A semantic view of the switching lemma

Dimitris J. Kavvadias
djk@math.upatras.gr

Lina Panagopoulou
linapan@master.math.upatras.gr

Department of Mathematics, University of Patras, GR-265 00 Patras, Greece

Abstract

The Switching Lemma is a key result in proving lower bounds in circuit complexity. In this paper we approach the Switching Lemma from the standpoint of the semantics of the boolean function, i.e., its set of satisfying assignments. This novel approach, gives the exact bound probability when the boolean function is of a special form and may also lead to a simpler proof of the general case.

1 Introduction

Circuit complexity is the field of computational complexity which studies the limitations of a simple computational model, the *logical circuit*. Logical circuits provide a different approach in the study of lower bounds for computational problems than the more widely used Turing machine, partly because of their simplicity, and have given interesting results in the past.

A key tool in proving such bounds is the so-called Switching Lemma which was proven by J. Håstad in [Has86] by improving on earlier results found in [Ajt83, FSS81, Yao85]. In the literature it is now commonly referred to as *Håstad's Switching Lemma*.

The Switching Lemma deals with Boolean expressions that are in conjunctive normal form and more specifically have at most k literals in every clause (k-CNF expressions). In effect it says that if we randomly give values (0 or 1) to a large proportion of the variables which are also randomly selected (this procedure is called a random restriction), then the resulting formula is, with very high probability, s-DNF that is, it is equivalent to a Boolean expression in disjunctive normal form with at most s literals in every term. By employing De Morgan's law, a symmetric form of this lemma is easily derived, where the original expression is k-DNF and after the application of the random restriction, the resulting expression is s-CNF.

Using this lemma, lower bounds for small depth circuits can be shown. For example Håstad showed in [Has86] that parity is not in AC^0, meaning that it cannot be computed by constant depth, unbounded fan-in circuits with polynomial number of gates.

Håstad's original proof used quite involved arguments on conditional probabilities. Since Håstad's original proof, simpler proofs appeared by Razborov [Raz92] and Beame [Bea94]. Razborov's proof uses an information-theoretic argument to count bad assignments, while Beame's proof deals with the decision tree of the resulting formula.

In this paper we approach the Switching Lemma focusing on the *semantics* of the given k-CNF, that is, its set of satisfying assignments. The application of a random restriction is now equivalent to selecting an appropriate subset of the assignments and projecting out the fixed variables. Using a result from [KS99] we are able to rephrase the Switching Lemma as a property of binary vectors and calculate the bound probability in a special case. The approach may also prove usefull for the general case

In: L. Ferrari, M. Vamvakari (eds.): Proceedings of the GASCom 2018 Workshop, Athens, Greece, 18–20 June 2018, published at http://ceur-ws.org

The rest of the paper is organized as follows. In the next section we give the necessary definitions and notation. In Section 3 we give a novel formulation of the Switching Lemma in terms of binary vectors. In Section 4 we give the exact bound for a specific case and outline a simplified proof of the general bound from the bibliography. We conclude with a list of references.

2 Definitions

Let $X = \{x_1, \ldots, x_n\}$ be a set of Boolean variables. A *literal* is either a variable (called *positive* literal) or its negation (called *negative* literal). A *clause* is a disjunction of one or more literals such that each variable appears at most once, while a *term* is, analogously, a conjunction of literals. A *Boolean expression* may come in a variety of syntactic forms with more populars being the *conjunctive normal form* (CNF) and the *disjunctive normal form* (DNF). A CNF is a conjunction of clauses $\varphi = C_1 \wedge C_2 \wedge \cdots \wedge C_m$, while a DNF is a disjunction of terms $\varphi = T_1 \vee T_2 \vee \cdots \vee T_t$. In the first relation each $C_i, i = 1, ..., m$ is a clause, that is, $C_i = \ell_1 \vee \ell_2 \vee \cdots \vee \ell_k$, where each $\ell_j, j = 1, \ldots, k$ is a literal. If every clause in φ is the conjunction of at most k literals then we say that φ is in k-CNF. Similarly, each term $T_i, i = 1, \ldots, t$ is of the form $T_i = \ell_1 \wedge \cdots \wedge \ell_s$. If every term includes at most s literals then we say that φ is in s-DNF.

A *truth assignment* v is a function $v : X \longrightarrow \{0, 1\}$. We often view a binary vector in $\{0, 1\}^n$ as a truth assignment that assigns the i-th bit of v to the i-th variable in X, taken in lexicographic order.

