Isomorphic Implication*

Michael Bauland[1] and Edith Hemaspaandra[2]

[1] Theoretische Informatik, Universität Hannover, Appelstr. 4, D-30167 Hannover, Germany, email: bauland@thi.uni-hannover.de.
Work done in part while visiting the Laboratory for Applied Computing at Rochester Institute of Technology.
[2] Department of Computer Science, Rochester Institute of Technology, Rochester, NY 14623, U.S.A., e-mail: eh@cs.rit.edu.
Work done in part while on sabbatical at the University of Rochester.

Abstract. We study the isomorphic implication problem for Boolean constraints. We show that this is a natural analog of the subgraph isomorphism problem. We prove that, depending on the set of constraints, this problem is in P, NP-complete, or NP-hard, coNP-hard, and in $\mathrm{P}^{\mathrm{NP}}_{||}$. We show how to extend the NP-hardness and coNP-hardness to $\mathrm{P}^{\mathrm{NP}}_{||}$-hardness for some cases, and conjecture that this can be done in all cases.

1 Introduction

One of the most interesting and well-studied problems in complexity theory is the graph isomorphism problem (GI). This is the problem of determining whether two graphs are isomorphic, i.e., whether there exists a renaming of vertices such that the graphs become equal. This is a fascinating problem, since it is the most natural example of a problem that is in NP, not known to be in P, and unlikely to be NP-complete (see [KST93]).

The obvious analog of graph isomorphism for Boolean formulas is the formula isomorphism problem. This is the problem of determining whether two formulas are isomorphic, i.e., whether we can rename the variables such that the formulas become equivalent. This problem has the same behavior as the graph isomorphism problem one level higher in the polynomial hierarchy: The formula isomorphism problem is in Σ_2^p, NP-hard, and unlikely to be Σ_2^p-complete [AT00].

Note that graph isomorphism can be viewed as a special case of Boolean isomorphism, since graph isomorphism corresponds to Boolean isomorphism for 2-positive-CNF formulas, in the following way: Every graph G (without isolated vertices) corresponds to the (unique) formula $\bigwedge_{\{i,j\}\in E(G)} x_i \vee x_j$. Then two graphs without isolated vertices are isomorphic if and only if their corresponding formulas are isomorphic.

One might wonder what happens when we look at other restrictions on the set of formulas. There are general frameworks for looking at all restrictions on

* Supported in part by grants NSF-CCR-0311021 and DFG VO 630/5-1.

Boolean formulas: The most often used is the Boolean constraint framework introduced by Schaefer [Sch78]. Basically (formal definitions can be found in the next section) we look at formulas as CNF formulas (or sets of clauses) where each clause is an application of a constraint (a k-ary Boolean function) to a list of variables. Each finite set of constraints gives rise to a new language, and so there are an infinite number of languages to consider. Schaefer studied the satisfiability problem for all finite sets of constraints. He showed that all of these satisfiability problems are either in P or NP-complete, and he gave a simple criterion to determine which of the cases holds.

The last decade has seen renewed interest in Schaefer's result, and has seen many dichotomy (and dichotomy-like) theorems for problems related to the satisfiability of Boolean constraints. For example, such results were obtained for the maximum satisfiability problem [Cre95], counting satisfying assignments [CH96], the inverse satisfiability problem [KS98], the unique satisfiability problem [Jub99], the minimal satisfying assignment problem [KK01], approximability problems [KSTW01], and the equivalence problem [BHRV02]. For an excellent survey of dichotomy theorems for Boolean constraint satisfaction problems, see [CKS01].

Most of the results listed above were proved using methods similar to the one used by Schaefer [Sch78]. A more recent approach to proving results of this form is with the help of the so-called algebraic approach [Jea98,JCG97,BKJ00]. This approach uses the clone (closed classes) structure of Boolean functions called Post's lattice, after Emil Post, who first identified these classes [Pos44]. A good introduction of how this can be used to obtain short proofs can be found in [BCRV04]. However, this approach does not work for isomorphism problems, because it uses existential quantification.

For the case of most interest for this paper, the Boolean isomorphism problem for constraints, Böhler et al. [BHRV02,BHRV04,BHRV03] have shown that this problem is in P, GI-complete, or GI-hard, coNP-hard, and in $\mathrm{P}_{||}^{\mathrm{NP}}$ (the class of problems solvable in polynomial time with one round of parallel queries to NP). As in Schaefer's theorem, simple properties of the set of constraints determine the complexity.

A problem closely related to the graph isomorphism problem is the subgraph isomorphism problem. This is the problem, given two graphs G and H, to determine whether G contains a subgraph isomorphic to H. In contrast to the graph isomorphism problem, the subgraph isomorphism problem can easily be seen to be NP-complete (it contains, for example, CLIQUE, HAMILTONIAN CYCLE, and HAMILTONIAN PATH).

To further study the relationship between the isomorphism problems for graphs and constraints, we would like to find a relation $\mathcal{R}$ on constraints that is to isomorphism for constraints as the subgraph isomorphism problem is to graph isomorphism.

Such a relation $\mathcal{R}$ should at least have the following properties:

1. A graph G is isomorphic to a graph H if and only if G contains a subgraph isomorphic to H and H contains a subgraph isomorphic to G. We want the

same property in the constraint case, i.e., for S and U sets of constraint applications, S is isomorphic to U if and only if $S\mathcal{R}U$ and $U\mathcal{R}S$.

2. The subgraph isomorphism problem should be a special case of the decision problem induced by $\mathcal{R}$, in the same way as the graph isomorphism problem is a special case of the constraint isomorphism problem. In particular, for G and H graphs, let $S(G)$ and $S(H)$ be their (standard) translations into sets of constraint applications of $\lambda xy.(x \vee y)$, i.e., $S(G) = \{x_i \vee x_j \mid \{x_i, x_j\} \in E(G)\}$ and $S(H) = \{x_i \vee x_j \mid \{x_i, x_j\} \in E(H)\}$. For G and H graphs without isolated vertices, G is isomorphic to H if and only if $S(G)$ is isomorphic to $S(H)$. We want G to have a subgraph isomorphic to H if and only if $S(G)\mathcal{R}S(H)$.

Borchert et al. [BRS98, p. 692] suggest using the subfunction relations $\gg_v$ and $\gg_{cv}$ as analogs of subgraph isomorphism. These relations are defined as follows. For two formulas ϕ and ψ, $\phi \gg_v \psi$ if and only if there exists a function π from variables to variables such that $\pi(\phi)$ is equivalent to ψ. $\phi \gg_{cv} \psi$ if and only if there exists a function π from variables to variables and constants such that $\pi(\phi)$ is equivalent to ψ [BR93]. Borchert and Ranjan [BR93] show that these relations satisfy our first desirable property, i.e., S is isomorphic to U if and only if $S \gg_v U$ and $U \gg_v S$, and that S is isomorphic to U if and only if $S \gg_{cv} U$ and $U \gg_{cv} S$. They also show that the problem of determining whether $\phi \gg_v \psi$ and the problem of determining whether $\phi \gg_{cv} \psi$, for unrestricted Boolean formulas, are Σ_2^p-complete.

But Borchert et al.'s subfunction relations will not give the second desirable property. Consider, for example, the graphs G and H such that $V(G) = V(H) = \{1, 2, 3\}$, $E(G) = \{\{1, 2\}, \{1, 3\}, \{2, 3\}\}$, and $E(H) = \{\{1, 2\}, \{1, 3\}\}$. Clearly, G contains a subgraph isomorphic to H, but $(x_1 \vee x_2) \wedge (x_1 \vee x_3) \wedge (x_2 \vee x_3) \not\gg_{cv} (x_1 \vee x_2) \wedge (x_1 \vee x_3)$.

How could the concept of a subgraph be translated to sets of constraint applications? As a first attempt at translating subgraph isomorphism to constraint isomorphism one might try the following: For sets of constraint applications S and U, does there exist a subset $\widehat{S}$ of S that is isomorphic to U. Certainly, such a definition satisfies the second desired property. But this definition does not satisfy the first desired property, since it is quite possible for sets of constraint applications to be equivalent without being equal.

We claim that isomorphic implication satisfies both desired properties, and is a natural analog of the subgraph isomorphism problem for Boolean constraints.

For S and U sets of constraint applications over variables X, we say that S isomorphically implies U (notation: $S \cong\!\!\!\!> U$) if and only if there exists a permutation π on X such that $\pi(S) \Rightarrow U$. In Section 4, we show that, depending on the set of constraints, the isomorphic implication problem is in P, NP-complete, or NP-hard, coNP-hard, and in $\mathrm{P}_{||}^{\mathrm{NP}}$. Our belief is that the isomorphic implication problem is $\mathrm{P}_{||}^{\mathrm{NP}}$-complete for all the cases where it is both NP-hard and coNP-hard. In Section 5, we prove this conjecture for some cases.

2 Preliminaries

We will mostly use the constraint terminology from [CKS01].

Definition 1. *1. A* constraint *C (of arity k) is a Boolean function from $\{0,1\}^k$ to $\{0,1\}$.*

2. If C is a constraint of arity k, and $z_1, z_2, \ldots, z_k$ are (not necessarily distinct) variables, then $C(z_1, z_2, \ldots, z_k)$ is a constraint application *of C.*

3. If C is a constraint of arity k, and for $1 \leq i \leq k$, z_i is a variable or a constant (0 or 1), then $C(z_1, z_2, \ldots, z_k)$ is a constraint application of C with constants*.*

4. If S is a set of constraint applications [with constants] and X is a set of variables that includes all variables that occur in S, we say that S is a set of constraint applications [with constants] over variables *X.*

Definition 2. *Let C be a k-ary constraint.*

- *C is* 0-valid *if $C(0, \ldots, 0) = 1$.*
- *C is* 1-valid *if $C(1, \ldots, 1) = 1$.*
- *C is* Horn *(or weakly negative) if $C(x_1, \ldots, x_k)$ is equivalent to a CNF formula where each clause has at most one positive literal.*
- *C is* anti-Horn *(or weakly positive) if $C(x_1, \ldots, x_k)$ is equivalent to a CNF formula where each clause has at most one negative literal.*
- *C is* bijunctive *if $C(x_1, \ldots, x_k)$ is equivalent to a 2CNF formula.*
- *C is* affine *if $C(x_1, \ldots, x_k)$ is equivalent to an XOR-CNF formula.*
- *C is* 2-affine *(or affine of width 2) if $C(x_1, \ldots, x_k)$ is equivalent to an XOR-CNF formula, such that every clause contains at most two literals.*
- *C is* complementive *(or C-closed) if for every $s \in \{0,1\}^k$, $C(s) = C(\overline{s})$, where $\overline{s} \in \{0,1\}^k =_{def} (1, \ldots, 1) - s$, i.e., $\overline{s}$ is obtained by flipping every bit of s.*

Let $\mathcal{C}$ be a finite set of constraints. We say $\mathcal{C}$ is 0-valid, 1-valid, Horn, anti-Horn, bijunctive, affine, 2-affine, or complementive, if *every* constraint $C \in \mathcal{C}$ has this respective property. We say that $\mathcal{C}$ is *Schaefer* if $\mathcal{C}$ is Horn, anti-Horn, affine, or bijunctive.

