Recognition and Dualization of Disguised Bidual Horn Functions*

Thomas Eiter† Toshihide Ibaraki‡ Kazuhisa Makino§

Abstract

We consider the problem of dualizing a Boolean function f given by CNF, i.e., computing a CNF for its dual f^d. While this problem is not solvable in quasi-polynomial total time in general (unless SAT is solvable in quasi-polynomial time), it is so in case the input belongs to special classes, e.g., the class of bidual Horn CNF φ [9] (i.e., both φ and its dual φ^d represent Horn functions). In this paper, we show that a disguised bidual Horn CNF φ (i.e., φ becomes a bidual Horn CNF after renaming of variables) can be recognized in polynomial time, and its dualization can be done in quasi-polynomial total time. We also establish a similar result for dualization of prime CNFs.

Keywords: algorithm, output-polynomial, Boolean function, dualization, Horn function, and bidual Horn function.

1 Introduction

Dualization of a Boolean function $f : \{0,1\}^n \to \{0,1\}$ is the problem of computing a conjunctive normal form (CNF) of f^d from a given CNF of f, where $f^d(x) = \overline{f}(\overline{x})$. By the law of De Morgan, a disjunctive normal form (DNF) of f^d is obtained from a CNF of f by interchanging (1) disjunctions and conjunctions, and (2) constants 0 and 1, and we can regard this problem as computing a DNF ψ of f from a given CNF φ of f.

Dualization is a fundamental problem in Boolean theory, which has been extensively studied and is, on certain classes of functions, polynomially equivalent to many other important problems encountered in various fields such as hypergraph theory, operations research, artificial intelligence, database theory, and reliability theory; for example, computing an Armstrong relation for a given set of functional dependencies (see e.g. [7]), or computing a prime implicant cover for a set of clauses in knowledge compilation (cf. [5]). As well-known, the size of the output DNF ψ can be exponentially larger than the size of the input CNF φ, and in general, the output DNF ψ is not uniquely defined. In such cases, efficient computation is usually measured by the combined size of the input and the shortest permissible output. An algorithm is called *polynomial total time* [16] (or *output-polynomial*), if it runs in polynomial time in the input size and the shortest output size.

Unfortunately, an easy reduction from the classical satisfiability problem shows that there exists no polynomial total time algorithm for the dualization problem of general Boolean functions unless P=NP. Therefore, research has been focused on important restricted classes of Boolean functions, and in particular on positive (also called monotone) and Horn CNFs (e.g., [3, 7, 12, 16, 17]). Recall that a CNF is *positive* (resp., *Horn*) if each clause contains only positive literals (resp., at most one positive literal). It may happen that a CNF is neither positive

*This work was supported by the Austrian Science Fund (FWF) Project Z29-INF and by Grants-in-Aid for Scientific Research by the Ministry of Education, Science, Sports and Culture of Japan.

†Institut für Informationssysteme, Technische Universität Wien, Favoritenstraße 9-11, A-1040 Wien, Austria. (eiter@kr.tuwien.ac.at)

‡Department of Applied Mathematics and Physics, Graduate School of Informatics, Kyoto University, Kyoto 606-8501, Japan. (ibaraki@i.kyoto-u.ac.jp)

§Division of Systems Science, Graduate School of Engineering Science, Osaka University, Toyonaka, Osaka, 560-8531, Japan. (Email: makino@sys.es.osaka-u.ac.jp)

nor Horn, but becomes so after changing the polarity of some of the variables; such a CNF is called *unate* and *disguised* (or *hidden*) *Horn*, respectively. A respective change of polarities can be efficiently found [18, 1].

It is known [12] that there exists a quasi-polynomial total time (i.e., $N^{o(\log N)}$, where N denotes the combined size of the input and output) algorithm for dualization of positive CNFs, and many polynomial total time algorithm were constructed for special classes of positive functions including positive k-CNF, regular, threshold, read-once, acyclic functions (e.g., [4, 6, 7, 20, 19, 22]). However, it is still open whether the positive dualization problem has a polynomial total time algorithm or not. For Horn functions, our state of knowledge is less advanced; we have a quasi-polynomial total time algorithm for generating a DNF which contains all prime implicants of a Horn CNF [17], but it is not known whether the Horn dualization problem has a polynomial (or even only quasi-polynomial) total time algorithm. Note, however, that this problem is at least as hard as the positive dualization problem, since by changing the polarities of all literals in a positive CNF, we obtain a negative CNF, which is Horn.

