\mathbb{E}[|h(x) - f(x)|] \leq \epsilon$$

*range normalized to [0,1]

Learning Submodular Functions

Gupta-Hardt-Roth-Ullman-11

Learner can make value queries on the target submodular function f.

Must return a hypothesis h such that

$$\mathbb{E}[|h(x) - f(x)|] \leq \epsilon$$

✓ Product Distributions: $n^{O(1/\epsilon^2)}$ time

Learning Submodular Functions

Cheraghchi-Klivans-K-Lee-12

random examples, ℓ_1-error, product distributions.

Time: $n^{O(1/\epsilon^2)}$

✓Works in the *agnostic* setting:

$$\mathbb{E}[|h(x) - f(x)|] \leq opt + \epsilon$$

Learning Submodular Functions

Cheraghchi-Klivans-K-Lee-12

random examples, ℓ_1-error, [illegible]tions.

Tim

$$opt = \min_{submodular\ s} \mathbb{E}[|h(x) - s(x)|]$$

✓Works in the *agnostic* setting:

$$\mathbb{E}[|h(x) - f(x)|] \leq opt + \epsilon$$

Learning Submodular Functions

Raskhodnikova-Yaroslavtsev 2013

✓ Submodular functions with discrete range {0, 1,...,k}

Value query access, disagreement-error.

Learning Submodular Functions

Raskhodnikova-Yaroslavtsev 2013

✓Submodular functions with discrete range {0,1,....,k}

Value query access, disagreement-error.

✓Time: poly$(n)k^{k \log k/\epsilon}$

Learning Submodular Functions

Our Algorithmic Results

✓PAC learning in time $\tilde{O}(n^2) \cdot 2^{O(1/\epsilon^4)}$

✓Agnostic learning with queries in time $\text{poly}(n, 2^{1/\epsilon^2})$

✓Agnostic learning with queries, discrete range {0,1,....,k} in time $\text{poly}(n, 2^k, 1/\epsilon)$.

Learning Submodular Functions

Our Lower Bounds

✓PAC learning *monotone* Submodular functions requires $2^{\Omega(\epsilon^{-2/3})}$ value queries.

✓Agnostically learning *monotone* submodular functions in time $n^{o(1/\epsilon^{2/3})}$ implies time $n^{o(k)}$ algorithm to learn k-parities with noise.

Learning Submodular Functions

Our Results

Representation and Approximation

✓ Approximating submodular functions by shallow real-valued decision trees.

✓ Corollary 1: Submodular functions are approximated by juntas!

✓ Corollary 2: Simple proof for approximating submodular functions by low-degree polynomials.

Learning Submodular Functions

Our Results

Representation and A[illegible]n

✓ Approximating submo[illegible]
shallow real-valued d[illegible]

✓ Corollary 1: Submodul[illegible]
approximated by juntas!

number of variables depend only on the accuracy!

✓ Corollary 2: Simple proof for approximating submodular functions by low-degree polynomials.

Overview of Structural Results

1. *Representing* submodular functions by low-rank decision trees of *lipschitz* submodular functions.

2. *Approximating* submodular functions by low-rank real-valued decision trees.

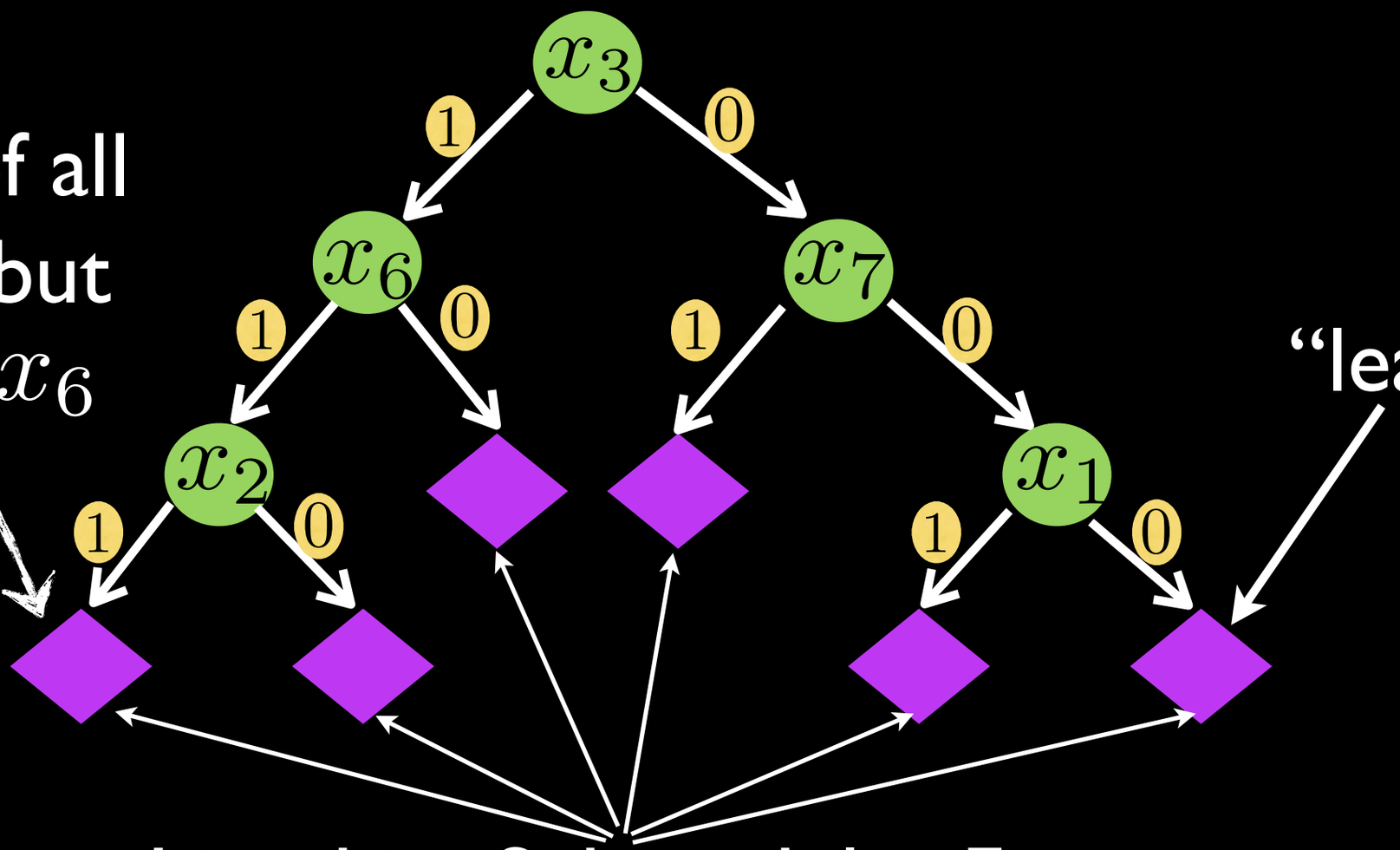
3. *Approximating* low-rank binary decision trees by low-depth decision trees.

