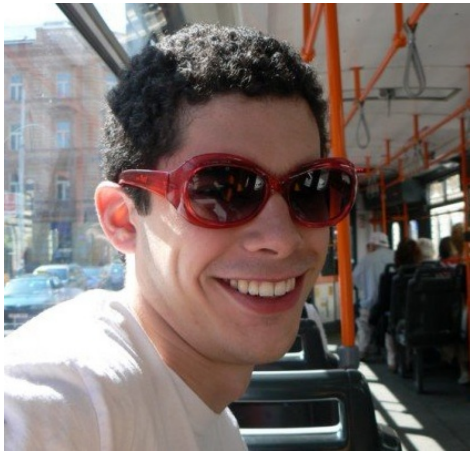
Attribute-Efficient Learning and Polynomial Threshold Functions

Li-Yang Tan
Columbia University

Joint work with
Rocco A. Servedio (Columbia) and Justin Thaler (Harvard)



Tsinghua University Theory Seminar, 20 April 2011

- Machine learning: the study of algorithms that make accurate predictions from raw data
- A major algorithmic challenge in machine learning:

Learning in the presence of irrelevant information

High dimensional data (n dimensions) that only depend on k << n unknown dimensions

- Same problem, different names: feature selection, sparsity, the junta problem, *etc.*
- Significant practical importance, especially in the age of big data
- This talk: clean theoretical formulation of the problem (A. Blum 1991)

the learning framework

- Goal: Learn unknown target function $f : \{0,1\}^n \to \{0,1\}$, where f only depends on $k << n$ unknown coordinates (*e.g.* $k = \log(n)$ or constant).

$$f(x_1, x_2, \ldots, x_n) = g(x_2, x_7, x_9, x_{11}, x_{34})$$

- f belongs to some known concept class C (*e.g.* conjunctions, decision lists, decision trees, *etc.*)
- What does it mean to learn f?
  - Learner is given information about how f labels the data $\{0,1\}^n$
  - Computes hypothesis $h : \{0,1\}^n \to \{0,1\}$
  - Performance determined by how well h predicts f
- This talk: Online mistake bound model (Littlestone 1988). Clean and simple theoretical model!

the learning model

- Goal: Learn unknown function f : $\{0,1\}^n$ -> {0,1}, where f only depends on k << n unknown coordinates, and f belongs to a known concept class C.

Learning consists of a sequence of trials. In each trial:

- Learner is given some x from $\{0,1\}^n$
- Learner outputs h(x), her guess as to what f(x) is
  - If h(x) = f(x), great!
  - If h(x) ≠ f(x), learner is charged a mistake
- If learner makes a mistake, she updates h

Goal: efficient algorithm that minimizes number of mistakes over all possible sequences of trials

Ideally, runs in time poly(n) per trial, and total number of mistakes at most poly(k, log(n)).

Goal: efficient algorithm that minimizes number of mistakes over all possible sequences of trials

A malicious adversary who, for each trial

- chooses x in $\{0,1\}^n$
- says “correct” or “incorrect” as he wishes

in order to make learner incur as many mistakes as possible. His only constraint: at any time, there must be at least one concept in C consistent with his responses so far!

two easy mistake-bound algorithms

Totally trivial algorithm for any concept class C:

- Pick arbitrary c in C as initial hypothesis
- Whenever mistake incurred, switch to different c
- Constant run time per trial, but mistake bound |C|

Not-so-trivial, but still easy algorithm for any concept class C:

- Take majority vote of concepts in C as initial hypothesis
- Whenever mistake incurred, eliminate all inconsistent c
- "Halving algorithm" (why?)
- Mistake bound $\log_2(|C|)$, but run time |C|

attribute-efficient learning

Ideally, algorithm runs in time poly(n) per trial, and total number of mistakes at most poly(k, log(n))

- If this is possible, we say that the concept class C is "attribute-efficiently learnable"
- Often difficult even for simple C! Very few classes known to be attribute-efficiently learnable

This talk: tradeoffs between run time and mistake bound, both upper and lower bounds

Side note: standard results in learning theory translate efficient algorithms in the online mistake bound model into efficient algorithms in Valiant's Probably Approximately Correct (PAC) model

outline for rest of talk

- Decision lists (what we want to learn) and linear threshold functions (how we will learn them)
- Expanded-Winnow algorithm
  - Learning becomes concrete complexity
  - Low-degree low-weight polynomial threshold functions yield efficient learning algorithms
- PTF degree-weight tradeoffs for decision lists
- Lower bounds, and a new Markov-type inequality

one slide summary of this talk

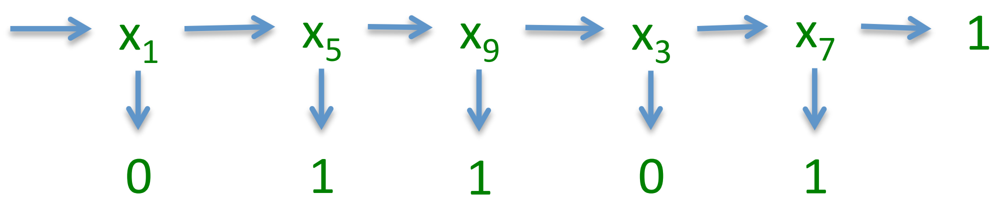
- We make progress on the well-studied problem of attribute-efficiently learning decision lists
- Our upper bounds yield algorithms with the best known running time and mistake bound