In this paper, we consider the dualization problem for disguised bidual Horn functions. Here a function f is *bidual Horn* [9], if both f and its dual f^d can be described by Horn CNFs. The biduality constraint balances the roles of the sets $T(f)$ and $F(f)$ of all true and false vectors of f, respectively. From the logical perspective, the bidual Horn functions are those functions f such that $T(f)$ and $F(f)$ are described by using implications $x_{i_1} \wedge x_{i_2} \wedge \cdots \wedge x_{i_k} \rightarrow \ell$ and $\overline{x}_{i_1} \wedge \overline{x}_{i_2} \wedge \cdots \wedge \overline{x}_{i_k} \rightarrow \ell$, respectively, where $k \geq 0$ and ℓ is a literal [9]. Thus, if the false vectors describe illegal states, then they are fully characterized by dependencies of literals from false facts, and the legal states are also characterized by similar dependencies of literals from true facts. Note that bidual Horn functions can be seen as a natural generalization of negative functions, and enjoy similar properties (e.g., all irredundant CNFs have the same number of clauses). In particular, they include all functions defined by Horn CNFs in which only one variable occurs positively. The disguised bidual Horn functions are an intermediate class between the classes of unate and disguised Horn functions. It contains, as easily seen from results in [8], the double Horn functions (i.e., f and its complement $\overline{f}$ are Horn) as a subclass, but not the submodular functions (i.e., f and its contra-dual $\overline{f^d}(x) = f(\overline{x})$ are Horn) [10].

Let us call a CNF *disguised bidual Horn* if, after an appropriate renaming, it becomes a Horn CNF representing a bidual Horn function. As we show in this paper, it can be checked in polynomial time whether a given CNF is disguised bidual Horn. Combining this with the results for bidual Horn functions [9], we obtain a quasi-polynomial total time algorithm for dualizing disguised bidual Horn CNFs. This result enlarges the quasi-polynomially dualizable classes. As a further result, we show that, given a disguised bidual Horn CNF, an equivalent smallest CNF is computable in polynomial time. It is possible that a CNF representing a disguised bidual Horn function is not disguised bidual Horn in the above sense. However, we also show that in case where the input CNF φ is *prime* (i.e., no clause contains redundant literals), φ is a disguised bidual Horn CNF if and only if it represents a disguised bidual Horn function. Prime CNFs are important representations of Boolean functions and propositional knowledge bases (cf. [24, 13]). In such cases, our results yield quasi-polynomial total time and polynomial-time algorithms for dualization and minimization of disguised bidual Horn functions, respectively. Note that double Horn and submodular functions can be dualized in polynomial time and polynomial total time, respectively, and minimized in polynomial time [8, 10].

2 Disguised bidual Horn functions

A *clause* c is a disjunction $\bigvee_{i \in P(c)} x_i \vee \bigvee_{j \in N(c)} \overline{x}_j$ of literals on Boolean variables $x_1, \ldots, x_n$ such that $P(c) \cap N(c) = \emptyset$. We denote the variable indices in c by $V(c) = P(c) \cup N(c)$. A conjunctive normal form (CNF) φ is a *conjunction* $\bigwedge_{i=1}^{k} c_i$ of clauses; a disjunctive normal form (DNF) is dually defined as usual. A clause c is *Horn*, if $|P(c)| \leq 1$, and a CNF $\varphi = \bigwedge_i c_i$ is *Horn*, if all c_i are Horn. A Boolean function $f : \{0,1\}^n \to \{0,1\}$ is *Horn*, if it can be represented by some Horn CNF. It is well-known that this is equivalent to the condition that the set of true vectors $T(f) = \{v \in \{0,1\}^n \mid f(v) = 1\}$ is closed under componentwise conjunction. A clause c is an *implicate* of a formula φ (resp., function f), if $c \geq \varphi$ (resp., $c \geq f$) holds, and is *prime*, if no proper subclause c' of c is an implicate. Here, c and φ are regarded as the Boolean functions they represent, and $f_1 \geq f_2$ denotes $f_1(v) \geq f_2(v)$ holds for all vectors v. A CNF $\varphi = \bigwedge_i c_i$ is called *prime*, if all clauses c_i are prime implicates.

For example, $c_1 = (\overline{x}_1 \vee \overline{x}_4 \vee \overline{x}_5 \vee x_6)$ is a Horn clause ($P(c_2) = \{6\}$ and $N(c_2) = \{1,4,5\}$), while $c_2 = (\overline{x}_1 \vee \overline{x}_4 \vee x_5 \vee x_6)$ is not. The CNFs $\varphi^{(1)} = \overline{x}_2(\overline{x}_1 \vee \overline{x}_3)(\overline{x}_1 \vee \overline{x}_4)$ and $\varphi^{(2)} = x_2(\overline{x}_1 \vee x_2)(\overline{x}_3 \vee \overline{x}_4)$, respectively, are Horn and thus represent Horn functions. As easily checked, each clause of $\varphi^{(1)}$ is a prime implicate, and thus $\varphi^{(1)}$ is prime. However, $\varphi^{(2)}$ is not prime since $\overline{x}_1 \vee x_2$ is not a prime implicate.

