Regular Resolution Lower Bounds for the Weak Pigeonhole Principle

Toniann Pitassi*
Department of Computer Science
University of Toronto
toni@cs.toronto.edu

Ran Raz†
Department of Computer Science
Weizmann Institute
ranraz@wisdom.weizmann.ac.il

Abstract

We prove that any regular resolution proof for the weak pigeon hole principle, with n holes and any number of pigeons, is of length $\Omega(2^{n^{\epsilon}})$, (for some global constant $\epsilon > 0$).

1 Introduction

In the last three decades, a tremendous amount of research has been directed towards understanding the mathematical structure of the satisfiability problem, and towards developing algorithms for satisfiability testing and propositional theorem proving. Much of this research has centered around the method of *resolution*.

The *resolution principle* says that if C and D are two clauses and x_i is a variable then any assignment that satisfies both of the clauses, $C \vee x_i$ and $D \vee \neg x_i$, also satisfies $C \vee D$. The clause $C \vee D$ is called the *resolvent* of the clauses $C \vee x_i$ and $D \vee \neg x_i$ on the variable x_i. A *resolution refutation* for a CNF formula F is a sequence of clauses $C_1, C_2, \ldots, C_s$, such that: (1) Each clause C_j is either a clause of F or a resolvent of two previous clauses in the sequence. (2) The last clause, C_s, is the empty clause (and hence it has no satisfying assignments). We can represent a resolution refutation as an acyclic directed graph on vertices $C_1, \ldots, C_s$, where each clause of F has out-degree 0, and any other clause has two edges pointing to the two

*Research supported by NSF grant CCR-9820831, US-Israel BSF grant 98-00349, and an NSERC grant.
†Research supported by US-Israel BSF grant 98-00349, and NSF grant CCR-9987077.

clauses that were used to produce it. It is well known that resolution is a sound and complete propositional proof system, i.e., a formula F is unsatisfiable if and only if there exists a resolution refutation for F. We think of a refutation for an unsatisfiable formula F also as a proof for the tautology $\neg F$. Resolution is the most widely studied approach to propositional theorem proving, and there is a large body of research exploring resolution algorithms, i.e., algorithms that on input an unsatisfiable formula F, output a resolution refutation for F. A well-known and widely studied restricted version of resolution (that is still complete) is called *regular resolution*. In a regular resolution refutation, along any path in the directed acyclic graph, each variable is resolved upon at most once.

One of the main directions of research in propositional proof theory is proving lower bounds for the length of proofs for different tautologies in different proof systems [CR]. One of the most widely studied tautologies in this context is the *Pigeon Hole Principle* (PHP). The tautology PHP_n is a DNF encoding of the following statement: There is no one to one mapping from $n+1$ pigeons to n holes. The *Weak Pigeon Hole Principle* (WPHP) is a version of the pigeon hole principle that allows a larger number of pigeons. The tautology $WPHP_n^m$ (for $m \geq n+1$) is a DNF encoding of the following statement: There is no one to one mapping from m pigeons to n holes. For $m > n+1$, the weak pigeon hole principle is a weaker statement than the pigeon hole principle. Hence, it may have much shorter proofs in certain proof systems.

The weak pigeon hole principle is a fundamental combinatorial principle that underlies the induction principle and is used in many mathematical proofs. In particular, it is used in most probabilistic counting arguments and hence in many combinatorial proofs. In addition, much of elementary number theory (including the existence of infinitely many primes) can be formalized in weak systems with the weak pigeon hole principle [PWW].

Moreover, as observed by Razborov, there are certain connections between the weak pigeon hole principle and the problem of $P \neq NP$ [Razb1]. Indeed, the weak pigeon hole principle (with relatively large number of pigeons m) can be interpreted as an encoding of the following statement: There are no small DNF formulas for SAT. Hence, in most proof systems, a short proof for a certain formulation of the statement $NP \not\subset P/poly$ can be translated into a short proof for the weak pigeon hole principle. That is, a lower bound for the length of proofs for the weak pigeon hole principle implies a lower bound for the length of proofs for a certain formulation of the statement $NP \not\subset P/poly$. While we do not regard our result as a step towards understanding the problem of $P \neq NP$, we do believe that the above connection demonstrates the usefulness and the generality of the weak pigeon hole principle.

There are trivial resolution proofs (and regular resolution proofs) of length $2^n \cdot poly(n)$ for the pigeon hole principle and for the weak pigeon hole principle. In a seminal paper, Haken proved that for the pigeon hole principle, the trivial proof is (almost) the best possible [Hak]. More specifically, Haken proved that any resolution proof (or regular resolution proof) for the tautology PHP_n is of length $2^{\Omega(n)}$. Haken's argument was further developed in several other papers (e.g., [Urq, BeP1, BSW]). It was shown that a similar argument gives lower bounds also for the weak pigeon hole principle, for small values of m. More specifically, super-polynomial lower bounds were proved for any resolution proof (or regular resolution proof) for the tautology $WPHP_n^m$, for $m < c \cdot n^2/\log n$ (for some constant c) [BT].

For the weak pigeon hole principle with large values of m, there do exist resolution proofs which are much shorter than the trivial ones. In particular, it was proved by Buss and Pitassi that for $m > c^{\sqrt{n}\log n}$ (for some constant c), there are resolution proofs of length $poly(m)$ for the tautology $WPHP_n^m$ [BuP]. Can these upper bounds be further improved ? It was widely believed that for any m, any resolution proof (or regular resolution proof) for the tautology $WPHP_n^m$ is of length exponential in n. However, for $m \geq n^2$, no non-trivial lower bound was previously known. The only partial progress was made by Razborov, Wigderson and Yao, who proved exponential lower bounds for regular resolution proofs, but only when the (regular resolution) proof is of a certain restricted form [RWY].

In this paper, we prove that for any m, any regular resolution proof for the weak pigeon hole principle $WPHP_n^m$ is of length $\Omega(2^{n^\epsilon})$, (where $\epsilon > 0$ is some global constant).

1.1 Subsequent work

As mentioned above, our main result is a lower bound of $\Omega(2^{n^\epsilon})$ for any Regular Resolution proof for the weak pigeonhole principle. Following our result, a lower bound of $\Omega(2^{n^\epsilon})$ for any Resolution proof for the weak pigeonhole principle was proved in [Raz]. The bound was further improved, simplified and generalized in [Razb2, Razb3, Razb4].

For a recent survey on the propositional proof complexity of the pigeonhole principle (including the above mentioned subsequent work), see [Razb5]. For a recent survey on the main research directions in propositional proof theory, see [BeP2].

2 Preliminaries

A *literal* is either an atom (i.e., a variable x_i) or the negation of an atom. A *clause* is a disjunction of literals. As mentioned in the introduction, a *resolution refutation* of a CNF formula F is a sequence of clauses, such that, each clause is either a clause of F or is derived from two previous clauses by the resolution rule, and such that, the final clause is the empty clause. We think of a resolution refutation for F also as a proof for $\neg F$. The *length* of a resolution proof is the number of clauses in it. A resolution proof is *regular* if (in the corresponding graph) along any path from an initial clause (i.e., a clause of F) to the final clause, every atom is resolved upon at most once.