Definition 3 ([BHRV02]). *Let $\mathcal{C}$ be a finite set of constraints.*

1. *$\mathrm{ISO}(\mathcal{C})$ is the problem, given two sets S and U of constraint applications of $\mathcal{C}$ over variables X, to decide whether S is isomorphic to U (denoted by $S \cong U$), i.e., whether there exists a permutation π of X such that $\pi(S) \equiv U$; Here $\pi(S)$ is the set of constraint applications that results when we simultaneously replace every variable x in S by $\pi(x)$.*
2. *$\mathrm{ISO}_c(\mathcal{C})$ is the problem, given two sets S and U of constraint applications of $\mathcal{C}$ with constants, to decide whether S is isomorphic to U.*

Theorem 4 ([BHRV02,BHRV04,BHRV03]). *Let $\mathcal{C}$ be a finite set of constraints.*

1. *If $\mathcal{C}$ is not Schaefer, then* $\mathrm{ISO}(\mathcal{C})$ *and* $\mathrm{ISO}_c(\mathcal{C})$ *are* coNP*-hard,* GI*-hard, and in* $\mathrm{P}^{\mathrm{NP}}_{||}$.
2. *If $\mathcal{C}$ is Schaefer and not 2-affine, then* $\mathrm{ISO}(\mathcal{C})$ *and* $\mathrm{ISO}_c(\mathcal{C})$ *are polynomial-time many-one equivalent to* GI.
3. *Otherwise, $\mathcal{C}$ is 2-affine and* $\mathrm{ISO}(\mathcal{C})$ *and* $\mathrm{ISO}_c(\mathcal{C})$ *are in* P.

The isomorphic implication problem combines isomorphism with implication in the following way.

Definition 5. *Let $\mathcal{C}$ be a finite set of constraints.*

1. $\mathrm{ISO\text{-}IMP}(\mathcal{C})$ *is the problem, given two sets S and U of constraint applications of $\mathcal{C}$ over variables X, to decide whether S isomorphically implies U (denoted by $S \cong\!\!\Rightarrow U$), i.e., whether there exists a permutation π of X such that $\pi(S) \Rightarrow U$; Here $\pi(S)$ is the set of constraint applications that results when we simultaneously replace every variable x in S by $\pi(x)$.*
2. $\mathrm{ISO\text{-}IMP}_c(\mathcal{C})$ *is the problem, given two sets S and U of constraint applications of $\mathcal{C}$ with constants, deciding whether S isomorphically implies U.*

To show that this definition is well defined we need to show that if S and U are sets of constraint applications over variables X, Y is a set of variables disjoint from X, and there exists a permutation π of $X \cup Y$ such that $\pi(S) \Rightarrow U$, then there exists a permutation ρ of X such that $\rho(S) \Rightarrow U$.

Suppose that π is a permutation of $X \cup Y$ such that $\pi(S) \Rightarrow U$ and $||\{y \in Y \mid \pi(y) \in X\}||$ is minimal and at least one. Let $x, x' \in X$ and $y, y' \in Y$ be such that $\pi(x') = y$ and $\pi(y') = x$. Define a new permutation ρ as follows: $\rho(x') = x$, $\rho(y') = y$, and $\rho(z) = \pi(z)$ for all $z \in (X \cup Y) - \{x', y'\}$. We will show that $\rho(S) \Rightarrow U$. This is a contradiction, since $||\{y \in Y \mid \rho(y) \in X\}|| = ||\{y \in Y \mid \pi(y) \in X\}|| - 1$.

Let Z be a list of the variables in $(X \cup Y) - \{x, y\}$. Suppose that $\rho(S)(Z, x, y) \not\Rightarrow U(Z, x, y)$. Then there exists a string $s \in \{0,1\}^{||Z||}$ and $a, b \in \{0, 1\}$ such that $\rho(S)(s, a, b) = 1$ and $U(s, a, b) = 0$. Since y does not occur in U, $U(s, a, \overline{b}) = 0$. Since $\pi(S)(Z, x, y) \Rightarrow U(Z, x, y)$, it follows that $\pi(S)(s, a, b) = 0$ and $\pi(S)(s, a, \overline{b}) = 0$. Since y' does not occur in S, x does not occur in $\pi(S)$, and so $\pi(S)(s, \overline{a}, b) = 0$ and $\pi(S)(s, \overline{a}, \overline{b}) = 0$. It follows that $\pi(S)(s, x, y) \equiv 0$. But note that $\pi(S)(s, b, a) = \rho(S)(s, a, b) = 1$. This is a contradiction.

Definition 6. *1. The* graph isomorphism problem *is the problem, given two graphs G and H, to decide whether G and H are isomorphic, i.e., whether there exists a bijection π from $V(G)$ to $V(H)$ such that $\pi(G) = H$. $\pi(G)$ is the graph such that $V(\pi(G)) = \{\pi(v) \mid v \in V(G)\}$ and $E(\pi(G)) = \{\{\pi(v), \pi(w)\} \mid \{v, w\} \in E(G)\}$.*

2. *The subgraph isomorphism problem is the problem, given two graphs G and H, to decide whether G contains a subgraph isomorphic to H, i.e., whether there exists a graph G' such that $V(G') \subseteq V(G)$ and $E(G') \subseteq E(G)$ and G' is isomorphic to H.*

Theorem 7 ([GJ79,Coo71]). *The subgraph isomorphism problem is* NP*-complete.*

Corollary 8. *The subgraph isomorphism problem for graphs without isolated vertices is* NP*-complete.*

3 Subgraph Isomorphism and Isomorphic Implication

We will now show that the isomorphic implication problem is a natural analog of the subgraph isomorphism problem, in the sense explained in the introduction.

Lemma 9. *1. Let S and U be sets of constraint applications of $\mathcal{C}$ with constants. Then $S \cong U$ if and only if $S \overset{\cong}{\Rightarrow} U$ and $U \overset{\cong}{\Rightarrow} S$.*

2. For graphs G and H without isolated vertices, G contains a subgraph isomorphic to H if and only if $S(G) \overset{\cong}{\Rightarrow} S(H)$, where S is the "standard" translation from graphs to sets of constraint applications of $\lambda xy.x \vee y$, i.e., for $\widehat{G}$ a graph, $S(\widehat{G}) = \{x_i \vee x_j \mid \{i, j\} \in E(\widehat{G})\}$.

Proof.

1. We claim that $S \cong U$ if and only if $S \overset{\cong}{\Rightarrow} U$ and $U \overset{\cong}{\Rightarrow} S$. The left-to-right direction is immediate. For the converse, let X be the set of variables that occur in $S \cup U$. Suppose that π is a permutation of the variables occurring in $S \cup U$ such that $\pi(S) \Rightarrow U$ and that ρ is a permutation of X such that $\rho(U) \Rightarrow S$. Suppose for a contradiction that $\pi(S) \not\equiv U$. Then there exists an assignment that satisfies U, and that does not satisfy $\pi(S)$. Since $\rho(U) \Rightarrow S$, there are at least as many satisfying assignments for $\pi(S)$ as for U. It follows that there exists an assignment that satisfies $\pi(S)$ and not U. But that contradicts the assumption that $\pi(S) \Rightarrow U$.
2. Let G and H be graphs without isolated vertices. We will show that G contains a subgraph isomorphic to H if and only if $S(G) \overset{\cong}{\Rightarrow} S(H)$.
   For the left-to-right direction, let G' be a subgraph of G such that $G' \cong H$. Let π be a bijection from the vertices of G' to the vertices of H such that $\pi(G') = H$. Let ρ be a permutation of the variables occurring in $S(G) \cup S(H)$ such that $\rho(x_i) = x_{\pi(i)}$ for all $i \in V(G')$. It is easy to see that $\rho(S(G')) = S(\pi(G')) = S(H)$. Since G' is a subgraph of G, $S(G') \subseteq S(G)$. It follows that $S(H) \subseteq \rho(S(G))$, and thus, $\rho(S(G)) \Rightarrow S(H)$.
   For the converse, suppose that $S(G) \overset{\cong}{\Rightarrow} S(H)$. Let ρ be a permutation on the variables occurring in $S(G) \cup S(H)$ such that $\rho(S(G)) \Rightarrow S(H)$. It is easy to see that if $\rho(S(G)) \Rightarrow x_i \vee x_j$, then $x_i \vee x_j \in \rho(S(G))$. It follows that $S(H) \subseteq \rho(S(G))$. Let G' be such that $S(H) = \rho(S(G'))$ and G' does not have isolated vertices (take $G' = \pi^{-1}(H)$). Since $\rho(S(G')) \subseteq \rho(S(G))$, it follows that $S(G') \subseteq S(G)$, and thus, G' is a subgraph of G. Since $S(H) = \rho(S(G'))$ and H and G' do not contain isolated vertices, it follows that G' is isomorphic to H.

□

4 Complexity of the Isomorphic Implication Problem

The following theorem gives a trichotomy-like theorem for the isomorphic implication problem.

Theorem 10. *Let $\mathcal{C}$ be a finite set of constraints.*

1. *If every constraint in $\mathcal{C}$ is equivalent to a constant or a conjunction of literals, then* ISO-IMP($\mathcal{C}$) *and* ISO-IMP$_c$($\mathcal{C}$) *are in* P.
2. *Otherwise, if $\mathcal{C}$ is Schaefer, then* ISO-IMP($\mathcal{C}$) *and* ISO-IMP$_c$($\mathcal{C}$) *are* NP-*complete.*
3. *If $\mathcal{C}$ is not Schaefer, then* ISO-IMP($\mathcal{C}$) *and* ISO-IMP$_c$($\mathcal{C}$) *are* NP-*hard,* coNP-*hard, and in* $\mathrm{P}^{\mathrm{NP}}_{||}$.

4.1 Upper bounds

The NP upper bound for sets of constraints that are Schaefer is easy to see.

Claim 11 *If $\mathcal{C}$ is Schaefer, then* ISO-IMP$_c$($\mathcal{C}$) *is in* NP.

