Theoretical Computer Science 107 (1993) 63–76
Elsevier

On read-once threshold formulae and their randomized decision tree complexity

Rafi Heiman
The Weizmann Institute of Science, Rehovot, Israel

Ilan Newman and Avi Wigderson*
The Hebrew University, Jerusalem, Israel

Abstract

Heiman, R., I. Newman and A. Wigderson, On read-once threshold formulae and their randomized decision tree complexity, Theoretical Computer Science 107 (1993) 63–76.

TC^0 is the class of functions computable by polynomial-size, constant-depth formulae with threshold gates. Read-once TC^0 (RO-TC^0) is the subclass of TC^0 which restricts every variable to occur exactly once in the formula.

Our main result is a (tight) linear lower bound on the randomized decision tree complexity of any function in RO-TC^0.

This relationship between threshold circuits and decision trees bears significance on both models of computation. Regarding decision trees, this is the first class of functions for which such a strong bound is known. Regarding threshold circuits, it may be considered as a possible first step towards proving $TC^0 \neq NC^1$; generalizing our lower bound to all functions in TC^0 would establish this separation.

Another structural result we obtain is that a read-once threshold formula uniquely represents the function it computes.

1. Introduction

1.1. Boolean decision trees

The Boolean decision tree is an extremely simple model for computing Boolean functions. It charges only for reading input variables. Every function on n variables has complexity $\leqslant n$. Perhaps surprisingly, decision trees turned out to be fundamental

* Partially supported by Israel–American Binational Science Foundation Grant No. 87-00082

in studying the complexity of Boolean functions in general models, such as CREW-PRAM [10], and AC^0-circuits [9].

The first major result for this model was the linear lower bound of Rivest and Vuillemin [13] for the class of monotone graph properties, proving the Aanderaa–Rosenberg conjecture.

A conjecture that an $\Omega(n)$ lower bound applies to this class even if we allow randomization, is attributed to Karp. This has been proven for a few special monotone graph properties, but the best general lower bound is $\Omega(n^{2/3})$ of Hajnal [5] (improving on Yao [21] and King [8]).

Our main result exhibits a natural class of functions for which a linear lower bound holds. The proof combines generalizing techniques developed in [15] to study read-once formulae, and understanding "partial" computation of threshold functions by decision trees.

1.2. Threshold circuits

The study of circuits with threshold gates and, in particular, those of polynomial size and constant depth (the class TC^0) also has several motivations. These circuits capture essential aspects in neural net computations [14, 7]. They have been shown to be equivalent to constant-depth arithmetic circuits over finite fields [12, 19], and were recently related to simulating the polynomial hierarchy by counting oracles [20, 2].

The fundamental question of whether the inclusion $TC^0 \subseteq NC^1$ is proper, surfaced naturally after $AC^0 \neq TC^0$ was resolved ([1, 4] and their improvements), and after the results about constant-depth circuits with prime modulo gates were proved [11, 17]. This question has been under attack in the last few years.

Two important steps were made in the direction of separating TC^0 from NC^1. The first, by Hajnal et al. [6], separated depth-2 from depth-3 polynomial-size threshold circuits. The second, by Yao [22], separated the monotone analogues of the classes TC^0 and NC^1.

In 1986 Saks suggested a bold approach to separating these classes: Show that every function in TC^0 has high (say linear) randomized decision tree complexity (in terms of its deterministic complexity). This would suffice, as there are several examples of evasive (deterministic complexity n) functions in NC^1 with randomized complexity n^α for $\alpha < 1$ [18, 3, 15].

This approach reduces a lower bound in the circuits model to a lower bound in the information theoretical model of randomized decision trees. It is particularly original and intriguing, since the separation will be proved by showing that functions in the smaller class are harder (in the second model).

Our result can be considered as a first step in this direction. It proves the desired lower bound for read-once TC^0-functions. It is naive to be optimistic just because every TC^0-function is a simple projection of a read-once TC^0-function; it is not clear what happens to decision tree complexity under projections. However, the proof of the lower bound reveals that, from the point of view of randomized decision trees, hreshold gates are no more powerful than ANDs and ORs, which hints that this may ›e the right direction to pursue.

2. Definitions and statement of results

2.1. Boolean decision trees

A *deterministic decision tree* T is a labeled binary tree. Each nonleaf node is labeled by some input variable x_i. The two outgoing edges of such nodes are labeled, one by "1" and the other by "0". Each leaf is labeled by an output value which is either "1" or "0".

The *path* of T on the input setting $\varepsilon=\varepsilon_1,\ldots,\varepsilon_n\in\{0,1\}^n$, termed $\text{Path}_T(\varepsilon)$, is that (unique) path in the tree which starts at the root, and at each node, labeled x_i, follows the edge labeled ε_i. $\text{Var}_T(\varepsilon)$ denotes the set of variables labeling the nodes of $\text{Path}_T(\varepsilon)$. The *output* of T given ε, termed $\text{Output}_T(\varepsilon)$, is the bit labeling the leaf of $\text{Path}_T(\varepsilon)$. T *computes* the Boolean function f if $\text{Output}_T(\varepsilon)=f(\varepsilon)$ for every ε.

