INFINITE GAMES, ONLINE PROBLEMS AND AMPLIFICATION

by

Sanjeev Mahajan

B.Tech, Indian Institute of Technology, Delhi, 1982
M.Sc., Simon Fraser University, 1987

A THESIS SUBMITTED IN PARTIAL FULFILLMENT
OF THE REQUIREMENTS FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY
in the School
of
Computing Science

SIMON FRASER UNIVERSITY
September 1992

APPROVAL

Name: Sanjeev Mahajan

Degree: Doctor of Philosophy

Title of Thesis: Infinite Games, Online Problems and Amplification

Examining Committee:

Dr. Tiko Kameda, Senior Supervisor

Dr. Arvind Gupta, Senior Supervisor

Dr. Pavol Hell, Supervisor

Dr. Ramesh Krishnamurti, Internal Examiner

Dr. Russell Impagliazzo, External Examiner
Assistant Professor,
Computer Science, UC, San Diego

Abstract

This thesis consists of two parts. The part on infinite games and online problems is joint work with Xiaotie Deng [DM91] and the second part which is on Amplification is work done jointly with Arvind Gupta [GM92].

In an online problem, requests come online, and they need to be answered without knowing future requests. *Competitive ratio*, a performance measure for an online algorithm, measures how well the algorithm performs against an optimal algorithm which knows all the requests in advance. It is the worst case ratio of the cost incurred by the online algorithm versus the cost of the optimal offline algorithm. If the competitive ratio of an online algorithm is not more than α, it is called an *α-competitive* algorithm.

Ben-David, Borodin, Karp, Tardos and Wigderson(1990) initiated a systematic study of randomization in online problems. They formalized online problems as request-answer games, and also clarified several issues regarding randomization in online problems. They argued that several papers on randomized algorithms for online problems had used different notions of adversary. The different adversaries were then identified and formalized: oblivious adversary, adaptive online adversary, adaptive offline adversary. Among these, oblivious adversary is the weakest and adaptive offline adversary is the strongest. Among the several seminal theorems, they showed the following beautiful and simple theorem:

Theorem [BDBK+90]. If there exists randomized online strategy for a problem that is α competitive against an adaptive offline adversary, then there exists an α competitive deterministic strategy.

A natural question that arises in this context is whether this theorem can be made constructive. We show that it cannot. In fact, we show that there exists an online problem such that there is a very simple computable randomized strategy that is 1-competitive, but no deterministic computable strategy that is α-competitive for any

finite α.

We also show an interesting game-theoretic result which asserts that the BBKTW theorem is the tightest possible.

In my thesis, I also consider the following issue:

Consider a random boolean formula that approximately realizes a boolean function. Amplification (first proposed by Valiant) is a technique wherein several independent copies of the formula are combined in some manner to prove the existence of a formula that exactly computes the function. Valiant used amplification to produce polynomial size formulas for the majority function over the basis $\{\wedge, \vee\}$. Boppana then showed that Valiant achieved best possible amplification. We use amplification to show the existence of small formulas for majority when the basis consists of small(fixed) majority gates. The obtained formula sizes are optimal modulo the amplification method.

Acknowledgements

Given the time I have spent at SFU, it goes without saying that I owe a lot to a lot of different people. However incorrigible ingrate that I am, I am only going to thank those people whom it is absolutely necessary to thank.

I am grateful to Prof. Tiko Kameda to have accepted an intractable and stubborn person like me as a student. He has been extremely patient with my various intolerable idiosyncrasies, and has supported me financially during the course of this research. I am also extremely thankful to Xiaotie Deng to whom I owe many intellectually stimulating conversations, not just on infinite games, but also on such wide-ranging topics such as the applicability of bolshevism to chinese chess. It was with Xiaotie that I fleshed out the first part of my thesis.

Arvind Gupta deserves my sincere thanks for uncountably infinite number of reasons. He has been not just a supervisor, but also a good friend. He has given me sage advice on millions of things, not the least of which is how to make Indian cheese out of homogenized milk. Arvind and I have had a fruitful research relationship and I hope wherever I go after my Ph.D., I will be able to find people like Arvind.

Contents

Chapter 1

Introduction to Part I

1.1 Motivation

Consider the following problem, called the paging problem which arises in design of most operating systems: Let us assume that the main memory has k pages. Every time there is a request for a page, the operating system looks at the main memory, and if the requisite page is in the memory (a *hit*), the page is read by the requesting process. Otherwise (a *miss*), the required page is in the secondary memory, and it replaces *some* page of the main memory. There is a non-negligible cost associated with a miss, so we need to minimize the number of misses. Now the question is: In the case of a miss, which page in the main memory should be replaced? If the operating system knew in advance, which page in the main memory would not be requested for the longest time into the future, one could replace such a page. However decisions have to be made *online*, that is, as requests are received,and the future is not certain. So we need to design a strategy for page replacement that will do well under all circumstances. In order to measure the performance of such a strategy, we compare its cost (measured in terms of the number of misses) versus the cost of an optimal algorithm which knows all the page requests in advance.

1.2 Background

Let us assume we have a system which gets requests one at a time which must be served as soon as they are received. Further assume that each request can be served in one of several possible ways, and there is a cost associated with serving the request. The problem of servicing the requests so as to minimize the total cost is called an online problem. The method used to service the requests is an online strategy. If the strategy is computable, it is called an online algorithm. We define a *state* of the system to be all the information that must be known to determine the present and future behavior of the system. After serving the request, the system reaches a new state which also depends on the request, how the request is served and the old state of the system. The cost incurred in serving the request depends on the current state of the system, the request and the way in which the request is served. In the paging system example above, the state is all the pages in the main memory, and the cost in serving a given request is 0 in case of a hit and 1 othertwise. The cost of servicing a sequence of requests σ using a strategy A is simply the sum of costs of servicing each request using A and is denoted $cost_A(\sigma)$.

Sleator and Tarjan [ST85] introduced the notion of *competitive ratio* to measure the performance of online strategies. Competitive ratio of an online strategy A is defined to be the worst case ratio, over all finite request sequences σ, of the cost incurred by A on σ versus the cost incurred by an optimal strategy which knows all the requests in advance (or an *offline* strategy) in σ. To avoid trivialities, we will assume that the cost function is unbounded, that is, for every natural n, there is a request sequence σ, such that the cost of the best offline strategy over σ is bigger than n. In such a case, we can define a weaker notion of *competitiveness*. We say that a strategy A is *α-competitive* (in the weak sense) if for every request sequence σ:

$$\lim_{cost_{off}(\sigma)\to\infty} sup\frac{cost_A(\sigma)}{cost_{off}(\sigma)} \leq \alpha$$

where $cost_{off}$ is the cost function of the best offline strategy, and d is a constant independent of σ.

Mannasse, Mcgeoch and Sleator [MMS88] defined the following canonical online

problem, called the k server problem:
Let M be a metric space, and let k *servers* be placed on different points of M. *Requests* come, one at a time, at points of the metric space. A request is said to be *served* if one of the servers moves to the position of the request. The cost incurred in serving the request is the distance moved by the server. The cost to serve a sequence of requests is defined in the obvious manner. The problem is to design a strategy that compares well (in terms of cost) against the optimal offline strategy.

This problem models a wide variety of problems such as the paging problem, where M is a uniform metric space. [MMS88] also show a lower bound of k for the competitive ratio. For $k = 2$, they also show an upper bound of 2 which completely solves the 2-server problem. Much effort has been spent in designing good algorithms when $k > 2$. Fiat, Rabani and Ravid [FRR90] exhibit an algorithm that achieves competitive ratio of $(O(e^{k \log(k)})$. Before this, no algorithm was known which achieved a constant ratio independent of the cost of the sequence.

Having discussed deterministic online algorithms, we now consider what happens if we give coin-flipping capability(randomization) to the algorithms. Before we start to meaningfully address this question, we have to clarify what randomization means for online problems. As we will see, depending on what we mean by randomization, it either helps us immensely or it does not help at all.

Ben-David et al clarified the notion of randomization for online problems in their seminal paper [BDBK+90]. They observed that online problems could be studied in terms of two person games between a player and an adversary. The adversary gives requests and the player serves the requests in one of several possible ways. The adversary also serves the requests and can use an optimal algorithm to serve its own requests. The *competitive ratio* at any point is the ratio of the player's cost to the adversary's cost. Suppose the player claims that he can achieve a competitive ratio of α. The job of the adversary is to make the competitive ratio bigger than α, so she gives requests keeping this in mind. If we are using the weak notion of competitive ratio, the adversary has to make the ratio bigger than α infinitely often, whereas if the stronger notion is used, then she only has to make the ratio bigger than α once.

In the deterministic strategy, the adversary knows how the player is going to behave at any particular instant, so we can assume that she chooses the request sequence in advance. However, if the player adopts a randomized strategy (that is, he has the ability to throw coins and decide how to answer requests accordingly), then the adversary can either decide the request sequence in advance, or adapt her requests to the answers given by the player. Clearly, the second kind of adversary - the *adaptive* adversary, is at least as strong as the first - the *oblivious* (also called the *weak* adversary), and in fact is strictly stronger than the oblivious adversary as was shown in [BDBK+90]. In the case of the oblivious adversary, since she knows the request sequence in advance, she uses the best offline algorithm to service her own requests. However the adaptive adversary does not know the request sequence in advance and only comes to know what requests she should give incrementally as the game progresses. Therefore it may make a difference whether she serves her own requests online or offline in computing her costs. The first kind of adaptive adversary is called a *medium* adversary and the second is called a *strong* adversary. We now have three different notions of randomization corresponding to the three different adversaries: oblivious or weak adversary, medium adversary and strong adversary. [BDBK+90] show the following theorem in their paper:

Theorem [BDBK+90]: For an online problem, if there exists a randomized strategy that is α-competitive against a strong adversary, then there is a deterministic α-competitive strategy.

This theorem shows that randomizing against a strong adversary does not help.

[MMS88] showed that if the weak notion of adversary is used, one can achieve a competitive ratio of $\log k$ for the k-server problem when the underlying metric space is uniform. On the other hand, the argument by [MMS88] can be modified to show that any randomized online strategy for the k-server problem must have a competitive ratio of at least k against a medium adversary. This shows that the weak adversary is strictly weaker than the medium adversary.

1.3 Overview of the Results

In this thesis, we show that the derandomization thesis of [BDBK+90] cannot be constructivized. In particular, we show the existence of an online problem for which there is a 1-competitive randomized algorithm (that is, a computable strategy) against any offline adversary, but no α–competitive deterministic algorithm (that is, a computable strategy). To prove this main theorem, we need to define the notion of *semicomputable determinacy* which is analogous to the notion of *determinacy* in classical infinite game theory except that one player has access to only computable strategies and the other has access to all possible strategies. This models the notion of a computer playing against an infinitely powerful adversary. We will define these notions more carefully in a later section. We then prove some characterization theorems about semicomputable determinacy which help us establish our main results. We also show an elegant result about classical infinite game theory. The derandomization thesis of [BDBK+90] applies to any determinate game (as we shall see, online problems are special cases of determinate games). We show that the derandomization thesis does not hold for indeterminate games, that is, we construct an indeterminate game for which there exist randomized winning strategies for both players. This proof relies on the Axiom of Choice and the Continuum Hypothesis. Before we can meaningfully discuss our results, we give some preliminary background on topology and measure theory.

1.4 Some Basics of Topology and Measure Theory

Topology is a branch of mathematics that deals with mathematical properties that are invariant under deformations and continuous transformations. For the sake of illustration, we will study the real line. The $\epsilon - \delta$ definition of continuous functions on the real line says that a function f is continuous at x_0 iff for every $\epsilon > 0$ there is a $\delta > 0$ such that, whenever x differs from x_0 by at most δ, $f(x)$ differs from $f(x_0)$ by at most ϵ. Note that, as continuity is a topological property (that is, a continuous function remains continuous when we deform it and vice-versa), the preceding definition is not very satisfactory, as it depends on the notion of a metric on the real line. To remedy this situation, we observe that, if a function f is discontinuous at some point x_0 (say when one approaches x_0 from the left, the function value is f_1 and when one approaches from the right, the function value is f_2 where $f_1 < f_2$), then the inverse image of the open interval (f_1, f_2) (under f) maps to the single point x_0. This hints at the following definition of continuity: f is continuous iff whenever V is a disjoint union of open intervals, $f^{-1}(V)$ is also a disjoint union of open intervals. One can show that this definition of continuity and the $\epsilon - \delta$ definition of continuity are equivalent, and therefore one can dispense with the $\epsilon - \delta$ definition.