Proof. Let S and U be sets of constraint applications of $\mathcal{C}$ with constants over variables X. Then $S \overset{\sim}{\Rightarrow} U$ if and only if there exists a permutation π of X such that $\pi(S) \Rightarrow U$. Clearly, $\pi(S) \Rightarrow U$ if and only if $\pi(S) \cup U \equiv \pi(S)$. Since $\mathcal{C}$ is Schaefer, it can be determined in polynomial time whether two sets of constraint applications of $\mathcal{C}$ with constants are equivalent [BHRV02, Theorem 6]. □

Claim 12 *For any finite set $\mathcal{C}$ of constraints,* ISO-IMP$_c$($\mathcal{C}$) *is in* $\mathrm{P}^{\mathrm{NP}}_{||}$.

Proof. (Similar to the argument before [BHRV02, Corollary 23].) Let S and U be sets of constraint applications of $\mathcal{C}$ with constants. Let X be the set of all variables that occur in $S \cup U$. From [BHRV02, proof of Claim 22], we know that we can in polynomial time with parallel access to NP compute the set of all constraint applications of $\mathcal{C}$ with constants over X that are implied by S. Call this set $\widehat{S}$. It is easy to see that $\pi(S) \Rightarrow U$ if and only if $U \subseteq \widehat{S}$. It takes one query to NP to find out whether there exists such a permutation. Since two rounds of queries to NP are the same as one round of queries to NP [BH91] it follows that we can determine whether $S \overset{\sim}{\Rightarrow} U$ in $\mathrm{P}^{\mathrm{NP}}_{||}$. □

Claim 13 *Let $\mathcal{C}$ be a finite set of constraints such that every constraint is equivalent to a constant or to a conjunction of literals. Then* ISO-IMP$_c$($\mathcal{C}$) *is in* P.

Proof. Let S and U be sets of constraint applications of $\mathcal{C}$ with constants. We will view S and U as sets of literals and constants. We first consider the case where S or U is equivalent to a constant. Note that it is easy to check if a set X of literals and constants is equivalent to 0 or 1, since X is equivalent to 1 if and only if $X = \{1\}$, and X is equivalent to 0 if and only if $0 \in X$ or $\{p, \overline{p}\} \subseteq X$ for some variable p. It is easy to see that if S or U is equivalent to a constant, then determining whether $S \overset{\sim}{\Rightarrow} U$ takes polynomial time, since

- If $S \equiv 1$, then $S \widetilde{\Rightarrow} U$ iff $U \equiv 1$.
- If $S \equiv 0$ or $U \equiv 1$, then $S \widetilde{\Rightarrow} U$.
- If $U \equiv 0$, then $S \widetilde{\Rightarrow} U$ iff $S \equiv 0$.

It remains to consider the case that neither S nor U is equivalent to a constant. We claim that in this case, $S \widetilde{\Rightarrow} U$ iff the number of positive literals in S is greater or equal than the number of positive literals in U and the number of negative literals in S is greater or equal than the number of negative literals in U. This completes the proof of Claim 13. It remains to show the above claim.

First suppose that π is a permutation of the variables of $S \cup U$ such that $\pi(S) \Rightarrow U$. Since $\pi(S) - \{1\}$ is a satisfiable set of literals, it follows that, for all literals ℓ, if $\pi(S) \Rightarrow \ell$, then $\ell \in \pi(S)$. This implies that $U - \{1\} \subseteq \pi(S)$, and thus the number of positive literals in $\pi(S)$ is greater or equal than the number of positive literals in U and the number of negative literals in $\pi(S)$ is greater or equal than the number of negative literals in U.

For the converse, suppose that the number of positive literals in S is greater or equal than the number of positive literals in U and the number of negative literals in S is greater or equal than the number of negative literals in U. Since no variable occurs both positively and negatively in S or U, it is easy to compute a permutation π of the variables in $S \cup U$ such that every variable that occurs positively in U is mapped to by a variable that occurs positively in S and such that every variable that occurs negatively in U is mapped to by a variable that occurs negatively in S. It is immediate that $U - \{1\} \subseteq \pi(S)$, and thus $S \widetilde{\Rightarrow} U$. $\square$

4.2 Lower bounds

When proving dichotomy or dichotomy-like theorems for Boolean constraints, the proofs of some of the lower bounds are generally most involved. In addition, proving lower bounds for the case without constants is often a lot more involved than the proofs for the case with constants. This is particularly true in the case for isomorphism problems, since here, we cannot introduce auxiliary variables.

The approach taken in [BHRV02,BHRV04,BHRV03], which examine the complexity of the isomorphism problem for Boolean constraints, is to first prove lower bounds for the case with constants, and then to show that all the hardness reductions can be modified to obtain reductions for the cases without constants.

In contrast, in this paper we will prove the lower bounds directly for the case without constants. We have chosen this approach since careful analysis of the cases shows that proving the NP lower bounds boils down to proving NP-hardness for ten different cases (far fewer than in the isomorphism paper).

It should be noted that our NP lower bound results do not at all follow from the lower bound results for the isomorphism problem. This is also made clear by comparing Theorems 4 and 10: In some cases, the complexity jumps from P to NP-complete, in other cases we jump from GI-hard to NP-complete.

Lemma 14. *Let C be a k-ary constraint such that $C(x_1, \ldots, x_k)$ is not equivalent to a conjunction of literals. Then there exists a set of constraint applications of C that is equivalent to one of the following ten constraint applications:*

– $t \wedge (x \vee y)$, $\overline{f} \wedge t \wedge (x \vee y)$, $\overline{f} \wedge (\overline{x} \vee \overline{y})$, $\overline{f} \wedge t \wedge (\overline{x} \vee \overline{y})$,
– $x \leftrightarrow y$, $t \wedge (x \leftrightarrow y)$, $\overline{f} \wedge (x \leftrightarrow y)$, $\overline{f} \wedge t \wedge (x \leftrightarrow y)$,
– $x \oplus y$, *or* $\overline{f} \wedge t \wedge (x \oplus y)$.

Proof. Let C be a k-ary constraint such that $C(x_1, \ldots, x_k)$ is not equivalent to a conjunction of literals. First suppose that C is not 2-affine. It follows from [BHRV03, Lemma 24] that there exists a set S of constraint applications of C such that S is equivalent to $\overline{x} \wedge y$, $\overline{x} \vee y$, $x \oplus y$, $x \leftrightarrow y$, $t \wedge (\overline{x} \vee y)$, $t \wedge (x \leftrightarrow y)$, $t \wedge (x \vee y)$, $\overline{f} \wedge (\overline{x} \vee y)$, $\overline{f} \wedge (x \leftrightarrow y)$, or $\overline{f} \wedge (\overline{x} \vee \overline{y})$.

If $S(x, y)$ is equivalent to $\overline{x} \vee y$, then $S(x, y) \cup S(y, x)$ is equivalent to $x \leftrightarrow y$. If $S(t, x, y)$ is equivalent to $t \wedge (\overline{x} \vee y)$, then $S(t, x, y) \cup S(t, y, x)$ is equivalent to $t \wedge (x \leftrightarrow y)$. If $S(f, x, y)$ is equivalent to $\overline{f} \wedge (\overline{x} \vee y)$, then $S(f, x, y) \cup S(f, y, x)$ is equivalent to $\overline{f} \wedge (x \leftrightarrow y)$.

The only case that needs more work is the case that $S(x, y)$ is equivalent to $\overline{x} \wedge y$. From the proofs of Theorems 15 and 17 of [BHRV03], it follows that there exists a constraint application A of C such that $A(0, 1, x, y)$ is equivalent to $x \vee y$, $\overline{x} \vee \overline{y}$, $\overline{x} \vee y$, or $x \oplus y$. It follows that $S(f, t) \cup \{A(f, t, x, y), A(f, t, y, x)\}$ is equivalent to $\overline{f} \wedge t \wedge (x \vee y)$, $\overline{f} \wedge t \wedge (\overline{x} \vee \overline{y})$, $\overline{f} \wedge t \wedge (x \leftrightarrow y)$, or $\overline{f} \wedge t \wedge (x \oplus y)$. This completes the proof for the case that C is not 2-affine.

To finish the proof of Lemma 14, suppose that C is 2-affine. Since $C(x_1, \ldots, x_k)$ is not equivalent to a conjunction of literals, it is also not equivalent to 0, and it follows from [BHRV04, Lemma 9] that $C(x_1, \ldots, x_k)$ is equivalent to a formula of the form

$$\bigwedge_{x \in Z} \overline{x} \wedge \bigwedge_{x \in O} x \wedge \bigwedge_{i=1}^{\ell} \left(\left(\bigwedge_{x \in X_i} x \wedge \bigwedge_{y \in Y_i} \overline{y} \right) \vee \left(\bigwedge_{x \in X_i} \overline{x} \wedge \bigwedge_{y \in Y_i} y \right) \right),$$

where $Z, O, X_1, Y_1, \ldots, X_\ell, Y_\ell$ are pairwise disjoint subsets of $\{x_1, \ldots, x_k\}$ such that $X_i \cup Y_i \neq \emptyset$ for all $1 \leq i \leq \ell$. Since $C(x_1, \ldots, x_k)$ is not equivalent to a conjunction of literals, there exists an i such that $||X_i \cup Y_i|| \geq 2$.

In $C(x_1, \ldots, x_k)$, replace all variables in Z by f, and all variables in O by t.

If for some i, $X_i \neq \emptyset$ and $Y_i \neq \emptyset$, then replace all variables in $\bigcup_j X_j$ by x, and replace all variables in $\bigcup_j Y_j$ by y. In this case, the resulting constraint application is equivalent to $x \oplus y$, $t \wedge (x \oplus y)$, $\overline{f} \wedge (x \oplus y)$, or $\overline{f} \wedge t \wedge (x \oplus y)$. In the second case, note that $\{t \wedge (x \oplus y), t \wedge (t \oplus f)\}$ is a set of constraint applications of C that is equivalent to $\overline{f} \wedge t \wedge (x \oplus y)$. In the third case, note that $\{\overline{f} \wedge (x \oplus y), \overline{f} \wedge (t \oplus f)\}$ is a set of constraint applications of C that is equivalent to $\overline{f} \wedge t \wedge (x \oplus y)$.

If for all i, $X_i = \emptyset$ or $Y_i = \emptyset$, let i be such that $||X_i|| \geq 2$ or $||Y_i|| \geq 2$. Replace one of the variables in $X_i \cup Y_i$ by x and replace all other variables in $\bigcup_j X_j \cup Y_j$ by y. In this case, the resulting constraint application is equivalent to $x \leftrightarrow y$, $t \wedge (x \leftrightarrow y)$, $\overline{f} \wedge (x \leftrightarrow y)$, or $\overline{f} \wedge t \wedge (x \leftrightarrow y)$. □

4.3 The 10 reductions

We will now show that in each of the 10 cases of Lemma 14, the isomorphic implication problem is NP-hard. Some work can be avoided by observing that the isomorphic implication problem is computationally equivalent to the same problem where every constraint is replaced by a type of "complement."