The *time* consumed by T, termed $\text{Time}_T(\varepsilon)$, is simply $|\text{Var}_T(\varepsilon)|$. (Every variable is probed at most once in a path.) The *complexity* of T is the time consumed for a worst-case input. The *deterministic decision tree complexity* of f, termed $DC(f)$, is the complexity of the best deterministic decision tree that computes f,

$$DC(f)=\min_T \max_\varepsilon \text{Time}_T(\varepsilon). \tag{1}$$

A *randomized decision tree* for f, RT, is a distribution over the deterministic decision trees for f. Given ε, a deterministic decision tree is chosen according to this distribution and is "executed". This makes the path and the time consumed, random variables (however, the output is always correct). The complexity of RT is the expected time (i.e., the expected number of variables it probes in order to determine the output) for a worst-case input. The *randomized decision tree complexity* of f, termed $RC(f)$, is the complexity of the best randomized decision tree that computes f,

$$RC(f)=\min_{RT} \max_\varepsilon \mathbf{E}_{T\in RT}[\text{Time}_T(\varepsilon)], \tag{2}$$

where $\mathbf{E}$ stands for expectation and $T\in RT$ stands for a random T chosen according to the distribution RT.

By a lemma of Yao [21], which is based on the minimax theorem, we have the following equivalence between $RC(f)$ and the *distributional complexity* of f.

$$RC(f)=\max_D \min_T \mathbf{E}_{\varepsilon\in D}[\text{Time}_T(\varepsilon)], \tag{3}$$

where D ranges over all distributions on input settings of f, T ranges over all deterministic decision trees for f, and $\varepsilon\in D$ stands for a random input setting ε chosen according to the distribution D. The distributional complexity is a useful tool for proving lower bounds. One can guess some D and then prove a lower bound on $\min_T \mathbf{E}_{\varepsilon\in D}[\text{Time}_T(\varepsilon)]$.

A *partial decision tree* T for f is very similar to a deterministic one, except that a leaf in it may contain a "?". T is required to satisfy $\text{Output}_T(\varepsilon)=f(\varepsilon)$ for every ε with $\text{Output}_T(\varepsilon)\neq$ "?". For example, the trivial decision tree, which contains a single node (a leaf) labeled by a "?", is a partial decision tree for every Boolean function. Central to our proof is an inequality satisfied by all partial decision trees computing a simple threshold function.

2.2. *Read-once threshold formulae*

A *threshold gate*, denoted T_l^k for some $k>1$ and $1\leqslant l\leqslant k$, is a Boolean gate with k inputs that outputs "1" iff at least l of its inputs are "1". For example, T_1^k and T_k^k are, respectively, OR- and AND-gates of fan-in k.

A *read-once threshold formula* is a formula with threshold gates in which each variable appears exactly once. We point out here that disallowing negation gates does not restrict the generality of our results. Negation gates can be "pushed" to be applied to inputs only. Then renaming all negative literals as positive ones (as input variables) does not change relevant combinatorial properties, such as the deterministic and the randomized decision tree complexities.

An example of read-once AND–OR formula is the AND–OR tree function $g^{(d)}$, defined for every depth d on $n=2^d$ input variables:

$$g^{(0)}(x_1)=x_1,$$

and

$$g^{(d+1)}(x_1,\ldots,x_{2^{d+1}})=g^{(d)}(x_1,\ldots,x_{2^d})\diamond g^{(d)}(x_{2^d+1},\ldots,x_{2^{d+1}}),$$

where $\diamond$ is AND for even d and is OR for odd d. This function is in NC^1; its formula depth is logarithmic in the number of variables. It is easy to see that its deterministic decision tree complexity is maximal, $DC(g^{(d)})=n$. However, its randomized complexity is low, $RC(g^{(d)})=\Theta(n^\alpha)$ for $\alpha=\log_2[(1+\sqrt{33})/4]=0.753\ldots$ [15]. The large (logarithmic) depth enables iterated savings that turn out to yield this low randomized complexity.

2.3. *Statement of results*

Our main result says that large depth is necessary for low randomized complexity.

Theorem 2.1. *Let f be a Boolean function computed by a read-once threshold formula of depth d over n input variables. Then $RC(f)\geqslant n/2^d$.*

The next section is devoted to the proof of this theorem. The proof is based on generalizing techniques of [15], as well as on using the new concept of partial decision ɜes. A weaker lower bound, namely $RC(f) \geqslant n/4^d$, can be proved more simply by .ing the lower-bound result of [15]. The direct proof given here is, we believe, a more ɜnificant step in the study of the randomized decision tree complexity in general, and threshold circuits in particular. This direct proof has another advantage. It works so in a more powerful model. This model enables, in particular, gates that compute bitrary symmetric functions.

efinition. A Boolean function g, defined on k input variables, is said to *contain a flip* there exists an l, $1 \leqslant l \leqslant k$, such that g outputs the same value whenever exactly l of its puts are "1", and it outputs the opposite value whenever exactly $l-1$ of its inputs e "1".

orollary. *Let f be a Boolean function computed by a read-once formula of depth d over input variables whose gates are functions that each contains a flip. Then* $RC(f) \geqslant n/2^d$.

One may verify that the proof given in the next section works for these gates as well.

Our second result says that a Boolean function that can be represented by a read-ıce threshold formula has a unique such representation.

efinition. A read-once threshold formula is *nondegenerate* if no input of some T_1^k-gate (OR) is the output of some other $T_1^{k'}$-gate and, similarly, no input of a T_k^k-gate ıND) is the output of a $T_{k'}^{k'}$-gate.

heorem 2.2. *Two nondegenerate read-once threshold formulae that compute the same oolean function are identical.*

This theorem is proved in Section 4.

