

TECHNICAL REPORT No. 2003/07/02

CHECKING MONOTONE BOOLEAN DUALITY WITH LIMITED
NONDETERMINISM

Dimitris J. Kavvadias¹ and Elias C. Stavropoulos^{2,*}

July, 2003

Abstract

We present an algorithm that checks the duality of a pair of monotone Boolean expressions in disjunctive normal form in polynomial time plus $O(\log^2 n / \log \log n)$ nondeterministic guesses, where n is the size of the input, thus improving a previous result of Kavvadias and Stavropoulos [IPL, 85(1):1–6, 2003] and Eiter et al. [STOC 2003] which placed the problem in $\text{co-NP}[\log^2 n]$, the subclass of co-NP where only the first $O(\log^2 n)$ steps are nondeterministic.

¹ University of Patras, Department of Mathematics, GR-265 04 Patras, Greece.
E-mail: djk@math.upatras.gr

² University of Patras, Computer Engineering & Informatics Department, GR-265 04 Patras, Greece. E-mail: estavrop@ceid.upatras.gr

* Corresponding author.

1 Introduction

Let $f(x) = f(x_1, \dots, x_N)$ and $g(x) = g(x_1, \dots, x_N)$ be a pair of monotone Boolean expressions given by their irredundant disjunctive normal forms

$$f = \bigvee_{I \in F} \bigwedge_{i \in I} x_i \quad \text{and} \quad g = \bigvee_{J \in G} \bigwedge_{j \in J} x_j,$$

where F and G are the sets of prime implicants $I, J \subseteq \{1, \dots, N\}$ of f and g , respectively. The problem of interest here is defined as follows:

MONOTONE BOOLEAN DUALITY (MBD) Given a pair of monotone Boolean expressions f and g in their irredundant disjunctive normal forms, decide whether f, g are mutually dual, i.e.,

$$f(x_1, \dots, x_N) = \bar{g}(\bar{x}_1, \dots, \bar{x}_N) \quad \text{for all } x = (x_1, \dots, x_N) \in \{0, 1\}^N.$$

If f and g are not mutually dual, then there is a vector $x \in \{0, 1\}^N$ such that $f(x_1, \dots, x_N) = g(\bar{x}_1, \dots, \bar{x}_N)$. Obviously, such a disqualifier can be guessed and verified in time that is a polynomial to the size $n = |F| + |G|$ of f and g . Thus, MONOTONE BOOLEAN DUALITY is easily placed in the class co-NP. However, its exact computational complexity is still unknown. The most notable result was given in 1996 by Fredman and Khachiyan [7]. In this work, the authors presented an algorithm that checks the duality of a pair of monotone DNF expressions in quasi-polynomial time $n^{o(\log n)}$. This result gives evidence that MONOTONE BOOLEAN DUALITY lies in an intermediate class between P and co-NP.

The algorithm of Fredman and Khachiyan can also be used for enumerating the prime implicants of the dual expression of a monotone DNF in *incremental output-subexponential* time [10]. Since the size of the dual expression may be exponentially larger than the input one, more elaborate complexity measures for the efficiency of algorithms for problems like MONOTONE BOOLEAN DUALITY (i.e., with large output) must be defined, that will take into account not only the size of the input but the size of the output as well. The reader is referred to see [11, 16] for discussions of algorithms with large output and performance criteria.

Generating the prime implicants of a monotone DNF is equivalent to the generation of the *minimal transversals* of a simple hypergraph [2, 7, 13] and to the generation of the *maximal models* of a Boolean expression in conjunctive normal form [12]. These problems are central in various fields of Computer Science (see [6, 7, 9] for an exposition of applications of these problems).