I

Representation of Submodular Functions by Low-Rank Decision Trees of Lipschitz Submodular Functions

Representing Submodular Functions by Decision Trees

function of all variables but x_2, x_3, x_6



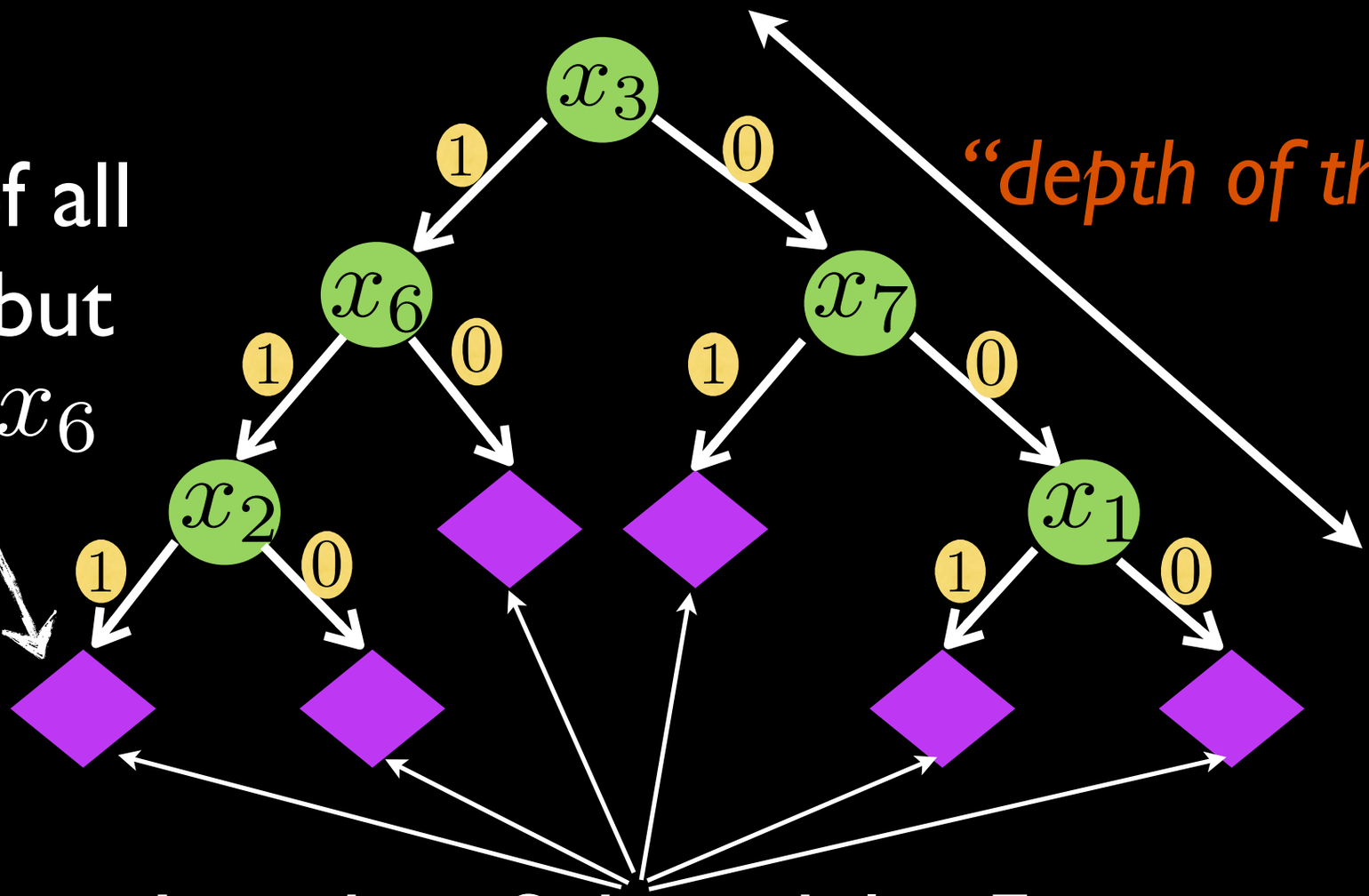
"leaf"