To prove our lower bound, we will exploit the equivalence between regular resolution proofs and read-once branching programs [CS, LNNW] (see also [Kra]). For any unsatisfiable formula $F = C_1(x_1, \ldots, x_l) \wedge \ldots \wedge C_k(x_1, \ldots, x_l)$, let us consider the following search problem S_F: Given a truth assignment $a \in \{0,1\}^l$, find v such that $C_v(a) = 0$.

A *Boolean branching program* in l variables is a directed acyclic graph with nodes of out-degrees 0 or 2, such that: (1) There is only one source node (i.e., only one node of in-degree 0), called, the root. (2) Every non-sink node (i.e., every node of out-degree 2) is labeled by one of the variables $x_1, \ldots, x_l$, and the two out-going edges are labeled 0 and 1 respectively. If a node v is labeled by x_i we say that x_i is *queried* at v. Each input string $a \in \{0,1\}^l$ determines a path from the root to a sink node of the branching program. A Boolean branching program solves a search problem S (with inputs in $\{0,1\}^l$) if for every $a \in \{0,1\}^l$, the path determined by a leads to a sink labeled by a valid solution for the search problem S on the input a. The *size* of a branching program is the number of nodes in it. A branching program is *read-once* if along any path on it, each variable is queried at most once.

A read-once branching program is *uniform* if the following holds: (1) For a path p beginning at the root, the set of variables queried along p depends only on the terminal node of p. (2) For any path p beginning at the root and terminating at a sink node, the set of variables queried along p is the set of all variables. It is not hard to see that any read-once branching program can be made uniform with little increase in its size:

Proposition 2.1 *Any branching program in l variables can be simulated by a uniform branching program whose size is larger by at most a factor of l.*

The following Lemma, due to Krajíček , gives the connection between branching programs and regular resolution.

Lemma 2.1 *Let F be an unsatisfiable CNF formula. Then, the minimum length of a regular resolution refutation for F is equal to the minimum size of a read-once branching program solving the search problem S_F.*

The propositional weak pigeon hole principle, $WPHP_n^m$, states that there is no one-to-one mapping from m pigeons to n holes. The underlying Boolean variables, $x_{i,j}$, for $1 \le i \le m$ and $1 \le j \le n$, represent whether or not pigeon i is mapped to hole j. The negation of the pigeonhole principle, $\neg WPHP_n^m$, is expressed in conjunctive normal form (CNF) as the conjunction of *m pigeon clauses* and $\binom{m}{2} \cdot n$ *hole clauses*. For every $1 \le i \le m$, we have a pigeon clauses, $(x_{i,1} \vee \ldots \vee x_{i,n})$, stating that pigeon i maps to some hole. For every $1 \le i_1 < i_2 \le m$ and every $1 \le j \le n$, we have a hole clauses, $(\neg x_{i_1,j} \vee \neg x_{i_2,j})$, stating that pigeons i_1 and i_2 are not both map to hole j. We refer to the pigeon clauses and the hole clauses also as pigeon axioms and hole axioms.

A branching program for the weak pigeon hole principle queries the variables $x_{i,j}$ and finds either a pigeon clause that is not satisfied or a hole clause that is not satisfied.

3 The Lower Bound for Branching Programs

3.1 Basic notations

In this section we prove our lower bound on the size of read-once branching programs for the weak pigeon hole principle. We denote by n the number of holes, and by m the number of pigeons. We denote by $Holes$ the set of holes, and by $Pigeons$ the set of pigeons. That is,

Holes $= \{1, ..., n\}$.

Pigeons $= \{1, ..., m\}$.

We will usually denote a hole by j, and a pigeon by i. By $x_{i,j}$ we denote the variable corresponding to pigeon i and hole j. By $Hole_j$ we denote the set of variables corresponding to the j^{th} hole, and by $Pigeon_i$ the set of variables corresponding to the i^{th} pigeon. That is,

$\mathbf{Hole_j} = \{x_{i,j} | i \in Pigeons\}$.

$\mathbf{Pigeon_i} = \{x_{i,j} | j \in Holes\}$.

We will consider read once branching programs that query the variables $x_{i,j}$. By u we will usually denote a node in the branching program (BP). We say that $u' < u$ if there is a path in the BP from u' to u. By p we will usually denote a path in the BP. We denote by ϵ a small fixed constant (say $\epsilon = 1/20$). For simplicity, we assume that n is large enough (say $n \geq 10^{100}$). We will assume that our BP is of size $< 2^{n^\epsilon/10}$, and we will show that such a BP cannot solve the weak pigeon hole principle (we do not attempt here to optimize the value of ϵ). Note that since the size of the BP is lower than $2^{n^\epsilon/10}$, we can assume w.l.o.g. that $m < 2^{n^\epsilon/10}$ as well (as we can just ignore pigeons that were not mentioned in the branching program). For simplicity, we assume that expressions like $n^\epsilon, n^{1-\epsilon}/2$, etc., are all integers.

By Proposition 2.1, we can assume that the set of variables queried along each path to a node u is the same (i.e., the set of queries is independent of the path and depends only on the final node u). For a non-leaf node u define:

Label$(\mathbf{u})$ = the variable $x_{i,j}$ queried at u.

For any node u, define:

Queries$(\mathbf{u})$ = the set of variables $x_{i,j}$ queried along paths to u (not including the variable $Label(u)$). As mentioned above, by Proposition 2.1, this set is independent of the path taken to u.

$\mathbf{Queries_i}(\mathbf{u}) = Queries(u) \cap Pigeon_i$.

For any node u, define $Ones(u)$ to be the set of variables that the node u "remembers" to be 1, and $Zeros(u)$ to be the set of variables that the node u "remembers" to be 0, that is:

Ones$(\mathbf{u})$ = the set of variables $x_{i,j}$ in $Queries(u)$ that get the value 1 along every path to u.

Zeros$(\mathbf{u})$ = the set of variables $x_{i,j}$ in $Queries(u)$ that get the value 0 along every path to u.

We can now define for any node u,

OpenPigeons(u) = the set of pigeons i with $Queries_i(u) \subset Zeros(u)$.

(Intuitively, $OpenPigeons(u)$ is the set of pigeons that were not matched to a hole along any path to u. That is, the set of pigeons that their corresponding pigeon clauses were still not satisfied and hence they may still be output by the branching program as an answer).