The deterministic algorithm of Fredman and Khachiyan gives an upper bound for the time complexity of MONOTONE BOOLEAN DUALITY and implies that the problem can not be co-NP-hard, unless any co-NP-complete problem can be solved in quasi-polynomial time. In this paper we present a nondeterministic algorithm for checking the duality of a pair of monotone Boolean expressions in DNF. Our algorithm uses the decomposition rules of [7] (see next section) in a novel way and solves the problem in deterministic polynomial time plus $O(\log^2 n / \log \log n)$ nondeterministic steps. This result improves the previous result of Kavvadias and Stavropoulos [14] and Eiter et al. [4] which placed MONOTONE BOOLEAN DUALITY in co-NP[$\log^2 n$]. The class NP[$\log^2 n$], denoted $\beta_2\text{P}$ in [15], is the subclass of NP where only the first $O(\log^2 n)$

steps are nondeterministic. Such subclasses of NP can be defined by restricting the number of nondeterministic steps of the computation (see [15, 3]). For a survey on limited nondeterminism, see [8].

The same nondeterministic time bound was also given independently by Eiter et al. in [5]. Our approach differs from the one in [5] and its analysis is much simpler. Moreover, as it is mentioned there, the same result may also be obtained by appropriately applying Beigel and Fu's Theorem 11 in [1]. Having the above time bound, it is subsequently straightforward to obtain the $n^{o(\log n)}$ deterministic time bound of [7], thus avoiding the rather complicated analysis presented there.

The rest of the paper is organized as follows: In Section 2 we present the necessary duality properties and lemmas and shortly describe the algorithm of Fredman and Khachiyan. In the next section we present our nondeterministic version and give its time complexity. Finally, in Section 4 some conclusions are given.

2 Overview of the Fredman and Khachiyan algorithm

For the sake of completeness, in this section we briefly describe the main steps of the algorithm of Fredman and Khachiyan. The reader is referred to [7] for more details. Terminology and notation are also borrowed from there.

Suppose that the monotone DNF expressions f and g are mutually dual. Then, the following conditions hold:

$$I \cap J \neq \emptyset, \text{ for any } I \in F \text{ and } J \in G \quad , \quad (1)$$

$$\cup\{I : I \in F\} = \cup\{J : J \in G\} \quad , \quad \text{and} \quad (2)$$

$$\max\{|I| : I \in F\} \leq |G|, \max\{|J| : J \in G\} \leq |F| \quad . \quad (3)$$

Moreover (cf. [7], Lemma 1),

$$E = \sum_{I \in F} 2^{-|I|} + \sum_{J \in G} 2^{-|J|} \geq 1. \quad (4)$$

If any of these necessary duality conditions is violated, then f and g are not mutually dual and a succinct disqualifier can be found in polynomial time.

Let $x_i \in \{x_1, \dots, x_N\}$. The *frequency* ϵ_i^f of x_i in f is defined as the fraction of implicants of F that x_i occurs in, i.e.,

$$\epsilon_i^f = \frac{|\{I \in F : i \in I\}|}{|F|}.$$

Let $\epsilon \in (0, 1]$. We say that x_i occurs in f with *frequency at least* ϵ if $\epsilon_i^f \geq \epsilon$. The following lemma states an interesting property that holds for mutually dual expressions (cf. [7], Lemma 2):

Lemma 1 *Let f, g be a pair of mutually dual forms with $|F||G| \geq 1$. Then, there exists a variable that occurs either in f or g with frequency at least $1/\log n$, where $n = |F| + |G|$ is the size of f and g .*

Let $x_i \in \{x_1, \dots, x_N\}$. Then, the expressions f and g can be written as

$$f = x_i f_0(y) \vee f_1(y) \quad \text{and} \quad g = x_i g_0(y) \vee g_1(y),$$

where $y = \{x_1, \dots, x_{i-1}, x_{i+1}, \dots, x_N\}$ and f_0, f_1, g_0 , and g_1 are the monotone irredundant DNFs with implicant sets

$$F_0 = \{I \setminus \{i\} \mid i \in I, I \in F\}, \quad F_1 = \{I \mid i \notin I, I \in F\},$$

$$G_0 = \{J \setminus \{i\} \mid i \in J, J \in G\}, \quad \text{and} \quad G_1 = \{J \mid i \notin J, J \in G\},$$

respectively. It was shown in [7] that f, g are mutually dual if and only if

$$f_1 \text{ is dual to } g_0 \vee g_1 \quad \text{and} \quad g_1 \text{ is dual to } f_0 \vee f_1. \quad (5)$$

Hence, the initial problem (f, g) of size n is reduced to subproblems

$$(f_1, g_0 \vee g_1) \quad \text{and} \quad (6)$$

$$(g_1, f_0 \vee f_1) \quad (7)$$

of smaller sizes, where x_i acts like a *splitting* variable. If both pairs are mutually dual, then so is the initial one; otherwise a succinct disqualifier can be found.

