TOPIC 3
A Time-Space Tradeoff

(Related sections: 2)

Summary: Proof that every nondeterministic branching program computing an explicit function $f : D^n \to \{0,1\}$ in linear time must have exponential size.

In this lecture we will consider branching programs computing boolean functions $f : D^n \to \{0,1\}$ with $|D| > 2$. Our goal is to prove that some explicit functions f require branching programs of exponential size if the time is required to be linear in n, i.e. does not exceed kn for some constant k. Since the logarithm of the size of a branching program is of the same oder as the space used by a Turing machine, this gives a non-trivial tradeoff between space and time.

A celebrated recent result of Miklós Ajtai (1999) gives such a lower bound for the boolean case (when $|D| = 2$), but the proof in this case is based on very subtle combinatorial and probabilistic arguments. On the other hand, for larger domains such lower bounds can be obtained using a simple averaging argument, and we present it below.

We stress that, although we allow the domain D be large, the argument we describe below works also in the case of *nondeterministic* branching programs. The argument of Ajtai used for the boolean case $D = \{0,1\}$ essentialy employs the fact that the underlying branching program is *deterministic*: in this case input vectors are in one-to-one correspondence with the computations on them. Proving nontrivial lower bounds for nondeterministic time kn branching programs over $D = \{0,1\}$ remains an open problem, even for $k = 1+\varepsilon$ with an arbitrary small constant $\varepsilon > 0$.

The function $f : D^n \to \{0,1\}$, for which we are going to show that any nondeterministic time kn branching program must be large, is easy to describe. The domain D is the Galois field $\mathrm{GF}(q)$ where q is a sufficiently large with respect to k prime power.[1] The function $f(Y,x)$ has $n = m^2 + m$ variables, the first m^2 of which are arranged in an $m \times m$ matrix Y. Then $f(Y,x) = 1$ if and only if $Y \cdot x = 0$, i.e., if x is orthogonal (over $\mathrm{GF}(q)$) to all the rows of Y.

3.1. The model

We consider functions $f : D^n \to \{0,1\}$ where D is a finite domain. A standard model to compute such functions $f(x_1, \dots, x_n)$ is that of (deterministic) branching programs. Such a program is a directed acyclic graph whose nodes are labeled by

[1] "Sufficiently large" here means that $q \geq k^{\Omega(k)}$. The functions considered in Beame et. al. (1998) require $q \geq 2^{2^{\Omega(k)}}$.

variables x_i and have out-degree d; the outgoing edges are labeled by tests "$x_i = \delta$" for all $\delta \in D$. Sinks (i.e. out-degree 0 nodes) are labeled by constants 0 and 1. Such a program computes f in a natural way. We declare one node as a *source*. Given an input string $a = (a_1, \dots, a_n) \in D^n$, start the computation at the source; if the computation reaches a node labeled by x_i, then follow the outgoing edge labeled by "$x_i = a_i$". That is, we follow the path, the tests along which are consistent with the actual values of the corresponding coordinates of the input string. Proceeding this way, the computation reaches a sink; its label is the value $f(a)$.

We can introduce *nondeterminism* by allowing so-called guessing nodes. These nodes are unlabeled and have an arbitrary out-degree. If a computation reaches such a node then it can proceed further by following any of the outgoing edges. Such a program accepts an input string if and only if at least one path from the source to a 1-sink is consistent with this input. In this lecture we consider only nondeterministic programs.

The *size* of a program P, size(P), is the number of non-guessing nodes.

We say that a program computes a given function f in *time* T if for every input $a \in f^{-1}(1)$ there is a path from the source to a 1-sink which is consistent with a (i.e. is an accepting path) and along which at most T tests are made. Note that we do not require that other paths accepting this input a, must be short—it is enough that at least one of them is such!

3.2. Main lemma

In this section we prove that if a function f can be computed by a small program, then there is an input $a \in f^{-1}(1)$ and two disjoint subsets Z_1, Z_2 of its bits such that *simultaneausly* changing these bits in many ways does not change the value of f.

We will consider branching programs working in time $T \le kn$. Throughout this section, let

$$K := 2\sum_{j=0}^{k} \binom{3k^2}{k};$$

hence, $K \le k^{O(k)}$.

