Forschungsberichte
Mathematische Logik

On the Computational Complexity of some Classical Equivalence Relations on Boolean Functions

Bernd Borchert
Desh Ranjan
Frank Stephan

Bericht Nr. 18 December 1995

Universität Heidelberg
Mathematisches Institut

PostScript version for the World Wide Web

Home page: http://math.uni-heidelberg.de/logic

On the Computational Complexity of some Classical Equivalence Relations on Boolean Functions

Bernd Borchert [*] Desh Ranjan [†] Frank Stephan [‡]

Universität Heidelberg New Mexico State University Universität Heidelberg

Abstract

The paper analyzes in terms of polynomial time many-one reductions the computational complexity of several natural equivalence relations on Boolean functions which derive from replacing variables by expressions. Most of these computational problems turn out to be between co-NP and Σ_2^p.

1 Introduction

One of the best-known complexity classes is NP. This class is strongly related to the problem to decide whether a given Boolean formula is satisfiable, or in other words, whether the Boolean function described by this formula is not the function which outputs 0 for every input. So it is quite natural to look at the complexity of problems related to the Boolean functions described by given formulas, in particular to look whether two given formulas generate equivalent functions or not.

There are several different notions of equivalence. The most obvious one is of course to consider whether two formulas generate exactly the same function. In this case the formulas are called *equivalent* . All further notions are generalizations of this equivalence.

One natural generalization is to identify functions if they only differ by the names on the variables as the functions given by $x_1 \wedge x_2$ and $x_3 \wedge x_4$, respectively. Such functions are called *isomorphic*. Isomorphic functions are not always identical, for example the assignment τ with $\tau(x_1) = 0$, $\tau(x_2) = 0$, $\tau(x_3) = 1$, $\tau(x_4) = 1$ evaluates the first formula to 0 and the second to 1. We give some evidence for the hypothesis that it is more difficult to find out

[*] Mathematisches Institut, INF 294, 69120 Heidelberg, Germany, bb@math.uni-heidelberg.de.

[†] Department of Computer Science, Science Hall 159, Las Cruces, New Mexico 88011, USA, dranjan@cs.nmsu.edu.

[‡] Mathematisches Institut, INF 294, 69120 Heidelberg, Germany, fstephan@math.uni-heidelberg.de.
Supported by the Deutsche Forschungsgemeinschaft (DFG) under grant Am 60/9-1.

whether the functions generated by two given formulas are isomorphic than to find out whether they are identical. More formally, two Boolean functions are *isomorphic* if they are identical after a bijective renaming of the variables of one of the functions. Note the analogy with the same notion for graphs: two graphs are isomorphic if and only if they are identical after a bijective renaming of the nodes of one of the graphs.

Another - though less intuitive - way of identifying Boolean functions is the following. Say that two Boolean functions are *negation equivalent* if one can negate some of the variables in one of the two functions such that the resulting Boolean function is identical to the other. For example, the two Boolean functions described by the formulas $x_1 \vee x_2$ and $\neg(x_1 \wedge x_2)$ are negation equivalent (by negating both x_1 and x_2).

These two concepts can be combined: say that two Boolean functions are *congruent* if they are identical after a bijective renaming of the variables and an additional negation of some of the variables. For example, the two Boolean functions described by the formulas $x_1 \vee x_2$ and $x_3 \vee \neg x_4$ are congruent. This equivalence relation can be interpreted geometrically as congruence of the two corresponding Boolean cubes, see Section 2. The relation received attention already in the last century, see Section 2 and the second part of the References.

Isomorphism was defined by permutations. But permutations are a special kind of bijective linear mappings on the GF(2)-vectorspace $\{0,1\}^n$, namely the ones whose matrices have exactly one 1 in each line and each row. So it is natural to consider the following generalization of the Boolean isomorphism relation: say that two Boolean functions $F(x_1, \ldots x_n), G(x_1, \ldots x_n)$ are *linear equivalent* if there is a bijective linear mapping $i : \{0,1\}^n \to \{0,1\}^n$ such that $F = G \circ i$. This relation is generalized to *Boolean affine equivalence* with bijective affine mappings instead of linear ones.

An even more general equivalence notion is defined by the *cardinality*: For a Boolean function $F(x_1, \ldots, x_n)$ the cardinality $|F|$ of F is given by its share of satisfying assignments, i.e. by the number $2^{-n} \cdot |\{(x_1, \ldots, x_n) \in \{0,1\}^n : F(x_1, \ldots, x_n) = 1\}|$. Say that two Boolean functions $F(x_1, \ldots, x_n)$ and $G(x_1, \ldots, x_n)$ are *cardinality equivalent* if $|F| = |G|$.

This paper states some results about the computational complexity of recognizing the above relations if the Boolean functions are represented as formulas. For example, the computational problem corresponding to the congruence relation is the set of all pairs $\langle f, g \rangle$ such that f and g are formulas and the Boolean functions given by f and g are congruent.

The results in terms of polynomial time many-one reducibility are the following: the relations are situated between co-NP and Σ_2^p, the only exception is the cardinality equivalence which is complete for the class CP. Furthermore the problem whether two formulas are equvialent is complete for co-NP. The negation equivalence problem is reducible to the isomorphism and the congruence problem which have the same many-one complexity. These two problems are reducible to the linear and the affine equivalence relation, which have the same many-one complexity. A graphical summary is given in Figure 7.

2 The Definition of the Equivalence Relations

Let $\mathbf{C} := \{0,1\}$ be the set of the two *Boolean constants*. A Boolean function is a function $F : \mathbf{C}^n \to \mathbf{C}$ for some natural number $n \geq 0$, F will be written as $F(x_1, \ldots, x_n)$. The tuples from $\mathbf{C}^n$ are called *assignments*. We will use the usual formula notation in order to describe Boolean functions, for example the formula $x_1 \wedge \neg x_2$ describes the Boolean function $F(x_1, x_2)$ with $F(0,0) = 0, F(0,1) = 0, F(1,0) = 1, F(1,1) = 0$.

A Boolean function $F(x_1, \ldots, x_n)$ is identified with the Boolean function $G(x_1, \ldots, x_n, \ldots, x_m)$ for $m > n$ if $G(x_1, \ldots, x_n, \ldots, x_m) = F(x_1, \ldots, x_n)$ for all $(x_1, \ldots, x_n, \ldots, x_m) \in \mathbf{C}^m$, i.e. if G does not depend on the further variables. For example, the Boolean function described by $x_1 \wedge \neg x_2$ can be considered as a Boolean function $F(x_1, x_2)$ but also as a Boolean function $G(x_1, x_2, x_3, x_4)$ which is independent of x_3 and x_4.

Now the equivalence relations on Boolean functions mentioned in the introduction will formally be defined, and some basic properties of them will be stated.

Let $F(x_1, \ldots, x_n)$ be a Boolean function and let i be a function $\mathbf{C}^n \to \mathbf{C}^n$. Obviously, also $F \circ i$ is a Boolean function which we will call $F(i)$. We will only consider bijective functions $i : \mathbf{C}^n \to \mathbf{C}^n$, and some natural subsets of the set of these bijective functions are defined. First consider the set of functions $i : \mathbf{C}^n \to \mathbf{C}^n$ such that $i(x_1, \ldots, x_n) = (\pi(x_1), \ldots, \pi(x_n))$ and π is a permutation on the set $\{x_1, \ldots, x_n\}$. For obvious reasons these functions i are called *renamings*. Another type of bijective functions are the functions $i : \mathbf{C}^n \to \mathbf{C}^n$ such that $i(x_1, \ldots, x_n) = (f_1(x_1), \ldots, f_n(x_n))$ and each $f_i : \mathbf{C} \to \mathbf{C}$ is either the identity function of the negation function $n(0) = 1, n(1) = 0$. These functions are called *n-mappings*, the n stands for negation. The two concepts can be combined: let an *n-renaming* be a composition $j \circ i$ of a renaming i and an n-mapping j. Consider the set $\mathbf{C}^n$ as a vectorspace over the two-element field GF(2), addition is given by pointwise parity $\oplus$. Note that a renaming is a bijective linear function on $\mathbf{C}^n$ with the special property that in every row and every line of the representing matrix there is exactly one 1. Therefore, bijective linear functions are a generalization of renamings. Likewise, n-renamings are a special case of bijective affine functions on $\mathbf{C}^n$, namely the ones of the form $i(\vec{x}) = l(\vec{x}) \oplus \vec{c}$ such that its linear part l is represented by a matrix with the special form like above.

