An Introduction to Circuit Complexity and A Guide to Håstad's Proof

Allan Heydon

June, 1990
CMU-CS-90-1413

School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213

Abstract

This report provides a complete exposition of the main proof in Johan Håstad's thesis [Hås87]. The result gives a lower bound on the size of certain Boolean circuits computing the PARITY function, and it implies that $AC^0 \subset NC^1$. Every effort has been made to make the proof understandable for someone with no background in the area of theoretical circuit complexity. To that end, the report begins by introducing the basic definitions and classes of the field. The proof is then motivated by a section explaining why circuits are of interest to theoretical computer scientists.

Before stating and proving Håstad's result, some preliminary concepts are presented. These ideas are the "building blocks" of the proof itself. A brief history of related results is given. Then, an intuitive description of the proof and a "road map" of its structure (which has several levels and branches) are presented to provide an overall gist of what is going on behind the formal mathematics which follow. The heart of the proof is the so-called "Switching Lemma", which is given considerable attention. The main result and a corollary are then stated and proven.

This research was supported by the Defense Advanced Research Projects Agency (DOD) and monitored by the Avionics Laboratory, Air Force Wright Aeronautical Laboratories, Aeronautical Systems Division (AFSC), Wright-Patterson AFB, Ohio 45433-6543 under Contract F33615-87-C-1499, ARPA Order No. 4976, Amendment 20.

The views and conclusions contained in this document are those of the author and should not be interpreted as representing the official policies, either expressed or implied, of DARPA or the U.S. government.

Keywords: circuit complexity, exponential lower-bound, parity, AC^0.

Contents

List of Figures

Chapter 1

Introduction

This tech report summarizes the main result of Johan Håstad's thesis [Hås87], proving an exponential lower bound on the size of (non-uniform) AC^0 circuits computing the PARITY function. If those terms don't mean anything to you, don't worry — you're about to discover the world of circuit complexity. We'll start at a simple level in this introduction, rigorously defining all the important concepts and classes we'll need to understand Håstad's result. We'll also try to motivate the study of circuit complexity, and in so doing, to explain how it relates to complexity theory in general. Before doing that, however, we define the all-important structure at the center of circuit complexity: the Boolean circuit.

1.1 The General Boolean Circuit Model

Definition 1 (Boolean Circuit) *A* Boolean circuit *is a directed acyclic graph (DAG) in which each of the arcs between nodes represents a* wire *and in which the nodes are partitioned into 3 sets:* inputs, outputs, *and* gates. *We call the number of wires leading into a node its* fan-in, *and the number of wires leading out of a node its* fan-out. *Let $N = \{x_1, \ldots, x_n\}$ be the set of input variables. The inputs are nodes labeled with variables from N or their complement (a variable or its complement is called a* literal*), and they must have a fan-in of 0. The outputs are nodes labeled with variables $y_1, \ldots, y_r$, and they must have a fan-in of 1 and a fan-out of 0. Gates are the internal nodes which are neither inputs nor outputs; they are special because each one has a Boolean function associated with it, as explained below. Wires which lead into a gate are the* inputs *to that gate, and those which lead out of it are its* outputs.

Notice that the terms "input" and "output" in isolation are ambiguous because they can be associated with either entire circuits or individual gates. We use n and r to represent the number of inputs and outputs of the circuit, respectively. Both the inputs and the outputs must be ordered; we can then denote the inputs by $x_1, \ldots, x_n$ and the outputs by $y_1, \ldots, y_r$.

We want to use Boolean circuits to compute arbitrary *Boolean functions* — i.e., functions $F : \{0,1\}^n \to \{0,1\}^r$ from n inputs to r outputs. This is done by associating a Boolean function with each of the gates. That is to say, with each gate g_i we associate a Boolean function $f_i : \{0,1\}^k \to \{0,1\}$, where k is the fan-in of the gate.[1] The function associated with the gate determines the gate's *type*. If the gate happens to have a fan-out greater than 1, the single output

[1] It should be noted that the inputs to each gate must be ordered, just as the inputs to the overall circuit are. This is necessary to make the correspondence between the inputs to the gate g_i and the arguments to the function f_i unique.

computed by the gate will be propagated along all of its output wires. It should be noted that the Boolean functions computed by each gate are typically rather simple, the most common ones being AND ($\wedge$), OR ($\vee$), and NOT ($\neg$). However, there is no theoretical limit to the complexity of these functions.

Definition 2 (Circuit Size/Depth, Gate Level) *Two important properties of a circuit are its size and depth. The* size *of a circuit is simply the number of gates it contains, while its* depth *is the number of gates along the longest path from an input to an output. The* level *of a gate is the length of the longest path from any input to that gate. Notice that the depth of the circuit is the same as the maximum gate-level value over all gates.*

Definition 3 (Bottom Fan-In) *We define the* bottom fan-in *of a circuit to be the maximum fan-in of the gates on level 1 of the circuit.*

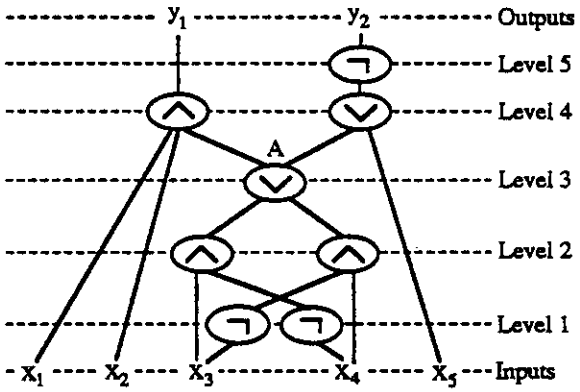


Figure 1.1: A Simple Boolean Circuit.

Example 4 Consider the circuit shown in Figure 1.1. The direction of each arc has been omitted for simplicity. We will implicitly assume that the direction of arcs in this paper's circuit drawings is upward, with the inputs toward the bottom of the page and the outputs toward the top.

In this particular case, the circuit computes a Boolean function of five inputs using AND, OR, and NOT gates, and produces two outputs. The size of the circuit is eight, and its depth is five; it therefore has five gate levels. All non-outputs in this circuit have fan-out one, except for the gate marked "A" and the inputs x_3 and x_4, which have fan-out two. The bottom fan-in of the circuit is one, since both gates on level 1 have fan-in one.

Once a Boolean function is assigned to each gate, the circuit as a whole computes a Boolean function. There is a natural way to compute the function $F_n(x_1, \ldots, x_n)$ represented by a circuit C_n. In essence, we work "up" the circuit, computing and propagating values whenever possible, until the outputs have been computed. First, we label each of the inputs with either 0 or 1 to correspond with the inputs $(x_1, \ldots, x_n)$; all gates are initially unlabeled. We then consider all the unlabeled gates. We pick any unlabeled gate g_i all of whose input wires originate from labeled gates. These are the inputs to that gate's Boolean function f_i. Since they are all specified, we can compute f_i, and we label g_i with the result. This process repeats until we have finally labeled the inputs to all of the output nodes, at which point the overall value $F_n(x_1, \ldots, x_n)$ of the circuit has been computed.

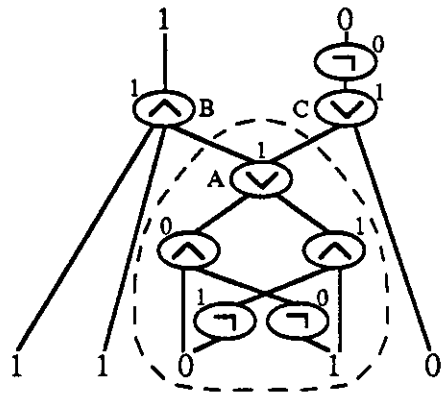


Figure 1.2: Circuit of Figure 1.1 Labeled on Input $\vec{x} = (1,1,0,1,0)$.

Example 5 The labeling for the circuit of Figure 1.1 on input $\vec{x} = (1,1,0,1,0)$ is shown in Figure 1.2. In this case, the output is $y_1 = 1$, $y_2 = 0$. By trying all 4 values for x_3 and x_4, we determine that gate A will be labeled with 1 if and only if either x_3 or x_4 is 1, but not both. Another way to put this is that gate A is labeled with the *exclusive OR* (XOR) of x_3 and x_4, or more generally, with their sum modulo 2.

Notice that the value of gate A is used as the input to two other gates. If we changed the rules of Boolean circuits to say that all internal gates must have a fan-out of exactly one, the sub-circuit inside the dotted line would have to be duplicated, thereby increasing the size of the circuit. The version of the circuit with unbounded fan-out is smaller because it takes advantage of a common sub-circuit, and "re-uses" its computation.

It is important to note that two different circuits may compute identical functions. That is to say, although G_n and H_n may be two different circuits on n inputs, it may be the case that for all inputs $x_1,\ldots,x_n$, $G_n(x_1,\ldots,x_n) = H_n(x_1,\ldots,x_n)$. Henceforth, if we do not explicitly state the number of inputs to a circuit, we will assume it has n inputs (or else the number will be clear from context).

Definition 6 (Equivalent Circuits) *Any two circuits G_n and H_n, having the same number of inputs and outputs and computing the same function, are said to be* equivalent.

Notation 7 ($G_n \equiv H_n$) *If G_n and H_n are equivalent circuits, we write $G_n \equiv H_n$. Notice that $\equiv$ is an equivalence relation on circuits.*

Definition 8 (Forced Circuits) *Let G be an arbitrary Boolean circuit. If $G \equiv 0$ or $G \equiv 1$, we say that G is* forced *to 0 or 1, respectively.*

1.2 Circuit Families

By definition, a circuit is a fixed structure constructed from a finite number of components. In particular, any given circuit has a certain number of inputs n, and it can *only* compute functions of n bits.

Compare circuits to computer programs. A program which sorts words will work on an input list of *any* size. If programs were like circuits, we would have to have a different sorting program for each input size — one program to sort 2 words, one program to sort 3 words, etc.. Just as there

are programs which work on inputs of any size, there are Boolean functions which can be defined for any number of inputs. Here are two important functions we will refer to later:

PARITY Evaluates to 1 iff the sum modulo 2 of the inputs is 1 (put another way, this function is 1 iff the number of 1's in the input is odd).

MAJORITY Evaluates to 1 iff at least half of the inputs are 1.

Since any given circuit has a fixed number of inputs, there is no way any single circuit could compute either of these functions for *all* input sizes. Despite that limitation, we would still like to be able to make statements about the asymptotic complexity of functions such as PARITY and MAJORITY relative to circuits. So long as we consider circuits in isolation however, there is no way to make statements regarding the complexity of circuits for arbitrary Boolean functions in terms of the input size.

We therefore generalize the notion of a single Boolean circuit C_n, computing a function of n bits, to that of a *circuit family* C which can compute a Boolean function F of any number of bits. The circuit family C is simply an infinite collection of Boolean circuits, such that exactly one circuit C_n is in the collection C for each input size $n = 1, 2, \ldots$. To compute the value of $C(x_1, \ldots, x_j)$, we simply select that circuit in the family which computes F on j inputs — namely, C_j — and apply it to the j bits of input.

Definition 9 (Circuit Family Size/Depth Complexity) *We say that the circuits for computing a function F are of* size (depth) complexity $O(f(n))$ *if there is some constant c such that each circuit C_n in the family for F is of size (depth) $\leq c \cdot f(n)$. Circuits for computing F are of* size (depth) complexity $\Omega(f(n))$ *if there is some constant c such that each circuit C_n in the family for F is of size (depth) $\geq c \cdot f(n)$. We often omit the word "complexity" and speak simply of the size or depth of a circuit family.*

We can now speak of, say, polynomial-sized circuits, by which we mean circuits taken from a circuit family whose size grow by a polynomial of n. Since the circuit family is actually a set of circuits, we often use the word "circuits" to mean "circuit family"; if the context is such that this meaning could be confused with "plural of circuit", we will explicitly say "circuit family". Note that saying "circuits of size $O(f(n))$ compute g," means that a circuit *family* for the function g has size *complexity* $O(f(n))$, whereas saying "g is computed by a circuit of size s," means that a *single* circuit containing s gates exists that computes the function g.

An important question that we address later is, "How is a circuit family C represented, and how do we know what the circuit C_n is for any given n?" This question will make more sense once we understand how circuits relate to Turing machines, as discussed in the next section.

1.3 Motivation

Why study circuits? One of the primary unsolved questions in complexity theory is that of whether problems solvable in polynomial time by a non-deterministic Turing machine can also be solved in polynomial time by a deterministic one, i.e., whether P = NP. Many computer scientists believe that in fact the two classes are not equal. To prove this, however, would require proving a super-polynomial deterministic-time lower bound for some problem in NP. Proving non-trivial lower bounds is very difficult in general because it requires showing that *each and every* possible algorithm we could ever imagine to solve some problem requires more than a given amount of

some resource for at least one input. In fact, the best lower bounds produced to date for any NP-complete problem are only slightly super-linear.

An intuitive reason why lower bounds are so difficult to prove is that the Turing machine model of computation is surprisingly complex. Although the rules by which a Turing machine operates are simple, the organization of the machine is flexible enough that its simple mechanism, when taken as a whole, is extremely powerful and difficult to constrain. All known lower-bound proof techniques related to Turing machines relativize to arbitrary oracles. That is to say, if we prove a relationship between two classes using current proof techniques, then that relationship still holds between the classes if they are both given the power of the same arbitrary oracle. Since there are oracles A and B such that $\mathrm{P}^A = \mathrm{NP}^A$ and $\mathrm{P}^B \neq \mathrm{NP}^B$ [BGS75], it is doubtful that lower-bound proofs on Turing machines using current techniques will be successful in proving $\mathrm{P} \neq \mathrm{NP}$.

Here is where circuits come in. Turing machine time is analogous to circuit size. In particular, the class of functions computable by polynomial-sized circuit families[2] is equivalent to that computable by polynomial-time Turing machines. The labeling algorithm given earlier for evaluating a circuit shows that a Turing machine can simulate a polynomial-sized circuit in polynomial time, and Pippenger and Fischer [PF79] show that any Turing machine which runs in time $O(T(n))$ can be built with circuits of size $O(T(n) \cdot \log T(n))$, proving the other direction. Therefore, if a super-polynomial size lower bound could be proven for circuits computing some NP-complete problem, we would know that $\mathrm{P} \neq \mathrm{NP}$!

Circuits are also of interest because they serve as a good model for parallel computation. The algorithm given earlier for simulating a circuit by a sequential machine is ideal for parallelization since, at any given time, there may be many gates all of whose inputs are labeled. We can assign a separate processor to each such gate, and label all of those gates in one step! In this case, the *depth* of a circuit corresponds to parallel *time* and the size corresponds (roughly) to the number of processors required for a parallel machine to evaluate a circuit in this way. Moreover, parallel machines can be built from circuits. These circuits will be considered reasonable if they have a polynomial number of gates (or processors) and require poly-log time. Since parallel machines have a fixed connection architecture, there is also a bound on the number of neighbors each gate may have. We will see later that the complexity class called NC corresponds to these requirements.

1.4 Restrictions on the General Model

One approach to making progress on meaningful lower bound proofs is to limit the flexibility (and hence the power) of the machine. The hope is that a more limited model may yield non-trivial lower bound results, but also, and more importantly, tools for proving general lower bounds which can then be brought to bear on the general model of computation.

There are several orthogonal aspects of the general Boolean circuit model which can be constrained. Here are some of the more important ones.

Size Limits can be placed on the number of gates in the circuits of a circuit family. Due to the correspondence between circuit size and the time required by a sequential machine to simulate the circuit, a reasonable request of circuits is that they be polynomial in size. This requirement is also important because circuit size corresponds to the number of processors required by a parallel machine to simulate the circuit.

[2]These circuits must be *uniform* as described later, and they must be built from gates which are polynomial-time computable.

Depth Limits can be placed on the depth of the circuits in a circuit family. Recall the correspondence between circuit depth and parallel time. We expect a parallel machine to be able to simulate a circuit much faster than a sequential one (otherwise, the extra processors are not doing much good!). Therefore, a reasonable request of polynomial-sized circuits is that they be sub-polynomial in depth, usually some power of $\log n$.

Gate Types Various models allow circuits to contain only certain types of gates. The standard set of allowed gates is { AND, OR, NOT }. Removing NOT gates from this set yields monotone circuits (described below). Augmenting the standard set with more powerful "meta-gates" is analogous to giving the standard model "oracles" for the advanced functions computed by the meta-gates.

For example, if we extend the standard set of gates by also allowing circuits to contain gates computing MAJORITY, we can build constant-depth, linear-size circuits for PARITY (an exercise left to the reader). As this paper will show, constant-depth circuits for PARITY using the three standard gates require an exponential number of gates. Therefore, the "oracle" MAJORITY gates reduce the circuit size of constant-depth PARITY circuits from exponential to linear!

Maximum Fan-In A bound can be placed on the maximum fan-in of every gate in a circuit. Note that if the fan-in is bounded, a function which depends on all its inputs requires at least $O(\log n)$ depth.

Maximum Fan-Out A bound can be placed on the maximum fan-out of every gate in a circuit. Unless otherwise stated, fan-out is typically unbounded. The case in which fan-out is bounded by 1 results in circuits called *Boolean formulas*. Their name comes from the fact that the function computed by any such circuit can be written as a logical formula; this representation is possible because none of the sub-circuits in this case are "re-used".

Uniformity *Uniformity* is a property not of an individual circuit, but of a circuit *family*. Notice that our definition of a circuit family does not preclude circuits which compute arbitrarily difficult functions. For example, we can easily imagine a circuit family for computing an undecidable set such as the halting set K.[3] Let $K_n = \{ x \in \{0,1\}^* \mid x \in K \text{ and } |x| = n \}$. For any input size n, there are 2^n possible inputs, and a certain (finite) subset of these is in K_n. Therefore, we can easily build an OR-of-ANDs[4] circuit which computes the characteristic function of K_n (denoted λ_{K_n}).[5] We construct the circuit as follows: for each $x \in K_n$ there will be an AND gate (call it g_x) which will evaluate to 1 iff the input to the circuit is exactly x; we then take the OR of all these AND gates.[6] It should be clear that this approach generalizes to arbitrarily complex functions.

Although circuits like the one described above for computing λ_{K_n} exist, they are computationally *impossible to construct* with Turing machines. In order for a Turing machine to be

[3] Let $M_1, M_2, \ldots$ be a standard enumeration of all Turing machines. Then $K = \{ i \mid M_i(i) \text{ halts} \}$. It is well known that there is no algorithm for deciding if an arbitrary i is in K; for a proof, see for example [HU79], Chapter 8.

[4] An OR-of-ANDs is a depth-2 circuit in which the top gate is an OR and all level-1 gates are AND's. We also speak of AND-of-ORs, which have the obvious reverse structure.

[5] The *characteristic function* of a set S is that function λ_S such that

$$\lambda_S(x) = \begin{cases} 1 & x \in S \\ 0 & \text{otherwise.} \end{cases}$$

[6] If gate fan-in is bounded, each AND or OR gate with unbounded fan-in can be implemented as a binary tree of AND or OR gates, respectively, with bounded fan-in.

able to simulate a circuit family, the Turing machine must determine the length of the input n, and then find the circuit C_n in the family corresponding to this input length. *Uniformity* is the ability to *construct* the circuit C_n from the input length n. Uniform circuit families are thus represented by the Turing machines that construct them. The name "uniform" comes from that fact that in order for an infinite collection of circuits (namely, the circuit family) to be constructible by a finite program (namely, the Turing machine), the circuits in the family must contain some sort of pattern which makes them uniform. There are varying *degrees of uniformity* depending on how much computational resource we allow the Turing machine that constructs the circuits to use. Common uniformity classes are P-uniform and LOGSPACE-uniform, where the bounds on time and space are measured in terms of the input length n.

Certain combinations of these restrictions have proven especially interesting. Associated with each model is a class of functions. Theoreticians just love defining new classes — most of which are named by acronyms. The world of circuit complexity is no exception. Here is a list of a few of the important classes related to circuits.

NC^i circuit families[7] have the following properties:

- { AND, OR, NOT } gates
- *polynomial* size
- $O\left(\log^i n\right)$ depth
- *bounded* fan-in of 2
- *unbounded* fan-out
- LOGSPACE-uniform. The literature is inconsistent on the uniformity-aspect of NC, so it is often specified explicitly, as in "(non-uniform) NC^1".

AC^i circuit families have all the same characteristics of NC^i circuit families, except they have *unbounded* fan-in.

Monotone Circuits A monotone (non-decreasing) function is one which does not decrease as the input is increased. *Monotone circuit families* are similar in that they implement monotone (non-decreasing) functions. In particular, changing a 0 to a 1 in the input cannot *decrease* the output. MAJORITY is an example of a monotone (non-decreasing) function, while PARITY is not (since changing any input always changes the output). Note that if each of a circuit's gates is monotone, then the circuit as a whole must also be monotone. Therefore, since AND and OR are both monotone functions, circuit families built only from AND and OR gates with no negated variables as input are always monotone.

