A Unifying Formal Approach to Importance Values in Boolean Functions

Citation for published version (APA):

DOI:
10.48550/arXiv.2305.08103

Document status and date:
Published: 14/05/2023

Document Version:
Publisher’s PDF, also known as Version of Record (includes final page, issue and volume numbers)

Please check the document version of this publication:
- A submitted manuscript is the version of the article upon submission and before peer-review. There can be important differences between the submitted version and the official published version of record. People interested in the research are advised to contact the author for the final version of the publication, or visit the DOI to the publisher’s website.
- The final author version and the galley proof are versions of the publication after peer review.
- The final published version features the final layout of the paper including the volume, issue and page numbers.

Link to publication

General rights
Copyright and moral rights for the publications made accessible in the public portal are retained by the authors and/or other copyright owners and it is a condition of accessing publications that users recognise and abide by the legal requirements associated with these rights.

- Users may download and print one copy of any publication from the public portal for the purpose of private study or research.
- You may not further distribute the material or use it for any profit-making activity or commercial gain
- You may freely distribute the URL identifying the publication in the public portal.

If the publication is distributed under the terms of Article 25fa of the Dutch Copyright Act, indicated by the “Taverne” license above, please follow below link for the End User Agreement:
www.tue.nl/taverne

Take down policy
If you believe that this document breaches copyright please contact us at:
openaccess@tue.nl
providing details and we will investigate your claim.

Download date: 30. Sep. 2023
A Unifying Formal Approach to Importance Values in Boolean Functions

Hans Harder$^{1,2}$, Simon Jantsch$^2$, Christel Baier$^{2,4}$ and Clemens Dubslaff$^{3,4}$

$^1$Paderborn University, Paderborn, Germany
$^2$Dresden University of Technology, Dresden, Germany
$^3$Eindhoven University of Technology, Eindhoven, The Netherlands
$^4$Centre for Tactile Internet with Human-in-the-Loop (CeTI), Dresden, Germany

hans.harder@uni-paderborn.de, \{simon.jantsch, christel.baier\}@tu-dresden.de, c.dubslaff@tue.nl

Abstract

Boolean functions and their representation through logics, circuits, machine learning classifiers, or binary decision diagrams (BDDs) play a central role in the design and analysis of computing systems. Quantifying the relative impact of variables on the truth value by means of importance values can provide useful insights to steer system design and debugging. In this paper, we introduce a uniform framework for reasoning about such values, relying on a generic notion of importance value functions (IVFs). The class of IVFs is defined by axioms motivated from several notions of importance values introduced in the literature, including Ben-Or and Linial’s influence and Checkler, Halpern, and Kupferman’s notion of responsibility and blame. We establish a connection between IVFs and game-theoretic concepts such as Shapley and Banzhaf values, both of which measure the impact of players on outcomes in cooperative games. Exploiting BDD-based symbolic methods and projected model counting, we devise and evaluate practical computation schemes for IVFs.

1 Introduction

Boolean functions arise in many areas of computer science and mathematics, e.g., in circuit design, formal logics, coding theory, artificial intelligence, machine learning, and system analysis [Crama and Hammer, 2011a; O’Donnell, 2014]. When modeling and analyzing systems through Boolean functions, many design decisions are affected by the relevance of variables for the outcome of the function. Examples include noise-reduction components for important input variables to increase reliability of circuits, prioritizing important variables in decision-making of protocols, or the order of variables in BDDs [Bryant, 1992; Bartlett and Andrews, 2001]. Many ideas to quantify such notions of importance of variables in Boolean functions have since been considered in the literature. To mention a few, influence [Ben-Or and Linial, 1985] is used to determine power of actors in voting schemes, [Hammer et al., 2000] devised measures based on how constant a function becomes depending on variable assignments, blame [Chockler and Halpern, 2004] quantifies the average responsibility [Chockler et al., 2008] of input variables on the outcome of circuits or on causal reasoning, and the Jeroslow-Wang value [Jeroslow and Wang, 1990] quantifies importance of variables in CNFs to derive splitting rules for SAT-solvers [Hooker and Vinay, 1995]. Closely related are notions of impact in cooperative games, e.g., through the Shapley value [Shapley, 1953] or the Banzhaf value [Banzhaf, 1965].

Although some of the aforementioned concepts are of quite different nature and serve different purposes, they share some common ideas. This raises the question of what characteristics importance values have and how the notions of the literature relate. The motivation of this paper is to advance the understanding of importance values, independent of concrete applications. For this purpose, we introduce a generic axiomatic framework that constitutes the class of importance value functions (IVFs). Our axioms are motivated by properties one would intuitively expect from IVFs, e.g., that independent variables have no importance or that permutations do not change importance values. We show basic relationships within and between IVFs and provide new insights for existing and new importance measures. By connecting Boolean functions and cooperative games through cooperative game mappings (CGMs) and using Shapley and Banzhaf values, we show how to generically derive new IVFs. All aforementioned notions of importance values from the literature satisfy our IVF axioms, showing that we provide a unifying framework for all these notions, including CGM-derived ones.

Most notions of importance are known to be computationally hard, e.g., computing influence or the Shapley value is \#P-complete [Traxler, 2009; Faigle and Kern, 1992; Deng and Papadimitriou, 1994]. We address computational aspects by devising practical computation schemes for IVFs using projected model counting [Aziz et al., 2015] and BDDs.

Contributions and outline. In summary, our main contribution is an axiomatic definition of IVFs for variables in Boolean functions (Section 3), covering notions of importance from the literature (Sections 4.1 and 4.2). Moreover, we derive novel IVFs by linking Boolean functions with cooperative games and related values (Section 4.3). Finally, we provide practical computation schemes for IVFs (Section 5).
Supplemental material. This is a preprint of the paper accepted at the 32nd International Joint Conferences on Artificial Intelligence (IJCAI’23). It includes proofs and other additional material in the appendix. An implementation of the computing schemes for IVFs can be found at https://github.com/graps1/impmeas.

2 Preliminaries

Let $X = \{x, y, z, \ldots\}$ be a finite set of $n = |X|$ variables, which we assume to be fixed throughout the paper.

Assignments. An assignment over $U \subseteq X$ is a function $u : U \rightarrow \{0, 1\}$, written in the form $u = x/0; y/1; \ldots$. We denote assignments by bold lower-case letters and their domains by corresponding upper-case letters. If $u$ and $v$ have disjoint domains, we write their concatenation as $w = uv$ with $W = V \cup U$ and $w(x) = u(x)$ if $x \in U$ and $w(x) = v(x)$ if $x \in V$. The restriction of $w$ to a domain $S \subseteq U$ is denoted by $u_{S}$. For a permutation $\sigma$ of $X$, we define $\sigma u$ as the assignment over $\sigma(U)$ with $\sigma u(x) = u(\sigma^{-1}(x))$.

Boolean functions. We call $f, g, h, \ldots : \{0, 1\}^{X} \rightarrow \{0, 1\}$ Boolean functions, collected in a set $\mathbb{B}(X)$. We write $g = x$ if $g$ is the indicator function of $x$, and we write $y$ for negation, $f \lor g$ for disjunction, $f \wedge g$ for conjunction and $f \oplus g$ for exclusive disjunction. The cofactor of $f$ w.r.t. an assignment $u$ is the function $f_{u}$ that always sets variables in $V$ to the value given by $u$, and is defined as $f_{u}(v) = f(v, u_{V\setminus V})$. The Shannon decomposition of $f$ w.r.t. variable $x$ is a decomposition rule stating that $f = x f_{x/1} \lor \overline{x} f_{x/0}$ holds, where $f_{x/1}$ and $f_{x/0}$ are the positive and negative cofactor of $f$ w.r.t. $x$. For a Boolean function $f$, variable $x$, and Boolean function or variable $s$, let $f[x/s] = s f_{x/1} \lor \overline{s} f_{x/0}$ be the function that replaces $x$ by $s$. For example, if $f = y \lor x z$, then $f_{x/1} = y \lor z$ and $f_{x/0} = y$. Moreover, for $s = x_{1} x_{2}$, we have

$$f[x/s] = s(y \lor z) \lor \overline{s} y = y \lor s z = y \lor x_{1} x_{2}.$$ 

For $\sim \in \{\leq, \geq, =\}$, we write $f \sim g$ if $f(u) \sim g(u)$ is true for all assignments. We collect the variables that $f$ depends on in the set $\text{dep}(f) = \{x \in X : f_{x/1} \neq f_{x/0}\}$. If $u$ is an assignment with $\text{dep}(f) \subseteq V$, then $f(u)$ denotes the only possible value that $f_{u}$ can take.

We say that $f$ is monotone in $x$ if $f_{x/1} \geq f_{x/0}$, and call $f$ monotone if $f$ is monotone in all of its variables. Furthermore, $f$ is the dual of $g$ if $f(u) = \overline{g(\overline{u})}$, where $\overline{\cdot}$ is the variable-wise negation of $u$. We call $f$ symmetric if $f = f_{u}$ for all permutations $\sigma$ of $X$, where $f_{u}(w) = f(\sigma^{-1}(u))$.

Expectations. We denote the expectation of $f$ w.r.t. the uniform distribution over $D$ by $E_{D}[f]$ and call $f$ the expected value of $f$. Furthermore, $f_{u}$ is the dual of $g$ if $f(u) = g(\overline{u})$, where $\overline{\cdot}$ is the variable-wise negation of $u$. We call $f$ symmetric if $f = f_{u}$ for all permutations $\sigma$ of $X$, where $f_{u}(w) = f(\sigma^{-1}(u))$.

Expectations. We denote the expectation of $f$ w.r.t. the uniform distribution over $D$ by $E_{D}[f]$ and call $f$ the expected value of $f$. Furthermore, $f_{u}$ is the dual of $g$ if $f(u) = g(\overline{u})$, where $\overline{\cdot}$ is the variable-wise negation of $u$. We call $f$ symmetric if $f = f_{u}$ for all permutations $\sigma$ of $X$, where $f_{u}(w) = f(\sigma^{-1}(u))$.