Claim 3.1 *If $u' \leq u$ then*

$$OpenPigeons(u) \subseteq OpenPigeons(u').$$

Proof:
Let $i \in OpenPigeons(u)$. For any $x_{i,j} \in Queries_i(u')$,

$$x_{i,j} \in Queries_i(u') \subset Queries_i(u) \subset Zeros(u).$$

Hence, $x_{i,j}$ gets the value 0 along every path to u. Since $u' < u$ and $x_{i,j} \in Queries_i(u')$, $x_{i,j}$ must get the value 0 along every path to u'. Hence, $x_{i,j} \in Zeros(u')$. Since this is true for any $x_{i,j} \in Queries_i(u')$, we conclude that $Queries_i(u') \subset Zeros(u')$, and hence, $i \in OpenPigeons(u')$. □

3.2 Types of axioms and types of holes

For integer $1 \leq k \leq n^\epsilon$, define

$$n_k = k \cdot n^{1-\epsilon},$$

and

$$m_k = 2^{n^\epsilon - k}$$

(recall that we assume that $n^\epsilon, n^{1-\epsilon}$ are integers). We will say that a node u is a pigeon-axiom of order k if there is a set $A \subset OpenPigeons(u)$, such that $|A| = m_k$ and for every $i \in A$, $|Queries_i(u)| \geq n_k$. Note that a pigeon-axiom of order $k = n^\epsilon$ corresponds to a standard pigeon-axiom of the pigeon hole principle. We say that a node u is a hole-axiom if there exists a hole j and two different pigeons i_1, i_2, such that $x_{i_1,j}, x_{i_2,j} \in Ones(u)$. Note that this corresponds to a standard hole-axiom of the pigeon hole principle. In our proof, we allow our BP to find a pigeon-axiom of any order k (and not only pigeon-axioms of order $k = n^\epsilon$ as in the usual weak pigeon hole principle). That is, we say that the BP solves the weak pigeon hole principle if it always stops at an axiom (i.e., all leaves of the BP are either hole-axioms or pigeon-axioms of some order). We assume w.l.o.g. that in our BP no non-leaf node is an axiom (otherwise, the BP can just stop at that node). In particular, no non-leaf node is a pigeon-axiom of order k, for any k.

Let $S_1 \cup \cdots \cup S_{n^\epsilon}$ be a random partition of *Holes* into n^ϵ sets of size $n^{1-\epsilon}$ each. The intuition is that the set of holes S_k will be used against pigeon-axioms of order k. (More precisely, for every S_k, the adversary strategy below will be designed in order to make it hard for the branching program to find pigeon-axioms of order k that are not satisfied). For each $1 \le k \le n^{\epsilon}$, let $S_{k,1} \cup \cdots \cup S_{k,n^{\epsilon}}$ be a random partition of S_k into n^{ϵ} sets of size $n^{1-2\epsilon}$ each.

Let I be the segment of integers $[1, ..., n^{1-2\epsilon}]$, We think of I as a set of indices. Let $I_1, ..., I_{3n^{\epsilon}}$ be the partition of I into $3n^{\epsilon}$ segments of length $n^{1-2\epsilon}/3n^{\epsilon} = (1/3) \cdot n^{1-3\epsilon}$ each. That is,

$$I_r = \left[(1/3) \cdot n^{1-3\epsilon} \cdot (r-1) + 1, ..., (1/3) \cdot n^{1-3\epsilon} \cdot r\right].$$

For each $1 \le r \le 3n^{\epsilon}$, let $I_{r,1}, ..., I_{r,3n^{\epsilon}}$ be the partition of I_r into $3n^{\epsilon}$ segments of length $(1/3) \cdot n^{1-3\epsilon}/3n^{\epsilon} = (1/9) \cdot n^{1-4\epsilon}$ each. That is,

$$I_{r,s} = [b_{r,s}, ..., e_{r,s}],$$

where

$$b_{r,s} = (1/3) \cdot n^{1-3\epsilon} \cdot (r-1) + (1/9) \cdot n^{1-4\epsilon} \cdot (s-1) + 1,$$

and

$$e_{r,s} = (1/3) \cdot n^{1-3\epsilon} \cdot (r-1) + (1/9) \cdot n^{1-4\epsilon} \cdot s.$$

Denote

$$\hat{n} = |I_{r,s}| = (1/9) \cdot n^{1-4\epsilon}.$$

The relevant set of indices for S_k will be I_{3k-1}. The relevant set of indices for $S_{k,l}$ will be $I_{3k-1,3l-1}$. Define:

$\mathbf{Queries_i^{k,l}(u)}$ = the set of variables $x_{i,j} \in Queries_i(u)$ such that $j \in S_{k,l}$.

$\mathbf{Stage_i^{k,l}}$ = the set of all nodes u such that $|Queries_i^{k,l}(u)| + 1 \in I_{3k-1,3l-1}$ (that is, $b_{3k-1,3l-1} \le |Queries_i^{k,l}(u)| + 1 \le e_{3k-1,3l-1}$).

$\mathbf{Over_i^{k,l}}$ = the set of all nodes u such that $|Queries_i^{k,l}(u)| + 1 > e_{3k-1,3l-1}$.

$\mathbf{Before_i^{k,l}}$ = the set of all nodes u such that $|Queries_i^{k,l}(u)| + 1 < b_{3k-1,3l-1}$.

(We will show below that for every u, i there is at most one pair k, l such that $u \in Stage_i^{k,l}$. Intuitively, k, l measure the size of $Queries_i(u)$ and the adversary strategy below will be defined differently for nodes u in different stages).

Claim 3.2 *With exponentially high probability (over the choice of the partition $\{S_{k,l}\}$), the following is satisfied for every k, l, i, u:*

1.

$$n^{-2\epsilon} \cdot |Queries_i(u)| < |Queries_i^{k,l}(u)| + \hat{n}/2.$$

2.

$$n^{-2\epsilon} \cdot |Queries_i(u)| > |Queries_i^{k,l}(u)| - \hat{n}/2.$$

Proof:
Denote,

$$Q_{i,u} = n^{-2\epsilon} \cdot |Queries_i(u)|$$

and

$$Q_{i,u}^{k,l} = |Queries_i^{k,l}(u)|.$$

Denote,

$$D_{i,u}^{k,l} = Q_{i,u}^{k,l} - Q_{i,u}.$$

Recall that $Queries_i^{k,l}(u)$ is the set of variables $x_{i,j} \in Queries_i(u)$ such that $j \in S_{k,l}$. Since $S_{k,l}$ is a random subset of $\{1, ..., n\}$ of fraction exactly $n^{-2\epsilon}$, the expectation of the random variable $Q_{i,u}^{k,l}$ is $Q_{i,u}$, and we can use the standard Chernoff-Hoeffding bounds to bound $D_{i,u}^{k,l}$ (for any $\alpha > 0$) by

$$\mathbf{Pr}[|D_{i,u}^{k,l}| \geq \alpha n] \leq 2 \cdot e^{-2\alpha^2 n}.$$

Hence,

$$\mathbf{Pr}[|D_{i,u}^{k,l}| \geq \hat{n}/2] \leq 2 \cdot e^{-2n^{1-8\epsilon}/18^2}.$$

The number of tuples (k, l, i, u) is at most

$$n^\epsilon \cdot n^\epsilon \cdot 2^{n^\epsilon/10} \cdot 2^{n^\epsilon/10}.$$

Since this number is exponentially smaller than

$$\left(2 \cdot e^{-2n^{1-8\epsilon}/18^2}\right)^{-1},$$

we conclude that with exponentially high probability for every k, l, i, u,

$$|D_{i,u}^{k,l}| < \hat{n}/2.$$

$\square$

Claim 3.3 *With exponentially high probability (over the choice of the partition $\{S_{k,l}\}$), the following is satisfied for every k, l, k', l', i, u:*

1. $$|Queries_i^{k,l}(u)| < |Queries_i^{k',l'}(u)| + \hat{n}.$$

2. $$|Queries_i^{k,l}(u)| > |Queries_i^{k',l'}(u)| - \hat{n}.$$

Proof:
Immediate from Claim 3.2. $\square$

We say that $(k', l') < (k, l)$ if either $(k' < k)$ or $(k' = k$ and $l' < l)$.