A Boolean function f is called *bidual Horn* [9], if f and its dual f^d are Horn. In other words, the bidual Horn functions are those functions such that $T(f)$ and $F(f)$ are respectively closed under componentwise conjunction and disjunction, where $F(f) = \{0,1\}^n \setminus T(f)$. For example, $f = (x_1 \vee \overline{x}_2 \vee \overline{x}_3)(\overline{x}_1 \vee x_3 \vee \overline{x}_4)(\overline{x}_2 \vee \overline{x}_3 \vee \overline{x}_4)$ is bidual Horn, because by De Morgan's law, $f^d = x_1\overline{x}_2\overline{x}_3 \vee \overline{x}_1 x_3 \overline{x}_4 \vee \overline{x}_2\overline{x}_3\overline{x}_4 = (x_1 \vee \overline{x}_4)(\overline{x}_1 \vee \overline{x}_2)(\overline{x}_2 \vee x_3)(\overline{x}_2 \vee \overline{x}_4)(\overline{x}_1 \vee \overline{x}_3)(\overline{x}_3 \vee \overline{x}_4)$ is also Horn. In particular, it is easy to see that every *negative* function, which is a function represented by a *negative* CNF (i.e., a CNF without positive literals, such as $\varphi^{(1)}$ from above), is bidual Horn.

Bidual Horn functions were extensively studied in [9]. The following result is known, where for a pair of clauses c_i and c_j, we denote by $c^{\pm}_{i,j}$ the negative clause such that $N(c^{\pm}_{i,j}) = V(c_i) \cup V(c_j)$; e.g., if $c_1 = (\overline{x}_1 \vee x_2 \vee \overline{x}_3)$ and $c_2 = (x_1 \vee \overline{x}_4)$, then $c^{\pm}_{1,2} = (\overline{x}_1 \vee \overline{x}_2 \vee \overline{x}_3 \vee \overline{x}_4)$.

Lemma 2.1 ([9]) *Let φ be a Horn CNF. Then φ represents a bidual Horn function if and only if*

$$c^{\pm}_{i,j} \ \geq \ \varphi \tag{2.1}$$

holds for all pairs of Horn clauses c_i and c_j in φ such that $|P(c_i) \cup P(c_j)| = 2$.

We call any Horn CNF satisfying (2.1) *bidual Horn*. Clearly, any bidual Horn CNF represents a bidual Horn function, but other CNFs may also represent bidual Horn functions.

A *renaming* is a vector $r = (r_1, r_2, \ldots, r_n) \in \{0,1\}^n$. For any Boolean function f, its renaming by r, denoted f^r, is the function defined by $f^r(a) = 1 \iff f(a \oplus r) = 1$, where for any vector $a = (a_1, a_2, \ldots, a_n) \in \{0,1\}^n$, $a \oplus r$ denotes the vector $(b_1, b_2, \ldots, b_n)$ such that

$$b_i = \begin{cases} a_i, & \text{if } r_i = 0; \\ 1 - a_i, & \text{otherwise.} \end{cases}$$

Note that $a \oplus r$ denotes the componentwise *exclusive or*, and is the vector obtained from a by changing the polarity of variables i with $r_i = 1$. Notice that $f^{r \cdot r} = f$ holds for any f and r. The renaming of a formula φ by r, denoted φ^r, is the formula resulting from φ by replacing each literal involving a variable x_i with $r_i = 1$ by its opposite. A CNF φ is *disguised Horn* (resp., *disguised bidual Horn*), if there exists a renaming r such that φ^r is Horn (resp., bidual Horn). For example, let $r = (0,1,1,0)$, and consider $a = (1,1,0,0)$ and $\varphi = (x_1 \vee x_2)(x_1 \vee x_3)(x_2 \vee \overline{x}_4)$. Then, $a \oplus r = (1,0,1,0)$, and $\varphi^r = (x_1 \vee \overline{x}_2)(x_1 \vee \overline{x}_3)(\overline{x}_2 \vee \overline{x}_4)$ is Horn. Thus, φ is disguised Horn; moreover, by Lemma 2.1, we can immediately see that φ^r is disguised bidual Horn.

Denote, for any class of Boolean functions $\mathcal{C}$, the closure of $\mathcal{C}$ under renamings by $\mathcal{C}^R$, and let $\mathcal{C}_{un}$, $\mathcal{C}_H$, and $\mathcal{C}_{BH}$ denote the classes of unate (i.e., disguised negative), Horn, and bidual Horn functions, respectively. Then we immediately have the following relationships.

Proposition 2.1 $\mathcal{C}_{un} \subseteq \mathcal{C}_{BH}^{R} \subseteq \mathcal{C}_{H}^{R}$.

The inclusions are clearly strict, as shown by the following simple examples. The CNF $\varphi = (\overline{x}_1 \vee \overline{x}_2)(x_2 \vee x_3)$ represents a disguised bidual Horn function (apply $r = (011)$), but not a unate function. The CNF $\varphi = (\overline{x}_1 \vee x_2)(\overline{x}_2 \vee x_3)(\overline{x}_3 \vee x_1)$ represents a Horn (hence trivially disguised Horn) function, but not a disguised bidual Horn function, because its dual represented by $\psi = (x_1 \vee x_2 \vee x_3)(\overline{x}_1 \vee \overline{x}_2 \vee \overline{x}_3)$ is not disguised Horn.