Lemma 3.1. *Let $f : D^n \to \{0,1\}$, $F \subseteq f^{-1}(1)$ and $\mu(F) = |F|/|D|^n$ be the density of F. Suppose that f can be computed by a nondeterministic branching program P in time kn. Then for every $s \le n/(2K)$ there exist two disjoint s-element subsets of variables Z_1, Z_2, an assignment $w : X \setminus (Z_1 \cup Z_2) \to D$, and sets A_j, $j = 1,2$ of assignments $a : Z_i \to D$ such that*

(i) *$f(a_1, w, a_2) = 1$ for all $(a_1, a_2) \in A_1 \times A_2$;*
(ii) *$|A_j| \ge \frac{1}{\Delta} \cdot |D|^s$ for both $j = 1,2$;*[2]
(iii) *$|A_j| \ge \frac{1}{\sqrt{\Delta}} \cdot |D|^s$ for at least one $j = 1,2$,*

[2]In this note we will not use this (stronger) property to derive explicit lower bounds. We include it just because the proof of this property for the boolean case ($|D| = 2$) with Δ much closer to $size(P)$ is the key of the whole Ajtai's argument. If time allows, we will try to discuss also this proof later.

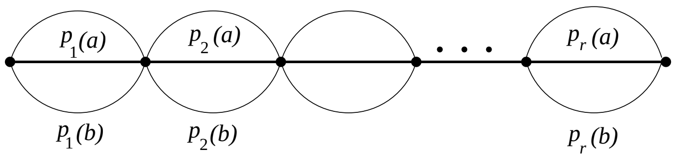

Figure 1. One factor-class of computations

where

$$\Delta = \frac{K^2\binom{n}{s}^2 \cdot \mathrm{size}(P)^{3k^2}}{2\mu(F)}.$$

Proof. We first prove the lemma under the assumption that, for every input $a \in F$, there is a computation $comp(a)$ along which no variable is tested more than k times; such programs are known as nondeterministic read-k branching programs (k-n.b.p.). We then show that essentially the same argument works for the programs of length kn.

Let P be a k-n.b.p. computing f. We may assume w.l.o.g. that each variable is tested along $comp(a)$ at least once. This can be achieved by adding a sequence of n "dummy" tests of all variables at the source; the size of a b.p, increases only by an additive factor of n, moreover the resulting b.p. is $(k+1)$-n.b.p.

For each input $a \in F$, fix one accepting read-k computation path $comp(a)$. Take $l := 3k$ and split each such path into $r \le lk$ sub-paths

$$comp(a) = \big(p_1(a), \dots, p_r(a)\big)$$

of length $\lceil n/l \rceil$ each (except, perhaps, the last sub-path); the length of a sub-path $p_i(a)$ is the number of different variables tested along it. That is, we have r *time intervals*, and in the i-th of them the computation on a follows the sub-path $p_i(a)$.

Factorize the (selected) computations $comp(a) = \big(p_1(a), \dots, p_r(a)\big)$, $a \in F$, according to the following equivalence relation: two paths are equivalent if the starting nodes of the corresponding sub-paths $p_1(a), \dots, p_r(a)$ coincide (see Fig. 1).

Since we have only size(P) nodes and each factor class corresponds to a sequence of at most $r \le 3k^2$ nodes, the number of factor-classes does not exceed

$$S := \mathrm{size}(P)^{3k^2}.$$

Thus, there is a subset $A \subseteq F$ of inputs such that $|A| \ge |F|/S$ and all the computations $comp(a)$ on inputs $a \in A$ belong to one factor-class. From now on we concentrate only on these computations.

By our assumption, no variable can appear in more than k time intervals. (Recall that we have $r \le kl$ such intervals, each of length $\lceil n/l \rceil$.) Define the *trace* $T(a, x)$ of a variable x along the computation $comp(a)$ as the set of those (at most k) time intervals along which this variable is tested.

Say that a pair Z_1, Z_2 of disjoint s-element subsets of variables is *good* for an input $a \in A$ if there exit two disjoint traces T_1, T_2 such that for both $j = 1, 2$ all the variables in Z_j have the same trace T_j along the computation $comp(a)$.

Claim 3.2. [3] *Every input $a \in A$ has at least one good pair.*

[3]Cf. Exercise 2.12 in the book.

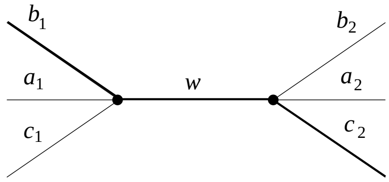

Figure 2

Proof. Take an input $a \in A$ and partite the set of all variables according to their traces along the computation *comp*(a). Since we have only $\sum_{j=0}^{k}\binom{r}{j} \leq K/2$ possible traces, the average size of one block in this partition is at least $2n/K$. Hence at least $n/2$ of the variables belong to blocks of size at least $n/K \geq s$.