Claim 3.4 *With exponentially high probability (over the choice of the partition $\{S_{k,l}\}$), the following is satisfied for every i, u and every $(k', l') < (k, l)$:*

1. $$u \in Stage_i^{k,l} \Longrightarrow u \in Over_i^{k',l'}.$$

2. $$u \in Stage_i^{k',l'} \Longrightarrow u \in Before_i^{k,l}.$$

Proof:
Immediate from Claim 3.3 and the definitions of $Stage_i^{k,l}$, $Before_i^{k,l}$, $Over_i^{k,l}$. □

Assume from now on that the partition $\{S_{k,l}\}$ is some fixed partition satisfying Claim 3.2, Claim 3.3 and Claim 3.4.

We say that a variable $x_{i,j}$ is of **type** $(\mathbf{k}, \mathbf{l})$ if $j \in S_{k,l}$. We say that a non-leaf node u is of **type** $(\mathbf{k}, \mathbf{l})$ if $Label(u)$ is of type (k, l).

3.3 The adversary strategy

We will define a probabilistic adversary strategy to answer queries of the BP. The answer given by the strategy depends not only on the node u but also on the path p that was taken to the node u (this is done for simplicity). In all the following, p is a path from the root to the node u (in some cases p will be a path starting from the root and u any node on it, and then we just think of p as a path to u and ignore the rest of it). We will define the strategy by defining it separately for each set $S_{k,l}$. The strategy for $S_{k,l}$ will answer queries $x_{i,j}$ of type (k, l). The answer $Strategy(p, u)$ for a node u of type (k, l) will be defined by induction on l, together with a set of pigeons $Stubborn^{k,l}(p, u)$. (Intuitively, the definition of the strategy for a node u of type (k, l) is designed in order to make it hard for the branching program to find pigeon axioms of order k that are not satisfied. This is done in n^ϵ steps. Intuitively, the set $Stubborn^{k,l}(p, u)$ is the set of pigeons that "survived" the first l steps). Thus, the definition of the strategy is actually done in $n^\epsilon \cdot n^\epsilon$ steps. Note that in step (k, l) of the definition, $Strategy(p, u)$ will be defined only for nodes u with $Label(u) = x_{i,j}$ such that $j \in S_{k,l}$. In contrast, the set $Stubborn^{k,l}(p, u)$ will be defined for any node u. First define for all k, p, u, the set
$$Stubborn^{k,0}(p, u) = Pigeons.$$
We will now assume that $Stubborn^{k,l-1}(p, u)$ is already defined and we will define the answer $Strategy(p, u)$ for nodes u of type (k, l). Let u be a node in the BP, with $Label(u) = x_{i,j}$ of type (k, l). Let p be a path from the root to u.

We say that u (with $Label(u) = x_{i,j}$ of type (k, l)) is **Relevant** for the path p (from the root to u) if all the following are satisfied:

1. $i \in OpenPigeons(u)$.
2. $i \in Stubborn^{k,l-1}(p,u)$.
3. $u \in Stage_i^{k,l}$.

If u is *Relevant* for p and there exists i' such that $x_{i',j} \in Ones(u)$ we say that for the path p, $x_{i,j}$ is **Forced** to 0 at u by $x_{i',j}$.

We now define $Strategy(p,u)$ in the following way:

Strategy$(\mathbf{p}, \mathbf{u})$, (for u with $Label(u) = x_{i,j}$ of type (k,l)):

- If u is not *Relevant* for p answer 0.
- If for the path p, $x_{i,j}$ is *Forced* to 0 at u answer 0.
- Otherwise, answer 1 with probability $1/2m_k$ and 0 with probability $1 - (1/2m_k)$.

The set of pigeons $Stubborn^{k,l}(p,u)$ is now defined (for any node u) in the following way:

$\mathbf{Stubborn^{k,l}(p,u)}$ = the set of pigeons i, such that all the following are satisfied:

1. $i \in Stubborn^{k,l-1}(p,u)$.
2. $u \in Over_i^{k,l}$.
3. At least $0.9 \cdot \hat{n}$ variables $x_{i,j}$ of type (k,l) were *Forced* to 0 at some node of the path p (up to the node u).

Claim 3.5 *Let p be a path from the root to u and let $u' < u$ be another node on the path p. Then for every k, l,*

$$Stubborn^{k,l}(p,u') \subseteq Stubborn^{k,l}(p,u).$$

Proof:
The proof is by induction on l. The base case, $l = 0$, follows by the definition of $Stubborn^{k,0}(p,u)$. For $l > 0$, let $i \in Stubborn^{k,l}(p,u')$. We will show that $i \in Stubborn^{k,l}(p,u)$.

1. By the inductive assumption, $i \in Stubborn^{k,l-1}(p,u')$ implies $i \in Stubborn^{k,l-1}(p,u)$.

2. If $u' \in Over_i^{k,l}$ and $u' < u$ then obviously $u \in Over_i^{k,l}$ (by the definition of $Over_i^{k,l}$).

3. Since $u' < u$, if at least $0.9 \cdot \hat{n}$ variables $x_{i,j}$ of type (k,l) were *Forced* to 0 at some node of the path p up to u' then all these variables were *Forced* to 0 at some node of the path p up to u.

Hence, $i \in Stubborn^{k,l}(p,u)$. $\square$

Claim 3.6 *Let p be a path from the root to u and let $u' < u$ be another node on the path p. Let k, l, i be such that $u, u' \in Over_i^{k,l}$ and $i \in Stubborn^{k,l}(p, u)$. Then*

$$i \in Stubborn^{k,l}(p, u').$$

Proof:
The proof is by induction on l. The base case, $l = 0$, follows by the definition of $Stubborn^{k,0}(p, u')$. For $l > 0$, let us show that $i \in Stubborn^{k,l}(p, u)$ implies $i \in Stubborn^{k,l}(p, u')$.

1. By Claim 3.4, $u, u' \in Over_i^{k,l-1}$. Hence, by the inductive assumption, $i \in Stubborn^{k,l-1}(p, u)$ implies $i \in Stubborn^{k,l-1}(p, u')$.
2. $u' \in Over_i^{k,l}$ by the assumption.
3. Since $u' \in Over_i^{k,l}$, every node in the path p after the node u' is also in $Over_i^{k,l}$ and hence is not *Relevant* for the path p. Therefore, if at least $0.9 \cdot \hat{n}$ variables $x_{i,j}$ of type (k, l) were $Forced$ to 0 at some node of the path p up to u then all these variables were $Forced$ to 0 at some node of the path p up to u'.

Hence, $i \in Stubborn^{k,l}(p, u')$. □

Let **Path** be the random path defined by applying the adversary strategy for answering questions of the BP and let **Leaf** be the random leaf of the BP reached by $Path$.

3.4 The main claim

The following is the main claim needed for our lower bound. The rest of the subsection is dedicated for the proof of the claim.