In the real line situation, the disjoint unions of open sets are called *open sets*, and the collection of all open sets is called the *natural topology*. In general, we can have any arbitrary ground set X and define a *topology* on it as a collection of *open* sets(which are subsets of X) which have the following properties:

- ϕ is open and X is open
- Arbitrary union of open sets is open
- Finite intersection of open sets is open

Observe that disjoint unions of open intervals satisfy all these properties. Also observe that the word 'finite' in the third condition cannot be replaced by 'arbitrary' as the intersection of open intervals $(\frac{-1}{n}, \frac{1}{n})$ for all positive integers n is $\{0\}$ which is clearly not a union of disjoint open intervals. Also notice that there are trivial collections of subsets of X which satisfy the above properties, such as the collection $\{\phi, X\}$ and the collection of all subsets of X. Under these two *topologies*, we do not get anything interesting. To make things non-trivial, we have to introduce what are called separation axioms. But we will not dwell on the niceties of such separation axioms, and instead talk about measures and measure spaces.

To fix thoughts, consider again the real line. We wish to generalize the notion of length of an interval to arbitrary subsets of reals. One such generalization which is used in Lebesgue integration theory is the *outer measure*. If Y (a subset of reals) is an open set under the natural topology (that is, Y is a union of disjoint intervals), then the outer measure of Y is defined to be the sum of lengths of the disjoint intervals contained in Y. Otherwise the outer measure of Y is defined to be the infimum over all open sets U which cover Y of the measure of U (intuitively it is the measure of the 'smallest' open set that is a superset of Y). It turns out that the outer measure satisfies the following properties:

- Subadditivity: The measure of a countable union of sets is less than or equal to sum of the measures of individual sets.
- Translation invariance: If every element of a set is translated by the same amount, the measure of the resulting set remains the same.
- Compatibility with the notion of length: The measure of an interval (open, half-open or closed) is its length.

The second and third properties are definitely desirable, but the first is not as strong as we would like it to be. We would like to have the measure of countable union of mutually disjoint sets to be exactly the sum of the measures of the individual sets. It turns out that if we restrict ourselves to *measurable* sets, that is, those sets Y for which the measure of the 'largest' open set contained in Y and the measure of the 'smallest' open set containing Y is the same, then we can change 'subadditivity' to 'additivity'. It also turns out that measurable sets are closed under complementation and countable union.

Consider the smallest collection of sets which contains all the open sets and is closed under complementation and countable union. The sets in this collection are measurable under any measure. These sets are given a special name: *Borel* sets.

1.5 Background on Infinite Game Theory

1.5.1 Definitions and Notations

An infinite game is described as an infinite tree on which two players make their moves in turn, starting at the root. The tree may be finitely or infinitely branching. We only consider the case where the tree is finitely branching. The game is played by 2 players, Player I *our player* and Player II the *adversary*. The set of all the infinite paths is partitioned into two subsets A and B, where A is the winning set for our player and B is the winning set of the adversary. Starting from the root a_0, the adversary chooses a child a_1 of a_0, and then our player chooses a child a_2 of a_1, and so on. In essence, the adversary and our player alternately choose the child of the current node. Our player wins if he can force the infinite path $(a_0, a_1, ...)$ to be in the set A; otherwise the adversary wins. A strategy τ (σ) for the adversary (our player) corresponds to a pruned tree T_τ (T_σ) from the original game tree on which each branching at even (odd) levels is pruned to allow at most one possible child. The resulting play (σ, τ) for a given τ and a given σ is specified by a path in the game tree. We call σ a *winning strategy* for our player if for every τ, (σ, τ) belongs to A. The winning strategies for the adversary are defined similarly. Finally, the game is *determinate* if either player has a winning strategy.

We now define a topology on the set of all plays. A subset S of all the paths is defined to be *open* iff for every path $(a_0, a_1, ...) \in S$, there exists a number n such that, for all $b_{n+1}, b_{n+2}, ...$,

$$(a_0, a_1, ..., a_n, b_{n+1}, b_{n+2}, ...) \in S.$$

A set of paths is *closed* if it is the complement of an open set.

A game is *open* (*closed*) if the winning set of our player is open (closed). A set is called F_σ (the σ in the index is not to be confused with the request sequence σ) if it is a union of countably many closed sets. We call a game F_σ if the winning set of our player is F_σ. In their classical paper on infinite games [GS53], Gale and Stewart showed that all open and closed games are determinate and that there exists a game that is indeterminate. Martin [Mar75] then showed that all Borel games are determinate (a game is *Borel* if the winning set of Player I (or II) is Borel under the topology defined above). Observe that, in classical infinite game theory, there is no restriction on the strategies in terms of computability.

In [BDBK+90], online problems are formulated as finite games, and Raghavan and Snir [RS89] give an infinite game formulation. Depending on the criteria of competitiveness, one may get different winning sets for an online problem. We provide a brief overview of this formulation.

Suppose that the adversary chooses requests from the request set $\{r_1, r_2 ....\}$ and that our player can answer from the answer set $\{a_1, a_2 ...\}$ (called respectively, the *choice sets* of the adversary and the player). The online problem is then modeled as the following infinite game.

The adversary and our player alternate between choosing a request from the request set and an answer from the answer set, respectively. Since both the adversary and our player can choose from several possible requests and answers respectively, this then describes an infinite game. The cost of our player OC at the stage of the game where our player has just answered a request, is a function of the history of the requests given by the adversary and the answers provided by our player and the answer currently given. We assume that OC is a monotonically non-decreasing function of the history of request-answer sequence to make this formulation of online problems meaningful. That is, $OC(r_1, a_1, ..., r_n, a_n) \leq OC(r_1, a_1, ..., r_n, a_n, r_{n+1}, a_{n+1})$. The cost of the adversary, denoted by AC is also a function of the history of the request sequence seen so far. However AC is only a function of the requests in this history and not the player's answers. In the case of the weak and strong adversaries, the function that defines AC is simply the optimal offline cost of serving the request sequence embedded in the request answer sequence seen so far, and in the case of the medium adversary, it is the online cost of serving the request sequence, when the medium adversary uses a particular online strategy to serve its own requests. At each stage of the game, the adversary's objective is to make the competitive ratio (so far) as large as possible, and our player's objective is to make the competitive ratio as small as possible. For instance, if our player claims that he can achieve a competitive ratio of α (in the strong sense), then our player wins if he can ascertain that at every stage of the game, the ratio achieved between our player's cost and adversary's cost so far is at most α, and the adversary wins if she can make this ratio bigger than α at least once. If the player claims that he can achieve a competitive ratio of α in the weak sense, then the adversary needs to make the ratio between our player's cost and the adversary's cost bigger than α infinitely often.

Let us now formalize precisely what we have just described. A simple $\alpha-$competitiveness requirement is defined by $OC \leq \alpha \cdot AC$. We will call this the strong *competitive* condition. When the cost function is accumulative, e.g., in the case of the server problem, strong competitiveness will define a closed game: The adversary wins iff the play reaches a node at which the condition is violated. Raghavan and Snir use a formulation which allows an arbitrary additive constant. This gives rise to a game with a winning set in F_σ:

$$\cup_i \cup_j \cap_{k \geq j} \{(x_0, x_1, \cdots) : OC(x_0, \cdots, x_k)$$

$$-\alpha \cdot AC(x_0, \cdots, x_k) \leq i\}.$$

Thus, infinite paths of constant cost are in the winning set. We call this the weak *competitive* condition.

Without loss of generality, we assume that each player has two choices at each turn of their play. A randomized strategy for our player is a function that makes an assignment of probability to the choices $a0, a1$ depending on the position of the node on the game tree. We say a randomized strategy is *computable* if the probability distribution on the choice space is a computable function. As noted in [RS89, HT89], a statement about a randomized strategy is true if it is true for all adversary strategies. Thus, given a randomized strategy, we consider each deterministic strategy τ of the adversary, and the induced probability distribution for the pruned tree T_τ. We specify a topology and a probability measure on the smallest $\sigma-$algebra (a *σ-algebra* is a collection of subsets of a given ground set, which is closed under countable unions and complementation) generated by the topology, by specifying the measure on all the basic open sets: A basic open set U is specified by a node x on T_τ such that it contains all the paths passing through x and its probability measure is the probability the randomized strategy reaches x. The measure is extended to all the Borel sets in the topology by a standard method described in any classical textbook in Measure Theory such as [CT78]. When we specify an adversary strategy τ, a similar method is applied to define the conditional distribution on the pruned trees T_τ. Again, a randomized strategy is $\alpha-$competitive almost surely, iff for all the pruned tree T_τ, it is $\alpha-$competitive almost surely with respect to this probability distribution.

Chapter 2

Main Theorems

We mentioned in the last chapter that [BDBK+90] showed the following:

Theorem 2.0.1. *[BDBK+90]: Given an online problem, whenever there is an α-competitive randomized strategy against the strong adversary, there is an α-competitive deterministic strategy.*

We call this the *Derandomization Hypothesis.* Now a natural question that arises is whether this derandomization can be constructivized. We show in this chapter that such is not the case. In fact, we show the following:

Theorem 2.0.2. *: There is an online problem for which there is a simple randomized computable 1-competitive strategy, but no α-competitive deterministic computable strategy for any α.*

The main thrust of this chapter is to prove the above theorem. However, in order to prove the theorem, we will have to develop the concept of *semicomputable determinacy* which is interesting in its own right.

In order to prove their theorems, [BDBK+90] and [RS89] use determinacy of finite and F_σ games respectively. The basic idea is that, because online problems can be modeled as F_σ games [RS89], (for the weak notion of competitive ratio), these games are determinate. We essentially need to show that, whenever there is a randomized winning strategy (say, for player I) in these games, then there is a deterministic winning strategy for player I. Assume not. Then by the determinacy of these games, there is a deterministic winning strategy τ for player II. Now if player II plays according to τ, then no matter what probability distribution player I puts on his answers, he can never win, as all the paths in τ are winning for II. This is a contradiction to the hypothesis.

As we need to establish that the computable derandomization hypothesis does not hold, we define the notion of semicomputable determinacy. The idea is that we will allow our player to use only computable strategies, and allow the adversary unlimited power. This models the worst case behavior of computable online problems.

Definition: An infinite game is *semicomputably determinate* (in terms of player I or our player) iff either there is a computable strategy for our player that wins against all strategies of player II (the adversary) or there is an adversary strategy that wins against all computable strategies of our player.

2.1 Semicomputable Determinacy.

While all Borel games are determinate [Mar75], we would like to know, under what topological conditions, a game is semicomputably determinate. First, we have

Theorem 2.1.1. *There is a semicomputably indeterminate closed game.*

We give both players two choices of actions: r_0, r_1 for the adversary and a_0, a_1 for our player. We first give some intuition on the proof of the theorem. We need to partition the set of all the paths into two sets A (the winning set for our player which is closed) and B (the winning set for the adversary) such that for each computable strategy σ of our player, there exists an adversary strategy τ such that (σ, τ) is in B (call this condition C1), and for each strategy τ of the adversary, there exists a computable strategy of our player σ such that (σ, τ) is in A (call this condition C2). C1 and C2 force certain plays to be put in A and B, respectively, and we should make sure that (A, B) is a partition. Moreover, we want a construction which makes A a closed set.

Observe that the indeterminacy proof given in [GS53] cannot be translated into this case. Our result is obtained via a new method which may be useful in other similar situations.

We construct A and B in stages. Initially both are empty. Let the computable strategies of our player be ordered as $\sigma_i, i = 0, 1, 2, \cdots$. Say that a strategy σ is ***killed*** in stage j, if we put $(\sigma, \tau') \in B$ for some strategy τ' of the adversary in stage j. Similarly for an adversary strategy. At each stage, we kill at least one σ and perhaps an uncountable number of τs, so that A and B remain disjoint and make sure that each σ and each τ is killed in some finite stage without destroying the disjointness criterion. A semicomputably indeterminate game is thus constructed. The construction will guarantee that the A so constructed is closed. Now we give the technical details of the result.

Proof. For simplicity, we assume each player has two choices at each step of their plays: The adversary has move r_0, r_1 and our player has move a_0, a_1. We list all (computable) strategies of our player in the set

$$\Sigma = \{\sigma_0, \sigma_1, \cdots, \sigma_n, \cdots\}$$

such that σ_0 is the strategy that choose move a_0 all the time. Informally, we need to construct a game with winning sets A for our player and B for the adversary such that

- [C1] for each $\sigma \in \Sigma$ there is a $\tau \in \mathcal{T}$ such that $(\sigma, \tau) \in B$;
- [C2] for each $\tau \in \mathcal{T}$ there is a $\sigma \in \Sigma$ such that $(\sigma, \tau) \in A$.