In [Hem04], it is shown that the complexity of (quantified) satisfiability problems for a set of constraints $\mathcal{C}$ is the same as the complexity of the same problem for the set of constraints $\mathcal{C}^c$, where $\mathcal{C}^c$ is defined as follows.

Definition 15 ([Hem04]).

1. *For C a k-ary constraint, C^c is the k-ary constraint such that for all $s \in \{0,1\}^k$, $C^c(s) = C(\overline{s})$, where, as in the definition of complementive, $\overline{s} = (1-s_1)(1-s_2)\cdots(1-s_k)$ for $s = s_1 s_2 \cdots s_k$.*
2. *For $\mathcal{C}$ a finite set of constraints, $\mathcal{C}^c = \{C^c \mid C \in \mathcal{C}\}$.*
3. *For S a set of constraint applications of $\mathcal{C}$, $S^c = \{C^c(z_1, \ldots, z_k) \mid C(z_1, \ldots, z_k) \in S\}$.*

It is easy to see that any isomorphism from S to U is also an isomorphism from $S^c \cong U^c$ (and vice versa). This implies the following.

Lemma 16. ISO-IMP($\mathcal{C}$) *is computationally equivalent to* ISO-IMP($\mathcal{C}^c$).

As mentioned in the introduction and proven in Section 3, the NP-complete subgraph isomorphism problem is closely related to the isomorphic implication problem for sets of constraint applications of $\lambda xy.x \vee y$, in the following way: For a graph $\widehat{G}$, let $S(\widehat{G})$ be defined as $\{x_i \vee x_j \mid \{i,j\} \in E(\widehat{G})\}$. It is easy to see that for two graphs G and H without isolated vertices, G contains a subgraph isomorphic to H if and only if $S(G)$ isomorphically implies $S(H)$. This correspondence is also the reason for the GI-hardness for the isomorphism problem for sets of constraint applications of $\lambda xy.x \vee y$ [BRS98,BHRV02].

We will use the observation above to prove NP-hardness for constraints that are similar to $\lambda xy.x \vee y$, namely, we will reduce the subgraph isomorphism problem to the isomorphic implication problems for $\lambda txy.t \wedge (x \vee y)$, $\lambda ftxy.\overline{f} \wedge t \wedge (x \vee y)$, $\lambda fxy.\overline{f} \wedge (\overline{x} \vee \overline{y})$, and $\lambda ftxy.\overline{f} \wedge t \wedge (\overline{x} \vee \overline{y})$.

Claim 17 *1.* ISO-IMP($\{\lambda txy.t \wedge (x \vee y)\}$) *is* NP-*hard.*
2. ISO-IMP($\{\lambda ftxy.\overline{f} \wedge t \wedge (x \vee y)\}$) *is* NP-*hard.*
3. ISO-IMP($\{\lambda fxy.\overline{f} \wedge (\overline{x} \vee \overline{y})\}$) *is* NP-*hard.*
4. ISO-IMP($\{\lambda ftxy.\overline{f} \wedge t \wedge (\overline{x} \vee \overline{y})\}$) *is* NP-*hard.*

Proof.

1. Let G and H be two graphs without isolated vertices. For $\widehat{G}$ a graph, define
$$S(\widehat{G}) = \{t \wedge (x_i \vee x_j) \mid \{i,j\} \in E(\widehat{G})\}.$$
We claim that G contains a subgraph isomorphic to H if and only if $S(G) \cong\!\!\!\!\Rightarrow S(H)$. First suppose that G' is a subgraph of G and that G' is isomorphic to H. Then $S(G') \subseteq S(G)$ and there exists a bijection π from $V(G')$ to $V(H)$ such that $\pi(G') = H$, which implies that $S(\pi(G')) = S(H)$. Let ρ be a permutation on the set $\{t\} \cup \{x_i \mid i \in V(G) \cup V(H)\}$ such that $\rho(t) = t$ and

$\rho(x_i) = x_{\pi(i)}$ for all $i \in V(G')$. It is immediate that $\rho(S(G')) = S(\pi(G')) = S(H)$ and that $\rho(S(G')) \subseteq \rho(S(G))$. It follows that $S(H) \subseteq \rho(S(G))$, and thus $S(G) \overset{\cong}{\Rightarrow} S(H)$.

For the converse, suppose that there exists a permutation ρ on the variables occurring in $S(G) \cup S(H)$ such that $\rho(S(G)) \Rightarrow S(H)$. First note that such a ρ must map t to t, since, for any graph $\widehat{G}$ without isolated vertices, t is the unique variable z such that $S(\widehat{G}) \Rightarrow z$. Also note that for all graphs $\widehat{G}$, if $S(\widehat{G}) \Rightarrow t \wedge (x_i \vee x_j)$, then $t \wedge (x_i \vee x_j) \in S(\widehat{G})$.

It is easy to see that if $\rho(S(\widehat{G})) \Rightarrow t \wedge (x_i \vee x_j)$, then $t \wedge (x_i \vee x_j) \in \rho(S(\widehat{G}))$. (For if it were not, setting t to true, x_i and x_j to false, and all other x-variables to true would satisfy $\rho(S(\widehat{G}))$.) It follows that $S(H) \subseteq \rho(S(G))$. Let G' be the graph isomorphic to H such that $S(H) = \rho(S(G'))$. Then $\rho(S(G')) \subseteq \rho(S(G))$, and thus G' is a subgraph of G.

2. For $\widehat{G}$ a graph, define

$$S'(\widehat{G}) = \{\overline{f} \wedge t \wedge (x_i \vee x_j) \mid \{i, j\} \in E(\widehat{G})\}.$$

We claim that for any graphs G and H without isolated vertices, $S(G) \overset{\cong}{\Rightarrow} S(H)$ if and only if $S'(G) \overset{\cong}{\Rightarrow} S'(H)$.

First suppose that ρ is a permutation of the variables occurring in $S(G) \cup S(H)$ such that $\rho(S(G)) \Rightarrow S(H)$. If we extend ρ by letting $\rho(f) = f$, then $\rho(S'(G)) \Rightarrow S'(H)$. For the converse, suppose that ρ is a permutation of the variables occurring in $S'(G) \cup S'(H)$ such that $\rho(S'(G)) \Rightarrow S'(H)$. Then $\rho(f) = f$, since, for any graph $\widehat{G}$ without isolated vertices, f is the unique variable z such that $S(\widehat{G}) \Rightarrow \overline{z}$. Since, for any graph $\widehat{G}$, $S'(\widehat{G})$ is equivalent to $\overline{f} \wedge S(\widehat{G})$ and f does not occur in $S(\widehat{G})$, it follows immediately that $\rho(S(G)) \Rightarrow S(H)$.

3. Note that $(\lambda fxy.\overline{f} \wedge (\overline{x} \vee \overline{y}))^c = \lambda fxy.f \wedge (x \vee y)$. The result follows immediately from part 1 of this claim and Lemma 16.
4. Note that $(\lambda ftxy.\overline{f} \wedge t \wedge (\overline{x} \vee \overline{y}))^c = \lambda ftxy.f \wedge \overline{t} \wedge (x \vee y)$. The result follows immediately from part 2 of this claim and Lemma 16.

$\square$

The remaining 6 constraints behave differently. In these cases, the isomorphism problem is in P. Thus, GI does not reduce to these isomorphism problems (unless GI is in P), and there does not seem to be a simple reduction from the subgraph isomorphism problem to the isomorphic implication problem. In these cases, we will prove NP-hardness by reduction from a suitable partitioning problem, namely, the unary version of the problem 3-Partition [GJ79, Problem SP15].

Definition 18. *[GJ79]* Unary-3-Partition *is the problem, given a set A of $3m$ elements, $B \in \mathbb{Z}^+$ a bound (in unary), and for each $a \in A$, a size $s(a) \in \mathbb{Z}^+$ (in unary) such that $B/4 < s(a) < B/2$ and such that $\sum_{a \in A} s(a) = mB$, to decide whether A can be partitioned into m disjoint sets $A_1, \ldots, A_m$ such that $\sum_{a \in A_i} s(a) = B$ for $1 \leq i \leq m$.*

Theorem 19 ([GJ79]). Unary-3-Partition *is* NP*-complete.*

Claim 20 *1.* ISO-IMP($\{\lambda xy.x \leftrightarrow y\}$) *is* NP*-hard.*
2. ISO-IMP($\{\lambda txy.t \wedge (x \leftrightarrow y)\}$) *is* NP*-hard.*
3. ISO-IMP($\{\lambda fxy.\overline{f} \wedge (x \leftrightarrow y)\}$) *is* NP*-hard.*
4. ISO-IMP($\{\lambda ftxy.\overline{f} \wedge t \wedge (x \leftrightarrow y)\}$) *is* NP*-hard.*

Proof.