Modular decompositions. We introduce a notion of modularity to capture independence of subfunctions as common in the theory of Boolean functions and related fields [Ashenhurst, 1957; Birnbaum and Esary, 1965; Shapley, 1967; Bioch, 2010]. Intuitively, $f$ is modular in $g$ if $f$ treats $g$ like a subfunction and otherwise ignores all variables that $g$ depends on. We define modularity in terms of a template function $\ell$ in which $g$ is represented by a variable $x$:

**Definition 1.** Let $f, g \in \mathbb{B}(X)$. We call $f$ modular in $g$ if $g$ is not constant and there is $\ell \in \mathbb{B}(X)$ and $x \in X$ such that $\text{dep}(f) \cap \text{dep}(g) = \emptyset$ and $f = \ell(x/g)$. If $\ell$ is monotone in $x$, then $f$ is monotonically modular in $g$.

If $f$ is modular in $g$ with $\ell$ and $x$ as above, then $f(u) = \ell(w)$, where $w$ is defined for $y \in X$ as

$$w(y) = \begin{cases} g(u) & \text{if } y = x, \\ u(y) & \text{otherwise.} \end{cases}$$

Thus, the value computed by $g$ is assigned to $x$ and then used by $\ell$, which otherwise is not influenced by the variables that $g$ depends on. For example, $f = x_{1} \lor x_{2} z_{2}$ is modular in $g = z_{1} z_{2}$ since $f$ can be obtained by replacing $x_{1}$ in $\ell = x_{1} \lor x_{2}$ by $y$. Note that $\text{dep}(f) = \{x_{1}, x_{2}, y\}$ and $\text{dep}(g) = \{z_{1}, z_{2}\}$ are disjoint. This property is crucial, since it ensures $f$ and $g$ are coupled through variable $x$ only.

If $f$ is modular in $g$, then the cofactors $\ell_{x/1}$ and $\ell_{x/0}$ must be unique since $g$ is not constant. (See Proposition 5 in the appendix.) Hence, we can define the cofactors of $f$ w.r.t. $g$ as $f_{g/1} = \ell_{x/1}$ and $f_{g/0} = \ell_{x/0}$. The instantiation is reversed by setting $f[y/x] = f_{g/1} \lor f_{g/0}$.

**Boolean derivatives.** We frequently rely on the derivative of a Boolean function $f$ w.r.t. variable $x$:

$$D_{x} f = f_{x/1} \oplus f_{x/0},$$

which encodes the undirected change of $f$ w.r.t. $x$. For example, $f = x \lor y$ has the derivative $D_{x} f = y$, with the intuition that $x$ can only have an impact if $y$ is set to zero. Furthermore, if $f$ is modular in $g$, we define the derivative of $f$ w.r.t. $g$ as $D_{g} f = f_{g/1} \oplus f_{g/0}$. Given this, we obtain the following lemma corresponding to the chain rule known in calculus:

**Lemma 1.** Let $f$ be modular in $g$ and $x \in \text{dep}(g)$. Then

$$D_{x} f = (D_{g} f)(D_{g} g).$$

3 Importance Value Functions

In this section, we devise axiomatic properties that should be fulfilled by every reasonable importance attribution scheme.

For a Boolean function $f$ and a variable $x$, we quantify the importance of $x$ in $f$ by a number $I_{x}(f) \in \mathbb{R}$, computed by some value function $I$. Not every value makes intuitive sense when interpreted as the “importance” of $x$, so we need to pose certain restrictions on $I$.

We argue that $I$ should be bounded, with 1 marking the highest and 0 the lowest importance; that functions which are independent of a variable should rate these variables the lowest importance (e.g., $I_{x}(f) = 0$ if $f = y \lor z$); that functions which depend on one variable only should rate these variables the highest importance (e.g., $I_{x}(f) = 1$ for $f = x$); that neither variable names nor polarities should play a role in determining their importance (e.g., $I_{x}(x y) = I_{x}(x z)$, cf. [Slepian, 1953; Golomb, 1959]):
Definition 2 (IVF). A value function is a mapping of the form $\mathcal{J}: X \times \mathbb{B}(X) \to \mathbb{R}$ with $(x, f) \mapsto \mathcal{J}_x(f)$. An importance value function (IVF) is a value function $\mathcal{J}$ where for all $x, y \in X$, permutations $\sigma$: $X \to X$, and $f, g, h \in \mathbb{B}(X)$:

**(Bound)** $0 \leq \mathcal{J}_x(f) \leq 1$.

**(Dum)** $\mathcal{J}_x(f) = 0$ if $x \notin \text{dep}(f)$.

**(Dic)** $\mathcal{J}_x(x) = \mathcal{J}_x(\overline{x}) = 1$.

**(Type)** (i) $\mathcal{J}_x(f) = \mathcal{J}_{\sigma(x)}(\sigma f)$ and (ii) $\mathcal{J}_x(f) = \mathcal{J}_x(f[y/\overline{y}])$.

**(MODEC)** $\mathcal{J}_x(f) \geq \mathcal{J}_x(h)$ if (i) $f$ and $h$ are monotonically modular in $g$, (ii) $f_{g/1} \geq h_{g/1}$ and $h_{g/0} \geq f_{g/0}$, and (iii) $x \in \text{dep}(g)$.

**Example.** Let $f = x_1 x_2 \lor x_3 x_4 x_5$, $h = x_3 x_4 \lor x_1 x_2 x_5$, and $\mathcal{J}$ be an IVF. Then $f$ encapsulates $h$ on $g = x_1 x_2$, since

$$\frac{1}{f_{g/1}} \geq \frac{x_3 x_4 \lor x_5}{h_{g/1}} \geq \frac{x_3 x_4}{h_{g/0}} \geq \frac{x_3 x_4 x_5}{f_{g/0}}.$$ 

We then get $\mathcal{J}_{x_1}(f) \geq \mathcal{J}_{x_1}(h)$ by application of MODEC. Swapping $x_1$ with $x_3$ and $x_2$ with $x_4$, we obtain a permutation $\sigma$ such that $h = \sigma f$. By TYPE, we derive $\mathcal{J}_{x_3}(h) = \mathcal{J}_{x_3}(f)$. Using TYPE on the other variables yields

$$\mathcal{J}_{x_1}(f) = \mathcal{J}_{x_2}(f) \geq \mathcal{J}_{x_3}(f) = \mathcal{J}_{x_4}(f) = \mathcal{J}_{x_5}(f).$$

Together with TYPE, MODEC implies the Winder pre-order, which is similar in spirit (see [Hammer et al., 2000]). However, MODEC generalizes to modular decompositions and allows inferring importance inequalities w.r.t. to different functions. (See Proposition 6 in the appendix).

**Biased and unbiased.** We say that an IVF is unbiased if $\mathcal{J}_x(g) = \mathcal{J}_x(\overline{g})$ holds for all Boolean functions $g$ and variables $x$. That is, unbiased IVFs measure the impact of variables without any preference for one particular function outcome, while biased ones quantify the impact to enforce a function to return one or zero. Biased IVFs can, e.g., be useful when the task is to assign responsibility values for the violation of a specification.

### 3.1 Further Properties
We defined IVFs following a conservative approach, collecting minimal requirements on IVFs. Further additional properties can improve on the predictability and robustness of IVFs.

**Definition 3.** A value function $\mathcal{J}$ is called

- **rank preserving**, if for all $f, g \in \mathbb{B}(X)$ such that $f$ is modular in $g$ and $x, y \in \text{dep}(g)$:

  $$\mathcal{J}_x(g) \geq \mathcal{J}_y(g) \implies \mathcal{J}_x(f) \geq \mathcal{J}_y(f),$$

- **chain-rule decomposable**, if for all $f, g \in \mathbb{B}(X)$ such that $f$ is modular in $g$ and $x \in \text{dep}(g)$:

  $$\mathcal{J}_x(f) = \mathcal{J}_x(g) \mathcal{J}_y(f),$$

  where $\mathcal{J}_y(f) = \mathcal{J}_{x_1}(f_{g/1}/g_{g/1})$ for some $x_1 \notin \text{dep}(g)$,

- **derivative dependent**, if for all $f, g \in \mathbb{B}(X)$, $x \in X$:

  $$\mathcal{D}_x f \geq \mathcal{D}_x g \implies \mathcal{J}_x(f) \geq \mathcal{J}_x(g).$$

We also consider weak variants of rank preserving and chain-rule decomposable where $f$ ranges only over functions that are monotonically modular in $g$.

**Rank preservation.** Rank preservation states that the relation between two variables should not change if the function is embedded somewhere else. This can be desired, e.g., during a modeling process in which distinct Boolean functions are composed or fresh variables added, where rank preserving IVFs maintain the relative importance order of variables. We see this as a useful but optional property of IVFs since an embedding could change some parameters of a function that might be relevant for the relationship of both variables. For example, if $f = g z$ with $z \notin \text{dep}(g)$, then the relative number of satisfying assignments is halved compared to $g$. If $x$ is more important than $y$ in $g$ but highly relies on $z$ taking value one, it might be that this relationship is reversed for $f$ (cf. example given in Section 4.1).

**Chain-rule decomposability.** If an IVF is chain-rule decomposable, then the importance of a variable in a module is the product of (i) its importance w.r.t. the module and (ii) the importance of the module w.r.t. the function. Many values studied in this paper satisfy this property (Section 4).

**Example.** Let $f = x_1 \oplus \cdots \oplus x_m$, and let $\mathcal{J}$ be a chain-rule decomposable IVFs with $\mathcal{J}_x(x \oplus y) = \alpha$. Since $f$ is modular in $g = x_1 \oplus \cdots \oplus x_{m-1}$ and $g$ modular in $x_1 \oplus \cdots \oplus x_{m-2}$, etc., we can apply the chain-rule property iteratively to get

$$\mathcal{J}_{x_1}(f) = \mathcal{J}_{x_1}(g) \mathcal{J}_y(g) = \mathcal{J}_{x_1}(g) \alpha = \cdots = \alpha^{m-1},$$

where we use TYPE to derive $\mathcal{J}_x(f) = \mathcal{J}_x(x_g \oplus x_m) = \alpha$.