Theorem (Servedio-T-Thaler): Let f be a length-k DL. For every $d \leq k$, we have an algorithm that learns f in time n^d with mistake bound $2^{(k/d)^{1/2}}$

- Our lower bounds suggest that significantly different techniques will be required to make further progress
- Both upper and lower bounds utilize tools from approximation theory
- We prove a sharpened version of a classical inequality that could be of independent interest

attribute-efficiently learning decision lists (DL)

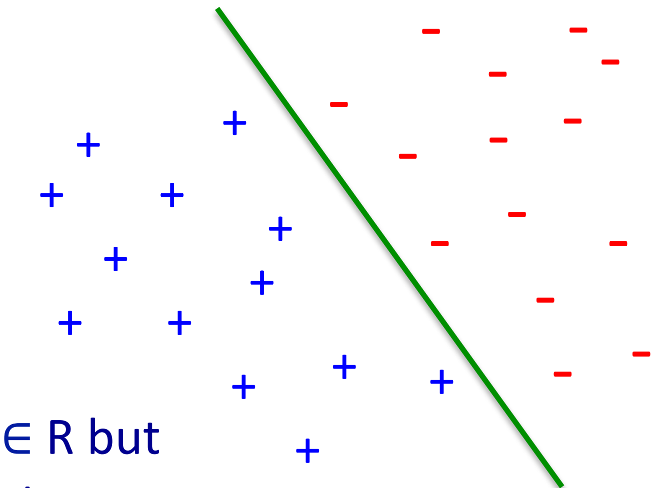
A length k decision list over $x_1, \ldots, x_n$



- A sequence of nested “if-then-else” statements
- Conjunctions and disjunctions can be expressed as DLs
- Attribute-efficiently learning DLs is a well-studied and challenging open problem!
- First posed by [Blum 1992], subsequently considered by many authors [Blum-Hellerstein-Littlestone 1990, Blum-Langley 1997, Valiant 1999, Nevo-El-Yaniv 2002, Klivans-Servedio 2006, Long-Servedio 2006]

linear threshold functions (LTFs)

$$f(x) = \mathrm{sgn}(w_1x_1 + w_2x_2 + \ldots + w_nx_n + \theta)$$
$$w_1, w_2, \ldots, w_n, \theta \in Z$$



- Usually defined with $w_1, w_2, \ldots, w_n, \theta \in R$ but for this talk we require that they are integers
- Different names, same object: halfspaces, weighted majorities, perceptrons, linear separators, threshold gates, *etc.*
- Complexity theory: TC_0 versus NP
- Social choice theory: voting schemes
- Learning theory: Perceptron, Winnow, SVMs

learning LTFs

Theorem (Littlestone 1988): Let
$f(x) = \text{sgn}(w_1x_1 + w_2x_2 + . . . + w_nx_n + \theta)$, $w_i, \theta \in Z$.
If $\sum|w_i| \leq W$ for all i, Winnow learns f with
run time O(n) per trial and mistake bound $O(W^2 \log(n))$

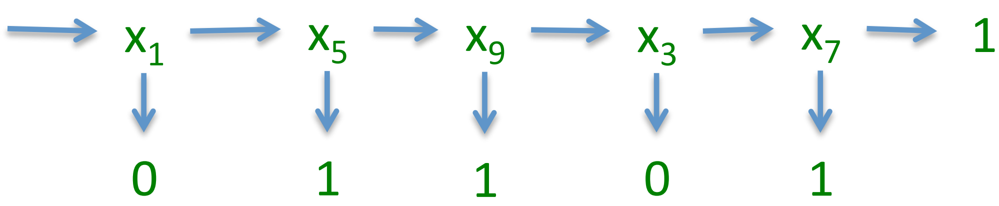
- If $\sum|w_i| \leq W$, we say that f is a "weight-W" LTF
- Learning f reduces to showing that f can be computed by a low-weight LTF
- Note that not all functions can be expressed as an LTF, even if arbitrarily high weights are allowed!
  - simple example: $f(x_1, x_2) = x_1 + x_2 \bmod 2$

Our main learning tool in this talk:
a higher-degree generalization of Winnow

But first, what does Winnow tell us about attribute-efficiently learning decision lists?

LTF weight of decision lists

A length k decision list over $x_1, \dots, x_n$



- First variable (x_1) more important than the second (x_5), second variable more important than the third (x_9), *etc.*
- Set weight of first variable to be larger than *sum* of weights of all other variables
- Easy induction: every DL has a 2^k weight LTF
- Not hard to show: there exists a DL that requires weight 2^k

two easy algorithms for decision lists

Theorem (Littlestone 1988): Let
$f(x) = \mathrm{sgn}(w_1x_1 + w_2x_2 + . . . + w_nx_n + \theta)$, $w_i, \theta \in Z$.
If $\sum|w_i| \leq W$ for all i, possible to learn f with
run time $O(n)$ per trial and mistake bound $O(W^2 \log(n))$