1. Let A be a set with $3m$ elements, $B \in \mathbb{Z}^+$ a bound (in unary), and for each $a \in A$, let $s(a) \in \mathbb{Z}^+$ be a size (in unary) such that $\sum_{a\in A} s(a) = mB$.
Let $X_1, \ldots, X_m$ be m pairwise disjoint sets of variables, each of size B. Let
$$S = \{x \leftrightarrow x' \mid x, x' \in X_i \text{ for some } i\}.$$
Let $\{\widehat{X}_a \mid a \in A\}$ be a collection of $3m$ pairwise disjoint sets of variables such that $||\widehat{X}_a|| = s(a)$ for all $a \in A$, and such that
$$\bigcup_{a\in A} \widehat{X}_a = \bigcup_{i=1}^{m} X_i.$$
Let
$$U = \{x \leftrightarrow x' \mid x, x' \in \widehat{X}_a \text{ for some } a \in A\}.$$
Note that since B and the $s(a)$'s are given in unary, S and U can be computed in polynomial time.
We claim that A can be partitioned into m disjoint sets $A_1, \ldots, A_m$ such that $\sum_{a\in A_i} s(a) = B$ for $1 \leq i \leq m$ if and only if $S \overset{\cong}{\Rightarrow} U$.
First suppose that $A_1, \ldots, A_m$ is a partition of A such that $\sum_{a\in A_i} s(a) = B$ for $1 \leq i \leq m$. Define a permutation π on $\bigcup_{i=1}^m X_i$ such that for all i, $\pi(X_i) = \bigcup_{a\in A_i} \widehat{X}_a$. Let $(x \leftrightarrow x') \in U$. Then, for some $a \in A$, $x, x' \in \widehat{X}_a$. Then there exists an i such that $\pi^{-1}(x)$ and $\pi^{-1}(x')$ are elements of X_i, which implies that $(\pi^{-1}(x) \leftrightarrow \pi^{-1}(x')) \in S$, and thus $(x \leftrightarrow x') \in \pi(S)$. It follows that $U \subseteq \pi(S)$, and thus $\pi(S) \Rightarrow U$.
For the converse, suppose π is a permutation of $\bigcup_{i=1}^m X_i$ such that $\pi(S) \Rightarrow U$. Let $A_i = \{a \mid \pi(X_i) \cap \widehat{X}_a \neq \emptyset\}$. We claim that $A_1, \ldots, A_m$ is a desired partition.
By definition, it is immediate that $\bigcup_{i=1}^m A_i = A$. Next suppose that $A_i \cap A_j \neq \emptyset$, for some $i \neq j$. Then for some $z \in X_i$, $z' \in X_j$, there exists an $a \in A$ such that $\pi(z), \pi(z') \in \widehat{X}_a$. Then $(\pi(z) \leftrightarrow \pi(z')) \in U$. But it easy to see that there exists a satisfying assignment of S such that z is true and z' is false. Thus, $S \not\Rightarrow (z \leftrightarrow z')$. This implies that $\pi(S) \not\Rightarrow (\pi(z) \leftrightarrow \pi(z'))$. But this contradicts the fact that $\pi(S) \Rightarrow U$.
It follows that $A_i \cap A_j = \emptyset$ for all $i \neq j$ and it follows that $\pi(X_i) = \bigcup_{a\in A_i} \widehat{X}_a$. Since π is an injection, $||X_i|| = ||\bigcup_{a\in A_i} \widehat{X}_a||$, and since the $\widehat{X}_a$'s are pairwise disjoint, $||\bigcup_{a\in A_i} \widehat{X}_a|| = \Sigma_{a\in A_i} s(a)$. Since $||X_i|| = B$, it follows that $\Sigma_{a\in A_i} s(a) = B$. This completes the NP-hardness proof for ISO-IMP($\{\lambda xy.x \leftrightarrow y\}$).

2. Now consider the case where the constraint is $\lambda txy.t \wedge (x \leftrightarrow y)$. Let

$$S' = \{t \wedge (x \leftrightarrow x') \mid x, x' \in X_i \text{ for some } i\}$$

and

$$U' = \{t \wedge (x \leftrightarrow x') \mid x, x' \in \widehat{X}_a \text{ for some } a \in A\}.$$

We claim that $S' \mathrel{\widetilde{\Rightarrow}} U'$ if and only if $S \mathrel{\widetilde{\Rightarrow}} U$. The left-to-right direction is immediate: Simply extend the permutation π on the variables occurring in $S \cup U$ such that $\pi(S) \Rightarrow U$ by letting $\pi(t) = t$. Then $\pi(S') \Rightarrow \pi(U')$.
For the converse, note that $S' \equiv t \wedge S$ and $U' \equiv t \wedge U$. Let π be such that $\pi(S') \Rightarrow U'$. Note that t is the unique variable z such that $S' \Rightarrow z$. It follows that π maps t to t. Since t does not occur in S and U, it follows that $\pi(S) \Rightarrow U$.
3. $(\lambda fxy.\overline{f} \wedge (x \leftrightarrow y))^c$ is equivalent to $\lambda txy.t \wedge (x \leftrightarrow y)$. The result follows immediately from part 2 of this claim and Lemma 16.
4. Now consider the case where the constraint is $\lambda ftxy.\overline{f} \wedge t \wedge (x \leftrightarrow y)$. Let

$$S'' = \{\overline{f} \wedge t \wedge (x \leftrightarrow x') \mid x, x' \in X_i \text{ for some } i\}$$

and

$$U'' = \{\overline{f} \wedge t \wedge (x \leftrightarrow x') \mid x, x' \in \widehat{X}_a \text{ for some } a \in A\}.$$

We claim that $S'' \mathrel{\widetilde{\Rightarrow}} U''$ if and only if $S \mathrel{\widetilde{\Rightarrow}} U$. The left-to-right direction is immediate: Simply extend the permutation π on the variables occurring in $S \cup U$ such that $\pi(S) \Rightarrow U$ by letting $\pi(f) = f$ and $\pi(t) = t$. Then $\pi(S'') \Rightarrow \pi(U'')$.
For the converse, note that $S'' \equiv \overline{f} \wedge t \wedge S$ and $U'' \equiv \overline{f} \wedge t \wedge U$. Let π be such that $\pi(S'') \Rightarrow U''$. Note that f is the unique variable z such that $S'' \Rightarrow \overline{z}$ and and that t is the unique variable z such that $S'' \Rightarrow z$. It follows that π maps t to t and f to f. Since t and f do not occur in S and U, it follows that $\pi(S) \Rightarrow U$.

$\square$

For the final two cases, we adapt the proof from the previous claim.

Claim 21 *1.* ISO-IMP($\{\lambda xy.x \oplus y\}$) *is* NP-*hard.*
2. ISO-IMP($\{\lambda ftxy.\overline{f} \wedge t \wedge (x \oplus y)\}$) *is* NP-*hard.*

Proof.

1. Let A be a set with $3m$ elements, $B \in \mathbb{Z}^+$ a bound (in unary), and for each $a \in A$, let $s(a) \in \mathbb{Z}^+$ be a size (in unary) such that $\sum_{a \in A} s(a) = mB$.
Let $X_1, \ldots, X_m, Y_1, \ldots, Y_m$ be $2m$ pairwise disjoint sets of variables, each of size B. Let

$$S = \{x \oplus y \mid x \in X_i \text{ and } y \in Y_i \text{ for some } i\}.$$

Let $\{\widehat{X}_a, \widehat{Y}_a \mid a \in A\}$ be a collection of $6m$ pairwise disjoint sets of variables such that $||\widehat{X}_a|| = ||\widehat{Y}_a|| = s(a)$ for all $a \in A$, and such that

$$\bigcup_{a \in A}(\widehat{X}_a \cup \widehat{Y}_a) = \bigcup_{i=1}^{m}(X_i \cup Y_i).$$

Let

$$U = \{x \oplus y \mid x \in \widehat{X}_a \text{ and } y \in \widehat{Y}_a \text{ for some } a \in A\}.$$

Note that since B and the $s(a)$'s are given in unary, S and U can be computed in polynomial time.

We claim that A can be partitioned into m disjoint sets $A_1, \ldots, A_m$ such that $\sum_{a \in A_i} s(a) = B$ for $1 \leq i \leq m$ if and only if $S \overset{\sim}{\Rightarrow} U$.

First suppose that $A_1, \ldots, A_m$ is a partition of A such that $\sum_{a \in A_i} s(a) = B$ for $1 \leq i \leq m$. Define a permutation π on $\bigcup_{i=1}^{m}(X_i \cup Y_i)$ such that for all i, $\pi(X_i) = \bigcup_{a \in A_i} \widehat{X}_a$ and $\pi(Y_i) = \bigcup_{a \in A_i} \widehat{Y}_a$. Consider an arbitrary element of U, say $x \oplus y$ for $x \in \widehat{X}_a$ and $y \in \widehat{Y}_a$. Then there exists an i such that $\pi^{-1}(x) \in X_i$ and $\pi^{-1}(y) \in Y_i$ (or vice versa), which implies that $(\pi^{-1}(x) \oplus \pi^{-1}(y)) \in S$, and thus $(x \oplus y) \in \pi(S)$. It follows that $U \subseteq \pi(S)$, and thus $\pi(S) \Rightarrow U$.

For the converse, suppose π is a permutation of $\bigcup_{i=1}^{m}(X_i \cup Y_i)$ such that $\pi(S) \Rightarrow U$.

Let

$$A_i = \{a \mid \pi(X_i \cup Y_i) \cap (\widehat{X}_a \cup \widehat{Y}_a) \neq \emptyset\}.$$

We claim that $A_1, \ldots, A_m$ is the desired partition.

By definition, it is immediate that $\bigcup_{i=1}^{m} A_i = A$. Next suppose that $A_i \cap A_j \neq \emptyset$, for some $i \neq j$. Then for some $z \in X_i \cup Y_i$, $z' \in X_j \cup Y_j$, there exists an $a \in A$ such that $\pi(z), \pi(z') \in \widehat{X}_a \cap \widehat{Y}_a$.

Then $U \Rightarrow (\pi(z) \oplus \pi(z'))$ or $U \Rightarrow (\pi(z) \leftrightarrow \pi(z'))$. But it easy to see that there exists a satisfying assignment of S such that z is true and z' is false, and that there exists a satisfying assignment of S such that z is true and z' is true. Thus, $S \not\Rightarrow (z \oplus z')$ and $S \not\Rightarrow (z \leftrightarrow z')$. This implies that $\pi(S) \not\Rightarrow (\pi(z) \oplus \pi(z'))$ and $\pi(S) \not\Rightarrow (\pi(z) \leftrightarrow \pi(z'))$. But this contradicts the fact that $\pi(S) \Rightarrow U$.

It follows that $A_i \cap A_j = \emptyset$ for all $i \neq j$ and it follows that $\pi(X_i \cup Y_i) = \bigcup_{a \in A_i}(\widehat{X}_a \cup \widehat{Y}_a)$. Since π is an injection, $||X_i \cup Y_i|| = ||\bigcup_{a \in A_i}(\widehat{X}_a \cup \widehat{Y}_a)|| = \Sigma_{a \in A_i} 2s(a)$. Since $||X_i \cup Y_i|| = 2B$ it follows that $\Sigma_{a \in A_i} 2s(a) = 2B$, which, of course, implies that $\Sigma_{a \in A_i} s(a) = B$, as required.

2. Now consider the case where the constraint is $\lambda ftxy.\overline{f} \wedge t \wedge (x \oplus y)$. Let

$$S' = \{\overline{f} \wedge t \wedge x \oplus y \mid x \in X_i \text{ and } y \in Y_i \text{ for some } i\}$$

and

$$U' = \{\overline{f} \wedge t \wedge x \oplus y \mid x \in \widehat{X}_a \text{ and } y \in \widehat{Y}_a \text{ for some } a \in A\}.$$

We claim that $S' \overset{\sim}{\Rightarrow} U'$ if and only if $S \overset{\sim}{\Rightarrow} U$. The left-to-right direction is immediate: Simply extend the permutation π on the variables occurring

in $S \cup U$ such that $\pi(S) \Rightarrow U$ by letting $\pi(f) = f$ and $\pi(t) = t$. Then $\pi(S') \Rightarrow \pi(U')$.
For the converse, note that $S' \equiv \overline{f} \wedge t \wedge S$ and $U' \equiv \overline{f} \wedge t \wedge U$. Let π be such that $\pi(S') \Rightarrow U'$. Note that f is the unique variable z such that $S' \Rightarrow \overline{z}$ and and that t is the unique variable z such that $S' \Rightarrow z$. It follows that π maps t to t and f to f. Since t and f do not occur in S and U, it follows that $\pi(S) \Rightarrow U$.