Fredman and Khachiyan presented an algorithm [7, Algorithm A] that utilizes Lemma 1 and recursively applies decomposition rule (5) to solve MBD in time $n^{O(\log^2 n)}$ for any pair of monotone disjunctive normal forms f and g of size at most n [7, Lemma 4]. As they next show, this running time can be further improved if one notices that subproblems $(f_1, g_0 \vee g_1)$ and $(g_1, f_0 \vee f_1)$ are not independent. Assuming, for example, that subproblem (6) is already solved (and, f_1 is dual to $g_0 \vee g_1$), then the solvability of subproblem (7) is equivalent to the solvability of a system of $|G_0|$ equations

$$g_1(y[J]) = f_0(\bar{y}[J]), \quad (8)$$

where G_0 is the set of prime implicants of g_0 , $J \in G_0$, and $y[J]$ is the vector obtained by y by the substitution $y_j = 1$ for all $j \in J$. However, each of the $|G_0|$ equations (8) is equivalent to MBD for the pair of forms (g_1^J, f_0^J) where g_1^J is obtained from $g_1(y)$ by setting $y_j = 1, j \in J$, and f_0^J is obtained from $f_0(y)$ by setting $y_j = 0, j \in J$. Thus, the initial problem (f, g) has been decomposed into $|G_0| + 1$ subproblems in total.

A symmetric decomposition holds if one assumes that subproblem (7) is already solved. The initial problem (f, g) can now be decomposed into $|F_0| + 1$ subproblems in total. We next give the decomposition rules, as presented in [7]:

- (i) Let f, g be a pair of monotone DNFs of *volume* $v = |F||G|$ and let a variable x_i occur in f with frequency ϵ_i^f . Then, in polynomial time the MBD problem for $f = x_i f_0(y) \vee f_1(y)$ and $g = x_i g_0(y) \vee g_1(y)$ can be decomposed into subproblem (6) of volume $|F_1||G| \leq (1 - \epsilon_i^f)|F||G| = (1 - \epsilon_i^f)v$, plus $|G_0|$ subproblems (g_1^J, f_0^J) of volume at most $|F_0||G_1| = \epsilon_i^f|F||G| \leq \epsilon_i^f v$ each.

The symmetric decomposition rule for g is as follows:

- (ii) If a variable x_i occurs in g with frequency ϵ_i^g , then in polynomial time the MBD problem for (f, g) can be decomposed into subproblem (7) of volume at most $(1 - \epsilon_i^g)v$, plus $|F_0|$ subproblems (f_1^f, g_0^f) of volume at most $\epsilon_i^g v$ each.

Finally, property (5) implies that

- (iii) The MBD problem for (f, g) can be decomposed into subproblems (6) and (7) of volumes $(1 - \epsilon_i^f)v$ and $(1 - \epsilon_i^g)v$, respectively.

The volume $v = |F||G|$ of f and g is an appropriate measure for the size of the input adopted in [7] and facilitates the analysis of the algorithms. We must note here that any variable with positive frequency may be used in the above rules. Algorithm B in [7] solves the MBD problem by recursively incorporating the above decomposition rules, until every subproblem that is produced has constant size in which case it can be solved in constant time. After a rather complicated analysis, Fredman and Khachiyan show that Algorithm B solves the MBD problem in quasi-polynomial time $n^{o(\log n)}$:

Theorem 1 ([7], Theorem 1) *The MONOTONE BOOLEAN DUALITY problem can be solved in $n^{4\chi(n)+O(1)}$ time, where $\chi(n)^{\chi(n)} = n$.*