3 Results

3.1 Recognizing disguised bidual Horn CNFs

We first consider the problem of recognizing disguised bidual Horn CNFs, i.e., the problem of deciding whether a given CNF φ can be renamed to a Horn CNF representing a bidual Horn function.

Our approach is to reduce the recognition problem to the problem of finding a renaming for an (already Horn) CNF to *another* Horn CNF, such that the biduality condition (2.1) is satisfied. This is achieved from a known polynomial time method for finding a Horn renaming by imposing some additional constraints, whose satisfiability can be decided in polynomial time.

Let $\varphi = \bigwedge_i c_i$ be a given CNF. We construct a 2SAT instance C, such that this instance is satisfiable if and only if φ is disguised Horn. Moreover, the satisfying assignments to C will give us all possible renamings r such that φ^r is bidual Horn. C consists of two groups of clauses C_1 and C_2 that serve for different purposes.

Let us introduce binary variables z_i, $i = 1, 2, \ldots, n$, such that $z_i = 1$ (resp., $z_i = 0$) means that x_i is renamed (resp., not renamed), i.e., $r_i = 1$ (resp., $r_i = 0$). The first group C_1 consists of the clauses expressing that a renaming of the variables must lead to a Horn expression. C_1 is the conjunction of the clauses generated by applying the following rules to all clauses $c = \bigvee_{i \in P(c)} x_i \vee \bigvee_{j \in N(c)} \overline{x}_j$ in φ:

$(1.i)$ $\quad z_i \to z_j$, for $i \in N(c), j \in P(c)$;

$(1.ii)$ $\quad z_i \to \overline{z}_j$, for $i, j \in N(c)$ and $i < j$; and

$(1.iii)$ $\quad \overline{z}_i \to z_j$, for $i, j \in P(c)$ and $i < j$,

where $y_i \to y_j$ stands for the clause $(\overline{y}_i \vee y_j)$ for literals y_i and y_j. The following lemma is well-known.

Lemma 3.1 (cf. [18]) *The CNF C_1 is satisfiable if and only if φ is disguised Horn. Moreover, there is a one-to-one correspondence between the satisfying assignments for C_1 and renamings r of φ such that φ^r is Horn.*

The second group C_2 consists of the clauses which prohibit renamings that lead to a violation of the biduality condition (2.1). For a CNF φ, let $r \in \{0,1\}^n$ be called a *Horn renaming* of φ if φ^r is Horn. If a Horn renaming r of φ has clauses c_1 and c_2 such that $c_3 = c_1^r$ and $c_4 = c_2^r$ satisfy $P(c_3) = \{i\}$, $P(c_4) = \{j\}$, $i \neq j$, and $c_{3,4}^{\pm} \not\geq \varphi^r$, then this r violates the biduality condition. Note that, since $c_{3,4}^{\pm} = c_{1,2}^{\pm}$,

$$\begin{aligned} c_{3,4}^{\pm} \not\geq \varphi^r \quad &\Longleftrightarrow \quad c_{1,2}^{\pm} \not\geq \varphi^r \\ &\Longleftrightarrow \quad \varphi^r \wedge \bigwedge_{i \in V(c_1) \cup V(c_2)} x_i \quad \text{is satisfiable} \\ &\Longleftrightarrow \quad \varphi \wedge \bigwedge_{i \in V(c_1) \cup V(c_2) : r_i = 0} x_i \;\wedge \bigwedge_{i \in V(c_1) \cup V(c_2) : r_i = 1} \overline{x}_i \quad \text{is satisfiable.} \end{aligned} \tag{3.2}$$

Hence, for each triple c_1, c_2, r of clauses c_1, c_2 in φ and Horn renaming r of $c_1 \wedge c_2$ such that $P(c_1^r) = \{i\}$, $P(c_2^r) = \{j\}$ and $i \neq j$, if $c_{3,4}^{\pm} \not\geq \varphi^r$ holds, then exactly one of the following clauses on the pair i and j is included in C_2 (in order to prohibit such a renaming r from the assignment to z):

$(2.i)$ $z_i \rightarrow z_j$, if $r_i = 1$ and $r_j = 0$;

$(2.ii)$ $z_i \rightarrow \overline{z}_j$, if $r_i = r_j = 1$; and

$(2.iii)$ $\overline{z}_i \rightarrow z_j$, if $r_i = r_j = 0$.

Note that if we first choose c_1, c_2, i and j in the above process, then the corresponding Horn renaming $r \in \{0,1\}^{V(c_1) \cup V(c_2)}$, if any exists, is uniquely determined. Moreover, given such c_1, c_2, i and j, (3.2) (and thus $c_{3,4}^{\pm} \not\geq \varphi^r$ without knowing the full renaming r) can be decided in polynomial time if φ is Horn.

Let $C = C_1 \wedge C_2$. (Note that C_1 and C_2 need not be disjoint.) Then we have the following result.