Proof of Theorem 2.1

In this section we prove Theorem 2.1.

In the definitions of time and complexity above we assumed a unit cost for probing variable. In order to carry out an induction argument, we generalize these notions, ıd define them relative to a *variables cost function*, $c: \{x_1, \ldots, x_n\} \to \mathbf{R}$. Given such c we define $\text{Time}_{c,T}(\varepsilon) = \sum_{x_i \in \text{Path}_T(\varepsilon)} c(x_i)$. $DC(f, c)$ and $RC(f, c)$ denote the complexiɜs relative to c and are defined similarly to (1) and (2). Analogue to (3) is

$$RC(f, c) = \max_D \min_T \mathbf{E}_{\varepsilon \in D}[\text{Time}_{c,T}(\varepsilon)]. \tag{4}$$

1. Overview of the proof

For a formula consisting of a single threshold gate the proof is not very difficult, 'en if variables have nonunit costs. One can use this case as a single step in a top-down inductional proof. However, this does not yield a lower bound on $RC(f)$, but, rather, a lower bound on the complexity of *directional* randomized decision trees for f. Directionality means that variables are probed in a restricted manner, depending on the formula's structure; if any variable in any subformula is probed, then after this probe the decision tree must first figure out the value of that subformula before probing any variable that appears in another part of the formula.

This is the reason for the use of a bottom-up induction given in the next subsection, whose single step (the shrinking lemma) consists of a global statement on the formula. Interestingly, Santha [16] developed a proof for a similar problem that uses a top-down induction and need not use Yao's lemma. In the proof of the shrinking lemma (Section 3.3) we carefully define a distribution on inputs and a set of decision trees that enable reducing the lemma's statement into a statement involving a simple threshold formula only, i.e., a single gate. The analogue to the evaluation of a simple threshold function (for the directional case) is a claim on partial decision trees that compute a simple threshold function (for the general case). Section 3.4 is devoted to this claim.

3.2. Reducing Theorem 2.1 to the shrinking lemma

The shrinking lemma. *Let F be a read-once threshold formula of depth $d>0$ that computes a Boolean function f. Consider an internal gate T_l^k, whose entries are all variables. Denote these variables by $Y=\{y_1,\ldots,y_k\}$. Denote the rest of the variables by $X=\{x_1,\ldots,x_m\}$. (See Fig. 1.) Let $c: X\cup Y\to\mathbf{R}$ be a cost function for the $m+k$ variables of f. Let F' be the formula obtained from F by replacing the subformula $T_l^k(y_1,\ldots,y_k)$ by a single variable v (see Fig. 2), and let f' be the function computed by F'. Define a new cost function c' by $c'(x_i)=c(x_i)$, $\forall\, 1\leqslant i\leqslant m$, and $c'(v)=c(Y)/2$, where $c(Y)=\sum_{i=1}^{k} c(y_i)$. Then $RC(f',c')\leqslant RC(f,c)$.*

Theorem 2.1 follows by applying the lemma inductively. The beginning is with unit variables cost. The last shrinking yields the simple formula consisting of a single variable, v', whose cost bounds $RC(f)$ from below, and is

$$c(v')=\sum_{i=1}^{n} 2^{-\mathrm{Depth}(x_i)}\geqslant n/2^d,$$

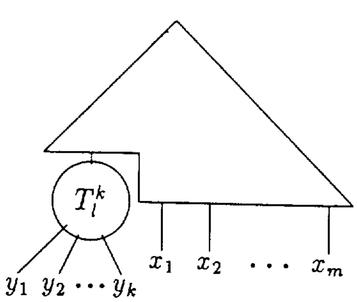

Fig. 1. The given F.

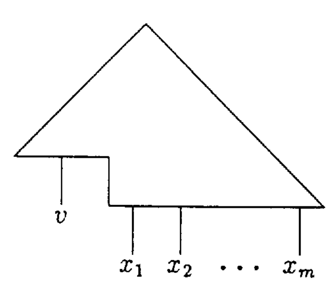

Fig. 2. The shrunk F'.

where $\text{Depth}(x_i)$ and d relate, respectively, to the depth of a variable x_i (which is well-defined, since x_i appears only once) and to the maximal depth over all variables in the (original) formula F.

3.3. *Reducing the shrinking lemma to the claim*

First, we introduce some necessary notations.

Notations: $[k]$ denotes the set $\{1,\ldots,k\}$. $\{{A \atop a}\}$ denotes the set of all subsets of a set A, which have cardinality a. θ^1 (θ^0) denotes the extension of a (partial) setting $\theta: X\to\{0,1\}$, on $X\cup\{v\}$ by $\theta^1(v)=1$ ($\theta^0(v)=0$). θ_M, where $M\subseteq[k]$ denotes the extension of $\theta: X\to\{0,1\}$ to $X\cup Y$ by $\theta_M(y_i)=1_{i\in M}$ (i.e., 1 if $i\in M$ and 0 otherwise).

$\Pr_D(E)$ denotes the probability of E, given a distribution D. $c(U)$ denotes the total cost of a subset U of input variables, $c(U)=\sum_{u\in U}c(u)$. 1_U denotes the input setting that gives 1 exactly to those variables in U. U_ε^T denotes the variables in some subset U of inputs that are probed by T given an input setting ε (to all variables), $U_\varepsilon^T=U\cap \text{Var}_T(\varepsilon)$.