**Derivative dependence.** Derivative dependence states that an IVF should quantify the change a variable induces on a Boolean function. It can be used to derive, e.g., the inequality $\mathcal{J}_x(x_1 \oplus x_2 x_3) \geq \mathcal{J}_x(x_2 \oplus x_1 x_3)$, which is not possible solely using MODEC since $x_1 \oplus x_2 x_3$ is neither monotone in $x_1$ nor in $x_2$. If a value function $\mathcal{J}$ (that is not necessarily an IVF) is derivative dependent, then this has some interesting implications. First, $\mathcal{J}$ is unbiased and satisfies MODEC. Second, if $\mathcal{J}$ is weakly chain-rule decomposable (weakly rank preserving), then it is also chain-rule decomposable (rank preserving). Finally, if $\mathcal{J}$ satisfies DIC and DUM, then it is also bounded by zero and one. As a consequence, if $\mathcal{J}$ is derivative dependent and satisfies DIC, DUM, and TYPE, then $\mathcal{J}$ is an IVF. (See Proposition 7 in the appendix.)

### 3.2 Induced Relations
In this section, we will establish foundational relations between IVFs. Recall that $f$ is a threshold function if
\[ f(u) = 1 \text{ iff } \sum_{x \in X} w_x u(x) \geq \delta \quad \forall u \in \{0, 1\}^X, \]
where \( \{w_x\}_{x \in X} \subseteq \mathbb{R} \) is a set of weights and \( \delta \in \mathbb{R} \) a threshold.

**Theorem 1.** Let \( \mathcal{I} \) be an IVF, \( f, g, h \in \mathbb{B}(X) \), \( x, y \in X \). Then:

1. (i) If \( f \) is symmetric, then \( \mathcal{I}_x(f) = \mathcal{I}_y(f) \).
2. (ii) If \( \mathcal{I} \) is unbiased and \( f \) is dual to \( g \), then \( \mathcal{I}_x(f) = \mathcal{I}_x(g) \).
3. (iii) If \( f \) is a threshold function with weights \( \{w_x\}_{x \in X} \subseteq \mathbb{R} \), then \( |w_x| \geq |w_y| \) implies \( \mathcal{I}_x(f) \geq \mathcal{I}_y(f) \).
4. (iv) If \( f \) is monotonically modular in \( g \) and \( x \in \text{dep}(g) \), then \( \mathcal{I}_x(g) \geq \mathcal{I}_x(f) \).
5. (v) If \( \mathcal{I} \) is derivative dependent and \( x \notin \text{dep}(g) \), then \( \mathcal{I}_x(h \oplus g) = \mathcal{I}_x(h) \).
6. (vi) If \( \mathcal{I} \) is (weakly) chain-rule decomposable, then it is (weakly) rank preserving.

For the case of threshold functions, Theorem 1 shows in (3) that any IVF will rank variables according to their absolute weights. In (4), it is stated that if a function is monotonically embedded somewhere, the importance of variables in that function can only decrease, e.g., \( \mathcal{I}_x(xy) \geq \mathcal{I}_x(xyz) \). Moreover, in (5), if derivative dependence is satisfied, plus-parts without the variable can be dropped. As a consequence, \( \mathcal{I}_x(f) = 1 \) whenever \( f \) is a parity function and \( x \in \text{dep}(f) \).

### 4 Instances of Importance Value Functions

In this section, we show that IVFs can be instantiated with several notions for importance values from the literature and thus provide a unifying framework.

#### 4.1 Blame

Chockler, Halpern, and Kupferman’s (CHK) notions of responsibility [Chockler et al., 2008] and blame [Chockler and Halpern, 2004] measure the importance of \( x \) in that function can only decrease, e.g., \( \mathcal{I}_x(xy) \geq \mathcal{I}_x(xyz) \). Moreover, in (5), if derivative dependence is satisfied, plus-parts without the variable can be dropped. As a consequence, \( \mathcal{I}_x(f) = 1 \) whenever \( f \) is a parity function and \( x \in \text{dep}(f) \).

**Definition 4** (Critical sets). A critical set of \( x \in X \) in \( f \in \mathbb{B}(X) \) under assignment \( u \) over \( X \) is a set \( S \subseteq X \setminus \{x\} \) where

\[ f(u) = f(\text{flip}_S(u)) \text{ and } f(u) \neq f(\text{flip}_{S \cup \{x\}}(u)). \]

We define \( \text{scs}^u_x(f) \) as the size of the smallest critical set, and set \( \text{scs}^u_x(f) = \infty \) if there is no such critical set.

**Example.** The set \( S = \{y\} \) is critical for \( x \) in \( f = x \lor y \) under \( u = x/1; y/1 \). It is also the smallest critical set. On the other hand, there is no critical set if \( u = x/0; y/1 \).

The responsibility of \( x \) for \( f \) under \( u \) is inversely related to \( \text{scs}^u_x(f) \). Using the following notion of a share function, we generalize the original notion of responsibility [Chockler et al., 2008]:

**Definition 5** (Share function). Call \( \rho : \mathbb{N} \cup \{\infty\} \to \mathbb{R} \) a share function if (i) \( \rho \) is monotonically decreasing, (ii) \( \rho(\infty) = \lim_{n \to \infty} \rho(n) = 0 \), and (iii) \( \rho(0) = 1 \).

In particular, we consider three instances of share functions:

- \( \rho_{\text{exp}}(k) = 1/2^k \).
- \( \rho_{\text{trac}}(k) = 1/(k+1) \).
- \( \rho_{\text{step}}(k) = 1 \) for \( k = 0 \) and \( \rho(k) = 0 \) otherwise.

Given a share function \( \rho \), the responsibility of \( x \) for \( f \) under \( u \) is defined as \( \rho(\text{scs}^u_x(f)) \). Note that \( \rho_{\text{trac}}(\text{scs}^u_x(f)) \) implements the classical notion of responsibility [Chockler et al., 2008]. While responsibility corresponds to the size of the smallest critical set in a fixed assignment, CHK’s blame [Chockler et al., 2008] is a global perspective and fits our notion of value function. It is the expected value of the responsibility (we restrict ourselves to uniform distributions):

**Definition 6** (Blame). For a share function \( \rho \), we define the \( \rho \)-blame as value function \( B^\rho \) where for any \( x \in X, f \in \mathbb{B}(X) \):

\[ B^\rho_x(f) = \mathbb{E}_{u \in \{0, 1\}^X}[\rho(\text{scs}^u_x(f))]. \]

**Example.** Let \( f = x \lor y \). To compute the importance of \( x \) we can count the number of times \( \text{scs}^u_x(f) = 0, 1, 2, \ldots, \infty \) occurs if \( u \) ranges over the assignments for \( \{x, y\} \): \( \text{scs}^u_x(f) = \infty \) happens once, \( \text{scs}^u_x(f) = 0 \) happens twice, and \( \text{scs}^u_x(f) = 1 \) occurs once. Therefore,

\[ B^\rho_x(f) = 1/4 \cdot \rho(\infty) + 1/2 \cdot \rho(0) + 1/4 \cdot \rho(1), \]

which is \( 5/8 \) for \( \rho = \rho_{\text{exp}} \).

Independent of \( \rho \), the blame is always an IVF:

**Theorem 2.** \( B^\rho \) is an unbiased IVF for any share function \( \rho \).

In full generality, the blame violates the optional properties for IVFs (see Section 3.1). For example, if \( \rho \neq \rho_{\text{trac}} \), then the \( \rho \)-blame is neither chain-rule decomposable nor derivative dependent, and one can find counterexamples for the rank-preservation property for \( \rho_{\text{trac}} \) and \( \rho_{\text{exp}} \).

**Proposition 1.** Let \( \rho \) be a share function. Then the following statements are equivalent:

1. \( B^\rho \) is weakly chain-rule decomposable,
2. \( B^\rho \) is derivative dependent, and
3. \( \rho = \rho_{\text{step}} \).

**Further, neither** \( B_{\rho_{\text{trac}}} \) nor \( B_{\rho_{\text{exp}}} \) **are weakly rank preserving.**

To give an example for the reason why the \( \rho_{\text{trac}} \)-blame is not weakly rank preserving, consider \( g = x_1 x_2 \lor x_3 \lor x_0 \lor x_3 \) and \( f = g \lor z \). Note that \( f \) is clearly monotonically modular in \( g \) – only \( z \) is added as fresh variable. Nevertheless, the order of \( x_0 \) and \( x_3 \) changes:

\[ B_{\rho_{\text{trac}}}(g) = 0.6302 < 0.7188 = B_{\rho_{\text{trac}}}(g) \]

\[ B_{\rho_{\text{trac}}}(f) = 0.4802 > 0.4688 = B_{\rho_{\text{trac}}}(f). \]

Intuitively, this is because by CHK’s definition of critical sets: for all Boolean functions \( h \), variables \( x \) and assignments \( u \),

\[ h(u) = 1, h_{x/1} \geq h_{x/0}, u(x) = 0 \implies \text{scs}^u_x(h) = \infty. \]
Hence, whenever an assignment \( u \) satisfies the premise for \( x \) in \( h \), the responsibility of \( x \) for \( h \) under \( u \) will be zero.

For \( x_3 \), this is more frequently the case in \( g \) than in \( f \) (19\% vs. 34\% of all assignments). On the other hand, there is always a critical set for \( x_2 \) in both \( f \) and \( g \). Partly for this reason, the importance of \( x_3 \) decreases more than \( x_2 \) when switching from \( g \) to \( f \).

**Modified Blame**

We modify the definition of critical sets in order to derive a modified blame that satisfies more optional properties for a wider class of share functions.

For a Boolean function \( f \), an assignment \( u \) over \( X \) and a variable \( x \), the modified \( scs \) is defined as the size \( mscs^u_x(f) \) of the smallest set \( S \subseteq X \setminus \{x\} \) that satisfies

\[
\text{if } \text{flip}_S(u) \neq \text{flip}_{S \cup \{x\}}(u),
\]

If there is no such set, we set \( mscs^u_x(f) = \infty \).

**Example.** The condition for critical sets is relaxed, hence \( mscs^u_x(f) \) provides a lower bound for \( scs^u_x(f) \). Let for example \( f = x \lor y \) and \( u = x/0 \lor y/1 \). Then \( mscs^u_x(f) = 1 < \infty = scs^u_x(f) \).

The definitions for responsibility and blame are analogous for the modified version, replacing \( scs \) by \( mscs \). We denote by \( MB^u \) the modified \( \rho \)-blame, which is (in contrast to \( B^\rho \)) always derivative dependent and even chain-rule decomposable if \( \rho \) is an exponential- or stepping-function.