Construction of A, B. Initially, we set

$$A = (r_0, a_0, r_0, a_0, \cdots, r_0, a_0, \cdots)$$

and $B = \varnothing$. Let us denote the root of the game tree to be Level 0. Incrementally assign level number to the tree. We will prune the tree in levels. First Level 1 is processed, and then we show inductively how to process Level n for each $n = 2, 3, \cdots$.

- Level 1. Denote by $\mathcal{T}_1$ all the strategies making first request as r_1. We assign the paths $\{(\sigma_0, \tau) : \tau \in \mathcal{T}_1\}$ to the set B_1 and all the other paths starting with r_1 are assigned to A_1. Update $A \leftarrow A \cup A_1$ and $B \leftarrow B \cup B_1$. Thus, Condition [C1] holds for σ_0 and Condition [C2] holds for all $\tau \in \mathcal{T}_1$.

- Level $2i$. The following is the inductive assumption for the pruning process at the end of Level $2i$. Each remaining node at Level $2i$ is a descendent of the adversary playing r_0 at all the past i requests. Thus, each node can be denoted by an i−bit binary number corresponding to the plays made by our player from the root to the node in the natural way. That is, each remaining node at level $2i$ corresponds to the player answering in one of the 2^i possible ways to the requests of the adversaries. Each such node can thus be encoded as an i-bit binary number where the jth position is a $0(1)$ if the player answers the jth request by r_0 (r_1). We say that a strategy σ of the player is *consistent* with an i bit vector x if the player answers according to x in the sense just described. At node j, Σ_j represents all our player strategies which are consistent with j up to this node. Σ_j's, $j = 0^i, \cdots, 1^i$, form a partition of the remaining members in Σ which does not satisfy Condition [C1] yet. All the adversary strategies remaining at node j are those which make i consecutive requests of r_0's, when played against our player which answers j correspondingly. We will denote them by $\mathcal{T}_j$.

- Level $2i+1$. Consider each node independently. Without loss of generality, let us look at node 0^i. Let

$$\Sigma_{0^i} = \{\sigma_{0^i{}_0}, \sigma_{0^i{}_1}, \cdots, \sigma_{0^i{}_n}, \cdots\}.$$

Denote by $\mathcal{T}_{0^i1}$ all the strategies in $\mathcal{T}_{0^i}$ which makes the $(i+1)$−st request as r_1. We assign the paths $\{(\sigma_{0^i{}_0}, \tau) : \tau \in \mathcal{T}_{0^i1}\}$ to the set B_{0^i1} and all the other paths starting from 0^i and continuing with r_1 are assigned to A_{0^i1}. At Level $2i+2$, according to the choice of our player, the strategy set for our player is partitioned into two subset $\Sigma_{0^i0}, \Sigma_{0^i1}$, where the lists for $\Sigma_{0^i0}, \Sigma_{0^i1}$ keep the same order as the list in Σ. Thus, Condition [C1] holds for $\sigma_{0^i{}_0}$ and Condition [C2] holds for all $\tau \in \mathcal{T}_{0^i1}$. We also do the similar operations on all the nodes j of i bits. For all j of i bits, Condition [C1] holds for σ_{j_0} and Condition [C2] holds for all $\tau \in \mathcal{T}_{j1}$. Update the set A and B by assigning $A \leftarrow A \cup_{j=0^i}^{1^i} A_{j1}$ and $B \leftarrow B \cup_{j=0^i}^{1^i} B_{j1}$.

Correctness Proof. We now prove that Conditions [C1] and [C2] are true for all adversary strategies and our player's strategies. Notice that our player's strategies are first enumerated in the set Σ and the ordering is kept when it is partitioned at each level. For the first strategy σ in Σ_j, there is an adversary strategy τ such that $(\sigma, \tau) \in B$, according to our pruning process. Therefore, for each $i = 1, 2, \cdots$, σ_i satisfies Condition [C1] no later than Level $2i$ in our construction. To prove that Condition [C2] holds for all adversary strategies, we consider two cases: one is the case the adversary plays r_0 all the time; the other is the case the adversary plays an r_1 at least once for some strategy. The first case is done by the initial assignment of the set A. For the second case, we notice that, for any other strategy τ of the adversary, it will play an r_1 at least once for a strategy of our player at a finite level. If the strategy of our player is not a computable strategy, we can simply truncate the infinite strategy at that finite level and append it by always playing a_0. This will be a computable strategy $\sigma(\tau)$. Suppose j is the node for the first step the adversary plays an r_1, then the adversary strategy τ will lose to $\sigma(\tau)$ at one path in $\mathcal{T}_{j1}$. **QED**

In contrast, all open games with a finite choice space for the adversary have enough mathematical structure to make them semicomputably determinate.

Theorem 2.1.2. *All open games are semicomputably determinate, if the choice space for the adversary is finite.*

Proof. Assuming that there is no computable strategy for our player that wins against all adversary strategies, we will show that there exists an adversary strategy that wins against all computable strategies of our player. Let the adversary's choice space be r_1 ,..., r_n. We claim that there is a request r by the adversary such that no computable strategy of our player can win if the adversary uses r as the first request. If this were not so, then for every request r_i ($1 \leq i \leq n$), there is a computable strategy σ_i of our player such that σ_i is a winning strategy against any adversary strategy which uses r_i as the first request. Thus the following computable strategy for our player:

- If the first request is r_1, play σ_1 else
- ....
- If the first request is r_n, play σ_n

wins against any adversary strategy, a contradiction. A partial adversary strategy at stage i is an adversary strategy when the game tree is truncated at stage i (that is, at adversary's ith move). We say that a partial adversary strategy α at stage $i+1$ *extends* a partial adversary strategy β at stage i, if α is the same as β upto stage i. We prove by induction on stages that at any stage i, there is a partial adversary strategy α_i at stage i, which extends α_{i-1}, such that if the adversary plays according to α_i until stage i, no computable strategy of our player can win against it. Towards a contradiction, assume that there is a path p (which is necessarily finite) in α_{i-1} such that there is an answer by our player at stage $i-1$ such that if p is extended by this answer, then for every request r_k there is a computable strategy σ_k of our player such that it wins against any adversary strategy which follows the partial path p. This, as before gives us a computable strategy for our player that wins against any adversary strategy which extends α_{i-1}. Then α_{i-1} is not a non-losing strategy for the adversary, a contradiction. The union of α_i's gives a complete adversary strategy that cannot lose against any computable strategy of our player. As the adversary's winning set is closed, this non-losing strategy of the adversary is also a winning strategy for the adversary. **QED**

We also introduce a stronger theorem for the case where the choice space of our player is also restricted to be finite.

Lemma 2.1.3. *Suppose that the choice spaces for the adversary and the player are both finite. Then for every open game, either there is an adversary strategy which wins over all the strategies of our player, or there is a strategy that can be encoded by a finite state machine of our player which wins over all the strategies of the adversary.*

Proof. Suppose no adversary strategy wins over all computable strategies of our player. Since open games are determinate [GS53], [Mar75], there is a strategy σ for our player which wins against all adversary strategies (although σ may not be computable). Consider the pruned tree T_σ. Since A is open, for each infinite path in T_σ, there is a node x on the path such that all the paths passing through x are in A. We can thus remove all the children of x, and all the siblings of x as well as their children, from the tree T_σ without changing the win/lose situation of the tree.

The game tree thus pruned has no infinite path. Since both players have only a finite number of choices at each node, the pruned tree is finite. Thus, if there is no adversary winning strategy, our player can simply code the structure of the pruned tree into a finite state machine and choose its moves accordingly. **QED**

We thus have the following corollary.

Corollary 2.1.4. *For closed games, either there is a finite state adversary strategy which wins against all strategies of our player or there is a strategy of our player which wins against all adversary strategies, if the choice spaces for both the adversary and our player are finite.*

For the notion of strong competitiveness, the winning set of our player is closed. If there is no deterministic winning strategy for our player, then the winning strategy of the adversary will enable us to prune the tree to a finite tree, according to the above corollary. We will thus easily conclude that there is no competitive randomized strategy for our player. The result of [BDBK+90], [RS89] for infinite games follows immediately. The above corollary also implies that, if we allow our player to use unlimited power, we only need to look for lower bounds by adversaries with a simple computational power: finite state machines.

2.2 Applications to Online Algorithms.

While online problems are formulated as closed and F_σ games, we would also like to formulate closed and F_σ games as online problems such that there is a winning strategy for our player in a given game iff there is an $\alpha-$competitive online algorithm for the corresponding online problem. This may not be true in general. However, for games constructed in this thesis, we want to make sure that the above condition is satisfied.

First, we construct a game similar to the one given in the last section for this goal.

Theorem 2.2.1. *There exists a semicomputably indeterminate F_σ game such that there is a computable randomized strategy for our player, which wins almost surely.*

Before going into the proof, we notice that the following two corollaries derived from the theorem give us the desired results for strong competitiveness and weak competitiveness, respectively.

Corollary 2.2.2. *There exists an online problem for which there is no computable deterministic strong competitive strategy but there is a computable, randomized, strongly* $1-$*competitive strategy almost surely (that is, with probability 1).*

Corollary 2.2.3. *There exists an online problem of accumulative cost such that there is a computable, randomized, weak* $1-$*competitive strategy almost surely, but there is no computable, deterministic, weak competitive strategy.*

Proof. We follow a similar construction to the game in the last section. The change is that in forming the sets A and B, we put all the paths irrelevant to the indeterminacy into A instead of B. Initially, we will put all the paths (σ_i, τ_0) into A, where τ_0 is the strategy that always requests r_0. In level one, we will choose one strategy $\tau^1 \in \mathcal{T}_1$ and put (σ_0, τ^1) in B_1 and put all the other paths starting with r_1 into A_1. Similarly, at node j, we will choose one strategy $\tau^j \in \mathcal{T}_j$ and put (σ_{j_0}, τ^j) in B_{j1} and all the other paths starting at node j and continuing with r_1 into A_{j1}. All other constructions follow the same pattern. Similar to the proof in Theorem 1, the game can be shown to be semicomputably indeterminate.

Consider the randomized algorithm which always chooses a_0, a_1 with probability $0.5 : 0.5$. We claim that this simple (computable) randomized algorithm wins with probability with probability 1. Consider a pruned tree T_τ corresponding to an arbitrary adversary strategy τ. From the construction of the winning set A, when the adversary first chooses r_1, the branch of T_τ starting from that node will contain exactly one winning path for the adversary and our player wins almost surely starting from that node. With this observation, we further prune T_τ as follows: Start from the root until a request r_1 is encountered and delete the branch after that node. Thus, the only infinite paths of the newly pruned tree will contain request r_0 only. Since those paths are all in the winning set of our player, the randomized strategy wins almost surely in T_τ. Because the game is semicomputably indeterminate, any given computable strategy is doomed to lose to some adversary. **QED**

We may have different formulations of infinite games as online problems but we shall use the two formulations defined below for our discussion.

If we adopt the concept of strong competitiveness, there is no need to require the adversary's cost grow as the game is played. For the strong condition, intuitively, we want to simply assign cost *zero* to each infinite path in the winning set of the player, and cost *one* to each path in the adversary winning set. That answers our question immediately. Strictly speaking, however, the preceding construction of the cost function is not very precise, as it does not make sense to assign costs to infinite paths when we are talking about online problems. Games for these problems are either closed or F_σ, as we have seen above. In the case of closed games, we can assign a cost of 1 to those nodes which are descendents of the defining nodes of the basic open sets comprising the winning set of the adversary (a *defining node* of a basic open set is the deepest node through which all the paths of the basic open set pass), and the rest of the nodes are assigned the cost of 0. Similar assignments of the cost function can be done for F_σ games. Corollary 2.2.2 follows immediately by assigning such cost function since the randomized strategy has cost 0 almost surely.

However, one may want to have an accumulative cost function such that it increases smoothly as the play proceeds and it is unbounded. For each request sequence, we eliminate r_0 at the head of the sequence until the request r_1 is at the beginning. We call the remaining request sequence the suffix. If all the requests leading to a node are r_0, the cost of reaching this node will be 0. For other nodes, the cost will be the number of the requests on *the suffix* before the node which are coincident to an infinite path in the winning set of the adversary. The cost of our player will be the cost of the node it is on. The cost of the adversary will be the minimum cost over all the nodes with the same request sequence (i.e., we consider an offline adaptive adversary). The cost of an infinite path for our player is defined as the limit of the cost of its intermediate nodes. The cost of the adversary is again defined as the minimum cost over all the infinite paths with the same request sequence.