Definition 10 (NC, AC) *We define* $\mathrm{NC} = \bigcup_{i=0}^{\infty} \mathrm{NC}^i$ *and* $\mathrm{AC} = \bigcup_{i=0}^{\infty} \mathrm{AC}^i$.

There are a few immediate consequences of these definitions. First, by definition, every NC^i circuit is an AC^i circuit, thus, $\forall i, \mathrm{NC}^i \subseteq \mathrm{AC}^i$. Furthermore, every gate in an AC^i circuit has at most a polynomial (of the input size n) number of inputs, since there are only that many gates in the entire circuit. We can simulate each gate in an AC^i circuit by a binary tree of gates of fan-in 2. This tree will have polynomial size and logarithmic depth.[8] By simulating every gate of an

[7] NC stands for "Nick's Class", named after Nicholas Pippenger; see [Coo79] for its first published definition.

[8] If the number of inputs to the AC^i gate is $O\left(n^k\right)$ for some k, then the binary tree with that many leaves simulating the gate has $O\left(n^k\right)$ nodes and $\log n^k = O(\log n)$ depth.

AC^i circuit in this way, we produce a circuit of polynomial size[9] and depth $O\left(\log n \cdot \log^i n\right)$, hence $\forall i, AC^i \subseteq NC^{i+1}$. Taken together, these results imply an alternating chain of containment of the classes NC and AC, namely

$$NC^0 \subseteq AC^0 \subseteq NC^1 \subseteq AC^1 \subseteq NC^2 \subseteq AC^2 \subseteq NC^3 \ldots,$$

so NC = AC. Since circuits in LOGSPACE-uniform NC and AC can be constructed in polynomial time and are polynomial in size by definition, they are easily simulated in polynomial time by a Turing machine, so NC = AC $\subseteq$ P.

The lowest non-trivial class in this chain is AC^0. NC^0 is uninteresting because in constant depth with a bounded fan-in the number of inputs is bounded. Therefore, even simple functions like PARITY and MAJORITY which depend on all or most of their inputs cannot possibly be computed by an NC^0 circuit family. However, it is known that PARITY and MAJORITY are both in NC^1 (the proof is left as an exercise). This leads us to the following natural question: "Can PARITY or MAJORITY be computed by an AC^0 circuit family?" As we shall now see, Johan Håstad's thesis provides a lower bound which proves the answer to be "no," thereby demonstrating that the containment of AC^0 in NC^1 is strict.

[9]Since a polynomial (the number of gates in the AC^i circuit) times a polynomial (the number of gates in the binary tree simulating each AC^i gate) is a polynomial.

Chapter 2

Preliminaries

This chapter outlines some definitions and tools used in the main proof that PARITY cannot be computed by (non-uniform) AC^0 circuits. Unless stated otherwise, we will henceforth be considering circuits (functions) with only 1 output. We will therefore speak of *the* output of a circuit and leave it unlabeled in circuit diagrams.

2.1 Canonical Circuit Model

We will be working with AC^0 circuit families, i.e., those families which contain circuits of some constant depth k, polynomial size n^c, unbounded fan-in, and unbounded fan-out. To make the proof easier, we will work with AC^0 circuits in the following canonical form.

Definition 11 (AC^0 Canonical Form) *We say that an AC^0 circuit is in* canonical form *if:*

- *it contains no* NOT *gates, and*
- *we can assign a "level" to each gate such that every gate's outputs lead to gates on higher levels, and such that all gates on the same level are of the same type.*

For example, the last circuit shown in Figure 2.2 (pg. 11) is in canonical form.

We will be making statements about circuits in canonical form. To extend our results to AC^0 circuits in general, it is important that we understand how much a circuit can grow when it is transformed into an equivalent circuit in canonical form. The following claim provides an answer to that question.

Claim 12 *We can convert an arbitrary AC^0 circuit to canonical form such that its depth does not increase, and such that its size increases by at most a factor of 2.*

Proof We first remove all NOT gates by pushing them down to the literals using DeMorgan's laws.[1] This process can at most *double* the size of the circuit, since we may need two copies of each internal gate — the original value of the gate and its complement. To see this, start at level 1 of the circuit and build a corresponding *complement gate* for each AND or OR gate (build the complement version using DeMorgan's laws). Then do the same at each successive level. We eliminate NOT

[1] DeMorgan's laws say that we can push a NOT through an AND by changing the AND to an OR and complementing each of the inputs. Similarly, we can push a NOT through an OR by changing the OR to an AND and complementing the inputs. In terms of Boolean algebra: $\neg(x_1 \wedge x_2 \wedge x_3) = (\neg x_1) \vee (\neg x_2) \vee (\neg x_3)$ and $\neg(x_1 \vee x_2 \vee x_3) = (\neg x_1) \wedge (\neg x_2) \wedge (\neg x_3)$.

gates in the obvious way, using the complement gate corresponding to the original input to the NOT gate. It should be clear that we will always have the gates we need at lower levels to build the original version and complement version of each gate without adding any new gates (except the new complement gate, of course). Therefore, this step at most doubles the number of gates, and does not increase the depth of the circuit.

We now have a circuit containing only AND and OR gates. We temporarily label each gate with a "level" as follows: we label the inputs with 0, and then label each internal gate with 1 plus the maximum level of all its inputs. The output gate will therefore have the highest label, and it will be the depth of the circuit.

We must now adjust the circuit to meet the second half of the canonical form requirement. We start at the highest level and work our way down the circuit. Since there is only one gate at the top of the circuit, its type determines the types of each of the successive levels. Assume we have forced gates on levels $i+1$ and higher to be of the appropriate type. Without loss of generality, assume the gates on level $i+1$ are all ORs.

We now consider the gates on level i. These gates are supposed to be ANDs, so we can leave the AND gates as they are. We handle the OR gates on level i as follows. The outputs of any such OR gate can go either to a gate on level $i+1$ or to a gate on some level above $i+1$ (or both). Wires of the former type must lead to an OR gate (since all gates on level $i+1$ are ORs). Since AC^0 circuits have unbounded fan-in, we can *merge* the two gates by routing the inputs of the OR on level i to the OR on level $i+1$ instead, as shown in the first transformation of Figure 2.1. Once we have removed all wires of the former type, all outputs go to gates on levels strictly above $i+1$, so we can simply *slide* the OR up one level by relabeling it with $i+1$. The process is shown in the second transformation of Figure 2.1. We will use these "merge" and "slide" operations in later proofs as well.

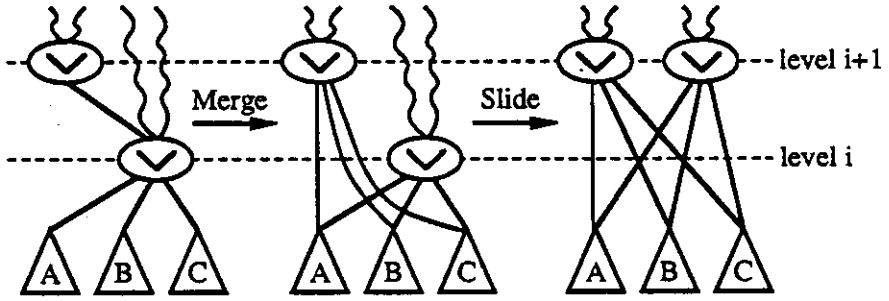


Figure 2.1: Merging and Shifting a Gate.

Notice that the overall levelling process cannot increase the depth of the circuit, nor can it increase the size of the circuit, since we never add a new gate. Therefore, transforming a circuit to canonical form increases its size by at most a factor of 2 and does not increase its depth. ■

Example 13 Figure 2.2 (pg. 11) illustrates the process of transforming a circuit to canonical form. We first remove the NOT gates, and then apply the process described above to level the circuit by gate type.

Observation 14 (Complementing a Canonical Circuit) A simple but important observation is that taking the complement of a circuit in canonical form does not change its size or depth. In fact, the complement version has the exact same structure as the original except for the fact that

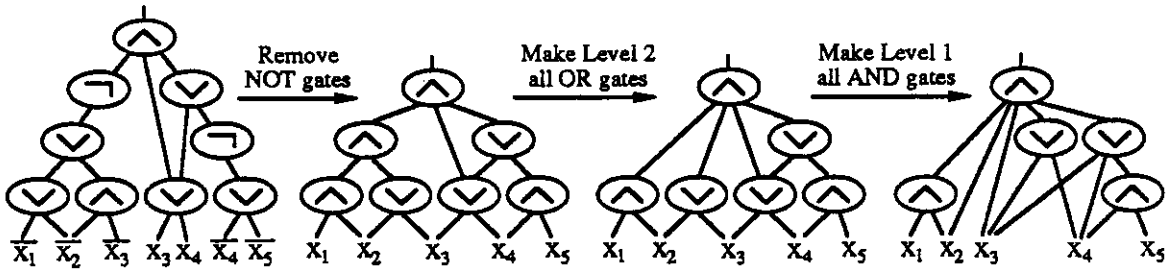


Figure 2.2: Transforming a Circuit to Canonical Form.

AND gates become OR gates (and vice-versa) and all input literals are complemented. This result follows directly from DeMorgan's laws.

2.2 Assignments

Recall that $N = \{ x_1, \ldots, x_n \}$ is the set of input variables.

Definition 15 (Assignment) *An* assignment *is a mapping* $\psi : N \to \{0,1\}$. *An assignment* ψ *is said to* satisfy *a circuit* C_n *(or to be a* satisfying assignment *of* C_n*) if* $C_n(\psi(x_1), \ldots, \psi(x_n)) = 1$.

One aspect of circuits which allows us to prove lower bounds on them is the fact that AND and OR gates can be *forced* to output a certain value by a single input of the proper type. In particular, we note the following:

Observation 16 (Forcing AND and OR Gates) An AND gate with a single input of 0 is forced to 0, regardless of the other inputs to the gate. Similarly, an OR gate with a single input of 1 is forced to 1, regardless of the other inputs to the gate. Furthermore, if one of the inputs to an AND gate is 1, then the output of the gate depends *only* on the values of the other inputs, so the input known to be 1 can be eliminated. In a sense, the 1 input is *absorbed* by the AND gate. If *all* the inputs are absorbed in this way (i.e., all of the inputs are known to be 1) then the AND gate is forced to 1. Similarly, 0 inputs are absorbed by OR gates, and if all the inputs to an OR gate are eliminated in this way, the OR gate is forced to 0.

2.3 Restrictions

Say we fix some of a circuit's inputs, but not necessarily all of them. This idea of a *partial assignment* to the input variables — which we shall call a *restriction* — can cause a "chain reaction" of gate forcings by Observation 16. Even though we do not know the values of the inputs which are unfixed, some gates on level 1 of the circuit may be forced. These forced gates may force other gates, which may force other gates, and so on, with the cascade of gate forcings possibly resulting in the circuit's final output being forced. We now formalize the notion of a restriction, and then discuss the consequences of applying restrictions to circuits.

Definition 17 (Restriction, $|\rho|$) *A* restriction *is a mapping* $\rho : N \to \{0,1,X\}$. *We call 0 and 1* values, *and* X *a* non-value. *A variable mapped to 0 (or 1) is* fixed *at 0 (or 1). A variable mapped to* X *is an undetermined input, and should be considered capable of taking on either the value 0 or the value 1; essentially, it remains a variable. The* size *of a restriction is the number of variables it maps to values. We denote the size of a restriction* ρ *by* $|\rho|$.

Notation 18 (ρ_S) *Let $S \subseteq N$. For any restriction ρ defined on the variable set N we write ρ_S to denote the restriction ρ with domain limited to S.*

Notation 19 ($\vec{X}_S, \vec{0}_S, \vec{1}_S$) *Let $S \subseteq N$. We write $\vec{X}_S$, $\vec{0}_S$, and $\vec{1}_S$ to denote the restrictions which map every variable in S to X, 0, or 1 respectively.*

Notation 20 *We will use the shorthand of representing assignments and restrictions by strings taken from $\{0,1\}^n$ and $\{0,1,X\}^n$ respectively. For example, if $n = 4$ and the restriction ρ is such that $\rho(x_1) = 1$, $\rho(x_2) = X$, $\rho(x_3) = 0$, and $\rho(x_4) = X$, we would write $\rho = 1X0X$.*

As mentioned earlier, when a restriction is applied as input to any particular circuit, there may be a cascade of forced gates, potentially forcing the output computed by the circuit. Whenever a gate becomes forced, we can replace the gate and all its input wires with a constant value. We then continue until no more gates are forced.

In this way, applying a restriction ρ to a circuit C_n results in a new circuit C'_n. Since all variables mapped to 0 or 1 by ρ will either force a gate or be "absorbed" into a gate, C'_n will have exactly $n - |\rho|$ inputs, one for each variable mapped to X by ρ. Thus, if $\rho = \vec{X}_N$, $C'_n \equiv C_n$; if ρ is an assignment, then $C'_n \equiv 0$ or $C'_n \equiv 1$, since the output of any circuit must be forced if all its inputs are fixed.

Notation 21 ($G\lceil\rho$) *We denote the circuit resulting from the application of restriction ρ to circuit G by $G\lceil\rho$, and call the circuit "G restricted by ρ". Restricting G by ρ and then restricting the resulting circuit by σ is written $(G\lceil\rho)\lceil\sigma$. In this case, σ must be a mapping on $n-|\rho|$ variables. We assume that the variables remaining after restricting G by ρ are in the same order as the variables of the original circuit.*

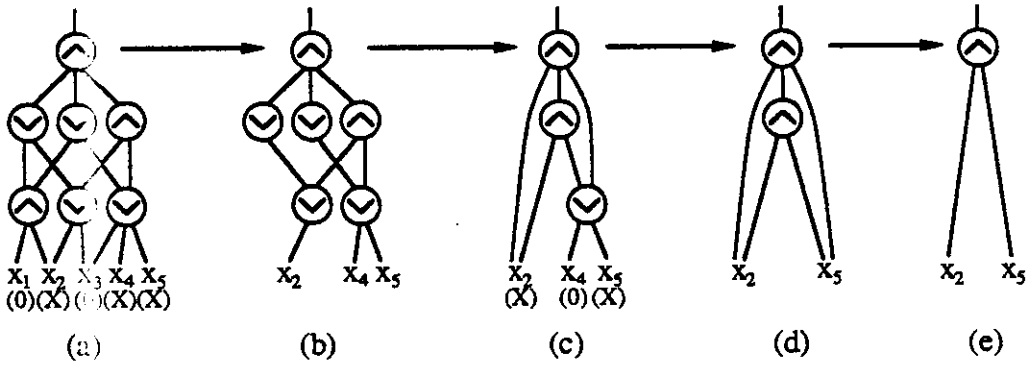


Figure 2.3: A Circuit Restricted by $\rho = 0X0XX$ and then by $\sigma = X0X$.

Example 22 Consider the circuit G shown in Figure 2.3a (for readability, the circuits shown are not in canonical form). Applying the restriction $\rho = 0X0XX$ to circuit (a) yields circuit (b). Using the fact that an OR with 1 input merely outputs that input, we can simplify this circuit to yield circuit (c). If we then apply the restriction $\sigma = X0X$ to this circuit, we get circuit (d). Finally, merging the AND's in this circuit yields circuit (e). Therefore, $(G\lceil\rho)\lceil\sigma \equiv (x_2 \wedge x_5)$.

Observation 23 (Restrictions of PARITY) An important property of PARITY we will use later is that *any* restriction of PARITY results in a circuit which computes PARITY or the complement of PARITY on the remaining inputs. The resulting circuit PARITY$\lceil\rho$ is PARITY or $\neg$PARITY depending on whether it maps an even or odd number of variables to 1, respectively.

Since X's could be further assigned either 0 or 1, a single restriction can be thought of as *generalizing* a set of "more specific" restrictions. We now make this notion precise.

Definition 24 (Generalization Order $\succeq$) *We define the partial order "is a* generalization *of" (denoted symbolically by $\succeq$) between two restrictions ρ, σ such that domain(ρ) = domain(σ) as follows:*

$$\rho \succeq \sigma \iff \begin{array}{l}\rho = \sigma \text{ or } \rho \text{ can be formed from } \sigma \text{ by} \\ \text{replacing some 0's and 1's in } \sigma \text{ with} \\ X\text{'s.}\end{array}$$

For example, $1XX \succeq 101$, $X0X0 \succeq 10X0$, but $X0X0 \not\succeq 1XX0$. Note that $\succeq$ is reflexive, anti-symmetric, and transitive, so it is a partial order. If $\rho \succeq \sigma$, we also say that σ specializes *ρ. If $\rho \succeq \sigma$ and $\rho \neq \sigma$, we write $\rho \succ \sigma$ and say "ρ is a* proper generalization *of σ" or "σ is a* proper specialization *of ρ."*

If circuit G and restriction ρ are such that $G\lceil\rho \equiv 0$ or $G\lceil\rho \equiv 1$, then *all* other restrictions σ such that $\rho \succeq \sigma$, when applied to G, will yield the same circuit. In the case where the resulting circuit is the constant circuit 1, the above statement is expressed symbolically as:

$$(G\lceil\rho \equiv 1 \wedge \rho \succeq \sigma) \implies G\lceil\sigma \equiv 1. \tag{2.1}$$

The key idea is that if σ specializes ρ and both are applied to the same circuit, then all the gates forced by ρ will also be forced (to the same values) by σ, and σ may even force some additional gates. This behavior results from the definition of the generalization order between restrictions and from the behavior of AND and OR gates observed earlier.

Observation 25 (DAG Induced by $\succeq$) The partial order $\succeq$ suggests a DAG of restrictions. In this DAG, the nodes on level i ($0 \leq i \leq n$) are all those restrictions containing exactly i X's. There will thus be $\binom{n}{i} \cdot 2^{n-i}$ nodes at level i; the top level ($i = n$) has one restriction (namely, $\vec{X}_N$), and the bottom level ($i = 0$) consists of the 2^n assignments. A node ρ at level $i+1$ is connected to a node σ at level i if and only if $\rho \succ \sigma$. In this DAG, a restriction containing i X's has exactly $2i$ children (since we can replace each X with 0 or 1 to get a child) and $n - i$ parents (since we can replace each value (0 or 1) with an X to get a parent).

2.4 Minterms

We now consider *minterms*, a certain type of restriction which will be used extensively in the lower-bound proof of PARITY. Put informally, the minterms of a circuit are the most general restrictions (according to the partial order $\succeq$) forcing the circuit to 1.

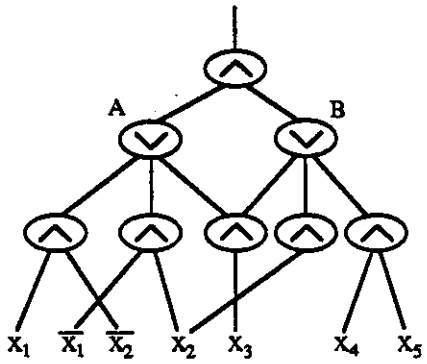
Definition 26 (Minterm) *A* minterm *σ of a circuit G is a restriction on the inputs of G with the following two properties:*

1. *$G\lceil\sigma \equiv 1$ and*
2. *any proper generalization of σ does* not *force G to 1, i.e.,$\forall$ restrictions ϕ ($\phi \succ \sigma \implies G\lceil\phi \not\equiv 1$).*

Notation 27 ($\Gamma(G)$) *For any circuit G, we write $\Gamma(G)$ for the set of all minterms of G.*

Example 28 (Minterms of PARITY) The minterms of PARITY are precisely the 2^{n-1} *satisfying assignments* of PARITY. There is an easy proof of this fact. PARITY obviously has 2^{n-1} satisfying assignments. We now show that each of these assignments must also be a minterm (i.e., we show that any proper generalization of a satisfying assignment cannot force PARITY to 1).

PARITY and its complement are special in the following sense: they are the only two Boolean functions such that changing any single input to the function changes its output. Therefore, any restriction of PARITY (or $\neg$PARITY) which maps some variable to X cannot be a minterm, because replacing the X in the restriction by both 0 and 1 will yield two different output values. By definition, a minterm must force the output to 1, regardless of which value each X takes on. Therefore, the minterms of PARITY and $\neg$PARITY can only be assignments.



Minterms

x_1	x_2	x_3	x_4	x_5
X	X	1	X	X
0	1	X	X	X
1	0	X	1	1

Figure 2.4: A Simple Circuit and its Minterms.