**Theorem 3.** \( MB^u \) is an unbiased, derivative-dependent IVF for any share function \( \rho \). If there is \( 0 \leq \lambda < 1 \) so that \( \rho(k) = \lambda^k \) for all \( k \geq 1 \), then \( MB^u \) is chain-rule decomposable.

**4.2 Influence**

The influence [Ben-Or and Linial, 1985; Kahn et al., 1988; O’Donnell, 2014] is a popular importance measure, defined as the probability that flipping the variable changes the function’s outcome for uniformly distributed assignments:

**Definition 7.** The **influence** is the function \( I_x(f) = E[D_xf] \) for all \( f \in \mathcal{B}(X) \) and variables \( x \in X \).

It turns out that the influence is a special case of blame:

**Proposition 2.** \( I = MB^{\rho_{step}} = B^{\rho_{step}} \).

Since \( \rho_{step}(k) = 0^k \) for \( k \geq 1 \), Proposition 2 and Theorem 3 show that the influence is a derivative-dependent, rank-preserving, and chain-rule decomposable IVF.

**Characterizing the Influence**

Call a value function \( \mathcal{I} \) cofactor-additive if for all Boolean functions \( f \) and variables \( x \neq z \):

\[
\mathcal{I}_x(f) = \nicefrac{1}{2} \cdot \mathcal{I}_z(f_{z/0}) + \nicefrac{1}{2} \cdot \mathcal{I}_z(f_{z/1}).
\]

Using this notion, we axiomatically characterize the influence as follows.

**Theorem 4.** A value function \( \mathcal{I} \) satisfies DIC, DUM, and cofactor-additivity if and only if \( \forall \mathcal{I} = I \).

**Remark.** A relaxed version of cofactor-additivity assumes the existence of \( \alpha_z, \beta_z \in \mathbb{R} \) for \( z \in X \) such that for all \( x \neq z \):

\[
\mathcal{I}_x(f) = \alpha_z \mathcal{I}_z(f_{z/0}) + \beta_z \mathcal{I}_z(f_{z/1}).
\]

This, together with the assumption that \( \mathcal{I} \) satisfies TYPE, DUM and DIC, implies \( \alpha_z = \beta_z = 1/2 \). Hence, another characterization of the influence consists of TYPE, DUM, DIC, and relaxed cofactor-additivity. (See Corollary 1 in the appendix.)

Moreover, we give a syntactic characterization of the influence by a comparison to the two-sided Jeroslaw-Wang heuristic used for SAT-solving [Jerolmack and Wang, 1990; Hooker and Vinay, 1995; Marques-Silva, 1999]. This value is defined for families of sets of literals, which are sets of subsets of \( \{z : z \in X\} \), and it weights subsets that contain \( x \) or \( \overline{x} \) by their respective lengths:

**Definition 8** ([Hooker and Vinay, 1995]). Let \( D \) be a family of sets of literals. The two-sided Jeroslaw-Wang value for a variable \( x \) is defined as

\[
JW_x(D) = \sum_{C \in D} I_x(C) \cdot 2^{-|C|}
\]

We call a set \( C \) of literals *trivial* if there is a variable \( x \) such that \( x \in C \) and \( \overline{x} \in C \). For a variable \( x \), say that \( D \) is \( x \)-orthogonal if for all \( C, C' \in D \), \( C \neq C' \), there is a literal \( \eta \notin \{x, \overline{x}\} \) such that \( \eta \in C \) and \( \overline{\eta} \in C' \). Orthogonality is well-studied for DFNs [Crama and Hammer, 2011b]. The two-sided Jeroslaw-Wang value and the influence agree up to a factor of two for some families of sets of literals when interpreting them as DFNs:

**Theorem 5.** Let \( D \) be a family of sets of literals such that all of its elements are non-trivial, and let \( x \) be variable such that \( D \) is \( x \)-orthogonal. Then:

\[
I_x(\bigvee_{C \in D} C) = 2 \cdot JW_x(D).
\]

A simple example that illustrates Theorem 5 would be \( D = \{\{x, y, z\}, \{y, \overline{z}\}\} \). Note that we can interpret \( D \) as a CNF as well, since the influence does not distinguish between a function and its dual (Theorem 1). Note that every Boolean function can be expressed by a family \( D \) that satisfies the conditions of Theorem 5. For this, we construct the canonical DNF corresponding to \( f \) and resolve all monomials that differ only in \( x \). (See Proposition 8 in the appendix.)

**4.3 Cooperative Game Mappings**

Attribution schemes analogous to what we call value functions were already studied in the context of game theory, most often with emphasis on Shapley- and Banzhaf values [Shapley, 1953; Banzhaf, 1965]. They are studied w.r.t. cooperative games, which are a popular way of modeling collaborative behavior. Instead of Boolean assignments, their domains are subsets (coalitions) of \( X \). Specifically, cooperative games are of the form \( v : 2^X \to \mathbb{R} \), in which the value \( v(S) \) is associated with the payoff that variables (players) in \( S \) receive when collaborating. Since more cooperation generally means higher payoffs, they are often assumed to be monotonically increasing w.r.t. set inclusion. In its unconstrained form, they are essentially pseudo Boolean functions.

We denote by \( \mathcal{G}(X) \) the set of all cooperative games. If \( \text{image}(v) \subseteq \{0, 1\} \), then we call \( v \) simple. For a cooperative game \( v \), we denote by \( \partial_x v \) the cooperative game that computes the “derivative” of \( v \) w.r.t. \( x \), which is \( \partial_x v(S) = v(S \cup \{x\}) - v(S \setminus \{x\}) \).

We compose cooperative games using operations such as \( \cdot, +, -, \land, \lor \) etc., where \( (v \circ w)(S) = \)
\(\nu(S) \circ \nu(S)\). For \(\sim \in \{\geq, \leq, =\}\), we also write \(\nu \sim \nu\) if \(\nu(S) \sim \nu(S)\) for all \(S \subseteq X\). The set of variables \(\nu\) depends on is defined as \(\text{dep}(\nu) = \{x \in X : \partial_x \nu \neq 0\}\).

Cooperative game mappings map Boolean functions to cooperative games. Specific instances of such mappings have previously been investigated by [Hammer et al., 2000; Biswas and Sarkar, 2022]. We provide a general definition of this concept to show how it can be used to construct IVFs.

**Definition 9 (CGM).** A cooperative game mapping (CGM) is a function \(\tau : \mathcal{B}(X) \to \mathcal{G}(X)\) with \(f \mapsto \tau_f\). We call \(\tau\) importance inducing if for all \(x, y \in X\), permutations \(\sigma : X \to X\), and \(f, g, h \in \mathcal{B}(X)\):

\[
\begin{align*}
(\text{BOUNDCG}) \quad & \partial_x \tau_f \leq 1, \\
(\text{DUMCG}) \quad & \partial_x \tau_f = 0 \text{ if } x \notin \text{dep}(f), \\
(\text{DICCG}) \quad & \partial_x \tau_x = \partial_x \tau_\emptyset = 1, \\
(\text{TYPECG}) \quad & (i) \tau_f(S) = \tau_{x|\sigma(S)}(\sigma(S)) \text{ and } \\
& (ii) \tau_f(S) = \tau_{f|\sigma(S)}(\sigma(S)) \text{ for all } S \subseteq X. \\
(\text{MODECG}) \quad & \partial_x \tau_f \geq \partial_x \tau_h \text{ if } \partial_x f \text{ and } \partial_x h \text{ are monotonically modular in } g, \\
& (iii) f, g, h \text{ are already-known CGMs}, \\
& \tau_f \text{ is weakly chain-rule decomposable if this holds for all cases where } f \text{ is monotonically modular in } g.
\end{align*}
\]

We call \(\tau\) unbiased if \(\tau_\emptyset = \tau_{\emptyset}\) for all \(x \in \mathcal{B}(X)\).

An example is the characteristic CGM \(\zeta\) given by \(\zeta_f(S) = f(1_S), \) where \(1_S(x) = 1\) iff \(x \in S\). We study various importance-inducing CGMs in the following sections. Note that \(\zeta\) is not important inducing: for example, it violates \(\text{BOUNDCG}\) since \(\partial_x \zeta_f(\emptyset) = -1\) for \(f = \emptyset\).

The restriction to importance-inducing CGMs ensures that compositions with the Banzhaf or Shapley values are valid IVFs (Lemma 2). These CGMs satisfy properties that are related to Definition 2: \(\tau_f\) should be monotone \((0 \leq \partial_x \tau_f)\), irrelevant variables of \(f\) are also irrelevant for \(\tau_f\) (\(\text{DUMCG}\)), etc. In an analogous fashion, we can think of properties related to Definition 3:

**Definition 10.** A CGM \(\tau\) is called

\begin{itemize}
\item **chain-rule decomposable**, if for all \(f, g \in \mathcal{B}(X)\) such that \(f\) is modular in \(g\) and \(x \in \text{dep}(g)\):
\[
\partial_x \tau_f = (\partial_x \tau_f)(\partial_y \tau_f),
\]
where \(\partial_x \tau_f = \partial_x \tau_{f|\{x\}}\) for some \(x \notin \text{dep}(f)\). We call \(\tau\) weakly chain-rule decomposable if this holds for all cases where \(f\) is monotonically modular in \(g\).

\item **derivative dependent**, if for all \(f, g \in \mathcal{B}(X), x \in X\):
\[
D_x f \geq \partial_x g \implies \partial_x \tau_f \geq \partial_x \tau_g.
\]
\end{itemize}

Since (weak) rank-preservation for value functions uses an IVF in its premise, it cannot be stated naturally at the level of CGMs. Let us now define the following abstraction, which captures Shapley and Banzhaf values:

**Definition 11.** Call \(E : X \times \mathcal{G}(X) \to \mathbb{R}, (x, v) \mapsto E_x(v)\) a value function for cooperative games. Call \(E\) an expectation of contributions if there are weights \(c(0), \ldots, c(n-1) \in \mathbb{R}\) such that for all \(v \in \mathcal{G}(X)\) and \(x \in X\):

\[
\sum_{S \subseteq X \setminus \{x\}} c(|S|) = 1 \quad \text{and} \quad E_x(v) = \sum_{S \subseteq X \setminus \{x\}} c(|S|) \cdot \partial_x v(S).
\]

If \(E\) is an expectation of contributions, then \(E_x(v)\) is indeed the expected value of \(\partial_x \nu(S)\) in which \(\forall S \subseteq X \setminus \{x\}\) has probability \(\nu(S)\). The Banzhaf and Shapley values are defined as the expectations of contributions with weights:

\[
\begin{align*}
\nu_b(x) &= \frac{1}{|X|} (\text{Bz})\quad \text{and} \quad \nu_b(x) = \frac{1}{|X|^2} \binom{n-1}{k} (\text{Sh}).
\end{align*}
\]

Observe that there are \(\binom{n-1}{k}\) sets of size \(k\) in \(\{0, \ldots, n-1\}\), so the weights of the Shapley value indeed sum up to one.