To prove Corollary 2.2.3, we want to use the accumulative cost function defined above. First, let us consider an adversary which makes its first request on r_1. In the construction of the game, we notice that there is only one winning path for the adversary from this node on. The adversary's cost will thus be 1 since it can avoid that path by serving the first request with an answer which is not on that path. Since all the paths except the one in the adversary's winning set in this pruned tree has bounded cost, the randomized algorithm is 1−competitive almost surely. For the pruned tree corresponding to each adversary strategy, we can take those nodes for which all but the last request are r_0 and the last request is r_1. From the construction of the game, the conditional distributions from those branches on are the same as the above case. Since paths with all requests being r_0 are in the winning set of our player anyway, the result follows. We also notice that similar statements hold when we use the notion of *expected value* for competitiveness instead of *almost surely* except that the expected cost of the adversary is bounded. We can make the expected cost of the adversary unbounded if we use instead the cost function which is the product of the original cost function and the number of requests starting from the first r_1 in the path.

In this case, we achieve a competitive ratio of three, whereas no deterministic algorithm can achieve a finite ratio. In fact, for the case where the first request is r_1, the best strategy for the adversary is to put requests only when our player answers so that the resulting path coincides currently with the unique winning path of the adversary, stopping whenever our player deviates from this path. The randomized algorithm of our player will incur a cost of i^2 at level i. For this adversary strategy, our player incurs this cost of i^2 with probability $\frac{1}{2^i}$ for all $i \geq 1$, and the adversary incurs a cost of i with probability $\frac{1}{2^i}$ for all $i \geq 1$. So the expected cost of our player is $\sum_{i=1}^{\infty} i^2 * \frac{1}{2^i} = 6$, and the expected cost of the adversary is $\sum_{i=1}^{\infty} i * \frac{1}{2^i} = 2$, resulting in a competitive ratio of three. Observe that the expected cost of the adversary goes to infinity if the adversary strategy truncates at the same level regardless of how our player moves.

If a further restriction that the winning set be closed is imposed, we can choose an open set of measure ϵ which contains (σ_0, τ^1) to put in B_1 and do similar things to all the B'_{j1}s.

Theorem 2.2.4. *For any $\epsilon > 0$, there exists a semicomputably indeterminate closed game such that there is a computable randomized strategy for our player which wins with probability $1 - \epsilon$.*

As an immediate corollary of Lemma 2.1.3, we have

Theorem 2.2.5. *The computable derandomization hypothesis holds for all open games.*

2.3 Indeterminacy and Randomization

In the section, we discuss indeterminate infinite games and the power of randomization in this case.

The results of [BDBK+90] and [RS89] basically say that, for a determinate game, whenever there is a randomized strategy for Player I, which wins with probability 1, there is a deterministic winning strategy for Player I. If this result were extendible to the indeterminate games, it would mean that, for every indeterminate game, there is no randomized winning strategy for any of the players. The following theorem shows that this does not happen. Moreover, this artificially constructed game has another counter-intuitive implication: Even though Player I has a randomized strategy winning almost surely against all the deterministic strategies of Player II, that randomized strategy does not necessarily win almost surely against all the randomized strategies of Player II. This theorem, which is apparently new, is a pure game theoretic result, and hence might be of interest to descriptive set theorists and game theorists. For a good introduction on infinite games, we refer the reader to [Mos80]. The book by Blackwell and Girshick [BG64] is a good reference for randomized (or mixed) strategies.

Theorem 2.3.1. *Assuming the Axiom of Choice and the Continuum hypothesis, there is an indeterminate game for which Player I has a randomized winning strategy which wins almost surely against any deterministic strategy of Player II, and vice-versa.*

The theorem shows that the [BDBK+90] and [RS89] result is the best possible in the sense that if we drop the condition that the game be determinate, the derandomization hypothesis does not always hold. This result assumes the Axiom of Choice and the Continuum Hypothesis. The Axiom of Choice seems necessary here because it is not even known if indeterminate games exist in the absence of the Axiom of Choice.

Proof. We assume that each player has two choices at any point in the game. By Axiom of Choice, we can well-order Player I's deterministic strategies as σ_α for $\alpha < 2^{\aleph_0}$, and Player II's deterministic strategies as τ_β for $\beta < 2^{\aleph_0}$.

Our randomized strategy for either the Player I or II (δ and γ, respectively) assigns a probability of 0.5 to each of the two possible moves. We now construct the winning set for Player I and that for Player II so that both these strategies are winning strategies if the other player uses only deterministic strategies.

We need to satisfy the following conditions:

1. For each deterministic strategy σ of Player I, only countably many paths in the pruned tree corresponding to σ belong to the winning set of Player I and the rest belong to the winning set of Player II.

2. For each deterministic strategy τ of Player II, only countably many paths in the pruned tree corresponding to τ can belong to the winning set of Player II.

It is clear that if we can satisfy these conditions, then δ wins with probability 1 against any deterministic strategy of Player II, and γ wins with probability 1 against any deterministic strategy of Player I (Each path has probability measure 0, and by the countable additivity of probability measure, countably many such paths will have measure 0).

We say a deterministic strategy σ is *killed* if we can satisfy Condition 1 for this σ (define killing of τ symmetrically). We kill σ's and τ's in stages. At stage $\alpha < 2^{\aleph_0}$ we kill σ_α and then τ_α making sure that the winning sets of the two players are disjoint. We denote the winning set of Player I by A and the winning set of Player II by B. Initially, they are empty. They are updated in each stage by transfinite induction on the stages.

At stage α, we put in B all paths of the pruned tree T_{σ_α} corresponding to σ_α, that have not already been put in A. Then we put in A all paths of the pruned tree T_{τ_α} corresponding to τ_α, that have not already been put in B.

This completes the construction. It is easy to see that the disjointness condition of A and B is automatically satisfied when these sets were constructed. We prove

Conditions 1 and 2 by transfinite induction on stages. To verify Condition 1, consider stage α. In the pruned tree T_{σ_α} for Player I's strategy σ_α, the paths already put in A are $T_{\sigma_\alpha} \cap A$. Since $A \subseteq \cup_{\beta<\alpha} T_{\tau_\beta}$, $T_{\sigma_\alpha} \cap A \subseteq \cup_{\beta<\alpha} T_{\sigma_\alpha} \cap T_{\tau_\beta}$. By Continuum Hypothesis, each $\alpha < 2^{\aleph_0}$ is either finite or countable. $T_{\sigma_\alpha} \cap T_{\tau_\beta}$ is a single path. Therefore, $T_{\sigma_\alpha} \cap A \subseteq \cup_{\beta<\alpha} T_{\sigma_\alpha} \cap T_{\tau_\beta}$ contains only a countable number of paths. This proves Condition 1. Condition 2 can be proven similarly. **QED**

We notice that the above proof still works with minor modifications even if the Continuum Hypothesis is replaced by a strictly weaker axiom, Martin's Axiom. One of the consequences of Martin's Axiom is that for any cardinal κ strictly between $\aleph_0$ and $2^{\aleph_0}$, the union of κ sets (as subsets of R) of Lebesgue Measure 0 has Lebesgue measure 0. The topology that we use for the game tree is similar to the real line, and the probability measure on the paths induced by γ or δ is similar to the Lebesgue Measure. So this consequence applies to our case.

Although the randomized strategy δ for player I (γ for player II) wins against all deterministic strategies of player II (player I), it does not win against all randomized strategies of player II (player I). In particular, δ does not win against γ (and vice versa). Perhaps the power of randomization in this case results from its easy access to all deterministic strategies at once.

We have addressed the situation where one player uses randomized strategy and the other uses deterministic strategy. What happens when both use randomized strategies? Is it possible to obtain an equilibrium solution? That is, is there a pair of randomized strategies of the players such that none can gain by deviating from this randomized strategy? The problem has been long open when the choice space is continuous [Mos80]. We conjecture that this is true for games with *universally measurable winning* sets. A set is *universally measurable* if it is measurable under any probability measure. Even if the conjecture is confirmed, we still need to know if there is an indeterminate game which is also universally measurable. For a complete understanding of the exact power randomization provides to infinite games, we need to resolve these problems.

Chapter 3

Conclusions to Part I

While the [BDBK+90] and [RS89] results are a first step in understanding the relationship of randomized strategies and deterministic strategies in online problems, our study attempts to get a more refined understanding of this relationship in terms of computability. The answer to our main question is not very satisfactory since it is done by an artificially constructed problem. It would be much more interesting if this could be done on natural problems.

Although online problems can be easily formulated as infinite games [BDBK+90], [RS89], there is no immediate transformation from the latter to the former. Even though we tried to construct a game to emulate the behavior of online problems, one may notice that the construction of the specific game for our main result needs the power of enumerating all computable strategies, which makes the game noncomputable. Thus, there is still a gap to be filled between our result and the result of [BDBK+90], [RS89]. A more legitimate candidate for infinite games as online problems is closed computable games. A closed game is computable, if there is a Turing machine which can test for membership of basic open sets of the winning set of the adversary. We thus have an immediate question.

- [1] Does randomization provide more power to computable closed games?

Another question is whether or not the result of [BDBK+90], [RS89] can be strengthened to apply to semicomputably determinate games.

- [2] Is there an α–competitive *computable* deterministic strategy if there is an α–competitive randomized strategy against an off-line adaptive adversary and the game is semicomputably determinate?

[BDBK+90], [RS89] also show that competitive ratio of a strategy versus offline adaptive adversaries is related to that versus online adaptive adversaries by a quadratic function. But the exact relative power of these two types of adversaries is still unknown. In particular, does there exist an online problem which separates online adaptive adversaries from offline adaptive adversaries?

There are two notions of separability that one can talk about. In the first notion, we ask if there is a randomized algorithm for a problem, which is α–competitive (for some $\alpha \geq 1$) against any on-line adversary, but not α–competitive against some off-line adversary. This question was answered in the affirmative by Raghavan and Snir [RS89]. However, there is another notion of separability. In this context, we ask if there is an on-line problem for which there exists an α–competitive randomized algorithm (for some $\alpha \geq 1$) against any on-line adversary, but there does not exist any α–competitive randomized algorithm against an off-line adversary for the problem. A negative answer to this question would give a positive answer to the k-server conjecture for resistive metric spaces [CDRS90].

[CDRS90] give an example of an online problem where the two adversaries can be separated, however this example is not very natural. Therefore, the question on natural online problems remains open though we believe that they should be separable, at least for some natural online problems.

- [3] Can offline adaptive adversaries be separated from online adaptive adversaries for a natural online problem?

We know that the k-server game whose winning set is defined by the set of all those paths which achieve a ratio of less than c for any $c < k$ is semicomputably determinate from the lower bound results of [MMS88]. We also know from Fiat et al's result [FRR90] that when $c > e^{O(klogk)}$, the k-server game is semicomputably determinate for every metric space. (Observe that the 2-server game is semicomputably determinate for any c and any metric space, as we have a computable algorithm [MMS88] whose competitive ratio is 2).

- [4] Can we show that the k-server game is semicomputably determinate when c is in neither of these ranges?

Chapter 4

Introduction to Part II

4.1 Motivation

In this part of the thesis, we will be concerned with designing polynomial size formulas for certain boolean functions. A *formula* is a rooted tree where the leaves are the input variables and the internal nodes represent $\wedge$, $\vee$, $\neg$ operations or some other operations taken from a predefined basis. An internal node such as an $\wedge$ is said to *represent* (recursively) the AND of the functions represented by its children. The function *evaluated* by a formula is the function represented by the root.

As there are 2^{2^n} boolean functions on n variables, and there are only $2^{O(n^k)}$ functions with representing formulas of size at $O(n^k)$, there are functions for which there do not exist polynomial size formulas. As we wish to gain insight into how we can construct polynomial size formulas for boolean functions, it seems advisable to look at simple functions first. We will therefore restrict ourselves to what are known as *monotone symmetric boolean functions.* These functions are symmetric in that they are insensitive to the interchange of any two input variables, and they are monotone in that they can be represented by formulas which only involve $\wedge$ and $\vee$ gates.