- Every length k DL is a weight 2^k LTF
- Mistake bound $2^k \log(n)$, run time $O(n)$

Halving algorithm

- Take majority vote of concepts in C as initial hypothesis
- Whenever mistake incurred, eliminate all inconsistent c
- Mistake bound $\log_2(|C|)$, but run time $|C|$

- There are $n^{O(k)}$ length-k DLs over n variables
- Mistake bound $O(k \log(n))$, run time $n^{O(k)}$

This talk: best known trade-offs
between run time and mistake bound

Expanded Winnow

A natural generalization of LTFs $f(x) = \mathrm{sgn}(w_1x_1 + w_2x_2 + . . . + w_nx_n + \theta)$:

$f(x) = \mathrm{sgn}(p(x_1, \ldots , x_n))$, where p is a degree-d polynomial

We say that f is a degree-d polynomial threshold function (PTF)

Theorem (Klivans-Servedio 2006):
Let $\mathrm{sgn}(p(x_1, \ldots , x_n))$, where p is a degree-d polynomial with integer coefficients whose magnitude sum to W. Then we can learn f with run time $n^{O(d)}$ per trial and mistake bound $O(W^2\, d \log(n))$

Proof. Every degree-d PTF is an LTF over $n^{O(d)}$ variables!
Make every monomial a new variable ("feature expansion")

PTFs for decision lists

Theorem (Klivans-Servedio 2006):
Let $\text{sgn}(p(x_1, \dots, x_n))$, where p is a degree-d polynomial with integer coefficients whose magnitude sum to W. Then we can learn f with run time $n^{O(d)}$ per trial and mistake bound $O(W^2 d \log(n))$

- Attribute-efficient learning of DLs reduces to showing that every DL has a low-degree, low-weight PTF
- That is, every DL computed by the sign of a low-degree polynomial with small integer coefficients
- Tradeoffs between degree and weight a natural question on its own!
  - For a fixed degree, how small can the weights be?
  - Are there DLs that require high degree and weight?

Klivans-Servedio 2006

Theorem (Klivans-Servedio 2006): Let f be a length-k DL. For every $d \leq k$, there is degree d, weight $2^{(k/d^2)+d}$ PTF computing f

	run time	mistake bound
Winnow	n	$2^k \log(n)$
Halving	n^k	$k \log(n)$
Klivans-Servedio (for every $d \leq k$)	n^d	$2^{(k/d^2)+d} \log(n)$

Theorem (Beigel 1996): There is a DL such that for every $d \leq k$, any degree d PTF computing f requires weight $2^{k/d^2}$

our contribution

Theorem (Klivans-Servedio 2006): Let f be a length-k DL. For every $d \le k$, there is degree d, weight $2^{(k/d^2)+d}$ PTF computing f

Theorem (Beigel 1996): There is a DL such that for every $d \le k$, any degree d PTF computing f requires weight $\ge 2^{k/d^2}$

- The function k/d^2: decreasing for $d \le k^{1/3}$, but increasing after. Beigel's lower bound shows that the Klivans-Servedio result is optimal for all $d \le k^{1/3}$.
- Situation unclear for $d \ge k^{1/3}$. For example, for $d = k^{1/2}$:
  - Klivans-Servedio result gives upper bound of $2^{k^{1/2}}$, worse than the $2^{k^{1/3}}$ bound for $d = k^{1/3}$!
  - Beigel's lower bound vacuous!

This talk: we complete the picture for all $d \ge k^{1/3}$, giving matching upper and lower bounds

our contribution

Theorem (Servedio-T-Thaler): Let f be a length-k DL. For every $d \leq k$, there is degree d, weight $2^{(k/d)^{1/2}}$ PTF computing f

Theorem (Servedio-T-Thaler): There is a DL such that for every $d \leq k$, any degree d PTF computing f requires weight $\geq 2^{(k/d)^{1/2}}$

	run time	mistake bound
Winnow	n	$2^k \log(n)$
Halving	n^k	k log(n)
Klivans-Servedio (for every $d \leq k^{1/3}$)	n^d	$2^{(k/d^2)+d} \log(n)$
Servedio-T-Thaler (for every $d \geq k^{1/3}$)	n^d	$2^{(k/d)^{1/2}} \log(n)$

Our lower bounds, along with Beigel's, suggest that significantly different techniques will be required to make further progress on the problem

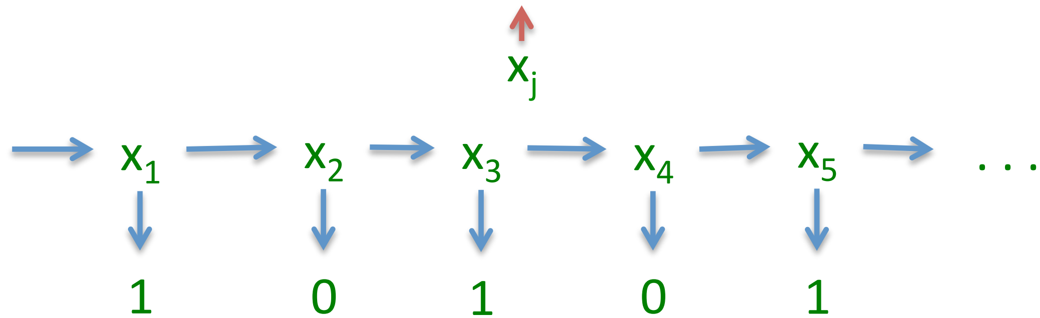
the lower bound

We prove that there exists a DL such that any low degree PTF for the DL requires high weight

the ODD-MAX-BIT function

Look at the right-most bit set to 1. If it is at an odd coordinate, output 1, else output 0