Let Y be the union of these (large) blocks; hence, $|Y| \geq n/2$. The fact that the variables from different blocks have different traces alone does not yield a good pair – we need these traces be *disjoint*. To obtain such a pair we will use the fact that each sub-path can test at most n/l variables.

Take any of large blocks Z_1. By the definition, $|Z_1| \geq s$ and all the variables in Z_1 have the same trace T_1, say, $T_1 = \{1, \ldots, k\}$. The corresponding to this trace sub-paths $p_1(a), \ldots, p_k(a)$ can test at most $k \cdot \frac{n}{l} < \frac{n}{2} \leq |Y|$ different variables. Hence, there must be a variable $y \in Y$ which is tested in none of these sub-paths. This means that the trace $T(a, y)$ of y is disjoint from the trace T_1 of the variables in Z_1. Let Z_2 be the (unique) large block containing y. Since all the variables in Z_2 have the same trace $T_2 = T(a, y)$, the pair Z_1, Z_2 is good for a, as desired. □

We have shown that every input $a \in A$ has at least one good pair Z_1, Z_2. The number of all possible pairs of disjoint s-element subsets Z_1, Z_2 of variables does not exceed

$$\binom{n}{s}\binom{n-s}{s} \leq \binom{n}{s}^2,$$

and the number of possible pairs T_1, T_2 of (corresponding to these subsets of variables Z_1, Z_2) traces does not exceed K^2. Hence,[4] some pair must be good for many inputs: there exists a pair Z_1, Z_2 of disjoint s-element subsets of variables, a pair T_1, T_2 of disjoint traces, and a subset of inputs $B \subseteq A$ such that

$$|B| \geq |A| \cdot \binom{n}{s}^{-2} \cdot K^{-2} \geq |F| \cdot S^{-1} \cdot \binom{n}{s}^{-2} \cdot K^{-2} = \frac{2}{\Delta} \cdot |D|^n.$$

and the following holds for every computation *comp*$(a) = (p_1(a), \ldots, p_r(a))$ on an input $a \in B$ and every time interval $i = 1, \ldots, r$:

(*) *If $i \in T_j$ then no variable from Z_{3-j} is tested along $p_i(a)$.*

We will use this property of these inputs to "cut-and-paste" them. Let $b_1, c_1 : Z_1 \to \{0,1\}$ be an arbitrary assignments to the variables in Z_1, $b_2, c_2 : Z_1 \to \{0,1\}$

[4]This is the step which makes the Lemma useless in the case of boolean programs. At this place Ajtai argues in a much more subtle way: he picks the time intervals at random and shows that the obtained pairs Z_1, Z_2 for different inputs $a \in A$ are "not very different."

an arbitrary assignments to the variables in Z_2, and $w : X \setminus (Z_1 \cup Z_2) \to \{0,1\}$ an assignment to the remaining variables (see Fig. 2).

Claim 3.3. *If both inputs $b = (b_1, w, b_2)$ and $c = (c_1, w, c_2)$ belong to B, then also the combined inputs (b_1, w, c_2) and (c_1, w, b_2) belong to B.*

Proof. We will make use of the fact that the computations $comp(a) = \big(p_1(a), \ldots, p_r(a)\big)$ on all inputs $a \in B$ belong to the same factor-class. That is, the starting nodes of the corresponding sub-paths $p_1(a), \ldots, p_r(a)$ are the same for all inputs $a \in B$.

Take the combined input $e = (b_1, w, c_2)$, and consider the computation

$$comp(e) = \big(p_1(e), \ldots, p_r(e)\big)$$

on it. In oder to prove that e belongs to B, it is enough to show that during each time interval $i = 1, \ldots, r$, the i-th sub-path $p_i(e)$ coincides with the i-th sub-path of the computation on *at least one* input from B; then the pair Z_1, Z_2 is also good for e, implying that input e must belong to B (recall that B is a *maximal* subset of A for which this pair is good).

If $i \in T_1$ then, by (*), only the variables from $X \setminus Z_2$ can be tested along $p_i(c)$. Since on these variables e coincides with c, we have that in this case $p_i(e) = p_i(b)$.

If $i \notin T_1$ then, again by (*), only the variables from $X \setminus Z_1$ can be tested along $p_i(b)$. Since on these variables e coincides with b, we have that in this case $p_i(e) = p_i(b)$. □

Let $B_j(a)$ denote the set of all inputs $b \in B^n$ such that b coincides with a on $X \setminus Z_j$. We will use the fact that the set of inputs B is large, $|B| \geq \frac{2}{\Delta} \cdot |D|^n$, to show that for at least one input $a \in B$ *both* sets $B_1(a)$ and $B_2(a)$ must be large.