Lemma 3.2 *The CNF C is satisfiable if and only if φ is disguised bidual Horn. Moreover, there is a one-to-one correspondence between the satisfying assignments for C and all renamings r of φ such that φ^r is bidual Horn.*

Proof. Let us first show the only-if-part of the first statement. Let r be a satisfying assignment to the variables z_ℓ, $\ell = 1, 2, \ldots, n$, in C, Then, it is easily checked from the definition of C and Lemma 2.1 that φ^r is bidual Horn. To prove the if-part, assume that φ is disguised bidual Horn. Then there is a renaming r of f such that φ^r is bidual Horn. Clearly, φ^r must be Horn, and moreover satisfy condition (2.1). Then, r satisfies all the clauses in C. The one-to-one correspondence follows from the arguments in the proof of the first statement. □

As a result of Lemmas 3.1 and 3.2, the following algorithm solves the recognition problem for disguised bidual Horn CNFs (Step 1 is not needed for correctness, but ensures that testing (3.2) in Step 2 is efficient):

Algorithm CHECK-RBH
Input: A CNF φ on variables $x_1, \ldots, x_n$.
Output: A renaming r if there is an r such that φ^r is bidual Horn; otherwise, "No."

Step 1. Construct C_1 from φ and solve it;
  if no satisfying assignment is found for C_1 **then** output "No" and halt
  else let s be a satisfying assignment to C_1, and $\psi := \varphi^s$ **fi**.

Step 2. Construct again C_1 from ψ; $C_2 := \emptyset$;
  for each triple c_1, c_2, r of clauses c_1, c_2 in ψ and Horn renaming r of $c_1 \wedge c_2$
    such that $P(c_1^r) = \{i\}$, $P(c_2^r) = \{j\}$ and $i \neq j$ **do**
      if (3.2) holds **then** add to C_2 the binary clause on z_i and z_j according to (2.i)–(2.iv) **fi**
  end{for}.

Step 3. Solve $C = C_1 \wedge C_2$;
  if a satisfying assignment r is found **then**
    output r
  else output "No" **fi**. □

Example 3.1 Let us execute CHECK-RBH on $\varphi = (x_1 \vee x_2)(x_1 \vee x_3)(x_2 \vee \overline{x}_4)$, which is not Horn.

Step 1. C_1 is as follows:

$$C_1 = (\overline{z}_1 \rightarrow z_2)(\overline{z}_1 \rightarrow z_3)(z_4 \rightarrow z_2) \;=\; (z_1 \vee z_2)(z_1 \vee z_3)(\overline{z}_4 \vee z_2).$$

The first two clauses are included by (1.iii), and the third by $(1.i)$. The assignment $z = (1000)$ for example satisfies C_1. Hence, by the renaming $s = (1000)$ obtained in Step 1, we have a Horn CNF:

$$\psi := \varphi^s = (\overline{x}_1 \vee x_2)(\overline{x}_1 \vee x_3)(x_2 \vee \overline{x}_4).$$

Step 2. For this ψ, we construct

$$C_1 = (z_1 \to z_2)(z_1 \to z_3)(z_4 \to z_2) = (\overline{z}_1 \vee z_2)(\overline{z}_1 \vee z_3)(\overline{z}_4 \vee z_2).$$

The clauses c_1, c_2 in ψ and the corresponding Horn renamings r of $c_1 \wedge c_2$ such that $P(c_1^r) = \{i\}$, $P(c_2^r) = \{j\}$ and $i \neq j$ are shown below, as well as whether the condition for biduality, i.e., (3.2) holds or not for ψ.

	c_1, c_2 and r	$P(c_1^r) \cup P(c_2^r)$	(3.2) holds for ψ
(1)	$(\overline{z}_1 \vee z_2)$, $(\overline{z}_1 \vee z_3)$ and $(000) \in \{0,1\}^{\{1,2,3\}}$	$\{2,3\}$	Yes
(2)	$(\overline{z}_1 \vee z_2)$, $(\overline{z}_4 \vee z_2)$ and $(111) \in \{0,1\}^{\{1,2,4\}}$	$\{1,4\}$	Yes
(3)	$(\overline{z}_1 \vee z_3)$, $(\overline{z}_4 \vee z_2)$ and $(0000) \in \{0,1\}^{\{1,2,3,4\}}$	$\{2,3\}$	Yes
(4)	$(\overline{z}_1 \vee z_3)$, $(\overline{z}_4 \vee z_2)$ and $(1010) \in \{0,1\}^{\{1,2,3,4\}}$	$\{1,2\}$	Yes
(5)	$(\overline{z}_1 \vee z_3)$, $(\overline{z}_4 \vee z_2)$ and $(0101) \in \{0,1\}^{\{1,2,3,4\}}$	$\{3,4\}$	No
(6)	$(\overline{z}_1 \vee z_3)$, $(\overline{z}_4 \vee z_2)$ and $(1111) \in \{0,1\}^{\{1,2,3,4\}}$	$\{1,4\}$	Yes

Accordingly, we have

$$C_2 = (\overline{z}_2 \to z_3)(\, z_1 \to \overline{z}_4)(z_1 \to z_2) = (z_2 \vee z_3)(\overline{z}_1 \vee \overline{z}_4)(\overline{z}_1 \vee z_2).$$

The first clause is from (1) and (3); the second clause is obtained from (2) and (6); and the last clause from (4).