Example 29 Consider the circuit G (in canonical form) shown in Figure 2.4. Since the top-level gate is an AND, any minterm must force the OR gates A and B to 1. Setting x_3 to 1 guarantees this to happen, so $\sigma = XX1XX$ is one minterm of the circuit. We now consider the case where x_3 is 0 in order to find minterms which map x_3 to X. Setting x_2 to 1 forces B to 1 as required, but we must also set $\overline{x_1}$ to 1 in order to force A to 1. Therefore, $\sigma = 01XXX$ is also a minterm of G. Finally, we consider the case where both x_3 and x_2 are 0. For B to be forced to 1 we must set both x_4 and x_5 to 1; for A to be forced to 1 we must set x_1 to 1 and x_2 to 0. From this we conclude that $\sigma = 10X11$ is another minterm of G. Since we have considered every possible way of forcing both A and B to 1, these are the only minterms of the circuit. Notice that — as opposed to PARITY — the minterms of this circuit are different sizes, namely 1, 2, and 4.

At this point, it may seem that the minterms of Example 29 were found in a rather haphazard way, and one might even doubt that all of them were indeed found. Fortunately, there is an easy and precise way to determine a circuit's minterms. Recall the DAG of restrictions discussed in Observation 25 (pg. 13). We determine the minterms of a particular circuit from this DAG by a marking algorithm, which works as follows. First, we mark the nodes on level 0 which are labeled with satisfying assignments of the circuit; all other nodes are initially unmarked. Then, for each higher level in turn, we mark any node all of whose children are marked.

Claim 30 *For any circuit G, this marking algorithm marks exactly those nodes which are restrictions ρ such that $G\lceil\rho \equiv 1$.*

Proof (by induction on the level i)

Base Case By definition, any satisfying assignment ψ is such that $G\lceil\psi \equiv 1$. All other (non-satisfying) assignments force G to 0, so they are not marked.

Inductive Step Assume that all nodes on levels 0 through i are properly marked. We must show that a node ρ on level $i+1$ becomes marked iff $G\lceil\rho \equiv 1$. From the structure of the DAG we know that all children of ρ are marked iff all proper specializations of ρ are marked. By the inductive hypothesis, this is equivalent to saying that all proper specializations of ρ force the circuit to 1.

To complete the proof, we must therefore show that

$$G\lceil\rho \equiv 1 \iff \forall\sigma\,(\rho \succ \sigma \implies G\lceil\sigma \equiv 1).$$

($\Longrightarrow$) If $G\lceil\rho \equiv 1$, then by equation (2.1) (pg. 13), all proper specializations of ρ must also force G to 1. ($\Longleftarrow$) If every assignment ψ, where $\rho \succ \psi$, is such that $G\lceil\psi \equiv 1$, then $G\lceil\rho(\vec{x}) = 1$ for all inputs $\vec{x}$ to $G\lceil\rho$, so the function computed by $G\lceil\rho$ is precisely 1, i.e., $G\lceil\rho \equiv 1$. ■

Once the marking algorithm terminates, the minterms are precisely those nodes σ such that: 1) σ is marked, and 2) *none* of σ's proper ancestors is marked.[2] However, these are precisely the two conditions stated in the minterm definition on page 13.

Observation 31 (Disjunctive Normal Form Using Minterms) We can build a small circuit consisting of a single AND gate to represent any minterm σ. The inputs to the gate are literals corresponding to variables mapped to values by σ: we use input literal x_i if $\sigma(x_i) = 1$ and literal $\overline{x_i}$ if $\sigma(x_i) = 0$. Notice that the fan-in of this AND gate is precisely $|\sigma|$.

It is easy to see that this AND circuit is satisfied by any assignment ψ such that $\sigma \succeq \psi$. Say we build such a circuit for every minterm in $\Gamma(G)$, and we then take the OR of these minterm circuits, resulting in a depth-2, OR-of-ANDs circuit M. Call this the *disjunctive normal form circuit* for G. Notice that the bottom fan-in of M is given by the size of the largest minterm of G. We claim without proof that $G \equiv M$, but the reader should be able to verify this claim easily using the ideas of the restriction DAG and the marking algorithm for determining the minterms of G.

Observation 31 tells us that any circuit G can be represented by the OR of its minterm circuits. In the proofs to come, we will be interested in bounding the bottom fan-in of this latter circuit. Since the bottom fan-in of the OR-of-ANDs is precisely the size of the largest minterm of G, the following notation for expressing the size of the largest minterm of a circuit will be useful.

Notation 32 ($\mathrm{lmt}(G)$) *Let G be an arbitrary Boolean circuit. We define* $\mathrm{lmt}(G)$ *to be the size of the largest minterm of G, i.e.,*

$$\mathrm{lmt}(G) = \max_{\sigma \in \Gamma(G)} |\sigma|$$

If G has no minterms (i.e., if $\Gamma(G) = \emptyset$), we define[3] $\mathrm{lmt}(G) = 0$.

[2] If some proper ancestor of σ is marked, then some parent of σ must also be marked, so it suffices to check that none of σ's *parents* is marked.

[3] This definition will make sense in later contexts, where we will be writing $\mathrm{lmt}(G) < s$ to represent the event that G can be written as an OR-of-ANDs with bottom fan-in $< s$. Certainly, if G has no minterms, $G \equiv 0$, and it is clear that the 0 function *can* be written as an OR-of-ANDs with bottom fan-in $< s$, so defining $\mathrm{lmt}(G) = 0$ in this case is done to make $\mathrm{lmt}(G) < s$ when $\Gamma(G) = \emptyset$.

Observation 33 Although $\mathrm{lmt}(G) < s$ implies the existence of an OR-of-ANDs for G with bottom fan-in $< s$ (by Observation 31), it is interesting to note that the converse is not true. That is, there are OR-of-ANDs circuits having a minterm strictly larger than the maximum fan-in of the AND gates. For example, consider the OR-of-ANDs

$$G \equiv (x_1 \wedge x_2 \wedge x_3) \vee (\overline{x_1} \wedge x_2 \wedge x_4) \vee (x_1 \wedge \overline{x_2} \wedge x_5) \vee (\overline{x_1} \wedge \overline{x_2} \wedge x_6).$$

G has bottom fan-in 3. However, since one of $(x_1 \wedge x_2)$, $(\overline{x_1} \wedge x_2)$, $(x_1 \wedge \overline{x_2})$, or $(\overline{x_1} \wedge \overline{x_2})$ must be true, we see that $\rho = XX1111$ is a minterm of G with size 4, so the largest minterm of G is strictly greater than its bottom fan-in.

2.5 Probability Lemmas

We will need to make use of the following simple lemmas from probability theory throughout the proof. They are stated without proof, but are easily verified. Capital letters in the following lemmas represent events drawn from some universe of events U. We use the standard notation for conditional probability: $\Pr[B \mid A]$ is the probability of event B occurring given that event A occurs.

Lemma 34 *Let $A_1, A_2, \ldots, A_k$ be a partition[4] of the universe of events U. Then for any event B,*

$$\Pr[B] \leq \max(\Pr[B \mid A_1], \Pr[B \mid A_2], \ldots, \Pr[B \mid A_k]).$$

Lemma 35 *Let $A_1, A_2, \ldots, A_k$ be k (not necessarily disjoint) events. Then*

$$\Pr[A_1 \vee A_2 \vee \ldots \vee A_k] \leq \sum_{i=1}^{k} \Pr[A_i],$$

with equality if and only if the events are pairwise disjoint.

Lemma 36 $\Pr[A \wedge B] = \Pr[A] \times \Pr[B \mid A] = \Pr[B] \times \Pr[A \mid B]$.

Lemma 37 $(\Pr[A \mid B \wedge C] \leq \Pr[A \mid C]) \iff (\Pr[B \mid A \wedge C] \leq \Pr[B \mid C])$.

[4]The sets $A_1, A_2, \ldots, A_n$ are said to *partition* a set S if they *cover* S (i.e., $A_1 \cup A_2 \cup \cdots \cup A_n = S$) and if they are *pairwise disjoint* (i.e., $\forall i \neq j,\ A_i \cap A_j = \emptyset$).

Chapter 3

Proof Overview

In this chapter, we present a "high-level" version of the lower bound proof. The proof has several levels and branches. Therefore, a "road-map" of the actual proof has also been included; it can be used as a reference and to determine the overall structure of the document.

For those who also wish to refer to Håstad's thesis, the numbers of those lemmas and theorems which appear in [Hås87] are given parenthetically, as in "([Hås87] Theorem 5.1)". The write-up of the proof maintains nearly the same structure as Håstad's, but introduces a slightly different notation.

Recall that AC^0 circuits have constant depth, unbounded fan-in and fan-out, and polynomial size. We will be proving an exponential size lower bound for circuit families computing PARITY which share all the properties of AC^0 circuits *except* for polynomial size. However, we will loosely speak of AC^0 PARITY circuits; the reader should understand that the circuits are AC^0 in every respect except size.

3.1 History

Let k represent the maximum depth of all the circuits in an AC^0 circuit family for PARITY. Furst, Saxe, and Sipser ([FSS84]) produced the first super-polynomial size lower bound for such circuits. Ajtai ([Ajt83]) independently derived a slightly stronger super-polynomial lower bound. Later, Yao ([Yao85]) proved a strictly exponential lower bound, which led to important results regarding the polynomial time hierarchy. However, Yao's proof is considered technical and difficult to understand. Hastad's proof ([Hås86], [Hås87]) is simpler than Yao's, and it also tightens the exponential lower bounds somewhat. The details of these results are shown in Table 3.1 (pg. 18). These lower bounds should be compared with the best known upper bound of:

$$\text{Size}(AC^0\text{-PARITY}) = O\left(n^{\frac{k-2}{k-1}} 2^{n^{1/(k-1)}}\right), \text{ so } \text{Size}(AC^0\text{-PARITY}) = O\left(n 2^{n^{1/(k-1)}}\right).$$

Constructing circuits satisfying this upper-bound is an interesting exercise.

3.2 Intuition Behind the Proof

The following proof outline should provide an intuition into the rigorous proof of Theorem 68 (pg. 42). However, this description does not use the notion of bottom fan-in, which is crucial

[1] In this bound, $\log^{(i)} n$ denotes the log function iterated i times.

[2] In these bounds, c_k is a constant depending on k.

Reference	Lower Bound
[FSS84][1]	$\Omega\left(n^{\log^{(3k-6)} n}\right)$
[Ajt83][2]	$\Omega\left(n^{c_k \log n}\right)$
[Yao85]	$\Omega\left(2^{n^{1/(4k)}}\right)$
[Hås86][2]	$\Omega\left(2^{c_k n^{1/(k-1)}}\right)$

Table 3.1: A History of Lower Bound Results for AC^0 (depth-k) PARITY Circuit Sizes.

in that proof. Furthermore, it takes the proof of the Switching Lemma in Chapter 4 as a given. Despite such omissions, it is hoped that this sketch will provide some intuition into the ideas behind the math of the formal proof.

3.2.1 A First Stab at the Proof

We will show that no "small" AC^0 circuit families for PARITY exist. What we mean by "small" will be formalized in the actual proof, but we can think of "small" as being "sub-exponential in size" for now. The proof is done by induction on the depth k of possible AC^0 circuit families for PARITY. For the base case ($k=2$), it is fairly easy to show that depth-2 AC^0 circuits require $\Omega(2^n)$ gates.[3] For the inductive step, we use a proof by contradiction: we show that if small depth-$k+1$ PARITY circuits did exist, then small depth-k PARITY circuits would also exist, thereby contradicting the inductive hypothesis.

Therefore, assume a small depth-$k+1$ circuit-family C^{k+1} exists for PARITY. Now, consider any circuit C_n^{k+1} in the family. Without loss of generality, this proof sketch assumes that all circuits are in canonical form[4] and that additionally, every wire from a gate on level i goes to a gate on level $i+1$. We say such circuits are in *strict canonical form*.[5] If we could, in general, transform this circuit into a small depth-k circuit computing PARITY, then applying the transformation to all the circuits in the family would result in the depth-k family we desire.

It is important to note at this point that the depth-k circuit produced by the transformation need not necessarily compute PARITY on all n of the original inputs for us to achieve the same result. For example, suppose that we could convert each of the circuits in the family C^{k+1} to a depth-k circuit computing PARITY on only *half* its inputs. Then, since circuit families are infinite by definition, the new circuit family C^k would also be infinite, and it would have a PARITY circuit for each input size as required.

3.2.2 Reducing Depth by One: Switching Gate Types

One way to decrease the depth of the circuit would be to rewrite it as a circuit computing an *equivalent* function such that the gate types of two adjacent levels of the original circuit are "swapped" (i.e., the roles of one AND level and an adjacent OR level are swapped). Of course, for the new circuit to compute the same function as the old one, we might have to add or remove gates on the two levels involved, and re-wire the circuit near those two levels; the important thing is that the

[3] We will show in Claim 70 (pg. 43) that they require at least $1 + 2^{n-1}$ gates, to be exact.

[4] We do not lose generality because, by Claim 12 (pg. 9), converting a circuit to canonical form can cause its size to at most double. Therefore, if we prove that the canonical version of some circuit family for PARITY must be exponential in size, the original family itself must also be exponential in size.

[5] Converting to strict canonical form may cause a large size blow-up. However, we get around this problem in the actual proof by introducing a more relaxed version of canonical form called *quasi-canonical form*.

changes to the circuit would be local. In the new circuit, there would then be at least two adjacent levels of the same gate type which could be merged to reduce the depth by one.[6] For this swapping operation to work, it would have to:

1. preserve the function computed by the circuit (namely PARITY), but not necessarily the number of inputs (as pointed out above), and
2. result in a small circuit.

We will choose to swap the roles of AND and OR gates on the bottom two levels of the circuit. Consider the set of sub-circuits rooted at gates on level 2. It may be that some of these sub-circuits share level-1 gates with each other (i.e., some of the level-1 gates may have fan-out greater than 1). To make each sub-circuit independent from the others, we first duplicate each of the shared level-1 gates (as many times as necessary), so all gates on level 1 have a fan-out of 1. This step may cause us to create many new gates, but not too many; if we started with a sub-exponential number of gates, we will still have a sub-exponential number of gates after this step, so our circuit will still be "small".

Since the overall circuit is in strict canonical form, all sub-circuits rooted at gates on level 2 are either AND-of-ORs or OR-of-ANDs. Without loss of generality, say they are all AND-of-ORs. We call the operation of writing an AND-of-ORs as an equivalent, small OR-of-ANDs (or vice-versa) the *switching* operation. If we switch each of the AND-of-ORs sub-circuits, we can then merge the two levels of AND gates on levels 2 and 3, thereby reducing the depth of the overall circuit by one. The problem thus boils down to being able to turn any AND-of-ORs into an equivalent OR-of-ANDs (or vice-versa) which does not blow up too much in size.

Unfortunately, it is not always possible to switch an AND-of-ORs to achieve an equivalent OR-of-ANDs of "small" size. For example, consider the AND-of-ORs shown in Figure 3.1a. This circuit's inputs have been divided into $n/2$ groups, with inputs x_i and x_{i+1} (i odd) in the same group. The circuit is true iff at least one of the inputs in each of the $n/2$ groups is true. Notice that the size of this circuit — $n/2 + 1$ — is linear in n. However, when we try to express this function as an OR-of-ANDs, the size blows up to an exponential in n, namely $2^{n/2} + 1$ (see Figure 3.1b). It is clear that simply rewriting the circuit with the roles of AND and OR gates interchanged will not suffice to achieve our result.

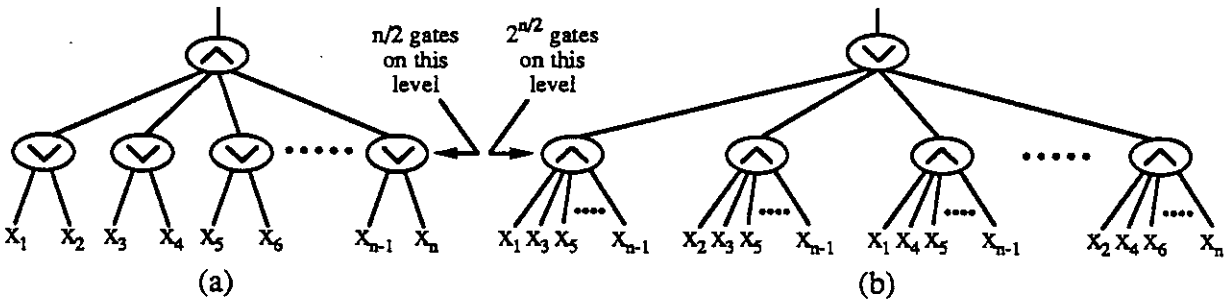


Figure 3.1: Switching May Cause an Exponential Size Blow-up.

Recall from Observation 23 (pg. 12) that PARITY has the following special property: any restriction of PARITY results in either PARITY or ¬PARITY. This leads to the idea of first applying some restriction ρ to our PARITY circuit C_n^{k+1}, and *then* attempting to switch each of the sub-circuits

[6] This merging is possible because fan-in is unbounded in the definition of AC.

rooted at a level-2 gate to produce a circuit equivalent to $C_n^{k+1}\lceil\rho$. The important point is that this resulting circuit can be reduced to depth k and computes PARITY (or its complement).

Essentially, we make the application of a restriction ρ the first step in the overall switching operation. The hope is that the restriction will eliminate enough gates that we can switch the sub-circuits rooted at level-2 gates without causing the resulting circuit to grow too much in size. Figure 3.2 below illustrates the complete switching process applied to a simple depth-3 circuit. Of course, we will only be transforming PARITY circuits this way in the actual proof.

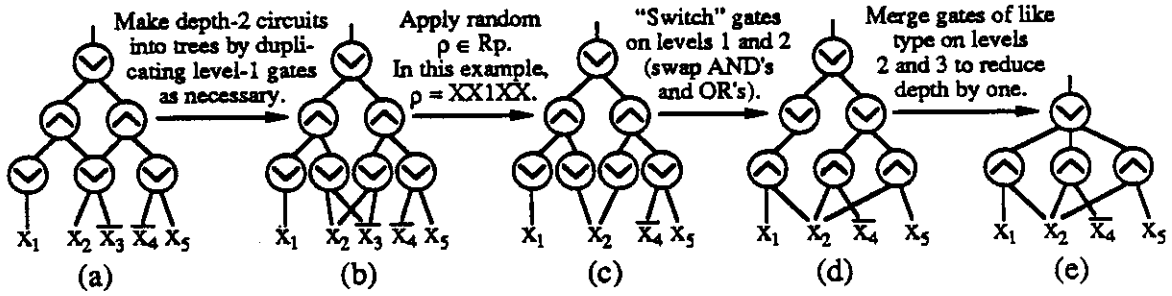


Figure 3.2: Switching a Simple Circuit to Reduce Its Depth by One.

3.2.3 Objections to the Proposed Switching Technique

There are still several possible problems with making this means of converting a depth-$k+1$ family to a depth-k family feasible. We will enumerate them in turn, and then show how each can be dealt with.

1. We said earlier that the switching operation must result in a circuit which computes the same function as the original. What do we do if $C_n^{k+1}\lceil\rho \equiv \neg$PARITY.
2. How do we choose the restriction ρ to use for each conversion? Intuitively speaking, it doesn't seem like a particular restriction (or even a particular *kind* of restriction) will succeed in allowing us to switch *every* possible circuit. Is there some clever way of picking a particular restriction to apply depending on certain key properties of the circuit, or is there perhaps a more powerful method?
3. If ρ maps any variables to values, then the new circuit $C_n^{k+1}\lceil\rho$ will have $n' < n$ inputs. For the new circuit to still be "small", its size must be small in terms of the new number of inputs n' and the new depth k.
4. We have said that we will convert the family C^{k+1} to the family C^k by converting each of the individual circuits C_n^{k+1}, $n = 1, 2, \ldots$. Of course, this process never terminates because there are an infinite number of circuits to convert, but we need only show that such a conversion is theoretically possible. Define *new-inputs*(n) to be the number of inputs to the circuit resulting from the conversion of C_n^{k+1} $(n = 1, 2, \ldots)$. A problem occurs if there is some constant n_0 such that $\forall n$, *new-inputs*$(n) \leq n_0$, because then the new circuit family C^k is finite, and hence, incomplete.

   Observation 38 Notice that a "gap" in the image of *new-inputs* is not a problem because we can always pick a circuit in C^k with more inputs than we need and fix the extra inputs to 0. Therefore, it is not necessary for *new-inputs* to be onto, but it is necessary for the

image of *new-inputs* to be infinite. Of course, this latter condition is also *sufficient* to answer objection (4).

3.2.4 Answering Objections (1) and (2)

We answer objection (1) by pointing out that if $C_n^{k+1}\lceil\rho \equiv \neg\text{PARITY}$, we can just take the complement of the restricted circuit. This is sufficient because taking the complement of a circuit in canonical form maintains its size and depth (see Observation 14 (pg. 10)). We will now answer objection (2), and in so doing, we will also address objections (3) and (4).

Furst, Saxe, and Sipser [FSS84] introduced the following very powerful idea: rather than trying to pick a certain restriction to use in converting each circuit, we will pick one *at random* from some probability distribution of restrictions. We will then show that the probability of being able to switch the roles of AND and OR gates and still end up with a "small" circuit (taking objection (3) into account) is high enough that objection (4) will also be answered.