Definition 1 *Two Boolean functions* $F(x_1, \ldots, x_n)$ *and* $G(x_1, \ldots, x_n)$ *are said to be* isomorphic (negation equivalent, congruent, linear equivalent, affine equivalent, cardinality equivalent)*, written* $F \sim G$ *(*$F \equiv_{\neg} G, F \cong G$*,* $F \equiv_{\text{lin}} G, F \equiv_{\text{aff}} G, F \equiv_{\text{card}} G$*), if there is a renaming (n-mapping, n-renaming, bijective linear function, bijective affine function, bijective function)* $i : \mathbf{C}^n \to \mathbf{C}^n$ *such that* $F = G(i)$*.*

In other words, two Boolean functions are isomorphic if and only if they are identical modulo a renaming of the variables, they are negation equivalent if and only if they are identical modulo a negation of some variables, and they are congruent if and only if they identical modulo a renaming of the variables and an additional negation of some of them. They are linear (affine, cardinality) equivalent if they are identical after the application of a linear (affine, any) bijective function on the assignments.

Relation	*Operation*	*Typical replacement*
$\sim$	renaming	$x_i \to x_j$
$\equiv_\neg$	n-mapping	$x_i \to x_i$ $x_i \to \neg x_i$
$\cong$	n-renaming	$x_i \to x_j$ $x_i \to \neg x_j$
$\equiv_{\mathrm{lin}}$	bijective linear function	$x_i \to x_{j_1} \oplus x_{j_1} \oplus \ldots \oplus x_{j_n}$
$\equiv_{\mathrm{aff}}$	bijective affine function	$x_i \to x_{j_1} \oplus x_{j_1} \oplus \ldots \oplus x_{j_n}$ $x_i \to x_{j_1} \oplus x_{j_1} \oplus \ldots \oplus x_{j_n} \oplus 1$

Figure 1: Typical Replacements

The following Proposition 2 guarantees that Definition 1 respects the way we identified Boolean functions (it states that the introduction of dummy variables does not change the equivalence notions):

Proposition 2 *For $n \geq m$ let $F(x_1, \ldots, x_m) = F'(x_1, \ldots, x_m, \ldots, x_n)$ and $G(x_1, \ldots, x_m) = G'(x_1, \ldots, x_m, \ldots, x_n)$. Then $F \equiv_r G \iff F' \equiv_r G'$ for any of the equivalence relations $\equiv_r$ from Definition 1.*

An equivalent definition of the cardinality equivalence relation is given the following way: Let $F(x_1, \ldots, x_n)$ be given. Then the the cardinality $|F| \in [0, 1]$ of F is defined to be the number

$$2^{-n} \cdot |\{(x_1, x_2, \ldots, x_n) \in \{0,1\}^n : F(x_1, x_2, \ldots, x_n) = 1\}|.$$

It is easy to see that for two Boolean functions F, G it holds $|F| = |G|$ if and only if $F \equiv_{\mathrm{card}} G$.

An affine function $\mathbf{C}^n \to \mathbf{C}^n$ can be represented as a list of replacements $(x_1 \to i(x_1), \ldots, x_n \to i(x_n))$, where each $i(x_i)$ is of the form $x_{j_1} \oplus x_{j_2} \oplus \ldots \oplus x_{j_n}$ or $x_{j_1} \oplus x_{j_2} \oplus \ldots \oplus x_{j_n} \oplus 1$. For the more special operations this representation is even easier, for example, the list $(x_1 \to \neg x_3, x_2 \to x_1, x_3 \to \neg x_2)$ describes in an obvious way an n-renaming on $\mathbf{C}^3$. The table in Figure 1 summarizes these representations. Remember that all operations have to be bijective.

Example. Let x_1, x_2, x_3, x_4 be four different variables. Let F, G, H be the three Boolean functions described by the formulas $x_1 \wedge x_2$, $x_3 \wedge x_4$, and $x_1 \wedge \neg x_2$, respectively. Note that the three Boolean functions are pairwise different. Let $i : \mathbf{C}^4 \to \mathbf{C}^4$ be the renaming represented by the list $(x_1 \to x_3, x_2 \to x_4, x_3 \to x_1, x_4 \to x_2)$. We have $F = G(i)$ which shows that F and G are isomorphic. Let j be the n-mapping $j : \mathbf{C}^4 \to \mathbf{C}^4$ represented by the list $(x_1 \to x_1, x_2 \to \neg x_2, x_3 \to x_3, x_4 \to x_4)$. j witnesses that F and H are negation

$\sim$
0
$x \wedge y$
$x \wedge (\neg y)$ $(\neg x) \wedge y$
$(\neg x) \wedge (\neg y)$
x y
$\neg x$ $\neg y$
$x \oplus y$
$\neg(x \oplus y)$
$x \vee y$
$x \vee (\neg y)$ $(\neg x) \vee y$
$(\neg x) \vee (\neg y)$
1

$\equiv_{\neg}$
0
$x \wedge y$ $x \wedge (\neg y)$ $(\neg x) \wedge y$ $(\neg x) \wedge (\neg y)$
x $\neg x$
y $\neg y$
$x \oplus y$ $\neg(x \oplus y)$
$x \vee y$ $x \vee (\neg y)$ $(\neg x) \vee y$ $(\neg x) \vee (\neg y)$
1

$\cong$
0
$x \wedge y$ $x \wedge (\neg y)$ $(\neg x) \wedge y$ $(\neg x) \wedge (\neg y)$
x y $\neg x$ $\neg y$
$x \oplus y$ $\neg(x \oplus y)$
$x \vee y$ $x \vee (\neg y)$ $(\neg x) \vee y$ $(\neg x) \vee (\neg y)$
1

$\equiv_{\mathrm{lin}}$
0
$x \wedge y$ $x \wedge (\neg y)$ $(\neg x) \wedge y$
$(\neg x) \wedge (\neg y)$
x y $x \oplus y$
$\neg x$ $\neg y$ $\neg(x \oplus y)$
$x \vee y$
$x \vee (\neg y)$ $(\neg x) \vee y$ $(\neg x) \vee (\neg y)$
1

$\equiv_{\mathrm{aff}}$
0
$x \wedge y$ $x \wedge (\neg y)$ $(\neg x) \wedge y$ $(\neg x) \wedge (\neg y)$
x y $\neg x$ $\neg y$ $x \oplus y$ $\neg(x \oplus y)$
$x \vee y$ $x \vee (\neg y)$ $(\neg x) \vee y$ $(\neg x) \vee (\neg y)$
1

Figure 2: Partitions induced by the equivalence relations

equivalent, i.e., $F = H(j)$. The n-renaming $k = i \circ j$ witnesses that G and H are congruent, i.e., $G = H(k)$. Let $l : \mathbf{C}^2 \to \mathbf{C}^2$ be the bijectice linear function represented by the list $(x_1 \to x_1, x_2 \to x_1 \oplus x_2)$. Then it is easy to see that $F = H(l)$, therefore F and H are linear equivalent.