α-Lipschitz Submodular Functions

$$\forall S, i \ |f(S \cup \{i\}) - f(S)| \leq \alpha$$

Representing Submodular Functions by Decision Trees

function of all variables but x_2, x_3, x_6

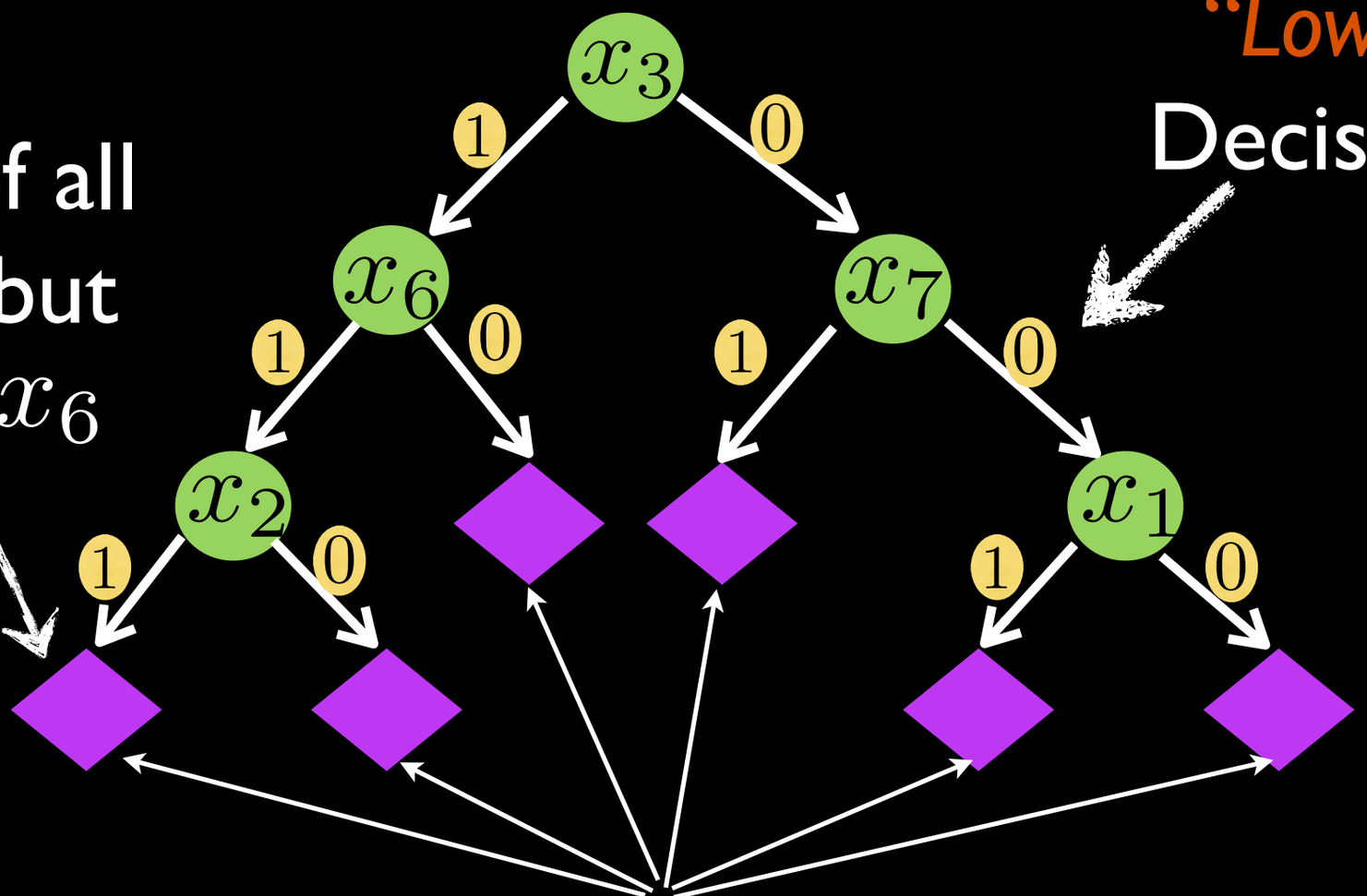


α-Lipschitz Submodular Functions

Representing Submodular Functions by Decision Trees

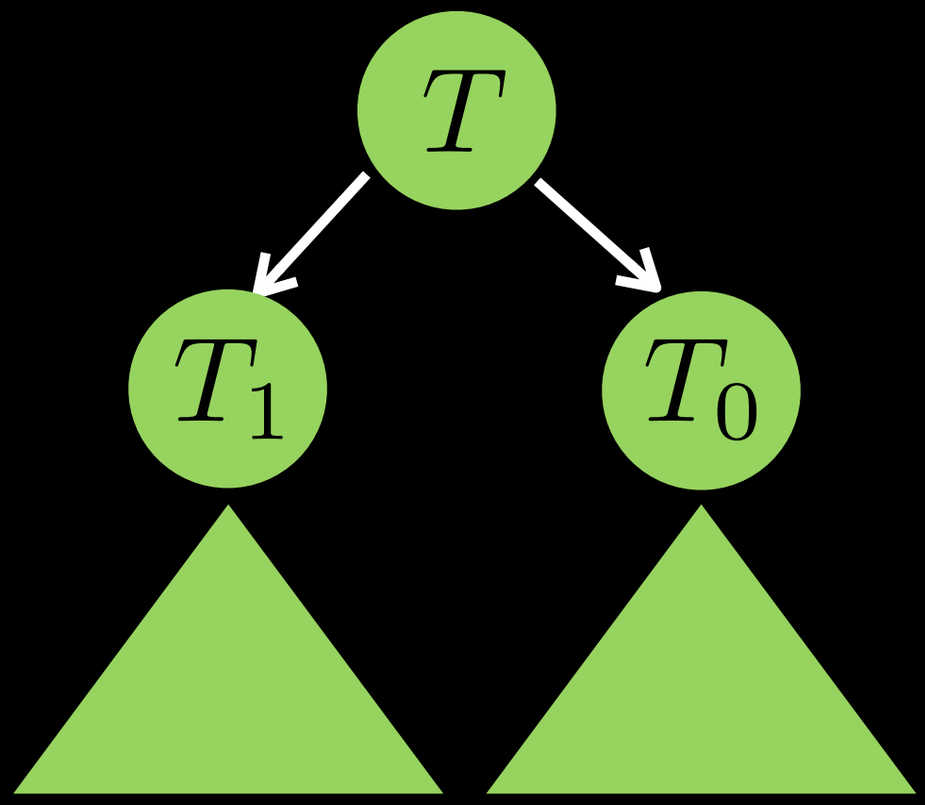
"Low-Rank" Decision Tree

function of all variables but x_2, x_3, x_6



α-Lipschitz Submodular Functions

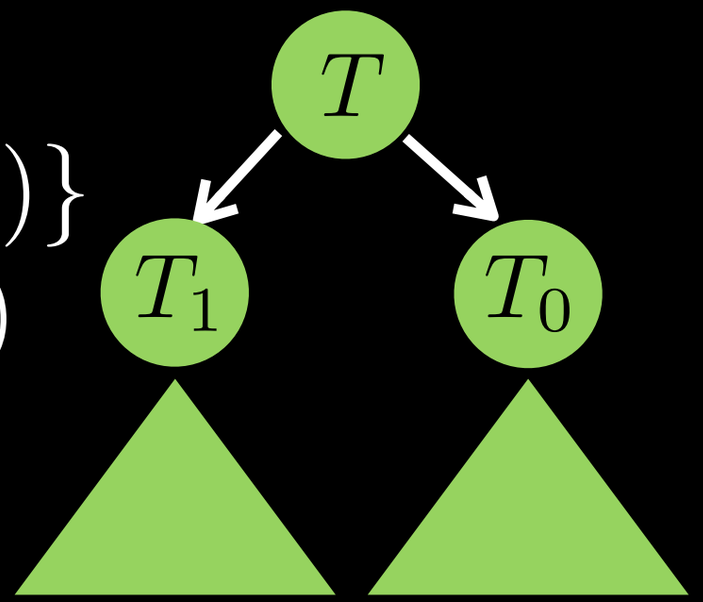
Rank of a Decision Tree



Ehrenfeucht-Haussler-89

Rank of a Decision Tree

$$rank(T) = \begin{cases} 0 \text{ if } T \text{ is a leaf} \\ \\ \max\{rank(T_0), rank(T_1)\} \\ \text{if } rank(T_0) \neq rank(T_1) \\ \\ rank(T_0) + 1 \text{ o/w} \end{cases}$$

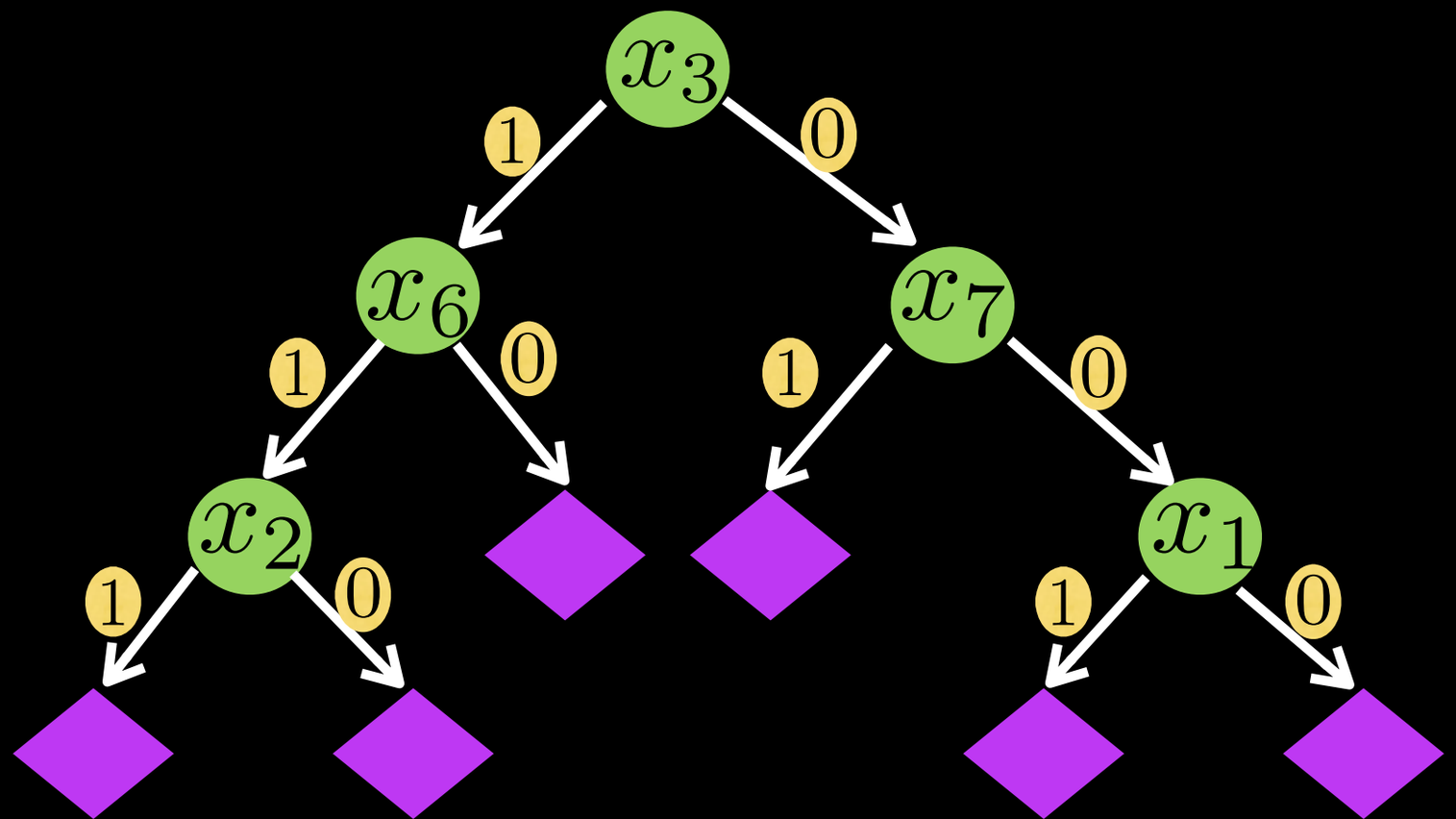


"*depth* of the largest complete binary tree that can be embedded in T"