3 Duality checking with limited nondeterminism

In this section we present a nondeterministic algorithm for checking the duality of a pair of monotone Boolean expressions in disjunctive normal form. Our algorithm proceeds in phases which are deterministic. Each phase starts with a single subproblem, say (f, g) , on which and on all of its descendant subproblems we apply the appropriate decomposition rule until all produced subproblems have volume at most a fraction t , $0 < t < 1$, of the volume v of the initial problem of the phase. (Actually, t is a function of the volume v of the initial problem and selecting the appropriate value for it is crucial.) As stated in the previous section, the duality of the initial problem is now equivalent to the duality of all produced subproblems, each of them having volume at most tv . Thus, if f, g are not mutually dual, then at least one of the produced subproblems is not dual. At this point we *nondeterministically* select one of the produced subproblems and check its duality in a new phase. The above are repeated until the selected subproblem is reduced to constant volume. We explain the whole procedure in detail in the sequel.

3.1 The deterministic phase

Let (f, g) be the initial problem of a deterministic phase, i.e., a pair of monotone Boolean expressions in DNF of volume $v = |F||G|$. Let also a variable x_i that occurs in f, g with frequencies $\epsilon_i^f > 0$ and $\epsilon_i^g > 0$, respectively. Suppose, without loss of generality, that $\epsilon_i^g \leq \epsilon_i^f$. We select and apply a decomposition rule to the initial problem (f, g) so that the number of produced subproblems having volume greater than tv each is minimized. Which decomposition rule is suitable depends on the frequencies of x_i in f, g . Frequencies ϵ_i^f and ϵ_i^g can be ordered among the quantities t and $1 - t$ according to the following six distinct cases (see Figure 1):

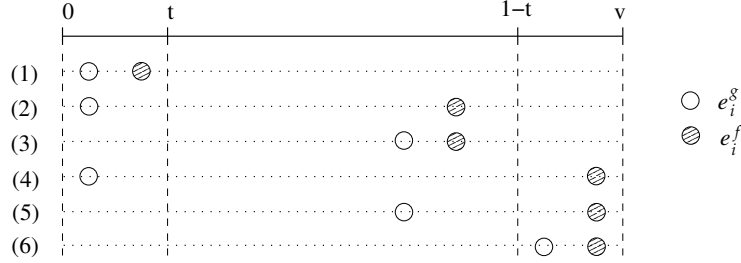


Figure 1: Ordering of frequencies ϵ_i^f and ϵ_i^g of an arbitrary variable x_i in f and g , respectively.

1. $0 < \epsilon_i^g \leq \epsilon_i^f \leq t$. The application of decomposition rule (i) would result in one subproblem of volume $(1 - \epsilon_i^f)v$ plus at most v subproblems of volume $\epsilon_i^f v$ each. In this case, $(1 - \epsilon_i^f)v \leq v - 1$ while $\epsilon_i^f v \leq tv$. Thus, decomposition rule (i) results in one "large" subproblem plus $O(v)$ "small" ones.

Notice that the same results (concerning the volumes of the produced subproblems as well as the number of them) can be obtained if decomposition rule (ii) is applied. On the contrary, by applying decomposition rule (iii), two large subproblems would emerge: one of volume $(1 - \epsilon_i^f)v \leq v - 1$ and another one of volume $(1 - \epsilon_i^g)v \leq v - 1$.

2. $0 < \epsilon_i^g \leq t < \epsilon_i^f \leq 1 - t$. The application of decomposition rule (ii) results in one large subproblem of volume $(1 - \epsilon_i^g)v \leq v - 1$ plus $O(v)$ small subproblems of volume $\epsilon_i^g v \leq tv$ each.
3. $t < \epsilon_i^g \leq \epsilon_i^f \leq 1 - t$. The application of decomposition rule (iii) results in two subproblems, one of volume $(1 - \epsilon_i^f)v$ and another one of volume $(1 - \epsilon_i^g)v$. Since $t < \epsilon_i^f$, we have that $(1 - \epsilon_i^f)v \leq (1 - t)v$. Also, since $t < \epsilon_i^g$, it is $(1 - \epsilon_i^g)v \leq (1 - t)v$. Thus, in this case, decomposition rule (iii) results in two large subproblems, each of them having volume at most $(1 - t)v$.