In Figure 2 the Boolean functions which depend on at most two variables x and y are grouped according to the five equivalence-relations $\sim$, $\equiv_{\neg}$, $\cong$, $\equiv_{\mathrm{lin}}$, and $\equiv_{\mathrm{aff}}$. The nonimplications of the following Proposition 3 can all except the last easily be verified by this table.

Proposition 3 *The relations* $\sim, \equiv_{\neg}, \cong, \equiv_{\mathrm{lin}}, \equiv_{\mathrm{aff}}, \equiv_{\mathrm{card}}$ *are equivalence relations. Figure 3 shows the inclusion relation among these equivalence relations.*

The equivalence relations above were considered already in the previous century, especially the relation of being congruent (in our terminology) received much attention since then, see the second part of the References. The best overview about the definitions and results maybe found in the papers of Harrison [Ha1964, Ha1971]. It should be remarked that there does not seem to be a standard terminology, so we felt free to choose our own symbols and names. People studying these equivalence relations were not interested in the computational complexity of the relations. Instead, they were interested in determining the number and the size of the equivalence classes when only a finite number of variables

$$\begin{tikzcd}
& F \equiv_{\mathrm{card}} G \\
& F \equiv_{\mathrm{aff}} G \arrow[u] \\
F \equiv_{\mathrm{lin}} G \arrow[ur] & F \cong G \arrow[u] \\
F \sim G \arrow[u] \arrow[ur] & F \equiv_{\neg} G \arrow[u] \\
F = G \arrow[u] \arrow[ur] &
\end{tikzcd}$$

Figure 3: Natural Implications of the form $F \equiv_1 G \implies F \equiv_2 G$.

are involved, see Figure 2. The mayor breakthrough was achieved by Polya in [Po1940] who applied his famous general combinatorial result from [Po37] to the special case of congruence of Boolean functions.

Some Motivation for Boolean Congruence. Justifying its name, the Boolean congruence relation will easily be interpreted as a geometrical congruence problem, remember that two sets of points in $\mathbb{R}^n$ are called *congruent* if there is a distance-preserving function $\mathbb{R}^n \to \mathbb{R}^n$ which maps one set of points bijectively to the other. Let a Boolean function $F(x_1, \ldots, x_n)$ be given. The *n-dimensional geometrical Boolean cube representing F* is defined to be the subset of $\mathbb{R}^n$ which consists the tuples $t = (c_1, \ldots, c_n)$, where each c_1 is either 0 or 1, such that that $F(t) = 1$, where $F(t)$ is defined to be the value $F(\tau)$ for a total assignment τ which maps x_i to $c_i \in \mathbf{C} \subset \mathbb{R}$ for each $i \in \{1, \ldots, n\}$. See Figure 4 for this definition, also Figure 5 may give some intuition. The Boolean congruence relation will also be interpreted as a graph isomorphism problem: Let the *n-dimensional graphical Boolean cube representing F* be the labeled undirected graph (N, E, λ) defined as follows. The set of nodes N consists of the 2^n different n-tuples from $\mathbf{C}^n$. The set of edges E consists of the pairs (t, t') of tuples which have Hamming distance 1, i.e. V is the set of (unordered) pairs $((c_1, \ldots, c_n), (c'_1, \ldots, c'_n))$ for which there is exactly one $i \in \{1, \ldots, n\}$ such that $c_i \neq c'_i$, and $c_j = c'_j$ for all $j \neq i$. The labeling function $\lambda : N \to \mathbf{C}$ maps a tuple $t \in \mathbf{C}^n$ to the value of $F(t)$. See Figure 5 for this construction. The following proposition is proven in a straightforward way.

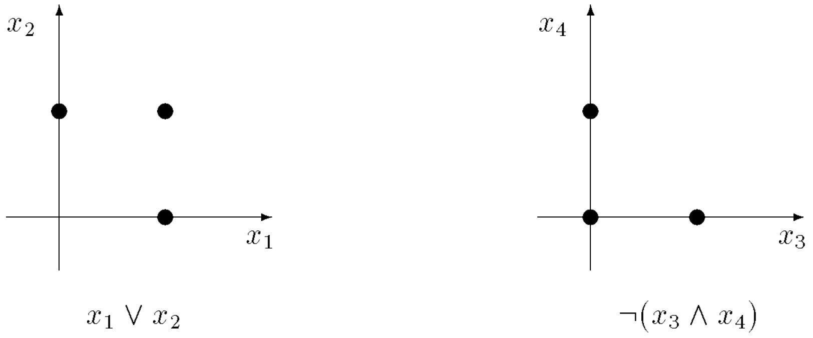

Figure 4: Two 2-dimensional geometrical Boolean cubes which are congruent

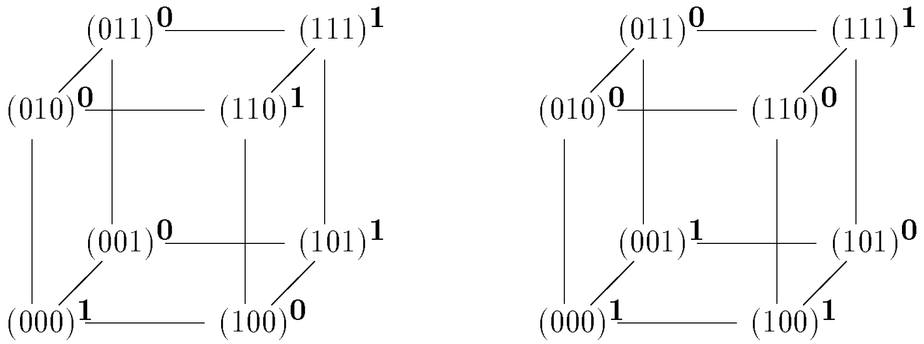

Figure 5: Two 3-dimensional graphical Boolean cubes which are isomorphic

Proposition 4 *For two Boolean functions $F = F(x_1, \ldots, x_n)$ and $G = G(x_1, \ldots, x_n)$ the following statements (a), (b), (c) are equivalent.*
(a) F and G are congruent.
(b) The n-dimensional geometrical Boolean cubes representing F and G are congruent.
(c) The n-dimensional graphical Boolean cubes representing F and G are isomorphic.

Say that an equivalence relation e is *induced by a preorder* p if $e(x,y) \iff (p(x,y)$ and $p(y,x))$. For example, the equivalence relation $\equiv_m^p$ is induced by the preorder $\leq_m^p$. Borchert and Ranjan [BR93] show that the equivalence relations $\sim$ and $\cong$ are induced by two preorders which express that one Boolean function is the (monotone) projection of the other, see [We87]. Considered as computational problems on formulas these two preorders are Σ_2^p-complete, see [BR93]. Note the analogy to the isomorphism relation on graphs: it is induced by a preorder, namely the subgraph isomorphism relation which as a computational problem is NP-complete, see [GJ78], p. 202.

3 The Complexity of the Equivalence Relations

The paper studies the complexity of the equivalence relations defined in the previous chapter when they are considered as computational problems. Karp [Ka72] introduced the polynomial-time many-one reducibility: A is reducible to B iff there is a polynomial time computable function h such that $w \in A \iff h(w) \in B$ for all w. Garey, Johnson [GJ78] and Papadimitriou [Pa94] give an overview on this and other standard notions from complexity theory. The notions *p-m-reducible* and *p-m-equivalent* are abbrevations for *polynomial time many-one reducible* and *polynomial time many-one equivalent*, respectively. The symbols for these relations will be as usual $\leq_m^p$ and $\equiv_m^p$, respectively. The completeness and hardness notion will always refer to $\leq_m^p$.

