BRICS

Basic Research in Computer Science

# On Ajtai's Lower Bound Technique for $R$-way Branching Programs and the Hamming Distance Problem

Jakob Pagter

BRICS Report Series RS-00-11

ISSN 0909-0878 May 2000

BRICS RS-00-11 J. Pagter: Ajtai, Branching Programs and the Hamming Distance Problem

See back inner page for a list of recent BRICS Report Series publications.
Copies may be obtained by contacting:

BRICS
Department of Computer Science
University of Aarhus
Ny Munkegade, building 540
DK–8000 Aarhus C
Denmark

| | |
|---|---|
| Telephone: | +45 8942 3360 |
| Telefax: | +45 8942 3255 |
| Internet: | BRICS@brics.dk |

BRICS publications are in general accessible through the World Wide Web and anonymous FTP through these URLs:

`http://www.brics.dk`
`ftp://ftp.brics.dk`
This document in subdirectory `RS/00/11/`

# On Ajtai's Lower Bound Technique for $R$-way Branching Programs and the Hamming Distance Problem

Jakob Pagter[*]

**BRICS**[†]
Department of Computer Science
University of Aarhus
Denmark

### Abstract

In this report we study the proof employed by Miklos Ajtai [*Determinism versus Non-Determinism for Linear Time RAMs with Memory Restrictions*, 31st Symposium on Theory of Computation (STOC), 1999] when proving a non-trivial lower bound in a general model of computation for the HAMMING DISTANCE problem: given $n$ elements decide whether any two of them have "small" Hamming distance. Specifically, Ajtai was able to show that any $R$-way branching program deciding this problem using time $O(n)$ must use space $\Omega(n \lg n)$.

We generalize Ajtai's original proof allowing us to prove a time-space trade-off for deciding the HAMMING DISTANCE problem in the $R$-way branching program model for time between $n$ and $\alpha n \frac{\lg n}{\lg \lg n}$, for some suitable $0 < \alpha < 1$. In particular we prove that if space is $O(n^{1-\epsilon})$, then time is $\Omega(n \frac{\lg n}{\lg \lg n})$.

[*] E-mail: pagter@brics.dk. Part of this work was done while the author was visiting University of Toronto.

[†] **B**asic **R**esearch in **C**omputer **S**cience,
Centre of the Danish National Research Foundation.

# 1 Introduction

Recently Miklos Ajtai [Ajt99b] was able to prove that any $R$-way branching program using sub-linear space deciding Element Distinctness must use super-linear time. The technical details of the proof are complicated. However, the basic ideas underlying the proof are not and based on these Ajtai is able to prove an even stronger bound for the so called Hamming Distance problem—a generalization of Element Distinctness—using much simpler arguments. An interesting aspect of these bounds is that they are valid in the well known RAM model with word size $\lg R$, as time in the $R$-way model corresponds to the number of times we read a word from the input, and thus bounds in this model are valid for RAMs independent on the specific instruction set.

The Element Distinctness *theorem* is the more interesting of Ajtai's two theorems, as Element Distinctness is an interesting and very well-studied problem not only when considering time, but also when considering time-space trade-offs [BFMadH$^+$87, Kar86, Yao94]. On the other hand, the Hamming Distance *proof* is arguably the more interesting of the two proofs, as it is gives the best possible bound in the model using a clear and elegant proof.

The purpose of this report is to study the latter proof with several goals: 1) To what extent does the Hamming Distance proof generalize, i.e. for what intervals of time and space do we have a lower bound in the form of a time-space trade-off? 2) How strong is the proof, e.g. can the simpler of the two proofs actually be used to prove something for Element Distinctness? 3) Can the Hamming Distance proof be used to give non-trivial lower bounds in the Boolean model?

The answer to the two latter questions is negative, but the proof can be generalized to achieve the following:

**Theorem 1 (General theorem, vanilla version)** *Any $R$-way branching program deciding* Hamming Distance *in time $O(k(n) \cdot n)$ , $k(n) < \alpha \frac{\lg n}{\lg \lg n}$*[1] *for a suitable constant $0 < \alpha < 1$ must use space*

$$S(n) > \beta \frac{n \lg n}{k(n)^{10k(n)}},$$

*for some constant $\beta > 0$.*

[1] Following [Knu98] we use "lg" to denote the logarithm in base 2.

The full version of this theorem which can be found in Section 4 will imply all three answers. For now, let us just observe that Theorem 1 implies that any $R$-way branching program, and thus *any* RAM with word size $\lg R$, solving the Hamming Distance problem using $O(n^{1-\epsilon})$ bits of space must use time $\Omega(n\frac{\lg n}{\lg\lg n})$. In terms of time this is the strongest result that can be achieved using our results. Similarly, using time $O(n\frac{\lg n}{\lg\lg n}))$ implies using space $\Omega(n^{1-\epsilon})$ for some $0 < \epsilon < 1$.

The contributions of this report is the actual generalization of Ajtai's original proof, together with some technical modifications allowing us to achieve a trade-off for time up to $\Omega(n\frac{\lg n}{\lg\lg n})$ rather than $\Omega(n\frac{\lg\lg n}{\lg\lg\lg n})$, which is what the immediate generalization yields. We also observe that the proof works for randomized branching programs with one-sided error. Our final contribution is the presentation of the proof, which we hope is more accessible than the original.

# 2 Previous work

Proving lower bounds in general models of computation is notoriously hard. Early approaches restricted the computational model, for example to comparison based models. Natural models such as the RAM do not obey such restrictions. Another approach has been to restrict the resources available, e.g. limiting the amount of space which we allow a given algorithm to use. This gives rise to time-space trade-offs, for which several breakthroughs have recently been achieved [Ajt99a, Ajt99b, BST98].