If \(\tau\) is a CGM, then its composition with \(E\) yields \((E \circ \tau)(f) = E_x(\tau_f),\) which is a value function for Boolean functions. Then every composition with an expectation of contributions is an IVF if the CGM is importance inducing:

**Lemma 2.** If \(\tau\) is an importance-inducing CGM and \(E\) an expectation of contributions, then \(E \circ \tau\) is an IVF. If \(\tau\) is unbiased/derivative dependent, then so is \(E \circ \tau\). Finally, if \(\tau\) is (weakly) chain-rule decomposable, then so is \(E \circ \tau\).

In the following sections, we study two novel and the already-known CGM of [Hammer et al., 2000]. By Lemma 2 we can focus on their properties as CGMs, knowing that any composition with the Shapley value or other expectations of contributions will induce IVFs.

**Simple Satisfiability-Biased Cooperative Game Mappings**

The first CGM interprets the “power” of a coalition as its ability to force a function’s outcome to one: If there is an assignment for a set of variables that yields outcome one no matter the values of other variables, we assign this set a value of one, and zero otherwise.

**Definition 12.** The dominating CGM \(\omega\) is defined as

\[
\omega_f(S) = \begin{cases} 
1 & \text{if } \exists u \in \{0, 1\}^S, \forall w \in \{0, 1\}^{X \setminus S} : f(u, w), \\
0 & \text{otherwise}.
\end{cases}
\]

**Example.** Let \(f = x \lor (y \land z)\). We have \(\omega_f(\{y, z\}) = 1\) since \(f_u = 1\) for \(u = y/1, z/0\). On the other hand, \(\omega_f(\{y\}) = 0\), since \(x/0, z/1\) resp. \(x/0, z/0\) falsify \(f_y/1\) and \(f_y/0\).

**Theorem 6.** The dominating CGM is weakly chain-rule decomposable and importance inducing.

**Example.** Let \(Z\) be the expectation of contributions with \(\omega(0) = 1\), i.e., \(Z_x(v) = v(\{x\}) - v(\emptyset)\). By Lemma 2 and Theorem 6, the mapping

\[
(Z \circ \omega)(f) = \begin{cases} 
1 & \text{if } f \neq 1 \text{ and } f_x/0 = 1 \text{ or } f_x/1 = 0, \\
0 & \text{otherwise}.
\end{cases}
\]

is an IVF. Intuitively, \(x\) has the highest importance if the function is falsifiable and there is a setting for \(x\) that forces the function to one. Otherwise, \(x\) has an importance of zero.

**Biasedness and rank preservation.** The dominating CGM is biased: Consider \(g = x \lor (y \lor z)\) with \(\text{F} = \text{F} \land (\text{F} \lor z)\). Note that \(\omega_T(S) = 1\) for \(S = \{x\}\) while \(\omega_{\text{F}}(S) = 0\), which shows biasedness. Composing \(\omega\) with the Banzhaf value yields

\[
\begin{align*}
\text{Bz} \circ \omega_T(g) : z &= 0.25 : y = 0.25 : x = 0.75, \\
\text{Bz} \circ \omega_T(\text{F}) : z &= 0.25 : y = 0.25 : x = 0.25,
\end{align*}
\]

One can force \(g\) to one by controlling either \(x\) or both \(y\) and \(z\), so \(x\) is rated higher than the others. But to force \(\text{F}\) to one, control over all variables is required, so all variables in \(\text{F}\) have the same importance.
Since $g$ is modular in $\overline{7}$, we also obtain a counterexample for rank preservation: 
\[(Bz \circ \omega)_g(\overline{7}) \geq (Bz \circ \omega)_x(\overline{7})\]
does not imply \[(Bz \circ \omega)_g(g) \geq (Bz \circ \omega)_x(g)\].
However, weak rank preservation is fulfilled by $Bz \circ \omega$ since it is weakly chain-rule decomposable by Theorem 6 and Lemma 2. Then the claim follows with Theorem 1.

A dual to the dominating CGM. One can think of a dual notion of the CGM $\omega$ that reverses the order of both quantifiers. Intuitively, we are now allowed to choose an assignment depending on the values of the remaining variables:

**Definition 13.** The rectifying CGM $\nu$ is defined as 
$$
\nu_f(S) = \begin{cases} 
1 & \text{if } \forall w \in \{0, 1\}^{X \setminus S}, \exists u \in \{0, 1\}^S. f(u; w), \\
0 & \text{otherwise}.
\end{cases}
$$

If we compose $\nu$ with an expectation of contributions that satisfies $c(k) = c(n-1-k)$ for all $k \in \{0, \ldots, n-1\}$, which is a condition satisfied both by the Shapley and Banzhaf values, the induced importance of a variable equals its importance w.r.t. $\omega$ and the negated function:

**Proposition 3.** Let $E$ be an expectation of contributions with $c(k) = c(n-1-k)$ for all $k \in \{0, \ldots, n-1\}$. Then for all $g \in E(X)$ and $x \in X$:
\[(E \circ \omega)_x(g) = (E \circ \nu)_x(g)\]

We now discuss connections to the influence. If a Boolean function is monotone, and we “control” a set of variables $S$, the best towards satisfaction (resp. falsification) is to set all variables in $S$ to one (resp. to zero). This can be used to show that both $Bz \circ \omega$ and $Bz \circ \nu$ agree with the influence:

**Proposition 4.** Let $f$ be a monotone Boolean function and $x$ a variable. Then $(Bz \circ \omega)_x(f) = (Bz \circ \nu)_x(f) = I_x(f)$.

A Constancy-Based Cooperative Game Mapping
Hammer, Kogan and Rothblum [Hammer et al., 2000] (HKR) defined a CGM that measures the power of variables by how constant they make a function if assigned random values. It depends on the following notion of constancy measure:

**Definition 14.** We call a mapping $\kappa : [0, 1] \rightarrow [0, 1]$ a constancy measure if (i) $\kappa$ is convex, (ii) $\kappa(0) = 1$, (iii) $\kappa(1) = \kappa(1-x)$, and (iv) $\kappa(1/2) = 0$.

The following functions are instances of constancy measures:
- $\kappa_{\text{quad}}(a) = 4(a - 1/2)^2$,
- $\kappa_{\text{log}}(a) = 1 + \log(b) + (1-a)\log(1-a)$ with $0\log(0) = 0$,
- $\kappa_{\text{val}}(a) = 2|a - 1/2|$.

For a constancy measure $\kappa$ and a Boolean function $f$, the $\kappa$-constancy of $f$ is the value $\kappa(E[f])$, which measures how balanced the share of ones and zeros is. It is close to one if $f$ is very unbalanced and close to zero if the share of zeros and ones in $f$ is (almost) the same. The power of a set of variables $S$ is now measured in terms of the expected $\kappa$-constancy of $f$ if variables in $S$ are fixed to random values:

**Definition 15.** ([Hammer et al., 2000]). Given a constancy measure $\kappa$, we define the CGM $H^\kappa$ by $H^\kappa_f(S) = E_{a \in \{0, 1\}^S} [\kappa(E[f_a])].$

**Example.** Let $f = x \lor y \lor z$ and $S = \{x\}$. We obtain $H^\kappa_f(S) = 1/2 \cdot \kappa(3/4) + 1/2 \cdot \kappa(1)$, since $E[f_{x/0}] = \kappa(3/4)$ and $E[f_{x/1}] = 1$.

Setting $x$ to zero does not determine $f$ completely, while setting it to one also sets $f$ to one, i.e., makes it constant. The measure then gives a lower value to the less-constant cofactor, a higher value to the more-constant cofactor and computes the average. For this example and $\kappa = \kappa_{\text{abs}}$, we obtain $H^\kappa_f(S) = \kappa(3/4) + 1/2 = 1$ and $\kappa(1) = 1$.

Theorem 7 shows that $H^\kappa_{\text{quad}}$ is a chain-rule decomposable and importance-inducing CGM. It is open whether other constancy measures are importance inducing too.

**Theorem 7.** Suppose $\kappa$ is a constancy measure. Then $H^\kappa$ is an unbiased CGM that satisfies BOUNDCG, DICCG, DUMCG, and TYPECG. Further, $H^\kappa_{\text{quad}}$ is chain-rule decomposable and satisfies MODECG.

**Example.** For the special case where $\kappa = \kappa_{\text{quad}}$, note that $1/2 \cdot \kappa(a) + 1/2 \cdot \kappa(b) = 1/2 \cdot a + 1/2 \cdot b = (a - b)^2$. Using $E[f] = 1/2 \cdot E[f_{x/1}] + 1/2 \cdot E[f_{x/0}]$, this implies $(Z \circ H^\kappa)_x(f) = (E[f_{x/1}] - E[f_{x/0}])^2$, where $Z$ is again the expectation of contributions with $Z_x(v) = v(\{x\}) - v(\emptyset)$.

The value $Z \circ H^\kappa$ is an IVF according Lemma 2 and Theorem 7. In contrast to derivative-dependent IVFs, $Z \circ H^\kappa$ assigns low values to variables in parity-functions: for $f = x \lor y$, we have $E[f_{x/1}] = E[f_{x/0}]$, and thus $(Z \circ H^\kappa)_x(f) = 0$.