In order to make relations on Boolean functions computational problems we will represent Boolean function by formulas using the constants $0, 1$, the variables $x_1, x_2, \ldots$, 1-ary negation ($\neg$), and 2-ary conjunction ($\wedge$), disjunction ($\vee$), implication ($\rightarrow$), equivalence ($\leftrightarrow$), and parity ($\oplus$). In the obvious way each formula f describes a Boolean function $\overline{f} = \overline{f}(x_1, \ldots, x_n)$ where n is a variable index larger then the largest index of a variable occurring in f. Say that the formulas f and g are *equivalent*, written $f \equiv g$, if they describe the same Boolean function, i.e. $\overline{f} = \overline{g}$, for example $\neg(x_1 \vee x_2) \equiv \neg x_2 \wedge \neg x_1$. We will use some natural way to encode formulas as words on some fixed finite alphabet Σ. Let $\mathcal{F}$ be the set of all encoded formulas. In order to handle relations let $\langle \ldots \rangle$ be some usual pairing function on Σ^*. For the details concerning formulas and the pairing function see for example [Pa94].

For any of the equivalence relations $\equiv_r$ in Definiton 1 we will transfer the notion from the Boolean functions to the representing formulas, i.e. for two formulas f, g we write $f \equiv_r g$ if $\overline{f} \equiv_r \overline{g}$.

The uniform definitions of the computational problems we consider are the following. Figure 6 gives a summary of the terminology concerning the equivalence relations.

$$
\begin{array}{lll}
\textbf{BE} & = \{\langle f, g\rangle \mid f \equiv g\} & \text{(the Boolean equivalence problem)} \\
\textbf{BI} & = \{\langle f, g\rangle \mid f \sim g\} & \text{(the Boolean isomorphism problem)} \\
\textbf{BNE} & = \{\langle f, g\rangle \mid f \equiv_\neg g\} & \text{(the Boolean negation equivalence problem)} \\
\textbf{BC} & = \{\langle f, g\rangle \mid f \cong g\} & \text{(the Boolean congruence problem)} \\
\textbf{BLE} & = \{\langle f, g\rangle \mid f \equiv_{\mathrm{lin}} g\} & \text{(the Boolean linear equivalence problem)} \\
\textbf{BAE} & = \{\langle f, g\rangle \mid f \equiv_{\mathrm{aff}} g\} & \text{(the Boolean affine equivalence problem)} \\
\textbf{BCE} & = \{\langle f, g\rangle \mid f \equiv_{\mathrm{card}} g\} & \text{(the Boolean cardinality equality problem)}
\end{array}
$$

The problems **BE** and **BCE** can be shown to be complete for well-known classes (for the other problems we will not be able to show this). The class CP was introduced in [Wa86], the class is known to be in PSPACE and to include both NP and co-NP but it is not known to be in the Polynomial Hierachy.

Proposition 5 *(a)* **BE** *is* co-NP*-complete, (b)* **BCE** *is* CP*-complete.*

Relation Name	*Rel. Symb.*	*Def. Operation*	*Comp. Problem*
equivalence	$\equiv$	(identity function)	**BE**
isomorphism	$\sim$	renaming	**BI**
negation equivalence	$\equiv_\neg$	n-mapping	**BNE**
congruence	$\cong$	n-renaming	**BC**
linear equivalence	$\equiv_{\text{lin}}$	bijective linear function	**BLE**
affine equivalence	$\equiv_{\text{aff}}$	bijective affine function	**BAE**
cardinality equivalence	$\equiv_{\text{card}}$	any bijective function	**BCE**

Figure 6: Summarized terminology

Proof. (a) The tautology problem **TAUT** $= \{f \mid \forall\tau \in \mathbf{C}^n : f(\tau) = 1\}$ is known to be co-NP-complete. The problem **BE** is p-m-equivalent to **TAUT** since $\langle f, g\rangle \in$ **BE** $\iff$ $f \leftrightarrow g \in$ **TAUT** and $f \in$ **TAUT** $\iff \langle f, 1\rangle \in$ **BE**.

(b) The problem $A = \{f \in \mathcal{F} \mid |f| = \frac{1}{2}\}$ is known to be complete for CP. A is reducible to **BCE** by the reduction function which maps a formula f to the pair $\langle f, x_1\rangle$, note that $|x_1| = \frac{1}{2}$. **BCE** is reducible to A the following way: given two formulas $f(x_1, \ldots, x_n)$ and $g(x_1, \ldots, x_n)$, consider the following formula $h(x_1, \ldots, x_n, x_{n+1})$:

$$(x_{n+1} \wedge f(x_1, \ldots, x_n)) \vee (\neg x_{n+1} \wedge \neg g(x_1, \ldots, x_n)).$$

For $2^n \cdot |f|$ values of $(x_1, \ldots, x_n)$ the formula $f(x_1, \ldots, x_n)$ is true, thus exactly $2^n \cdot |f|$ tuples $(x_1 \ldots, x_n, x_{n+1})$ satisfy $x_{n+1} \wedge f(x_1, \ldots, x_n)$. Further $2^n \cdot (1 - |g|)$ of the tuples $(x_1, \ldots, x_n)$ satisfy $\neg g(x_1, \ldots, x_n)$ and exactly $2^n \cdot (1 - |g|)$ of the tuples satisfy $\neg x_{n+1} \wedge \neg g(x_1, \ldots, x_n)$. Since the first half of the formula is satisfied only when $x_{n+1} = 1$ while the second half is satisfied only when $x_{n+1} = 0$, in total $2^n(1 + |f| - |g|)$ of the tuples $(x_1, \ldots, x_n, x_{n+1})$ satisfy h, with other words $|h| = \frac{2^n(1+|f|-|g|)}{2^{n+1}}$. By this expression is obvious that $|f| = |g|$ if and only if $|h| = \frac{1}{2}$. Therefore, **BCE** is reducible to A by the reduction function which maps $\langle f, g\rangle$ to h. □

Proposition 6 **BI**, **BNE**, **BC**, **BLE** *and* **BAE** *are* co-NP-*hard members of* Σ_2^p.

Proof. The function $h \to \langle h, 1\rangle$ is a many-one reduction from the tautology problem **TAUT** to each **BI**, **BNE**, **BC**, **BLE** and **BAE**. The reduction is verified by the observation that any bijective function i maps tautologies to tautologies, i.e., $f \in$ **TAUT** $\iff$ $f \circ i \in$ **TAUT** $\iff$ $\langle f, 1\rangle \in$ **BI** (**BNE**, **BC**, **BLE** and **BAE** respectively).

Membership of the problems in Σ_2^p is witnessed by the following algorithm: for a formula $f(x_1, \ldots, x_n)$ first guess a representation list $(x_1 \to i(x_1), \ldots, x_n \to i(x_n))$ representing a renaming (n-mapping, n-renaming, bijective linear function, bijective affine function) i, see Figure 1. Then check for all assignments $(x_1, \ldots, x_n)$ from $\mathbf{C}^n$ if $f(x_1, \ldots, x_n)$ and $f(i(x_1), \ldots, i(x_n))$ evaluate to the same value. $\square$

On the following pages the complexities of the problems **BI**, **BNE**, **BC**, **BLE** and **BAE** will be compared with each other.

Theorem 7 **BI** *and* **BC** *are p-m-equivalent.*

Proof. One obtains a reduction from **BI** to **BC** as follows: Given two formulas f and g depending on $x_1, x_2, \ldots, x_n$, the reduction constructs two new formulas c and d in the old variables $x_1, x_2, \ldots, x_n$ and the new additional variables y_1, y_2, y_3, y_4, y_5:

$$\begin{array}{rcl} c & = & ((y_1 + y_2 + y_3 + y_4 + y_5 \geq 4) \wedge x_1 \wedge \ldots \wedge x_n) \vee (\neg y_1 \wedge \neg y_2 \wedge \neg y_3 \wedge f(x_1, \ldots, x_n)), \\ d & = & ((y_1 + y_2 + y_3 + y_4 + y_5 \geq 4) \wedge x_1 \wedge \ldots \wedge x_n) \vee (\neg y_1 \wedge \neg y_2 \wedge \neg y_3 \wedge g(x_1, \ldots, x_n)). \end{array}$$

The conjunctive normal forms of both formulas contain exactly five monomials of degree $n+4$, namely the conjunctions of all variables $x_1, x_2, \ldots, x_n$ and four of the variables y_1, y_2, y_3, y_4, y_5. Since all variables in these monomials appear in positive form, every n-renaming witnessing $c \cong d$ has to preserve all variables in the positive form and thus is already a renaming of the variables. Therefore $\langle c, d\rangle \in \mathbf{BC} \Longleftrightarrow \langle c, d\rangle \in \mathbf{BI}$.