A model that has been very useful for this line of study is the branching program model. In particular much insight has been gained using the $R$-way branching program model introduced by Borodin and Cook [BC82]. Results obtained in this model include general lower bounds for Sorting [Bea91, BC82] and Universal Hashing [MNT93]. The problems for which the bounds are obtained are characterized by having a large output domain, which is essential for the employed proofs. It was only recently that non-trivial lower bounds for a decision problem were obtained in this model [Ajt99b, BST98].

When proving upper bounds for problems like those mentioned in this report (Hamming Distance, Element Distinctness, Sorting, Universal Hashing), the *de facto* standard model is the transdichotomous RAM. Much debate has taken place over which instruction set is better, giving rise to some confusion in the literature. One of the very nice properties of $R$-way branching programs is that they are strictly more powerful than a RAM with word size $\lg R$ and *any* instruction set as we measure time in terms of how many times we look at the input, meaning that the bounds obtained in this model will hold in the RAM no matter what specific instruction set we are working with.

For a great many people the holy grail of computational complexity is to show non-trivial (i.e. $T \in \omega(n)$ or even just $T > n$) lower bounds for Boolean decision problems. When studying Boolean complexity in the time-space setting the model used is the Boolean branching program. The first non-trivial bound in this model was given by Beame et al. [BST98], who exhibited a problem for which they could prove a lower bound of $1.0178\,n$ (sic) on time for space restricted to being sub-linear. Ajtai [Ajt99a] later found another problem for which he was able to prove that super-linear time is required when space is sub-linear. His proof relies heavily on the ELEMENT DISTINCTNESS proof from [Ajt99b].

Lower bounds in the $R$-way model can be transferred to the Boolean model—at a cost.

**Proposition 1** *Suppose we can prove a lower bound of $f(n)$ for some problem in the $R$-way branching program model, then this translates into a $f(n)/\lg R$ lower for the same problem in the Boolean branching program model. Likewise for space.*

**Proof:** In the Boolean model we measure the size of the input, $n$, in bits. In the $R$-way model $n$ refers to the number of words *each* consisting of $\lg R$ bits. Thus, if we translate inputs of size $n$ from the $R$-way model, they will become inputs of size $n \lg R$ bits in the Boolean model. □

So if we have that $R \in n^{O(1)}$, we need $\omega(n \lg n)$ bounds in the $R$-way model to infer $\omega(n)$ bounds in the Boolean model.

# 3 Model of computation

We employ the $R$-way branching program model, which is at least as strong as a RAM with word size $\lceil \lg R \rceil$ and any instruction set. In this report we will only give an informal presentation of the model. For a detailed account see [BC82] or [Sav98]. Boolean branching programs are the special case of $R = 2$, corresponding to looking at one bit at a time.

An $R$-way branching program is a directed acyclic graph. It has one node with in degree 0 which is called the *start node*. Every node with out degree 0 is called a *terminal node*, and is labelled either "YES" or "NO", depending on whether the branching program accepts or rejects the given input. Every node which is not a terminal node is called a *computation node* and is labelled with some index $i$ (referring to the $i$'th word of the input). A computation node has exactly $R$ outgoing edges, each with a unique label from $1, \ldots, R$. The input to the branching program consists of $n$ elements from some universe of size $R$ (which may be a function of $n$), and computation proceeds as follows: we start in the start node where we read the word indexed by the label of this node; if the value in this word is $r$, then we follow the outgoing edge with label $r$. We continue this procedure for the computation nodes we encounter until we end up in some terminal node which tells us the result of the computation.

As measures of complexity we define time $T$ to be the height of our branching program, and space $S$ to be $\lceil \lg \#\text{nodes} \rceil$.

For the rest of this report we will assume all branching programs to be levelled, which means that we can partition the nodes into $T$ disjoint sets $V_1, V_2, \ldots V_T$, such that all edges originating from $V_i$ go to $V_{i+1}$. For asymptotic purposes we may assume this without loss of generality. For proofs and definition see e.g. Borodin et al. [BFK$^+$81].

In this model of computation all "internal" computation is free, we only "pay" for reading the input. As a consequence we have the following easy proposition.

Let $\Delta \subseteq U \times U$ be a binary relation on a universe $U$. For such a relation we define a computational decision problem $D_\Delta \subseteq U^n$: given $x = \langle x(1), x(2), \ldots, x(n) \rangle$, $D_\Delta(x) =$ "YES" if and only if $\exists i \neq j : \Delta(x(i), x(j))$.

**Proposition 2** *Let $\Delta$ be any binary relation on a universe $U$ of size $R$. We can decide $D_\Delta$ on $U^n$ in time $T$ and space $S$ such that*

$$T \cdot S \in O(n^2 \lg R),$$

*for time between $n$ and $n^2$.*

**Proof:** First observe that in time $n$ we may decide $\Delta$ using $R^n$ nodes or $n \lg R$ bits of space, using an $R$-way decision tree—i.e. reading all the inputs once, remembering them, and then exploit the free internal computation.

This easily generalizes by splitting the input into $b(n)$ blocks each containing $n/b(n)$ words. Now what we will do is do decide $\Delta$ for each of the $O(b(n)^2)$ pairs of blocks. For each pair we may do the computation in time $2n/b(n)$ using $2(n/b(n)) \lg R$ bits of space. In total we use time $O(nb(n))$ and space $O((n/b(n)) \lg R)$, yielding the desired trade-off. □

For time $n$ this construction off course works for *any* problem, which shows that for time $\Theta(n)$ one cannot give space-bounds better than $\Omega(n \lg n)$ (when $R = 2^{c \lg n}$), i.e. we cannot hope for a better general bound for *any* problem in this model.