The *semantics* of a Boolean expression φ are captured by its set of *satisfying assignments* $M(\varphi)$, the assignments which make the expression true. It is well known that for every Boolean expression there exist equivalent (i.e., with the same set of satisfying assignments) expressions in CNF and DNF. In this case the expression φ is called CNF (respectively, DNF) expressible and similarly, k-CNF (k-DNF) expressible when the number of literals in each clause (term) is bounded by k. We also call a set $M \subseteq \{0, 1\}^n$ of binary vectors, a kCNF set, if M is exactly the set of satisfying assignments of some k-CNF expression. Similarly for a kDNF set. We also use the notation $\overline{M}$ to denote the set $\{0, 1\}^n \setminus M$ i.e., the set of all n-dimensional vectors not in M.

Given a Boolean expression φ defined on a set of Boolean variables X, a *random restriction* ρ of φ, denoted $\varphi \,|_{\,\rho}$ is the Boolean expression obtained by randomly selecting with probability p every variable and assigning each selected variable the value 0 or 1 each with probability 1/2. Therefore, a random restriction ρ maps each variable x to the set $\{0, 1, *\}$ with the following probability distribution.

$$\Pr(\rho(\mathrm{x}) = 0) = \Pr(\rho(\mathrm{x}) = 1) = \mathrm{p}/2, \qquad \Pr(\rho(\mathrm{x}) = *) = 1 - \mathrm{p}$$

By $\rho(x) = *$ we denote the event that x was not selected by ρ.

The following is a definition from [KS99] that we shall need later.

Let n be a positive integer and let $M \subseteq \{0, 1\}^n$ be a set of Boolean vectors. For $k > 1$, we say that a Boolean vector $v \in \{0, 1\}^n$ is k-compatible with M if for any sequence of k positions $0 \leq i_1 < \ldots < i_k \leq n$, there exists a vector in M that agrees with v in these k positions.

The above definition implies that a vector $m \in \{0, 1\}^n$ is not k-compatible with a set of Boolean vectors M if there exists a sequence of k positions in m that does not agree with any vector of M.

3 A binary vectors formulation

The object of this paper is to show the following proposition.

Lemma 3.1 *(Håstad's Switching Lemma). Let φ be a k-CNF expressible Boolean expression. Let ρ be a random restriction that selects a variable of φ with probability p. Then for every $s \geq 2$.*

$$\Pr(\varphi \,|_{\,\rho} \text{ is not } s\text{-DNF}) \leq (5k(1 - p))^s$$

Instead of studying the effects of the restriction on the expression φ itself, we shall study the set $M(\varphi)$ of satisfying assignments of φ. To this end, the following result from [KS99] will prove useful.

Lemma 3.2 *Let $M \subseteq \{0, 1\}^n$ be a set of binary vectors. Then the following are equivalent:*

(a) M is a kCNF set.

(b) If $m \in \{0, 1\}^n$ is k-compatible with M, then $m \in M$.

By De Morgan's rule the negation of a k-DNF expression φ is k- CNF and since $M(\neg\varphi) = \{0, 1\}^n \setminus M(\varphi)$, we immediately get the symmetric lemma to the above.

Lemma 3.3 *Let $M \subseteq \{0,1\}^n$ be a set of binary vectors. Then the following are equivalent:*

(a) M is a kDNF set.

(b) If $m \in \{0,1\}^n$ is k-compatible with $\overline{M}$, then $m \in \overline{M}$.

Let M be the set of models of the expression φ (we shall henceforth use the simpler notation M instead of $M(\varphi)$ when no confusion arises) and let us fix the variable x of φ to a certain value, say 0. Obviously any model in M with value $x = 0$, with this value in the position of x projected out, is a model of $\varphi\,|_{x=0}$. Conversely, any model of $\varphi\,|_{x=0}$ augmented by the value 0 in the position of x (taken lexicographically), belongs in M. This observation is easily extended to any number of variables and hence the following fact holds.

Fact. The set of models of $\varphi\,|_\rho$ follows from M by selecting every model in M that agrees with ρ in every variable selected by ρ and projecting out all positions corresponding to these variables. By also dropping out any multiple appearence of a truncated vector, we get the set of models of $\varphi\,|_\rho$. Let us denote for simplicity by N this set of models of $\varphi\,|_\rho$. We call the above procedure on a set of binary vectors M a *random restriction on* M and denote it by $M\,|_\rho$, extending the notion to binary vectors. Therefore $N = M\,|_\rho$.