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



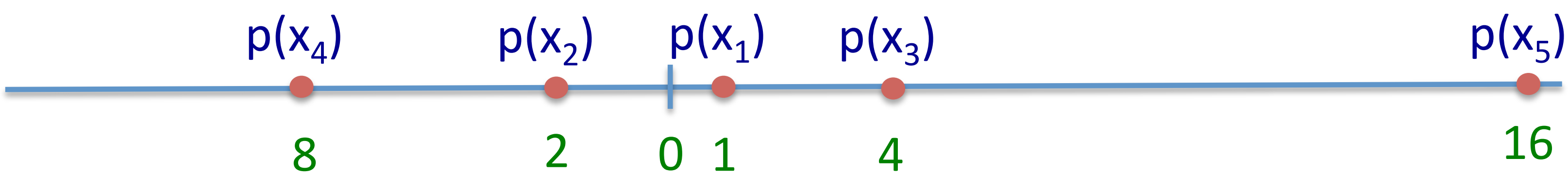
Theorem (Beigel 1996): Any degree d PTF for the OMB function must have weight $\geq 2^{k/d^2}$

Theorem (Servedio-T-Thaler): Any degree d PTF for the OMB function must have weight $\geq 2^{(k/d)^{1/2}}$

Recall: Beigel's bound is stronger for $d \leq k^{1/3}$, our bound is stronger for $d \geq k^{1/3}$. Both are tight.

main idea behind lower bound

Construct a sequence of inputs $x_1, x_2, \ldots, x_{k/d^2}$ such that $p(x_{i+1}) \geq 2\,|p(x_i)|$

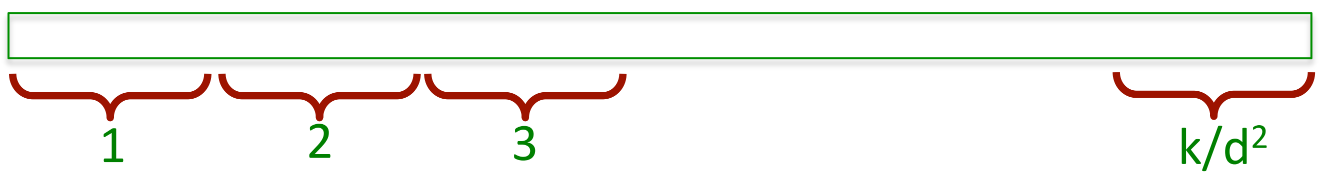


If we succeed in finding such a sequence, then $|p(x_{k/d^2})| \geq 2^{k/d^2}$.

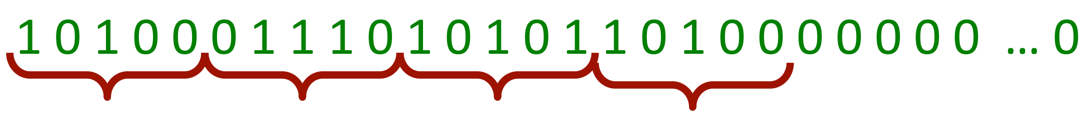
If p attains value $\geq 2^{k/d^2}$ then p must have weight $\geq 2^{k/d^2}$!

main idea behind lower bound

Construct a sequence of inputs $x_1, x_2, \ldots, x_{k/d\text{^}2}$ such that $p(x_{i+1}) \geq 2\,|p(x_i)|$



- Break k coordinates up into k/d^2 blocks of size d^2
- For all i, x_i will be an input such that all blocks from i+1 onwards are 0's



Prove the existence of this doubling sequence of inputs by induction. Base case: there exists an x_1 such that $p(x_1) \geq 1$ (trivial!)

inductive step

Suppose we have found x_i such that $p(x_i) \geq M$. We will prove existence of x_{i+1} such that $p(x_{i+1}) \leq -2M$. Proceed by contraction; suppose no such input exist.

- Define F(k) to be the average of p's values of all inputs y such that
  - y agrees with x_i in the first i blocks
  - y has k 1's in even coordinates the i+1 block
  - y has all 0's in the i+2 block onwards

1 0 1 0 0 0 1 1 1 0 0 1 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 … 0