# 4 The result

Before stating the result, a few definitions are required. We study problems on inputs $x = \langle x(1), x(2), \dots, x(n) \rangle \in U^n$—typically $U = \{0,1\}^{c \lg n}$ (where $c$ is a suitable fixed natural number), i.e. $x(i) \in \{0,1\}^{c \lg n}$ and $|U| = n^c$. We emphasize the difference in notation between $x_i \in U^n$ (an entire input consisting of $n$ elements from $U$) and $x(i) \in U$ (the i'th element from $U$ of the input $x$).

For $0 < \gamma < \frac{1}{2}$ we define the parameterized Hamming Distance relation $HD_\gamma(a,b) \subset \{0,1\}^{c \lg n} \times \{0,1\}^{c \lg n}$ which relates $a$ and $b$ if and only if they differ in at most $\gamma c \lg n$ positions. To alleviate notation $HD$ will refer to $HD_{1/4}$ and Hamming Distance, or $D_{HD}$, will refer to the computational version of this relation with $\gamma = 1/4$. Equality is the problem of deciding whether the input contains to identical words. It should be clear that Hamming Distance is a natural generalization of Equality [2].

A relation $\Delta$ is said to be $\lambda(n)-$full iff for any pair of subsets of $U$ of size bigger than $\lambda(n)|U|$ there exists a pair of elements, one from each of the two subsets, satisfying the relation. Formally, $\Delta$ is $\lambda(n)-$full if the following holds

$$\forall A, B \subseteq U : \, |A|, |B| > \lambda(n)|U| \quad \Rightarrow \quad \exists a \in A, \, b \in B : \, \Delta(a,b) \qquad (1)$$

It should be clear that Equality is $\frac{1}{2}-$ full, as any two subsets of $U$ containing more than half of $U$ will have a non-empty intersection. In [Ajt99b] it is proved that for all $0 < \gamma < \frac{1}{2}$, $HD_\gamma$, and in particular $HD$, is $2^{-\delta \lg n}-$ full on $U = \{0,1\}^{c \lg n}$ where $c$ and $\delta$ are natural numbers suitably chosen (independent on $n$, but depending on $\gamma$) such that $c > \delta$.

[2] When Ajtai says Element Distinctness in [Ajt99b], he actually means the dual problem Equality.

This means that if we have two subsets of $U$ of size bigger than $2^{(c-\delta)\lg n}$, we are guaranteed to have two elements with low Hamming distance.

A relation $\Delta$ is said to be $\zeta(n)-$sparse iff the number of inputs of length $n$ for which $D_\Delta(x) =$ "YES" is less than $\zeta(n)|U|$ (a $\zeta(n)$ fraction of all possible inputs). Intuitively this means that the computational version of the relation has many "NO" instances. In [Ajt99b] it is proved that for all $0 < \gamma < \frac{1}{2}$ $HD_\gamma$ is $\frac{1}{2}-$ sparse for a fixed $c \in \mathbf{N}$ (recall that $|U| = n^c$) and $n$ sufficiently large. This means that for at least half of all input to HAMMING DISTANCE the answer is "NO". Ajtai [Ajt99b] also proves that EQUALITY is $c_= -$ sparse for a suitable fixed $c_= > 0$.

We are now ready to state the result.

**Theorem 2 (General theorem, full version)** *Let $\Delta$ be a $\lambda(n)-$full and (non-trivial) $\zeta(n)-$sparse relation on $U \times U$ with $|U| = R$. Consider an $R$-way branching program deciding $D_\Delta$, in time $T(n) = k(n) \cdot n$ and space $S(n)$. If,*

$$72\, k(n) \lg k(n) < \lg \tfrac{1}{\lambda(n)}, \tag{2}$$

*and*

$$\lg \tfrac{1}{\zeta(n)} < \frac{n \lg \frac{1}{\lambda(n)}}{k(n)^{5k(n)}}. \tag{3}$$

*Then,*

$$S(n) \geq \frac{n \lg \frac{1}{\lambda(n)}}{k(n)^{4k(n)}}. \tag{4}$$

Let us discuss this theorem. First of all, many of the constant in the above statement may possibly be improved, albeit not significantly.

What kind of fullness is required to achieve a non-trivial result? ¿From Ajtai's paper it might seem as if the HAMMING DISTANCE proof works only for what we might call *polynomial fullness* i.e., $1/|U|^{O(1)}$-fullness, whereas the ELEMENT DISTINCTNESS (or rather EQUALITY) proof handles *constant fullness.* If $k(n) \in O(1)$, we see from (2) that constant fullness, specifically $\lambda(n) < 2^{-144}$, actually does give rise to a non-trivial lower bound. However, it does not seem to be the case that the parameters of the proof can be improved enough to achieve anything for $\frac{1}{2}-$fullness, i.e., we cannot prove anything for ELEMENT DISTINCTNESS, but we can get closer than what Ajtai's original statement suggests.

If $\lambda(n)|U| < 1$ the relation in question would be trivial to decide (as two subsets of size 1 would then be enough to ensure satisfaction of the fullness property), hence we can assume that $\lambda(n)|U| \geq 1$. Combining this with (2) yields,

$$\lg |U| \geq \lg \frac{1}{\lambda(n)} > 72\, k(n) \lg k(n).$$

Recalling Proposition 1, we see that the best result we can achieve in the Boolean model is,

$$\frac{T(n)}{\lg |U|} < \frac{k(n) \cdot n}{126\, k(n) \lg k(n)} \in \omega(n).$$

In conclusion, no matter how we might chose the parameters, we can get no non-trivial implications for the Boolean model.

As stated previously, on $U = \{0,1\}^{c \lg n}$ Hamming Distance is $n^{\delta}$ – full and $\frac{1}{2}$ – sparse, giving Theorem 1 as corollary to Theorem 2. Interesting special cases include Ajtai's original theorem: time in $O(n)$ implies space in $\Omega(n \lg n)$, and space $O(n^{1-\epsilon})$ implies time $\Omega(n \frac{\lg n}{\lg \lg n})$.