**Derivative dependence.** This property cannot be achieved, as witnessed by $f = x \lor y$ and $g = x$. Due to $D_x f = D_x g$, it suffices to show that $D_x H^\kappa_f \neq D_x H^\kappa_g$ holds for all $\kappa$. Note that $E[f_{x/0}] = 1/2$, $E[f_{x/1}] = 1/2$, $E[g_{x/0}] = 0$, $E[g_{x/1}] = 1$, and $E[f] = E[g] = 1/2$. Thus, for all constancy measures $\kappa$,
- $D_x H^\kappa_f(\emptyset) = 1/2 \cdot \kappa(1/2) + 1/2 \cdot \kappa(1/2) - \kappa(1/2) = 0$,
- $D_x H^\kappa_g(\emptyset) = 1/2 \cdot \kappa(1/2) + 1/2 \cdot \kappa(0) - \kappa(1/2) = 1$,
which shows $D_x H^\kappa_f \neq D_x H^\kappa_g$.

5 Computing Importance Values
In this section, we present and evaluate computation schemes for blame, influence, and CGMs. While there exists a practical approach based on model counting for the influence in CNFs [Traxler, 2009], we are only aware of naïve computations of CHK’s blame [Dubslaff et al., 2022]. Details are given in the appendix.

**Blame.** We focus on the modified blame. CHK’s blame can be computed in a very similar fashion. Observe that for a Boolean function $f$ and $x \in X$,

$$
MB^\kappa_f(x) = E[\gamma_0] + \sum_{k=1}^{n-1} \rho(k)(E[\gamma_k] - E[\gamma_{k-1}]),
$$
where $\gamma_k$ is the Boolean function for which $\gamma_k(u) = 1$ iff $mscs^\kappa_f(u) \leq k$. We devise two approaches for computing $E[\gamma_k]$. The first represents $\gamma_k$ through BDDs using the following recursion scheme: $mscs^\kappa_f(u) \leq k$ holds iff
• $k = 0$ and $f(u) \neq f(\text{flip}_x(u))$, or
• $k > 0$ and
  - $\text{mscs}_k^y(f) \leq k - 1$ or
  - there is $y \neq x$ such that $\text{mscs}_k^y(f[y/\gamma]) \leq k - 1$.

This allows us to construct BDDs for $\gamma_k$ from $\gamma_{k-1}$, which lends itself to BDD-based approaches since $\gamma_k$ does not necessarily increase in size as $k$ grows. The second approach introduces new existentially quantified variables in the input formula of $f$ to model occurrences of variables in critical sets of $\text{mscs}_k^y(f)$. With an additional cardinality constraint restricting the number of variables in critical sets to at most $k$, we can use projected model counting to compute $E[\gamma_k]$.

**Influence.** In case $f$ is given as a CNF formula, we use Traxler’s method to compute the influence [Traxler, 2009]. For all other formulas, note that standard satisfiability-preserving transformations do not preserve influence values: For example, applying the Tseytin transformation to $x \lor y$ results in a CNF where $x$ has a higher influence than $y$.

However, the influence is proportional to the number of models of $D_x f$. If $f$ is given by a BDD, computing a representation of $D_x f$ means squaring $f$’s size in the worst case, while the formula-based representation only doubles it. For the latter case, we can count the models of $D_x f$ using a Tseytin transformation and a standard model counter.

**BDD representations of satisfiability-biased CGMs.** The dominating CGM computes a simple game, which is essentially a Boolean function, and therefore permits a representation by BDDs. Moreover, using a BDD representation of $f$, we compute $\omega_f$ using a recursion on cofactors of variables $z$,

$$(\omega_f)_{z=1} = \omega_{f_z} \lor \omega_{f_z/0} \quad \text{and} \quad (\omega_f)_{z=0} = \omega_{f_z} \land \omega_{f_z/1}.$$

The Banzhaf value of $x$ in $\omega_f$ is then just

$$E[(\omega_f)_{x=1}] - E[(\omega_f)_{x=0}],$$

which poses no effort once the BDD of $\omega_f$ is constructed. The rectifying CGM can be computed analogously.

**Implementation and evaluation.** We have implemented Traxler’s method and our new computation schemes in Python, using BuDDy [Lind-Nielsen, 1999] as BDD backend with automatic reordering and GPMC [Suzuki et al., 2015; Suzuki et al., 2017] for (projected) model counting. To evaluate our approaches, we conducted experiments on Boolean functions given as CNFs that were either randomly generated or generated from the ISCAS’99 dataset [Davidson, 1999; Compile! Project, 2023]. We always computed importance values w.r.t. the first variable in the input CNF and averaged the timings over 20 runs each. Our experiments were carried out on a Linux system with an i5-10400F CPU at 2.90GHz and 16GB of RAM. To compare our BDD-based and model counting approaches, Figure 1 shows timings for blame computations on random CNFs. Here, the BDD-based approach clearly outperforms the one based on projected model counting. This is also reflected in real-world benchmarks from ISCAS’99 shown in Table 1, where the approach based on model counting runs into timeouts for even small instances. Table 1 shows that computations for influence values based on model counting scale better than the BDD-based approach, mainly due to an expensive initial BDD construction. Computing the BDD of the dominating CGM is done without much overhead once the BDD for the CNF is given.

### 6 Conclusion

This paper introduced IVFs as a way to formally reason about importance of variables in Boolean functions. We established general statements about IVFs, also providing insights on notions of importance from the literature by showing that they all belong to the class of IVFs. Apart from revealing several relations between known IVFs, we have shown how to generate new ones inspired by cooperative game theory.

For future work, we will study properties with strict importance inequalities, IVFs for sets of variables, IVFs for pseudo Boolean functions, and global values similar to the total influence [O’Donnell, 2014]. On the empirical side, the generation of splitting rules for SAT-solvers and variable-order heuristics for BDDs based on different instances of IVFs are promising avenues to pursue.

**Acknowledgments.** The authors were partly supported by the DFG through the DFG grant 389792660 as part of TRR 248 and the Cluster of Excellence EXC 2050/1 (CeTI,
project ID 390696704, as part of Germany’s Excellence Strategy) and “SAIL: SustAInable Life-cycle of Intelligent Socio-Technical Systems” (Grant ID NW21-059D), funded by the program “Netzwerke 2021” of the Ministry of Culture and Science of the State of North Rhine-Westphalia, Germany.

References


A Section 2, Preliminaries

Notation

Since most of the “heavy lifting” is done here in the appendix, we need to introduce some more notation in order to keep everything as simple as possible.

- We consider pseudo Boolean functions, which take \{0, 1\}^X to \mathbb{R} and can be composed using +, -, etc. Cofactors, the set of dependent variables and comparison operations are defined as for Boolean functions.
- If two assignments \(u, v\) are defined over the same domain, we can compose them variable-wise using operations such as \(\land, \lor, \oplus\). I.e., \((u \oplus v)(x) = u(x) \oplus v(x)\), etc.
- For \(S \subseteq X\), denote the corresponding indicator function by \(1_S\), with \(1_S(x) = 1\) iff \(x \in S\). The domain of \(1_S\) depends on the context, but contains at least \(S\). Write \(1_x = 1_{\{x\}}\).
- Instead of substituting only one variable, we can use substitution chains of the form \(f[x_i/s_i : i \in N]\) or similar if the order of replacement does not matter.

Note that:

- For any permutation \(\sigma\) of \(X\), the variable-wise interpretations of \(\lor, \land, \oplus\) imply that \(\sigma\) distributes over these operations when applied to functions or assignments, e.g. \(\sigma(u \land v) = \sigma(u) \land \sigma(v)\) or \(\sigma(u; v) = \sigma(u; v)\). The same holds for sets \(A, B \subseteq X\) with e.g. \(\sigma(A \cap B) = \sigma(A \cap B)\).
- The identity \(u \oplus 1_S = \text{flip}_S(u)\) applies, which makes it obvious that the flipping-operation can be written as a composition of assignments. For more basic properties, we refer to Lemma 3.

Lemma 3. Let \(\sigma\) be a permutation of \(X\), \(h\) be a Boolean function and \(T \subseteq S \subseteq X\) be arbitrary subsets. Then

\[
\{0, 1\}^S \rightarrow \{0, 1\}^{\sigma(S)}, \ u \mapsto \sigma(u \oplus 1_T) \text{ is a bijection.} \tag{A.1}
\]

\[
(\sigma u)_{\sigma(T)} = \sigma(u_T) \quad \forall u \in \{0, 1\}^S \tag{A.2}
\]

\[
\sigma u \oplus 1_{\sigma(T)} = \sigma(u \oplus 1_T) \quad \forall u \in \{0, 1\}^S \tag{A.3}
\]

\[
(\sigma h)_{\sigma u} = \sigma(h_u) \quad \forall u \in \{0, 1\}^S \tag{A.4}
\]

Proof.

(A.1) This mapping can be inverted by taking \(\{0, 1\}^{\sigma(S)}\) back to \(\{0, 1\}^S\) with \(v \mapsto (\sigma^{-1} v) \oplus 1_T\). Thus, it is a bijection.

(A.2) If \(x \in \sigma(T)\) and \(y = \sigma^{-1}(x)\), then

\[
(\sigma u)_{\sigma(T)}(x) = \sigma u(x) = u(y).
\]

Since \(y \in T\) by construction,

\[
u(y) = u_T(y) = u_T(\sigma^{-1}(x)) = \sigma(u_T)(x).
\]

Furthermore, if \(x \notin \sigma(T)\), then \((\sigma u)_{\sigma(T)}\) is not defined on \(x\) and neither is \(\sigma(u_T)\) due to \(y \notin T\). Therefore, both assignments must be equal.