Claim 3.4. *There exists an $a \in B$ such that $|B_j(a)| \geq \frac{1}{\Delta} \cdot |D|^s$ for both $j = 1, 2$.*

Proof. Given an $j \in \{1, 2\}$, we can partition the set B into at most $|D|^{n-s}$ classes: two inputs are in one class if they coincide outside Z_j. Each class of this partition can have at most $|D|^s$ inputs. Say that a class is *small* if it has $< \frac{1}{\Delta} \cdot |D|^s$ inputs. The average size of a class is at least $N = |B|/|D|^{n-s}$. Hence, for every $0 \leq \gamma \leq 1$, at most $\gamma \cdot |B|$ inputs from B can belong to classes of size at most γN (see Exercise 2.11 in the book). Taking

$$\gamma := |B|^{-1} \cdot \frac{1}{\Delta} \cdot |D|^n$$

we obtain that at most

$$\gamma \cdot |B| = \frac{1}{\Delta} \cdot |D|^n < |B|/2$$

inputs can belong to small classes.[5] Therefore, there must be an input $a \in B$ which belongs to large classes in *both* partitions (w.r.t. Z_1 and Z_2). □

Fix now an input $a \in B$, guaranteed by Claim 3.4, and let A_j $(j = 1, 2)$ be the projection of $B_j(a)$ onto the set Z_j. Since all the inputs from $B_1(a) \cup B_2(a)$ coincide on the set $X \setminus (Z_1 \cup Z_2)$, Claim 3.3 gives us the first condition (i) of Lemma 3.1. The second condition (ii) is also fulfilled by Claim 3.4, because $|A_j| \geq |B_j(a)|$ for both $j = 1, 2$. To ensure the third condition (iii), we can use the following version of Claim 3.4.

[5]Because $\gamma N = \frac{1}{\Delta} \cdot |D|^s$ and $|B| > \frac{1}{\Delta}|D|^n$.

Claim 3.5. *There exists an $a \in B$ such that $|B_j(a)| \geq \frac{1}{\sqrt{\Delta}} \cdot |D|^s$ for at least one $j = 1, 2$.*

Proof. Arguing as in the previous claim with $Z = Z_1 \cup Z_2$ instead of Z_i, we obtain a subset $B' \subseteq B$ of $|B'| \geq \frac{1}{\Delta} \cdot |D|^{2s}$ inputs such that all they coincide on $X \setminus Z$. Let A_j be the projection of these inputs onto Z_j. Since, by Claim 3.3, $|B'| = |A_1| \cdot |A_2|$, the claim follows. □

This completes the proof of Lemma 3.1 in the case of k-n.b.p.

In the (more general) case of branching programs of length at most kn the argument is the same: just observe that if a path has length kn then at least $n/2$ variables must be tested along it at most $2k$ times. Hence, if $comp(a) = (p_1, \ldots, p_r)$ then at least $n/2$ of the variables appear in at most $2k$ of the intervals $p_1, \ldots, p_r$; call these variables *legal* for a, and consider the traces of only these variables. The rest is the same with n replaced by $n/2$ and k replaced by $2k$. □

3.3. Application to linear codes

Let $q \geq 2$ be a prime power and consider the domain $D = GF(q)$. Given an $n \times n$ matrix A over $GF(q)$, let $f_A : D^n \to \{0, 1\}$ denote the function such that $f_A(x) = 1$ if and only if $A \cdot x = 0$, i.e. if the vector x is orthogonal (over $GF(q)$) to all the rows of A.

Theorem 3.6. **([17])** *There exists an absolute constant $c > 0$ such that for every positive integer k and for every prime power $q \geq k^{ck}$ there exists an $n \times n$ matrix A over $GF(q)$ such that every nondeterministic branching program computing $f_A(x)$ in time kn has size exponential in $n \cdot k^{-O(k)}$.*

Although the matrix A is not explicitly given, we can take it as an input an obtain an *explicit* function $f'(A, x) = f_A(x)$ in $N = n^2 + n$ variables such that every nondeterministic branching program computing this function in time kn has size at least $2^{\Omega(\sqrt{N}/k^{O(k)})}$. Moreover, in this case the time spent to test the variables in A may be arbitrarily long.