For any monotone boolean function f, if the function is true for some truth assignment of input variables, it cannot become false if any of the false input variable is assigned to be true. It is clear that symmetric functions only depend on the number of input variables which are set to true (or equivalently on the number of input variables which are set to false). For symmetric monotone functions, if the number of true input variables is bigger than a certain threshold, then the function is true, otherwise the function is false For this reason, symmetric monotone boolean functions are also known as *Threshold functions.* Among the Threshold functions, we will be particularly interested in the *Majority* function, which is defined as follows: If the number of true input variables exceeds $n/2$ (n being the number of input variables), then the function is true else it is false. The techniques that we describe to prove the existence of polynomial size formulas for the Majority function can be extended to other Threshold functions. The basis over which these formulas are constructed will consist of $\wedge$, $\vee$ and small majority gates (that is, gates which compute the majority function over a constant number of input variables).

To show good upper bounds on the size of the majority function, we will use a technique called *amplification*, first developed by Valiant [Val84]. The idea of this technique is to consider a simple random boolean formula that only approximately computes a given function and then combine several independent copies of this formula to prove the existence of a deterministic boolean formula that exactly computes the function. Valiant [Val84] used amplification to prove the existence of $O(n^{5.3})$ size formulas (where n is the number of variables) for majority over the basis $\{\wedge, \vee\}$ (observe that the naive DNF formula for majority has exponentially many terms). In order to prove his result, Valiant uses a random boolean function which is 1 with probability $p + \Theta(1/n)$ when the input is more than half ones, and otherwise it is 1 with probability $p - \Theta(1/n)$. Here $0 < p < 1$ is a constant. Ideally we would like to have the first probability to be 1 and the second one to be 0. Valiant *amplifies* the *separation* $(p - \Theta(1/n), p + \Theta(1/n))$ to $(c, 1-c)$ for some constant $0 < c < 1$ by composing the initial random boolean function with a deterministic boolean function of size $O(n^{3.27})$, and then amplifies $(c, 1-c)$ to $(2^{-n-1}, 1 - 2^{-n-1})$ using an amplifier of size $O(n^2)$. Then using probabilistic arguments, it is easy to show that there is a deterministic boolean formula of size $O(n^{5.3})$ which computes majority over n variables.

Boppana [Bop89] showed that Valiant achieved optimal amplification in both stages. In particular, he showed that if one starts with an initial separation of $(p, p + 1/n)$ (where $0 < p < 1$ is a constant), then any *read − once* formula which amplifies this to $(c, 1-c)$ has size $\Omega(n^{3.27})$, and any read-once formula that amplifies $(c, 1-c)$ to $(2^{-n-1}, 1-2^{-n-1})$ has size $\Omega(n^2)$. A read-once formula is one in which every variable occurs at most once. Boppana uses a novel information theoretic argument to prove his result. We will describe this argument in detail, in the next chapter.

Using ideas from Valiant and Boppana, we show the existence of optimal formulas (upto amplification) for majority when small majority gates are allowed in the basis.

4.2 Overview of the Results

Suppose we have a probabilistic(random) Boolean formula (a *Probabilistic Boolean Formula* is a random variable picked according to a specified probability distribution on deterministic formulas) which approximates some Boolean function. The basic idea behind amplification is to combine several independent copies of such probabilistic formulas to prove the existence of small formulas for the function.

In 1984, Valiant [Val84] used amplification to show the existence of $O(n^{5.3})$ size monotone formulas for majority. Boppana [Bop89] showed that amplification can not yield better bounds. In both of these papers, the basis functions were $\{\wedge, \vee\}$.

As the amplifier obtained in [Val84] is optimal, a natural question which arises from this work is whether we can get smaller size formulas if we use basis functions other than $\wedge$ and $\vee$.

The first nontrivial symmetric monotone function on more than 2 inputs is majority on 3 inputs (we denote majority on i inputs by M_i). Using some techniques from [Val84] and [Bop89], we show that there are $O(n^{4.29})$ size monotone formulas for the n^{th} majority function over the basis $\{\wedge, \vee, M_3\}$. In this construction only M_3 gates are used in the formula. We also give a matching lower bound on formula size over this basis. Combining these two results we obtain the surprising conclusion that, using amplification, the optimal formula for majority over $\{\wedge, \vee, M_3\}$ is a tree consisting of only M_3 gates.

We next extend our results to computing majority when the basis consists of $\{\wedge, \vee, M_3, M_5, ..., M_{2k+1}\}$ where k is any fixed number. Here the optimal formula size for the n^{th} majority function is $n^{3+\Theta(\frac{1}{\log k})}$. Again these lower bounds are with respect to amplification.

In amplification, an initial set of functions with some probability distribution is required. Since different distributions will yield different bounds on formula size, the choice of distribution is quite important. For example, if the set of formulas consists of all functions and the distribution assigns probability 1 to the optimal majority formula then we trivially obtain the optimal bound. In [Val84] the set consisting of the projection functions and constant function 0 is chosen. Here we investigate a large class of initial distributions and show that better bounds are not possible with these distributions. Thus amplification technique used in [Val84] is optimal modulo both the amplifier and any initial distribution from our class.

4.2.1 Previous Work

Valiant [Val84] was the first to use amplification to prove upper bounds of $O(n^{5.3})$ on the size of monotone formulas for majority over the basis $\{\wedge, \vee\}$. Previous to this, the best known upper bound was $O(n^{9.3} \log n)$ given by Friedman [Fri86]. Over the basis $\{\wedge, \vee, \neg\}$, Paterson, Pippenger and Zwick [PPZ90] showed that majority can be computed by formulas of size $O(n^{4.85})$. For the lower bound, Khrapchenko [Khr72] showed that over $\{\wedge, \vee, \neg\}$ the formula size for majority is $\Omega(n^2)$. No better bound is known for $\{\wedge, \vee\}$.

A considerable amount of research has been dedicated to finding upper and lower bounds for threshold functions. Boppana [Bop89] showed that over $\{\wedge, \vee\}$, there is a formula of size $O(k^{4.3} n \log n)$ which computes the k^{th} threshold function. He also shows that Valiant's amplifier is optimal for both stages of the amplification process. Observe however, that this does not show that Valiant's amplifier is optimal if one were to directly go from a separation of $(p, p + \Theta(\frac{1}{n}))$ to $(2^{-n-1}, 1 - 2^{-n-1})$. Dubiner and Zwick [DZ92] show that Valiant's amplifier is indeed optimal even in this sense. They also show that Valiant's amplifier is optimal even if we include XOR gates in the basis. Recently, Radhakrishnan [Rad91] has shown a lower bound of $\Omega(kn \log \frac{n}{k-1})$ for monotone contact networks. This result improved over the previous lower bounds of $\Omega(kn)$ due to Khrapchenko [Khr72] and $\Omega(n \log n)$ due to Krichevskii [Kri64] and

Hansel [Han64].

Chapter 5

Preliminaries

5.1 Basic Definitions

A *monotone Boolean function* is one which is non-decreasing in all it inputs, that is, if some input is changed from a 0 to 1 (with other inputs unchanged) then the value of the function cannot change from a 1 to a 0. If $\mathcal{B}$ is a set of monotone functions then a monotone *formula* over $\mathcal{B}$ is a Boolean formula which only uses gates labelled by functions from $\mathcal{B}$. $\mathcal{B}$ is also called the *basis* over which the formula is constructed. In general, we will not distinguish between a gate and the corresponding function of that gate. The *size* of a Boolean formula (over $\mathcal{B}$) is the number of occurrences of variables. A formula is *read-once* if every variable of the formula occurs exactly once.

For n odd, the n^{th} majority function, M_n, is 1 iff more than half its inputs are 1. Notice that M_n is monotone for every n.

Following the notation in [Bop89], for $f : \{0,1\}^n \rightarrow \{0,1\}$, its *amplification function*, $A_f : [0,1] \rightarrow [0,1]$ is given by

$$A_f(p) = Pr(f(x_1, \ldots, x_n) = 1)$$

where $x_1, \ldots, x_n$ are independently chosen to be 1 with probability p. For $0 \leq p, q, p', q' \leq 1$, the function f *amplifies* (p, q) to (p', q') if $A_f(p) \leq p'$ and $A_f(q) \geq q'$.

For $\mathcal{D}$ a distribution, the *support* of $\mathcal{D}$, is the set of all elements of the underlying universe which have non-zero probability.

Let $n = 2m + 1$. Let $\mathcal{Y}_0, \mathcal{Y}_1 \subseteq \{0,1\}^n$ be the set of all vectors having at most m 1's and at least $(m + 1)$ 1's respectively. For $Y_i \in \mathcal{Y}_i$ and $0 \leq p < q \leq 1$, a distribution $\mathcal{D}$ on deterministic formulas has *separation* (p, q) if $Pr(F(Y_1) = 1) > q$ and $Pr(F(Y_0) = 1) < p$ when F is chosen from $\mathcal{D}$.

Let $x_1, \ldots, x_n$ be boolean indeterminates (that is, variables). Then $sym(k, n)$ is the set of all M_{2k+1}'s on all possible $\binom{n}{2k+1}$ indeterminates from $\{x_1, \ldots, x_n\}$. For example, $sym(1, 4)$ is $\{M_3(x_1, x_2, x_3), M_3(x_1, x_2, x_4), M_3(x_1, x_3, x_4), M_3(x_2, x_3, x_4)\}$. We denote by $\mathcal{D}(k, n)$ the uniform distribution with support $sym(k, n)$.

Throughout this thesis, all logarithms will be base 2 unless otherwise specified. The entropy function appears throughout the thesis.

Definition: The entropy function $H : [0,1] \to [0,1]$ is defined as:

$$H(p) = -p \log p - (1 - p) \log(1 - p).$$

5.2 Intuition behind Amplification

Consider a probabilistic formula F which only *approximately* computes the majority function, that is the probability that F is 1 over vectors which have more than half the number of ones is at least q and the probability that F is 0 over vectors which have at most half the number of ones is at most p (where $p < q$). We call the interval (p, q) the *separation* of F. Clearly, if q is 1 and p is 0, F exactly computes majority (that is, every function in its support computes majority). However it is sufficient that $q \geq 1 - 2^{-n-1}$ and $p \leq 2^{-n-1}$ since if we want at least one function in the support of F to exactly compute M_n) because then

$$\begin{aligned} Prob(F \not\equiv M_n) &= Prob(\exists x (F(x) \neq M_n(x)) \\ &\leq \sum_{x \in \{0,1\}^n} Prob(F(x) \neq M_n(x)) \\ &\leq 2^n * 2^{-n-1} = \frac{1}{2} \end{aligned}$$

So the $Prob(F \not\equiv M_n) \geq \frac{1}{2}$, and therefore there exists a function in the support of F that exactly computes the majority. So how do we get an F with separation

$(2^{-n-1}, 1-2^{-n-1})$? Valiant starts out with a probabilistic formula G (with corresponding distribution $\mathcal{G}$) with separation $(c, c+\Omega(\frac{1}{n}))$ where $0 < c < 1$ is a constant. He then amplifies the desired separation with the help of a deterministic boolean function f (called the *amplifier*), that is, he *composes* f with G to get another probabilistic formula F with separation $(2^{-n-1}, 1-2^{-n-1})$. The composition operation here involves substituting independently, a copy from $\mathcal{G}$ for each variable in f. Valiant's amplifier f is a tree of alternating $\wedge$ and $\vee$ gates. Alternately, one can think of f as iteratively amplifying the separation, that is at stage $i+1$, we compose the function $g = (x_1 \vee x_2) \wedge (x_3 \vee x_4)$ with the probabilistic function at stage i. This means that we draw 4 independent copies of the probabilistic formula at stage i and replace x_1, x_2, x_3 and x_4 respectively in g by them. Valiant then shows that $O(\log n)$ stages are enough to get the required separation. Note that the amplifying formula f so constructed is *read − once*, that is every variable in it appears at most once.