(A.3) Follows from (A.2),

\[
\sigma u \oplus 1_{\sigma(T)} = (\sigma u)_{X \setminus \sigma(T); \overline{\sigma T}} = \sigma(u_{X \setminus T}; \overline{\sigma T})
\]

\[
= \sigma(u_{X \setminus T}; \overline{\sigma T})
\]

\[
= \sigma(u \oplus 1_T).
\]

(A.4) For all \(v \in \{0, 1\}^X\),

\[
(\sigma h)_{\sigma u}(v) = \sigma h(\sigma u; v_{X \setminus \sigma(S)})
\]

\[
= \sigma h(\sigma(u; \sigma^{-1}(v_{X \setminus \sigma(S)})))
\]

\[
= h(u; \sigma^{-1}(v_{X \setminus \sigma(S)}))
\]

\[
= h(u; (\sigma^{-1} v)_{X \setminus S})
\]

\[
= h_u(\sigma^{-1} v)
\]

\[
= \sigma(h_u)(v).
\]
There are a few properties of the derivative operator which are useful to point out. First, taking the derivative with respect to a function that does not depend on \( x \) results in the constant zero function (A.5). Second, it is multiplicative in the sense that functions which do not depend on \( x \) can be moved in front of the operator (A.6). And third, it is additive in the sense of (A.7):

**Lemma 4.** Let \( f, h, g \) be Boolean functions such that \( x \notin \text{dep}(g) \). Then

\[
\begin{align*}
D_x g &= 0 \quad \text{(A.5)} \\
D_x(gf) &= gD_x f \quad \text{(A.6)} \\
D_x(f + h) &= D_x f + D_x h. \quad \text{(A.7)}
\end{align*}
\]

**Proof.** (A.5) follows from the fact that \( g_{x/1} = g_{x/0} \). For (A.6), we get \( (gf)_{x/1} \oplus (gf)_{x/0} = g(f_{x/1} \oplus f_{x/0}) \) since \( g_{x/1} = g_{x/0} = g \). Finally, for (A.7), note that \( (f + h)_{x/1} \oplus (f + h)_{x/0} = (f_{x/1} \oplus f_{x/0}) \oplus (h_{x/1} \oplus h_{x/0}) \).

Expectations are frequently used throughout this paper, so let us establish a few simple equalities. First, one can compute the expected value of a pseudo Boolean function with respect to every superset of its dependent variables. Second, if two pseudo Boolean functions depend on mutually exclusive variables (dependence defined as for Boolean functions), then the expectation of their product can be decomposed. And third, expected values of modular Boolean functions admit decompositions as well:

**Lemma 5.**

(i) If \( f \) is a pseudo Boolean function and \( \text{dep}(f) \subseteq R \), then \( E_{u \in \{0,1\}^R}[f(u)] = E_{u \in \{0,1\}^n}[f(u)] \).

(ii) If \( f, g \) are pseudo Boolean functions such that \( \text{dep}(f) \cap \text{dep}(g) = \emptyset \), then \( E[f \cdot g] = E[f] \cdot E[g] \).

(iii) If \( f, g \) are Boolean functions such that \( f \) is modular in \( g \), then \( E[f] = E[g]E[f_{g/1}] + (1 - E[g])E[f_{g/0}] \).

**Proof.**

(i)

\[
E_{u \in \{0,1\}^R}[f(u)] = E_{v \in \{0,1\}^n \setminus R}[E_{u \in \{0,1\}^R}[f(u; v)]] \quad \text{uniform distribution}
\]

\[
= E_{v \in \{0,1\}^n \setminus R}[E_{u \in \{0,1\}^R}[f(u)]] \quad \text{dep}(f) \cap (X \setminus R) = \emptyset
\]

\[
= E_{u \in \{0,1\}^n}[f(u)] \quad \text{E[const] = const.}
\]

(ii)

\[
E[f \cdot g] = E_{u \in \{0,1\}^{\text{dep}(f)}}[E_{v \in \{0,1\}^{\text{dep}(f)}}[f(u; v) \cdot g(u; v)]] \quad \text{uniform distribution}
\]

\[
= E_{u \in \{0,1\}^{\text{dep}(f)}}[E_{v \in \{0,1\}^{\text{dep}(f)}}[f(u) \cdot g(v)]] \quad \text{dep}(f) \cap \text{dep}(g) = \emptyset \text{ and } \text{dep}(g) \subseteq X \setminus \text{dep}(f)
\]

\[
= E_{u \in \{0,1\}^n}[f(u)] \cdot E_{v \in \{0,1\}^{\text{dep}(f)}}[g(v)] \quad \text{using (i)}
\]

\[
= E[f] \cdot E[g].
\]

(iii)

\[
E[f] = E[g_{f/1}] + (1 - g)g_{f/0} \quad \text{E}[g_{f/1} \lor \overline{g}g_{f/0}] = g_{f/1} + (1 - g)g_{f/0} \quad \text{linearity of E}
\]

\[
= E[g]E[f_{g/1}] + (1 - E[g])E[f_{g/0}] \quad \text{independence, using (ii)}
\]

**Modular Decompositions**

The following proposition shows that the cofactors of a modular Boolean function are indeed uniquely determined:

**Proposition 5.** Suppose \( f \) is modular in \( g \) and let \( s, t \in \mathbb{B}(X) \) and \( x, y \in X \) such that \( f = s[y/g] = t[x/g] \) with \( \text{dep}(g) \cap (\text{dep}(s) \cup \text{dep}(t)) = \emptyset \). Then \( s_{y/1} = t_{x/1} \) and \( s_{y/0} = t_{x/0} \).

**Proof.** Recall that

\[
f = g s_{y/1} \lor \overline{g} s_{y/0} = g t_{x/1} \lor \overline{g} t_{x/0}.
\]

Since \( g \) is not constant, pick \( u_1, u_0 \in \{0,1\}^{\text{dep}(s)} \) such that \( g_{u_1} = 1 \) and \( g_{u_0} = 0 \). Then, due to the fact that the cofactors \( s_{y/1}, s_{y/0}, t_{x/1} \) and \( t_{x/0} \) do not depend on any variable of \( g \),

\[
s_{y/1} = f_{u_1} = t_{x/1},
\]

\[
s_{y/0} = f_{u_0} = t_{x/0}.
\]
Boolean Derivatives
The following property mirrors the chain rule as known in calculus. It follows from some basic rules:

**Lemma 1.** Let \( f \) be modular in \( g \) and \( x \in \text{dep}(g) \). Then
\[
D_x f = (D_x g)(D_y f).
\]

**Proof.**
For \( f \) and \( g \) as described:
\[
D_x f = D_x (g f_{g/1} \oplus (1 \oplus g) f_{g/0}) = (D_x g) f_{g/1} \oplus ((D_x 1) \oplus (D_x g)) f_{g/0} = (D_x g)(D_y f)
\]

**B Section 3, Importance Value Functions**
The Winder preorder according to [HKR00] (Section 3 therein) is defined for a Boolean function \( f \) and variables \( x, y \) as
\[
x \geq_f y \text{ iff } f_{x/1; y/0} \geq f_{x/0; y/1}.
\]

Similar to [HKR00] (Theorem 3.1 therein), we focus only on the case were \( f \) is monotone in \( x \) and \( y \). MoDEC can then be seen as a generalization of this order in two senses: first, we abstract this order to the level of modules. Second, we abstract from an inequality between variables in one function to an inequality between variables in two functions. The following proposition now links the Winder preorder with MoDEC and Type:

**Proposition 6.** Suppose \( \mathcal{I} \) is an importance value function and let \( f \) be a Boolean function monotone in \( x \) and \( y \). Then:
\[
x \geq_f y \implies \mathcal{I}_x(f) \geq \mathcal{I}_y(f).
\]

**Proof.** Suppose that \( x \geq_f y \). Define \( h = \sigma f \), where \( \sigma \) is the permutation of \( X \) that swaps \( x \) and \( y \). Since \( f_{x/c; y/d} \) does not depend on \( x \) nor \( y \), note that \( f_{x/c; y/d} = \sigma(f_{x/c; y/d}) \), which in turn implies \( f_{x/c; y/d} = h_{x/d; y/c} \) according to Lemma 3. Thus, we can derive \( f_{x/1} \geq h_{x/1} \) and \( h_{x/0} \geq f_{x/0} \) since
\[
f_{x/1} \geq h_{x/1} \iff \begin{cases} f_{x/1; y/1} \geq h_{x/1; y/1} \\ f_{x/1; y/0} \geq h_{x/1; y/0} \end{cases} \quad \text{and} \quad h_{x/0} \geq f_{x/0} \iff \begin{cases} h_{x/0; y/0} \geq f_{x/0; y/0} \\ h_{x/0; y/1} \geq f_{x/0; y/1} \end{cases}
\]
Since both \( f \) and \( h \) are monotone in \( x \), apply MoDEC to derive \( \mathcal{I}_x(f) \geq \mathcal{I}_x(h) \). Using Type, conclude \( \mathcal{I}_x(h) = \mathcal{I}_y(f) \). ■

The following proposition formalizes and proves claims that were made in Section 3 with respect to various implications of derivative dependence:

**Proposition 7.** Suppose \( \mathcal{I} \) is a derivative-dependent value function. Then,
1. It is unbiased.
2. It satisfies MoDEC.
3. If \( \mathcal{I} \) is weakly chain rule decomposable, it is chain rule decomposable.
4. If \( \mathcal{I} \) is weakly rank preserving, then it is rank preserving.
5. If \( \mathcal{I} \) satisfies DIC and DUM, then it satisfies BOUND.

**Proof.**
1. Follows from the fact that \( D_x g = D_x \overline{g} \) for all Boolean functions \( g \) and variables \( x \): First note \( \overline{g} = 1 \oplus g \), then apply \( A.7 \) and \( A.5 \).
2. Assume that \( f \) encapsulates \( h \) on \( g \) and \( x \in \text{dep}(g) \). We want to show that \( \mathcal{I}_x(f) \geq \mathcal{I}_x(h) \). To do so, we note that due to \( f_{g/0} \geq h_{g/0} \geq 0 \) and \( h_{g/0} = f_{g/1} \oplus f_{g/0} \) it is indeed the case that \( D_y f \geq D_y h \). Furthermore, by Lemma 1, mind that \( D_x f = (D_x g)(D_y f) \) and \( D_x h = (D_x g)(D_y h) \). Due to \( D_x g \geq 0 \), it must be the case that \( D_x f \geq D_x h \). Application of derivative dependence now shows \( \mathcal{I}_x(f) \geq \mathcal{I}_x(h) \).
3. Suppose that \( \mathcal{I} \) is weakly chain rule decomposable, that \( f \) is modular in \( g \) and \( x \in \text{dep}(g) \). We want to show \( \mathcal{I}_x(f) = \mathcal{I}_x(g) \mathcal{I}_y(f) \). Let us define \( h = g(f_{g/1} \oplus f_{g/0}) \vee (f_{g/0} \| f_{g/0}) \). It is easy to see that \( h \) is monotonically modular in \( g \) and that \( D_y h = D_y f \). Furthermore, by Lemma 1, note that \( D_x f = (D_x g)(D_y f) = (D_x g)(D_y h) = D_x h \). Therefore, since \( \mathcal{I} \) is derivative dependent and weakly chain rule decomposable,
\[
\mathcal{I}_x(f) = \mathcal{I}_x(h) = \mathcal{I}_x(g) \mathcal{I}_x(h) = \mathcal{I}_x(g) \mathcal{I}_y(f).
\]

4. Suppose that \( J \) is weakly rank preserving, that \( f \) is modular in \( g \) and \( x, y \in \text{dep}(g) \) such that \( J_x(g) \geq J_y(g) \). We want to show \( J_x(f) \geq J_y(f) \). If we define \( h \) as in \( J \), we can derive \( J_x(f) = J_x(h) \) and \( J_y(f) = J_y(h) \) due to \( J \)'s derivative dependence. Furthermore, note that \( J_x(h) \geq J_y(h) \) must hold by weak rank preservation. So \( J_x(f) \geq J_y(f) \) holds as well.