# 5 Proof of Theorem 2

We would like to stress that almost all the arguments closely follow those of Ajtai [Ajt99b], yet we give the proof in full detail. This has two reasons: 1) in the generalized version many constants, say $c$, are replaced by functions, $c(n)$, and it is imperative to give the full proof to see exactly where the proof holds and where it breaks down; 2) it makes it clear exactly where we deviate from the original proof.

The proof presented here deviates from that of Ajtai in two aspects. When counting we use tighter estimates where it is possible, this has little effect on Ajtai's original proof, but significantly extends the interval of time for which the generalized proof works—an immediate generalization will work for time up to roughly $n \frac{\lg \lg n}{\lg \lg \lg n}$, whereas our proof works up to $\Theta(n \frac{\lg n}{\lg \lg n})$. Another deviation is our proof of Lemma 1 (corresponding to Lemmas 7 and 8 in [Ajt99b]). Besides these technical differences, changes have also been made for reasons of presentation.

The overall intuition behind the proof is as follows. Suppose we have a "large" set of inputs for which any algorithm basically behaves the same, i.e. it gives the same answer for all these inputs. Suppose further that the nature of the problem at hand is such that if we have a large set of inputs (of a certain kind), then for at least one of these inputs must be accepted. Ajtai's scheme is to use the space restriction to construct a large set of inputs that are all rejected by the algorithm. The structure and size of this set will be such that we can utilize the fullness property of our problem to ensure that at least one input, which we will call $x_{\text{fail}}$, in the above set must be accepted. This of course gives a contradiction on the assumed time and space restrictions of the algorithm.

Assume we have an $R$-way branching program, $\mathcal{A}$, deciding $D_\Delta$ in time $k(n) \cdot n$ and space $S(n)$ satisfying (2) and (3) but *not* (4). We will show that then there exists an input $x$ such that $\mathcal{A}(x) =$ "NO" but $D_\Delta(x) =$ "YES", contradicting our assumption that is, assuming (2) and (3) we may conclude (4).

Define $\texttt{state}(x,t)$ as the state of $\mathcal{A}$ (one of $2^{S(n)}$) after time step $t$ on input $x$. For a time interval $I$, define $\texttt{state}(x,I)$ to be $\texttt{state}(x,t_I)$, where $t_I$ is the *last* element of $I$ that is, given $x$ and a time interval $I$ we get the state of the program when leaving $I$. An arbitrary set of times $T$ may be written as a disjoint union of maximal intervals $I_j$, i.e. $T = \cup_j I_j$. If we assert that $\texttt{state}(x,T) = \texttt{state}(y,T)$ we mean that $\forall j : \texttt{state}(x,I_j) = \texttt{state}(y,I_j)$.

We will construct $x_{\text{fail}}$ from another input $x$ with some desirable properties. The first property is that $D_\Delta(x) =$ "NO" and hence $\mathcal{A}(x) =$ "NO", as $\mathcal{A}$ is assumed to decide $D_\Delta$ correctly. Suppose that we have two disjoint subsets of indices $W_1, W_2 \subset \{1,\ldots,n\}$ and two disjoint subsets of times $T_1, T_2 \subset \{1,\ldots,k(n)\cdot n\}$. Suppose now that on input $x$ the indices of $W_i$ $(i = 1,2)$ are not read outside $T_i$; let

$$\begin{aligned} S_i^x \subseteq \ & \{x_i | (\text{ obtained by modifying } x \text{ on } W_i \text{ only}) \\ & \qquad\qquad\qquad \wedge \quad \texttt{state}(x,T_i) = \texttt{state}(x_i,T_i)\}, \end{aligned}$$

i.e. $x$ and $x_i$ are identical outside $W_i$ and each time we leave $T_i$ on both $x$ and $x_i$ we are in the same state. Hence $\mathcal{A}(x) = \mathcal{A}(x_i)$ for all $x_i \in S_i^x$ as the only differences between the two inputs are in $W_i$ and hence "forgotten" when we leave $T_i$, the only place where $W_i$ is read.

Based on $x$ we can thus construct $x_1$ and $x_2$ such that $\mathcal{A}(x) = \mathcal{A}(x_1) = \mathcal{A}(x_2) =$ "NO". As $W_1$ and $W_2$ are disjoint we may make these two different changes to $x$ simultaneously, obtaining the input $x_{\text{fail}}$. We would like to ensure that $\mathcal{A}(x) = \mathcal{A}(x_{\text{fail}}) =$ "NO", but currently this might not be the case. The reason is that in the set of time intervals, say $T_1$, $\mathcal{A}$ is in principle able to look outside $W_1$ and hence $\mathcal{A}$ might look at $W_2$. This is not a problem as long as $x$ on $W_2$ is unchanged, but when making the two changes simultaneously we no longer have any guarantee that the states are fixed. The way to eliminate this problem is to enforce that the indices of $W_j$ are not read in $T_i$ on $x_i$, removing the possibility that $\mathcal{A}$ detects the change in $W_2$ when in $T_1$ and vice versa.

We can now construct $x_{\text{fail}}$ such that $\mathcal{A}(x_{\text{fail}}) = \mathcal{A}(x) =$ "NO" by making changes to $x$ on $W_1$ and $W_2$. The plan is of course to choose $x_{\text{fail}}$ such that $D_\Delta(x_{\text{fail}}) =$ "YES", giving the desired contradiction. One way to achieve this is to have so many choices for each of $x_1$ and $x_2$ that we can put the fullness property into play. If $S \subset U^n$, let $S(l)$ be $S$ projected onto the $l$'th dimension (or index).