Proof. Take $\beta := 1/(2K)$ and $\varepsilon := \beta + 1/n$. From the Gilbert-Varshamov bound (see any textbook in Coding Theory), we know that for all sufficiently large n there exist linear codes $F \subseteq GF(q)^n$ of minimum relative distance ε and rate[6] $1 - H_q(\varepsilon)$; that is, $\log_q |F| \geq (1 - H_q(\varepsilon))n$ and every two vectors from F differ in at least εn coordinates. Let A be the parity-check matrix of F, that is,

$$F = \{x \in D^n : A \cdot x = 0\}.$$

Consider the function $f(x) := f_A(x)$. This function accepts $|F| \geq 2^{(1-H_q(\varepsilon))n}$ vectors, and the Hamming distance between any two of them is at least εn.

Let P be a nondeterministic branching program computing f in time kn, and set $s = \lceil \beta n \rceil$. By Lemma 3.1, there exist two disjoint s-element subsets of variables Z_1, Z_2; an assignment $w : X \setminus (Z_1 \cup Z_2) \to D$, and sets A_i, $i = 1, 2$ of assignments

[6]Here $H_q(x)$ is the q-ary entropy function defined by $H_q(0) = 0$ and

$$H_q(x) = x \log_q(q - 1) - x \log_q x - (1 - x) \log_q(1 - x)$$

for $0 < x \leq (q - 1)/q$.

$a : Z_i \to D$ such that $f(a_1, w, a_2) = 1$ for all $(a_1, a_2) \in A_1 \times A_2$ and $|A_j| > \frac{1}{\sqrt{\Delta}} \cdot q^s$ for at least one $j = 1, 2$, where

$$\Delta = \frac{K^2\binom{n}{s}^2 \cdot \mathrm{size}(P)^{3k^2}}{2\mu(F)}.$$

But since each two codewords must differ in at least $\varepsilon n > s$ coordinates, none of the sets A_1, A_2 can have more than one assignment. Hence,

$$\Delta > q^{2s}.$$

Since $\mu(F) \geq q^{-nH_q(\varepsilon)}$, this implies that $\mathrm{size}(P)^{3k^2}$ must be at least

$$\frac{2\mu(F) \cdot \Delta}{K^2\binom{n}{s}^2} \geq \frac{2q^{2s-H_q(\varepsilon)}}{K^2\binom{n}{s}^2} \geq 2^{\alpha n},$$

where (remembering that $s = \lceil \beta n \rceil$ and $\binom{n}{s} \leq (en/s)^s$)

$$\alpha = (2\beta - H_q(\varepsilon)) \cdot \log q - 2\beta \log(e/\beta) - (2 \log K)/n.$$

Let us estimate $H_q(x)$. Using $\log_q x = (\log_2 x)/(\log_2 q)$ we obtain

$$\begin{aligned} H_q(x) &= x\frac{\log(q-1)}{\log q} - x\frac{\log x}{\log q} - (1-x)\frac{\ln(1-x)}{\ln q} \leq x + x\frac{\log(1/x)}{\log q} + x\frac{1-x}{\ln q} \\ &\leq x \cdot \left(1 + \frac{1 + \log(1/x)}{\log q}\right). \end{aligned}$$

In our case $x = \varepsilon = \beta + 1/n \sim \beta = 1/(2K) = 1/k^{\Theta(k)}$. Hence, if $\log q \geq ck \log k$ for a sufficiently large constant c, then $H_q(\varepsilon) \leq (3/2)\beta$. If the constant c is even larger, then

$$\alpha \geq (1/2)\beta \cdot \log q - 2\beta \log(e/\beta) - (2 \log K)/n \geq \beta.$$

Thus, if $\log q \geq ck \log k$ and $c > 0$ is a sufficiently large constant, then $\alpha \geq \beta = k^{-\Theta(k)}$ and the desired lower bound $\mathrm{size}(P) \geq 2^{\alpha n/(3k^2)} \geq 2^{\Omega(n/k^{O(k)})}$ follows. □

Ex. 3.7. Let $f : D^n \to \{0,1\}$. A *Boolean version* of a function $f : D^n \to \{0,1\}$ is a Boolean function f' on $N = n \cdot \lceil \log_2 |D| \rceil$ variables obtained by encoding the elements of D by strings in $\{0,1\}^m$ with $m = \lceil \log_2 |D| \rceil$. Show that every k'-n.b.p. for f' can be transformed to a $|D|$-way k-n.b.p. for f of the same size with $k \leq k' \cdot \lceil \log_2 |D| \rceil$. Why the lower bound for the function f_A given by Theorem 3.6 does not yield a non-trivial lower bound on the size of (boolean) k'-n.b.p. computing f'_A, even for $k' = 2$?