Our Results: I

Representing Submodular Functions by Decision Trees

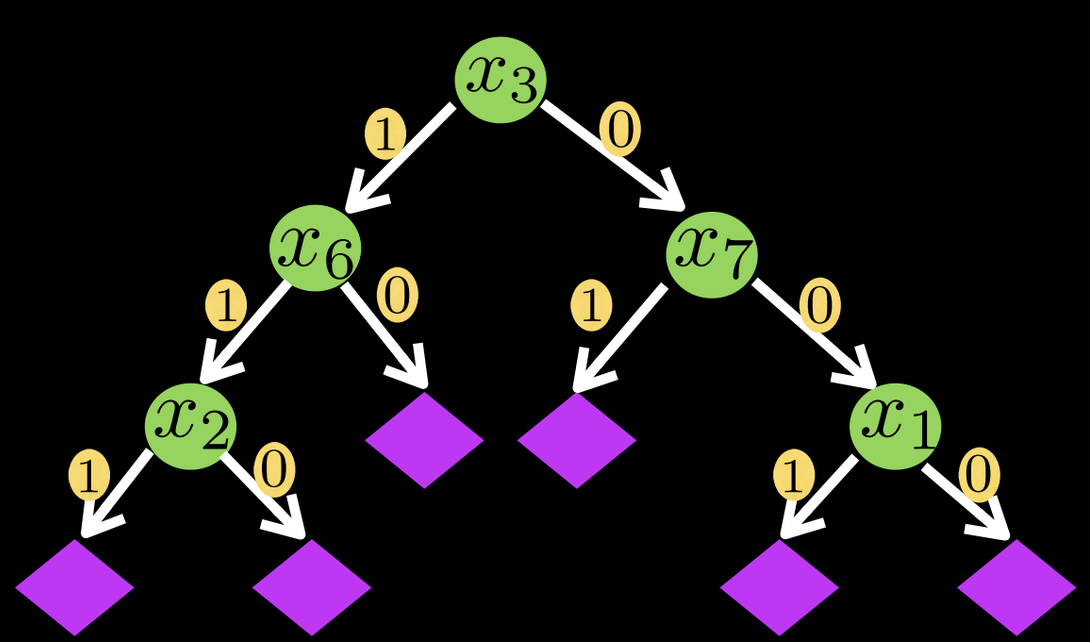
Submodular functions can be computed by decision trees of rank $2/\alpha$ *with* α*-Lipschitz submodular functions at each leaf.*



Our Results: I

Representing Submodular Functions by Decision Trees

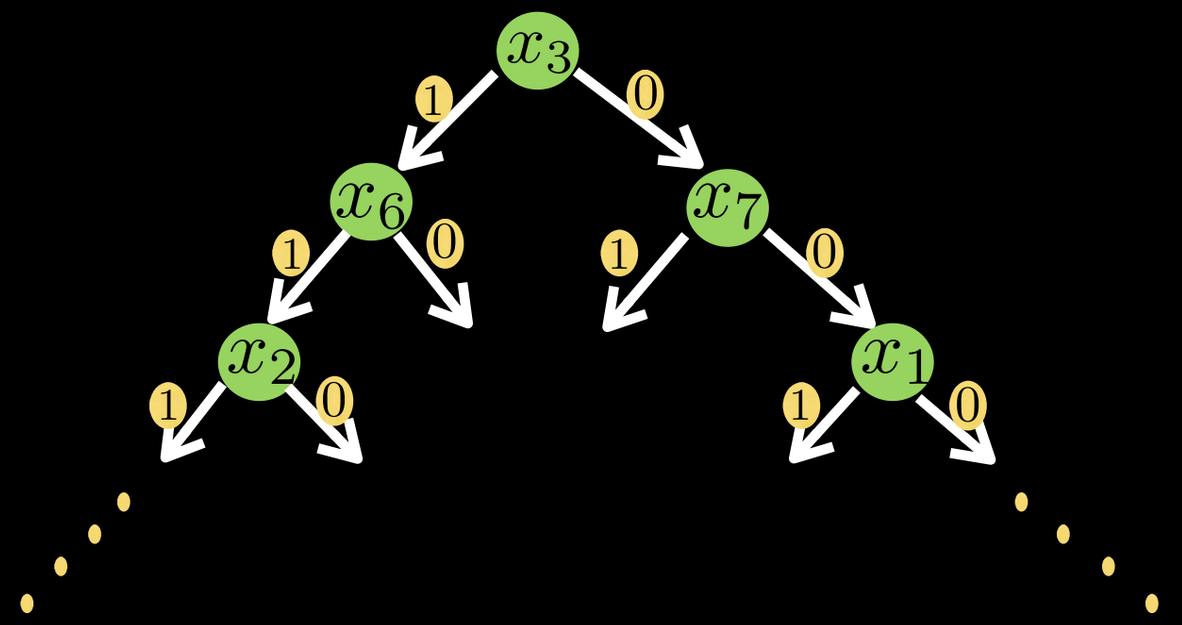
Submodular functions can be computed by decision trees of rank $2/\alpha$ with α-Lipschitz submodular functions at each leaf.



- Based on decomposition of submodular functions into Lipschitz submodular functions by GHRU-11.

Representing Submodular Functions by DTs

f is α-Lipschitz: leaf.*



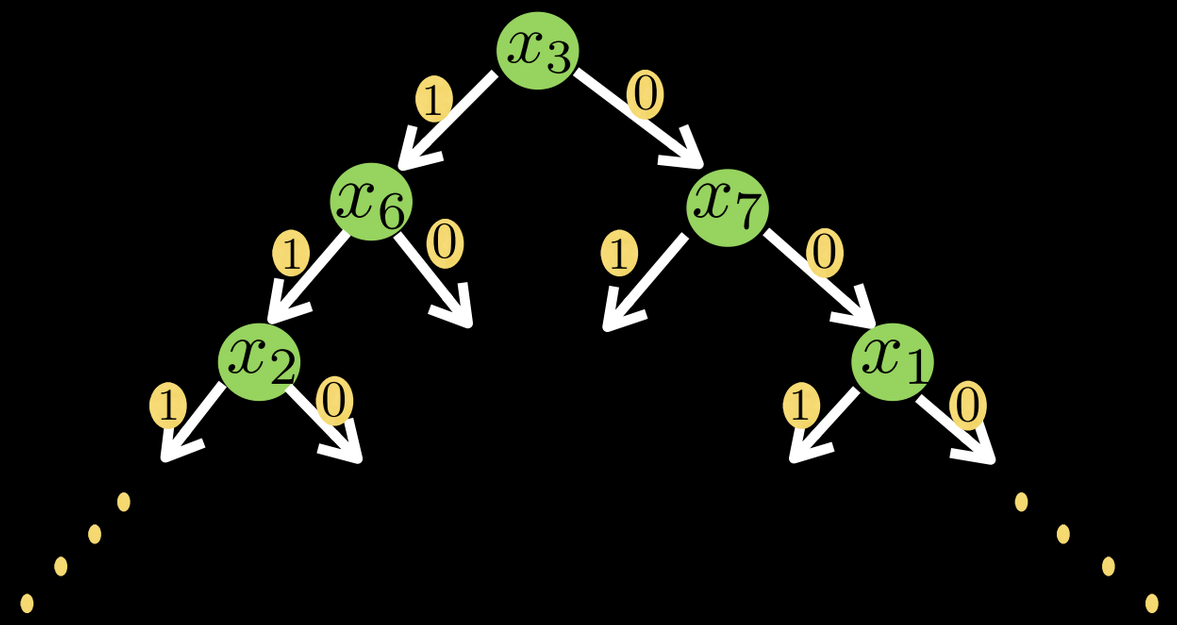
*f is monotone submodular.

Representing Submodular Functions by DTs

f is α-Lipschitz: leaf.