Using the above notation and Lemmas 2 and 3 we may reformulate Lemma 1 as follows.

Lemma 3.4 *(Håstad's Switching Lemma, semantic form). Let $M \subseteq \{0,1\}^n$ be a set of binary vectors with the property that every vector $m \in \overline{M}$ is not k-compatible with M. Let $M\,|_\rho$ be a random restriction on M that selects a position with probability p and let $N = M\,|_\rho$. Then for every $s \geq 2$.*

$$\Pr(N \text{ includes an } s\text{-compatible vector with } \overline{N}) \leq (5k(1-p))^s$$

Let M be a kCNF set. By Lemma 3.2 every vector in $\overline{M}$ is not k-compatible with M and therefore it has a tuple of k positions in which it disagrees with all models in M. Let T be such a k-tuple with the corresponding values. There exist 2^{n-k} vectors that have the same values in T and all belong in $\overline{M}$. It follows that every vector in $\overline{M}$ belongs in at least one family F_T of vectors that all are produced by fixing k positions to certain "disagreeing" values and assuming all possible values for the rest of the variables.

In order to understand the structure of N, it is useful to view φ as being empty initially, and consider its clauses as being added one by one. Initially, M consists of models that agree with ρ in all positions that are selected by ρ and have all possible values in every other position. Therefore $|M| = 2^t$, where t is the number of non-selected variables.

Let us now add a clause C in φ. First consider the case where C contains only selected variables. Now if there exists one variable of C that is assigned a satisfying value by ρ then C is, in effect, discarded as it does not alter M. If however C is not assigned a satisfying value, then this results in M becoming empty. We call this case as C being *matched* by ρ and obviously this results in N also being empty and ρ being a good restriction as this corresponds to a contradiction.

Let us now add a clause C whose variables are not all selected by ρ. Let V_1 be the set of variables of C selected by ρ and V_2 the rest. As before if ρ assigns a satisfying value to a variable of V_1, C again has no effect in M as it is satisfied. But if ρ gives falsifying values to all variables of V_1, then the unset variables of V_2 now form a new constraint. Therefore any vector that falsifies all these variables is excluded from N and it is included in $\overline{N}$. It follows that every vector in $\overline{N}$ belongs in at least one family F_{C_i} of vectors that are produced by fixing the positions of the unset variables of the clause C_i to certain disagreeing values. The structure of N is therefore similar to that of M but with the constraints in the role of the clauses.

In summary therefore, a random restriction ρ assigns values from $\{0,1,*\}$ to the variables of φ and it may have the following results on a clause C of φ.

i. It hits all of the variables of C by a falsifying value. In this case $C \equiv 0$ and therefore ρ is *good*.

ii. It hits C assigning a satisfying value to at least one of its variables. In this case $C \equiv 1$ and plays no role in $\varphi\,|_\rho$.

iii. It partially hits some (or none) of the variables of C by falsifying values. The rest of the variables are not hit (they are assigned *) and now form a clause of $\varphi\,|_\rho$ with falsifying values those of the same variables of C. We say that in this case C *survives* from ρ.

Of the above three listed cases only the first allows us to immediately decide whether ρ is good or bad. Case (ii), completely removes C and therefore reduces the size of the instance by one clause.

It is therefore the third case that is the most interesting. Let us denote by ℓ the number of surviving clauses.

It is clear from the above that $\varphi\,|_\rho$ has reduced to a CNF expression with ℓ clauses, those who have survived from ρ and in general each is a sub-clause of a clause of φ. If $\varphi\,|_\rho$ is unsatisfiable then ρ is obviously a good restriction. If now $\varphi\,|_\rho$ is satisfiable, consider applying the distributive law of conjunction. We get a DNF expression and since we assumed that ℓ clauses have survived, each term of this expression may have up to ℓ literals, i.e. it is an ℓDNF expression. We have thus shown the following.

Lemma 3.5 *If $\ell \leq s$ clauses survive in $\varphi\,|_\rho$ then ρ is a good restriction.*

If $\ell > s$ then in general $\varphi\,|_\rho$ will not be ℓDNF expressible unless the same literal appears in several clauses. It is clear therefore that

$$\Pr(\varphi\,|_\rho \text{ is not } s\text{DNF})) \leq \Pr(\ell > s \text{ clauses survive in } \varphi\,|_\rho)$$

Notice that in some cases, as for example the case where all clauses of φ are disjoint, the above relation holds as equality and we are able to calculate the *exact* probability of a bad restriction.