Claim 3.7 *With probability exponentially close to 1, for every k, l,*

$$|Stubborn^{k,l}(Path, Leaf)| \leq (0.9)^l \cdot m.$$

Proof:
The proof is by induction on l. The base case, $l = 0$, is true (with probability 1) by the definition of $Stubborn^{k,0}(p, u)$.

We say that a path p is **Cheating** if for some $u \in p$, $|Ones(u)| \geq 2n^\epsilon$.

Claim 3.8 *The probability that $Path$ is $Cheating$ is at most 2^{-n^ϵ}.*

Proof:
First note that by the definition of $Strategy(p, u)$, the adversary strategy never answers a query by 1 with probability more than 1/2 (i.e., the probability for 0 is always at least 1/2). We say that a node u is *Cheating* if $|Ones(u)| \geq 2n^{\epsilon}$. Let u be a node that is *Cheating*. Then there exist at least $2n^{\epsilon}$ variables $x_{i,j}$ in $Ones(u)$. Each one of these variables is in $Queries(u)$ and it gets the value 1 along every path to u. In order to have $u \in Path$, each one of these variables must be queried along $Path$ (and must get the value 1 along $Path$). Since the adversary strategy answers by 1 with probability of at most 1/2, the probability that all these variables get the value 1 along $Path$ is at most $2^{-2n^{\epsilon}}$. Hence, the probability that one particular *Cheating* node u satisfies $u \in Path$ is at most $2^{-2n^{\epsilon}}$. Since the total number of nodes in the BP is (by our assumption) at most $2^{n^{\epsilon}/10}$, the probability that there exists a *Cheating* node $u \in Path$ is at most $2^{n^{\epsilon}/10} \cdot 2^{-2n^{\epsilon}} < 2^{-n^{\epsilon}}$. $\square$

Let p be a path starting at the root. Define:

$\mathbf{Forced^{k,l}(p)}$ = the set of variables $x_{i,j}$ of type (k, l) such that $x_{i,j}$ was *Forced* to 0 at some node of the path p.

$\mathbf{Tough_i^{k,l}(p)}$ = the set of variables $x_{i,j}$ of type (k, l) such that all the following are satisfied:

1. For some $u \in p$ and some pigeon i', $x_{i,j}$ was *Forced* to 0 at u by $x_{i',j}$.
2. $x_{i',j}$ was queried at $u' \in p$ such that $u' \in Before_i^{k,l}$.

Claim 3.9 *If a path p is not Cheating then for every k, l, i,*

$$|Tough_i^{k,l}(p)| < 2n^{\epsilon}.$$

Proof:
Denote (in the proof of this claim) $n' = 2n^{\epsilon}$. Let v be the first node on p such that $v \in Stage_i^{k,l}$. Assume for a contradiction that $|Tough_i^{k,l}(p)| \geq n'$. Then there exist n' different nodes $u_1, ..., u_{n'} \in p$, and n' different nodes $u'_1, ..., u'_{n'} \in p$, and n' different variables $x_{i,j_1}, ..., x_{i,j_{n'}}$ of type (k, l), and n' different variables $x_{i'_1,j_1}, ..., x_{i'_{n'},j_{n'}}$ (of type (k, l)), such that for every $1 \leq r \leq n'$:

1. x_{i,j_r} was *Forced* to 0 at u_r by $x_{i'_r,j_r}$.
2. $x_{i'_r,j_r}$ was queried at u'_r and $u'_r \in Before_i^{k,l}$.

By (1) we know that $u_1, ..., u_{n'} \in Stage_i^{k,l}$. Hence,

$$v \leq u_1, ..., u_{n'}.$$

By (2) we know that $u'_1, ..., u'_{n'} \in Before_i^{k,l}$. Hence,

$$u'_1, ..., u'_{n'} < v.$$

By (2), each variable $x_{i'_r,j_r}$ was queried at u'_r and hence

$$x_{i'_r,j_r} \in Queries(v).$$

By (1), each variable $x_{i'_r,j_r}$ satisfies $x_{i'_r,j_r} \in Ones(u_r)$ and hence

$$x_{i'_r,j_r} \in Ones(v).$$

Therefore,

$$|Ones(v)| \geq 2n^{\epsilon},$$

that is, the path p is *Cheating*. □

Let u' be a non-leaf node of type (k, l) such that $Label(u') = x_{i',j}$, and let p be a path from the root to u'. Define:

> $\mathbf{Shadow(p, u')}$ = (for u' with $Label(u') = x_{i',j}$ of type (k, l)) the set of variables $x_{i,j}$ such that all the following are satisfied:
>
> 1. $i \in OpenPigeons(u')$.
> 2. $i \in Stubborn^{k,l-1}(p, u')$.
> 3. $u' \in Stage_i^{k,l}$.

Claim 3.10 *For any non-leaf u' of type (k, l) and any path p from the root to u':*

1.

$$|Shadow(p, u')| < 2m_k.$$

2.

$$|Shadow(p, u')| \leq |Stubborn^{k,l-1}(p, u')|.$$

Proof:
Part (2) is obvious by the definition of $|Shadow(p, u')|$. We will prove part (1).

If $k = 1$ then $2m_k = 2^{n^{\epsilon}}$ and the claim follows by our assumption that $m < 2^{n^{\epsilon}/10}$. For $k > 1$, assume for a contradiction that $|Shadow(p, u')| \geq 2m_k$. Then there exist $2m_k = m_{k-1}$ different pigeons $i \in OpenPigeons(u')$ such that $u' \in Stage_i^{k,l}$. By the definition of $Stage_i^{k,l}$, each one of these pigeons satisfies $|Queries_i^{k,l}(u')| \geq b_{3k-1,3l-1} - 1$, and hence by Claim 3.2,

$$|Queries_i(u')| > n^{2\epsilon} \cdot (b_{3k-1,3l-1} - \hat{n}) \geq n^{2\epsilon} \cdot (1/3) \cdot n^{1-3\epsilon} \cdot (3k - 2) > (k - 1) \cdot n^{1-\epsilon} = n_{k-1}.$$

Hence, u' is a pigeon-axiom of order $k - 1$ and cannot be a non-leaf node. □

Let p be a path starting at the root. Define:

$\mathbf{Bad^{k,l}(p)}$ = the set of variables $x_{i,j}$ of type (k,l) such that all the following are satisfied:

1. For some $u' \in p$, $x_{i,j} \in Shadow(p, u')$.
2. The variable $x_{i',j} = Label(u')$ was set to 1 along the path p.

$\mathbf{Good^{k,l}(p)}$ = the set of variables $x_{i,j}$ of type (k,l) such that all the following are satisfied:

1. For some $u \in p$, $x_{i,j} = Label(u)$ (i.e., $x_{i,j}$ was queried at u).
2. u was *Relevant* for p and $x_{i,j}$ was not *Forced* to 0 at u (hence, $Strategy(p, u)$ answered the query $x_{i,j}$ by 1 with probability $1/2m_k$).

Claim 3.11 *For any i, j, k, l, p, if $x_{i,j} \in Forced^{k,l}(p)$ then:*

1.

$$x_{i,j} \in Tough_i^{k,l}(p) \bigcup Bad^{k,l}(p).$$

2.

$$x_{i,j} \notin Good^{k,l}(p).$$

Proof:
Part (2) is obvious by the definition of $Good^{k,l}(p)$. We will prove part (1).