**Proposition 3** *Let $W \subseteq \{1, \ldots, n\}$ and let $S \subset U^n$ be a set of inputs that are all identical outside $W$. If $|S| > (\lambda(n)|U|)^{|W|}$ then for at least one $l \in W$ it will be the case that $|S(l)| > \lambda(n)|U|$, i.e. some $x(l)$ (over $x$'s in $S$) must take on more than $\lambda(n)|U|$ values from $U$.*

**Proof:** Suppose that $\forall l \in W$ it is the case that $|S(l)| \leq \lambda(n)|U|$ then clearly $|S| \leq (\lambda(n)|U|)^{|W|}$. □

If we have many different inputs on $W_i$ to choose from, there will be an index for which we have many values to choose from, allowing us to exploit the fullness property.

Based on the above idea, we define a $\mu(n)-$*hard set*, $H_\mathcal{A}$, for a branching program $\mathcal{A}$ deciding $D_\Delta$ in time $k(n)\cdot n$.

**Definition 4 (Hard Set)** *A set of inputs $H_\mathcal{A} \subset U^n$ is called a $\mu(n)-$hard set for a branching program $\mathcal{A}$ deciding $D_\Delta$ in time $k(n)\cdot n$ and space $S(n)$ if we have*

- $W_1, W_2 \subset \{1, \ldots, n\}$ *with* $W_1 \cap W_2 = \emptyset$ *and* $|W_i| \geq \mu(n)\cdot n$,
- $T_1, T_2 \subset \{1, \ldots, k(n)\cdot n\}$ *with* $T_1 \cap T_2 = \emptyset$,

*such that* $\forall x \in H_\mathcal{A}$ :

- $D_\Delta(x) =$ *"NO".*
- *The indices of $W_i$ are only read in $T_i$ (on $x$).*
- `state`$(x, I_{ij})$ *is fixed for all the intervals comprising $T_i$—i.e. every time we leave $T_i$, $\mathcal{A}$ will behave identically for all $x \in H_\mathcal{A}$.*

□

The proof of Theorem 2 splits naturally in three parts. In Lemma 1 we show that if we have a large hard set, we may obtain $x_1$ and $x_2$ in "many" ways (relative to the fullness property). Then in Lemma 2 we show that given the time and space restrictions there exists a "large" hard set (relative to the sparseness property). Finally these two lemmata are combined.

**Lemma 1** *Let $\Delta$ be a $\lambda(n)$–full relation on $U$, $\mathcal{A}$ be a branching program deciding $D_\Delta$ in time $k(n) \cdot n$ and space $S(n)$, and let $H_\mathcal{A}$ be a $\mu(n)-$hard set for $\mathcal{A}$ so that,*

$$|H_\mathcal{A}| > 4\lambda(n)^{\mu(n)\cdot n}|U|^n,$$

*then there exists some $x \in H_\mathcal{A}$ from which we can find $x_{fail}$ such that $D_\Delta(x_{fail}) =$ "NO" but $\mathcal{A}(x) =$ "YES".*

**Proof:** Define $x_{|W}$, where $W \subseteq \{1, \ldots, n\}$, to be $x$ where all values at positions are overwritten with some standard symbol, say $\perp \notin U$. Suppose we have some set of inputs $H \subseteq U$ and some set of indices $W \subseteq \{1, \ldots, n\}$; define

$$S^x = \{y \in H | x_{|W} = y_{|W}\}, \tag{5}$$

i.e. the set of elements in $H$ which are identical to $x$ outside $W$. We claim the following

$$\#x \in H_\mathcal{A} \text{ with } |S^x| \leq \tfrac{1}{4}\frac{|H|}{|U|^{n-|W|}} \text{ is less than } \tfrac{1}{4}|H|. \tag{6}$$

Given $W$, define a partition of $H$ according to $S^x$, i.e. $x$ and $y$ are in the same class if and only if $S^x = S^y$ (thus $y \in S^x$, and $x \in S^y$). Clearly there are no more than $|U|^{n-|W|}$ classes, as we have at most this many ways of choosing a $y$ which is different from $x$ outside $W$. The number of inputs in classes of size at most $\frac{1}{4}|H|/(|U|^{n-|W|})$ is at most this number (the maximum size of these classes) times $|U|^{n-|W|}$ (the maximum total number of classes), implying (6).

Based on a hard set $H_\mathcal{A}$, let $S_i^x$ be defined as in (5) based on $H_\mathcal{A}$ and $W_i$. Clearly (6) holds for $S_1^x$ and $S_2^x$, so these sets are relatively large for most inputs in $H_\mathcal{A}$. We would like an $x$ for which both $S_1^x$ and $S_2^x$ are large, but in principle the $x$'s that give large $S_1^x$ might not be the same as those that give large $S_2^x$ (and vice versa). According to (6), $|S_i^x| \geq \frac{1}{4}|H|/(|U|^{n-|W|})$ for $\frac{3}{4}$ of the elements in $H$, hence for $\frac{1}{4}$ of these elements both $S_1^x$ and $S_2^x$ are no smaller than $\frac{1}{4}|H|/(|U|^{n-|W|})$; certainly for any $H_\mathcal{A}$, $W_1$ and $W_2$, this gives us an $x$ such that both $S_1^x$ and $S_2^x$ has this size.