In the next section we calculate the above probability of the case where all clauses of φ are disjoint and outline how to bound the probability of $\ell > s$ for the general case.

4 The bound

In the previous section we have summarized the effects of ρ on a clause C. Let us calculate now the corresponding probabilities

i. A clause C, containing k literals is falsified when ρ has selected all its variables with a falsifying value to every one. The probability that a variable is selected and assigned a falsifying value is obviously $\frac{p}{2}$ and hence $\Pr(C \equiv 0) = (\frac{p}{2})^k$.

ii. By the above, it follows that the probability that at least one of the variables is assigned a satisfying value is $1 - (1 - \frac{p}{2})^k$ and therefore $\Pr(C \equiv 1) = 1 - (1 - \frac{p}{2})^k$.

iii. Let us denote this probability by P_s. By the above and since the three events partition the sample space, we get that $\Pr(C \text{ survives from } \rho) = P_s = (1 - \frac{p}{2})^k - (\frac{p}{2})^k$.

When all clauses of φ are disjoint, then selecting a variable by ρ in a clause C, is independent from selecting a variable in any other clause. In this case, ρ is good iff up to s clauses survive and this probability is calculated as follows.

Let m be the number of clauses of φ. If at least one of them is matched (falsified) by ρ then ρ is good. By case (i) above, this probability is $(\frac{p}{2})^k$ for a specific clause C and hence, the probability that at least one clause C (among the m clauses of φ) is falsified is
$\Pr(\text{at least one } C \text{ is matched by } \rho) = P_m = 1 - (1 - (\frac{p}{2})^k)^m$.

The other possibility (which is disjoint from the above) for ρ being good, is to have at most s clauses survive in $\varphi\,|_\rho$ and the rest of them not survive (by assigning to at least one variable of each clause a satisfying value). This probability for a specific selection of j clauses is $P_s^j \cdot (1 - (1 - \frac{p}{2})^k)^{m-j}$. Since the variable sets of any two clauses are disjoint, we simply need to sum over all possible selections of j clauses for $0 \leq j \leq s$. We thus have

$$\Pr(\varphi\,|_\rho \text{ is } s\text{DNF})) = P_m + \sum_{j=0}^{s} \binom{m}{j} \cdot P_s^j \cdot (1 - (1 - \frac{p}{2})^k)^{m-j}$$

and therefore,

$$\Pr(\varphi\,|_\rho \text{ is not } s\text{DNF})) = (1 - (\frac{p}{2})^k)^m - \sum_{j=0}^{s} \binom{m}{j} \cdot P_s^j \cdot (1 - (1 - \frac{p}{2})^k)^{m-j}.$$

It is easy to verify that indeed the previous probabiblity is strictly less than $(5k(1-p))^s$. □

The general case is more involved as the independence in selecting a variable between two clauses does no longer hold.

Let ℓ be the number of surviving clauses of φ after the application of ρ. Let m be the number of clauses of φ and let us denote by $e(\varphi, s)$ the event that exactly $\ell = s$ clauses survive in $\varphi\,|_\rho$ and all the rest $m - s$ are satisfied. Let us also denote by $E(\varphi, s)$ the event that $\ell > s$ clauses survive in $\varphi\,|_\rho$ and all the rest $m - \ell$ are satisfied. We would like to show that

Lemma 4.1

$$\Pr(E(\varphi, s)) \leq (5k(1-p))^s$$

We apply induction on the number m of clauses of φ.

Base. For $m = 1$, $\Pr(\ell \geq s+1) = 0$ for every $s \geq 1$.

Step. For $m > 1$ let us assume that for every kCNF expression with up to $m-1$ clauses (on any number of variables), Lemma 4.1 holds. Consider a kCNF expression φ with m clauses. Let C be one of its clauses. For ease of reference, we denote by $\varphi' = \varphi \setminus C$ (that is, the expression that results from φ when we remove C) and by $H_s = (5k(1-p))^s$.