If $x_{i,j} \in Forced^{k,l}(p)$ then $x_{i,j}$ was *Forced* to 0 at some node u of the path p, by some variable $x_{i',j}$. The variable $x_{i',j}$ was queried at some node $u' < u$ of the path p and was set to 1 (along the path p). If $u' \in Before_i^{k,l}$ then $x_{i,j} \in Tough_i^{k,l}(p)$ (by the definition of $Tough_i^{k,l}(p)$) and the claim follows. Otherwise, $u' \notin Before_i^{k,l}$, and since u is *Relevant* for p, we also know that $u \in Stage_i^{k,l}$ and hence that $u' \notin Over_i^{k,l}$. Hence, $u' \in Stage_i^{k,l}$. We will show that in that case $x_{i,j} \in Shadow(p, u')$ and hence $x_{i,j} \in Bad^{k,l}(p)$.

1. Since u is *Relevant* for p, we know that $i \in OpenPigeons(u)$. Hence by Claim 3.1, $i \in OpenPigeons(u')$.
2. Since $u, u' \in Stage_i^{k,l}$, we know by Claim 3.4 that $u, u' \in Over_i^{k,l-1}$. Since u is *Relevant* for p, we know that $i \in Stubborn^{k,l-1}(p, u)$. Hence by Claim 3.6, $i \in Stubborn^{k,l-1}(p, u')$.
3. We already showed, $u' \in Stage_i^{k,l}$.

Hence, $x_{i,j} \in Shadow(p, u')$. □

Denote $B = |Bad^{k,l}(Path)|$. Denote $G = |Good^{k,l}(Path)|$. Denote $D = B - G$. Denote the variables in $Good^{k,l}(Path)$ by $g_1, ..., g_G$, where the ordering is according to the order of these queries in $Path$. Note that $B, G, D, g_1, ..., g_G$ are random variables. Denote

$$\hat{m} = (0.9)^{l-1} \cdot m.$$

Denote

$$m' = \min[2m_k, \hat{m}].$$

For $1 \leq r \leq G$, denote by y_r a random variable that gets the value (-1) if g_r was set to 0 along $Path$, and the value $(m'-1)$ if g_r was set to 1 along $Path$. For $r > G$ define y_r to be an (independently chosen) random variable that gets the value (-1) with probability $1-(1/2m_k)$ and the value $(m'-1)$ with probability $1/2m_k$. Note that $B, G, D, g_1, ..., g_G, \hat{m}, m', y_1, ..., y_G$ depend on (k,l). For simplicity of the notations we do not use here the superscript ^{k,l}.

Claim 3.12 *$y_1, y_2, ...$ are independent random variables, and each one of them gets the value (-1) with probability $1-(1/2m_k)$ and the value $(m'-1)$ with probability $1/2m_k$.*

Proof:
Follows immediately by the definition of the adversary strategy, by the definition of the set $Good^{k,l}(p)$ and by the definition of the random variables $y_1, y_2, ...$ □

Claim 3.13 *Assume that $|Stubborn^{k,l-1}(Path, Leaf)| \leq \hat{m}$ (the inductive assumption). Then*

$$D \leq \sum_{r=1}^{G} y_r.$$

Proof:
Since $D = B - G$, it is enough to prove that

$$B \leq G + \sum_{r=1}^{G} y_r = \sum_{r=1}^{G} (y_r + 1).$$

First note that by Claim 3.10 and Claim 3.5, for any $u' \in Path$ of type (k,l),

$$|Shadow(Path, u')| \leq m'.$$

Every $x_{i,j} \in Bad^{k,l}(Path)$ is in $Shadow(Path, u')$ for some $u' \in Path$ (of type (k,l)) such that the variable $x_{i',j} = Label(u')$ was set to 1 along $Path$. Note, however, that the only variables of type (k,l) that were queried along $Path$ and that could be set to 1 are the variables $g_1, ..., g_G$ (these are the only queries for which the answer 1 was given with non-zero probability). Denote by $u'_1, ..., u'_G$ the nodes on $Path$ where $g_1, ..., g_G$ were queried. Then, $x_{i',j} = g_r$ for some g_r that was set to 1 along $Path$, and hence, $x_{i,j} \in Shadow(Path, u'_r)$ for some u'_r such that g_r was set to 1 along $Path$ (we will say in this case that g_r is responsible for $x_{i,j} \in Bad^{k,l}(Path)$). The total number of variables $x_{i,j}$ that g_r is responsible for is at most m' if g_r was set to 1 (along $Path$), and is 0 if g_r was not set to 1 (along $Path$). That is, the total number of variables $x_{i,j}$ that g_r is responsible for is bounded by $(y_r + 1)$. Hence,

$$|Bad^{k,l}(Path)| \leq \sum_{r=1}^{G} (y_r + 1).$$

□

Claim 3.14 *Assume that* $|Stubborn^{k,l-1}(Path, Leaf)| \leq \hat{m}$ *(the inductive assumption) and assume (for a contradiction) that* $|Stubborn^{k,l}(Path, Leaf)| > 0.9 \cdot \hat{m}$*. Then all the following are satisfied:*

1.
$$G < 0.2 \cdot \hat{m} \cdot \hat{n}.$$

2. *If Path is not Cheating then*
$$B > 0.8 \cdot \hat{m} \cdot \hat{n}.$$

3. *If Path is not Cheating then*
$$D > 0.6 \cdot \hat{m} \cdot \hat{n}.$$

Proof:
First note that by the assumption $|Stubborn^{k,l-1}(Path, Leaf)| \leq \hat{m}$ and by Claim 3.5 and by the fact that along any path the total number of nodes in $Stage_i^{k,l}$ is at most $\hat{n}$, we can conclude that the total number of nodes $u \in Path$ of type (k, l) that are *Relevant* for *Path* is at most $\hat{m} \cdot \hat{n}$.

Denote $F = |Forced^{k,l}(Path)|$. Since $|Stubborn^{k,l}(Path, Leaf)| > 0.9 \cdot \hat{m}$, and since for each $i \in Stubborn^{k,l}(Path, Leaf)$, at least $0.9 \cdot \hat{n}$ variables $x_{i,j}$ of type (k, l) were *Forced* to 0 at some node of *Path* (by the definition of $Stubborn^{k,l}(p, u)$), we can conclude that $F > 0.81 \cdot \hat{m} \cdot \hat{n}$.

Now, part (1) follows by Claim 3.11. Part (2) follows by Claim 3.11, Claim 3.9 and by the fact that $2n^{\epsilon} < 0.01 \cdot \hat{n}$. Part (3) follows from part (1) and part (2). □

Claim 3.15 *Assume (for a contradiction) that* $|Stubborn^{k,l}(Path, Leaf)| > 0.9 \cdot \hat{m}$*. Then at least one of the following events occurs:*

1. $|Stubborn^{k,l-1}(Path, Leaf)| > \hat{m}$.

2. *Path is Cheating.*

3. *For some* $s < 0.2 \cdot \hat{m} \cdot \hat{n}$*, we have* $\sum_{r=1}^{s} y_r \geq 0.6 \cdot \hat{m} \cdot \hat{n}$.