□

To complete the proof of Theorem 10, it remains to show the following claim.

Claim 22 *Let $\mathcal{C}$ be a finite set of constraints. If $\mathcal{C}$ is not Schaefer, then* ISO-IMP($\mathcal{C}$) *is* coNP*-hard.*

Proof. The exact same reductions that show the coNP-hardness for ISO($\mathcal{C}$) from [BHRV02, Claim 19] also show coNP-hardness for ISO-IMP($\mathcal{C}$). This is because for all pairs of sets of constraint applications (S, U) of $\mathcal{C}$ that these reductions map to, it holds that $U \overset{\sim}{\Rrightarrow} S$. Under this condition, $S \cong U$ if and only if $S \overset{\sim}{\Rrightarrow} U$. □

5 Toward a Trichotomy Theorem

The current main theorem (Theorem 10) is not a trichotomy theorem, since for $\mathcal{C}$ not Schaefer, it states that ISO-IMP($\mathcal{C}$) is NP-hard, coNP-hard, and in $\mathrm{P}^{\mathrm{NP}}_{||}$. The large gap between the lower and upper bounds is not very satisfying. We conjecture that the current lower bounds for ISO-IMP($\mathcal{C}$) for $\mathcal{C}$ not Schaefer can be raised to $\mathrm{P}^{\mathrm{NP}}_{||}$ lower bounds, which would give the following trichotomy theorem.

Conjecture 23 *Let $\mathcal{C}$ be a finite set of constraints.*

1. *If every constraint in $\mathcal{C}$ is equivalent to a constant or a conjunction of literals, then* ISO-IMP($\mathcal{C}$) *and* ISO-IMP$_c$($\mathcal{C}$) *are in* P.
2. *Otherwise, if $\mathcal{C}$ is Schaefer, then* ISO-IMP($\mathcal{C}$) *and* ISO-IMP$_c$($\mathcal{C}$) *are* NP*-complete.*
3. *If $\mathcal{C}$ is not Schaefer, then* ISO-IMP($\mathcal{C}$) *and* ISO-IMP$_c$($\mathcal{C}$) *are* $\mathrm{P}^{\mathrm{NP}}_{||}$*-complete.*

We believe this conjecture for two reasons. First of all, it is quite common for problems that are NP-hard, coNP-hard, and in $\mathrm{P}^{\mathrm{NP}}_{||}$ to end up being $\mathrm{P}^{\mathrm{NP}}_{||}$-complete. (For an overview of this phenomenon, see [HHR97].) Secondly, we will prove $\mathrm{P}^{\mathrm{NP}}_{||}$ lower bounds for some cases in Theorem 28.

To raise NP and coNP lower bounds to $\mathrm{P}^{\mathrm{NP}}_{||}$ lower bounds, the following theorem by Wagner often plays a crucial role, which it will also do in our case.

Theorem 24 ([Wag87]). *Let L be a language. If there exists a polynomial-time computable function h such that*

$$||\{i \mid \phi_i \in \mathrm{SAT}\}|| \textit{ is odd iff } h(\phi_1, \ldots, \phi_{2k}) \in L$$

for all $k \geq 1$ and all Boolean formulas $\phi_1, \ldots, \phi_{2k}$ such that $\phi_i \in \mathrm{SAT} \Rightarrow \phi_{i+1} \in \mathrm{SAT}$, then L is $\mathrm{P}_{||}^{\mathrm{NP}}$-hard.

The basic idea behind applying Wagner's theorem to turn an NP lower bound and a coNP lower bound into a $\mathrm{P}_{||}^{\mathrm{NP}}$ lower bound is the following.

Lemma 25. *Let L be a language. If L is* NP*-hard and* coNP*-hard, and (L has polynomial-time computable and- and ω-or functions or L has polynomial-time computable or- and ω-and functions), then L is $\mathrm{P}_{||}^{\mathrm{NP}}$-hard.*[1]

Proof. First suppose that L has a polynomial-time computable and-function *and*, and a polynomial-time computable ω-or function *or*. Let f be a reduction from $\overline{\mathrm{SAT}}$ to L and let g be a reduction from SAT to L.

Let $k \geq 1$ and let $\phi_1, \ldots, \phi_{2k}$ be formulas such that $\phi_i \in \mathrm{SAT} \Rightarrow \phi_{i+1} \in \mathrm{SAT}$. Note that $||\{i \mid \phi_i \in \mathrm{SAT}\}||$ is odd if and only if there exists an i such that $1 \leq i \leq k$, $\phi_{2i-1} \notin \mathrm{SAT}$, and $\phi_{2i} \in \mathrm{SAT}$.

Define $h(\phi_1, \ldots, \phi_{2k})$ as

$$\mathit{or}\,(\mathit{and}(f(\phi_1), g(\phi_2)), \mathit{and}(f(\phi_3), g(\phi_4)), \ldots, \mathit{and}(f(\phi_{2k-1}), g(\phi_{2k})))\,.$$

It is immediate that h is computable in polynomial-time and there exists an i such that $1 \leq i \leq k$, $\phi_{2i-1} \notin \mathrm{SAT}$, and $\phi_{2i} \in \mathrm{SAT}$ if and only if $h(\phi_1, \ldots, \phi_{2k}) \in L$. It follows that L is $\mathrm{P}_{||}^{\mathrm{NP}}$-hard by Theorem 24.

Now consider the case that L has a polynomial-time computable or-function, and a polynomial-time computable ω-and function. Then $\overline{L}$ has a polynomial-time computable and-function, and a polynomial-time computable ω-or function. By the argument above, $\overline{L}$ is $\mathrm{P}_{||}^{\mathrm{NP}}$-hard. Since $\mathrm{P}_{||}^{\mathrm{NP}}$ is closed under complement, it follows that L is $\mathrm{P}_{||}^{\mathrm{NP}}$-hard. □

Agrawal and Thierauf [AT00] proved that the Boolean isomorphism problem has ω-and and ω-or functions. Since the Boolean isomorphism problem is trivially coNP-hard, we obtain the following corollary.

Corollary 26. *If the Boolean isomorphism problem is* NP*-hard, then it is $\mathrm{P}_{||}^{\mathrm{NP}}$-hard.*

Unfortunately, Agrawal and Thierauf's ω-or function does not work for Boolean isomorphic implication. Their ω-and function seems to work for Boolean isomorphic implication, but since this function or's two formulas together, it will not work for sets of constraint applications.

To prove our $\mathrm{P}_{||}^{\mathrm{NP}}$ lower bounds, we need to come up with completely new constructions. In the proof, we will use the following lemma.

[1] An or-function for a language L is a function f such that for all $x, y \in \Sigma^*$, $f(x, y) \in L$ iff $x \in L$ or $y \in L$. An ω-or-function for a language L is a function f such that for all $x_1, \ldots, x_n \in \Sigma^*$, $f(x_1, \ldots, x_n) \in L$ iff $x_i \in L$ for some i; and-functions are defined similarly [KST93].

Lemma 27. *Let S and U be two sets of constraint applications, let X be the set of variables occurring in S, and let Y be the set of variables occurring in U. If $S \cong\!\!\!\Rightarrow U$, $||X|| \geq ||Y||$, and $X \cap Y = \emptyset$, then there exists a permutation π of $X \cup Y$ such that $\pi(S) \Rightarrow U$ and $\pi(Y) \cap Y = \emptyset$.*

Proof. Let π' be a permutation such that $\pi'(S) \Rightarrow U$. If there exist $y, y' \in Y$ with $\pi'(y') = y$, then, since $||X|| \geq ||Y||$, there exist $x, x' \in X$ such that $\pi'(x') = x$. Construct a new permutation ρ as follows: $\rho(y') = x$, $\rho(x') = y$, $\rho(z) = \pi'(z)$ for all $z \in (X \cup Y) - \{x', y'\}$. We will show that $\rho(S) \Rightarrow U$. By repeatedly applying this construction, we get a permutation π such that $\pi(S) \Rightarrow U$ and $\pi(Y) \subseteq X$. Since $X \cap Y = \emptyset$, it follows that $\pi(Y) \cap Y = \emptyset$.

It remains to show that $\rho(S) \Rightarrow U$. For this, suppose that $\rho(S) \not\Rightarrow U$. Let Z be a list of the variables in $(S \cup U) - \{x', y'\}$. Then $\rho(S)(Z, x', y') \not\Rightarrow U(Z, x', y')$ and thus there exist $a, b \in \{0, 1\}$ and $s \in \{0, 1\}^{||Z||}$, such that $\rho(S)(s, a, b) = 1$ and $U(s, a, b) = 0$. Since x' does not occur in U and y' does not occur in S, we also have $\rho(S)(s, a, \overline{b}) = 1$ and $U(s, \overline{a}, b) = 0$. Since $\pi'(S)(Z, x, y) \Rightarrow U$ it follows that $\pi'(S)(s, \overline{a}, b) = \pi'(S)(s, a, b) = 0$. Because y' does not occur in S, also $\pi'(S)(s, \overline{a}, \overline{b}) = \pi'(S)(s, a, \overline{b}) = 0$. So, $\pi'(S)(s, x, y) \equiv 0$, but $\pi'(S)(s, b, a) = \rho(S)(s, a, b) = 1$. □

Theorem 28. *Let $\mathcal{D}$ be a set of constraints that is 0-valid, 1-valid, not complementive, and not Schaefer. Let $\mathcal{C} = \mathcal{D} \cup \{\lambda xy.x \vee y\}$. Then* ISO-IMP($\mathcal{C}$) *is* $\mathrm{P}_{||}^{\mathrm{NP}}$*-complete.*

Proof. By Theorem 10, ISO-IMP($\mathcal{C}$) is in $\mathrm{P}_{||}^{\mathrm{NP}}$. Thus it suffices to show that ISO-IMP($\mathcal{C}$) is $\mathrm{P}_{||}^{\mathrm{NP}}$-hard. Let $k \geq 1$ and let $\phi_1, \ldots, \phi_{2k}$ be formulas such that $\phi_i \in \mathrm{SAT} \Rightarrow \phi_{i+1} \in \mathrm{SAT}$. We will construct a polynomial-time computable function h such that

$$||\{i \mid \phi_i \in \mathrm{SAT}\}|| \text{ is odd iff } h(\phi_1, \ldots, \phi_{2k}) \in \text{ISO-IMP}(\mathcal{C}).$$

By Theorem 24, this proves that ISO-IMP($\mathcal{C}$) is $\mathrm{P}_{||}^{\mathrm{NP}}$-hard.