To prove the shrinking lemma we have to show that $RC(f',c')\leqslant RC(f,c)$. Using (4) we show that

$$(\forall D')(\exists D)(\forall T)(\exists T')\ \mathbf{E}_{\varepsilon'\in D'}[\text{Time}_{c',T'}(\varepsilon')]\leqslant \mathbf{E}_{\varepsilon\in D}[\text{Time}_{c,T}(\varepsilon)], \tag{5}$$

where D (D') is a distribution on the input settings to f (f'), and T (T') is a deterministic decision tree for f (f').

Let D' be given. Define a distribution D as follows. For every X-setting $\theta: X\to\{0,1\}$ and subset $M\subseteq[k]$, define

$$\Pr_D(\theta_M)=\begin{cases}\Pr_{D'}(\theta^1)\cdot\Pr(M) & \text{if } |M|=l,\\ \Pr_{D'}(\theta^0)\cdot\Pr([k]\setminus M) & \text{if } |M|=l-1,\\ 0 & \text{otherwise,}\end{cases} \tag{6}$$

where

$$\Pr(S)=\frac{\sum_{i\in S}c(y_i)}{\binom{k-1}{|S|-1}\cdot c(Y)} \tag{7}$$

for a nonempty set $S\subseteq[k]$. (The point here is to split $\Pr_{D'}(\theta^1)$ and $\Pr_{D'}(\theta^0)$ among the extensions of θ that are difficult to separate. These are the extensions θ_M for which $|M|=l$ or $|M|=l-1$. The portion of probability that such an extension gets is proportional to the cost of the 'meaningful' Y-variables in it.)

Now, let T be given. We do not define T' explicitly. Rather, we define a set of candidate deterministic decision trees and prove that (5) holds for at least one of them. The candidates are the following $k\cdot\binom{k-1}{l-1}$ decision trees, $T_{(i,W)}$, indexed by pairs (i,W), where $i\in[k]$ and $W\in\{{[k]\setminus\{i\} \atop l-1}\}$.

$T_{(i,W)}$ is defined as the "projection" of T under the following actions:

(1) Each question "y_i?" (in T) is replaced by the question "v?" (in $T_{(i,W)}$).

(2) For each $j \in W$, $T_{(i,W)}$ assumes that $y_j = 1$. Namely, for each node of T containing the question "y_j?", $T_{(i,W)}$ passes down this question to the 1-direction while deleting the node and the whole subtree under the 0-direction. (See Fig. 3.)

(3) For all other j (i.e., $j \in [k] \setminus W, j \neq i$), $T_{(i,W)}$ assumes that $y_j = 0$: For each node of T containing "y_j?", $T_{(i,W)}$ similarly passes down the question, this case to the 0-direction. (See Fig. 4.)

It remains to show that inequality (5) holds for some $T_{(i,W)}$. We do this by proving that the following convex combination of these $k \cdot \binom{k-1}{l-1}$ inequalities holds.

$$\sum_{i \in [k],\, W \in \left\{ {[k] \setminus \{i\} \atop l-1} \right\}} p_{(i,W)} \mathbf{E}_{\varepsilon' \in D'} [\mathrm{Time}_{c', T_{(i,W)}}(\varepsilon')] \leqslant \mathbf{E}_{\varepsilon \in D} [\mathrm{Time}_{c,T}(\varepsilon)], \tag{8}$$

where the appropriate coefficients $\{p_{(i,W)}\}$ will be defined when used.

First we write the explicit terms for the two expectations above:

$$\mathbf{E}_{\varepsilon' \in D'} [\mathrm{Time}_{c', T_{(i,W)}}(\varepsilon')] \stackrel{\text{def}}{=} \sum_{\theta : X \to \{0,1\}} [\Pr_{D'}(\theta^1) \cdot \mathrm{Time}_{c', T_{(i,W)}}(\theta^1) + \Pr_{D'}(\theta^0) \cdot \mathrm{Time}_{c', T_{(i,W)}}(\theta^0)],$$

and

$$\begin{aligned} \mathbf{E}_{\varepsilon \in D} [\mathrm{Time}_{c,T}(\varepsilon)] &\stackrel{\text{def}}{=} \sum_{\theta : X \to \{0,1\}} \sum_{M \subseteq [k]} \Pr_D(\theta_M) \cdot \mathrm{Time}_{c,T}(\theta_M)] \\ &\stackrel{(6)}{=} \sum_{\theta : X \to \{0,1\}} [\Pr_{D'}(\theta^1) \cdot \sum_{M \in \left\{ {[k] \atop l} \right\}} \Pr(M) \cdot \mathrm{Time}_{c,T}(\theta_M) \\ &\quad + \Pr_{D'}(\theta^0) \cdot \sum_{M \in \left\{ {[k] \atop l-1} \right\}} \Pr([k] \setminus M) \cdot \mathrm{Time}_{c,T}(\theta_M)]. \end{aligned}$$

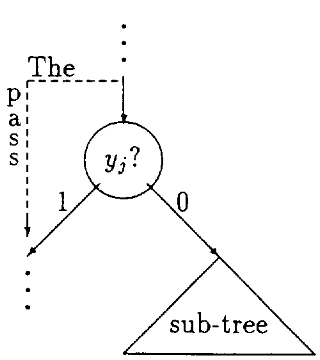

Fig. 3. Assuming $y_j = 1$.