Step 3. Solving $C = C_1 \wedge C_2$ yields six satisfying assignments in the following table. The resulting bidual Horn functions are also shown.

r	ψ^r
(0100)	$(\overline{x}_1 \vee \overline{x}_2)(\overline{x}_1 \vee x_3)(\overline{x}_2 \vee \overline{x}_4)$
(0010)	$(\overline{x}_1 \vee x_2)(\overline{x}_1 \vee \overline{x}_3)(x_2 \vee \overline{x}_4)$
(0110)	$(\overline{x}_1 \vee \overline{x}_2)(\overline{x}_1 \vee \overline{x}_3)(\overline{x}_2 \vee \overline{x}_4)$
(0101)	$(\overline{x}_1 \vee \overline{x}_2)(\overline{x}_1 \vee x_3)(\overline{x}_2 \vee x_4)$
(0111)	$(\overline{x}_1 \vee \overline{x}_2)(\overline{x}_1 \vee \overline{x}_3)(\overline{x}_2 \vee x_4)$
(1110)	$(x_1 \vee \overline{x}_2)(x_1 \vee \overline{x}_3)(\overline{x}_2 \vee \overline{x}_4)$

Now we consider the complexity of algorithm CHECK-RBH. Denote, for any CNF φ, its size by $|\varphi|$, which is the number of symbols in it, where negative literals $\overline{x}_i$ are also counted as single symbols.

Theorem 3.1 *Algorithm* CHECK-RBH *correctly decides whether a given CNF φ is disguised bidual Horn in $O(n^2m^2|\varphi|)$ time, where n is the number of variables and m the number of clauses in φ.*

Proof. Correctness follows from Lemmas 3.1 and 3.2. Let us consider the time complexity. In Steps 1 and 2, constructing C_1 is clearly possible in time $O(n^2m)$ (since we have at most n^2 pairs i and j for each clause), and solving 2SAT for C_1 can be done in linear time, i.e., in $O(|C_1|) = O(n^2)$ time [11, 2]. In Step 2, constructing C_2 is possible in $O(n^2m^2|\varphi|)$ time; for each pair of clauses c_1, c_2, there are at most n^2 Horn renamings of $c_1 \wedge c_2$, since there are at most n^2 pairs of i and j satisfying $P(c_1^r) = \{i\}$, $P(c_2^r) = \{j\}$ and $i \neq j$. Moreover, since ψ is Horn, testing (3.2) for each triple c_1, c_2, r can be done in linear time, i.e., in $O(|\psi|)$ time using a proper data structure [21]. Overall, Step 2 requires $O(n^2m^2|\varphi|)$ time. Finally, Step 3 (solving 2SAT for $C = C_1 \wedge C_2$) is possible in linear time [11, 2], i.e., in $O(|C|) = O(n^2)$ time. Thus, the total running time of the algorithm is $O(n^2m^2|\varphi|)$. □

Remarks (1) Step 1 of algorithm CHECK-RBH (finding a Horn renaming of a CNF φ) can in fact be done in linear time, i.e., $O(|\varphi|)$ time. The set C_1 can be replaced by a set of $O(|\varphi|)$ many binary clauses C_1' (which involve auxiliary variables) such that the construction of C_1' can be done in $O(|\varphi|)$ time and the satisfying assignments of C_1' correspond to the Horn renamings (see e.g. [1]). Likewise, the construction of C_1 in Step 3 can be replaced by C_1'. However, the worst-case running time of the improved algorithm is still $O(n^2m^2|\varphi|)$.

(2) Since all satisfying assignments of a 2SAT-instance can be output with polynomial delay (cf. [23]), all renamings r that make φ bidual Horn can be output with polynomial delay in Step 3, once C has been constructed.

3.2 Recognizing disguised bidual Horn functions

Let us now turn from recognition of bidual Horn CNFs (i.e., the syntactic level) to recognition of bidual Horn *functions* (i.e., the semantic level). That is, given a CNF φ, decide whether it represents some $f \in \mathcal{C}_{BH}^R$. At this point, we emphasize that the above two concepts are different. Although any disguised bidual Horn CNF φ clearly represents an $f \in \mathcal{C}_{BH}^R$, it may happen that a CNF φ is not a disguised bidual Horn CNF, but still represents some $f \in \mathcal{C}_{BH}^R$. For example, consider $\varphi = x_1x_2\overline{x}_3(x_1 \vee \overline{x}_2 \vee \overline{x}_3)(\overline{x}_1 \vee x_2 \vee \overline{x}_3)(\overline{x}_1 \vee \overline{x}_2 \vee \overline{x}_3)$. This CNF is Horn, but not bidual Horn. Furthermore, there is no renaming r such that φ^r is bidual Horn, since φ^r is not Horn for any renaming r different from identity. However, φ represents the function $f = x_1x_2\overline{x}_3$, which is disguised bidual Horn (e.g., renaming both x_1 and x_2 makes f negative, and thus disguised bidual Horn). Informally, the last three clauses in φ, which are subsumed by the prime implicate $\overline{x}_3$, contain redundant literals x_1, $\overline{x}_2$, $\overline{x}_1$ and x_2 which prevent a suitable Horn renaming. Changing the polarity of any variable in φ violates its Horn property.