Furthermore y_1, y_2, y_3, y_4, y_5 belong to exactly four monomials of degree $n+4$ while $x_1, x_2, \ldots, x_n$ belong to all five monomials of degree $n+4$. So it follows that each x_k has to be mapped to some other x_m and any renaming witnessing $c \sim d$ witnesses already $f \sim g$: to see this fix the values of y_1, y_2, y_3, y_4, y_5 to 0 and the so restricted functions of c and d are just f and g. Therefore $\langle f, g\rangle \in \mathbf{BI} \Longleftrightarrow \langle c, d\rangle \in \mathbf{BC}$.

A reduction from **BC** to **BI** is given the following way. Let a pair of formulas $\langle f(x_1, \ldots, x_n), g(x_1, \ldots, x_n)\rangle$ be given. Let $y_1, \ldots, y_n$ and z denote $n+1$ new variables and define c, d by

$$\begin{array}{rcl} c & = & (x_1 \oplus y_1) \wedge \ldots \wedge (x_n \oplus y_n) \wedge (z \vee f(x_1, \ldots, x_n)) \\ d & = & (x_1 \oplus y_1) \wedge \ldots \wedge (x_n \oplus y_n) \wedge (z \vee g(x_1, \ldots, x_n)) \end{array}$$

Now it is shown that $f \cong g \iff c \sim d$. If $f \cong g$ via a n-renaming i then $c \sim d$ via the following j:

$$\begin{array}{llll} j(z_1) = z & \text{and} & j(z_2) = z_2 & \\ j(x_m) = x_k & \text{and} & j(y_m) = y_k & \text{if} \quad i(x_m) = x_k \\ j(x_m) = y_k & \text{and} & j(y_m) = x_k & \text{if} \quad i(x_m) = \neg x_k \end{array}$$

So the main idea is that the y_k represent $\neg x_k$ and so the negation is removed by introducing a new variable. The form of c and d enforces, that $c(x_1, \ldots, x_n, y_1, \ldots, y_n, z) = 1$ only if $x_k = \neg y_k$ for $k = 1, \ldots, n$ and on the other hand, $c(x_1, \ldots, x_n, y_1, \ldots, y_n, z) = f(x_1, \ldots, x_n)$ if $z = 0$ and $x_k = \neg y_k$ for $k = 1, \ldots, n$. ¿From this thoughts it follows that $f \cong g \implies c \sim d$.

For the other way around assume that $c \sim d$ via j. Call v and w c-incompatible iff there is no satisfying assignment for c with $v = 1$ and $w = 1$. One can see that v and w are c-incompatible iff $v = x_k$ and $w = y_k$ for some k; the same holds for the corresponding notion of d-incompatibility. If $c \sim d$ via j then j has to map any pair of c-incompatible variables to a pair of d-incompatible variables: so for each k there is an m such that either $j(x_k) = x_m$ and $j(y_k) = y_m$ or $j(x_k) = y_m$ and $j(y_k) = x_m$. So one immediately obtains the n-renaming

$$i(x_k) = \begin{cases} x_m & \text{if } j(x_k) = x_m; \\ \neg x_m & \text{if } j(x_k) = y_m. \end{cases}$$

¿From $f(x_1, \ldots, x_n) = c(x_1, \ldots, x_n, \neg x_1, \ldots, \neg x_n, 0)$ and the corresponding equality for g versus d it follows that i witnesses $f \cong g$. □

Theorem 8 **BNE** *is p-m-reducible to* **BI**.

Proof. The following p-m-reduction from **BNE** to **BI** is similar to the one from **BC** to **BI**. Let a pair of formulas $\langle f(x_1, \ldots, x_n), g(x_1, \ldots, x_n) \rangle$ be given. Choose $3n + 1$ different variables $y_1, y_1', z_1, \ldots, y_n, y_n', z_n, z$ and construct the pair of formulas $\langle c, d \rangle$ where

$$\begin{aligned} c \;=\; & (\neg z_1)(y_1 \vee y_1') \vee z_1(\neg z_2)(y_2 \vee y_2') \vee \ldots z_1 z_2 \ldots (\neg z_n)(y_n \vee y_n') \vee \\ & z_1 z_2 \ldots z_n (y_1 \oplus y_1') \wedge \ldots \wedge (y_n \oplus y_n') \wedge (z \vee f(y_1, \ldots, y_n)) \end{aligned}$$

and d similarly depends on g. It holds that $f \equiv_\neg g \iff c \sim d$. The verification is done the same way like for the reduction from **BC** to **BI**, where here the variables z_i guarantee that y_i is mapped only to y_i or y_i'. □

Consider $\mathbf{C}^n$ to be a GF(2) vector space. Let $[x_k]$ denote the linear subspace $\{(a_1, \ldots, a_n) : (\forall h \neq k)\,[a_h = 0]\}$ and $[x_m, x_k]$ the subspace generated by $[x_m]$ and $[x_k]$ and so on. Furthermore $\mathbf{u}[x_m, x_k]$ denotes the projection (a_m, a_k) on the given variables. Note that if i is a Boolean linear isomorphism, then i maps subspaces to other linear subspaces of the same dimension and cardinality.

Theorem 9 **BAE** *is p-m-equivalent to* **BLE**.

Proof. The m-reduction from **BAE** to **BLE** is $\langle f, g \rangle \to \langle x \wedge f, x \wedge g \rangle$ where x is a variable that neither occurs in f nor in g. It remains to be shown that the $\langle f, g \rangle \in \mathbf{BAE}$ iff $\langle x \wedge f, x \wedge g \rangle \in \mathbf{BLE}$:

$f \equiv_{\text{aff}} g \implies x \wedge f \equiv_{\text{lin}} x \wedge g$:

There is an affine mapping i witnessing $f = g \circ i$. Let the y_m and z_m denote variables other than x and let

$$j(y_m) = \begin{cases} z_1 \oplus \ldots \oplus z_k & \text{if } i(y_m) = z_1 \oplus \ldots \oplus z_k; \\ x \oplus z_1 \oplus \ldots \oplus z_k & \text{if } i(y_m) = 1 \oplus z_1 \oplus \ldots \oplus z_k. \end{cases}$$

For $x = 1$, $i(y_m) = j(y_m)$ and therefore $(x \wedge f)(1, y_1, \ldots, y_n) = f(y_1, \ldots, y_n) = g(i(y_1), \ldots, i(y_n)) = (x \wedge g)(1, j(y_1), \ldots, j(y_n))$. For $x = 0$, $(x \wedge f)(0, y_1, \ldots, y_n) = 0 = (x \wedge g)(0, j(y_1), \ldots, j(y_n))$ holds independently of the values $j(y_1), \ldots, j(y_n)$. Thus $(x \wedge f) \circ j = x \wedge g$.

Since i^{-1} can be transformed in the same way to j^{-1}, the new linear mapping j is invertable and $x \wedge f \equiv_{\text{aff}} x \wedge g$.