Notice that if decomposition rule (i) is applied, then the produced subproblems ($O(v)$ in total) are all large ones since $(1 - \epsilon_i^f)v \leq v - 1$ and $\epsilon_i^f v \leq v - 1$, too. This situation also holds for decomposition rule (ii).

4. $0 < \epsilon_i^g \leq t < 1 - t < \epsilon_i^f$. The application of decomposition rule (iii) results in one small subproblem of volume $(1 - \epsilon_i^f)v \leq tv$ (since $1 - t < \epsilon_i^f$) and in one large subproblem of volume $(1 - \epsilon_i^g)v \leq v - 1$.
5. $t < \epsilon_i^g \leq 1 - t < \epsilon_i^f$. The application of decomposition rule (iii) results in one small subproblem of volume $(1 - \epsilon_i^f)v \leq tv$ (since $1 - t < \epsilon_i^f$) and in one large subproblem of volume $(1 - \epsilon_i^g)v \leq (1 - t)$ (since $t < \epsilon_i^g$).
6. $1 - t < \epsilon_i^g \leq \epsilon_i^f$. The application of decomposition rule (iii) results in two subproblems with volumes $(1 - \epsilon_i^f)$ and $(1 - \epsilon_i^g)$, respectively. Since $1 - t < \epsilon_i^g$ and $1 - t \leq \epsilon_i^f$, the volumes of both subproblems are at most tv .

Case	# subproblems			Rule
	volume $\leq tv$	volume $\leq (1-t)v$	volume $\leq v-1$	
1. $0 < \epsilon_i^g \leq \epsilon_i^f \leq t$	$O(v)$	—	1	i
2. $0 < \epsilon_i^g \leq t < \epsilon_i^f \leq 1-t$	$O(v)$	—	1	ii
3. $t < \epsilon_i^g \leq \epsilon_i^f \leq 1-t$	—	2	—	iii
4. $0 < \epsilon_i^g \leq t < 1-t < \epsilon_i^f$	1	—	1	iii
5. $t < \epsilon_i^g \leq 1-t < \epsilon_i^f$	1	1	—	iii
6. $1-t < \epsilon_i^g \leq \epsilon_i^f$	2	—	—	iii

Table 1: Total number of subproblems emerging in Cases 1–6 after the application of the appropriate decomposition rule.

The number of subproblems emerging in every case is summarized in Table 1. Procedure **Deterministic Phase** (see Algorithm 1) takes as input a problem (f, g) of volume v and recursively combines decomposition rules (i)–(iii) according to Cases (1)–(6) until the volume of every descendant subproblem is at most tv . See that Case 6 gives no large subproblem while in all other cases, at least one large subproblem emerges (Case 3 gives two large ones). Its volume is either at most $(1-t)v$, or at most $v-1$. By suitably applying decomposition rules (i)–(iii), we achieve in each case to minimize the number of the large descendant subproblems. We next proceed in upper bounding the number of subproblems produced by a deterministic phase as a function of the volume of the initial problem. To do this, the value of the fraction t has to be determined. For the rest of our analysis, we set $t = \frac{1}{\log v}$. This is the optimal value with respect to minimizing the number of nondeterministic guesses of our approach (the optimization procedure is omitted).

Lemma 2 *The number of the subproblems produced by a Deterministic Phase is $O(v^2)$, where v is the volume of the initial problem of the phase.*

Proof We first proceed in upper bounding the number of subproblems produced by a deterministic phase whose initial problem has volume v . By examining Table 1, we notice that a problem of size greater than tv can only give two large subproblems by the application of Case 3. In all other cases at most one large subproblem may result. Viewing, therefore, the whole decomposition process as a tree, we notice that an internal (i.e., non-leaf) node of the tree gives only one large descendant (small subproblems are not considered as being part of the tree) except when Case 3 is applied, in which case it gives two. Consider now a path from the root of the tree to one of its leaves. The maximum number of times that Case 3 was applied along this path, denote it by k , is the solution of the equation

$$(1-t)^k v = tv.$$

Solving for k we get that $k = \log_{1-t} t$. The number of nodes now of any layer of the tree can only double in the next layer, if we assume that in all these subproblems, Case 3, was applied.