As it turns out, our algorithm can be readily applied to recognize disguised bidual Horn functions if the input CNF has no such redundancies, i.e., it is prime. We note the following lemmas.

Lemma 3.3 *Let f and g be functions and r be a renaming. Then $f \geq g$ holds if and only if $f^r \geq g^r$ holds.*

Proof. Indeed, if $f \geq g$, then for every v, $g^r(v) = g(v^r) = 1$ implies that $f(v^r) = f^r(v) = 1$, and hence $f^r \geq g^r$. The converse is similar. □

Lemma 3.4 *Let φ be a prime CNF that defines a function f. Then, for any renaming r, φ^r is a prime CNF for f^r. In particular, the CNF φ^r is Horn if and only if the function f^r is Horn.*

Proof. If φ is prime, then φ^r must consist of prime implicates of f^r. Otherwise, by using Lemma 3.3, a contradiction to primality of φ is easily derived. Next, since a function is Horn if and only if all prime implicates are Horn [14], it follows that φ^r is Horn if and only if f^r is Horn. □

Thus, combined with Theorem 3.1, we obtain the following result.

Theorem 3.2 *Algorithm* CHECK-RBH *correctly decides whether a given prime CNF φ represents a function $f \in \mathcal{C}_{BH}^R$ in $O(n^2m^2|\varphi|)$ time, where n is the number of variables and m the number of clauses in φ, and outputs a renaming r such that $f^r \in \mathcal{C}_{BH}$ if this is the case.*

By virtue of this result, we can recognize disguised bidual Horn functions also from non-prime CNFs φ in polynomial time provided that an equivalent prime CNF is computable in polynomial time. In particular, this is the case if the CNF φ is disguised Horn.

Corollary 3.1 *Deciding whether a given disguised Horn CNF φ represents an $f \in \mathcal{C}_{BH}^{R}$ is possible in $O(n^2m^2|\varphi|)$ time, where n is the number of variables and m the number of clauses in φ.*

Proof. First, find a Horn renaming r of φ, which is computable in $O(|\varphi|)$ time (e.g., [1]). Then make φ^r prime, which can be easily done in $O(|\varphi|^2)$ time (cf. [14]). Finally, apply algorithm CHECK-RBH, which takes $O(n^2m^2|\varphi|)$ time (note that Step 1 can be omitted, since φ is already Horn). Since $|\varphi| \leq nm$, it follows that the overall running time of the algorithm is $O(n^2m^2|\varphi|)$. □

Recall that the example given in the beginning of this subsection shows that making φ prime is in fact needed for this result. Observe that, unsurprisingly, the recognition problem $f \in \mathcal{C}_{BH}^{R}$ from an arbitrary CNF is intractable. This is easily obtained from the following general result.

Theorem 3.3 ([15]) *The recognition problem from a CNF is* co-NP-*hard for any class $\mathcal{C}$ of functions which contains $f = 1$ for each arity, does not contain all functions, and is closed under projections (i.e., $f \in \mathcal{C}$ implies that the functions $f_{x_i \leftarrow 1}$ and $f_{x_i \leftarrow 0}$ on $n-1$ variables obtained by fixing the value of any variable x_i are also in $\mathcal{C}$).*

Clearly, $\mathcal{C}_{BH}^{R}$ is closed under projections, establishing the next

Corollary 3.2 *Deciding whether a given CNF φ represents an $f \in \mathcal{C}_{BH}^{R}$ is* co-NP-*hard.*

On the other hand, by exploiting Theorem 3.2, we obtain that the complexity of this problem does not drastically exceed $\mathrm{co\text{-}NP}$. Indeed, we can make a given CNF φ prime in polynomial time with the help of an NP oracle, by iteratively removing redundant literals from the clauses in φ; note that deciding whether a particular literal can not be removed from a clause of φ is in NP. After that, we may apply CHECK-RBH on the resulting prime CNF ψ. Thus, the problem in Corollary 3.2 is in the complexity class P^{NP}. Making φ prime seems unlikely to be P^{NP}-hard, since different from typical P^{NP}-hard problems such as the Traveling Salesman Problem, it appears that the oracle calls do not have to follow a particular strict order. Therefore, we conjecture that recognizing disguised bidual Horn functions from arbitrary CNFs is not complete for P^{NP}. The precise complexity is open.

3.3 Dualization and minimization

Now we consider the dualization and minimization of a disguised bidual Horn function. As discussed in introduction, it is known [9] that, given a bidual Horn CNF, the dualization and computing a smallest CNF representation can be done in quasi-polynomial total time and polynomial time, respectively. Therefore, the above results imply the following nice theorem.