By Proposition 3 we would like to have $|S_i^x|$ strictly larger than $(\lambda(n)|U|)^{|W_i|}$, as this would give $x_{\text{fail}}$ such that by the fullness property

$D_\Delta(x_{\text{fail}})$ = "YES", but $\mathcal{A}(x_{\text{fail}}) = \mathcal{A}(x) = 0$, as $x \in H_\mathcal{A}$. We would like $|S_i^x| > (\lambda(n)|U|)^{|W_i|}$, which is the case if $\frac{1}{4}|H|/(|U|^{n-|W|}) > (\lambda(n)|U|)^{|W_i|}$. Recalling that $|W_i| \geq \mu(n)\cdot n$ we are done. □

**Lemma 2** *Let $\Delta$ be a $\zeta(n)-$sparse relation on $U$, $\mathcal{A}$ be a branching program deciding $D_\Delta$ in time $k(n)\cdot n$ and space $S(n)$, and let $\mu(n) = k(n)^{-4k(n)}$. $\mathcal{A}$ has a $\mu(n)-$hard set of size*

$$|H_\mathcal{A}| \geq \frac{\zeta(n)|U|^n}{\binom{n}{\mu(n)\cdot n}^2\ (3k(n))^{8k(n)}\ 2^{4k(n)S(n)}}.$$

**Proof:** We start by constructing a partitioning $P_x$ of the indices as follows. Split time into $9k^2(n)$ intervals of length[3] $n/(9k(n))$. Two indices $i$ and $j$ are in the same class of $P_x$ if and only if they are read in exactly the same intervals on input $x$. $P'_x$ is $P_x$ restricted to classes of $P_x$ whose members are queried in at most $2k(n)$ time intervals on $x$. Finally $\Gamma_x$ is $P'_x$ restricted to classes whose size is at least $n/(4|P'_x|)$.

Define $\texttt{intervals}(x, W)$ to be the intervals in which $W$ is queried on input $x$. By construction, $\texttt{intervals}(x, W)$ contains at most $2k(n)$ intervals for all $W \in \Gamma_x$.

Our restricted partitioning $\Gamma_x$ has the following properties,

$$\forall W \in \Gamma_x :\ |W| \geq \mu(n)\cdot n, \tag{7}$$

$$\forall x : \exists W_1, W_2 \in \Gamma_x :\ \texttt{intervals}(x, W_1) \cap \texttt{intervals}(x, W_2) = \emptyset. \tag{8}$$

Each class of $P'_x$ is uniquely identified by the corresponding set of at most $2k(n)$ intervals in which the indices of the class are read. Hence we may bound the number of classes in $P'_x$ by the number of ways to choose up to $2k(n)$ intervals from $9k^2(n)$,

$$\begin{aligned}|P'_x| &\leq \sum_{i=0}^{2k(n)} \binom{9k^2(n)}{i}\\ &= 2^{9k^2(n)H(\frac{2}{9k(n)})-\frac{1}{2}\lg 9k^2(n)+O(1)}\\ &\leq k(n)^{3k(n)},\end{aligned}$$

[3] The interval length is parameterized in [Ajt99b], however the present choice leaves little room for improvement.

according to [GKP94, p. 492][4], $H(m)$ is the entropy function $m \lg \frac{1}{m} + (1-m)\lg\frac{1}{1-m}$. By the lower bound on the size of the classes in $\Gamma_x$ we have proved (7).

It is a fact that no more $n/2$ elements can be queried in more than $2k(n)\cdot n$ intervals, as then we would have more than $n$ indices in total; hence $P'_x$ covers at least $n/2$ elements. Classes of $P'_x$ with size no greater than $n/(4|P'_x|)$ can cover at most $n/4$ indices (as we have most $|P'_x|$ such classes). Thus $\Gamma_x$ covers at least $\frac{n}{2} - \frac{n}{4} = \frac{n}{4}$ indices.

Define $\texttt{indices}(x, I)$ to be the indices queried in time interval $I$ on input $x$. We will now prove (8); in fact we will prove the stronger statement that

$$\forall W_1 \in \Gamma_x \, \exists W_2 \in \Gamma_x : \texttt{intervals}(x, W_1) \cap \texttt{intervals}(x, W_2) = \emptyset. \quad (9)$$

Suppose that this is not the case, namely that for some fixed $x$ there exists $W_1$ such that

$$\forall W_2 : \texttt{intervals}(x, W_1) \cap \texttt{intervals}(x, W_2) \neq \emptyset, \quad (10)$$

which implies that for *each* $W_2$ there exists some interval $I \in \texttt{intervals}(x, W_1)$ such that $W_2 \subseteq \texttt{indices}(x, I)$—all of $W_2$ is read in $I$ on $x$. This implies that

$$\bigcup_{W_2 \in \Gamma_x} W_2 \subseteq \bigcup_{I \in \texttt{intervals}(x, W_1)} \texttt{indices}(x, I).$$

[4] These estimates are not correct for $k(n) \in O(1)$, in which case we can just use Ajtai's original estimate. Also, using $\binom{n}{k} < (\frac{n}{e})^k$ gives a bound that is almost as good, but we use the above estimate to emphasize that significantly better estimates are not possible.

Of course $\cup_{W_2 \in \Gamma_x} W_2$ covers all of $\Gamma_x$, but its size is bounded by

$$\begin{aligned} |\Gamma_x| &\leq \left| \bigcup_{W_2 \in \Gamma_x} W_2 \right| \\ &\leq \left| \bigcup_{I \in \texttt{intervals}(x, W_1)} \texttt{indices}(x, I) \right| \\ &\leq \sum_{I \in \texttt{intervals}(x, W_1)} |\texttt{indices}(x, I)| \\ &\leq \sum_{I \in \texttt{intervals}(x, W_1)} \frac{n}{9k(n)} \\ &\leq 2k(n) \frac{n}{9k(n)} \\ &< \tfrac{1}{4} n, \end{aligned}$$

hence (10) is not true, as $|\Gamma_x| \geq \frac{1}{4}$, and we may conclude (9), which in turn imply (8).

To conclude the proof we will use $\Gamma_x$ to construct a hard set. For each $x \in U^n$ with $D_\Delta(x) =$ "NO" let $W_1$ and $W_2$ be the set of indices whose existence is promised in (8), and let $T_i = \texttt{intervals}(x, W_i)$. Define $H^x_{\mathcal{A}}$ to be the set of $y \in U^n$ that satisfy,

- $D_\Delta(x) =$ "NO".
- $T_i = \texttt{intervals}(y, W_i)$ $(= \texttt{intervals}(x, W_i))$, hence $W_i$ is a class of both $\Gamma_x$ and $\Gamma_y$.
- $\texttt{state}(x, T_i) = \texttt{state}(y, T_i)$.