If not, by submodularity, there is a variable x_3:

$$f(\{3\}) - f(\emptyset) > \alpha$$



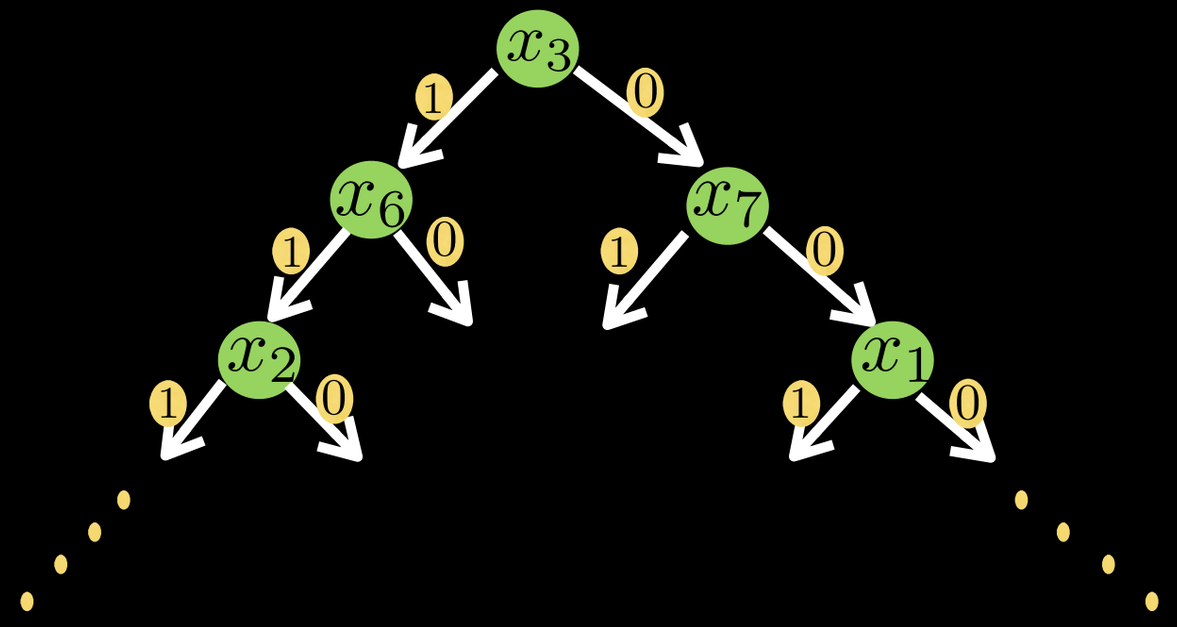
Representing Submodular Functions by DTs

f is α-Lipschitz: leaf.

If not, by submodularity, there is a variable x_3:

$$f(\{3\}) - f(\emptyset) > \alpha$$

Make x_3 a node.



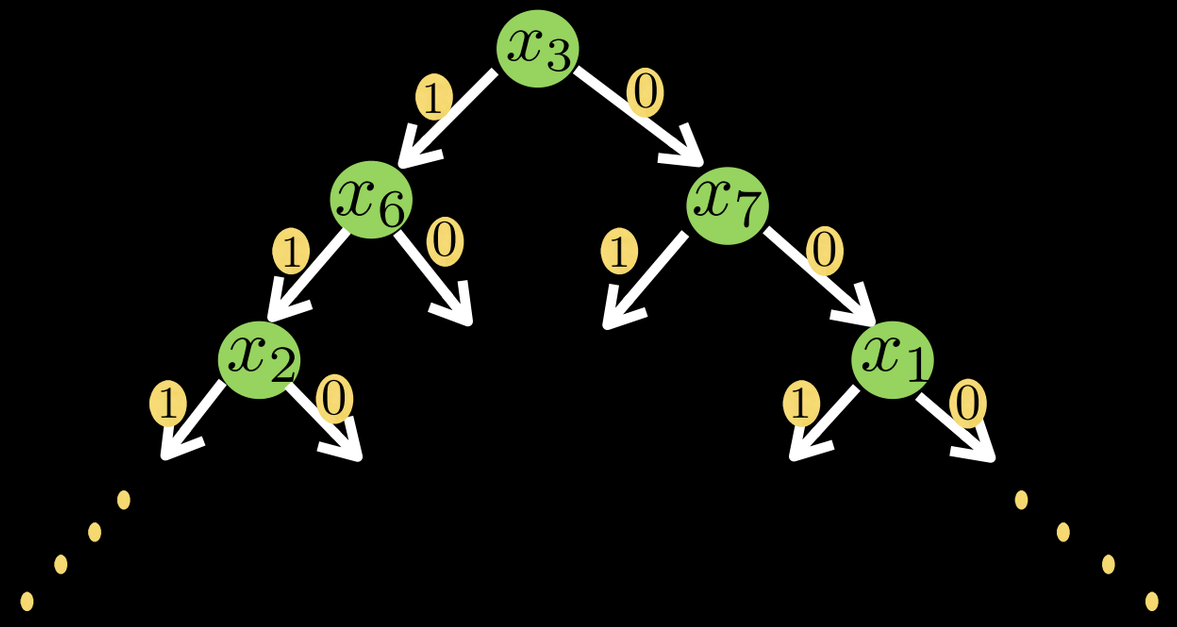
Representing Submodular Functions by DTs

f is α-Lipschitz: leaf.

If not, by submodularity, there is a variable x_3:

$$f(\{3\}) - f(\emptyset) > \alpha$$

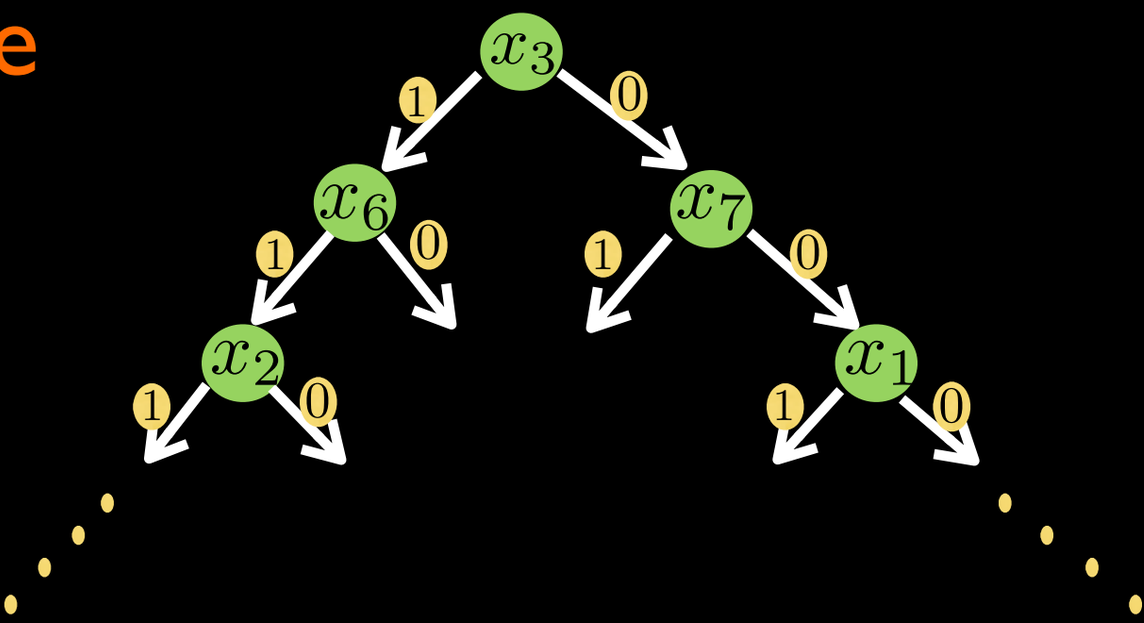
Make x_3 a node.



Repeat the procedure on the children nodes.