$x \wedge f \equiv_{\text{lin}} x \wedge g \implies f \equiv_{\text{aff}} g$:

Let j be a linear mapping witnessing $x \wedge f \equiv_{\text{lin}} x \wedge g$. Let the y_m and z_m denote variables other than x and let

$$i(y_m) = \begin{cases} z_1 \oplus \ldots \oplus z_k & \text{if } i(y_m) = z_1 \oplus \ldots \oplus z_k; \\ 1 \oplus z_1 \oplus \ldots \oplus z_k & \text{if } i(y_m) = x \oplus z_1 \oplus \ldots \oplus z_k. \end{cases}$$

Note that $f(y_1, \ldots, y_n) = (x \wedge f)(1, y_1, \ldots, y_n) = (x \wedge g)(1, j(y_1), \ldots, j(y_n)) = g(i(y_1), \ldots, i(y_n))$, so the equivalence follows. Again i is invertable since j is invertable.

For the other way round consider given functions f, g which have the variables $x_1, \ldots, x_n$. Now new variables $y_1, \ldots, y_{n+1}$ and $z_1, \ldots, z_{n+2}$ are generated; further let X be the n-dimensional Boolean vectorspace generated by the basis $x_1, \ldots, x_n$, Y be the $n+1$-dimensional vectorspace generated by $y_1, \ldots, y_{n+1}$ and Z be the $n+2$-dimensional vectorspace generated by $z_1, \ldots, z_{n+2}$. X, Y, Z are identified with the corresponding subsets of the vectorspace $X \oplus Y \oplus Z$. Let $\mathbf{0}$ denote the shared 0-vector of all four vectorspaces. Now the functions $f : X \to \{0, 1\}$ is extended to a function $F : X \oplus Y \oplus Z \to \{0, 1\}$ as follows:

$$F(\alpha) = \begin{cases} f(\alpha) & \text{if } \alpha \in X; \\ f(\mathbf{0}) & \text{if } \alpha \in Y \cup Z; \\ \neg f(\mathbf{0}) & \text{otherwise, i.e., } \alpha \in W \text{ where } W = X \oplus Y \oplus Z - X \cup Y \cup Z. \end{cases}$$

Similarly g is extended to G. Now it has to be shown that the mapping $\langle f, g \rangle \to \langle F, G \rangle$ is an p-m-reduction from **BLE** to **BAE**. Obviously the mapping is polynomial time computable.

$f \equiv_{\text{lin}} g \implies F \equiv_{\text{aff}} G$:

Let i witness that $f \equiv_{\text{lin}} g$. Since i is linear, $f(\mathbf{0}) = g(\mathbf{0})$. Given $\alpha \in X$, $\beta \in Y \oplus Z$, let $j(\alpha \oplus \beta) = i(\alpha) \oplus \beta$. j is linear and thus affine. Since i is bijective, so is j. j maps X to X, therefore for all $\alpha \in X$, $G(j(\alpha)) = G(i(\alpha)) = g(i(\alpha)) = f(\alpha) = F(\alpha)$. If $\alpha \in Y \cup Z$ then $j(\alpha) = \alpha$ and $F(\alpha) = f(\mathbf{0}) = g(\mathbf{0}) = G(\alpha)$. Since j maps $X \cup Y \cup Z$ to $X \cup Y \cup Z$, j also maps W to W. So for $\alpha \in W$, $F(\alpha) = \neg f(\mathbf{0})$ and $G(j(\alpha)) = \neg g(\mathbf{0}) = \neg f(\mathbf{0})$. j witnesses that $F \equiv_{\text{aff}} G$.

$F \equiv_{\text{aff}} G \implies f \equiv_{\text{lin}} g$:

Let i witness that $F \equiv_{\text{aff}} G$. F takes the value $f(\mathbf{0})$ only on $X \cup Y \cup Z$, that means on at most $2^n + 2^{n+1} + 2^{n+2}$ arguments while it takes the value $\neg f(\mathbf{0})$ at least on W. Since $|W| > |X \cup Y \cup Z|$ but $|F| = |G|$ it follows that $f(\mathbf{0}) = g(\mathbf{0})$.

Z is a $n+2$-dimensional linear subspace where F takes the value $f(\mathbf{0})$. So G must take on $i(Z)$ again the value $f(\mathbf{0}) = g(\mathbf{0})$ and since $i(Z)$ is an affine $n+2$-dimensional space, $i(Z) = Z$. Since Y intersects Z in one point, $i(Y)$ intersects $i(Z)$ also in one point. From this information it can be deduced that $i(Y) = Y$. Since $\mathbf{0} \in Y \cup Z$, $i(\mathbf{0}) \in i(Y) \cup i(Z)$ and it follows that $i(\mathbf{0}) = \mathbf{0}$, i.e., i is linear.

Let $U = X \cap i^{-1}(X)$. U is a linear subspace of X. The restriction of i to U can be extended to a linear bijective function $j : X \to X$. Now assume by the way of contradiction that $f(\alpha) \neq g(j(\alpha))$ for some $\alpha \in X$. Then $\alpha \notin U$. Thus $i(\alpha) \in W$ and $f(\alpha) = G(i(\alpha)) = \neg f(\mathbf{0})$. Further there is γ with $i(\gamma) = j(\alpha)$. Since $j(\alpha) \notin i(U)$, $\gamma \in W$ and $g(j(\alpha)) = G(i(\gamma)) = F(\gamma) = \neg f(\mathbf{0})$. So such an α does not exist and j witnesses $f \equiv_{\mathrm{lin}} g$.

Therefore **BAE** $\equiv^p_m$ **BLE**. □

Theorem 10 **BI** *is p-m-reducible to* **BLE**.

Proof. Consider the formulas f, g with the variables $x_1, \ldots, x_n$. Now introduce for each $k = 1, \ldots, n$ the new variables $y_{k,0}, \ldots, y_{k,n}$. Let c derive from f by the formula

$$c(\mathbf{u}) = \begin{cases} f(\mathbf{u}) & \text{if } \mathbf{u} \in [x_1, \ldots, x_n]; \\ 0 & \text{if } \mathbf{u} \in [x_k, y_{k,0}, \ldots, y_{k,n}] - [x_k] \text{ for some } k; \\ 1 & \text{otherwise;} \end{cases}$$

and let d similarly derive from g. Now one shows that $f \sim g$ iff $c \equiv_{\mathrm{lin}} d$. If i witnesses $f \sim g$ then i can be extended such that i witnesses $c \sim d$ which immediately implies $c \equiv_{\mathrm{lin}} d$: For each k find the m with $i(x_k) = x_m$ and let $i(y_{k,h}) = y_{m,h}$. The verification that this works is left to the reader.

So the interesting case is to translate a Boolean linear equivalence j witnessing $c \equiv_{\mathrm{lin}} d$ into a renaming witnessing $f \sim g$. c takes on any subspace $[x_k, y_{k,0}, \ldots, y_{k,n}]$ at most two 1s: they appear on the subspace $[x_k]$ which is the intersection with $[x_1, \ldots, x_n]$. Thus d takes on the subspace $[j(x_k), j(y_{k,0}), \ldots, j(y_{k,n})]$ at most two 1s. If this subset contains non-zero vectors in at least two subspaces of the form $[x_m, y_{m,0}, \ldots, y_{m,n}]$, then d maps at most $2^{n+1} + 2^n + 1 < 2^{n+2} - 2$ (w.l.o.g. $n > 1$) of its elements to 0: There are at most $2^{n+1} + 1$ vectors in any nontrivial union of orthogonal generating subspaces and at most 2^n vectors in $[x_1, \ldots, x_n]$. From this contradiction it follows that there is an m such that j maps $[x_k, y_{k,0}, \ldots, y_{k,n}]$ to $[x_m, y_{m,0}, \ldots, y_{m,n}]$. So it is suitable to define $i(x_k) = x_m$ for this m. For any assignment $(a_1, \ldots, a_n)$ for the variables $x_1, \ldots, x_n$ let

$$A(a_1, \ldots, a_n) = \{\mathbf{u} : (\forall k)\, [\mathbf{u}[x_k, y_{k,0}, \ldots, y_{k,n}] \neq \mathbf{0} \iff a_k = 1]\,\}$$

and $(b_1, \ldots, b_n)$ be the tuple which satisfies that $a_k = b_m$ whenever $i(x_k) = x_m$. Now i witnesses $f \sim g$ iff $f(a_1, \ldots, a_n) = g(b_1, \ldots, b_n)$. Note that by the linearity of j, j maps each set $A(a_1, \ldots, a_n)$ to $A(b_1, \ldots, b_n)$. Now the verification needs the following case-distinction:

$a_1 + \ldots + a_n = 0$: $f(a_1, \ldots, a_n) = 1 \iff c(\mathbf{0}) = 1 \iff d(\mathbf{0}) = 1 \iff g(b_1, \ldots, b_n) = 1$;

$a_1 + \ldots + a_n = 1$: $f(a_1, \ldots, a_n) = 1 \iff$
$c(\mathbf{u}) = 1$ for some $\mathbf{u} \in A(a_1, \ldots, a_n) \iff$
$d(\mathbf{u}) = 1$ for some $\mathbf{u} \in A(b_1, \ldots, b_n) \iff$
$g(b_1, \ldots, b_n) = 1$;

$a_1 + \ldots + a_n > 1$: $f(a_1, \ldots, a_n) = 1 \iff$
$c(\mathbf{u}) = 1$ for all $\mathbf{u} \in A(a_1, \ldots, a_n) \iff$
$d(\mathbf{u}) = 1$ for all $\mathbf{u} \in A(b_1, \ldots, b_n) \iff$
$g(b_1, \ldots, b_n) = 1$.

The existential and universal quantification are due to the fact, that if $a_1 + \ldots + a_n = 1$ then $c(\mathbf{u}) = 0$ for all $\mathbf{u} \in A(a_1, \ldots, a_n)$ except the one $\mathbf{u}$ which is also in $[x_1, \ldots, x_n]$ and that if $a_1 + \ldots + a_n > 1$ then $c(\mathbf{u}) = 1$ for all $\mathbf{u} \in A(a_1, \ldots, a_n)$ except the one $\mathbf{u}$ which is also in $[x_1, \ldots, x_n]$. The $\mathbf{u} \in A(a_1, \ldots, a_n) \cap [x_1, \ldots, x_n]$ takes the value $f(a_1, \ldots, a_n)$. The rest of the equivalence is due to the fact the the number of 1s which c takes on $A(a_1, \ldots, a_n)$ is equal to the number of 1s which d takes on $A(b_1, \ldots, b_n)$. □

Blass and Gurevich [BG82] defined the problem **USAT** $= \{f \in \mathcal{F} \mid$ exactly one of the 2^n finite assignments to the n variables which occur in f evaluates f to $1\}$ and showed that it is co-NP–hard. Chang and Kadin [CK90] showed that **USAT** is not in co-NP unless the Polynomial Hierarchy collapses. The following construction is a p-m-reduction from **USAT** to **BNE** (and also to **BC**):

$$f(x_1, \ldots, x_n) \to \langle f(x_1, \ldots, x_n), x_1 \wedge \ldots \wedge x_n \rangle.$$

Proposition 11 USAT *is p-m-reducible to* **BNE***.*

Corollary 12 *If the Polynomial Hierarchy does not collapse then* **BNE***,* **BI***,* **BC***,* **BLE** *and* **BAE** *are not in* co-NP*.*

For the definition of the Graph Isomorphism problem **GI** see [KST93]. R. Chang [BR93, Prop. 2] obtained the following result:

Proposition 13 GI *is p-m-reducible to* **BI***.*

Proof. Let for a graph $G = (V, E)$ the formula h_G be defined as follows: for every vertex $i \in V$ in G choose a different variable v_i, and let $h_G := \bigvee_{(i,j) \in E}(v_i \wedge v_j)$. Now, it is not difficult to see that G_1 and G_2 are isomorphic if and only if the two Boolean functions described by h_{G_1} and h_{G_2} are isomorphic. □

Corollary 14 BI*,* **BC***,* **BLE** *and* **BAE** *are not in* co-NP *unless* **GI** *is in* co-NP*.*

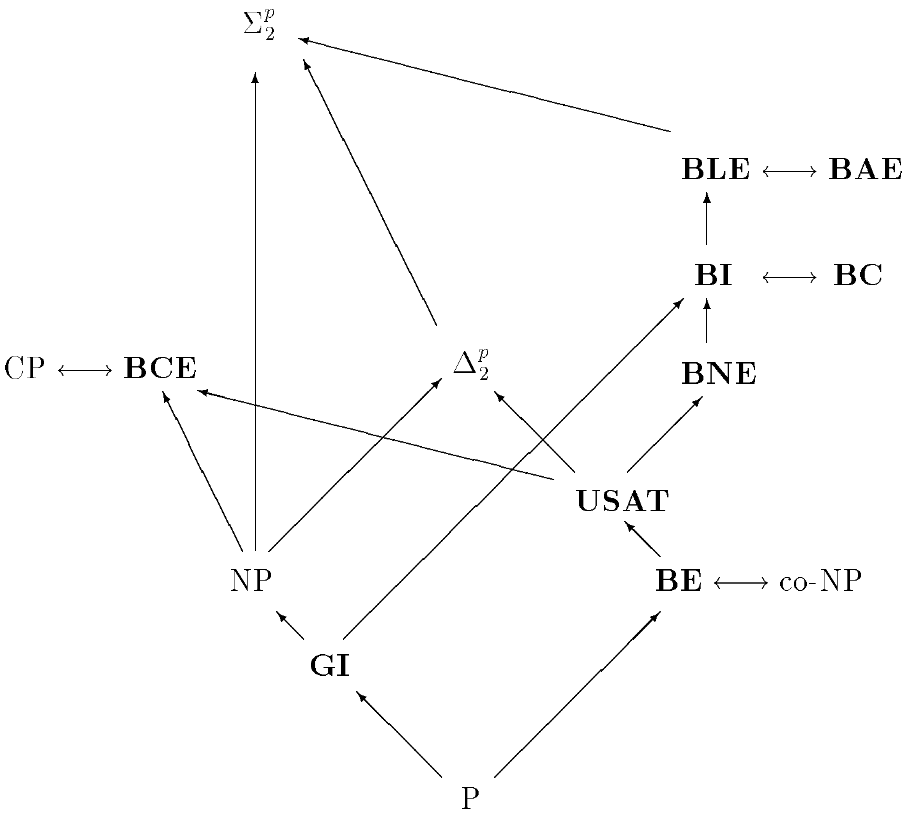

Figure 7: Summary of the results

The stated results are summarized in Figure 7 where an arrow denotes the proven existence of a p-m-reduction between two problems (in the case of a class consider a complete problem).

The automorphism problems which correspond to the equivalence relations defined in this paper are studied in a forthcoming paper of B. Borchert and A. Lozano, it is shown there that similar results holds like in the case of Graph Isomorphism versus Graph Automorphism [KST93], e.g. the automorphism problems are p-m-reducible to the corresponding isomorphism problems.

4 Conclusion

In this paper the following equivalence relations on Boolean functions were considered: Boolean isomorphism, Boolean negation equivalence, Boolean congruence, Boolean linear equivalence, and Boolean affine equivalence. The following results on their polynomial time many-one complexity are shown: They all are situated above co-NP and – besides the last relation – below Σ_2^p. There is strong evidence that none of the problems is in co-NP.

Unfortunately, we could not give evidence that the three problems are not Σ_2^p-complete, and neither we could find out whether they are in the Turing closure Δ_2^p of NP.

Our conjecture that Boolean Isomorphism may have intermediate complexity between the first and the second level of the Polynomial Time Hierarchy, like Graph Isomorphism may have intermediate complexity between the bottom level and the first level of the Polynomial Time Hierarchy.