1 0 1 0 0 0 1 1 1 0 0 1 0 1 0 0 1 0 1 0 0 0 0 0 0 0 0 0 … 0

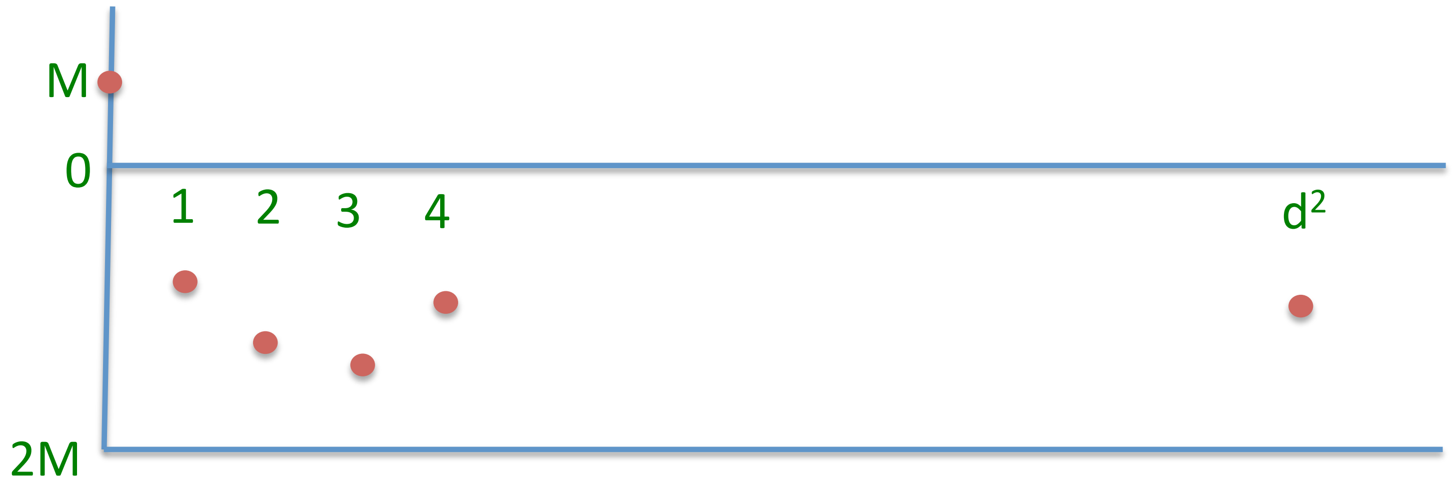
k 1's in even positions in the i+1 block

- What is F(0)? $F(0) = p(x_i) = M$
- What is F(1)? F(1) = average of p's values on inputs with rightmost bit in an even position. So $F(1) \in [-1,-2M]$
- Same for F(2), F(3), *etc.*

- What is F(0)? F(0) = $p(x_i)$ = M
- What is F(1)? F(1) = average of p's values on inputs with rightmost bit in an even position. So F(1) ∈ [-1,-2M]
- Same for F(2), F(3), *etc.*

- What is the degree of F? Since F is the average of p's values on some inputs, deg(F) ≤ deg(p) ≤ d.

Can F have degree ≤ d?



- Properties of F:
  - F is bounded between [-2M, M] on the interval $[0,d^2]$
  - $|F'(t)| > M$ for some t in [0,1]
- Shifting and scaling, transform F into H : [-1,1] -> [-1,1] such that $|H'(t)| > d^2$ for some t in [-1,1] and deg(H) = deg(F).

Theorem (Markov): Let H : [-1,1] -> [-1,1].
Then deg(H) ≥ max{|H'(t)| : t in [-1,1]}

- So F attains value < -2M. But F is simply the average of p's values on a few inputs, so there must exist an x_{i+1} such that $p(x_{i+1}) < -2M$.