To see that $O(\log n)$ stages are enough, let p_i be the probability that the random formula at stage i is 0 on a vector which contains more than half the number of 1's, and q_i be the probability that the random formula at stage i is 1 on a vector which contains at most half the number of 1's. That is, p_i's and q_i's are error probabilities. Then p_{i+1} and q_{i+1} are some polynomial functions of p_i and q_i respectively, say $g(p_i)$ and $h(p_i)$ respectively. Let us concentrate on q_i's as the argument for p_i's is similar. We want to get to a stage i where q_i is at most 2^{-n-1}. h has a fixed point α where $0 < \alpha < 1$. If q_i ever assumes the value α, the amplification process will stop. We can avoid this problem by starting out with an initial probabilistic boolean function with separation $(\alpha - \Omega(\frac{1}{n}), \alpha + \Omega(\frac{1}{n}))$. Now if $q_i = \alpha - \epsilon$, then by Taylor series expansion around α, we have that $q_{i+1} = h(\alpha) - h'(\alpha)\epsilon + O(\epsilon^2) = \alpha - h'(\alpha)\epsilon + O(\epsilon^2)$ since α is a fixed point. Now it turns out that $h'(\alpha) > 1$, so for any $1 < \gamma < h'(\alpha)$, we can choose a constant k small enough such that if the q_i's are less than $\alpha - k$, then

$$q_i = \alpha - \epsilon \Rightarrow q_{i+1} < \alpha - \gamma\epsilon$$

Then as $q_0 = \alpha - \Omega(\frac{1}{n})$, we can, in $\log_\gamma(n)$ iterations, get $q_i < \alpha - k$, that is we are able to achieve a constant separation. Now after a further constant number of iterations, we can get any desired constant separation.

To get to a stage where $q_i < 2^{-n-1}$, it turns out that $q_{i+1} < dq_i^2$, where d is a large enough constant. So if we have achieved a large enough constant separation, then in a further $\log_2(n)$ iterations we can get a separation of $(2^{-n-1}, 1-2^{-n-1})$. The size of the formula so achieved is $4^{\log_\gamma(n)+O(1)+\log_2(n)} = O(n^{5.27})$

Chapter 6

Computing Majority with Small Majority Gates

In this chapter, we show there is a formula of size $O(n^{4.29})$ for M_n over $\{\wedge, \vee, M_3\}$. In the next section we will show that this bound is nearly optimal.

Let $n = 2m + 1$ and let $\mathcal{D}$ be the uniform distribution with support $\{X_1, \ldots, X_n\}$ where X_i is the i^{th} projection function (the ith *projection function* X_i is defined as follows: $X_i(x_1, ..., x_n) = x_i$). Then, for $Y \in \mathcal{Y}_1$, and F chosen from $\mathcal{D}$, $Pr(F(Y) = 1) \geq \frac{1}{2} + \frac{1}{2(2m+1)}$ and for $Y \in \mathcal{Y}_0$, $Pr(F(Y) = 1) \leq \frac{1}{2} - \frac{1}{2(2m+1)}$.

We construct a sequence of probability distributions $\mathcal{D}_0, \mathcal{D}_1, \ldots$ as follows:

1. $\mathcal{D}_0 = \mathcal{D}$;

2. Suppose we have constructed $\mathcal{D}_i$. Then, $\mathcal{D}_{i+1}$ is defined as follows: Let F_1, F_2, F_3 be independently chosen from $\mathcal{D}_i$. Then, $F = M_3(F_1, F_2, F_3)$ is in $\mathcal{D}_{i+1}$.

We note that this construction is equivalent to composing a read-once amplifying function with the initial distribution $\mathcal{D}_0$.

Let $F \in \mathcal{D}_i$. Then for $Y \in \mathcal{Y}_1$, let $p_i = Pr(F(Y) = 0)$ and for $Y \in \mathcal{Y}_0$, let $q_i = Pr(F(Y) = 1)$. Here p_i and q_i are the probabilities that a function in $\mathcal{D}_i$ is different from M_n for $Y \in \mathcal{Y}_1$ and $Y \in \mathcal{Y}_0$ respectively. Let $g(x) = 3x^2 - 2x^3$. Then, for $F_1, F_2, F_3 \in \mathcal{D}_i$ and $Y \in \mathcal{Y}_1$, we obtain that

$$p_{i+1} = Pr(M_3(F_1(Y), F_2(Y), F_3(Y)) = 0) = g(p_i)$$

and for $Y \in \mathcal{Y}_0$,

$$q_{i+1} = Pr(M_3(F_1(Y), F_2(Y), F_3(Y)) = 1) = g(q_i).$$

The following lemma is not difficult.

Lemma 6.0.1. *The only fixed point of $g(x)$ in $(0,1)$ is $\frac{1}{2}$.*

Thus, $\frac{1}{2}$ is the fixed point for both recurrence relations (i.e. for both p_i and q_i). Note that $\mathcal{D}_0$ was chosen so that this fixed point would lie in $(q_0, 1-p_0)$.

Now, suppose that for some i, $p_i = \frac{1}{2} - \epsilon$ for some $\epsilon > 0$. Then, $p_{i+1} = \frac{1}{2} - \epsilon g'(1/2) + O(\epsilon^2)$. Therefore

$$p_i = \frac{1}{2} - \epsilon \;\Rightarrow\; p_{i+1} = \frac{1}{2} - \epsilon\gamma$$

for each γ, $1 < \gamma < g'(\frac{1}{2})$ and a sufficiently small ϵ. Since $g'(\frac{1}{2}) = \frac{3}{2}$ we can choose such a γ.

We can now show that there are constants c, c' independent of n such that after $k = \log_{g'(\frac{1}{2})} n + c$ iterations, $p_k, q_k = \frac{1}{4}$ and after a further $k' = \log n + c'$ iterations, $p_{k+k'}, q_{k+k'} < 2^{-n-1}$. Using a standard probability argument, this implies the existence of a deterministic formula for M_n of the same size. Since the formula size increases by a factor of 3 at each iteration, the size of the final formula is $3^{k+k'} = O(n^{\log_{\frac{3}{2}} 3 + \log 3}) \approx O(n^{4.29})$.

In summary, our construction proceeds in two phases described by the following lemmas.

Lemma 6.0.2. *There is a formula of size $O(n^{\log_{\frac{3}{2}} 3})$ which amplifies $(q_0, 1-p_0) = (\frac{1}{2} - \Theta(\frac{1}{n}), \frac{1}{2} + \Theta(\frac{1}{n}))$, to $(q_k, 1-p_k) = (\frac{1}{4}, \frac{3}{4})$.*

Lemma 6.0.3. *There is a formula of size $O(n^{\log 3})$ which amplifies $(\frac{1}{4}, \frac{3}{4})$ to $(2^{-n-1}, 1-2^{-n-1})$.*

6.1 Nearly optimal lower bounds for majority over $\{\wedge, \vee, M_3\}$

In this section we prove the following two results.

Theorem 6.1.1.

1. *Over the basis* $\{\wedge, \vee, M_3\}$, *for* $0 < p < 1$ *and* $n \geq 1$, *every read-once formula that amplifies* $(p, p+\frac{1}{n})$ *to* $(\frac{1}{4}, \frac{3}{4})$ *must have size* $\Omega((H(p)n)^{\alpha})$ *where* $\alpha = \log_{\frac{3}{2}} 3 \approx 2.709$.

2. *For any constant* c, $0 < c < 1$, *every formula which amplifies* $(c, 1-c)$ *to* $(2^{-n-1}, 1-2^{-n-1})$ *must have size* $\Omega(n^{\log 3})$.

Lemma 6.1.2. *For all* $x, y \in [0,1]$ *and* $\gamma \geq 1$,

$$G(x,y,\gamma) = (H(xy)) - ((yH(x))^{\gamma} + (xH(y))^{\gamma})^{\frac{1}{\gamma}}$$

is a non-decreasing function with respect to γ.

Proof: Without loss of generality assume that $yH(x) \leq xH(y)$. Then, $G(x,y,\gamma) = H(xy) - xH(y)(1 + (\frac{yH(x)}{xH(y)})^{\gamma})^{\frac{1}{\gamma}}$. Since $\frac{yH(x)}{xH(y)} \leq 1$ and $\gamma \geq 1$, $(1 + (\frac{yH(x)}{xH(y)})^{\gamma})^{\frac{1}{\gamma}}$ is a non-increasing function of γ. Thus, $G(x,y,\gamma)$ is a non-decreasing function of γ. **QED**

Lemma 6.1.3. *Let* $\beta = \log 3$. *Then, for all* $x, y, z \in [0,1]$,

$$F(x,y,z) = (H(xy + yz + xz - 2xyz))^{\beta} - ((y + z - 2yz)H(x))^{\beta} -$$

$$((x + z - 2xz)H(y))^{\beta} - ((x + y - 2xy)H(z))^{\beta} \geq 0$$

Proof: We show that $F(\frac{1}{2}, \frac{1}{2}, \frac{1}{2}) = 0$ is a global minimum. From this the result clearly follows. By checking the first partial derivatives and the Hessian of F [BL67], it is straightforward to check that F reaches a local minimum at $(\frac{1}{2}, \frac{1}{2}, \frac{1}{2})$. If one of x, y or z is 0, say $z = 0$, then $F(x,y,z) \geq 0$ iff $G(x,y,\beta) \geq 0$. But, $G(x, y, \frac{1}{\log \frac{\sqrt{5}-1}{2}}) \geq 0$ [Bop89] and $\beta \geq \frac{1}{\log \frac{\sqrt{5}-1}{2}}$ so by Lemma 6.1.2, the result holds in this case. A similar observation can be made when one of x, y or z is 1.

Finally, for any point (x_0, y_0, z_0) bounded away from $(0,0,0), (1,1,1)$ and $(\frac{1}{2}, \frac{1}{2}, \frac{1}{2})$, we show the function is positive by a non-negligible amount, $\delta(x_0, y_0, z_0)$, by considering an ϵ-box (ϵ a function of (x_0, y_0, z_0)) around the point. We consider the Taylor series of F about $(\frac{1}{2}, \frac{1}{2}, \frac{1}{2})$ and take sufficient terms to make the remainder term less than $\delta(x_0, y_0, z_0)$. Since ϵ can be chosen arbitrarily small, the result follows. **QED**

The remainder of this section is a proof of the main theorem. For any read-once formula f, we first obtain an upper bound (in terms of $size(f)$) for $A'_f(p)$ over the basis $\{\wedge, \vee, M_3\}$.

Lemma 6.1.4. *For f a read-once formula over the basis $\{\wedge, \vee, M_3\}$ and $\alpha = \frac{\log 3}{\log 3 - \log 2}$,*

$$A'_f(p) \leq (size(f))^{\frac{1}{\alpha}} \frac{H(A_f(p))}{H(p)}.$$

Proof: The proof is by structural induction on f. The case where $size(f) = 1$ is trivial. For $f = f_1 \wedge f_2$ or $f = f_1 \vee f_2$, the proof is similar to that in [Bop89]. Now, suppose $f = M_3(f_1, f_2, f_3)$. Then,

$$A_f(p) = A_{f_1}(p)A_{f_2}(p) + A_{f_2}(p)A_{f_3}(p) + A_{f_1}(p)A_{f_3}(p) - 2A_{f_1}(p)A_{f_2}(p)A_{f_3}(p).$$

Therefore,

$$\begin{aligned} A'_f(p) = {} & A'_{f_1}(p)(A_{f_2}(p) + A_{f_3}(p) - 2A_{f_2}(p)A_{f_3}(p)) + \\ & A'_{f_2}(p)(A_{f_1}(p) + A_{f_3}(p) - 2A_{f_1}(p)A_{f_3}(p)) + \\ & A'_{f_3}(p)(A_{f_1}(p) + A_{f_2}(p) - 2A_{f_1}(p)A_{f_2}(p)). \end{aligned}$$

By the induction hypothesis,

$$\begin{aligned} A'_f(p) \leq {} & \frac{1}{H(p)}(size(f_1)^{\frac{1}{\alpha}} H(A_{f_1}(p))(A_{f_2}(p) + A_{f_3}(p) - 2A_{f_2}(p)A_{f_3}(p)) + \\ & size(f_2)^{\frac{1}{\alpha}} H(A_{f_2}(p))(A_{f_1}(p) + A_{f_3}(p) - 2A_{f_1}(p)A_{f_3}(p)) + \\ & size(f_3)^{\frac{1}{\alpha}} H(A_{f_3}(p))(A_{f_1}(p) + A_{f_2}(p) - 2A_{f_1}(p)A_{f_2}(p))). \end{aligned}$$

Using Hölder's inequality [BB65], we have,

$$\begin{aligned} A'_f(p) \leq {} & \frac{1}{H(p)}(size(f))^{\frac{1}{\alpha}}(((H(A_{f_1}(p))(A_{f_2}(p) + A_{f_3}(p) - 2A_{f_2}(p)A_{f_3}(p)))^{\beta} + \\ & (H(A_{f_2}(p))(A_{f_1}(p) + A_{f_3}(p) - 2A_{f_1}(p)A_{f_3}(p)))^{\beta} + \\ & (H(A_{f_3}(p))(A_{f_1}(p) + A_{f_2}(p) - 2A_{f_1}(p)A_{f_2}(p)))^{\beta}))^{\frac{1}{\beta}} \end{aligned}$$

where $\beta = \log 3$. Finally, by Lemma 6.1.3, the result follows. **QED**

To prove the theorem, we note the following observation first made in [Bop89]. Let f be a function amplifying $(p, p+1/n)$ to $(\frac{1}{4}, \frac{3}{4})$. Then by the mean-value theorem of calculus, there is an η, $p \leq \eta \leq p + \frac{1}{n}$ such that $A'_f(\eta) = n(A_f(p+\frac{1}{n}) - A_f(p))$. Since $A_f(p+\frac{1}{n}) - A_f(p) \geq \frac{1}{2}$, then by Lemma 6.1.4 $(size(f))^{\frac{1}{\alpha}} \frac{(H(A_f(p)))}{H(p)} \geq A'_f(\eta) \geq \frac{n}{2}$ and thus part one of Theorem 6.1.1 holds.