The power of this idea cannot be overstated. To show that some restriction *exists* which will allow us to switch a circuit, we need only show that the probability that a random one does the job is non-zero! Unfortunately, we must ensure that the probability is high enough to overcome objection (4). However, we will see that proving a constant lower bound of 2/3 on the probability of switching success suffices.

3.2.5 Random Restrictions: The Distribution R_p

We now define the distribution R_p from which all random restrictions in the rest of this paper will be drawn. The distribution R_p has a parameter p ($0 \leq p \leq 1$) which affects the character of the restrictions drawn from it; the higher p is, the more likely a random restriction $\rho \in R_p$ is to map variables to X.

Definition 39 (R_p) *Picking a restriction ρ at random according to the distribution R_p means the following. For each of the variables $x_1, \ldots, x_n$, we will* independently *choose to map x_i to an element of $\{X, 0, 1\}$ with the following probability:*

$$\rho(x_i) = \begin{cases} X & \textit{with probability } p; \\ 0 & \textit{with probability } (1-p)/2; \\ 1 & \textit{with probability } (1-p)/2. \end{cases}$$

Notice that the chances of any given variable being mapped to 0 or 1 are equal. Also, since the probability that a variable remains indeterminate (i.e., gets mapped to X) is p, the expected number of variables remaining after applying a random restriction to a circuit with n inputs is pn.

For the rest of this paper, all probabilities $\Pr[\cdots]$ will be taken over the universe of random restrictions ρ, drawn with distribution R_p, unless stated otherwise. Let G be an AND-of-ORs. Informally, we say that a restriction ρ *fails* in switching G if there is no way to write $G\lceil\rho$ as a *small* OR-of-ANDs.[7] If ρ does not fail to switch G, we say ρ *succeeds* in switching G.

[7] Notice that it is always possible to express a function as an OR-of-ANDs using the disjunctive normal form circuit (see Observation 31 (pg. 15)). However, expressing an arbitrary function as a *small* OR-of-ANDs is another matter altogether. For example, there is no way to express PARITY as a small OR-of-ANDs (this will be shown in Claim 70 (pg. 43)).

3.2.6 Answering Objections (3) and (4)

Let us now summarize the proposed means of converting the family C^{k+1} into the family C^k. We convert the whole family by converting each of its depth-$k+1$ circuits C_n^{k+1} ($n = 1, 2, \ldots$) to a depth-k circuit. Now, to convert each of the depth-$k+1$ circuits, we first duplicate gates on level 1 as necessary to ensure that all gates on level 1 have fan-out 1. We then pick a restriction ρ at random according to R_p and apply it to each of the depth-2 sub-circuits on the bottom two levels of C_n^{k+1}. We hope to show that when each of these AND-of-ORs (or OR-of-ANDs) sub-circuits is hit by ρ, there is a very good chance that it can be rewritten as a small OR-of-ANDs (AND-of-ORs). In particular, there will be at least one restriction which succeeds in switching *all* of the depth-2 sub-circuits. This done, we can merge the two levels of adjacent like gate types[8] to produce the depth-k circuit $C_{n'}^k$ desired, where $n' = \textit{new-inputs}(n)$.

We must still deal with the very important issue of what is meant by "a very good chance [of switching success]." We fail to switch the entire circuit C_n^{k+1} if we fail to switch *any* of the sub-circuits rooted at a level-2 gate. That is, we fail overall if we fail on the first such sub-circuit or the second or the third and so on. Say we can bound the probability of failing to switch any depth-2 circuit in isolation. Then by Probability Lemma 35 (pg. 16), and since the probability of failure for each sub-circuit is bounded by the same value,

$$\Pr[\text{failure overall}] \leq (\#\text{ of depth-2 sub-circuits}) \times \Pr[\text{failure for a single depth-2 circuit}]. \quad (3.1)$$

There can be at most a small (i.e., sub-exponential) number of depth-2 sub-circuits to switch (since the number of depth-2 sub-circuits is the same as the number of gates on level 2, and is certainly limited by the size of the original circuit). For the overall probability of failure to be small (i.e., < 1), the probability that a random ρ fails to switch any single depth-2 circuit must therefore be *extremely* small, hopefully exponentially so.

The proof now reduces to the problem of bounding the probability that a random restriction fails to switch a depth-2 circuit in isolation. Håstad proves such a bound in a result he calls the *switching lemma*. We will defer its proof to Chapter 4. This lemma does the real work of the overall proof. Essentially, it gives an exponentially decreasing upper bound on the probability that a random restriction will *fail* to switch an arbitrary AND-of-ORs (a corollary shows that the bound on switching failure applies to OR-of-ANDs circuits as well). Such a bound implies, by the argument above, that the probability of failure is strictly less than 1 for sufficiently large n. Therefore, there is always at least one restriction which succeeds.

However, finding one restriction that succeeds is not enough to fully answer objections (3) and (4), because the restriction may not map enough variables to X. For example, we can always find a restriction that leaves only one variable behind, but then our circuit family will be finite. Therefore, we need to place a lower bound on the number of variables mapped to X (i.e., $\textit{new-inputs}(n)$).

One way to answer objections (3) and (4) would be as follows. We consider a function $\ell b : \mathcal{N} \to \mathcal{N}$ that will be a lower bound on $\textit{new-inputs}(n)$. To answer objections (3) and (4), we require that ℓb have the following properties:

1. $\forall n,\ \ell b(n) \leq n$,
2. $\ell b^{-1}(n) = \textit{poly}(n)$, and
3. $\lim_{n\to\infty} \ell b(n) = \infty$.

Property 1 is necessary because applying a restriction to a circuit cannot increase the number of inputs.

[8] Namely, levels 2 and 3.

Now, suppose we could show that, for all circuits G, there is some restriction ρ which succeeds in reducing G's depth by one *and* which maps *at least* $\ell b(n)$ of the inputs to X. Such a result would immediately answer both remaining objections! Here is why. Consider objection (3). If the size of the new circuit in terms of the original number of inputs is given by the sub-exponential function $size(n, k+1)$, then its size in terms of the new number of inputs and depth is at most $size(\ell b^{-1}(n), k)$. Property 2 implies that this new function of n and k is still sub-exponential. Now consider objection (4). Property 3 and the assumption $\ell b(n) \leq new\text{-}inputs(n)\ \forall n$ together imply that $\lim_{n\to\infty} new\text{-}inputs(n) = \infty$. By Observation 38 (pg. 20), this answers objection (4).

Therefore, we will have completed the proof if we can pick such a function ℓb and show not only that the probability of overall switching success is non-zero, but moreover, that it is large enough that at least one of the successful restrictions maps $\geq \ell b(n)$ of the inputs to X (for sufficiently large n).

Definition 40 ("Small" Restriction) *A restriction is* "small" *if it maps* $\geq \ell b(n)$ *of its inputs to* X. *Note that by definition,* ρ *is "small" iff* $|\rho| < n - \ell b(n)$.[9]

We can show that at least one of the successful restrictions is "small" by showing that it is impossible for *all* of the "small" restrictions to be failing. If the "small" restrictions were a subset of the failing ones, then the probability of failure would be at least as great as the probability of being "small". Therefore, if

$$\Pr[\text{failure overall}] < \Pr[\rho \text{ is "small"}], \tag{3.2}$$

the "small" restrictions cannot be a subset of the failing ones, and so at least one successful restriction must be "small". By equation (3.1) (pg. 22), equation (3.2) follows if we can show that

$$(\#\text{ of depth-2 sub-circuits}) \times \Pr[\text{failure for a single depth-2 circuit}] < \Pr[\rho \text{ is "small"}]. \tag{3.3}$$

Notice that the right-hand side of this inequality can be simplified:

$$\begin{aligned}
\Pr[\rho \text{ is "small"}] &= \Pr[\rho \text{ maps } \geq \ell b(n) \text{ variables to } X] \\
&= \sum_{k=\ell b(n)}^{n} \Pr[|\rho| = n-k] \\
&= \sum_{k=\ell b(n)}^{n} \binom{n}{k} p^k (1-p)^{n-k}.
\end{aligned} \tag{3.4}$$

In the actual proof, we will choose p to be a slowly monotone decreasing function of n, such as $p = n^{-1/10}$. Therefore, as the size of circuits in the circuit family grows, the chance that a random ρ will map a variable to X will decrease slowly. We will then choose $\ell b(n)$ to be the expected number of variables remaining after applying ρ — namely, pn — so $\ell b(n) = pn = n^{9/10}$. Notice that this ℓb satisfies the three requirements listed on pg. 22.

Since $\ell b(n)$ is the expected number of variables remaining, the sum of equation (3.4) is the "area" under the tail of a binomial distribution curve with probability p starting at the point where the curve achieves its maximum. Proving inequality (3.3) is a matter of approximating this distribution and bounding it from below. We defer the actual statistical analysis until the proof of

[9] "Small" is in quotes because our choice for $\ell b(n)$ in the actual proof will imply that these "small" restrictions are actually quite large with respect to n. In fact, for any constant fraction $c < 1$, there is a "small" restriction with size $> cn$ for sufficiently large n.

Theorem 68 (pg. 42). For now, suffice it to say that the sum (3.4) approaches $1/2$ as $n \to \infty$. Using the result of the switching lemma, we will show that the probability of overall switching failure is $\leq 1/3$ for sufficiently large n, so equation (3.2) will be true for sufficiently large n.

This completes the proof overview. A final observation may be in order regarding the fact that the result holds only for sufficiently large n. Remember that all our definitions were phrased in terms of *asymptotic* complexity! Keep in mind that any finite set of circuits can be shown to obey an arbitrary size (or depth) bound just by using a large enough constant. What concerns us is how circuit *families* of infinite size behave for large n. Therefore, so long as our results pertain to circuits with sufficiently many inputs, they reflect the true requirements (in the case of lower bounds proofs) of circuits computing PARITY.

3.3 A Proof Road Map

Figures 3.3 (pg. 25) and 3.4 (pg. 26) present a "road map" of the proof. In these diagrams, an arrow from one box to another means the former is used in the proof of the latter. Wherever possible, the main result of a lemma or theorem has been given in quotes.

These diagrams are intended solely for your reference while you are following the proof. They will certainly look confusing to you now. Don't worry! The notation will make sense to you as you get deeper and deeper into the proof; you will probably want to keep checking your position in the proof against these diagrams so you don't lose the forest for the trees, so to speak.

Figure 3.3 shows the proof tree for the Stronger Switching Lemma, and corresponds roughly to the proofs of Chapter 4. Figure 3.4 has the Stronger Switching Lemma at its base and builds up from there; this proof tree corresponds roughly to the results of Chapter 5.

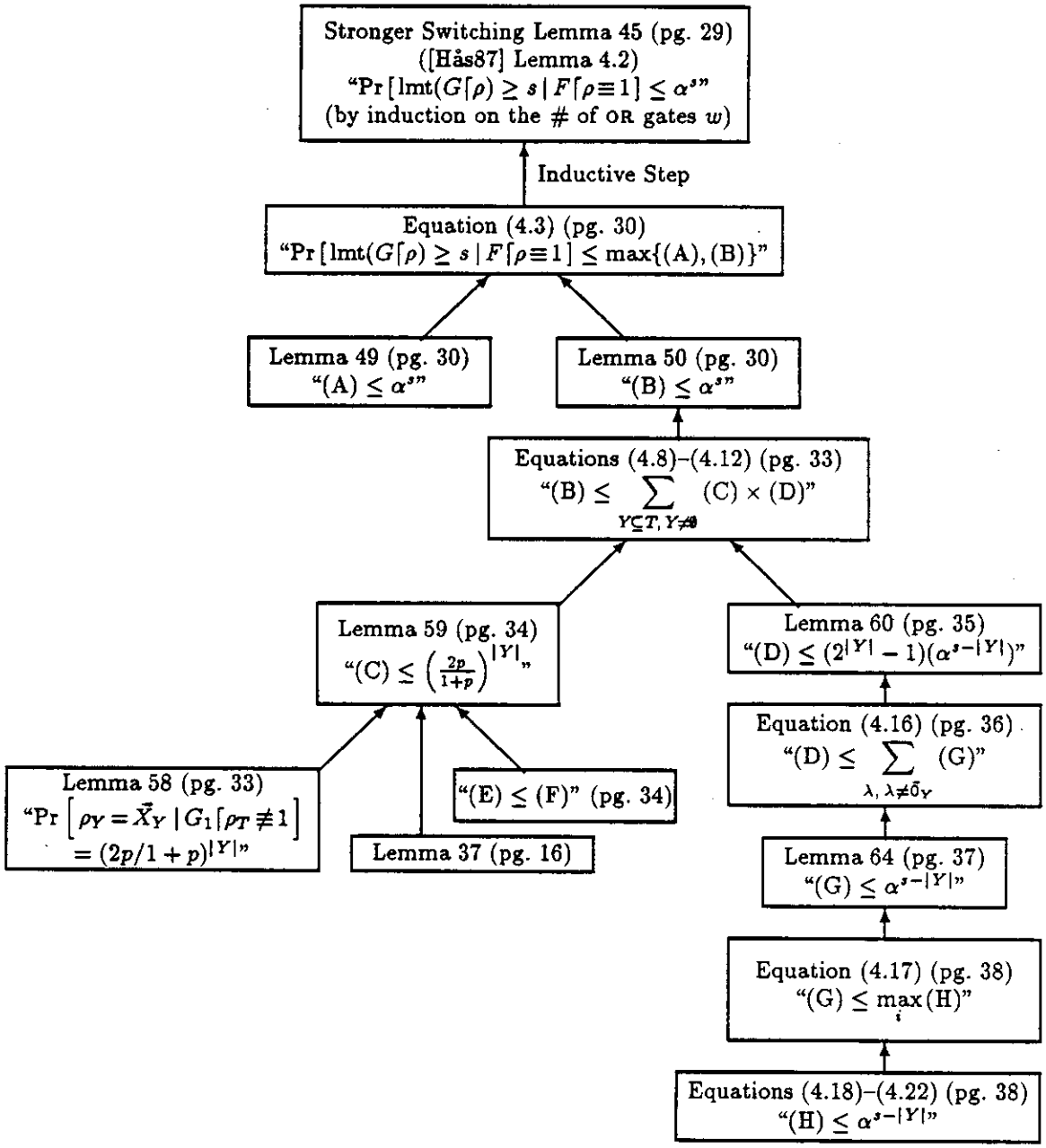


Figure 3.3: A Road Map of the Proof's Lower Level.

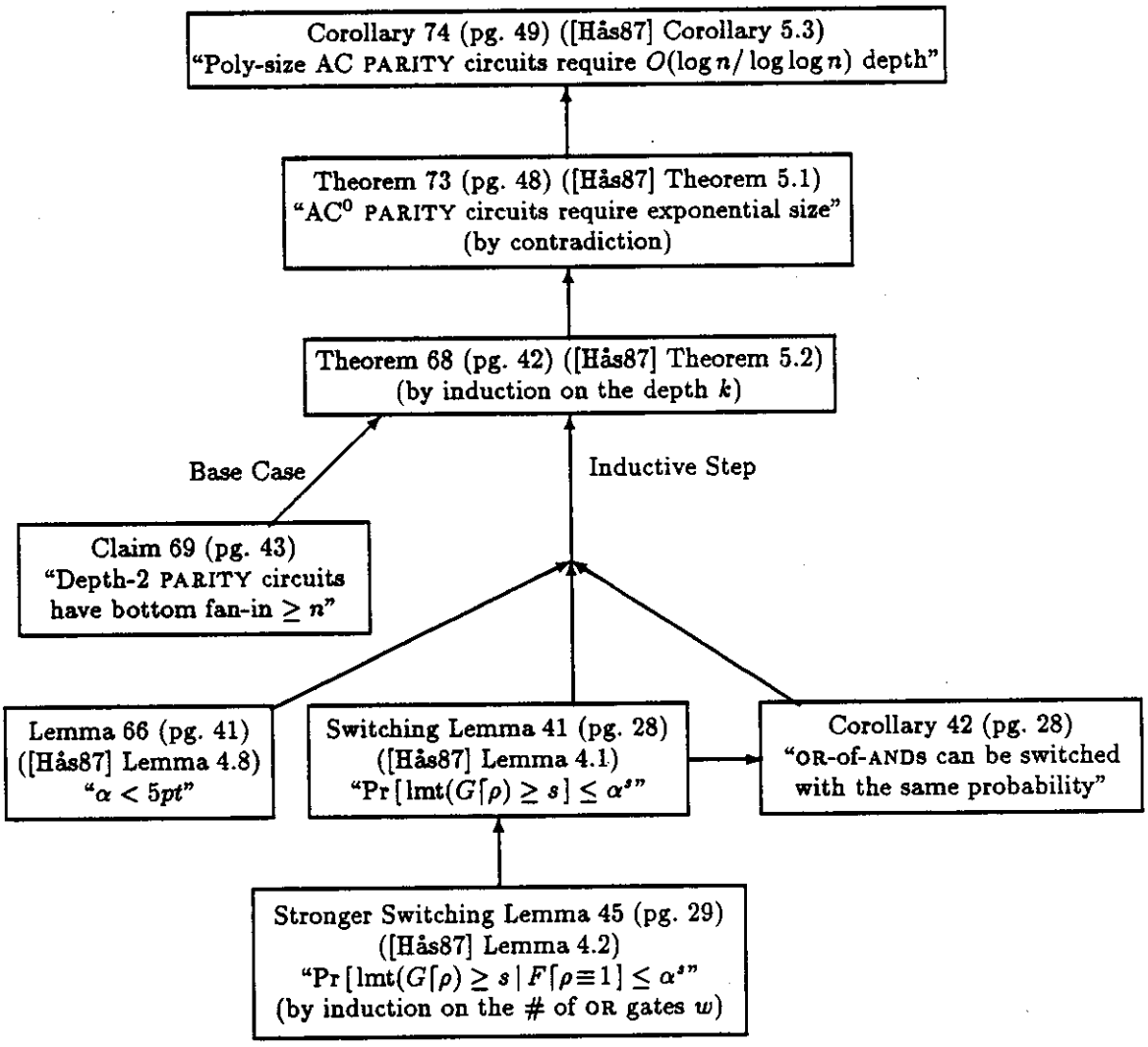


Figure 3.4: A Road Map of the Proof's Upper Level.

Chapter 4

The Switching Lemma

We now state and prove the switching lemma, which, loosely speaking, allows us to "switch" the roles of AND and OR gates on the bottom two levels of a circuit in canonical form. For the rest of this chapter, we will be working with a depth-2, AND-of-ORs G, as shown in Figure 4.1. Let G have w OR sub-circuits denoted by $G_1, \ldots, G_w$, each of fan-in at most t, so G as a whole has bottom fan-in $\leq t$.

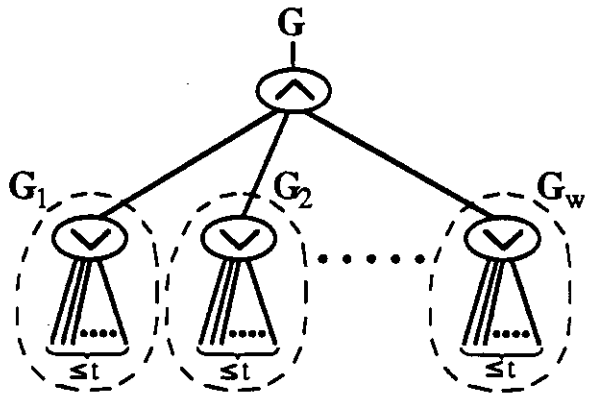


Figure 4.1: The AND-of-ORs Circuit G.

Say we have chosen a certain value for p (later, we will pick an exact value for p so that our bounds on success are high enough to make the result work). We then randomly choose a restriction ρ according to R_p. Consider the circuit $G\lceil\rho$. We would like to switch the roles of OR's and AND's in $G\lceil\rho$ and express $G\lceil\rho$ as an OR of small ANDs. By a *small gate* we mean that the gate has fan-in $< s$ for some value $s > 0$ (also to be chosen later). We also say a minterm is *small* if it has size $< s$. Informally, the switching lemma states that the probability of not being able to express $G\lceil\rho$ as an OR of small ANDs is an exponentially decreasing function of s (i.e., like 2^{-s}).

We know by Observation 31 (pg. 15) that if each of $G\lceil\rho$'s minterms is small (i.e., if $\text{lmt}(G\lceil\rho) < s$), then $G\lceil\rho$ *can* be expressed as an OR of small ANDs. Therefore, the only times a random restriction ρ may *not*[1] succeed in letting us express $G\lceil\rho$ as an OR of small ANDs is if $\text{lmt}(G\lceil\rho) \geq s$. The switching lemma tells us that the probability of this happening, for suitable values of p and t,

[1]Notice we say "may not" instead of "does not" because Observation 33 (pg. 16) showed that it is possible to write $G\lceil\rho$ as an OR of small ANDs even if $\text{lmt}(G\lceil\rho) \geq s$. Therefore,

$$\Pr[G\lceil\rho \text{ cannot be written as an OR of small ANDs}] \leq \Pr[\text{lmt}(G\lceil\rho) \geq s].$$

is an exponentially small function of s. The corollary to the switching lemma then shows that we also fail to switch an OR-of-ANDs with the same exponentially small probability.