We now have

$$\Pr(E(\varphi, s)) = \Pr(e(\varphi', s) \wedge (C \textit{ survives})) + \Pr(E(\varphi', s) \wedge (C \not\equiv 0))$$

Note that when more than s clauses have survived in φ', C must not be false in order to have a bad restriction. The event $C \not\equiv 0$ includes both the event $C \equiv 1$ and the event C *survives*. The first term is simplified as follows. Let $\mathcal{A}$ be the set of assignments to the variables of C from the set $\{0, 1, *\}$ that make C survive. Let α be an assignment from $\{0, 1, *\}$ to the variables of C. We denote by $e(\alpha)$ the event that ρ assigns the values of α to the variables of C.

$$\begin{aligned}
&\Pr(e(\varphi', s) \wedge (C \textit{ survives})) = \\
&\Pr(e(\varphi', s) \wedge \bigvee_{\alpha \in \mathcal{A}} e(\alpha)) = \Pr(\bigvee_{\alpha \in \mathcal{A}} (e(\varphi', s) \wedge e(\alpha))) \leq \\
&\sum_{\alpha \in \mathcal{A}} \Pr(e(\varphi', s) \wedge e(\alpha)) \leq \sum_{\alpha \in \mathcal{A}} \Pr(E(\varphi', s-1) \wedge e(\alpha)) = \\
&\sum_{\alpha \in \mathcal{A}} \Pr(E(\varphi', s-1) \mid e(\alpha)) \cdot \Pr(e(\alpha))
\end{aligned}$$

The last inequality holds since the second event contains the first. Observe now that if we substitute to the variables of C any values from $\{0, 1, *\}$, some clauses of φ' may be satisfied and thus removed from φ', and some variables may be removed from the remaining clauses either because they are unsatisfied, or they are assigned *. In any case the resulting CNF has at most $m-1$ clauses with at most k literals each and thus by induction Lemma 4.1 holds. There is a problem however that stems from the conditional probability: by setting specific values to some variables of φ, we no longer have a restriction with the same distribution for the remaining variables. There are ways to overcome this complication by splitting the assignment of values to specific sets of variables and summing over all possible such sets. The reader is referred to the original proof of Hastad or in [BS90]. Using these techinques we have

$$\sum_{\alpha \in \mathcal{A}} \Pr(E(\varphi', s-1) \mid e(\alpha)) \leq H_{s-1}$$

Hence

$$\sum_{\alpha \in \mathcal{A}} \Pr(E(\varphi', s-1) \mid e(\alpha)) \cdot \Pr(e(\alpha)) \leq H_{s-1} \sum_{\alpha \in \mathcal{A}} \Pr(e(\alpha)) = H_{s-1} P_s,$$

since the sum of the probabilities is over all assignments that make C survive. Similarly, the second term gives

$$\Pr(E(\varphi', s) \wedge (C \not\equiv 0)) \leq H_s(1 - (\frac{p}{2})^k)$$

The proof now rests on showing that

$$H_{s-1} P_s + H_s(1 - (\frac{p}{2})^k) \leq H_s,$$

or after substituting and simplifying

$$\left(1 - \frac{p}{2}\right)^k \leq (5k(1-p) + 1) \left(\frac{p}{2}\right)^k,$$

or equivalently

$$(5k(1-p)+1)\left(\frac{p}{2-p}\right)^k - 1 \geq 0.$$

This last inequality is easily shown by first proving that the left-hand side is an increasing function of k for all $p \in [0.8, 1]$ and thus by substituting the smallest permissible value of $k = 2$, it suffices to show that the produced function of p is increasing and positive for $p = 0.8$. This is indeed the case and it can be shown by taking the derivative over p.

References

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

[FSS81] M. Furst, J. Saxe, M. Sipser. Parity, circuits and the polynomial time hierarchy. *Mathematical Systems Theory*, 17:13-27, 1984, Prelim version FOCS '81.

[Has86] J. Hastad. Almost optimal lower bounds for small depth circuits. *STOC*, 6-20, May 1986.

[KS99] D. Kavvadias, M. Sideri. The Inverse Satisfiability Problem. *SIAM Journal on Computing*, 28:152-163, 1999.

[Bea94] P. Beame. *A switching lemma primer*. Technical Report UW-CSE-95-07-01, Department of Computer Science and Engineering, University of Washington, November 1994.

[Raz92] A. Razborov. An Equivalence between Second Order Bounded Domain Bounded Arithmetic and First Order Bounded Arithmetic. *In the volume Arithmetic, Proof Theory and Computational Complexity*, 247-277, 1992.

[Yao85] A. C. C. Yao. Separating the polynomial-time hierarchy by oracles. *FOCS*, 1-10, 1985.

[BS90] R. B. Boppana, M. Sipser. The complexity of finite functions. *Handbook of theoretical computer science (vol. A)*, 757-804, 1990.