Define $F_i$ to be a function that given $x$ and $T_i$ lists $\texttt{state}(x, I_{ij})$ where $T_i$ is comprised of $\cup_j I_{ij}$; $F_i$ gives an ordered lists of the states of $\mathcal{A}$ each time we leave $T_i$.

The above is a hard set as (7) means that $|W_i| \geq \mu(n) \cdot n$ as required, and the indices of $W_i$ are certainly not read in $T_j$, in fact they are *only* read in $T_i$.

Each hard set $H^x_{\mathcal{A}}$ is uniquely determined by the six-tuple $\langle W_1, W_2, T_1, T_2, F_1, F_2 \rangle$. If we can choose this six-tuple in at most $h(n)$ ways, there must be an $x$ such that $|H^x_{\mathcal{A}}| \geq \frac{\zeta(n)|U|^n}{h(n)}$ as there are at least $\zeta(n)|U|^n$ inputs $x$ with $D_\Delta(x) =$ "NO".

Observe that $W_i$ need not be bigger than $\mu(n) \cdot n$; if we have more indices than this, just take the first $\mu(n) \cdot n$. This may collapse a number of classes into one, meaning that we may choose each class $W_i$ in $\binom{n}{\mu(n) \cdot n}$ ways. This restriction on the size of $W_i$ significantly increases the size of the interval in which we can obtain a bound.

As $T_i$ consists of at most $2k(n)$ out of $9k(n)$ intervals, the number of ways to choose $T_i$ is bounded by $(9k(n))^{2k(n)} = (3k(n))^{4k(n)}$ (actually a better estimate should be possible, but it will not improve the result significantly). Finally, as $T_i$ consists of at most $2k(n)$ intervals we fix the state of our computation in at most this many places, each with $2^{S(n)}$ choices, meaning that $F_i$ can be chosen in at most $(2^{S(n)})^{2k(n)}$ ways. In total we get that

$$\begin{aligned} h(n) &\leq \binom{n}{\mu(n)\cdot n}^2 \cdot \left((3k(n))^{4k(n)}\right)^2 \cdot \left((2^{S(n)})^{2k(n)}\right)^2 \\ &= (\binom{n}{\mu(n)\cdot n}^2 (3k(n))^{8k(n)}\ 2^{4k(n)S(n)}. \end{aligned}$$

□

**Proof of Theorem 2:** Let $\Delta$ be a $\lambda(n)$–full and $\zeta(n)$–sparse relation on $U$. Assume that $\mathcal{A}$ is an $R$-way branching program deciding $D_\Delta$ on $U^n$ in time $k(n)\cdot n$ and space $S(n)$, such that (2) and (3) but not (4) holds. Based on these assumptions our aim is to arrive at a contradiction, thus proving the theorem. Specifically we will show that there exists an input $x_{\text{fail}}$ such that $D_\Delta(x_{\text{fail}}) =$ "YES", but the branching program answers $\mathcal{A}(x_{\text{fail}}) =$ "NO".

Combining Lemma 1 and Lemma 2 we see that we can find $x_{\text{fail}}$ if

$$\frac{\zeta(n)|U|^n}{\binom{n}{\mu(n)\cdot n}^2\ (3k(n))^{8k(n)}\ 2^{4k(n)S(n)}} > 4\lambda(n)^{\mu(n)\cdot n}|U|^n.$$

Using Stirling's approximation for $n!$ (see eg. [Knu97, p. 115]) we get that

$$\lg \binom{n}{\mu(n)n} < 3\mu(n)n \lg \tfrac{1}{\mu(n)}$$

for $n$ sufficiently large, it is sufficient that

$$2 + \lg \tfrac{1}{\zeta(n)} + 6\mu(n)n \lg \tfrac{1}{\mu(n)} + 8k(n) \lg 3k(n) + 4k(n)S(n) < \mu(n)n \lg \tfrac{1}{\lambda(n)}.$$

If $k(n) < n$ (which it will be) then $8k(n)\lg 3k(n) + 4k(n)S(n) < 12k(n)S(n)$ as $S(n) > \lg n$ (necessarily). Likewise the constant 2 is dominated by $k(n)S(n)$. Hence, the above is satisfied if

$$\lg \tfrac{1}{\zeta(n)} + 6\mu(n)n \lg \tfrac{1}{\mu(n)} + 13k(n)S(n) < \mu(n)n \lg \tfrac{1}{\lambda(n)}.$$

This certainly holds if each of the three terms on the left hand side is less than a third of the term on the right hand side. We have the desired $x_{\text{fail}}$ if

$$\lg \tfrac{1}{\zeta(n)} < \tfrac{1}{3}\mu(n)n \lg \tfrac{1}{\lambda(n)},$$

$$6\mu(n)n \lg \tfrac{1}{\mu(n)} < \tfrac{1}{3}\mu(n)n \lg \tfrac{1}{\lambda(n)},$$

$$13k(n)S(n) < \tfrac{1}{3}\mu(n)n \lg \tfrac{1}{\lambda(n)}.$$

Which is implied by (2), (3) and ¬(4), since $\mu(n) = k(n)^{-4k(n)}$. □

# 6 Randomization

**Definition 5 (Randomized $R$-way branching program)** *A* randomized $R$-way branching program *using $r$ random bits, consists of a collection of $2^r$ deterministic $R$-way branching programs. Each execution of the randomized branching program starts by uniformly at random choosing one of the $2^r$ deterministic programs which is then executed.*

*We say that a randomized $R$-way branching program $\mathcal{A}$ deciding a problem $D$ has constant 1-sided error if for $0 \leq \epsilon < \frac{1}{2}$ the following holds*

- *If $D(x) = 1$ then our randomized branching program $A$ answers correctly on input $x$.*
- *If $D(x) = 0$ then* $\Pr[\mathcal{A}(x) = 0] > 1-\epsilon$.