4.1 Statement and Corollary of the Switching Lemma

Lemma 41 (Switching Lemma — [Hås87] Lemma 4.1) *Let $G = \bigwedge_{i=1}^{w} G_i$, where each G_i is an* OR *of fan-in $\leq t$. Let ρ be a restriction chosen randomly according to R_p. Then for any $s > 0$, we bound the chance of failure of being able to write $G\lceil\rho$ as an* OR *of small* ANDs *by:*

$$\Pr[\mathrm{lmt}(G\lceil\rho) \geq s] \leq \alpha^s,$$

where $\alpha\,(< 1)$ is the unique positive root to the equation

$$\left(1 + \left(\frac{4p}{1+p}\right)\left(\frac{1}{\alpha}\right)\right)^t = \left(1 + \left(\frac{2p}{1+p}\right)\left(\frac{1}{\alpha}\right)\right)^t + 1. \tag{4.1}$$

We state parenthetically that $\alpha < 1$; later we will be able to bound α by $1/2$ for suitable values of p and t. In any case, the important point is that the probability of failing to be able to switch the circuit is an exponentially decreasing function of s (for fixed p and t).

Corollary 42 *If G is an* OR*-of-*ANDs*, where each* AND *is of size $\leq t$, then we fail to be able to write $G\lceil\rho$ as an* AND*-of-*ORs*, where each* OR *is of size $< s$, with the same probability as that given in Lemma 41.*

Proof We will again use Observation 14 (pg. 10), which states that taking the complement of a circuit in canonical form simply swaps all the AND and OR gates, and complements the input literals. We first make the following two observations:

Observation 43 ($\Pr[\rho] = \Pr[\overline{\rho}]$) Let $\overline{\rho}$ denote the complement of ρ; i.e., $\overline{\rho}$ is the same as ρ except that it maps to 1 whatever ρ maps to 0, and vice-versa. Then $\forall\rho$, ρ and $\overline{\rho}$ have an equal probability of occurring in the distribution R_p.

Observation 44 For any circuit C and any fixed restriction ρ, $\neg(C\lceil\rho) \equiv (\neg C)\lceil\overline{\rho}$. This is true because taking the complement of C complements its literals, so applying the complement restriction $\overline{\rho}$ after complementing C yields the same result $\neg(C\lceil\rho)$. Notice that if one circuit is an AND-of-ORs, the other will be too.

Taken together, these two observations imply that the order in which we apply a restriction to a circuit and take its complement is irrelevant. That is,

$$\forall C, C' : \Pr[\neg(C\lceil\rho) \equiv C'] = \Pr[(\neg C)\lceil\rho \equiv C']. \tag{4.2}$$

If we complement G, making it an AND-of-ORs, apply the switching lemma to it, and then complement the result to "undo" the effect of the first complement operation, we fail in switching $\neg G$ with the same probability as that given in Lemma 41. However, Equation (4.2) tells us that the process described is equivalent to *first* applying a random restriction ρ, and *then* taking the complement before applying the switching lemma and complementing a second time. Hence, we fail in switching G with the required probability. ■

We will not prove the switching lemma because it turns out to be easier to prove a stronger version of it. The stronger version is identical except that it conditions on an arbitrary Boolean function F being forced to 1 by ρ. Conditioning upon this event facilitates the proof.

4.2 The Stronger Switching Lemma

Lemma 45 (Stronger Switching Lemma — [Hås87] Lemma 4.2) *Let $G = \bigwedge_{i=1}^{w} G_i$, where each G_i is an* OR *of fan-in $\leq t$. Let F be an arbitrary Boolean function, and let ρ be a restriction chosen randomly according to R_p. Then for any $s > 0$,*

$$\Pr[\mathrm{lmt}(G\lceil\rho) \geq s \mid F\lceil\rho \equiv 1] \leq \alpha^s$$

where $\alpha(<1)$ is the unique positive root to equation (4.1) (pg. 28). If there is no restriction ρ such that $F\lceil\rho \equiv 1$ (e.g., if $F \equiv 0$), then we use the convention that the conditional probability is 0.

Corollary 46 *The stronger switching lemma implies the switching lemma.*

Proof We simply choose $F \equiv 1$. Since $1\lceil\rho \equiv 1$, the probability in this case is conditioned over *all* ρ, just as it is in the regular switching lemma. ■

We will need to make statements about the actual values of inputs reaching the gate G_1. Without loss of generality, we will assume that all inputs to the circuit are not complemented. We do not lose generality for two reasons. First, R_p has been defined such that variables get mapped to 0 or 1 with equal probability. Second, no single AND or OR gate has both a variable and its complement as inputs, because if it did, the gate could then be replaced by 0 or 1, respectively.

Notation 47 (T, N', T') *Recall that we denote the set of all input variables by N. Let T denote the set of inputs to G_1. Furthermore, let N' and T' be the subsets of N and T, respectively, mapped to X by ρ. Therefore, minterms of $G\lceil\rho$ apply to the variables in N', and minterms of $G_1\lceil\rho_T$ apply to the variables in T'. See Figure 4.2 for a pictorial representation of these sets.*

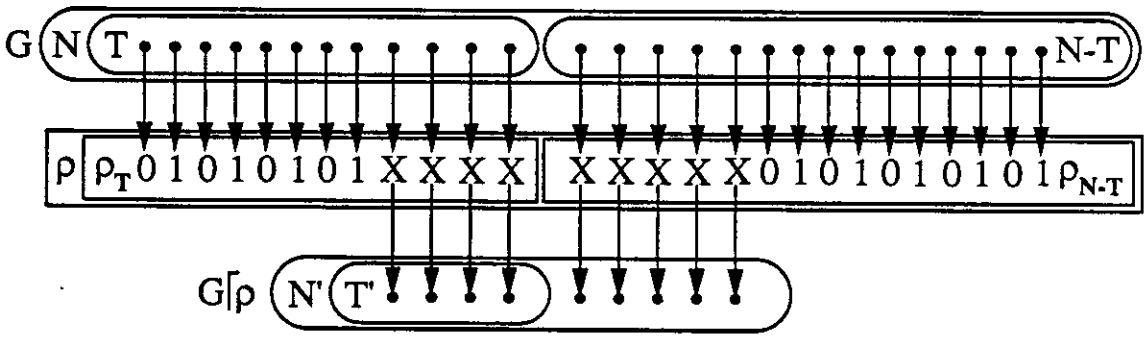


Figure 4.2: Applying a Typical Restriction to the Circuit G.

Example 48 Say $N = \{x_1, x_2, \ldots, x_7\}$ and the inputs to G_1 are $T = \{x_1, x_3, x_5, x_7\}$. If we choose $\rho = 0XX10XX$, then $N' = \{x_2, x_3, x_6, x_7\}$ and $T' = \{x_3, x_7\}$. If, on the other hand, we choose $\rho = 1X101X1$, then $\rho_T = \vec{1}_T$.

Proof of Stronger Switching Lemma (by induction on w)

Base Case ($w = 0$) In this case, $G \equiv 1$ (recall from Observation 16 (pg. 11) that an AND with no inputs is trivially true), so the probability is in fact 0, and the result follows immediately.

Inductive Step We will consider the induction hypothesis to hold for the circuit $G' = \bigwedge_{i=2}^{w} G_i$, so consider the leftmost OR circuit G_1. We first partition the universe of all 3^n possible restrictions into two sets:

$$A_1 = \{\, \rho \mid G_1\lceil\rho_T \equiv 1 \,\} \text{ and } A_2 = \{\, \rho \mid G_1\lceil\rho_T \not\equiv 1 \,\}.$$

By Lemma 34 (pg. 16), it follows that

$$\Pr[\,\mathrm{lmt}(G\lceil\rho) \geq s \mid F\lceil\rho \equiv 1\,] \leq \max\{\overbrace{\Pr[\,\mathrm{lmt}(G\lceil\rho) \geq s \mid F\lceil\rho \equiv 1 \wedge G_1\lceil\rho_T \equiv 1\,]}^{\text{(A)}}, \underbrace{\Pr[\,\mathrm{lmt}(G\lceil\rho) \geq s \mid F\lceil\rho \equiv 1 \wedge G_1\lceil\rho_T \not\equiv 1\,]}_{\text{(B)}}\} \tag{4.3}$$

To prove the Lemma, we must therefore show that (A) $\leq \alpha^s$ and (B) $\leq \alpha^s$. We now consider these two probabilities separately.

Lemma 49 (A) $\leq \alpha^s$.

Proof We are conditioning on $G_1\lceil\rho_T \equiv 1$. Since G has an AND gate at its top level, we know that $G_1\lceil\rho_T \equiv 1 \implies G \equiv G'$. Therefore, (A) $\leq \alpha^s$ by the induction hypothesis. ∎

Lemma 50 (B) $\leq \alpha^s$.

Proof Since the rest of the proof is concerned with establishing this case, it will be helpful to characterize the restrictions we are conditioning on. We start with the following two observations.

Observation 51 Recall that G_1 is simply an OR gate. Hence,

$$G_1\lceil\rho_T \not\equiv 1 \iff \forall x_i \in T, (\rho(x_i) = 0 \vee \rho(x_i) = X)\,. \tag{4.4}$$

Within this case, there is the special subcase of $\rho_T = \vec{0}_T$. This case is special because it implies that $G\lceil\rho$ has no minterms! Here is the reason why:

$$(\rho_T = \vec{0}_T) \implies (G_1\lceil\rho_T \equiv 0) \implies (G\lceil\rho \equiv 0).$$

The first implication follows from the fact that an OR with all 0 inputs is forced to 0; the second implication follows because G has an AND gate at its top level. Furthermore, given the initial assumption that $G_1\lceil\rho_T \not\equiv 1$, we know that

$$\rho_T = \vec{0}_T \iff T' = \emptyset, \tag{4.5}$$

since ρ_T does not map any variables to X. At some point, we will have to deal with the special subcase $\rho_T = \vec{0}_T$ separately.

Observation 52 By definition, every minterm of $G\lceil\rho$ is a mapping only from the variables in N'. Since $G\lceil\rho$ is an AND-of-ORs, any minterm $\sigma \in \Gamma(G\lceil\rho)$ must make every $G_i\lceil\rho$ true. In particular, we have

$$\forall \sigma \in \Gamma(G\lceil\rho), (G_1\lceil\rho_T)\lceil\sigma_{T'} \equiv 1. \tag{4.6}$$

Furthermore, since $G_1\lceil\rho_T$ is an OR circuit, equation (4.6) implies that every minterm maps at least one variable in T' to 1:

$$\forall \sigma \in \Gamma(G\lceil\rho), \exists\, x_i \in T' : \sigma_{T'}(x_i) = 1. \tag{4.7}$$

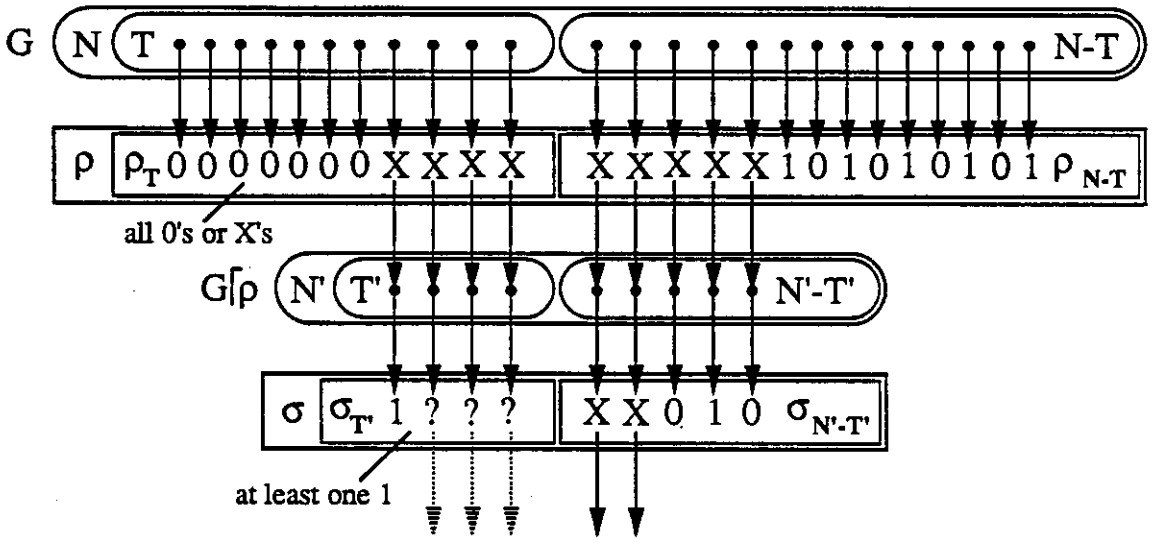


Figure 4.3: Pictorial Description of ρ and σ for Lemma 50.

Taken together, equation (4.4) and equation (4.7) can be represented graphically, as shown in Figure 4.3. This representation shows us that ρ_T maps the variables T only to 0 or X, whereas ρ_{N-T} maps the variables $N - T$ to either 0, 1, or X. It also shows us that $\sigma_{T'}$ maps *at least one* of the variables in T' to 1.

We next partition the minterms in $\Gamma(G\lceil\rho)$ by the subset of T' they map to values. More specifically, we partition $\Gamma(G\lceil\rho)$ by the equivalence relation $\sim$:

$$\sigma \sim \tau \iff \forall x_i \in T', (\sigma(x_i) \neq X \iff \tau(x_i) \neq X).$$

Two minterms are in the same equivalence class if and only if they map identical subsets of T' to values. We will represent an equivalence class by that subset Y of T' which all minterms in the class map to values.

Normally, we will imagine that $Y \subseteq T'$, which is equivalent to saying that ρ maps all variables of Y to X (symbolically, $\rho_Y = \vec{X}_Y$). However, for technical reasons, we will need to let $Y \subseteq T$. We will consider the case $Y \not\subseteq T'$ to be degenerate, but we will have to take it into account.

Example 53 Let $n = 7$, so $N = \{x_1, x_2, \ldots, x_7\}$. Furthermore, let $T = \{x_1, x_2, x_3, x_4\}$. Say we choose $\rho = X0XX1XX$. Then $N' = \{x_1, x_3, x_4, x_6, x_7\}$ and $T' = \{x_1, x_3, x_4\}$. Suppose further that the rows of the following table represent the minterms of $G\lceil\rho$:

$\sigma_{T'}$			$\sigma_{N'-T'}$	
x_1	x_3	x_4	x_6	x_7
X	0	1	X	1
X	1	0	X	X
1	X	1	1	0
1	X	X	0	1
0	X	X	X	0
X	X	1	1	0

Horizontal lines have been drawn dividing the minterms into their respective equivalence classes according to the $\sim$ relation. Reading from the top down, the equivalence classes are represented by the sets $Y = \{x_3, x_4\}$, $\{x_1, x_4\}$, $\{x_1\}$, and $\{x_4\}$.

Notation 54 ($\mathrm{lmt}^Y(G\lceil\rho)$) *Let $Y \subseteq T$. We write $\mathrm{lmt}^Y(G\lceil\rho)$ to denote the size of the largest minterm $\sigma \in \Gamma(G\lceil\rho)$ such that $\sigma_{T'}$ assigns values to* exactly *the variables in Y (i.e., the size of the largest minterm in the equivalence class represented by Y). If there are such minterms, it is clear that $\mathrm{lmt}^Y(G\lceil\rho) \geq |Y|$, since these minterms map at least the variables in Y to values. If there are no such minterms, or if $Y \not\subseteq T'$ (i.e., ρ does not map all variables of Y to X), we define*[2] $\mathrm{lmt}^Y(G\lceil\rho) = 0$. *The pictorial representation of $\sigma_{T'}$ mapping exactly the variables in Y to values is shown in Figure 4.4.*

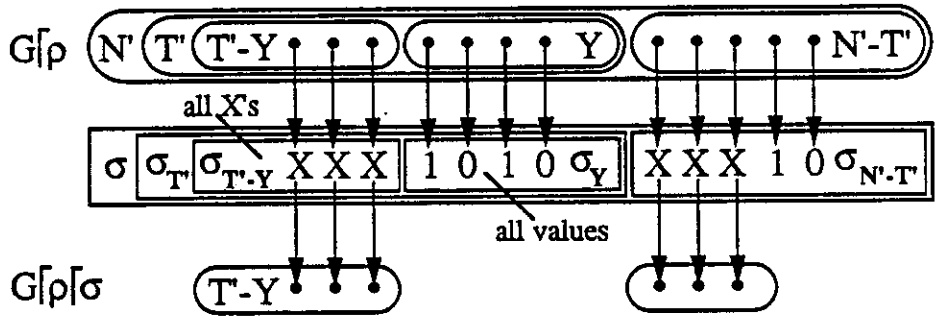


Figure 4.4: The Minterm $\sigma_{T'}$ Assigns Values to Exactly the Variables Y.

Example 55 If we consider the minterms shown on page 31, we see that the first minterm listed in each equivalence class is the largest. Therefore, $\mathrm{lmt}^{\{x_3,x_4\}}(G\lceil\rho) = 3$, $\mathrm{lmt}^{\{x_1,x_4\}}(G\lceil\rho) = 4$, $\mathrm{lmt}^{\{x_1\}}(G\lceil\rho) = 3$, and $\mathrm{lmt}^{\{x_4\}}(G\lceil\rho) = 3$. As examples of the special cases, $\mathrm{lmt}^{\{x_3\}}(G\lceil\rho) = 0$ and $\mathrm{lmt}^{\{x_2,x_3\}}(G\lceil\rho) = 0$.

Let us make use of the new notation to summarize our knowledge to this point.

Claim 56 $(Y = \emptyset) \Longrightarrow \Pr\left[\mathrm{lmt}^Y(G\lceil\rho) \geq s \mid G_1\lceil\rho_T \not\equiv 1\right] = 0$

Proof Assume $Y = \emptyset$. There are two cases to consider: $\Gamma(G\lceil\rho) = \emptyset$ (i.e., $\rho_T = \vec{0}_T$) and $\Gamma(G\lceil\rho) \neq \emptyset$. In the former case, $\mathrm{lmt}^Y(G\lceil\rho) = 0$ by definition, so the claim follows from the fact that $s > 0$. The latter case follows from equation (4.7) (pg. 30), since every minterm (regardless of its size) must map at least one variable in T' to 1, so Y would have to be non-empty in this case. Since $Y = \emptyset$, there is no ρ satisfying the event, so the probability is zero. ■

Claim 57 *For all ρ such that $G_1\lceil\rho_T \not\equiv 1$,* $(\mathrm{lmt}(G\lceil\rho) \geq s) \iff \bigvee_{Y \subseteq T, Y \neq \emptyset} (\mathrm{lmt}^Y(G\lceil\rho) \geq s)$.

Proof For the forward direction, we note that if a minterm of size $\geq s$ exists for $G\lceil\rho$, it must assign values to some $Y \subseteq T' \subseteq T$. Moreover, by Claim 56 above, we know that $Y \neq \emptyset$. For the reverse direction, we know at least one $Y \subseteq T$, $Y \neq \emptyset$ makes $\mathrm{lmt}(G\lceil\rho) \geq s$ true. As an aside, we know in fact that $Y \subseteq T'$, since otherwise $\mathrm{lmt}^Y(G\lceil\rho) = 0 < s$. In any event, there is some minterm of size $\geq s$, which is all we need to establish the reverse direction. ■

[2] As in Notation 32 (pg. 15), this definition is made solely for the sake of making the event $\mathrm{lmt}(G\lceil\rho) \geq s$ false when $\Gamma(G\lceil\rho) = \emptyset$ or $Y \not\subseteq T'$.