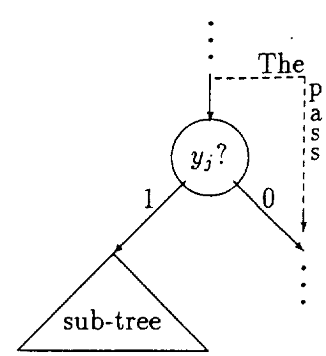

Fig. 4. Assuming $y_j = 0$.

Inserting these terms into (8), and using the definition of D, we note that it is sufficient to show for each $\theta: X \to \{0, 1\}$ that the following inequalities hold:

$$\sum_{i\in[k],\, W\in\binom{[k]\setminus\{i\}}{l-1}} p_{(i,W)} \mathrm{Time}_{c',T_{(i,W)}}(\theta^1) \leqslant \sum_{M\in\binom{[k]}{l}} \Pr(M)\cdot \mathrm{Time}_{c,T}(\theta_M), \tag{9}$$

and

$$\sum_{i\in[k],\, W\in\binom{[k]\setminus\{i\}}{l-1}} p_{(i,W)} \mathrm{Time}_{c',T_{(i,W)}}(\theta^0) \leqslant \sum_{M\in\binom{[k]}{l-1}} \Pr([k]\setminus M)\cdot \mathrm{Time}_{c,T}(\theta_M). \tag{10}$$

We prove (9), and (10) follows by duality: Change the roles of 1's and 0's in the Y-variables and consider the threshold gate T^k_{k-l+1}.

Next, we divide "Time" to the costs of X, Y and v, and use the notation above. Inequality (9) becomes

$$\sum_{(i,W)} p_{(i,W)}[c'(X_{\theta^1}^{T_{(i,W)}}) + c'(v)\cdot 1_{v\in \mathrm{Var}_{T_{(i,W)}}(\theta^1)}] \leqslant \sum_{M\in\binom{[k]}{l}} \Pr(M)[c(X^T_{\theta_M}) + c(Y^T_{\theta_M})]. \tag{11}$$

The key observation here is that $\mathrm{Path}_{T_{(i,W)}}(\theta^1)$ is the "projection" of $\mathrm{Path}_T(\theta_{\{i\}\cup W})$ under actions (1)–(3) above. In particular, $X_{\theta^1}^{T_{(i,W)}} = X^T_{\theta_{\{i\}\cup W}}$ and $v\in \mathrm{Var}_{T_{(i,W)}}(\theta^1)$ iff $y_i \in Y^T_{\theta_{\{i\}\cup W}}$. Using these and (7), and enumerating the pairs (i, W) as $\{(M, i)\colon M\in\binom{[k]}{l}, i\in M\}$, we find that (11) is equivalent to

$$\sum_{M\in\binom{[k]}{l}} \sum_{i\in M} p_{(i,W)}\cdot[c'(X^T_{\theta_M}) + c'(v)\cdot 1_{y_i\in Y^T_{\theta_M}}] \leqslant \frac{1}{\binom{k-1}{l-1}} \sum_{M\in\binom{[k]}{l}} \sum_{i\in M} \frac{c(y_i)}{c(Y)}\cdot[c(X^T_{\theta_M}) + c(Y^T_{\theta_M})].$$

By definition, $c'(X^T_{\theta_M}) = c(X^T_{\theta_M})$. To cancel these terms out we now define $p_i = c(y_i)/c(Y)$ and $p_{(i,W)} = p_i/\binom{k-1}{l-1}$ for $i\in[k]$ and $W\in\binom{[k]\setminus\{i\}}{l-1}$. (Note that these coefficients are nonnegative and their sum is 1.) Hence, canceling out and multiplying both sides by $\binom{k-1}{l-1}\cdot c(Y)$ reduces the last inequality to

$$c'(v)\cdot \sum_{M\in\binom{[k]}{l}} \sum_{i\in M} c(y_i)\cdot 1_{y_i\in Y^T_{\theta_M}} \leqslant \sum_{M\in\binom{[k]}{l}} \sum_{i\in M} c(y_i)\cdot c(Y^T_{\theta_M}),$$

which is equivalent to

$$c'(v)\cdot \sum_{M\in\binom{[k]}{l}} c(Y_M\cap Y^T_{\theta_M}) \leqslant \sum_{M\in\binom{[k]}{l}} c(Y_M)\cdot c(Y^T_{\theta_M}). \tag{12}$$

Note that we are now left with a problem involving the simple threshold subformula $F_{\text{sub}}=T_l^k(y_1,\ldots,y_k)$. The only role θ plays in (12) is to determine some projection of T that becomes a partial decision tree for F_{sub}. This projection is derived from T by passing down each x_i?-question to the direction $\theta(x_i)$. It is partial since T may compute F without computing F_{sub}. In other words, the claim in the next subsection implies (12) and completes the whole proof.

3.4. The reduced claim

The partial decision tree claim. *Let T be a partial decision tree for $T_l^k(y_1,\ldots,y_k)$, and let c be a leaf cost function on Y. Then*

$$\frac{c(Y)}{2}\cdot\sum_{L\in\binom{Y}{l}} c(L\cap \mathrm{Var}_T(1_L))\leqslant \sum_{L\in\binom{Y}{l}} c(L)\cdot c(\mathrm{Var}_T(1_L)).$$

Proof. The proof is by induction, with two base cases.