Proof:
Assume that (1) and (2) do not occur. Then by part (1) of Claim 3.14, $G < 0.2 \cdot \hat{m} \cdot \hat{n}$, and by part (3) of Claim 3.14, $D > 0.6 \cdot \hat{m} \cdot \hat{n}$. By Claim 3.13, $D \leq \sum_{r=1}^{G} y_r$. Hence, for some $s < 0.2 \cdot \hat{m} \cdot \hat{n}$,

$$\sum_{r=1}^{s} y_r \geq D > 0.6 \cdot \hat{m} \cdot \hat{n}.$$

□

Claim 3.16 *With probability exponentially close to 1, for every $s < 0.2 \cdot \hat{m} \cdot \hat{n}$,*

$$\sum_{r=1}^{s} y_r < 0.6 \cdot \hat{m} \cdot \hat{n}.$$

Proof:
This follows from Claim 3.12, by the standard Chernoff bounds. For any specific s, the probability that $\sum_{r=1}^{s} y_r \geq 0.6 \cdot \hat{m} \cdot \hat{n}$ is at most $2^{-c \cdot \hat{n}}$, where c is a small constant (say $c = 1/5$). Hence, the probability that this happens for some $s < 0.2 \cdot \hat{m} \cdot \hat{n}$ is at most $0.2 \cdot \hat{m} \cdot \hat{n} \cdot 2^{-c \cdot \hat{n}}$, which is smaller than $2^{-c' \cdot \hat{n}}$, where c' is a small constant (say $c' = 1/10$). $\square$

We can now complete the proof of Claim 3.7. By Claim 3.15, if $|Stubborn^{k,l}(Path, Leaf)| > 0.9 \cdot \hat{m}$ then one of 3 events occurs. The probability for the first event is exponentially small (by the inductive assumption for $l - 1$). The probability for the second event is bounded by $2^{-n^{\epsilon}}$ (by Claim 3.8). The probability for the third event is bounded by $2^{-c' \cdot \hat{n}}$ (by Claim 3.16). Note that since the number of possible k, l is relatively small (and hence also the number of induction steps is small), the probabilities add to an exponentially small probability. Hence, the event $|Stubborn^{k,l}(Path, Leaf)| > 0.9 \cdot \hat{m}$ occurs with an exponentially small probability. $\square$

3.5 The lower bound

Recall that we assume that the size of our BP is lower than $2^{n^{\epsilon}/10}$. We will show that such a BP cannot solve the weak pigeon hole principle. Denote by $\mathcal{E}$ the following event:

$\mathcal{E}$ = the event: for every k, l, $|Stubborn^{k,l}(Path, Leaf)| \leq (0.9)^l \cdot m$.

By Claim 3.7,

$$\mathbf{Pr}[\mathcal{E}] > 1/2.$$

Let u be a pigeon-axiom of order k. We will show that given the event $\mathcal{E}$, the probability that $Leaf = u$ is exponentially small. Denote,

$\mathcal{E}_{\mathbf{u}}$ = the event: $Leaf = u$.

Claim 3.17 *For any pigeon-axiom u of order k,*

$$\mathbf{Pr}[\mathcal{E}_u | \mathcal{E}] < 2 \cdot e^{-\hat{n}/20}.$$

Proof:
Let u be a pigeon-axiom of order k. Then, there is a set

$$A_u \subset OpenPigeons(u),$$

such that $|A_u| = m_k$ and for every $i \in A_u$, $|Queries_i(u)| \geq n_k$. By Claim 3.2 and by the definition of $Over_i^{k,l}$, for every $i \in A_u$ and every l,

$$u \in Over_i^{k,l}.$$

Let p be any path from the root to u, such that for every l,

$$|Stubborn^{k,l}(p,u)| \leq (0.9)^l \cdot m$$

(we assume this because otherwise we would have had $p \neq Path$, if the event $\mathcal{E}$ occurs). In particular, for $l = n^\epsilon$,

$$Stubborn^{k,n^\epsilon}(p,u) = \emptyset$$

(since we assume $m < 2^{n^\epsilon/10}$). At the other hand,

$$Stubborn^{k,0}(p,u) = Pigeons.$$

Therefore, for any pigeon i there exists $l = l_{p,u}(i)$, such that $i \in Stubborn^{k,l-1}(p,u)$ and $i \notin Stubborn^{k,l}(p,u)$. In particular, this applies for any pigeon $i \in A_u$. Thus, for every $i \in A_u$ we have (for $l = l_{p,u}(i)$):

1. $i \in Stubborn^{k,l-1}(p,u)$.
2. $u \in Over_i^{k,l}$.
3. $i \notin Stubborn^{k,l}(p,u)$.

By the definition of $Stubborn^{k,l}(p,u)$, this implies that no more than $0.9 \cdot \hat{n}$ variables $x_{i,j}$ of type (k,l) were *Forced* to 0 at some node of p (for $l = l_{p,u}(i)$). Recall that this is true for any $i \in A_u$.

At the other hand, for every $i \in A_u$, since $u \in Over_i^{k,l}$, there are exactly $\hat{n}$ nodes $u_1, ..., u_{\hat{n}} \in p$ of type (k,l), such that for every u_r, $Label(u_r) \in Pigeon_i$, and such that $u_1, ..., u_{\hat{n}}$ are in $Stage_i^{k,l}$. For each of these nodes u_r we have (for $l = l_{p,u}(i)$):

1. $i \in OpenPigeons(u)$, and hence by Claim 3.1, $i \in OpenPigeons(u_r)$.
2. Since $u_r \in Stage_i^{k,l}$, we know by Claim 3.4 that $u_r \in Over_i^{k,l-1}$, and hence also that $u \in Over_i^{k,l-1}$. Since $i \in Stubborn^{k,l-1}(p,u)$, we have (by Claim 3.6), $i \in Stubborn^{k,l-1}(p,u_r)$.
3. $u_r \in Stage_i^{k,l}$.

Hence, each u_r is *Relevant* for the path p. Since no more than $0.9 \cdot \hat{n}$ variables $x_{i,j}$ of type (k,l) were *Forced* to 0 at nodes of p, we conclude that for any $i \in A_u$ (and for $l = l_{p,u}(i)$) at least $0.1 \cdot \hat{n}$ variables $x_{i,j}$ of type (k,l) were *Relevant* for the path p and were not *Forced* to 0 (when queried).

We will now consider the path *Path*. Assume that the event $\mathcal{E}$ occurs. That is, for every l, $|Stubborn^{k,l}(Path, Leaf)| \leq (0.9)^l \cdot m$. Intuitively, in order to have $Leaf = u$, we must have for every l, $|Stubborn^{k,l}(Path,u)| \leq (0.9)^l \cdot m$, and by the above argument we conclude that for every $i \in A_u$, there exists l such that at least $0.1 \cdot \hat{n}$ variables $x_{i,j}$ of type (k,l) were *Relevant* for *Path* and were not *Forced* to 0 (when queried). For each one of these queries the adversary strategy answers by 0 with probability $(1 - 1/2m_k)$ (by the definition of *Strategy*(p,u)). Since $A_u \subset OpenPigeons(u)$, in order to have $Leaf = u$ the adversary strategy must answer all these queries by 0. Since $|A_u| = m_k$, this happens with probability of at most

$$(1 - 1/2m_k)^{0.1 \cdot \hat{n} \cdot m_k} < e^{-\hat{n}/20},$$

if we ignore the condition that $\mathcal{E}$ occurs. Since we need the conditional probability under the assumption that $\mathcal{E}$ occurs, we have an additional factor of 2 (using the bound that $\mathcal{E}$ occurs with probability of at least 1/2).