Note that $||\{i \mid \phi_i \in \mathrm{SAT}\}||$ is odd if and only if there exists an i such that $1 \leq i \leq k$, $\phi_{2i-1} \notin \mathrm{SAT}$, and $\phi_{2i} \in \mathrm{SAT}$. This is a useful way of looking at it, and we will prove that there exists an i such that $1 \leq i \leq k$, $\phi_{2i-1} \notin \mathrm{SAT}$ and $\phi_{2i} \in \mathrm{SAT}$ if and only if $h(\phi_1, \ldots, \phi_{2k}) \in$ ISO-IMP($\mathcal{C}$).

From Theorem 10 we know that ISO-IMP($\mathcal{C}$) is NP-hard and coNP-hard, and thus there exist (polynomial-time many-one) reductions from SAT to ISO-IMP($\mathcal{C}$) and from $\overline{\mathrm{SAT}}$ to ISO-IMP($\mathcal{C}$). We will follow the idea of the proof of Lemma 25, but we will look in more detail at the reductions, so that we can restrict the sets of constraint applications that we have to handle.

Let f be a polynomial-time computable function such that for all ϕ, $f(\phi)$ is a set of constraint applications of $\mathcal{D}$ and

$$\phi \in \overline{\mathrm{SAT}} \text{ iff } f(\phi) \cong\!\!\!\Rightarrow \bigcup_{1 \leq j, \ell \leq n} \{x_j \rightarrow x_\ell\}.$$

Here $x_1, \ldots, x_n$ are exactly all variables in $f(\phi)$. Such a function exists, since $\overline{\text{SAT}}$ is reducible to $\overline{\text{CSP}_{\neq \mathbf{0},\mathbf{1}}(\mathcal{D})}$ ($\text{CSP}_{\neq \mathbf{0},\mathbf{1}}(\mathcal{D})$ is the problem of deciding whether a set of constraint applications of $\mathcal{D}$ has a satisfying assignment other than $\mathbf{0}$ and $\mathbf{1}$), which is reducible to ISO-IMP($\mathcal{D}$) via a reduction that satisfies the properties above. (See the proofs of Claim 22 and [BHRV02, Claims 19 and 14].)

Let g be a polynomial-time computable function such that for all ϕ, $g(\phi)$ is a set of constraint applications of $\lambda xy.x \vee y$ without duplicates (i.e., if $z \vee z' \in g(\phi)$, then $z \neq z'$) and

$$\phi \in \text{SAT iff } g(\phi) \widetilde{\Rrightarrow} \{y_j \vee y_{j+1} \mid 1 \leq j < n\}.$$

Here $y_1, \ldots, y_n$ are exactly all variables occurring in $g(\phi)$. Such a function exists, since SAT is reducible to HAMILTONIAN PATH, which is reducible to ISO-IMP($\{\lambda xy.x \vee y\}$) via a reduction that satisfies the properties above. (Basically, use the standard translation from graphs to sets of constraint applications of $\lambda xy.x \vee y$: For G a connected graph on vertices $\{1, \ldots, n\}$, let $g(G) = \{y_i \vee y_j \mid \{i, j\} \in E(G)\}$.)

Recall that we need to construct a polynomial-time computable function h with the property that there exists an i such that $1 \leq i \leq k$, $\phi_{2i-1} \notin \text{SAT}$, and $\phi_{2i} \in \text{SAT}$ if and only if $h(\phi_1, \ldots, \phi_{2k}) \in \text{ISO-IMP}(\mathcal{C})$.

In order to construct h, we will apply the coNP-hardness reduction f on ϕ_i for odd i, and the NP-hardness reduction g on ϕ_i for even i. It will be important to make sure that all obtained sets of constraint applications are over disjoint sets of variables.

For every i, $1 \leq i \leq k$, we define O_i to be the set of constraint applications $f(\phi_{2i-1})$ with each variable x_j replaced by $x_{i,j}$. Clearly,

$$\phi_{2i-1} \notin \text{SAT iff } O_i \widetilde{\Rrightarrow} \bigcup_{1 \leq j, \ell \leq n_i} \{x_{i,j} \rightarrow x_{i,\ell}\},$$

where n_i is the n from $f(\phi_{2i-1})$.

For every i, $1 \leq i \leq k$, we define E_i to be the set of constraint applications $g(\phi_{2i})$ with each variable y_j replaced by $y_{i,j}$. Clearly,

$$\phi_{2i} \in \text{SAT iff } E_i \widetilde{\Rrightarrow} \{y_{i,j} \vee y_{i,j+1} \mid 1 \leq j < n'_i\},$$

where n'_i is the n from $g(\phi_{2i})$.

Note that the sets that occur to the right of $O_i \widetilde{\Rrightarrow}$ are almost isomorphic (apart from the number of variables). The same holds for the sets that occur to the right of $E_i \widetilde{\Rrightarrow}$. It is important to make sure that these sets are exactly isomorphic. In order to do so, we simply pad the sets O_i and E_i.

Let $n = \max\{n_i, n'_i + 2 \mid 1 \leq i \leq k\}$. For $1 \leq i \leq k$, let

$$\widehat{O}_i = O_i \cup \{x_{i,1} \rightarrow x_{i,j}, x_{i,j} \rightarrow x_{i,1} \mid n_i < j \leq n\}.$$

$\widehat{O}_i$ is a set of constraint applications of $\mathcal{D}$, since there exists a constraint application $A(x, y)$ of $\mathcal{D}$ that is equivalent to $x \rightarrow y$ (see [BHRV02, Claim 14]).

It is immediate that

$$\widehat{O}_i \stackrel{\sim}{\Rrightarrow} \bigcup_{1 \le j,\ell \le n} \{x_{i,j} \to x_{i,\ell}\} \text{ iff } O_i \stackrel{\sim}{\Rrightarrow} \bigcup_{1 \le j,\ell \le n_i} \{x_{i,j} \to x_{i,\ell}\}.$$

For $1 \le i \le k$, let

$$\widehat{E}_i = E_i \cup \{y_{i,j} \vee y_{i,n'_i+1} \mid 1 \le j \le n'_i\} \cup \{y_{i,j} \vee y_{i,j+1} \mid n'_i + 1 \le j < n\}.$$

Then

$$\widehat{E}_i \stackrel{\sim}{\Rrightarrow} \{y_{i,j} \vee y_{i,j+1} \mid 1 \le j < n\} \text{ iff } E_i \stackrel{\sim}{\Rrightarrow} \{y_{i,j} \vee y_{i,j+1} \mid 1 \le j < n'_i\}.$$

The right-to-left direction is immediate. The left-to-right to direction can easily be seen if we think about this as graphs. Since $n \ge n'_i + 2$, any Hamiltonian path in $\widehat{E}_i$ contains the subpath $n'_{i+1}, n'_{i+2}, \ldots, n$, where n is an endpoint. This implies that there is a Hamiltonian path in the graph restricted to $\{1, \ldots, n'_i\}$, i.e., in E_i.

So, our current situation is as follows. For all i, $1 \le i \le k$, $\widehat{O}_i$ is a set of constraint applications of $\mathcal{D}$ such that

$$\phi_{2i-1} \notin \text{SAT iff } \widehat{O}_i \stackrel{\sim}{\Rrightarrow} \bigcup_{1 \le j,\ell \le n} \{x_{i,j} \to x_{i,\ell}\}.$$

and $\widehat{E}_i$ is a set of constraint applications of $\lambda xy.x \vee y$ without duplicates such that

$$\phi_{2i} \in \text{SAT iff } \widehat{E}_i \stackrel{\sim}{\Rrightarrow} \{y_{i,j} \vee y_{i,j+1} \mid 1 \le j < n\}.$$

Our reduction is defined as follows

$$h(\phi_1, \ldots, \phi_{2k}) = \langle S, U \rangle,$$

where

$$S = \bigcup_{i=1}^{k} \left(\widehat{O}_i \cup \widehat{E}_i \cup \bigcup_{1 \le j,\ell \le n} \{x_{i,j} \to y_{i,\ell}\} \right)$$

and

$$U = \bigcup_{1 \le j,\ell \le n} \{x_j \to x_\ell\} \cup \bigcup_{j=1}^{n-1} \{y_j \vee y_{j+1}\} \cup \bigcup_{1 \le j,\ell \le n} \{x_j \to y_\ell\}.$$

Clearly, h is computable in polynomial time and S and U are sets of constraint applications of $\mathcal{C}$, since there exists a constraint application $A(x, y)$ of $\mathcal{D}$ that is equivalent to $x \to y$.

It remains to show that there exists an i such that $\widehat{O}_i \stackrel{\sim}{\Rrightarrow} \bigcup_{1 \le j,\ell \le n} \{x_{i,j} \to x_{i,\ell}\}$ and $\widehat{E}_i \stackrel{\sim}{\Rrightarrow} \{y_{i,j} \vee y_{i,j+1} \mid 1 \le j < n\}$ if and only if $S \stackrel{\sim}{\Rrightarrow} U$.

For the left-to-right direction, let i_0 be such that $1 \le i_0 \le k$, $\widehat{O}_{i_0} \stackrel{\sim}{\Rrightarrow} \bigcup_{1 \le j,\ell \le n} \{x_{i_0,j} \to x_{i_0,\ell}\}$ and $\widehat{E}_{i_0} \stackrel{\sim}{\Rrightarrow} \{y_{i_0,j} \vee y_{i_0,j+1} \mid 1 \le j < n\}$. Let π_x be a permutation of

$\{x_{i_0,1},\ldots,x_{i_0,n}\}$ such that $\pi_x(\widehat{O}_{i_0}) \Rightarrow \bigcup_{1\leq j,\ell\leq n}\{x_{i_0,j}\rightarrow x_{i_0,\ell}\}$ and let π_y be a permutation of $\{y_{i_0,1},\ldots,y_{i_0,n}\}$ such that $\pi_y(\widehat{E}_{i_0}) \Rightarrow \{y_{i_0,j}\vee y_{i_0,j+1} \mid 1\leq j<n\}$.

Define a permutation π on the variables occurring in $S\cup U$ such that for all $1\leq j,\ell\leq n$, $\pi(x_{i_0,j}) = x_\ell$ if $\pi_x(x_{i_0,j}) = x_{i_0,\ell}$ and $\pi(y_{i_0,j}) = y_\ell$ if $\pi_y(y_{i_0,j}) = y_{i_0,\ell}$. It is immediate that $\pi(\widehat{O}_{i_0}) \Rightarrow \bigcup_{1\leq j,\ell\leq n}\{x_j\rightarrow x_\ell\}$, $\pi(\widehat{E}_{i_0}) \Rightarrow \{y_j\vee y_{j+1} \mid 1\leq j<n\}$, and $\pi(\bigcup_{1\leq j,\ell\leq n}\{x_{i_0,j}\rightarrow y_{i_0,\ell}\}) \Rightarrow \bigcup_{1\leq j,\ell\leq n}\{x_j\rightarrow y_\ell\}$. It follows that $\pi(S)\Rightarrow U$.