Base case 1: $l=k$ (AND-gate).

The only $L\in\binom{Y}{k}$ is $L=Y$ and the case follows.

Base case 2: $l=1$ (OR-gate).

T does not probe a variable more than once. Hence, it is of the form of Fig. 5, where $0\leqslant s\leqslant k$ and $Z=\{z_1,\ldots,z_s\}\subseteq Y$.

Denote $Y\setminus Z$ by $W=\{w_1,\ldots,w_{k-s}\}$. $L\in\binom{Y}{1}$ consists of an element $y\in Y$. If y is some z_i then $\mathrm{Var}_T(1_L)=\{z_1,\ldots,z_i\}$, and if y is some w_i then $\mathrm{Var}_T(1_L)=Z$. We thus have to show that

$$\frac{c(Y)}{2}\cdot\sum_{i=1}^{s} c(z_i)\leqslant\sum_{i=1}^{s}\left[c(z_i)\cdot\sum_{j=1}^{i} c(z_j)\right]+\left[\sum_{i=1}^{k-s} c(w_i)\right]\cdot\left[\sum_{j=1}^{s} c(z_j)\right].$$

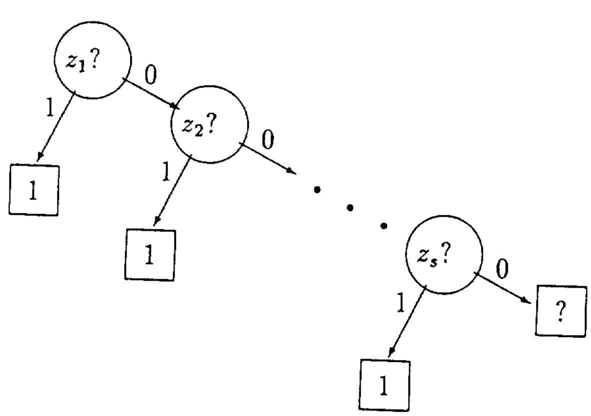

Fig. 5. A partial decision-tree for OR.

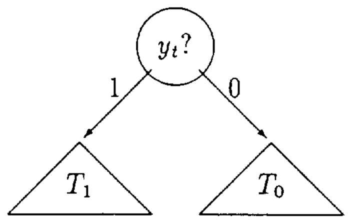

Fig. 6. A nontrivial tree T.

This indeed holds, since the quantity $\frac{1}{2}[\sum_{i=1}^{s} c(z_i)]^2 + [\sum_{i=1}^{k-s} c(w_i)] \cdot [\sum_{i=1}^{s} c(z_i)]$ is between the two values on both sides of the inequality, due to $\frac{1}{2}[\sum_{i=1}^{s} a_i]^2 \leqslant \sum_{i=1}^{s} [a_i \cdot \sum_{j=1}^{i} a_j]$.

The induction step: $1 < l < k$ (nontrivial threshold gate).

If T is trivial, i.e., it does not probe any variable, then for every L, $\mathrm{Var}_T(1_L)$ is empty, and the claim trivially holds.

Otherwise, let "y_t?" be the first question of T, and let T_1 and T_0 be the subtrees under the directions $y_t = 1$ and $y_t = 0$, respectively (see Fig. 6).

For L containing y_t, say $L = \{y_t\} \cup L'$, we have $\mathrm{Var}_T(1_L) = \{y_t\} \cup \mathrm{Var}_{T_1}(1_{L'})$. For L not containing y_t, we have $\mathrm{Var}_T(1_L) = \{y_t\} \cup \mathrm{Var}_{T_0}(1_L)$. In these terms the claim states that

$$\frac{c(Y)}{2} \cdot \left\{ \sum_{L' \in \binom{Y \setminus \{y_t\}}{l-1}} [c(y_t) + c(L' \cap \mathrm{Var}_{T_1}(1_{L'}))] + \sum_{L \in \binom{Y \setminus \{y_t\}}{l}} c(L \cap \mathrm{Var}_{T_0}(1_L)) \right\}$$
$$\leqslant \sum_{L' \in \binom{Y \setminus \{y_t\}}{l-1}} [c(y_t) + c(L')] \cdot [c(y_t) + c(\mathrm{Var}_{T_1}(1_{L'}))]$$
$$+ \sum_{L \in \binom{Y \setminus \{y_t\}}{l}} c(L) \cdot [c(y_t) + c(\mathrm{Var}_{T_0}(1_L))].$$

T_1 and T_0 are partial decision-trees for $T_{l-1}^{k-1}(Y \setminus \{y_t\})$ and $T_l^{k-1}(Y \setminus \{y_t\})$, respectively. Hence, by induction,

$$\frac{c(Y \setminus \{y_t\})}{2} \cdot \sum_{L' \in \binom{Y \setminus \{y_t\}}{l-1}} c(L' \cap \mathrm{Var}_{T_1}(1_{L'})) \leqslant \sum_{L' \in \binom{Y \setminus \{y_t\}}{l-1}} c(L') \cdot c(\mathrm{Var}_{T_1}(1_{L'})),$$

and

$$\frac{c(Y \setminus \{y_t\})}{2} \cdot \sum_{L \in \binom{Y \setminus \{y_t\}}{l}} c(L \cap \mathrm{Var}_{T_0}(1_L)) \leqslant \sum_{L \in \binom{Y \setminus \{y_t\}}{l}} c(L) \cdot c(\mathrm{Var}_{T_0}(1_L)).$$