We are now ready to make some progress toward bounding the difficult case (B) of equation (4.3) (pg. 30).

$$
\begin{aligned}
(\mathrm{B}) &= \Pr\left[\mathrm{lmt}(G\lceil\rho) \geq s \mid F\lceil\rho \equiv 1 \wedge G_1\lceil\rho_T \not\equiv 1\right] && (4.8)\\
&= \Pr\left[\bigvee_{Y\subseteq T, Y\neq\emptyset} \left(\mathrm{lmt}^Y(G\lceil\rho) \geq s\right) \mid F\lceil\rho \equiv 1 \wedge G_1\lceil\rho_T \not\equiv 1\right] && (4.9)\\
&\leq \sum_{Y\subseteq T, Y\neq\emptyset} \Pr\left[\mathrm{lmt}^Y(G\lceil\rho) \geq s \mid F\lceil\rho \equiv 1 \wedge G_1\lceil\rho_T \not\equiv 1\right] && (4.10)\\
&= \sum_{Y\subseteq T, Y\neq\emptyset} \Pr\left[\mathrm{lmt}^Y(G\lceil\rho) \geq s \wedge \rho_Y = \vec{X}_Y \mid F\lceil\rho \equiv 1 \wedge G_1\lceil\rho_T \not\equiv 1\right] && (4.11)\\
&= \sum_{Y\subseteq T, Y\neq\emptyset} \overbrace{\Pr\left[\rho_Y = \vec{X}_Y \mid F\lceil\rho \equiv 1 \wedge G_1\lceil\rho_T \not\equiv 1\right]}^{(\mathrm{C})} \times && (4.12)\\
&\qquad \underbrace{\Pr\left[\mathrm{lmt}^Y(G\lceil\rho) \geq s \mid F\lceil\rho \equiv 1 \wedge G_1\lceil\rho_T \not\equiv 1 \wedge \rho_Y = \vec{X}_Y\right]}_{(\mathrm{D})}.
\end{aligned}
$$

We now explain each of the equalities and inequalities in this chain. (4.8) = (4.9) by Claim 57 (pg. 32) above. (4.9) $\leq$ (4.10) follows by Lemma 35 (pg. 16). We reason that (4.10) = (4.11) as follows. By assumption, $s > 0$, so $\mathrm{lmt}^Y(G\lceil\rho) > 0$. But by definition of $\mathrm{lmt}^Y(G\lceil\rho)$, $\mathrm{lmt}^Y(G\lceil\rho)$ is non-zero only if ρ maps all variables Y to X, so $\mathrm{lmt}(G\lceil\rho) \geq s \implies \rho_Y = \vec{X}_Y$. Finally, (4.11) = (4.12) by Lemma 36 (pg. 16).

We will now analyze each of the factors (C) and (D) in that expression separately. Notice that the set Y is *fixed* when we evaluate (C) or (D) in isolation. In Lemma 58, we will bound probability (C) without conditioning on the fact that $F\lceil\rho \equiv 1$. In Lemma 59 (pg. 34), we will then bound probability (C) in the general case. Finally, we will bound probability (D) in Lemma 60 (pg. 35).

4.2.1 Bounding the Probability (C)

Lemma 58 ([Hås87] Lemma 4.3) *For any $Y \subseteq T$, $Y \neq \emptyset$, $\Pr\left[\rho_Y = \vec{X}_Y \mid G_1\lceil\rho_T \not\equiv 1\right] = \left(\frac{2p}{1+p}\right)^{|Y|}$.*

Proof By equation (4.4) (pg. 30), we know that $\forall x_i \in T$, $(\rho_T(x_i) = 0 \vee \rho_T(x_i) = X)$. Therefore:

$$
\begin{aligned}
\Pr\left[\rho_Y = \vec{X}_Y \mid G_1\lceil\rho_T \not\equiv 1\right] &= \left(\frac{\Pr[\rho_T(x_i) = X]}{\Pr[\rho_T(x_i) = 0 \vee \rho_T(x_i) = X]}\right)^{|Y|}\\
&= \left(\frac{p}{\left(\frac{1-p}{2} + p\right)}\right)^{|Y|}\\
&= \left(\frac{p}{\left(\frac{1+p}{2}\right)}\right)^{|Y|}\\
&= \left(\frac{2p}{1+p}\right)^{|Y|}.
\end{aligned}
$$

■

Lemma 59 ([Hås87] Lemma 4.5) *For any $Y \subseteq T$, $Y \neq \emptyset$,* $(C) \leq \left(\frac{2p}{1+p}\right)^{|Y|}$.

Proof By Lemma 58, we need to show that

$$(C) = \Pr\left[\rho_Y = \vec{X}_Y \mid F\lceil\rho \equiv 1 \wedge G_1\lceil\rho_T \not\equiv 1\right] \leq \Pr\left[\rho_Y = \vec{X}_Y \mid G_1\lceil\rho_T \not\equiv 1\right] \tag{4.13}$$

We use probability Lemma 37 (pg. 16). In this lemma, if we replace "A" by "$\rho_Y = \vec{X}_Y$", "B" by "$F\lceil\rho \equiv 1$", and "C" by "$G_1\lceil\rho_T \not\equiv 1$", then equation (4.13) is true if and only if

$$\underbrace{\Pr\left[F\lceil\rho \equiv 1 \mid \rho_Y = \vec{X}_Y \wedge G_1\lceil\rho_T \not\equiv 1\right]}_{(E)} \leq \underbrace{\Pr[F\lceil\rho \equiv 1 \mid G_1\lceil\rho_T \not\equiv 1]}_{(F)}$$

The intuitive idea used to prove this inequality is that forcing more inputs to be X can only *decrease* the probability that F is forced (to 1). The argument is formalized as follows. Let R'_p be the probability distribution R_p conditioned on $G_1\lceil\rho_T \not\equiv 1$; by equation (4.4) (pg. 30), the only restrictions in this space are those that map the variables T to 0 or X. We build a table of all possible ρ_Y versus ρ_{N-Y} values (see Figure 4.5 below). The table thus has an entry for every restriction in R'_p. We label the rows with all ways ρ_Y can map the variables Y, and we label the columns with all ways ρ_{N-Y} can map the variables $N - Y$. Note that $N - Y$ may contain some variables in T, and ρ must map these to 0 or X only.

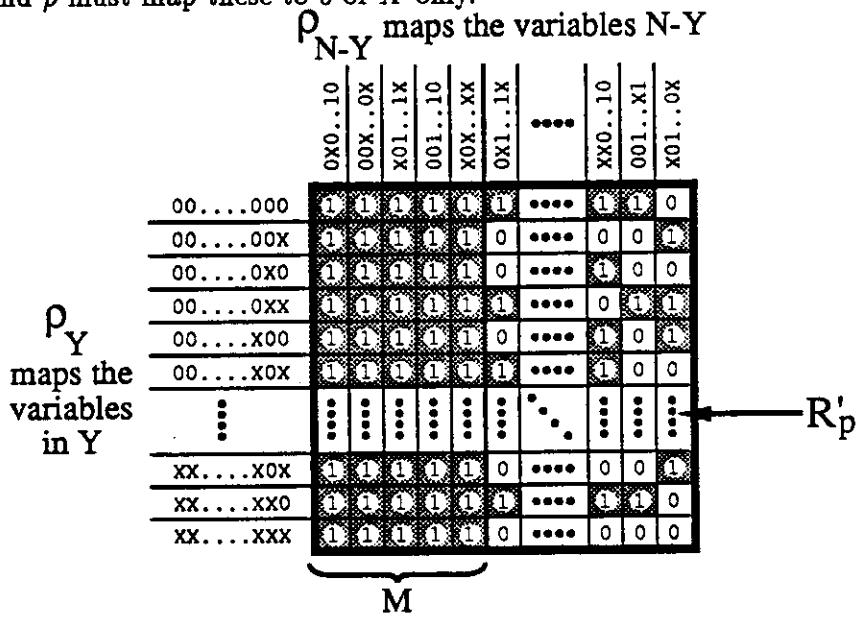


Figure 4.5: Table of Restrictions Used to Prove Lemma 59.

For each entry (i.e., for each restriction ρ), we write 0 or 1 in the table if $F\lceil\rho \not\equiv 1$ or $F\lceil\rho \equiv 1$, respectively. We randomly pick a restriction ρ according to the distribution R'_p by first picking a row according to R'_p, and then, *independently* picking a column according to R'_p. Let M be the set of columns with a 1 in row "$XX\cdots X$". The table has been drawn with all the columns of M grouped at the left.

Call the (i,j)'th entry of the table $a[i,j]$. Each row i and each column j have some partial probability of occurring; call these probabilities r_i and c_j respectively. Say q is the sum of the

probabilities of the columns of M, i.e., $q = \sum_{j \in M} c_j$. Now, (E) is the probability of a 1 occurring in the bottom row of the table, which is exactly the probability q. The probability (F), on the other hand, is the sum of the probabilities $r_i c_j$ for each entry (i,j) such that $a[i,j]=1$.

The key observation to be made is that any column with a 1 in the "$XX\cdots X$" row must have 1's in the *entire* column, because if $F\lceil\rho \equiv 1$ when $\rho_Y = \vec{X}_Y$, then it must also be true that $F\lceil\rho \equiv 1$ even if ρ_Y assigns 0 to some variables. Therefore, the columns of M are all 1's. This means that all the 1's in the columns of M make (F) at least as big as (E), and if any other 1's occur in columns not in M, then (F) is strictly bigger than (E). Symbolically:

$$\begin{aligned}
(\mathrm{F}) &= \sum_{a[i,j]=1} r_i c_j \\
&= \sum_{j \in M} \left[c_j \left(\sum_i r_i \right) \right] + \left(\sum_{j \notin M \wedge a[i,j]=1} r_i c_j \right) \\
&= \left(\sum_{j \in M} c_j \cdot 1 \right) + \left(\sum_{j \notin M \wedge a[i,j]=1} r_i c_j \right) \\
&= (\mathrm{E}) + \left(\sum_{j \notin M \wedge a[i,j]=1} r_i c_j \right) \\
&\geq (\mathrm{E}).
\end{aligned}$$

By showing (E) $\leq$ (F), we have proven equation (4.13) (pg. 34). ∎

4.2.2 Bounding the Probability (D)

Lemma 60 *The probability* (D) *introduced in equation (4.12) (pg. 33) is bounded as follows:*

$$(\mathrm{D}) = \Pr\left[\mathrm{lmt}^Y(G\lceil\rho) \geq s \mid F\lceil\rho \equiv 1 \wedge G_1\lceil\rho_T \not\equiv 1 \wedge \rho_Y = \vec{X}_Y \right] \leq (2^{|Y|}-1)(\alpha^{s-|Y|}).$$

Proof We are now conditioning on $\rho_Y = \vec{X}_Y$, and the probability is satisfied only by those restrictions ρ such that some minterm σ has size $\geq s$ and such that $\sigma_{T'}$ assigns values *exactly* to the variables in Y. This means that $\sigma_{T'-Y} = \vec{X}_{T'-Y}$. As before, it helps to have a picture describing ρ and σ in this branch of the proof, and it is given in Figure 4.6.

Notation 61 $(\lambda, \mathrm{lmt}^{Y,\lambda}(G\lceil\rho))$ *Let λ be an assignment on the variables of Y, i.e., $\lambda : Y \to \{0,1\}$. We write $\mathrm{lmt}^{Y,\lambda}(G\lceil\rho)$ to denote the size of the largest minterm $\sigma \in \Gamma(G\lceil\rho)$ such that: 1) σ_T assigns values to exactly the variables of Y (as shown in Figure 4.6) and 2) $\sigma_Y = \lambda$.*

We now summarize our knowledge using this new notation. Just as we split the probability before by considering all cases $Y \subseteq T$, $Y \neq \emptyset$, we now split the probability by considering all cases $\lambda : Y \to \{0,1\}$, $\lambda \neq \vec{0}_Y$. The following two claims are just slight variations (in spirit) of Claims 56 and 57 on page 32.

Claim 62 $(\lambda = \vec{0}_Y) \Longrightarrow \Pr\left[\mathrm{lmt}^{Y,\lambda}(G\lceil\rho) \geq s \mid G_1\lceil\rho_T \not\equiv 1 \wedge \rho_Y = \vec{X}_Y \right] = 0.$

Proof (by contradiction) Assume $\lambda = \vec{0}_Y$, and that the probability is non-zero. Then there is a restriction ρ and a minterm σ of $G\lceil\rho$ which have the properties exhibited in Figure 4.6, with the

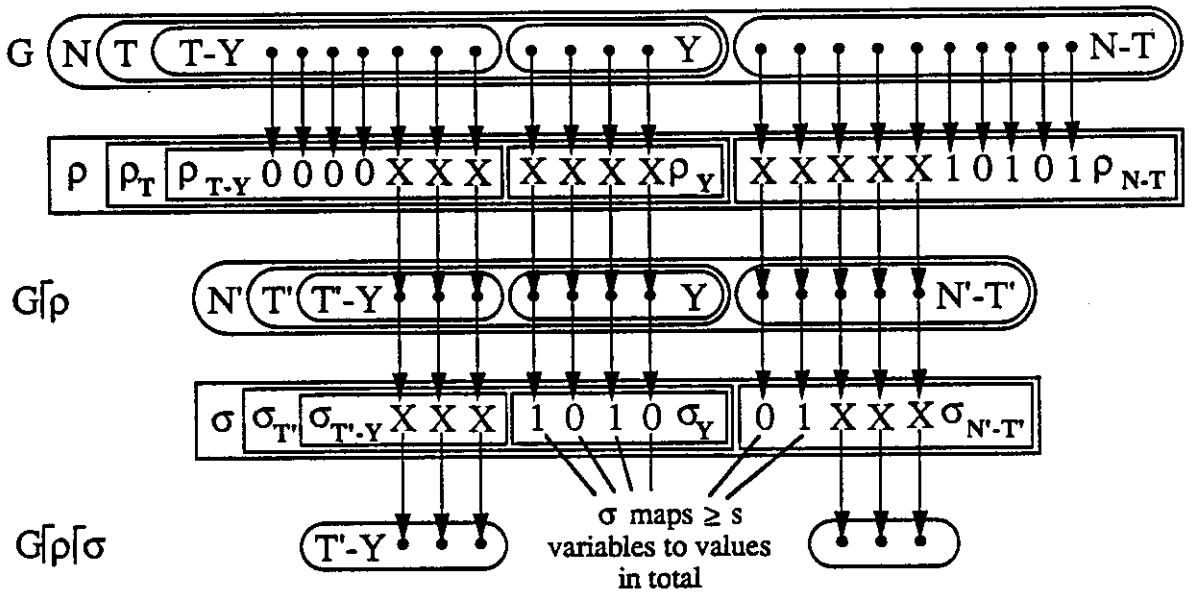


Figure 4.6: Pictorial Description of ρ and σ for Lemma 60.

additional condition that $\sigma_Y = \lambda$. Consider how $\sigma_{T'}$ maps the variables T'. The variables $T' - Y$ are all mapped to X, and the variables Y are all mapped to 0. These inputs feed into the OR gate G_1. Since it receives only 0 and X inputs, that OR gate is not forced to 1 (i.e., $(G_1 \lceil \rho_T) \lceil \sigma_{T'} \not\equiv 1$). However, $(G_1 \lceil \rho_T) \lceil \sigma_{T'}$ feeds into the top-level AND gate of the overall circuit $(G \lceil \rho) \lceil \sigma$, so it must also be the case that $(G \lceil \rho) \lceil \sigma \not\equiv 1$, contradicting our initial assumption that σ was a minterm of $G \lceil \rho$. $\Rightarrow\Leftarrow$ ■

Claim 63 *For all ρ such that $G \lceil \rho \not\equiv 1$ and $\rho_Y = \vec{X}_Y$,*

$$\left(\mathrm{lmt}^Y(G \lceil \rho) \geq s\right) \iff \left(\bigvee_{\lambda, \lambda \neq \vec{0}_Y} \mathrm{lmt}^{Y,\lambda}(G \lceil \rho) \geq s \right),$$

where λ ranges over all mappings $Y \to \{0,1\}$.

Proof This follows from Claim 62 above (it is for this reason that $\lambda \neq \vec{0}_Y$ in the $\bigvee$ on the right hand side) and from the simple fact that a minterm which assigns values to exactly the variables Y of T must make *some* assignment λ of 0's and 1's to the variables Y, and vice-versa. ■

Now, as before, we use probability Lemma 35 (pg. 16) to turn the $\bigvee$ into a sum:

$$\begin{aligned}
\text{(D)} &= \Pr\left[\mathrm{lmt}^Y(G \lceil \rho) \geq s \mid F \lceil \rho \equiv 1 \wedge G_1 \lceil \rho_T \not\equiv 1 \wedge \rho_Y = \vec{X}_Y\right] && (4.14)\\
&= \Pr\left[\bigvee_{\lambda, \lambda \neq \vec{0}_Y} \mathrm{lmt}^{Y,\lambda}(G \lceil \rho) \geq s \mid F \lceil \rho \equiv 1 \wedge G_1 \lceil \rho_T \not\equiv 1 \wedge \rho_Y = \vec{X}_Y\right] && (4.15)\\
&\leq \sum_{\lambda, \lambda \neq \vec{0}_Y} \underbrace{\Pr\left[\mathrm{lmt}^{Y,\lambda}(G \lceil \rho) \geq s \mid F \lceil \rho \equiv 1 \wedge G_1 \lceil \rho_T \not\equiv 1 \wedge \rho_Y = \vec{X}_Y\right]}_{\text{(G)}}. && (4.16)
\end{aligned}$$

(4.14) = (4.15) by Claim 63 above, and (4.15) $\leq$ (4.16) by Lemma 35 (pg. 16). We now attempt to bound the probability (G). When (G) is considered apart from the sum, *both* Y and λ are *fixed* in

(G) (Y is fixed because it was fixed in (D)). Once (G) is bounded, we can bound (D) and complete the proof of Lemma 60 (started on pg. 35).

4.2.3 Bounding the Probability (G)

Lemma 64 *For any $Y \subseteq T$, $Y \neq \emptyset$ and any $\lambda : Y \to \{0,1\}$, $\lambda \neq \vec{0}_Y$,*

$$(\mathrm{G}) = \Pr\left[\mathrm{lmt}^{Y,\lambda}(G\lceil\rho) \geq s \mid F\lceil\rho \equiv 1 \wedge G_1\lceil\rho_T \not\equiv 1 \wedge \rho_Y = \vec{X}_Y\right] \leq \alpha^{s-|Y|}.$$

Proof Let R_p'' be the probability distribution R_p conditioned on $G_1\lceil\rho_T \not\equiv 1$ and $\rho_Y = \vec{X}_Y$. Restrictions in this space are like the restriction ρ pictured in Figure 4.6 (pg. 36). As before, we define an equivalence relation $\approx$, and partition R_p'' into equivalence classes, where two restrictions are considered equivalent if they both map the variables in $T-Y$ to 0 and X in the same way. Symbolically,

$$\forall \rho, \delta \in R_p'', (\rho \approx \delta \iff \forall x_i \in T-Y, \rho(x_i) = \delta(x_i)).$$

There will be $2^{|T-Y|}$ equivalence classes. The class a restriction ρ is in depends solely on the element (0 or X) to which each variable in $T-Y$ gets mapped. Notice that a restriction ρ in any equivalence class can map the variables in $N-T$ to any element of $\{0,1,X\}$.

We can represent R_p'' by the table shown in Figure 4.7. The rows are labeled with the $2^{|T-Y|}$ mappings from the variables $T-Y$ to $\{0,X\}$, and the columns are labeled with the $3^{|N-T|}$ mappings from the variables in $N-T$ to $\{0,1,X\}$.[3] Each entry thus stands for a single restriction $\rho \in R_p''$, and the equivalence classes are precisely the *rows* of the table.

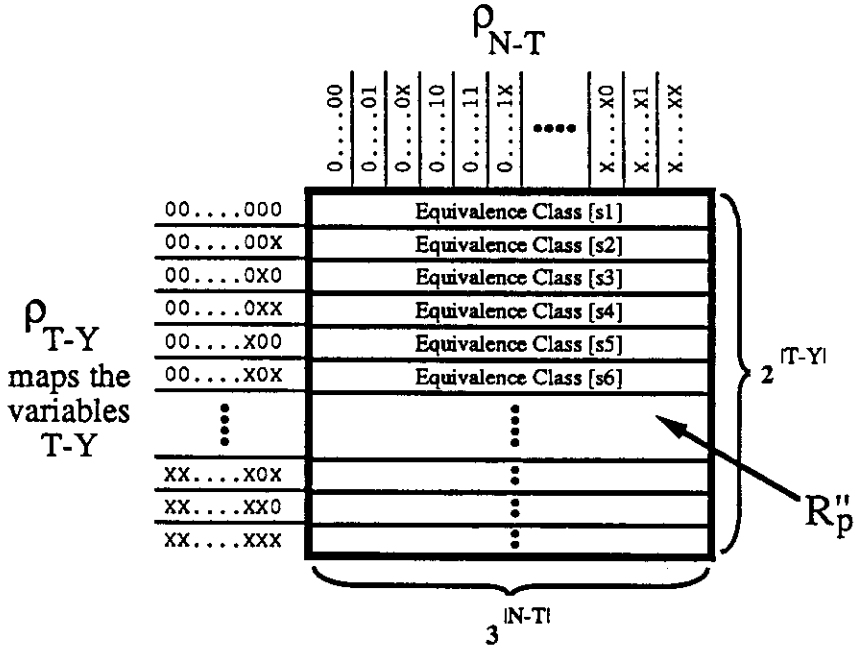


Figure 4.7: The Random Restriction Space R_p'' Partitioned into Equivalence Classes.

[3]The fact that all restrictions ρ in the table map all variables in Y to X is represented implicitly by showing that only restrictions $\rho \in R_p''$ are in the table.