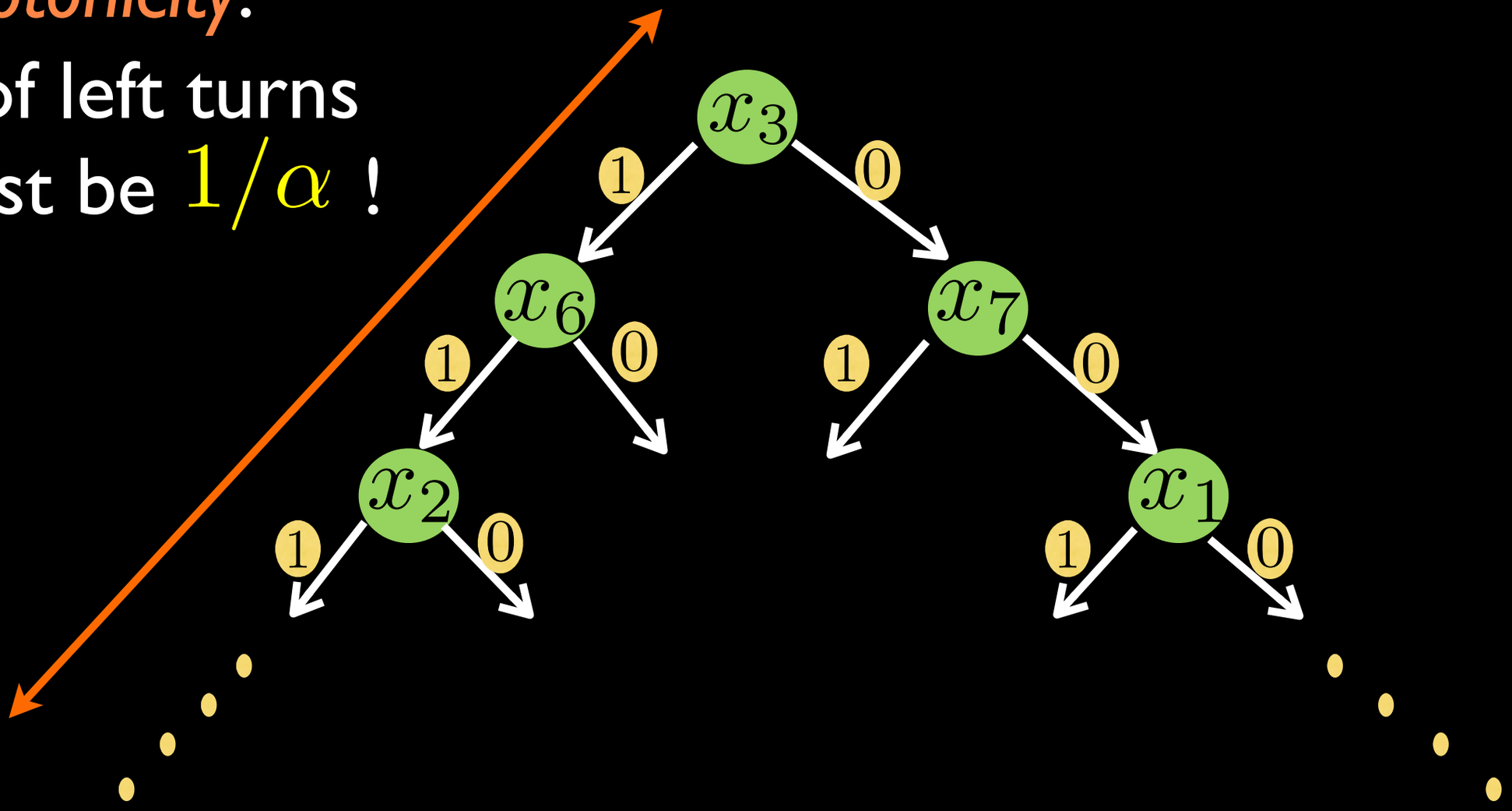
Representing Submodular Functions by DTs

Why is the rank of the tree low?



Representing Submodular Functions by DTs

by *monotonicity*:
number of left turns
can at most be $1/\alpha$!



Representing Submodular Functions by DTs

Non monotone f ?

- $f(S)$ is submodular → $f(\bar{S})$ is submodular.*

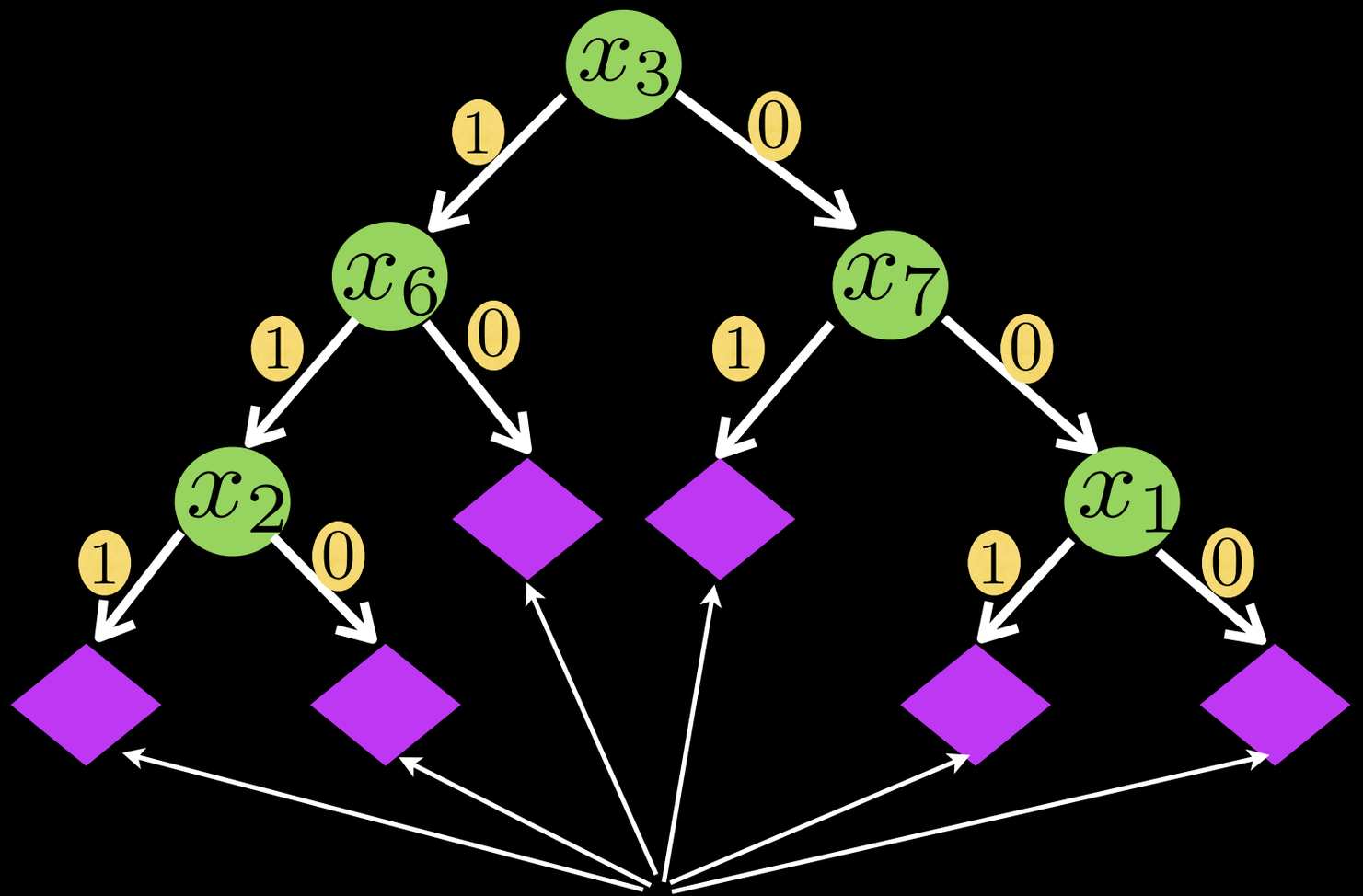
Obtain Lipschitz submodular leaves for non-monotone f.

*used in GHRU-11 also.