□

**Corollary 6** *The statement of Theorem 2 holds for randomized $R$-way branching programs with constant 1-sided error if we modify the constants slightly.*

**Proof:** By a standard averaging argument, one of the $2^r$ deterministic branching programs must correctly answer "NO" for at least a $1-\epsilon$ fraction of the inputs with answer "NO". Apply Theorem 2 to this *deterministic* branching program computing $D_\Delta$. Hence we only reduce the size of our hard set with a factor $\epsilon$. □

# 7 Acknowledgement

I would like to thank Faith Fich.

# References

[Ajt99a] Miklós Ajtai, *A Non-linear Time Lower Bound for Boolean Branching Programs*, 40th Annual Symposium on Foundations of Computer Science, IEEE, 1999.

[Ajt99b] ———, *Determinism versus Non-Determinism for Linear Time RAMs with Memory Restrictions*, Thirty-First ACM Symposium on Theory of Computing, ACM, 1999.

[BC82] Allan Borodin and Stephen Cook, *A Time-Space Tradeoff for Sorting on a General Sequential Model of Computation*, SIAM Journal on Computing **11** (1982), no. 2, 287–297.

[Bea91] Paul Beame, *A General Sequential Time-Space Tradeoff for Finding Unique Elements*, SIAM Journal on Computing **20** (1991), 270–277.

[BFK$^+$81] Allan Borodin, Michael J. Fischer, David G. Kirkpatrick, Nancy A. Lynch, and Martin Tompa, *A Time-Space Tradeoff for Sorting on Non-Oblivious Machines*, Journal of Computer and System Sciences **22** (1981), 351–364.

[BFMadH$^+$87] Allan Borodin, Faith E. Fich, Friedhelm Meyer auf der Heide, Eli Upfal, and Avi Wigderson, *A Time-Space Tradeoff for Element Distinctness*, SIAM Journal on Computing **16** (1987), 97–99.

[BST98] Paul Beame, Michael Saks, and Jayram S. Thathachar, *Time-Space Tradeoffs for Branching Programs*, 39th Annual Symposium on Foundations of Computer Science, IEEE, 1998.

[GKP94] Ronald Lewis Graham, Donald Erwin Knuth, and Oren Patashnik, *Concrete mathematics*, 2nd ed., Addison-Wesley, 1994.

[Kar86] Mauricio Karchmer, *Two Time-Space Tradeoffs for Element Distinctness*, Theoretical Computer Science **47** (1986), 237–246.

[Knu97] Donald Ervin Knuth, *Fundamental Algorithms*, 3rd ed., The Art of Computer Programming, vol. 2, Addison-Wesley, 1997.

[Knu98] ———, *Sorting and Searching*, 2nd ed., The Art of Computer Programming, vol. 3, Addison-Wesley, 1998.

[MNT93] Yishay Mansour, Noam Nisan, and Prasoon Tiwari, *The Computational Complexity of Universal Hashing*, Theoretical Computer Science (1993), no. 107, 121–133.

[Sav98] John Edmund Savage, *Models of Computation*, Addison-Wesley, 1998.

[Yao94] Andrew Chi-Chih Yao, *Near-optimal Time-Space Tradeoff for Element Distinctness*, SIAM Journal on Computing **23** (1994), 966–975.

# Recent BRICS Report Series Publications

RS-00-11 Jakob Pagter. *On Ajtai's Lower Bound Technique for $R$-way Branching Programs and the Hamming Distance Problem*. May 2000. 18 pp.

RS-00-10 Stefan Dantchev and Søren Riis. *A Tough Nut for Tree Resolution*. May 2000. 13 pp.

RS-00-9 Ulrich Kohlenbach. *Effective Uniform Bounds on the Krasnoselski-Mann Iteration*. May 2000. 34 pp.

RS-00-8 Nabil H. Mustafa and Aleksandar Pekeč. *Democratic Consensus and the Local Majority Rule*. May 2000. 38 pp.

RS-00-7 Lars Arge and Jakob Pagter. *I/O-Space Trade-Offs*. April 2000. To appear in *7th Scandinavian Workshop on Algorithm Theory*, SWAT '98 Proceedings, LNCS, 2000.

RS-00-6 Ivan B. Damgård and Jesper Buus Nielsen. *Improved Non-Committing Encryption Schemes based on a General Complexity Assumption*. March 2000. 24 pp.

RS-00-5 Ivan B. Damgård and Mads J. Jurik. *Efficient Protocols based on Probabilistic Encryption using Composite Degree Residue Classes*. March 2000. 19 pp.

RS-00-4 Rasmus Pagh. *A New Trade-off for Deterministic Dictionaries*. February 2000.

RS-00-3 Fredrik Larsson, Paul Pettersson, and Wang Yi. *On Memory-Block Traversal Problems in Model Checking Timed Systems*. January 2000. 15 pp. Appears in Graf and Schwartzbach, editors, *Tools and Algorithms for The Construction and Analysis of Systems: 6th International Conference*, TACAS '00 Proceedings, LNCS 1785, 2000, pages 127–141.

RS-00-2 Igor Walukiewicz. *Local Logics for Traces*. January 2000. 30 pp.

RS-00-1 Rune B. Lyngsø and Christian N. S. Pedersen. *Pseudoknots in RNA Secondary Structures*. January 2000. 15 pp. To appear in *Fourth Annual International Conference on Computational Molecular Biology*, RECOMB '00 Proceedings, 2000.

RS-99-57 Peter D. Mosses. *A Modular SOS for ML Concurrency Primitives*. December 1999. 22 pp.