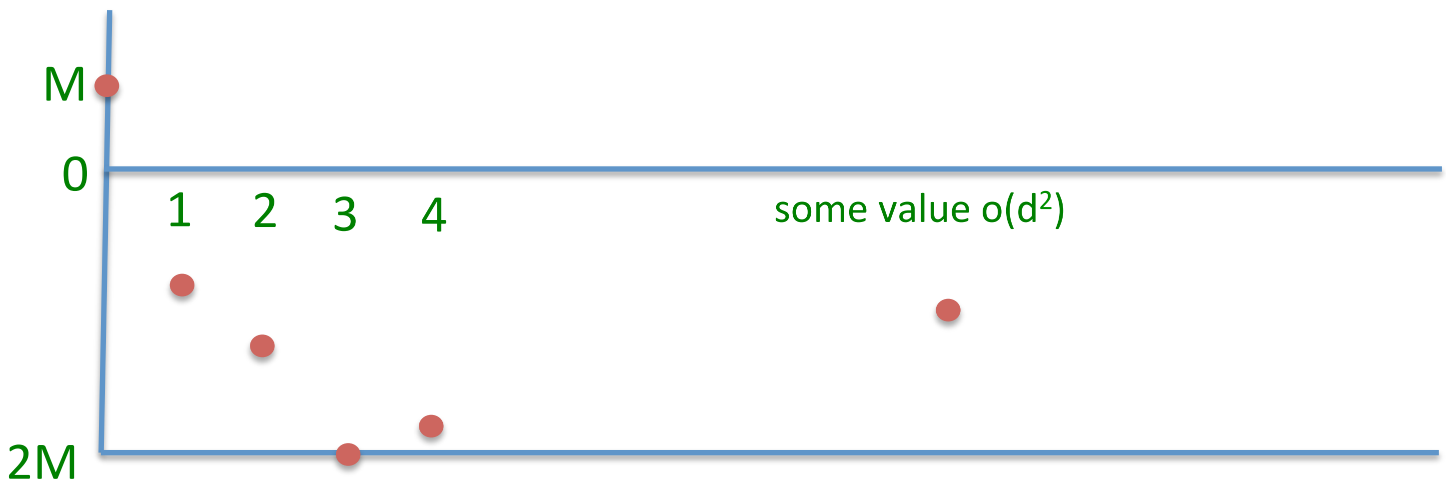
recap of Beigel's proof

- Break k coordinates up into k/d^2 blocks of size d^2
- Try to find sequence of "doubling inputs" $x_1, \ldots, x_{k/d^2}$ each twice the magnitude of the previous
- Suppose we have found x_i. If x_{i+1} does not exist, we use Markov's theorem to say deg(p) > d, a contradiction.
- This shows we can keep going, and so p must have weight $|p(x_{k/d^2})| \geq 2^{k/d^2}$

If only we could take blocks of smaller size (*i.e.* a longer sequence of double inputs), we would get a better bound! But Markov's theorem is tight.

Crux of our improvement: Take blocks of size $o(d^2)$. Suppose x_{i+1} does not exist. Instead of showing that p must have high degree, we show directly that p has high weight.

Beigel: “The polynomial must have degree > d, a contradiction!”



Us: “If the polynomial has degree > d, we get a contradiction. If the polynomial has degree ≤ d, it must have high weight!”

our refinement of Markov

Theorem (Markov, stated differently): Let F : [-1,1] -> [-1,1].
If deg(F) ≤ d, then max |F'(t)| ≤ d^2

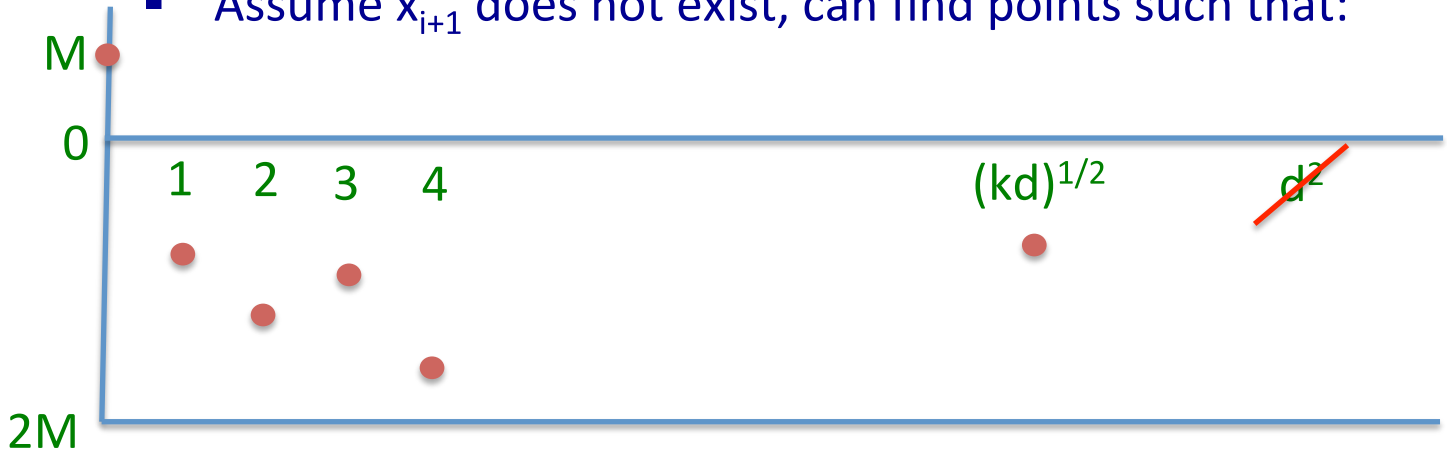
Theorem (Servedio-T-Thaler): Let F : [-1,1] -> [-1,1]. If deg(F) ≤ d and weight(F) ≤ W, then max |F'(t)| ≤ d log(W)

- Sharper than Markov as long as W ≤ 2^d
- Markov: "If F is bounded and attains large derivative, the its degree must be large"
- Us: "If F is bounded and attains large degree, then either its degree or its weight must be large"

using our Markov-type inequality

Theorem (Servedio-T-Thaler): Let F : [-1,1] -> [-1,1]. If deg(F) ≤ d and weight(F) ≤ W, then max |F'(t)| ≤ d log(W)

- Break k coordinates up into $(k/d)^{1/2}$ blocks of size $(kd)^{1/2}$
  - When is $(kd)^{1/2} < d^2$? When $d \geq k^{1/3}$.
- Suppose $p(x_i) = M$ for some x_i. Want to find $p(x_{i+1}) < -2M$
- If we can keep going, then $W \geq 2^{(k/d)^{1/2}}$
- Assume x_{i+1} does not exist, can find points such that:



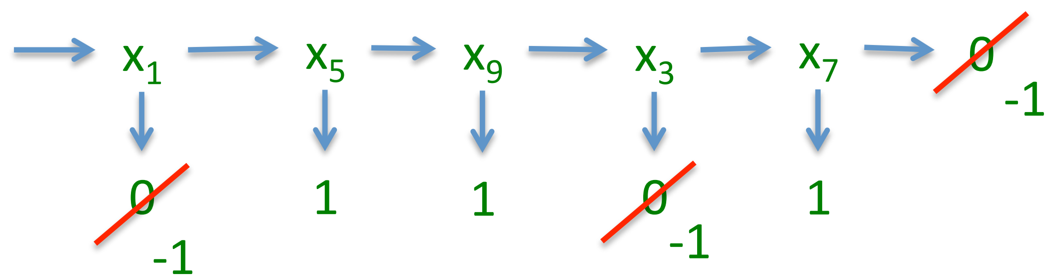
- Shifting and scaling, get F : [-1,1] -> [-1,1] such that |F'(t)| > $(kd)^{1/2}$ for some t in [-1,1]
- Apply our theorem to conclude that $W \geq 2^{(k/d)^{1/2}}$
- So we either find x_{i+1}, or directly conclude $W \geq 2^{(k/d)^{1/2}}$

rest of the talk

- Sketch of our construction of low-degree low-weight PTFs for DLs
- Introduce main technical tool: Chebyshev polynomials for L-infinity approximations

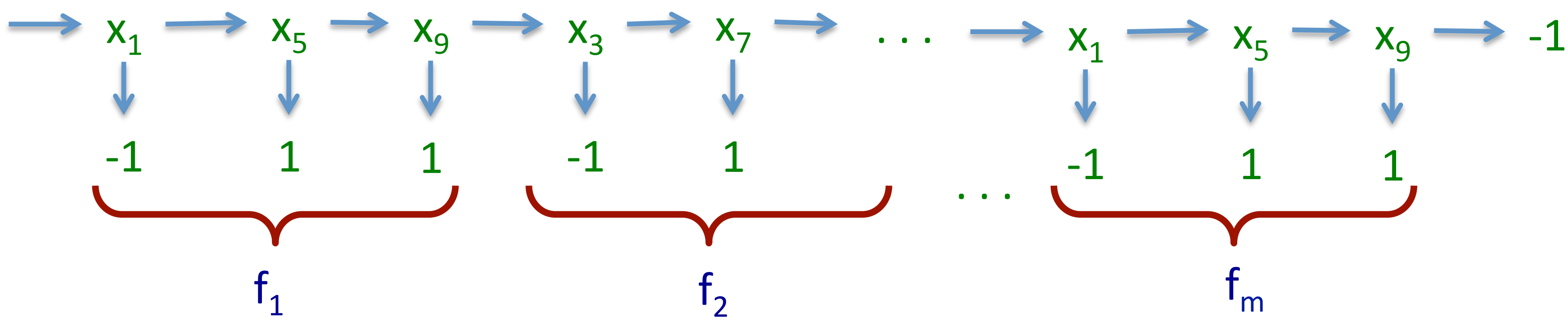
First, a minor technical point

Assume our DLs output {-1,1} instead of {0,1}

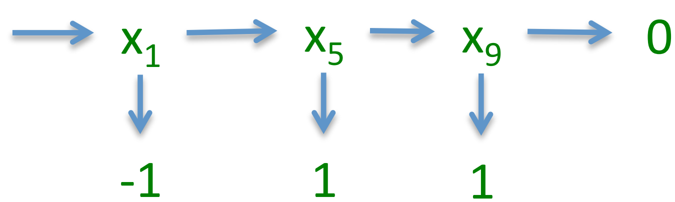


key idea for upper bound

Break DL upper into smaller DLs



$f_1 =$



Claim: $f = \mathrm{sgn}(3^m f_1 + 3^{m-1} f_2 + 3^{m-2} f_3 + \ldots + 3 f_m - 1)$

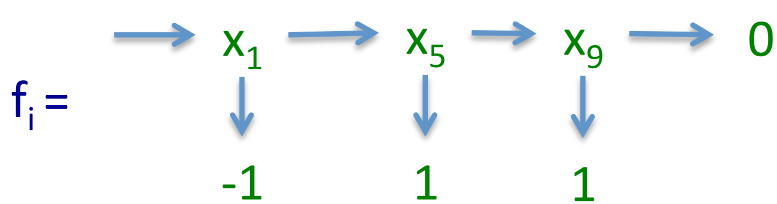
Proof. Suppose an input exits the list at f_i. Then $f_j(x) = 0$ for all $j < i$, and the weight of f_i overpowers the total weight of f_k for all $k > i$.

key idea for upper bound

$$f = \mathrm{sgn}(3^m f_1 + 3^{m-1} f_2 + 3^{m-2} f_3 + \ldots + 3 f_m - 1)$$

viewing f_i's as variables, this is a degree-1 weight 3^m polynomial

Suffices to get low-weight low-degree approximations p of each sub-DL f_i satisfying $|p(x)-f_i(x)| < \text{tiny}$ (we call these "L-infinity approximators")