For part 2 of the theorem, we begin with the following definitions which first appear in [MS56]. Our choice of notation appears in [Bop89].

Definition: Let f be a boolean function. Then, $L_0(f)$ is the minimum number of variables of f which must be set to 0 to force f to 0. Similarly, $L_1(f)$ is the minimum number of variables which must be set to 1 to force f to 1.

It is clear that for any read-once monotone boolean formulas f, g and h

1. $L_0(f \wedge g) = \min\{L_0(f), L_0(g)\}$ and $L_1(f \wedge g) = L_1(f) + L_1(g)$;
2. $L_0(f \vee g) = L_0(f) + L_0(g)$ and $L_1(f \vee g) = \min\{L_1(f), L_1(g)\}$;
3. $L_0(M_3(f,g,h)) = \min\{L_0(f) + L_0(g), L_0(f) + L_0(h), L_0(g) + L_0(h)\}$ and $L_1(M_3(f,g,h)) = \min\{L_1(f) + L_1(g), L_1(f) + L_1(h), L_1(g) + L_1(h)\}$.

Theorem 6.1.1 follows from the following lemma:

Lemma 6.1.5. *Let f be a read-once monotone formula over the basis $\{\wedge, \vee, M_3\}$. Then, $L_0(f)L_1(f) \leq size(f)^{\frac{2}{\log 3}}$.*

To see the theorem, we begin by noting that in [Bop89], it is shown that for any constant $0 < c < 1/2$, a function f which amplifies $(c, 1-c)$ to $(2^{-n-1}, 1-2^{-n-1})$ has $L_0(f), L_1(f) \geq kn$ where k is a constant depending on c. Therefore, by the lemma,

$$(kn)^2 \leq L_0(f)L1(f) \leq size(f)^{\frac{2}{\log 3}}.$$

Thus, $size(f) \geq (kn)^{\log 3}$.

Proof of Lemma 6.1.5: We must first prove the following result: For every odd $n > 1$ the formula g of size n which maximizes the product $L_0(g)L_1(g)$ is the tree of minimum height consisting entirely of M_3 gates. This proof follows from the fact that for a complete ternary tree of height k, the formula g obtained by labelling all internal nodes by M_3 has $L_0(g) = L_1(g) = 2^k$. Now induction on n gives this result.

The proof of the lemma is by structural induction on f. When f is $f_1 \wedge f_2$ or $f_1 \vee f_2$, it is straight-forward to verify the result. When f is $M_3(f_1, f_2, f_3)$ then by the above result, all f_i are full ternary trees. Furthermore, all the f_i must have the same size otherwise we can increase the product $L_0(f)L_1(f)$ by making the size of one of the f_i's smaller and another bigger. Furthermore, for $1 \le i \le 3$ we can choose f_i such that $L_0(f_i)$ and $L_1(f_i)$ is maximized. Therefore, $L_0(f_1) = L_0(f_2) = L_0(f_3)$ and $L_1(f_1) = L_1(f_2) = L_3(f_3)$ and

$$L_0(f)L_1(f) = (2L_0(f_1))(2L_1(f_1)) \le 4size(f_1)^{\frac{2}{\log 3}}.$$

Since $size(f_1) = \frac{size(f)}{3}$ and $3^{\frac{2}{\log 3}} = 4$, the result follows. **QED**

6.2 Computing majority with small majority gates

In this section we generalize the results of sections 6.0 and 6.1 to derive the upper and lower bounds on the size of a formula over the basis $\mathcal{B}_k = \{\wedge, \vee, M_3, M_5, \ldots, M_{2k+1}\}$ for M_n using amplification. We begin with the upper bounds.

Theorem 6.2.1. *Let $k > 0$ and $\alpha = \frac{\log(2k+1)}{\log(\binom{2k+1}{k}(k+1)(\frac{1}{2})^{2k})}$. Then:*

1. *Let $0 < p < 1$ be fixed and $n \ge 1$. Then, there is a monotone read-once formula over $\mathcal{B}_k$ of size $O(n^\alpha)$ which amplifies $(p, p + \frac{1}{n})$ to $(\frac{1}{4}, \frac{3}{4})$.*

2. *There is a monotone read-once formula over $\mathcal{B}_k$ of size $O(n^{\log_{k+1}(2k+1)})$ which amplifies $(\frac{1}{4}, \frac{3}{4})$ to $(2^{-n-1}, 1 - 2^{-n-1})$.*

By composing the two functions from the theorem and observations made in the previous section, we obtain the following corollary.

Corollary 6.2.2. *Let $k > 0$. Then, there is a monotone read-once formula of size $O(n^{\alpha + \log_{k+1} 2k+1})$ where α is as in Theorem 6.2.1, which computes M_n over the basis $\mathcal{B}_k$.*

Before proving the theorem we require a number of combinatorial identities.

Lemma 6.2.3. *Let $r, s \in \mathbb{N}$, $r \geq s+1$. Then,*

$$\sum_{i=1}^{r-s}(-1)^{i-1}\binom{r}{s+i} = \binom{r-1}{s}.$$

Proof: We prove the lemma by downward induction on s. For $s = r-1$, the assertion is trivial. So let us assume that the assertion holds for all $r-1 \geq s \geq j$. Then

$$\sum_{i=1}^{r-(j-1)}(-1)^{i-1}\binom{r}{j-1+i}$$

$$= \binom{r}{j} - \sum_{i=1}^{r-j}(-1)^{i-1}\binom{r}{j+i}$$

by the induction hypothesis, which further equals

$$\binom{r}{j} - \binom{r-1}{j} = \binom{r-1}{j-1}.$$

QED

Lemma 6.2.4. *Let $k \geq 1$. Then,*

$$\sum_{i=k}^{2k}(-1)^{i-k}\binom{i}{k}\binom{2k+1}{i+1}(1/2)^i = 1.$$

Proof: By straight-forward manipulation, we can reduce this problem to showing that

$$\binom{2k+1}{k}(k+1)(1/2)^k\sum_{i=0}^{k}(-1/2)^i\frac{1}{i+k+1}\binom{k}{i} = 1.$$

This further reduces to

$$2\binom{2k+1}{k}(k+1)(-1)^{k+1}\sum_{i=0}^{k}(-1/2)^{i+k+1}\frac{1}{i+k+1}\binom{k}{i} = 1.$$

But,

$$\sum_{i=0}^{k}(-1/2)^{i+k+1}\frac{1}{i+k+1}\binom{k}{i} = \int_0^{-\frac{1}{2}} x^k(1+x)^k dx$$

Now, using the substitution, $x = \frac{\sin\theta - 1}{2}$, the result follows. **QED**

Lemma 6.2.5. *Let $k \geq 1$. Then,*

$$\sum_{i=k}^{2k}(-1)^{i-k}(i+1)\binom{i}{k}\binom{2k+1}{i+1}(1/2)^i = \binom{2k+1}{k}(k+1)(1/2)^{2k}.$$

Proof: We can reduce the left-hand side of the identity to

$$\binom{2k+1}{k}(k+1)(1/2)^k\sum_{i=0}^{k}(-1/2)^i\binom{k}{i}$$

and since $\sum_{i=0}^{k}(-\frac{1}{2})^i\binom{k}{i} = (\frac{1}{2})^k$, the result follows. **QED**

We are now ready to prove the main theorems. We refer the reader to the first section of this chapter for a more detailed analysis when $k = 1$. As in the case $k = 1$, instead of using M_3 in our iterative process, we now use M_{2k+1}. The definition of p_i and q_i is analogous to that in Section 3 except we now use $2k+1$ instead of 3 in our distribution. We obtain the recurrence $p_{i+1} = g(p_i)$ and $q_{i+1} = g(q_i)$ where

$$g(x) = \sum_{i=k}^{2k}(-1)^{i-k}\binom{i}{k}\binom{2k+1}{i+1}x^{i+1}.$$

Here the coefficients of $g(x)$ are obtained using Lemma 6.2.3. By Lemma 6.2.4, $\frac{1}{2}$ is a fixed point of g. In fact, $\frac{1}{2}$ is the only fixed point in $(0,1)$. This is a key point since if g had more than one fixed point in $(0,1)$, we could never avoid getting stuck at one of those points. Furthermore,

$$g'(\frac{1}{2}) = \sum_{i=k}^{2k}(-1)^{i-k}(i+1)\binom{i}{k}\binom{2k+1}{i+1}(1/2)^i > 1$$

and by Lemma 6.2.5, $g'(\frac{1}{2}) = \binom{2k+1}{k}(k+1)(\frac{1}{2})^{2k}$. Now, the size of the formula is $n^{\frac{\log(2k+1)}{\log(g'(\frac{1}{2}))}}$.

For the second part, we see that for a large enough constant c, $p_{i+1} < cp_i^{k+1}$ and similarly $q_{i+1} < cq_i^{k+1}$. Let $\epsilon < \frac{1}{4}c^{-1/k}$. In $t = O(1)$ iterations, we can amplify $(\frac{1}{4}, \frac{3}{4})$ to $(\epsilon, 1-\epsilon)$. Solving the two inequalities $p_{i+1} < cp_i^{k+1}$, $q_{i+1} < cq_i^{k+1}$ we obtain that $p_i, q_i = 2^{(\log\epsilon + \frac{\log c}{k})(k+1)^i - \frac{\log c}{k}}$. Therefore after a further $\log_{k+1} n$ iterations, we will achieve a separation of $(2^{-n-1}, 1-2^{-n-1})$. The size of the formula is $n^{\frac{\log(2k+1)}{\log(k+1)}}$ which yeilds the required result. **QED**

We now show the lower bounds. For the first phase we obtain the following:

Theorem 6.2.6. *Let* $\alpha = \frac{\log(2k+1)}{\log(\binom{2k+1}{k}(k+1)(\frac{1}{2})^{2k})}$. *Let* $0 < c < 1/2$ *and* $0 < p < 1$. *Then, for any monotone read-once formula* f *which amplifies* $(p, p+\frac{1}{n})$ *to* $(c, 1-c)$, $size(f) \in \Omega((H(p)n)^{\alpha})$.

We prove this theorem in the next section. For the second phase, we obtain the following:

Theorem 6.2.7. *Let* $0 < c < 1/2$. *Let* f *be a monotone read-once formula over* $\mathcal{B}_k$ *which amplifies* $(c, 1-c)$ *to* $(2^{-n-1}, 1-2^{-n-1})$. *Then,* $size(f) \in \Omega(n^{\log_{k+1}(2k+1)})$.

The theorem follows in an analogous method to that in Section 6.1 from the following lemma.

Lemma 6.2.8. *For any monotone read-once formula* f *over the basis* $\mathcal{B}_k$, $L_0(f)L_1(f) \leq size(f)^{\frac{2}{\log_{k+1}(2k+1)}}$.

Since the proof of this lemma is similar to that of Lemma 6.1.5, we omit it.

6.3 Lower Bounds For the First Phase

In this section, we prove theorem 6.2.6.

Theorem 6.3.1. *Let* $k > 0$ *and* $\alpha = \frac{\log(2k+1)}{\log(\binom{2k+1}{k}(k+1)\frac{1}{2}^{2k}}$. *Then:*

- *Let* $0 < p < 1$ *be fixed and* $n \geq 1$. *Then every read-once formula over* $\mathcal{B}_k$ *which amplifies* $(p, p+1/n)$ *to* $(1/4, 3/4)$ *is of size* $\Omega((H(p)n)^{\alpha})$.
- *Every read-once formula over* $\mathcal{B}_k$ *which amplifies* $(1/4, 3/4)$ *to* $(2^{-n-1}, 1-2^{-n-1})$ *is of size* $\Omega(n^{\log_{k+1}(2k+1)})$.