Theorem 3.4 *If the input CNF φ represents an $f \in \mathcal{C}_{BH}^{R}$ and a prime CNF equivalent to φ is polynomially computable, then:*

(i) The dualization problem can be solved in quasi-polynomial total time.

(ii) A smallest CNF representation for f can computed in polynomial time.

4 Conclusion

In this paper, we have pushed the frontier of the dualization problem, which is solvable in quasi-polynomial total time, from positive CNFs to disguised bidual Horn CNFs. However, several problems still remain for further work. The most interesting ones, as we feel, are dualization of Horn CNFs and of prime CNFs (i.e., computing a shortest DNF from a given Horn CNF or prime CNF, respectively). The former problem reduces to the latter in polynomial time, since making a Horn CNF prime is easily accomplished in quadratic time. While dualizing a Horn CNF or a prime CNF is at least as hard as dualizing a positive CNF, it is open whether these problems are harder, and in particular, whether solvability by a quasi-polynomial total time algorithm would imply P=NP.

Acknowledgment. We thank the reviewers for their constructive comments which improved this paper.

References

[1] B. Aspvall. Recognizing disguised NR(1) instances of the satisfiability problem. *J. Algorithms*, 1:97–103, 1980.

[2] B. Aspvall, M. Plass, and R. Tarjan. A linear time algorithm for testing the truth of certain quantified Boolean formulas. *Information Processing Letters*, 8:121–123, 1979.

[3] C. Bioch and T. Ibaraki. Complexity of identification and dualization of positive Boolean functions. *Information and Computation*, 123:50–63, 1995.

[4] E. Boros, V. Gurvich, and P.L. Hammer. Dual subimplicants of positive Boolean functions. *Optimization Methods and Software*, 10:147–156, 1998.

[5] M. Cadoli and F. Donini. A Survey on Knowledge Compilation. *AI Communications 10*, 10(3-4):137–150, 1997.

[6] Y. Crama. Dualization of regular Boolean functions. *Discrete Applied Mathematics*, 16:79–85, 1987.

[7] T. Eiter and G. Gottlob. Identifying the minimal transversals of a hypergraph and related problems. *SIAM Journal on Computing*, 24(6):1278–1304, December 1995.

[8] T. Eiter, T. Ibaraki, and K. Makino. Double Horn functions. *Information and Computation*, 144:155–190, 1998.

[9] T. Eiter, T. Ibaraki, and K. Makino. Bidual Horn functions and extensions. *Discrete Applied Mathematics*, 96-97:55–88, 1999. Selected for the special volume *Discrete Applied Mathematics, Editors' Choice, Edition 1999*.

[10] O. Ekin, P.L. Hammer and U.N. Peled. Horn functions and submodular Boolean functions. *Theoretical Computer Science*, 175:257–270, 1997.

[11] S. Even, A. Itai, and A. Shamir. On the complexity of timetable and multicommodity flow problems. *SIAM Journal of Computing*, 5:691–703, 1976.

[12] M. Fredman and L. Khachiyan. On the complexity of dualization of monotone disjunctive normal forms. *J. Algorithms*, 21:618–628, 1996.

[13] G. Gottlob and C. Fermüller. Removing redundancy from a clause. *Artificial Intelligence*, 61:263–289, 1993.

[14] P.L. Hammer and A. Kogan. Horn functions and their DNFs. *Information Processing Letters*, 44:23–29, 1992.

[15] T. Hegedűs and N. Meggido. On the geometric separability of Boolean functions. *Discrete App. Math.*, 61:1–25, 1995.

[16] D. S. Johnson, M. Yannakakis, and C. H. Papadimitriou. On generating all maximal independent sets. *Information Processing Letters*, 27:119–123, 1988.

[17] R. Khardon. Translating between Horn representations and their characteristic models. *Journal of Artificial Intelligence Research*, 3:349–372, 1995.

[18] H. Lewis. Renaming a set of clauses as a Horn set. *Journal of the ACM*, 25:134–135, 1978.

[19] K. Makino. Efficient dualization of $O(\log n)$-term monotone disjunctive normal forms. Technical Report 00-07, Discrete Mathematics and Systems Science, Osaka University, 2000.

[20] K. Makino and T. Ibaraki. The maximum latency and identification of positive Boolean functions. *SIAM Journal on Computing*, 26:1363–1383, 1997.

[21] M. Minoux. LTUR: a simplified linear time unit resolution for Horn formulae and computer implementation. *Information Processing Letters*, 29:1–12, 1988.

[22] U. Peled and B. Simeone. An $O(nm)$ time algorithm for computing the dual of a regular function. *Discrete Applied Mathematics*, 16:79–85, 1994.

[23] L. G. Valiant. The complexity of enumeration and reliability problems. *SIAM Journal on Computing*, 8:410–421, 1979.

[24] I. Wegener. *The complexity of Boolean functions*. Wiley & Sons Ltd and Teubner, Stuttgart, 1987.