Using these, and dividing by $c(y_t)$, the claim reduces to

$$\frac{c(Y)}{2} \cdot \binom{k-1}{l-1} + \frac{1}{2} \sum_{L'} c(L' \cap \mathrm{Var}_{T_1}(1_{L'})) + \frac{1}{2} \sum_{L} c(L \cap \mathrm{Var}_{T_0}(1_L))$$
$$\leqslant \binom{k-1}{l-1} \cdot c(y_t) + \sum_{L'} c(L') + \sum_{L'} c(\mathrm{Var}_{T_1}(1_{L'})) + \sum_{L} c(L),$$

and this holds due to

$$\frac{c(Y)}{2}\cdot\binom{k-1}{l-1}=\frac{1}{2}\cdot\sum_{L\in\left\{Y \atop l\right\}} c(L)$$
$$=\frac{1}{2}\binom{k-1}{l-1}\cdot c(y_t)+\frac{1}{2}\cdot\sum_{L'\in\left\{Y\setminus\{y_t\} \atop l-1\right\}} c(L')+\frac{1}{2}\cdot\sum_{L\in\left\{Y\setminus\{y_t\} \atop l\right\}} c(L).$$

This completes the proofs of the claim, the lemma and Theorem 2.1. □

4. Proof of Theorem 2.2

In this section we prove Theorem 2.2. The proof is by induction on the number of variables n. The case of $n=1$ is trivial.

Let f be computed by the two nondegenerate read-once threshold formulae $F_1=T_l^k(h_1,\ldots,h_k)$ and $F_2=T_s^r(g_1,\ldots,g_r)$. Since F_1 and F_2 are read once, each variable appears in positive form (with no negation) in F_1 if and only if it appears in positive form in F_2. Thus, we assume from now on that F_1 and F_2 are monotone. (Change names of negative variables if there are any.)

The proof uses partial assignments and examines the restricted function and the restricted formulae. We note here that a restricted formula may be degenerate, however, in such a case we always change it to a nondegenerate form by merging AND-gates together and OR-gates together. This does not change the type of the output gate.

Let H_i, $1\leqslant i\leqslant k$, and G_j, $1\leqslant j\leqslant r$, be the variable sets of h_i and g_j, respectively.

Proposition 4.1. *If $H_i=G_j$ for some i,j then $h_i=g_j$ (as functions and as formulae).*

Proof. Any partial assignment of "0" to the variables of H_i that assigns "0" to h_i leaves the restricted function independent of the variables of $H_i=G_j$ and, therefore, g_j becomes constant too. By monotonicity, this constant must be "0". The same argument on g_j implies that $h_i=g_j$ as functions. By the inductive hypothesis, they are identical as formulae too. □

Proposition 4.2. *If $1<l<k$, $1<s<r$, and $h_i=g_j$ for some i,j, then F_1 is identical to F_2.*

Proof. Assume (w.l.o.g) that $i=j=1$. If $l\geqslant 3$, assign "1" to the variables in H_1. F_1 reduces to $F_1'=T_{l-1}^{k-1}(h_2,\ldots,h_k)$ (where the output gate is neither AND nor OR). By the inductive hypothesis, F_2 reduces to the same formula. It follows that F_1 and F_2 are identical. Dually, if $l\leqslant k-2$, then by the assignment of the variables in H_1 to "0", we obtain the result. Therefore, we may assume that $l=s=2$ and $k=r=3$.

Assign "0" to the variables in H_2. F_1 reduces to AND(h_1,h_3). By the inductive hypothesis at least one of g_2 or g_3 must become "0" (so that the restricted F_2 will also have AND as its output gate). Assume g_2 becomes "0". It follows that $G_2 \subseteq H_2$. Now re-assign "0" to the variables in G_2. The same argument yields $H_2 \subseteq G_2$. We have $H_2 = G_2$ and Proposition 4.1 implies that $h_2 = g_2$. Similarly, $h_3 = g_3$. □

Proof of Theorem 2.2 (*continued*). We now return to the proof of Theorem 2.2. Assume (w.l.o.g) that $H_k \cap G_r \neq \emptyset$. Let $x \in H_k \cap G_r$. There are basically two cases.

Case 1. $1 < l < k$ and $1 < s < r$.

If $h_k = g_r = x$ then by Proposition 4.2 we are done. Otherwise, there is an assignment to x such that at least one of h_k and g_r does not become constant; say it is h_k. The output gate of F_1 does not change by this restriction (so it is neither AND nor OR). By the inductive hypothesis, the two restricted formulae must be identical. In particular, the output gate of F_2 does not become AND or OR and since $k, r \geqslant 3$, there exist $i, j, i \neq k$, $j \neq r$ for which $h_i = g_j$. Again, by Proposition 4.2 we are done.

Case 2. $l = k$, i.e., $F_1 = \text{AND}(h_1, \ldots, h_k)$.