Notation 65 ($[s_i]$) *We denote the equivalence classes induced by $\approx$ on R_p'' by $[s_1], [s_2], \ldots, [s_{2^{|T-Y|}}]$, where s_i is the* string *equivalent of the elements to which every restriction in the class maps the variables $T-Y$. In the table above, s_1 = "$00\cdots00$", s_2 = "$00\cdots0X$", s_3 = "$00\cdots X0$", ..., $s_{2^{|T-Y|}}$ = "$XX\cdots XX$".*

Since $[s_1], \ldots, [s_{2^{|T-Y|}}]$ partition the restrictions R_p'', we can apply probability Lemma 34 (pg. 16) to obtain:

$$\begin{aligned}
(\mathrm{G}) &= \Pr\left[\mathrm{lmt}^{Y,\lambda}(G\lceil\rho) \geq s \mid F\lceil\rho \equiv 1 \wedge G_1\lceil\rho_T \not\equiv 1 \wedge \rho_Y = \vec{X}_Y\right] \\
&\leq \max_i \Pr\left[\mathrm{lmt}^{Y,\lambda}(G\lceil\rho) \geq s \mid F\lceil\rho \equiv 1 \wedge G_1\lceil\rho_T \not\equiv 1 \wedge \rho_Y = \vec{X}_Y \wedge \rho \in [s_i]\right] \qquad (4.17) \\
&= \max_i \underbrace{\Pr\left[\mathrm{lmt}^{Y,\lambda}(G\lceil\rho) \geq s \mid F\lceil\rho \equiv 1 \wedge \rho \in [s_i]\right]}_{(\mathrm{H})}.
\end{aligned}$$

Notice that the conditions $G_1\lceil\rho_T \not\equiv 1$ and $\rho_Y = \vec{X}_Y$ have been dropped from the last probability. They were dropped because they are superfluous:

$$\rho \in [s_i] \implies \rho \in R_p'' \implies G_1\lceil\rho_T \not\equiv 1 \wedge \rho_Y = \vec{X}_Y$$

by definition of R_p''.

Recall that Y and λ are fixed in probability (G). When we consider (H) in isolation, i ($1 \leq i \leq 2^{|T-Y|}$) is additionally fixed. If we can show that (H) $\leq \alpha^{s-|Y|}$ for every way of fixing i, we will have bounded (G) and proven Lemma 64.

We therefore focus our attention on (H). Throughout this proof, we have used $\Pr[A]$ to stand for the probability of event A occurring when a restriction ρ was drawn at random from the distribution R_p. By successively adding conditions for ρ to satisfy, we have slowly been paring down the universe over which the probability is taken. The condition required of ρ in (H), namely, $\rho \in [s_i]$, limits the universe of restrictions more than any other condition we have seen so far in the proof. In fact, we will now show that this condition limits the universe enough for us to invoke the induction hypothesis!

Here is the key idea: if $\rho \in [s_i]$, then fixing i completely determines ρ_T. By virtue of being in R_p'', we know ρ maps all the variables in Y to X. By virtue of being in $[s_i]$, ρ's mapping of the variables in $T-Y$ is also completely determined. Since ρ_T as a whole is therefore completely determined, the probability in (H) is really being taken only over $\rho_{N-T} \in R_p$. To show that the probability is ranging only over $\rho_{N-T} \in R_p$, we will write $\Pr_{\rho_{N-T} \in R_p}[A]$.

We can solve for (H) by splitting up ρ into two restrictions (parts), ρ_T and ρ_{N-T}, that are equivalent to ρ when applied one after the other. The first "part" corresponds to the part of ρ which we know to be fixed in (H), namely, that part which acts on T. Since this restriction is fixed, applying it to a circuit gives us a single, fixed circuit. The second "part" corresponds to that portion of ρ which is unfixed in (H): the part that acts on $N-T$. Using the fact that $\rho \in [s_i]$ completely determines ρ_T, we have:

$$\begin{aligned}
(\mathrm{H}) &= \Pr\left[\mathrm{lmt}^{Y,\lambda}(G\lceil\rho) \geq s \mid F\lceil\rho \equiv 1 \wedge \rho \in [s_i]\right] \\
&= \Pr_{\rho_{N-T} \in R_p}\left[\mathrm{lmt}^{Y,\lambda}((G\lceil\rho_T)\lceil\rho_{N-T}) \geq s \mid (F\lceil\rho_T)\lceil\rho_{N-T} \equiv 1\right]. \qquad (4.18)
\end{aligned}$$

At this point, we are very close to being able to invoke the induction hypothesis. Note that for $\mathrm{lmt}^{Y,\lambda}((G\lceil\rho_T)\lceil\rho_{N-T}) \geq s$ to be true, there must exist a minterm σ such that $|\sigma| \geq s$ and such that

σ maps *exactly* the variables in Y to values according to the assignment λ. Recall that we use G' to denote $\bigvee_{i=2}^{w} G_i$, and that the induction hypothesis says that the stronger switching lemma is true for G'. Consider applying first ρ_T, then $\sigma_{T'}$, and finally ρ_{N-T} to G. By definition of a minterm, we know that $(G_1 \lceil \rho_T) \lceil \sigma_{T'} \equiv 1$, so after applying ρ_T and $\sigma_{T'}$ to G, the sub-circuit G_1 has been forced to 1. Since the top-level gate of G is an AND, the circuit which results is G', so

$$((G \lceil \rho_T) \lceil \sigma_{T'}) \lceil \rho_{N-T} \equiv G' \lceil \rho_{N-T}$$

Since $\sigma_{T'}$ maps exactly the variables in Y to values, $|\sigma_{T'}| = |Y|$. Therefore, if there is a minterm σ meeting all the requirements discussed in the previous paragraph, $\sigma_{N'-T'}$ will be a minterm for $G' \lceil \rho_{N-T}$ of size $|\sigma_{N'-T'}| = |\sigma| - |\sigma_{T'}| = |\sigma| - |Y| \geq s - |Y|$. That is,

$$\mathrm{lmt}^{Y,\lambda}((G \lceil \rho_T) \lceil \rho_{N-T}) \geq s \implies \mathrm{lmt}(G' \lceil \rho_{N-T}) \geq s - |Y|. \tag{4.19}$$

Finally, letting F' denote the fixed circuit $F \lceil \rho_T$ in the probability of equation (4.18) gives:

$$\begin{aligned}
(4.18) &= \Pr_{\rho_{N-T} \in R_p} \left[\mathrm{lmt}^{Y,\lambda}((G \lceil \rho_T) \lceil \rho_{N-T}) \geq s \mid F' \lceil \rho_{N-T} \equiv 1 \right] && (4.20)\\
&\leq \Pr_{\rho_{N-T} \in R_p} \left[\mathrm{lmt}(G' \lceil \rho_{N-T}) \geq (s - |Y|) \mid F' \lceil \rho_{N-T} \equiv 1 \right] && (4.21)\\
&\leq \alpha^{s-|Y|} && (4.22)
\end{aligned}$$

The inequality (4.20) $\leq$ (4.21) follows from (4.19) above. The inequality (4.21) $\leq$ (4.22) holds by the induction hypothesis, where "small" fan-in has been changed to mean $< (s - |Y|)$, and α has been raised to that power in (4.22) accordingly. Therefore, (H) = (4.18) $\leq \alpha^{s-|Y|}$.

Equation (4.17) (pg. 38) tells us that

$$(\mathrm{G}) \leq \max_i (\mathrm{H}).$$

Since the bound on (H) is independent of i, these two facts taken together imply that

$$(\mathrm{G}) \leq \max_i \alpha^{s-|Y|} = \alpha^{s-|Y|}$$

thereby proving Lemma 64 (pg. 37). ∎

We can now resume the proof of Lemma 60 (pg. 35) and bound the probability (D). Recall from equations (4.14)–(4.16) (pg. 36) that

$$(\mathrm{D}) \leq \left[\sum_{\lambda,\, \lambda \neq \vec{0}_Y} (\mathrm{G}) \right] \leq \left[\sum_{\lambda,\, \lambda \neq \vec{0}_Y} \alpha^{s-|Y|} \right].$$

The sum is taken over $(2^{|Y|} - 1)$ possible values for λ. Since the bound on (G) is independent of λ, (D) $\leq (2^{|Y|} - 1)(\alpha^{s-|Y|})$. This completes the proof of Lemma 60 (pg. 35). ∎

We are now in a position to finally complete the proof of Lemma 50 (pg. 30) bounding the probability (B). We have shown so far that

$$(\mathrm{C}) \leq \left(\frac{2p}{1+p} \right)^{|Y|} \text{ and } (\mathrm{D}) \leq (2^{|Y|} - 1)(\alpha^{s-|Y|}).$$

Recall from equations (4.8)–(4.12) (pg. 33) that (B) $\leq \sum_{Y \subseteq T, Y \neq \emptyset}$ (C) $\times$ (D). Therefore,

$$\begin{aligned}
(\mathrm{B}) &= \Pr[\mathrm{lmt}(G\lceil\rho) \geq s \mid F\lceil\rho \equiv 1 \wedge G_1\lceil\rho_T \not\equiv 1] \\
&\leq \sum_{Y \subseteq T, Y \neq \emptyset} \left(\frac{2p}{1+p}\right)^{|Y|} (2^{|Y|}-1)(\alpha^{s-|Y|}) \\
&= \alpha^s \sum_{Y \subseteq T, Y \neq \emptyset} \frac{(2p)^{|Y|}2^{|Y|} - (2p)^{|Y|}}{((1+p)\cdot\alpha)^{|Y|}} \\
&= \alpha^s \sum_{Y \subseteq T, Y \neq \emptyset} \frac{(4p)^{|Y|} - (2p)^{|Y|}}{((1+p)\cdot\alpha)^{|Y|}} \qquad (4.23)
\end{aligned}$$

Since the summand in equation (4.23) depends only on the *size* of Y and not on its members, we now sum over all sizes of Y. The number of sets $Y \subseteq T$ of size i is $\binom{|T|}{i}$. We will sum over all sizes i; we can start at $i = 0$ because the summand in (4.23) is 0 when $|Y| = 0$. Using simple algebra, we can simplify equation (4.23) as follows.

$$\begin{aligned}
(4.23) &= \alpha^s \sum_{Y \subseteq T, Y \neq \emptyset} \frac{(4p)^{|Y|} - (2p)^{|Y|}}{((1+p)\cdot\alpha)^{|Y|}} \\
&= \alpha^s \sum_{i=0}^{|T|} \binom{|T|}{i} \left[\left(\frac{4p}{1+p}\cdot\frac{1}{\alpha}\right)^i - \left(\frac{2p}{1+p}\cdot\frac{1}{\alpha}\right)^i\right] \\
&= \alpha^s \left[\left(\sum_{i=0}^{|T|} \binom{|T|}{i}\left(\frac{4p}{1+p}\cdot\frac{1}{\alpha}\right)^i\right) - \left(\sum_{i=0}^{|T|} \binom{|T|}{i}\left(\frac{2p}{1+p}\cdot\frac{1}{\alpha}\right)^i\right)\right] \\
&= \alpha^s \left[\left(1+\left(\frac{4p}{1+p}\cdot\frac{1}{\alpha}\right)\right)^{|T|} - \left(1+\left(\frac{2p}{1+p}\cdot\frac{1}{\alpha}\right)\right)^{|T|}\right]. \qquad (4.24)
\end{aligned}$$

The last equality is achieved by employing the binomial coefficient formula

$$\sum_{i=0}^{n} \binom{n}{i} x^i = (1+x)^n$$

once for each sum, using $n = |T|$. Finally, using the fact that $|T| \leq t$, we have:

$$(4.24) \leq \alpha^s \left[\left(1+\left(\frac{4p}{1+p}\cdot\frac{1}{\alpha}\right)\right)^t - \left(1+\left(\frac{2p}{1+p}\cdot\frac{1}{\alpha}\right)\right)^t\right] = \alpha^s.$$

The equality follows from the fact that α was defined to be the root of

$$\left(1+\left(\frac{4p}{1+p}\cdot\frac{1}{\alpha}\right)\right)^t = 1 + \left(1+\left(\frac{2p}{1+p}\cdot\frac{1}{\alpha}\right)\right)^t$$

so

$$\left[\left(1+\left(\frac{4p}{1+p}\cdot\frac{1}{\alpha}\right)\right)^t - \left(1+\left(\frac{2p}{1+p}\cdot\frac{1}{\alpha}\right)\right)^t\right] = 1.$$

Therefore, (B) $\leq \alpha^s$. This completes the proof of Lemma 50 (pg. 30). ■

By Lemmas 49 (pg. 30) and 50 (pg. 30) we have thus shown that both (A) and (B) are $\leq \alpha^s$. By equation (4.3) (pg. 30), this completes the induction, and thus proves the stronger switching lemma. ■

To apply the switching lemma, we need a bound on the size of α. Håstad gives the following lemma, which we state without proof.

Lemma 66 ([Hås87] Lemma 4.8) *Let α solve the equation given for it in the switching lemma on page 28. Then for some absolute constant $p_0 > 0$, for all $p < p_0$, $\alpha < 5pt$.*

Chapter 5

Lower Bounds for Parity

We now use the power of the switching lemma to prove lower bounds on the size of AC^0 PARITY circuits. We start by proving a theorem suitable for induction, and then show that this implies an exponential lower bound on the size of constant-depth PARITY circuits. We then give a corollary which bounds the minimum depth of polynomial-size AC^0 circuits[1] computing PARITY.

5.1 A Preliminary Theorem

We would like to be able to work with circuits in canonical form. However, because wires from level-1 gates of circuits in canonical form may extend to gates above level 2, we cannot freely switch the sub-circuits rooted at level-2 gates. We therefore relax the definition of canonical form as follows.

Definition 67 (Quasi-Canonical Form) *A circuit is in* quasi-canonical form *if it is in canonical form, with the exception that gates on level 1 may be of either type.*

Theorem 68 ([Hås87] Theorem 5.2) PARITY *cannot be computed by AC^0 circuits in quasi-canonical form with:*

- **depth k,**
- **# of gates on levels 2 and above $\leq 2^{\frac{1}{10}n^{1/(k-1)}}$, *and***
- **bottom fan-in $\leq \frac{1}{10}n^{1/(k-1)}$.**

for sufficiently large n.

For the proof, assume without loss of generality that the gates on level 2 are AND gates.[2] The theorem states that circuits of the form pictured in Figure 5.1 cannot compute PARITY.

Proof (by induction on the depth k)

Base Case ($k = 2$) We first argue that a depth-2 circuit for PARITY in quasi-canonical form must in fact be in canonical form. If it were not, there would be some gate on level 1 that was the same type as the sole level-2 gate. In this case, fixing a single input (to 0 if the circuit

[1] Of course, these circuits are AC^0 in every respect except for being constant depth.

[2] We do not lose generality because the switching lemma works on both AND-of-OR and OR-of-AND circuits (by Corollary 42 (pg. 28)).

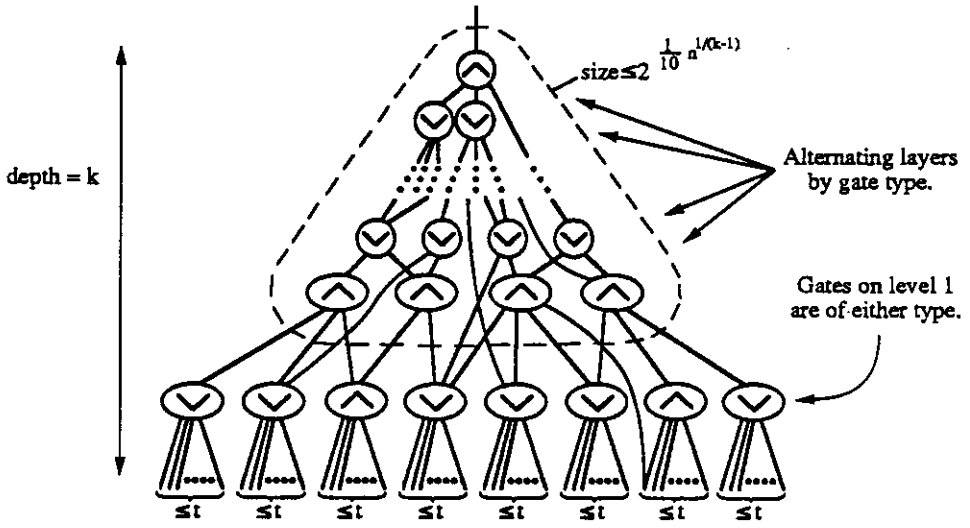


Figure 5.1: The Circuits of Theorem 68 Which Cannot Compute PARITY.

contained an AND of an AND, or to 1 if it contained an OR of an OR) would force the value of the circuit, so it could not compute PARITY.

Therefore, all of the gates on level 1 must be of the same type. To prove the result, all we must show is that every AND-of-ORs or OR-of-ANDs for PARITY must have bottom fan-in larger than $\frac{1}{10}n^{1/(k-1)}$. In fact, something much stronger than that can be proven:

Claim 69 *Every level-1 gate of any* OR-*of*-ANDs *or* AND-*of*-ORs PARITY *circuit must have fan-in* $\geq n$.

Proof (by contradiction) Say we had an OR-of-ANDs computing PARITY which contains an AND gate g (on level 1) with fan-in $n' < n$. Let ψ be any assignment to the variables which makes all n' literal inputs to g (and hence, g itself) true. Since the top-level gate of the circuit is an OR, the assignment ψ also makes the entire circuit true.

The fan-in of g is $< n$, so there is some variable x_i not appearing as an input to g. There is thus an assignment ψ' which is the same as ψ except that it maps the variable x_i to the other value. Notice that g, and hence the entire circuit, will still be true on the assignment ψ'. However, PARITY is such that changing any one of the inputs changes the output, so our circuit cannot be computing PARITY as assumed.$\Rightarrow\Leftarrow$

That AND-of-ORs for PARITY cannot have a level-1 gate with fan-in $< n$ follows from the fact that we could take the complement of any such circuit to yield an OR-of-ANDs for $\neg$PARITY. Since complementing an AND-of-ORs does not change any gate's fan-in, the resulting circuit has a level-1 gate with fan-in $< n$. However, it easily verified that the above argument also applies to OR-of-ANDs for $\neg$PARITY. ■

As an aside, the following claim immediately follows from Claim 69:

Claim 70 *Every* AND-*of*-ORs *or* OR-*of*-ANDs *for* PARITY *must have at least* $1 + 2^{n-1}$ *gates.*

Proof Without loss of generality, consider an OR-of-ANDs.[3] Any AND gate which has both a variable and its complement as inputs will be forced to 0, making it useless. Therefore, only count the AND gates with one input for each variable. By Claim 69, each AND gate has fan-in n, so it will have exactly one input for each variable. Thus, each AND gate is made true by exactly one assignment to the variables. Since PARITY has 2^{n-1} satisfying assignments, there must be at least that many AND gates. Taking the top-level OR gate into account, it thus follows that any OR-of-ANDs for PARITY must have at least $1 + 2^{n-1}$ gates. ■

Inductive Step Assume true for k and show true for $k+1$.

Proof (by contradiction) We assume such depth-$k+1$ circuits *do* exist for PARITY, and then show that this implies the existence of such depth-k circuits, thereby contradicting the induction hypothesis. The contradiction implies that depth-$k+1$ circuits of this form cannot, in fact, recognize PARITY.

Therefore, assume such depth-$k+1$ circuits for PARITY exist. They will have size $\leq 2^{\frac{1}{10}n^{1/k}}$ (not counting gates on level 1) and bottom fan-in $\leq \frac{1}{10}n^{1/k} = t$. We will apply the switching lemma to these depth-$k+1$ circuits and construct depth-k circuits for PARITY which meet the requirements of the theorem. To prepare the circuit for switching, we make some changes to the circuit. This involves the following 4-step process.

1. Duplicate gates on level 1 having fan-out greater than 1 to insure that all level-1 gates have fan-out 1.
2. Separate the gates on level 1 into two types, "high" and "low", depending on whether their output leads to a gate above level 2 or a gate on level 2, respectively.
3. Each "low" gate of the incorrect type (AND in this case) has an output leading to a gate on level 2 of the same type. Merge all such gates.
4. If an input has a wire leading to a level-2 gate, insert a gate on level 1 along this wire of the correct type (OR in this case); this gate will have fan-in and fan-out 1.

An example of these four transformations is shown in Figure 5.2.

The "high" level-1 gates will be set aside and left "as is" for the rest of the proof. If all the children of a level-2 gate are of the incorrect type for level 1, step 3 may increase the bottom-fanin of the circuit. However, step 4 will insert dummy gates, each having fan-in 1, to fix that problem. Therefore, we see that for any circuit, the overall transformation described above does not change the function it computes, its depth, the number of gates it contains on levels 2 and above, or its bottom fan-in. Moreover, the sub-circuits rooted at level-2 gates are AND-of-ORs sharing no level-1 gates, and the OR gates in these AND-of-ORs each have only one output going to a level-2 AND gate. We can therefore apply the switching lemma independently to these sub-circuits.