More formally, we need to define,

> $\mathbf{\Gamma_u}$ = the event: for every $i \in A_u$ there exists $l = l(i)$, such that at least $0.1 \cdot \hat{n}$ variables $x_{i,j}$ of type (k,l) were *Relevant* for *Path* and were not *Forced* to 0 (when queried).

Assuming that the event $\mathcal{E}$ occurs, we know that if Γ_u does not occur then $Leaf \neq u$. Therefore,

$$\mathbf{Pr}[\mathcal{E}_u|\mathcal{E}] = \mathbf{Pr}[\mathcal{E}_u|\Gamma_u, \mathcal{E}] \cdot \mathbf{Pr}[\Gamma_u|\mathcal{E}].$$

If $\Gamma_u, \mathcal{E}$ occur then for every $i \in A_u$ and for $l = l(i)$, at least $0.1 \cdot \hat{n}$ variables $x_{i,j}$ of type (k,l) were *Relevant* for *Path* and were not *Forced* to 0 (when queried). For each one of these queries the adversary strategy answers by 0 with probability $(1 - 1/2m_k)$, and in order to have $Leaf = u$ the adversary strategy must answer all these queries by 0. Hence,

$$\mathbf{Pr}[\mathcal{E}_u, \Gamma_u, \mathcal{E}] \leq (1 - 1/2m_k)^{0.1 \cdot \hat{n} \cdot m_k} < e^{-\hat{n}/20},$$

that is,

$$\mathbf{Pr}[\mathcal{E}_u|\Gamma_u, \mathcal{E}] < \frac{e^{-\hat{n}/20}}{\mathbf{Pr}[\Gamma_u, \mathcal{E}]}.$$

Therefore,

$$\mathbf{Pr}[\mathcal{E}_u|\mathcal{E}] = \mathbf{Pr}[\mathcal{E}_u|\Gamma_u, \mathcal{E}] \cdot \mathbf{Pr}[\Gamma_u|\mathcal{E}] < \frac{e^{-\hat{n}/20}}{\mathbf{Pr}[\Gamma_u, \mathcal{E}]} \cdot \mathbf{Pr}[\Gamma_u|\mathcal{E}] = \frac{e^{-\hat{n}/20}}{\mathbf{Pr}[\mathcal{E}]} < 2 \cdot e^{-\hat{n}/20}.$$

□

Our lower bound is now proved in the following way: By the definition of *Strategy*(p,u) the answer given is always 0 if $x_{i,j}$ is *Forced* to 0. Therefore, *Path* never reach a hole-axiom, and hence the probability that *Leaf* is a hole-axiom is 0. At the other hand, by Claim 3.17 the conditional probability that *Leaf* is a pigeon-axiom, given that the event $\mathcal{E}$ occurs, is at most

$$2 \cdot e^{-\hat{n}/20} \cdot m < 2 \cdot e^{-\hat{n}/20} \cdot 2^{n^{\epsilon}/10} < 1/2 < \mathbf{Pr}[\mathcal{E}]$$

(since the total number of leaves u is at most m). Therefore, there is a positive probability that $Leaf$ is not a pigeon-axiom, and hence is not an axiom at all. We conclude that a read once BP of size $< 2^{n^\epsilon/10}$ cannot solve the weak pigeon hole principle.

Corollary 3.1 *Any read once Branching Program for the weak pigeon hole principle is of size at least* $2^{n^\epsilon/10}$.

References

[BeP1] Beame, P., and Pitassi, T., "Simplified and improved resolution lower bounds," *Foundations of Computer Science*, 1996, pp. 274-282.

[BeP2] Beame, P., and Pitassi, T., "Propositional proof complexity: past, present, and future," *Current Trends in Theoretical Computer Science*, 2001, pp. 42-70.

[BuP] Buss, S., and Pitassi, T., "Resolution and the weak pigeonhole principle," *Lecture Notes in Computer Science, Springer-Verlag*, vol. 1414, 1998, pp. 149-156. (Selected Papers of Computer Science Logic 11th International Workshop, 1997).

[BSW] Ben-Sasson, E., and Wigderson, A., "Short proofs are narrow–resolution made simple," *Journal of the ACM*, 48(2), 2001, pp. 149-168.

[BT] Buss, S., and Turan, G., "Resolution proofs of generalized pigeonhole principles," *Theoretical Computer Science*, 62, 1988, pp. 311-317.

[CR] Cook, S., and Reckhow, R., "The relative efficiency of propositional proof systems," *Journal of Symbolic Logic*, 44(1), 1979, pp. 36-50.

[CS] Chvatal, V., and Szemeredi, E., "Many hard examples for Resolution," *Journal of the ACM*, 35, 1988, pp. 759-768.

[Hak] Haken, A. "The intractability of Resolution," *Theoretical Computer Science*, 39, 1985, pp. 297-308.

[Kra] Krajicek, J., *Bounded Arithmetic, Propositional Logic and Complexity Theory,* Cambridge University Press, 1996.

[LNNW] Lovasz, L., Naor, M., Newman, I., and Wigderson, A., "Search problems in the decision tree model," it SIAM Journal on Discrete Mathematics 8(1), 1995, pp. 119-132.

[PWW] Paris, J., Wilkie, A., and Woods, A., "Provability of the pigeonhole principle and the existence of infinitely many primes," *Journal of Symbolic Logic*, 53 Number 4, 1988, pp. 1235-1244.

[Raz] Raz, R., "Resolution lower bounds for the weak pigeonhole principle," *Symposium on Theory of Computing*, 2002, pp. 553-562.

[Razb1] Razborov, A., "Lower bounds for the polynomial calculus," *Computational Complexity*, vol. 7, 1998, pp. 291-324.

[Razb2] Razborov, A., "Improved resolution lower bounds for the weak pigeonhole principle", *Electronic Colloquium on Computational Complexity (ECCC)*, 8(055), 2001.

[Razb3] Razborov, A., "Resolution lower bounds for the weak functional pigeonhole principle", *Electronic Colloquium on Computational Complexity (ECCC)*, 8(075), 2001. (to appear in *Theoretical Computer Science*).

[Razb4] Razborov, A., "Resolution lower bounds for perfect matching principles", *Proc. of the 17th IEEE Conference on Computational Complexity*, 2002, pp. 29-38.

[Razb5] Razborov, A., "Proof complexity of pigeonhole principles," *Developments in Language Theory*, 2001, pp. 100-116.

[RWY] Razborov, A., Wigderson, A., and Yao, A., "Read-once branching programs, rectangular proofs of the pigeonhole principle, and the transversal calculus," *Symposium on Theory of Computing*, 1997, pp. 739-748.

[Urq] Urquhart, A., "Hard examples for Resolution," *Journal of ACM*, Vol. 34, Number 1, January 1987, pp. 209-219.