First assume that $s < r$ and obtain a contradiction as follows. Assign "1" to x. F_1 reduces to some nonconstant formula, F_1'. F_2 reduces to either $T_s^r(g_1, \ldots, g_{r-1}, g_r')$ or $T_{s-1}^{r-1}(g_1, \ldots, g_{r-1})$ (but the latter is possible only if $s \geqslant 2$). In any case, the output gate is not AND, and by the inductive hypothesis so is the output gate of F_1'. This is possible only if $k = 2$ and $F_1' = h_1$. Comparing the variable sets of the two restricted formulae, we deduce that $H_2 \subseteq G_r$. We obtain the contradiction by assigning "0" to all variables of H_2; F_1 becomes "0" while F_2 does not.

So far we have $F_2 = \text{AND}(g_1, \ldots, g_2)$. Assign "1" to the variables of H_k, yielding $\text{AND}(h_1, \ldots, h_{k-1}) = \text{AND}(g_1', \ldots, g_r')$ (the latter might be degenerate). By the inductive hypothesis on the h_i's and by the fact that each h_i cannot have AND as its output gate (otherwise F_1 is degenerate), we get that for every $i \leqslant k-1$, there is some j such that $H_i \subseteq G_j$. Similarly, for every $j \leqslant r-1$, there is some i such that $G_j \subseteq H_i$. Note that the H_i's are pairwise disjoint, and so are the G_j's. It follows that $r = k$, and that for every $i < k$ there is a (unique) $j < r$ such that $H_i = G_j$. Therefore, $H_k = G_r$ too. By Proposition 4.1, $h_i = g_j$ for every pair i, j as above, and also $h_k = g_r$.

The case where one of the output gates is an OR-gate, is dual to the last case above. □

Acknowledgment

We thank Uriel Feige, David Harel, Mike Saks and Moshe Tennenholz for helpful discussions and comments.

References

[1] M. Ajtai, Σ_1^1-formulae on finite structures, *Ann. Pure and Appl. Logic* **24** (1983) 1–48.

[2] E. Allender, A note on the power of threshold circuits, in: *Proc. 30th IEEE Symp. on Foundations of Computer Science* (1989) 580–584.

[3] R.B. Boppana, private communication.
[4] M. Furst, J. Saxe and M. Sipser, Parity, circuits and the polynomial time hierarchy, *Math. Systems Theory* **17** (1984) 13–27.
[5] P. Hajnal, An $\Omega(N^{4/3})$ lower bound on the randomized complexity of graph properties, Tech. Report 88-19, University of Chicago, 1988.
[6] A. Hajnal, W. Maass, P. Pudlák, M. Szegedy and Gy. Turán, Threshold circuits of bounded depth, in: *Proc. 28th IEEE Symp. on Foundations of Computer Science* (1987) 99–110.
[7] J.J. Hopfield, Neural network and physical systems with emergent collective computational abilities, *Nat. Acad. Sci. USA* **79** (1982) 2554–2558.
[8] V. King, Lower bounds on the complexity of graph properties, in: *Proc. 20th ACM Symp. on Theory of Computing* (1988) 468–476.
[9] N. Linial, Y. Mansour and N. Nisan, Constant depth circuits, Fourier transform, and learnability, in: *Proc. 30th IEEE Symp. on Foundations of Computer Science* (1989) 574–579.
[10] N. Nisan, CREW PRAMs and decision trees, in: *Proc. 21st ACM Symp. on Theory of Computing* (1989) 327–335.
[11] A.A. Razborov, Lower bounds on the size of bounded depth networks over a complete basis with logical addition, *Math. Notes* **41** (1987) 333–338.
[12] J. Reif, On threshold circuits and polynomial computations, in: *Proc. 2nd Structure in Complexity Theory Conf.* (1987) 118–125.
[13] R. Rivest and S. Vuillemin, On recognizing graph properties from adjacency matrices, *Theoret. Comput. Sci.* **3** (1978) 371–384.
[14] D.E. Rumelhardt, J.L. McClelland and the PDP research group, *Parallel Distributed Processing: Exploration in the Microstructure of Cognition, Vol. 1* (MIT Press, Cambridge, MA, 1986).
[15] M. Saks and A. Wigderson, Probabilistic Boolean decision trees and the complexity of evaluating game trees, in: *Proc. 27th IEEE Symp. on Foundations of Computer Science* (1986) 29–38.
[16] M. Santha, On the Monte Carlo Boolean decision tree complexity of read-once formulae, in: *Proc. 6th Structure in Complexity Theory Conf.* (1991) 180–187.
[17] R. Smolensky, Algebraic methods in the theory of lower bounds for Boolean circuit complexity, in: *Proc. 19th ACM Symp. on Theory of Computing* (1987) 77–82.
[18] M. Snir, Lower bounds for probabilistic linear decision trees, *Theoret. Comput. Sci.* **38** (1985) 69–82.
[19] C. Sturtivant, G. Frandsen and J. Boyar, Is finite field arithmetic a restricted model of computation?, manuscript, 1989.
[20] S. Toda, On the computational power of PP and ⊕ P, in: *Proc. 30th IEEE Symp. on Foundations of Computer Science* (1989) 514–519.
[21] A.C. Yao, Lower bounds to randomized algorithms for graph properties, in: *Proc. 28th IEEE Symp. on Foundations of Computer Science* (1987) 393–400.
[22] A.C. Yao, Circuits and Local Computation, in: *Proc. 21st ACM Symp. on Theory of Computing* (1989) 186–196.