Therefore, the maximum number of subproblems in any layer of the tree is

$$2^k = 2^{\log_{1-t} t} = t^{-\log(1-t)}.$$

Notice that this holds even though the height of the tree can be greater than k . (Actually, the tree may have height at most $v - tv$.) To see this, consider an internal node u with only one descendant. Eliminate this node, hanging its descendant directly from its ancestor. Also add one new leaf-node to the leaves that are in the subtree of u . Notice that with this operation the total number of nodes in every level has not decreased. Repeating this operation on every original internal node with only one descendant, results in a tree with the property that, if we go down from the root along any path, the nodes that we meet have two descendants until the first node with only one descendant is found from which point on, all nodes down to the leaves have only one descendant. Such a tree has nodes with two descendants at depth at most k from the root. After that, all nodes have only one descendant. Therefore, the maximum number of nodes in every layer is 2^k , as claimed above.

The height of the tree on the other hand is at most $v - tv$ which results that an upper bound on the number of nodes of the tree is

$$(v - tv)t^{-\log(1-t)} \leq vt^{-\log(1-t)}.$$

Since each such node may contribute at most $O(v)$ small subproblems, we conclude that a deterministic phase may produce at most

$$v^2 t^{-\log(1-t)}$$

small subproblems of size tv or less. By substituting $t = \frac{1}{\log v}$, the maximum number of subproblems is

$$v^2 \left(\frac{1}{\log v}\right)^{-\log\left(1 - \frac{1}{\log v}\right)}$$

which, after some algebraic manipulation, is equal to

$$v^2 \left(1 - \frac{1}{\log v}\right)^{\log \log v}.$$

However,

$$\left(1 - \frac{1}{\log v}\right)^{\log \log v} = \left(1 - \frac{1}{\log v}\right)^{\log v \frac{\log \log v}{\log v}} = \left[\left(1 - \frac{1}{\log v}\right)^{\log v}\right]^{\frac{\log \log v}{\log v}}$$

When v tends to infinity, then $\left(1 - \frac{1}{\log v}\right)^{\log v}$ tends to e^{-1} and, thus, the above term tends to 1. Thus, the total number of subproblems of a deterministic phase is $O(v^2)$. ■

3.2 The nondeterministic algorithm

The current **Deterministic Phase** lasts until the volume of every subproblem produced is at most tv_c , where v_c is the volume of the initial problem (f_c, g_c) of the phase and $0 < t < 1$. At the start of the algorithm, $(f_c, g_c) = (f, g)$, where (f, g) is the input problem of volume v . At this point a nondeterministic oracle is called to guess the next subproblem for the algorithm to proceed (and the next deterministic phase to start). This nondeterministic guessing is used only at the end of each deterministic phase. We summarize the whole procedure in Algorithm 2.

Each phase results (after the nondeterministic selection) in a subproblem of volume at most tv , where v is the volume of the parent subproblem. Thus, the volume of the largest subproblem decreases according to the sequence

$$v, v' = tv, v'' = tv' = t^2v, v''' = tv'' = t^3v, \dots,$$

where v is the volume of the input problem. Hence, the number of phases required for the **Nondeterministic Algorithm** to end is the solution l of the equation $t^l v = c$, where c is a constant. By solving it, we obtain that

$$l = \log_t c - \log_t v = O(-\log_t v) = O\left(\frac{\log v}{\log \log v}\right).$$

We have, thus, proven the following lemma:

Lemma 3 *The number of phases required for the **Nondeterministic Algorithm** to solve the MONOTONE BOOLEAN DUALITY problem is $O\left(\frac{\log v}{\log \log v}\right)$, where v is the volume of the input problem.*

As Lemma 2 states, the number of the subproblems produced by a deterministic phase is $O(v_c^2)$, where v_c is the volume of the initial problem of the phase. Since $v_c \leq v$, the end of each deterministic phase there are $O(v^2)$ subproblems in total. Thus, the number of nondeterministic bits required for each deterministic phase is $O(\log v)$. By applying the previous lemma, we get that the total number of nondeterministic steps is $O\left(\frac{\log^2 v}{\log \log v}\right) = O\left(\frac{\log^2 n}{\log \log n}\right)$, where $n = |F| + |G|$ is the size of the input problem. We have thus proved the following theorem:

Theorem 2 *The **Nondeterministic Algorithm** solves the MONOTONE BOOLEAN DUALITY problem in polynomial time plus $O\left(\frac{\log^2 n}{\log \log n}\right)$ nondeterministic guesses, where n is the size of the input.*

Proof Follows from the above discussion. ■

Thus, $O(\log^2 n / \log \log n)$ nondeterministic guesses suffice to find a succinct disqualifier and prove that the input pair of monotone Boolean expressions in DNF are not mutually dual. This nondeterministic time bound improves the result presented in [14] and, independently, in [4], reducing the nondeterministic guesses from $O(\log^2 n)$ to $O(\log^2 n / \log \log n)$. The same nondeterministic time bound was independently also given by Eiter et al. in [5].

4 Conclusion

The aim of this paper was to exploit and identify the limits of nondeterminism required for solving the problem of checking the duality of two monotone Boolean expressions in disjunctive normal form. We presented an algorithm that utilizes the decomposition rules given by Fredman and Khachiyan in [7] as well as the appearance of variables with appropriate frequency. Our algorithm checks monotone Boolean duality in deterministic polynomial time plus $O(\log^2 n / \log \log n)$ nondeterministic steps. This result places the problem in the subclass of co-NP where only the first $O(\log^2 n / \log \log n)$ steps are nondeterministic and improves the previous result given in [14, 4]. It also makes straightforward the quasi-polynomial running time of the deterministic version, avoiding the rather complicated analysis presented in [7]. Future work includes further investigation on the exact time complexity of the monotone Boolean duality, a long unresolved question.

References

- [1] R. Beigel and B. Fu. Molecular computing, bounded nondeterminism, and efficient recursions. *Algorithmica*, 25:222–238, 1999.
- [2] J. C. Bioch and T. Ibaraki. Complexity of identification and dualization of positive Boolean functions. *Information and Computation*, 123:50–63, 1995.
- [3] J. Diaz and J. Torán. Classes of bounded nondeterminism. *Mathematical Systems Theory*, 23:21–32, 1990.
- [4] T. Eiter, G. Gottlob, and K. Makino. New results on monotone dualization and generating hypergraph transversals. In *Proc. of 34th ACM Symposium on Theory of Computing*, Montreal, Quebec, Canada, May 19–21, 2002.
- [5] T. Eiter, G. Gottlob, and K. Makino. New results on monotone dualization and generating hypergraph transversals. *SIAM J. Computing*, 32(2):514–537, 2003.
- [6] T. Eiter and G. Gottlob. Identifying the minimal transversals of a hypergraph and related problems. *SIAM J. Computing*, 24(6):1278–1304, December 1995.
- [7] M. L. Fredman and L. Khachiyan. On the complexity of dualization of monotone disjunctive normal forms. *Journal of Algorithms*, 21:618–628, 1996.
- [8] J. Goldsmith, M. A. Levy, and M. Mundhenk. Limited nondeterminism. *ACM SIGACT News*, 27(2):20–29, June 1996.
- [9] D. Gunopulos, R. Khardon, H. Mannila, and H. Toivonen. Data mining, hypergraph transversals, and machine learning. In *Proc. of Sixteenth ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems*, pages 209–216, Tucson, Arizona, USA, May 12–14, 1997.