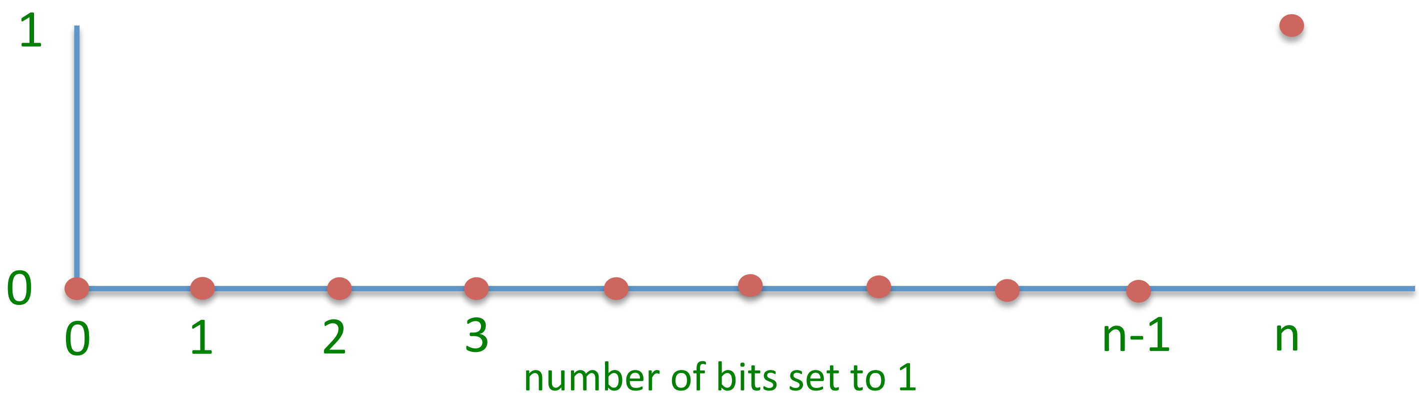
Can represent $f_i(x)$ as $(-1)\, x_1 + (1)\, (\neg x_1 \wedge x_5) + (1)\, (\neg x_1 \wedge \neg x_5 \wedge x_9)$

Suffices to get low-weight low-degree L-infinity approximators for AND! This is a well-studied (and well-understood) problem in approximation theory, and is a useful tool for many problems in concrete complexity.

approximating the AND function

$x_1 \wedge x_2 \wedge x_3 \wedge x_4 \wedge x_5 \wedge x_6 \wedge \ldots \wedge x_n$

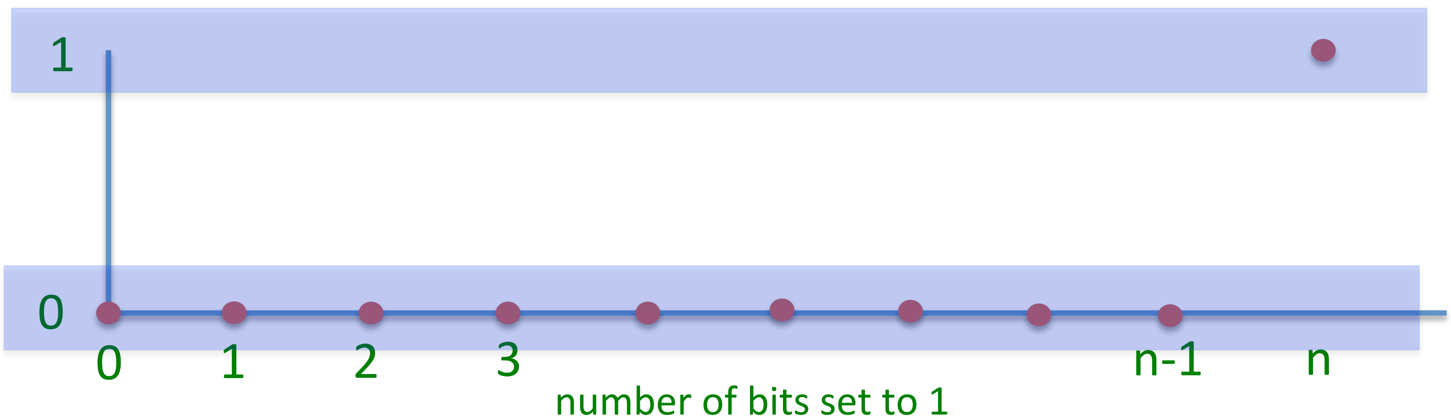
- Function is symmetric (*i.e.* its value only depends on the Hamming weight of the input)



- Any polynomial that interpolates these n points exactly has to have degree n (why?)
- Suppose we have a polynomial F such that
  - F(n) in [0.9, 1.1]
  - F(1),…,F(n-1) in [-0.1,0.1]
- Then $F(x_1 + x_2 + \ldots + x_n)$ is L-infinity approximator for AND

Goal: Low-degree polynomial F such that

- F(n) in [0.9, 1.1]
- F(1),…,F(n-1) in [-0.1,0.1]



Chebyshev approximators: There is a polynomial with degree $d = n^{1/2}$ and weight 2^d that achieves this

- More generally, can achieve ε error with degree $n^{1/2}\log(1/\varepsilon)$
- Matching upper and lower bounds for L-infinity approximators for all symmetric functions [Paturi 1992, Sherstov 2008, de Wolf 2008]

conclusion

- We make progress on the well-studied problem of attribute-efficiently learning decision lists
- Give provably optimal weight-degree tradeoffs for PTF computing decision lists
- Our upper bounds yield algorithms with best known running times and mistake bounds
- Our lower bounds suggest that significantly new techniques will be required to make further progress on the problem