Acknowledgements

The authors are grateful to Jin-yi Cai, Antoni Lozano, and Thomas Thierauf for discussions about the subject .

References

[BG82] A. Blass, Y. Gurevich. *On the unique satisfiability problem*, Information and Control **55**, 1982, pp. 80-88

[BR93] B. Borchert, D. Ranjan. *The Subfunction Relations are Σ_2^p–complete*, Technical Report MPI-I-93-121, MPI Saarbrücken, 1993

[BL95] B. Borchert, A. Lozano. *Succinct Circuit Representations and Leaf Languages are Basically the same Concept*, Forschungsbericht Nr. 17, Universität Heidelberg, 1995

[CK90] R. Chang, J. Kadin. *On the Structure of Uniquely Satisfiable Formulas*, Technical Report 90-1124, Cornell University, 1990

[GJ78] M. R. Garey, D. S. Johnson. *Computers and Intractability*, Freeman, San Francisco, 1978

[Ka72] R. Karp. *Reducibility among combinatorial problems*, in: R. E. Miller and J. W. Thatcher, eds., Complexity of Computer Computation, Plenum, New York, 1972, pp. 85–103

[KST93] J. Köbler, U. Schöning, J. Toran. *The Graph Isomorphism Problem: its Structural Complexity*, Birkhäuser Verlag, 1993

[Od89] P. Odifreddi. *Classical Recursion Theory*, North-Holland, Amsterdam, New York, 1989

[Pa94] C. Papadimitriou. *Computational Complexity*, Addison-Wesley, Reading, Massachusetts, 1994

[Po37] G. Polya. *Kombinatorische Anzahlbestimmungen für Gruppen, Graphen und chemische Verbindungen*, Acta Mathematica **68**, 1937, pp. 145–254

[Wa86] K. Wagner. *The complexity of combinatorial problems with succinct input representation*, Acta Informatica **23**, 1986, pp. 131–147

[We87] I. Wegener. *The Complexity of Boolean Functions*, Teubner, Stuttgart, 1987

Some references for the Boolean isomorphism relation $\sim$, Boolean negation equivalence relation $\equiv_\neg$, Boolean congruence relation $\cong$, Boolean linear equivalence relation $\equiv_{\text{lin}}$, and the Boolean affine equivalence relation $\equiv_{\text{aff}}$ in chronological order:

[Je1874] W. S. Jevons. *The Principles of Sciences*, London, 1874 (in the second edition, London and New York, 1892, see pp. 134–146)

[Cl1876] W. K. Clifford. *On the types of compound statement involving four classes*, Memoirs of the Literary and Philosophical Society of Manchester, Volume 16, No. 7, 1876-77, pp. 88-101. Also in W. K. Clifford. *Mathematical Papers*, London, 1882, pp. 1–16

[La1883] C. Ladd. *Studies in Logic*, by members of the John Hopkins University, C. S. Peirce (ed.), Boston, 1883, pp.1–69 (see p. 67)

[Sch1890] E. Schröder. *Algebra der Logik I*, 1890, (see Anhang 6)

[Na1893] A. Nagy. *Über Beziehungen zwischen logischen Grössen*, Monatshefte für Mathematik und Physik **4**, 1893, pp. 147-153

[Na1894] A. Nagy. *Über das Jevons-Clifford'sche Problem*, Monatshefte für Mathematik und Physik **5**, 1894, pp. 331–345

[Po1940] G. Pólya. *Sur les types des propositions composées*, Journal of Symbolic Logic **5**, No. 3, 1940

[Sh1949] C. E. Shannon. *The synthesis of two terminal switching circuits*, Bell Systems Technical Journal **28**, 1949, pp. 59–98 (see pp. 91-97)

[Si1952] T. Singer. *The theory of counting techniques*, ACM Pittsburg Meeting, 1952 pp. 287-291

[Sl1953] D. Slepian. *On the number of symmetry types of Boolean functions of n variables*, Canadian Journal of Mathematics **5**, 1953, pp.185–193

[Ne1958] E. I. Nechiporuk. *On the synthesis of networks using linear transformations of argument variables* (in Russian), Doklady Akad. Nauk **123**, No. 4, 1958, pp. 610–612; English translation appeared in Automation Express, 1959, pp. 12-13,

[El59] B. Elspas. *The theory of autonomous linear sequential networks*, IRE CT-6, 1959, pp. 45–60

[Go1959] S. W. Golomb. *On the classification of Boolean functions*, IRE Transactions on Circuit Theory, 1959, pp. 176–185

[Sl1960] D. Slepian. *Some further theory of group codes*, Bell Systems Technical Journal **39**, No. 5, 1960, pp. 1219–1252

[Go1961] S. W. Golomb. *A mathematical theory of discrete classification*, Proceedings of the Fourth London Symposium on Information Theory, London, 1961, pp. 404–425

[Ni1961] I. Ninomiya. *A study of the structures of Boolean functions and its application to the synthesis of switching circuits*, Memoirs of the Faculty of Engineering, Nagoya University, vol. 13, No. 2, 1961; see also Review R63-24, IEEE EC-12,p.152, 1963.

[He1961] L. Hellerman. *Equivalence Classes of Logical Functions*, IBM Technical Report No. 00.819, Poughkeepsie, New York, 1961

[He1963] L. Hellerman. *Equivalence Classes of Logical Functions*, Computer Design **1**, No. 1, 1963, pp. 34–37.

[Le1963] R. J. Lechner. *Transformations among switching function canonical forms*, IEEE Transactions on Electronic Computers, EC-12, 1963, pp.129-130

[Ha1963a] M. A. Harrison. *The number of transitivity set of Boolean functions*, Journal of SIAM **11**, No. 3, 1963, pp. 806–828

[Ha1963b] M. A. Harrison. *On the number of (n,k) switching networks*, Journal Franklin Institute **276**, 1963, pp. 313–327

[Ha1964] M. A. Harrison. *On the classification of Boolean functions by the general linear and affine groups*, Journal of SIAM **12**, No. 2, 1964, pp. 285–299

[Ha1965] M. A. Harrison. *Introduction to Switching and Automata Theory*, McGraw-Hill, New York, 1965

[Gi66] A. Gill. *Graphs of affine transformations, with applications to sequential circuits*, 7th Symposium on Switching and Automata Theory, 1966, pp. 127–135

[Ha1966] M. A. Harrison. *On asymptotic estimates in switching and automata theory*, Journal of the ACM **13**, 1966, pp. 151-157

[Le1967] R. J. Lechner. *A correspondence between equivalence classes of switching functions and group codes*, IEEE Transactions on Electronic Computers, EC-16, No. 5, 1967, pp. 621–624

[Le1970] R. J. Lechner. *A transform approch to logic design*, IEEE Transactions on Computers, C-19, No. 7, 1970, pp. 627–639

[Ha1971] M. A. Harrison. *Counting theorems and their applications to classification of switching functions*, Recent Developments in Switching Theory, A. Mukhopadyay (ed.), Academic Press, 1971

[Le1971] R. J. Lechner. *Harmonic analysis of switching functions*, Recent Developments in Switching Theory, A. Mukhopadyay (ed.), Academic Press, 1971

[Ke1974] P. Kerntopf. *On Boolean functions having maximal number of subfunction classes*, 15th Annual IEEE Symposium on Foundations of Computer Science (FOCS), 1974, pp. 205–211

[PR1987] G. Polya, R. C. Read. *Combinatorial Enumeration of Groups, Graphs, and Chemical Compounds*, Springer Verlag, New York, 1987 (see page 134)

[CK1991] P. Clote, E. Kranakis. *Boolean functions, invariance groups, and parallel complexity*, SIAM Journal on Computing **20**, No. 3, 1991, pp. 553-590