-
- [10] V. Gurvich and L. Khachiyan. On generating the irredundent conjunctive and disjunctive normal forms of monotone Boolean functions. Technical Report LCSR-TR-251, Department of Computer Science, Rutgers University, 1995.
- [11] D. S. Johnson, M. Yannakakis, and C. H. Papadimitriou. On generating all maximal independent sets. *Information Processing Letters*, 27:119–123, March 1988.
- [12] D. J. Kavvadias, M. Sideri, and E. C. Stavropoulos. Generating all maximal models of a Boolean expression. *Information Processing Letters*, 74(3-4):157–162, May 2000.
- [13] D. J. Kavvadias and E. C. Stavropoulos. Evaluation of an algorithm for the transversal hypergraph problem. In J. S. Vitter and C. D. Zaroliagis, editors, *Proc. of 3th International Workshop on Algorithm Engineering (WAE99)*, LNCS, pages 72–84, London, UK, 1999. Springer-Verlag.
- [14] D. J. Kavvadias and E. C. Stavropoulos. Monotone Boolean dualization is in $\text{co-NP}[\log^2 n]$. *Information Processing Letters*, 85(1):1–6, January 2003.
- [15] C. M. R. Kintala and P. Fischer. Refining nondeterminism in relativized polynomial-time bounded computations. *SIAM J. Computing*, 9:46–53, 1980.
- [16] C. H. Papadimitriou. NP-completeness: A retrospective. In *Proc. of 24th International Colloquium on Automata, Languages, and Programming*, pages 2–6, Bologna, Italy, July 7-11, 1997.

Deterministic Phase

Input: a pair of monotone Boolean expressions f and g in DNF satisfying the necessary duality condition (1).

Output: a family P of subproblems (i.e., pairs of monotone Boolean expressions in DNF) each of volume at most a portion t of the volume of the input such that the input pair is dual if and only if all pairs in the family are dual.

- 1: Delete all redundant implicants from F and G and set $n = |F| + |G|$ and $v = |F||G|$.
- 2: Check conditions (2), (3), and (4). If any of these conditions is violated, then f, g are not mutually dual and a succinct disqualifier can be found in polynomial time.
- 3: If $\min\{|F|, |G|\} \leq 2$, then the MBD problem can be solved in polynomial time.
- 4: Select a variable x_i appearing in f, g with positive frequencies ϵ_i^f and ϵ_i^g , respectively.
- 5: **switch** ($\epsilon_i^f, \epsilon_i^g$)
 - case** $0 < \epsilon_i^g \leq \epsilon_i^f \leq t$: Apply decomposition rule (i) to obtain one subproblem of volume at most $v - 1$ and $O(v)$ subproblems of volume at most tv each.
 - case** $0 < \epsilon_i^g \leq t < \epsilon_i^f \leq 1 - t$: Apply decomposition rule (ii) to obtain one subproblem of volume at most $v - 1$ and $O(v)$ subproblems of volume at most tv each.
 - case** $t < \epsilon_i^g \leq \epsilon_i^f \leq 1 - t$: Apply decomposition rule (iii) to obtain two subproblems of volume at most $(1 - t)v$
 - case** $0 < \epsilon_i^g \leq t < 1 - t < \epsilon_i^f$: Apply decomposition rule (iii) to obtain one subproblem of volume at most tv and one subproblem of volume at most $v - 1$.
 - case** $t < \epsilon_i^g \leq 1 - t < \epsilon_i^f$: Apply decomposition rule (iii) to obtain one subproblem of volume at most tv and one subproblem of volume at most $(1 - t)v$.
 - case** $1 - t < \epsilon_i^g \leq \epsilon_i^f$: Apply decomposition rule (iii) to obtain two subproblems of volume at most tv each.
- end switch**
- 6: Add the produced subproblems to P .
- 7: Apply Steps 1–6 to every subproblem of volume greater than tv . If such problem does not exist (i.e., the phase has ended), return P .

Algorithm 1: The Deterministic Phase.

Nondeterministic Algorithm

Input: a pair of monotone Boolean expressions f and g in DNF satisfying the necessary duality condition (1).

- 1: Nondeterministically guess $O(\log^2 v / \log \log v)$ bits and store them.
- 2: Apply the **Deterministic Phase** for the current problem (f_c, g_c) and let P be the family of the produced subproblems.

Comment: At the first run of the algorithm, $(f_c, g_c) = (f, g)$.

- 3: Utilize the first $\log |P|$ bits stored in Step 1 to identify a subproblem in P and make it the next current problem (f_c, g_c) . Delete these bits from the stored sequence of bits.
- 4: Go to Step 2.

Algorithm 2: The Nondeterministic Algorithm.