II

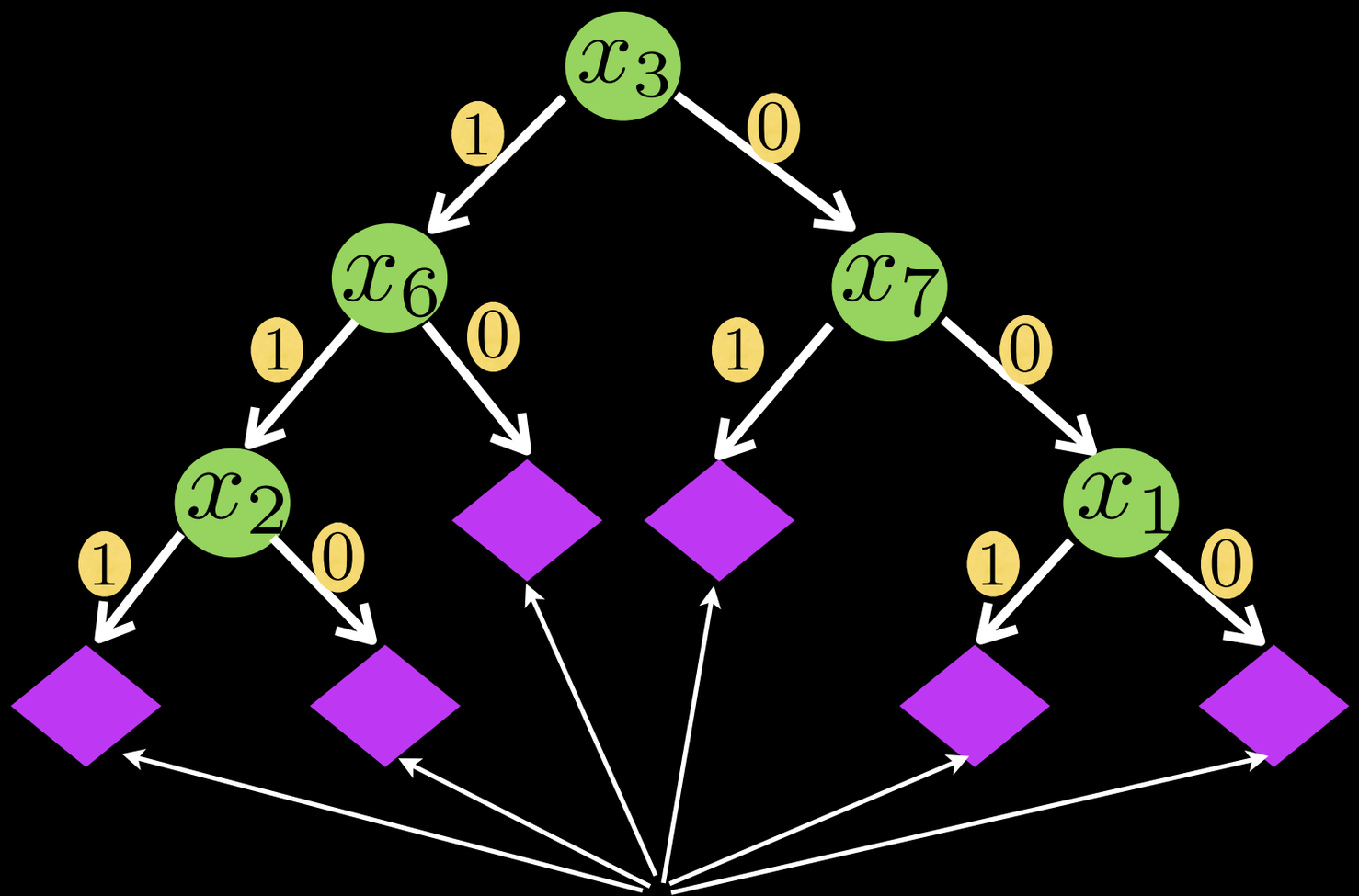
Approximating Submodular Functions by *Low-Rank* Decision Trees

Representing Submodular Functions by Decision Trees



α-Lipschitz Submodular Functions

Approximating Submodular Functions



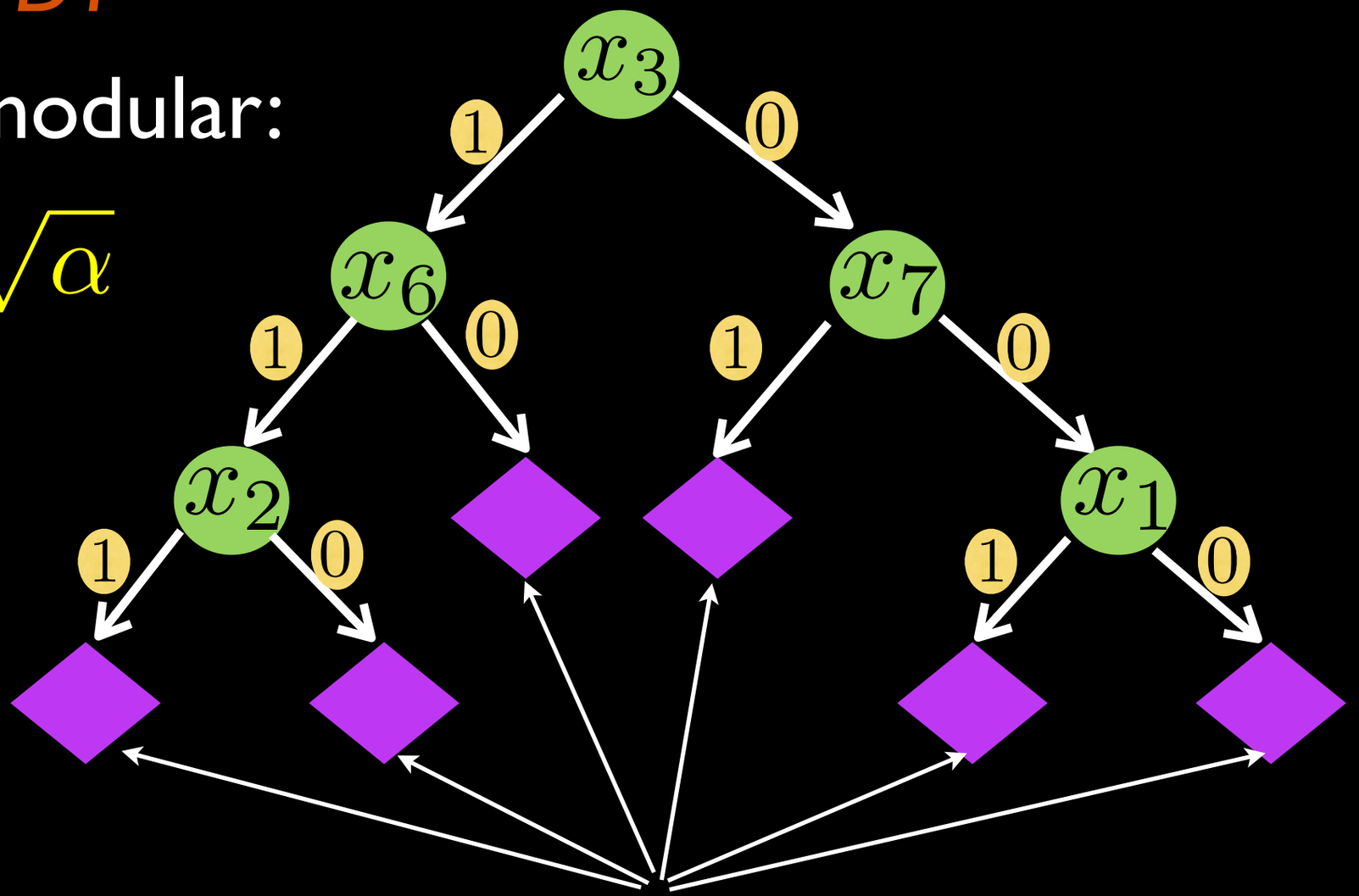
~~α-Lipschitz Submodular Functions~~

Approximating Submodular Functions

Real-Valued DT

If **f** is α-lipschitz submodular:

$$\mathbb{E}[|f - \mathbb{E}[f]|] \leq \sqrt{\alpha}$$



Concentration of lipschitz-submodular functions.