To prove these results, we need to establish the following properties of the entropy function.

Lemma 6.3.2. *Let $\beta = \frac{\alpha}{\alpha-1}$ where α is as in Theorem 6.3.1. Let $\binom{S}{i}$ represent the collection of all subsets X the $S = \{1, ..., 2k+1\}$ with i elements. We represent the characteristic function of a subset X of S by χ_X (that is $\chi_X(i) = 1$ if $i \in X$ else it is 0). For any $0 \leq x_1, ..., x_{k+1} \leq 1$, $F(x_1, ..., x_{2k+1}) \geq 0$ where*

$$F(x_1, ..., x_{2k+1}) = -(\sum_{j=1}^{n}(\frac{H(x_j)}{x_j}\sum_{i=k+1}^{2k+1}(-1)^{i-k+1}\binom{i-1}{k}\Pi_{X\in\binom{S}{i},j\in X}(x_i^{\chi_X(i)})^{\beta}) + (H(\sum_{i=k+1}^{2k+1}(-1)^{i-k+1}\binom{i-1}{k}\Pi x_i^{\chi_X(i)}))^{\beta}. \quad (3.1)$$

Proof: One can check that the function F reaches a local minimum at $x_1 = ... = x_{2k+1} = \frac{1}{2}$ and that $F = 0$ at $x_1 = ... = x_{2k+1} = \frac{1}{2}$. So if we can establish that in fact F has a global minimum at $x_1 = ... = x_n = \frac{1}{2}$, we will be done. That F has a global minimum at this point can be obtained by methods similar to the lower bound proof as obtained for the case $k = 1$, that is for M_3. **QED**

Theorem 6.3.3. *Let f be a read-once formula over the basis $\{\wedge, \vee, M_3, ..., M_{2k+1}\}$ which amplifies $(c, c+\frac{1}{n})$ to $(1/4, 3/4)$. Then $size(f) \in \Omega((H(c)n)^{\alpha}))$ where α is as in Theorem 6.3.1.*

Proof: We first show that if f is any read-once formula over the basis $\{\wedge, \vee, M_3, ..., M_{2k+1}\}$, then $A'_f(p) \leq (size(f))^{\frac{1}{\alpha}}\frac{H(A_f(p)}{H(p)}$. The proof is by structural induction on formulas. If f is a single variable, then the proof follows trivially. If f is of the form $f_1 \vee f_2$, $f_1 \wedge f_2$, the proof follows the same lines as in proof of Boppana's theorem [Bop89]. If f is $M_{2k+1}(f_1, ..., f_{2k+1})$, then

$$A_f(p) = \sum_{i=k+1}^{2k+1}(-1)^{i-k+1}\binom{i-1}{k} \times A_{f_i}(p)^{\chi_X(i)})$$

Then taking the first derivatives of the both sides of this equation, using the Holder's inequality, and the inequality for entropy in lemma 6.3.2, gives the result. **QED**

The proof of the main theorem of this section then follows as if f amplifies $(p, p+\frac{1}{n})$ to $(1/4, 3/4)$, then $A_f(p) \leq 1/4$ and $A_f(p+\frac{1}{n}) \geq 3/4$ and hence by Mean value theorem, there is a q such that $p \leq q \leq p + \frac{1}{n}$, such that $A'_f(q) = n/2$, which when combined with the above theorem gives the result.

6.4 Bounds on alternate distributions

Valiant in his original paper [Val84] considered a simple distribution with support $\{0, X_1, \ldots, X_n\}$ where X_i is the i^{th} projection function. Clearly, using amplification, not every distribution (over Boolean functions) can be generated from this one. For example, the distribution with support $\{X_1 + X_2\}$ cannot be obtained from any distribution with support $\{0, X_1, X_2\}$. Boppana shows that any initial distribution which obtains a separation (p, q) where $q - p \in O(\frac{1}{n})$ cannot be used to obtain better bounds. This immediately gives the result for Valiant's initial distribution.

In this section we study the problem of using amplification on alternate distributions. Observe that proving results which hold for all distributions is tantamount to proving categorical lower bounds for majority: Simply choose an initial distribution with support the optimal formula for majority. Here we prove that to compute formulas for M_n, if our initial distribution contains majority functions M_k then the distribution does no better than Valiant's initial distribution, where the formula size of M_k is that yielded by amplification over any of the bases discussed in this paper.

Throughout this section we will use the basis $\{\wedge, \vee\}$. Using the techniques from the previous sections, we can obtain similar results over bases which also contain small majority gates.

Recall that $\mathcal{D}(k, n)$ is the uniform distribution with support $sym(k, n)$. It is easy to see that for $k \neq k'$, $\mathcal{D}(k, n)$ cannot be generated from $\mathcal{D}(k', n)$ using amplification. Since majority is a symmetric function, we do not need to consider any distribution on $sym(k, n)$ other than $\mathcal{D}(k, n)$.

Theorem 6.4.1. *Let $k < n$. Then, using the initial distribution $\mathcal{D}(k, n)$, the size of the formula which amplifies the initial separation to a constant separation is $\Omega((n/k)^\alpha)$, where $\alpha = \frac{\log 2}{log(\sqrt{5}-1)}$.*

Proof: To prove the theorem, we first compute the initial separation.

Let $n = 2m + 1$. Consider an $x, y \in \{0, 1\}^n$ where x contains $(m + 1)$ 1's and y contains m 1's. Then, for F a random variable in $\mathcal{D}(k, n)$, $p = Pr(F(x) = 1) =$

$\sum_{j=k+1}^{2k+1} \frac{\binom{m+1}{j}\binom{m}{k}}{\binom{2m+1}{2k+1}}$ and $q = Pr(F(y) = 1) = \sum_{j=k+1}^{2k+1} \frac{\binom{m}{j}\binom{m+1}{k}}{\binom{2m+1}{2k+1}}$. Then, (q, p) is the separation of $\mathcal{D}_k$ and we asymptotically bound $\delta = p - q$.

$$\delta = p - q = \sum_{j=k+1}^{2k+1} \frac{(m+1)!m!(j-k)}{\binom{2m+1}{2k+1} j!k!(m-j)!(m-k)!(m-j+1)(m-k+1)}$$

Hence $\delta \in O(\frac{k}{m})$ and $p, q \in O(1)$. Now using results in [Bop89] gives the result. **QED**

Chapter 7

Conclusions to Part II

We have shown that amplification can be used to show the existence of small formulas for majority when small majority gates are allowed. As well, we have shown optimal lower bounds on the size of these formulas.

We note that our techniques combined with those of Boppana [Bop89] can be generalized to prove the existence of small formulas for threshold functions using small majority gates. However, what happens if we also allow small threshold gates in the formula? We conjecture that the bounds will not be improved.

The entropy function is used both in our proof and in [Bop89]. Recently, Boppana [Bop91] has shown that in his proof the function

$$(x(1-x))^{\frac{\phi+1}{\phi+2}}$$

where $\phi = \frac{\sqrt{5}+1}{2}$ can also be used. We can show that no polynomial and in general no function of the form $f(x)^c$ where $f(x)$ is analytic and $c \geq 1$ can be used both in our case and in [Bop89]. In our case Boppana's new function also will not work. An interesting question is whether there are other functions which will also work in our case. Further work to determine a relationship between entropy and amplification is required.

The work of Radhakrishnan [Rad91] suggests a new approach to computing lower bounds on formula size for threshold functions on $\{\wedge, \vee\}$. Can this work be extended to computing lower bounds for threshold functions using small majority gates?

Finally, we find the question of which initial distribution is used to be quite intriguing. Our results imply that no distribution on $sym(k,n)$ will improve the bounds. Therefore a natural problem is to investigate new classes of initial distributions.

Bibliography

[BB65] R. Beckenbach and R. Bellman. *Inequalities.* Springer Verlag, 1965.

[BDBK+90] S. Ben-David, A. Borodin, R. Karp, G. Tardos, and A. Wigderson. On the power of randomization in online algorithms. In *Proceedings of the* 22^{nd} *ACM Symposium on Theory of Computing*, pages 379–386, 1990.

[BG64] D. Blackwell and M. Girshick. *Theory of Games and Statistical Decisions.* John Wiley and Sons, Inc., 1964.

[BL67] Bers and Lipman. *Calculus.* Holt and Reinhart and Winston, 1967.

[Bop89] R. Boppana. Amplification of probabilistic boolean formulas. In S. Macali and F. Preparata, editors, *Advances in Computing Research*, volume 5, pages 27–45. Jai Press Inc., 1989.

[Bop91] R. Boppana, 1991. Private communication.

[CDRS90] D. Coppersmith, P. Doyle, P. Raghavan, and M. Snir. Random walks on weighted graphs, and application to on-line algorithms. In *Proceedings of the Twenty-Second Annual ACM Symposium on Theory of Computing*, pages 369–378, 1990.

[CT78] Y.S. Chow and H. Teicher. *Probability Theory.* Springer-Verlag, New York, 1978.

[DM91] X. Deng and S. Mahajan. Infinite games: Randomization, computability and applications to online problems. In *Proceedings of the Twenty-Fourth Annual ACM Symposium on Theory of Computing*, pages 289–298, 1991.

[DZ92] M. Dubiner and U. Zwick. Amplification and percolation. In *Thirty-Third Annual IEEE Symposium on Foundations of Computer Science*, 1992.

[Fri86] J. Friedman. Constructing $O(n \log n)$ size monotone formulae for the kth threshold function of n boolean variables. *SIAM Journal on Computing*, 15(3):641–654, 1986.

[FRR90] A. Fiat, Y. Rabani, and Y. Ravid. Competitive k-server algorithms. In *Proceedings of the 31st Symposium on Foundations of Computer Science*, pages 454–463, 1990.

[GM92] A. Gupta and S. Mahajan. Using amplification to compute majority with majority. In Preparation, 1992.

[GS53] D. Gale and F.M. Stewart. Infinite games with perfect information. In W.H. Kuhn and A.W. Tucker, editors, *Contributions to the Theory of Games Vol. II, Annals Mathematical Studies*, volume 28, pages 245–266. Princeton University Press, Princeton, New Jersey, 1953.

[Han64] G. Hansel. Nombre minimal de contacts de fermature nécessaires pour ráliser une fonction booléenne symétrique de n variables. *C.R. Acad. Sc. Paris 258*, pages 6037–6040, 1964.

[HT89] J.Y. Halpern and M.R. Tuttle. Knowledge, probability, and adversaries. In *Proceedings of the 8th Annual ACM Symposium on Principles of Distributed Computing*, pages 103–118, 1989.

[Khr72] V. Khrapchenko. A method of determining lower bounds for the complexity of π-schemes. *Math. Notes of the Academy of Sciences, USSR*, pages 474–479, 1972.

[Kri64] R. Krichevskii. Complexity of contact circuits realizing a function of logical algebra. *Soviet Physics Dokl.*, pages 770–772, 1964.

[Mar75] D.A. Martin. Borel determinacy. *Annals of Mathematics*, 102:363–371, 1975.

[MMS88] M.S. Manasse, L.A. McGeoch, and D.D. Sleator. Competitive algorithms for on-line problems. In *Twentieth ACM Annual Symposium on Theory of Computing*, pages 322–333, 1988.

[Mos80] Y.N. Moschovakis. *Descriptive Set Theory.* North-Holland, New York, 1980.

[MS56] E. Moore and C. Shannon. Reliable circuits using less reliable relays. *Journal of the Franklin Institute*, 262:191–208 and 281–297, 1956.

[PPZ90] M. Paterson, N. Pippenger, and U. Zwick. Faster circuits and shorter formulae for multiple addition, multiplication and symmetric Boolean functions. In 31^{st} *Symposium on the Foundations of Computer Science*, pages 642–650, 1990.

[Rad91] J. Radhakrishnan. Better bounds for threshold formulas. In 32^{nd} *Symposium on the Foundations of Computer Science*, pages 314–323, 1991.

[RS89] P. Raghavan and M. Snir. Memory vs. randomization in online algorithms. In *proceedings of the 16th ICALP*, 1989.

[ST85] D.D. Sleator and R.E. Tarjan. Amortized efficiency of list update and paging rules. *Communications of the ACM*, 28(2):202–208, 1985.

[Val84] L. Valiant. Short monotone formulae for the majority function. *Journal of Algorithms*, 5:363–366, 1984.