We next choose $s = t = \frac{1}{10}n^{1/k}$, fix $p = n^{-\frac{1}{k}}$, and apply a random restriction from R_p. This allows us to apply the switching lemma independently to the sub-circuits rooted at level-2 gates. By the switching lemma, the probability that we cannot write any particular one of these depth-2 sub-circuits as an OR of small ANDs is bounded by α^s. The probability that we fail overall in being able to write *all* the depth-2 sub-circuits as ORs of small ANDs is the same as the probability that we will fail on at least 1 of them (i.e., we fail on the first one or we fail on the second one or...).

[3]The result also holds for AND-of-ORs because the following proof works equally well for ¬PARITY, and we can take the complement in that case.

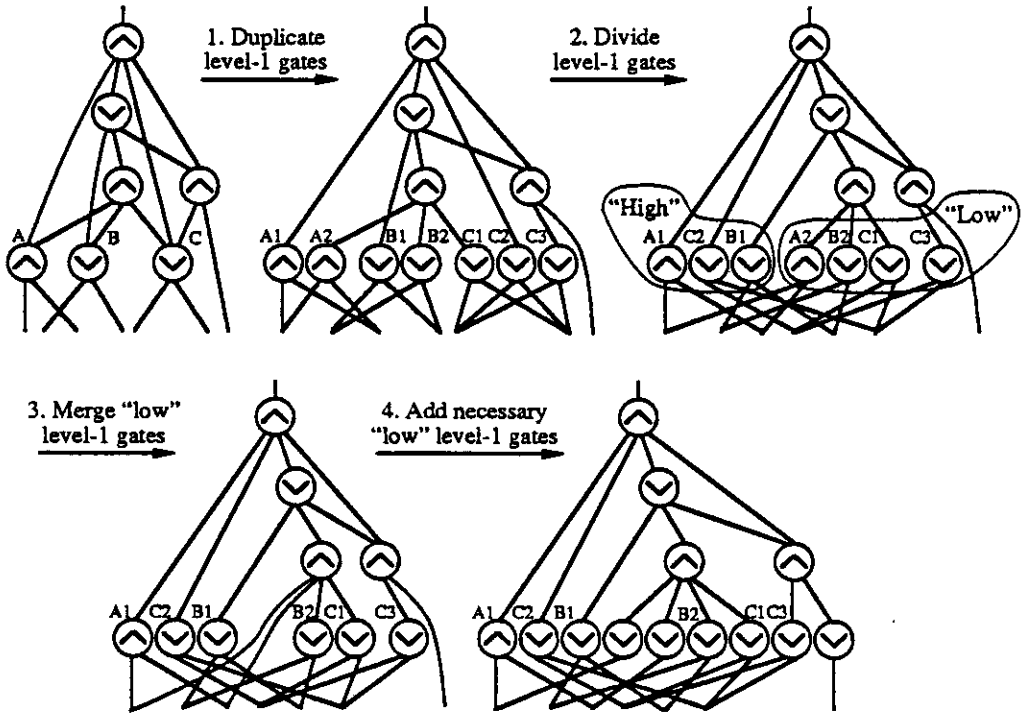


Figure 5.2: The Four-step Transformation of Theorem 68 (pg. 42).

Using probability Lemma 35 (pg. 16), the probability that a random restriction ρ fails overall is therefore bounded by the sum of the probabilities of failure on each sub-circuit. The bound on the probability of failure implied by the Switching Lemma holds irrespective of the sub-circuit being switched, so the probability of failure for each sub-circuit is bounded by the same value, namely α^s. Therefore:

$$\Pr[\text{failure overall}] \leq \alpha^s \times (\#\text{ of depth-2 sub-circuits}) \tag{5.1}$$

The total number of depth-2 sub-circuits is certainly bounded by the total number of gates on levels 2 and higher, so

$$(\#\text{ of depth-2 sub-circuits}) \leq 2^{\frac{1}{10}n^{1/k}}. \tag{5.2}$$

Taken together, equations (5.1) and (5.2) imply that

$$\Pr[\text{failure overall}] \leq \left(2^{\frac{1}{10}n^{1/k}}\right)(\alpha^s) = 2^s\alpha^s = (2\alpha)^s.$$

Therefore, with probability at least $1 - (2\alpha)^s$ we will *succeed* in switching all the depth-2 sub-circuits.

We now need to "merge" gates on levels 2 and 3. However, a level-2 gate could have outputs going to one or more gates on level 3 *and* to one or more gates above level 3. As in the construction of Claim 12 (pg. 9) that transformed a circuit into canonical form, we "merge" the gate with gates it connects to on level 3, and "slide" the gate up one level for the other outputs. In the worst case, every level-2 gate may need to slide up to level 3 in this way. As an example of this final process, Figure 5.3 shows what happens when we switch, merge, and slide the final circuit of Figure 5.2. After switching and then sliding and merging as necessary, we have a new circuit with the following properties:

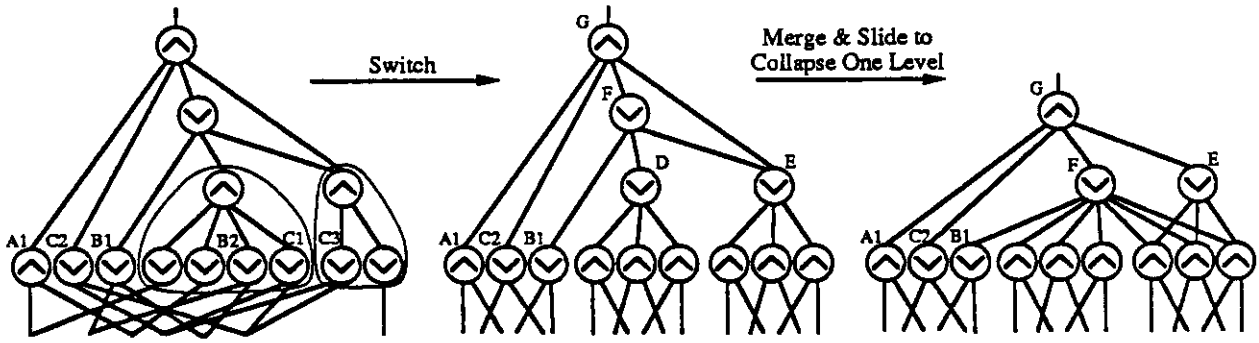


Figure 5.3: Switching and Collapsing the Final Circuit of Figure 5.2.

- **computes** PARITY **or** ¬PARITY

  This follows by Observation 23 (pg. 12). If the resulting circuit is ¬PARITY, we take its complement; by Observation 14 (pg. 10), the resulting circuit (which computes PARITY) is of the same size and depth.
- **depth** $= k$

  After merging and sliding the level-2 gates, there are no more gates on level 2. We can therefore collapse at level-2 to get an overall circuit of depth k. That is, we can decrease the level of all gates above level 1 by one.
- **# of gates on levels 2 and above** $\leq 2^{\frac{1}{10}n^{1/k}}$

  As we said earlier, all level-2 gates in the original circuit may slide up to level 3 in the new circuit in the worst case. Therefore, after collapsing at level 2, the number of gates on levels 2 and above in the new circuit is at most the number of gates on levels 2 and above in the old circuit.
- **bottom fan-in** $\leq s = \frac{1}{10}n^{1/k}$

  The bottom fan-in of the "high" level-1 gates we set aside does not change; it is $t = \frac{1}{10}n^{1/k}$. The bound given by the Switching Lemma for the bottom fan-in of the "low" level-1 gates is $s = \frac{1}{10}n^{1/k}$.
- **quasi-canonical form**

  We have not changed any part of circuit above level 3. The "sliding" operation adds new gates to level 3, but they are all of the correct type. There are no longer any gates on level 2. Therefore, the new circuit is now in quasi-canonical form.
- **# of inputs** $\geq pn = n^{(k-1)/k}$ (with non-zero probability)

  That ρ succeeds and/or maps at least pn variables to X with non-zero probability follows if we can show that the probability of mapping pn or more variables to X is strictly greater than the probability of failure (see the more complete description on pg. 23). Therefore, we must demonstrate that

$$(2\alpha)^s < \Pr[\rho \text{ maps} \geq pn \text{ variables to } X] = \sum_{i=pn}^{n} \binom{n}{i} p^i (1-p)^{n-i}. \qquad (5.3)$$

By Lemma 66 (pg. 41),

$$\alpha < 5pt = 5\left(\frac{1}{n^{1/k}}\right)\left(\frac{1}{10}n^{1/k}\right) = \frac{1}{2},$$

so $2\alpha < 1$. Since $s \to \infty$ as $n \to \infty$, $\lim_{n\to\infty}(2\alpha)^s = 0$. Hence, for any constant fraction (such as $1/3$), $(2\alpha)^s < 1/3$ for sufficiently large n. We will therefore complete the proof of inequality (5.3) by showing that:

Claim 71 *For sufficiently large n, $1/3 <$ the sum of inequality (5.3) (pg. 46).*

Proof (by citation) The proof requires some technical bounds from probability theory. See [Bol85], Chapter 1 or [Fel68], Chapter VII for a more complete explanation of these bounds. Let $S_{n,p}$ denote the random variable computed by summing n independent random variables, each of which has probability p of being 1 and probability $q = (1-p)$ of being 0. Then we have:

$$\sum_{i=pn}^{n} \binom{n}{i} p^i (1-p)^{n-i} = \Pr[\, S_{n,p} \geq pn \,].$$

We cite the following well-known theorem for bounding the area under the tail of such a binomial distribution.

Theorem 72 *(DeMoivre-Laplace) Let $0 < p < 1$ and $0 < x$ be functions of n. If $pqn \to \infty$ as $n \to \infty$, and if $x(pqn)^{1/2} = o\big((pqn)^{2/3}\big)$, then*

$$\lim_{n\to\infty} \Pr\left[\, S_{n,p} \geq pn + x(pqn)^{1/2} \,\right] = 1 - \Phi(x),$$

where

$$\phi(x) = \frac{1}{\sqrt{2\pi}} e^{-\frac{1}{2}x^2} \text{ and } \Phi(x) = \int_{-\infty}^{x} \phi(x)\, dx$$

are the normal density function and normal distribution function, respectively.

We choose $x = 1/n$. We must first verify that our p and x satisfy the conditions of the theorem. Indeed, $pqn = n^{(k-1)/k}(1 - n^{-1/k})$, so $pqn \to \infty$ as $n \to \infty$. The second condition is also easily satisfied.

It is well known that $\lim_{y\to\infty} \Phi(y) = 1$. Notice also that ϕ is a symmetric function. Therefore, since $x = 1/n$, $\lim_{n\to\infty} \Phi(x) = \lim_{y\to 0} \Phi(y) = 1/2$, so we have that:

$$\lim_{n\to\infty} \Pr\left[\, S_{n,p} \geq pn + x(pqn)^{1/2} \,\right] = 1/2.$$

Since the extra quantity $x(pqn)^{1/2} \to 0$ as $n \to \infty$,

$$\lim_{n\to\infty} \Pr[\, S_{n,p} \geq pn \,] = 1/2,$$

so $1/3 < \Pr[\, S_{n,p} \geq pn \,]$ for sufficiently large n. ∎

This result establishes inequality (5.3) (pg. 46), thereby proving that such a successful restriction occurs with non-zero probability. Since the probability that ρ succeeds *and* assigns at least $n^{(k-1)/k}$ variables to X is non-zero, in particular, there must be at least one such restriction. We apply that restriction to the circuit, take the complement if necessary, and consider the PARITY circuit that results.

In order to invoke the induction hypothesis on this new circuit, we must recalculate the bounds on the number of gates and the bottom fan-in in terms of the new number of inputs and the new depth. Letting $m = n^{(k-1)/k}$ (i.e., $n = m^{k/(k-1)}$) represent the number of inputs to the new circuit, we find that the resulting PARITY circuit has:

- **depth** k
- **# of inputs** $= m$
- **# of gates on levels 2 and above** $\leq 2^{\frac{1}{10}n^{1/k}} = 2^{\frac{1}{10}\left(m^{k/(k-1)}\right)^{(1/k)}} = 2^{\frac{1}{10}m^{1/(k-1)}}$
- **bottom fan-in** $\leq \frac{1}{10}n^{1/k} = \frac{1}{10}\left(m^{k/(k-1)}\right)^{(1/k)} = \frac{1}{10}m^{1/(k-1)}$,

and that the circuit is in quasi-canonical form. However, the induction hypothesis tells us that precisely such circuits cannot exist. $\Rightarrow\Leftarrow$ ■

5.2 The Main Result

Theorem 73 ([Hås87] Theorem 5.1) PARITY *cannot be computed by* AC^0 *circuits with:*

- **depth** k, *and*
- **size** $\leq 2^{\left(\frac{1}{10}\right)^{k/k-1}n^{1/k-1}}$

for sufficiently large n.

Proof (by contradiction) Assume such circuits exists. Without loss of generality, we can assume they are in canonical form.[4] Now, make each circuit in the family a depth-$k+1$ circuit by adding an extra gate on "level 0" for each wire leaving an input,[5] so the bottom fan-in of the new circuit is $t = 1$. Let $p = \frac{1}{10}$ and apply the switching lemma to the sub-circuits rooted at level-1 gates using $s = \frac{1}{10}\left(\frac{n}{10}\right)^{1/(k-1)}$. Again, we compute the properties of the new circuit:

- **depth** $= k$
  We switch the bottom 2 levels of the circuit, and then merge and slide the level-2 gates as before to collapse at level 2 and reduce the depth by one.
- **# of gates on levels 2 and above** $\leq 2^{\left(\frac{1}{10}\right)^{k/k-1}n^{1/k-1}}$
  In the worst case, we may have to slide all of the old level-2 gates. Therefore, the number of gates on levels 2 and above in the new circuit is at most the number of gates on levels 1 and above (i.e., all the gates) in the old circuit.
- **bottom fan-in** $\leq s = \frac{1}{10}\left(\frac{n}{10}\right)^{1/(k-1)}$
  This is the bound given by the Switching Lemma.
- **canonical form**
  We assumed the circuit started in canonical form. Canonical form is preserved by the switching operation on the bottom-most sub-circuits and by the merging and sliding operations.
- **# of inputs** $= pn = n/10$ (with non-zero probability)
  The fact that a restriction exists with non-zero probability which succeeds in switching and which maps at least $1/10$ of the variables to X follows by the counting arguments used above. These arguments work because $\alpha < 5pt = 5 \cdot (1/10) \cdot 1 = 1/2$ as before, so the probability of failure approaches 0 as $n \to \infty$. As before, the probability that a random restriction maps at least the expected number of variables to X approaches $1/2$ as $n \to \infty$.

[4] We do not lose generality because, by Claim 12 (pg. 9), transforming a circuit to canonical form does not increase its depth, and can at most double its size. The result can be proven for canonical circuits using a smaller constant in the exponent than 1/10 (such as 1/11), so for sufficiently large n, the result as stated holds for circuits in general; the factor of 2 size blow-up is drowned out by the smaller constant in the exponent for sufficiently large n.

[5] We use AND gates if the level-1 gates are ORs and OR gates if the level-1 gates are ANDs.

That such a circuit exists with non-zero probability implies in particular that at least one exists. Consider this new circuit. As in the proof of Theorem 68 (pg. 42), let m denote the number of inputs in the new circuit, and compute the bottom fan-in and size in terms of this value. Setting $m = n/10$ (i.e., $n = 10m$) we have:

$$\text{bottom fan-in} \leq \frac{1}{10}\left(\frac{n}{10}\right)^{1/(k-1)} = \frac{1}{10}m^{1/(k-1)}$$

and

$$\begin{aligned}\text{size (not counting gates on level 1)} &\leq 2^{\left(\frac{1}{10}\right)^{k/k-1}n^{1/k-1}} \\ &= 2^{\left(\frac{1}{10}\right)^{k/k-1}(10m)^{1/k-1}} \\ &= 2^{\left(\frac{1}{10}\right)^{k/(k-1)}\left(\frac{1}{10}\right)^{-1/(k-1)}m^{1/(k-1)}} \\ &= 2^{\frac{1}{10}m^{1/(k-1)}}\end{aligned}$$

Referring to Theorem 68 (pg. 42), we see that such circuits for PARITY cannot exist. $\Rightarrow\Leftarrow$ ■

5.3 A Lower Bound on Depth

Corollary 74 ([Hås87] Corollary 5.3) *Polynomial-size circuit families for* PARITY *must have depth at least* $(\log n)/(c + \log\log n)$ *for some constant* c.

Proof The lower bound on the size of constant-depth PARITY circuits given in Theorem 73 (pg. 48),

$$\text{size} \leq 2^{\left(\frac{1}{10}\right)^{k/k-1}n^{1/k-1}}, \tag{5.4}$$

is expressed in terms of the depth k and the number of inputs n. To prove this corollary, we ask how big the depth would need to be in order for (5.4) to be a polynomial of n. That is, we solve for the depth k which makes (5.4)$= n^c$ for some c. The result follows using simple algebra:

$$\begin{aligned}
n^c &= 2^{\left(\frac{1}{10}\right)^{k/k-1}n^{1/k-1}} \\
&\Downarrow && \text{(take the log of both sides)} \\
c\log n &= \left(\tfrac{1}{10}^k n\right)^{1/k-1} \\
&\Downarrow && \text{(take the log of both sides)} \\
\log(c\log n) &= \tfrac{1}{k-1}\left(k\log\tfrac{1}{10} + \log n\right) \\
&\Downarrow \\
\log c + \log\log n &= \tfrac{k}{k-1}\log\left(\tfrac{1}{10}\right) + \tfrac{1}{k-1}\log n \\
&\Downarrow && \text{(let } c' = \log c \text{ and } c'' = \log\left(\tfrac{1}{10}\right)\text{)} \\
c' + \log\log n &= \tfrac{kc''}{k-1} + \tfrac{1}{k-1}\log n \\
&\Downarrow && \text{(multiply right side by } \tfrac{k-1}{k} < 1\text{)} \\
c' + \log\log n &> c'' + \tfrac{1}{k}\log n \\
&\Downarrow && \text{(let } c''' = c' - c''\text{)} \\
c''' + \log\log n &> \tfrac{1}{k}\log n \\
&\Downarrow \\
k &> \tfrac{\log n}{c''' + \log\log n}
\end{aligned}$$

thereby demonstrating the bound to be proven. ■

Acknowledgements

I would like to thank Doug Tygar for his patience, his instruction. and his help in answering questions which arose in the writing of this paper. He was an inspiration at times when progress was slow. I am also greatly indepted to Stephen Guattery, David Kosbie, and Jay Sipelstein for their careful corrections and suggestions. Finally, I want to thank Merrick Furst for teaching me about circuit complexity.

Bibliography

[Ajt83] M. Ajtai. Σ_1^1-formulae on finite structures. *Annals of Pure and Applied Logic*, 24(1):1–48, July 1983.

[BGS75] T. Baker, J. Gill, and R. Solovay. Relativizations of the P =? NP question. *SIAM Journal on Computing*, 4(4):431–442, December 1975.

[Bol85] Béla Bollobás. *Random Graphs.* Academic Press, New York, 1985.

[Coo79] S. A. Cook. Deterministic CFL's are accepted simultaneously in polynomial time and log squared space. In *Proc. 11^{th} ACM Symposium on the Theory of Computing*, pages 338–345, 1979.

[Fel68] William Feller. *An Introduction to Probability Theory and Its Applications.* John Wiley & Sons, Inc., New York, third edition, 1968.

[FSS84] Merrick Furst, James B. Saxe, and Michael Sipser. Parity, circuits, and the polynomial-time hierarchy. *Mathematical Systems Theory*, 17(1):13–27, 1984.

[Hås86] J. Håstad. Almost optimal lower bounds for small depth circuits. In *Proc. 18^{th} ACM Symposium on the Theory of Computing*, pages 6–20, 1986.

[Hås87] Johan Håstad. *Computatonal Limitations of Small-Depth Circuits.* ACM Doctoral Dissertation Series. MIT Press, 1987.

[HU79] John E. Hopcroft and Jeffrey D. Ullman. *Introduction to Automata Theory, Languages, and Computation.* Addison-Wesley Series in Computer Science. Addison-Wesley Publishing Company, Inc., Reading, MA, 1979.

[PF79] N. Pippenger and M. J. Fischer. Relations among complexity measures. *Journal of the Association for Computing Machinery*, 26(2):361–381, 1979.

[Yao85] Andrew Chi-Chih Yao. Separating the polynomial-time hierarchy by oracles. In *26^{th} Annual Symposium on Foundations of Computer Science*, pages 1–10, 1985.

Index

Numbers of pages containing definitions of a concept are shown in **boldface**; number of pages containing examples of a concept are shown in *italics*. Symbols are listed in their approximate order of first appearence in the text.