For the converse, suppose that $S \stackrel{\sim}{\Rightarrow} U$. It is easy to see (see Lemma 27) that there exists a permutation π of the variables that occur in $S\cup U$ such that $\pi(S)\Rightarrow U$ and such that for all j, $1\leq j\leq n$, $\pi(x_j)$ and $\pi(y_j)$ do not occur in U.

We will now show that for all $1\leq j\leq n$, π cannot map a y-variable to x_j. For suppose that $\pi(y_{i,\ell}) = x_j$. S is satisfied by the assignment that sets all y-variables to 1 and all x-variables to 0, and S remains satisfied if in this assignment we change the value of $y_{i,\ell}$ to 0 (recall that if $z\vee z'\in E_j$ then $z\neq z'$). Then $\pi(S)$ is satisfied by the assignment that sets $\pi(y)$ to 1 for all y-variables and $\pi(x)$ to 0 for all x-variables, and $\pi(S)$ is still satisfied if in this assignment we change the value of $\pi(y_{i,\ell})$ to 0. But this is a contradiction, since $\pi(y_{i,\ell}) = x_j$ and changing the value of x_j in a satisfying assignment for U will always make U false.

Let i_0, j_0 be such that $\pi(x_{i_0,j_0}) = x_1$. Now suppose that $\pi(z) = x_j$. Then $z = x_{i,\ell}$. Since $\pi(S)\Rightarrow(x_1\leftrightarrow x_j)$, $S\Rightarrow(x_{i_0,j_0}\leftrightarrow x_{i,\ell})$. It follows that $i = i_0$, and thus, $\pi(\{x_{i_0,\ell}\mid 1\leq\ell\leq n\}) = \{x_\ell\mid 1\leq\ell\leq n\}$.

Next, suppose that $\pi(z) = y_j$. Since $\pi(S)\Rightarrow(x_1\rightarrow y_j)$, $S\Rightarrow(x_{i_0,j_0}\rightarrow z)$. It follows that $z = x_{i_0,\ell}$ or $z = y_{i_0,\ell}$. Since $\pi(\{x_{i_0,\ell}\mid 1\leq\ell\leq n\}) = \{x_\ell\mid 1\leq\ell\leq n\}$, the only possibility is $z = y_{i_0,\ell}$. It follows that $\pi(\{y_{i_0,\ell}\mid 1\leq\ell\leq n\}) = \{y_\ell\mid 1\leq\ell\leq n\}$.

Let α be the partial assignment that sets all y-variables to 1, and all x-variables except those in $\{x_\ell\mid 1\leq\ell\leq n\}$ to 0. Then $\pi(S)[\alpha]$ is equivalent to $\pi(\widehat{O}_{i_0})$ and $U[\alpha]$ is equivalent to $\bigcup_{1\leq j,\ell\leq n}\{x_j\rightarrow x_\ell\}$. Since $\pi(S)\Rightarrow U$, $\pi(S)[\alpha]\Rightarrow U[\alpha]$, i.e., $\pi(\widehat{O}_{i_0})\Rightarrow\bigcup_{1\leq j,\ell\leq n}\{x_j\rightarrow x_\ell\}$, and thus $\widehat{O}_{i_0} \stackrel{\sim}{\Rightarrow} \bigcup_{1\leq j,\ell\leq n}\{x_{i_0,j}\rightarrow x_{i_0,\ell}\}$.

Let β be the partial assignment that sets all x-variables to 0, and all y-variables except those in $\{y_\ell\mid 1\leq\ell\leq n\}$ to 1. Then $\pi(S)[\beta]$ is equivalent to $\pi(\widehat{E}_{i_0})$ and $U[\beta]$ is equivalent to $\bigcup_{j=1}^{n-1}\{y_j\vee y_{j+1}\}$. Since $\pi(S)\Rightarrow U$, $\pi(S)[\beta]\Rightarrow U[\beta]$. It follows that $\pi(\widehat{E}_{i_0})\Rightarrow\bigcup_{j=1}^{n-1}\{y_j\vee y_{j+1}\}$, and thus $\widehat{E}_{i_0} \stackrel{\sim}{\Rightarrow} \bigcup_{j=1}^{n-1}\{y_{i_0,j}\vee y_{i_0,j+1}\}$. This completes the proof of Theorem 28. $\square$

It should be noted that constructions similar to the proof of Theorem 28 can be used to prove $\mathrm{P}^{\mathrm{NP}}_{||}$-hardness for some other cases as well. However, new insights and constructions will be needed to obtain $\mathrm{P}^{\mathrm{NP}}_{||}$-hardness for all non-Schaefer cases.

6 Open Problems

The most important question left open by this paper is whether Conjecture 23 holds. In addition, the complexity of the isomorphic implication problem for Boolean formulas is still open. This problem is trivially in Σ_2^p, and, by Theorem 28, $\mathrm{P}_{||}^{\mathrm{NP}}$-hard. Note that an improvement of the upper bound will likely give an improvement of the best-known upper bound (Σ_2^p) for the isomorphism problem for Boolean formulas, since that problem is 2-conjunctive-truth-table reducible to the isomorphic implication problem.

Schaefer's framework is not the only framework to study generalized Boolean problems. It would be interesting to study the complexity of isomorphic implication in other frameworks, for example, for Boolean circuits over a fixed base.

Acknowledgments: The authors thank Henning Schnoor and Heribert Vollmer for helpful comments.

References

AT00. M. Agrawal and T. Thierauf. The formula isomorphism problem. *SIAM Journal on Computing*, 30(3):990–1009, 2000.

BCRV04. E. Böhler, N. Creignou, S. Reith, and H. Vollmer. Playing with Boolean blocks, part II: Constraint Satisfaction Problems. *SIGACT News*, 35(1):22–35, 2004.

BH91. S. Buss and L. Hay. On truth-table reducibility to SAT. *Information and Computation*, 91(1):86–102, 1991.

BHRV02. E. Böhler, E. Hemaspaandra, S. Reith, and H. Vollmer. Equivalence and isomorphism for Boolean constraint satisfaction. In *Proceedings of the 16th Annual Conference of the EACSL (CSL 2002)*, pages 412–426. Springer-Verlag *Lecture Notes in Computer Science #2471*, September 2002.

BHRV03. E. Böhler, E. Hemaspaandra, S. Reith, and H. Vollmer. The complexity of Boolean constraint isomorphism. Technical Report cs.CC/0306134, Computing Research Repository, http://www.acm.org/repository/, June 2003. Revised, April 2004.

BHRV04. E. Böhler, E. Hemaspaandra, S. Reith, and H. Vollmer. The complexity of Boolean constraint isomorphism. In *Proceedings of the 21st Symposium on Theoretical Aspects of Computer Science*, pages 164–175. Springer-Verlag *Lecture Notes in Computer Science #2996*, March 2004.

BKJ00. A. Bulatov, A. Krokhin, and P. Jeavons. Constraint satisfaction problems and finite algebras. In *Proceedings of the 27th International Colloquium on Automata, Languages and Programming*, pages 272–282. Springer-Verlag, 2000.

BR93. B. Borchert and D. Ranjan. The ciruit subfunction relations are Σ_2^p-complete. Technical Report MPI-I-93-121, MPI, Saarbrücken, 1993.

BRS98. B. Borchert, D. Ranjan, and F. Stephan. On the computational complexity of some classical equivalence relations on Boolean functions. *Theory of Computing Systems*, 31(6):679–693, 1998.

CH96. N. Creignou and M. Hermann. Complexity of generalized satisfiability counting problems. *Information and Computation*, 125:1–12, 1996.

CKS01. N. Creignou, S. Khanna, and M. Sudan. *Complexity Classifications of Boolean Constraint Satisfaction Problems*. Monographs on Discrete Applied Mathematics. SIAM, 2001.

Coo71. S. Cook. The complexity of theorem-proving procedures. In *Proceedings of the 3rd ACM Symposium on Theory of Computing*, pages 151–158. ACM Press, 1971.

Cre95. N. Creignou. A dichotomy theorem for maximum generalized satisfiability problems. *Journal of Computer and System Sciences*, 51:511–522, 1995.

GJ79. M. Garey and D. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman and Company, 1979.

Hem04. E. Hemaspaandra. Dichotomy theorems for alternation-bounded quantified Boolean formulas. Technical Report cs.CC/0406006, Computing Research Repository, http://www.acm.org/repository/, June 2004.

HHR97. E. Hemaspaandra, L. Hemaspaandra, and J. Rothe. Raising NP lower bounds to parallel NP lower bounds. *SIGACT News*, 28(2):2–13, 1997.

JCG97. P. Jeavons, D. Cohen, and M. Gyssens. Closure properties of constraints. *Journal of the ACM*, 44(4):527–548, 1997.

Jea98. P. Jeavons. On the algebraic structure of combinatorial problems. *Theoretical Computer Science*, 200(1-2):185–204, 1998.

Jub99. L. Juban. Dichotomy theorem for generalized unique satisfiability problem. In *Proceedings of the 12th Conference on Fundamentals of Computation Theory*, pages 327–337. Springer-Verlag *Lecture Notes in Computer Science #1684*, 1999.

KK01. L. Kirousis and P. Kolaitis. The complexity of minimal satisfiability problems. In *Proceedings of the 18th Symposium on Theoretical Aspects of Computer Science*, pages 407–418. Springer-Verlag *Lecture Notes in Computer Science #2010*, 2001.

KS98. D. Kavvadias and M. Sideri. The inverse satisfiability problem. *SIAM Journal on Computing*, 28(1):152–163, 1998.

KST93. J. Köbler, U. Schöning, and J. Torán. *The Graph Isomorphism Problem: Its Structural Complexity*. Birkhäuser, 1993.

KSTW01. S. Khanna, M. Sudan, L. Trevisan, and D. Williamson. The approximability of constraint satisfaction problems. *SIAM Journal on Computing*, 30(6):1863–1920, 2001.

Pos44. E. Post. Recursively enumerable sets of integers and their decision problems. *Bulletin of the AMS*, 50:284–316, 1944.

Sch78. T. Schaefer. The complexity of satisfiability problems. In *Proceedings of the 10th ACM Symposium on Theory of Computing*, pages 216–226, 1978.

Wag87. K. Wagner. More complicated questions about maxima and minima, and some closures of NP. *Theoretical Computer Science*, 51(1–2):53–80, 1987.