Boucheron-Massart-Lugosi-09, Vondrák-10, Balcan-Harvey-11

Our Results: II
Approximating Submodular Functions

Submodular Functions can be approximated within error ϵ by real valued decision trees of rank $4/\epsilon^2$*

*range normalized to [0,1]

III

Approximating low-rank binary decision trees by low-depth decision trees.

Our Results: III

Approximating Decision Trees

T : *Binary* Decision Tree

$T_{\leq d}$: Truncation to depth d

$$d = O(r + \log 1/\epsilon)$$

Then,

$$\Pr[T(x) \neq T_{\leq d}(x)] \leq \epsilon$$

General Pruning Procedure for any low-rank decision tree.

Our Results: III

Approximating Decision Trees

T : *Binary* Decision Tree

$T_{\leq d}$: Truncation to depth d

$$d = O(r + \log 1/\epsilon)$$

rank

Then,

$$\Pr[T(x) \neq T_{\leq d}(x)] \leq \epsilon$$

General truncation for any low-rank decision tree.

Our Results: III

Approximating Decision Trees

T : *Binary* Decision Tree

$T_{\leq d}$: Truncation to depth d

$$d = O(r + \log 1/\epsilon)$$

Then,

$$\Pr[T(x) \neq T_{\leq d}(x)] \leq \epsilon$$

generalizes truncation based on size from Kushilevitz-Mansour-93

Our Results: Corollary Approximating Submodular Functions

For every submodular f*, there is a real valued decision tree* T *of depth* $O(1/\epsilon^2)$ *such that:*

$$\mathbb{E}[|(f(x) - h(x))|] \leq \epsilon$$

Our Results: Corollary
Approximating Submodular Functions

For every submodular f, there is a real valued decision tree T of depth $O(1/\epsilon^2)$ such that:

$$\mathbb{E}[|(f(x) - h(x))|] \leq \epsilon$$

Thus, T depends on at most $2^{O(1/\epsilon^2)}$ variables.

Simple proof showing deg. $O(1/\epsilon^2)$ approximating polynomials (Cheraghchi et. al. 2012)

Our Results: Corollary
Approximating Submodular Functions

For every submodular f, there is a real valued decision tree T of depth $O(1/\epsilon^2)$ [illegible] that:

$$\mathbb{E}[|(f(x) - h(x))|] \leq \epsilon$$

In addition: "Junta" Approximation

Thus, T depends on at most $2^{O(1/\epsilon^2)}$ variables.

Simple proof showing deg. $O(1/\epsilon^2)$ approximating polynomials (Cheraghchi et. al. 2012)

Learning

Applications: I

PAC Learning on Product Distributions

There exists an algorithm, which, given random examples labeled by a submodular function ***f****, returns a hypothesis* ***h****, that depends on* $2^{O(1/\epsilon^2)}$ *variables and satisfies:*

$$\mathbb{E}[|(f(x) - h(x))|] \leq \epsilon$$

Time: $\tilde{O}(n)^2 \cdot 2^{O(1/\epsilon^4)}$

Examples: $\log(n) \cdot 2^{O(1/\epsilon^4)}$

Applications: I

PAC Learning on Product Distributions

Shallow Decision Tree Approximation

$\Rightarrow$

Low-degree polynomial approximation of a small number of variables

$+$

Efficient procedure to find "influential" variables of a submodular function using Fourier coefficients of degree at most 2.

$+$ ℓ_1-regression.

Applications: II
Agnostic Learning with Queries

There is an algorithm that agnostically learns submodular functions in poly$(n, 2^{1/\epsilon^2})$ *time and* poly$(\log(n), 2^{1/\epsilon^2})$ *value queries.*

attribute efficient Kushilevitz-Mansour (*Feldman-07*)

Agnostic Boosting (*Kalai-Kanade-09*,*Feldman-09*)

alternate: Gopalan-Kalai-Klivans-08, no attribute efficiency

Our Results: Lower Bounds

Information theoretic:

PAC learning *monotone* Submodular functions requires $2^{\Omega(\epsilon^{-2/3})}$ value queries.

Embed *any* boolean function in a monotone submodular function in a higher dimension.

Our Results: Lower Bounds

Information theoretic:

PAC learning *monotone* Submodular functions requires $2^{\Omega(\epsilon^{-2/3})}$ value queries.

$\Rightarrow$ PAC and agnostic with queries algorithms optimal up to the exponent of $1/\epsilon$

Our Results: Lower Bounds

Information theoretic:

PAC learning *monotone* submodular functions requires $2^{\Omega(\epsilon^{-2/3})}$ value queries.

Computational:

Agnostically learning *monotone* submodular functions in time $n^{o(1/\epsilon^{2/3})}$ implies time $n^{o(k)}$ algorithm to learn k-parities with noise.

Show a monotone submodular function that has large correlation with a parity.

Our Results: Lower Bounds

Information Theoretic:

PAC learning *monoto* ons
requires $2^{\Omega(\epsilon^{-}}$

Best: $\sim n^{0.8k}$
Valiant-12

Computational:

Agnostically learning *monotone* submodular functions in time $n^{o(1/\epsilon^{2/3})}$ implies time $n^{o(k)}$ algorithm to learn k-parities with noise.

Our Results: Lower Bounds

Information Theoretic:

PAC learning *monotone* Submodular functions requires $2^{\Omega(\epsilon^{-2/3})}$ value queries.

Computational:

Agnostically learning *monotone* submodular functions in time $n^{o(1/\epsilon^{2/3})}$ implies time $n^{o(k)}$ algorithm to learn k-parities with noise.

$\Rightarrow$ Agnostic algorithm of CKKL12 optimal up to the exponent of $1/\epsilon$.

Summary

Approximating submodular functions by low-depth decision trees

Almost optimal PAC and agnostic Learning Algorithms

Follow-Up Work

Feldman-Vondrák-13

Better junta approximation: $poly(1/\epsilon)$ variables!

PMAC learning submodular functions
on uniform distribution.

Feldman-K-13

Fully polynomial time algorithm for
PAC\PMAC learning *coverage* functions.

Open Questions

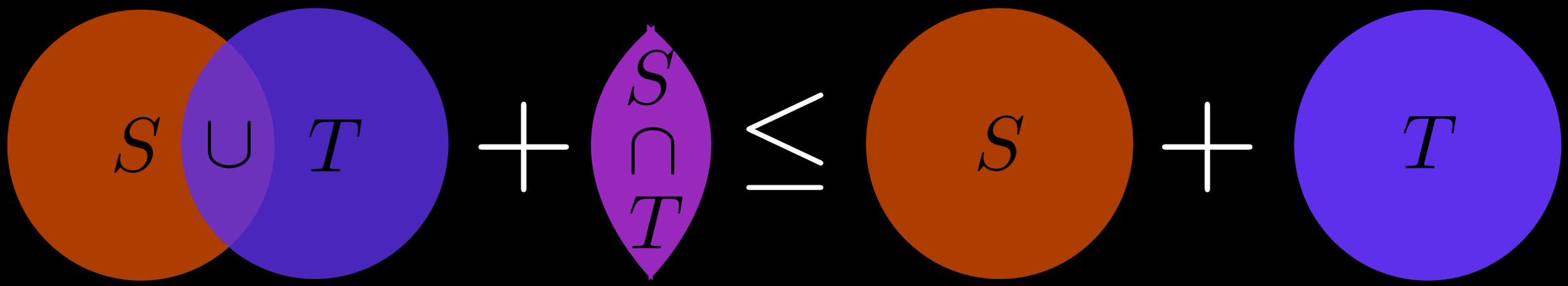
Learning\Approximation on more general distributions?


Questions?

Submodular Functions

A function $f : 2^{[n]} \to \mathbb{R}$ is *submodular* if:

$$f(S \cup T) + f(S \cap T) \leq f(S) + f(T)$$



Submodular Functions

$$\{0,1\}^n$$

A function $f : 2^{[n]} \to \mathbb{R}$ is *submodular* if:

$$f(S \cup T) + f(S \cap T) \leq f(S) + f(T)$$