5. Suppose \( f \) is an arbitrary Boolean function and \( x \) a variable. We want to show \( 0 \leq J_x(f) \leq 1 \). Mind that \( D_xf \leq 1 = D_xx \), so \( J_x(f) \leq J_x(x) = 1 \) due to Dic and \( J \)'s derivative dependence. Furthermore, if \( g \) is an arbitrary Boolean function with \( x \notin \text{dep}(g) \), then \( D_xg = 0 \leq D_xf \), which allows the application of DUM and derivative dependence to conclude \( 0 = J_x(g) \leq J_x(f) \).

\[ \begin{array}{ll}
\text{Theorem 1.} & \text{Let } J \text{ be an IVF, } f, g, h \in \mathbb{B}(X), \ x, y \in X. \text{ Then:} \\
\text{(1)} & \text{If } f \text{ is symmetric, then } J_x(f) = J_y(f). \\
\text{(2)} & \text{If } J \text{ is unbiased and } f \text{ is dual to } g, \text{ then } J_x(f) = J_x(g). \\
\text{(3)} & \text{If } f \text{ is a threshold function with weights } \{ w_x \}_{x \in X} \subseteq \mathbb{R}, \text{ then } |w_x| \geq |w_y| \implies J_x(f) \geq J_y(f). \\
\text{(4)} & \text{If } f \text{ is monotonically modular in } g \text{ and } x \in \text{dep}(g), \text{ then } J_x(g) \geq J_x(f). \\
\text{(5)} & \text{If } J \text{ is derivative dependent and } x \notin \text{dep}(g), \text{ then } J_x(h \oplus g) = J_x(h). \\
\text{(6)} & \text{If } J \text{ is (weakly) chain-rule decomposable, then it is (weakly) rank preserving.} \\
\end{array} \]

Proof.

1. Define \( \sigma \) to be the permutation that swaps \( x \) and \( y \). Then \( \sigma f = f \) by symmetry of \( f \), so we get \( J_x(f) = J_{\sigma(x)}(\sigma f) = J_y(f) \) due to TYPE.

2. Follows directly from the fact that \( f = \overline{\overline{\gamma}}[\overline{y}/\overline{y} : y \in X] \). So \( J_x(f) = J_x(\overline{\overline{y}}) \) from repeated application of TYPE. Finally, \( J_x(\overline{\overline{y}}) = J_x(y) \) due to \( J \)'s unbiasedness.

3. We first assume that all weights are non-negative and show the general case later. Define \( h = \sigma f \), where \( \sigma \) is the (self-inverse) permutation that swaps \( x \) and \( y \). Note that \( h(u) = 1 \) iff

\[ u(x)w_y + u(y)w_x + \sum_{z \in X \setminus \{x, y\}} u(z)w_z \geq \delta. \]

Let us demonstrate that \( \text{MoDEC} \) is applicable to derive \( J_x(f) \geq J_x(h) \): as \( w_x \geq 0 \) and \( w_y \geq 0 \), both \( f \) and \( h \) must be monotone in \( x \). Further, due to \( w_x \geq w_y \), we also have \( f_{x/1} \geq h_{x/1} \) and \( h_{x/0} \geq f_{x/0} \), i.e. \( f \) encapsulates \( h \) on \( x \). So use \( \text{MoDEC} \) to get \( J_x(f) \geq J_x(h) \) and then TYPE to get \( J_x(f) \geq J_{\sigma(x)}(\sigma h) = J_y(f) \).

Finally, suppose now that \( f \) is a threshold function with possibly negative weights. Let \( Y = \{ z \in X : w_z < 0 \} \) and define \( \ell = f[z/\overline{z} : z \in Y] \). Then \( \ell(u) = 1 \) iff

\[ \sum_{z \in Y}(1 - u(z))w_z + \sum_{z \in X \setminus Y} u(z)w_z \geq \delta \]

\[ \iff \sum_{z \in X} u(z)|w_z| \geq \delta - \sum_{z \in Y} w_z. \]

Thus, \( \ell \) is a threshold function with non-negative weights and we can apply TYPE and the same argument as above to get \( J_x(f) = J_x(\ell) \geq J_y(\ell) = J_y(f) \).

4. Note that \( g_{y/1} = 1 \geq f_{y/1} \) and \( f_{y/0} \geq 0 \). Since \( f \) is monotonically modular in \( g \) and \( x \in \text{dep}(g) \) by assumption, we can apply \( \text{MoDEC} \) and conclude \( J_x(g) \geq J_y(f) \).

5. Assume that \( J \) is derivative dependent. Since \( D_x(h \oplus g) = D_xh \oplus D_xg = D_xh \) due to (A.7) and (A.5), note that \( J_x(h \oplus g) = J_x(h) \) must hold.

6. Assume that \( J \) is (weakly) chain rule decomposable. We want to show that it is (weakly) rank preserving. Suppose \( f \) is (monotonically) modular in \( g \) and pick \( x, y \in \text{dep}(g) \) such that \( J_x(g) \geq J_y(g) \). Since \( J \) is (weakly) chain rule decomposable, \( J_x(f) = J_x(g)J_y(f) \) and \( J_y(f) = J_y(g)J_y(f) \). Now apply the fact that \( J_y(f) \geq 0 \) to conclude \( J_x(f) \geq J_y(f) \). Therefore, \( J \) must be (weakly) rank preserving.

\[ \blacksquare \]
C Section 4, Instances of Importance Value Functions

Blame

Let us introduce a modified version of Definition 4, which introduces the dependence on a constant $c \in \{0, 1\}$. This is a bit more useful for the coming proofs and statements.

**Definition 16.** Let $f$ be a Boolean function, $x$ be a variable, $c \in \{0, 1\}$ be a constant and $u$ be an assignment over $X$. Define

$$
s_{scs}^u(f, c) = \min_{S \subseteq X \setminus \{x\}} |S| \quad \text{s.t.} \quad c = f(u \oplus 1_S) \neq f(u \oplus 1_S \oplus 1_x).
$$

If there is no solution, set $s_{scs}^u(f, c) = \infty$.

Note that $s_{scs}^u(f) = s_{scs}^u(f, f(u))$. We say that a subset $S \subseteq X \setminus \{x\}$ is a *candidate solution* of $s_{scs}^u(f, c)$ if it satisfies $c = f(u \oplus 1_S) \neq f(u \oplus 1_S \oplus 1_x)$. We say that $S$ is an *optimal solution* if $|S|$ is minimal as well. When showing an inequality $s_{scs}^u(f, c) \geq s_{scs}^u(g, c)$, it suffices to show that every candidate solution of $s_{scs}^u(f, c)$ translates to a candidate solution of $s_{scs}^u(g, c)$ of equal cardinality. Furthermore, let us introduce the following value, which measures the minimal number of changes one has to make to an assignment to make it satisfy a function:

**Definition 17.** Let $f$ be a Boolean function and $u$ be an assignment over $X$. Define

$$
d_u^u(f) = \min_{S \subseteq X} |S| \quad \text{s.t.} \quad f(u \oplus 1_S) = 1.
$$

If there is no solution, set $d_u^u(f) = \infty$.

As for $s_{scs}^u(f, c)$, we say that $S \subseteq X$ is a *candidate solution* of $d_u^u(f)$ if $f(u \oplus 1_S) = 1$, and if $|S|$ is minimal as well, we say that $S$ is an *optimal solution*. It is now possible to establish some helpful identities regarding the smallest critical set. More precisely, we can see that it does not distinguish between functions and their complements (equality (C.1)), that an assignment $u$ and its "neighbour" $u \oplus 1_x$ share similarities (equality (C.2)), and that the smallest critical set does not care about variable names or polarities (equalities (C.3) and (C.4)). Finally, we observe that there is a useful decomposition of $s_{scs}$ for the monotonically modular case (equality (C.5)).

**Lemma 6 (Properties of $s_{scs}$).** Suppose $g$ is a Boolean function, $u$ an assignment, $x, z$ variables, $c \in \{0, 1\}$ a constant and $\sigma$ a permutation. Then

$$
s_{scs}^u(g, c) = s_{scs}^\overline{u}(g, \overline{c}) \quad \text{(C.1)}
$$

$$
= s_{scs}^{u \oplus 1_x}(g, \overline{c}) \quad \text{(C.2)}
$$

$$
= s_{scs}^{u \oplus 1_z}(g[z/\overline{c}], c) \quad \text{(C.3)}
$$

$$
= s_{scs}^{\sigma \overline{u}}(\sigma g, c). \quad \text{(C.4)}
$$

If $f$ is monotonically modular in $g$ and $x \in \text{dep}(g)$, then

$$
s_{scs}^u(f, c) = s_{scs}^u(g, c) + d_u^u(f_{g/1} \overline{f_{g/0}}). \quad \text{(C.5)}
$$

**Proof.**

(C.1) One can simply check that every candidate solution $T$ of $s_{scs}^u(g, c)$ is also a candidate solution of $s_{scs}^{\overline{u}}(g, \overline{c})$ and vice versa. So both values must be equal. The same line of argument is applicable to (C.2) and (C.3). I.e., for all $T \subseteq X \setminus \{x\}$,

$$
c = g(u \oplus 1_T) \neq g(u \oplus 1_T \oplus 1_x) \iff \overline{c} = \overline{g}\left((u \oplus 1_T) \oplus 1_x\right) \neq \overline{g}\left((u \oplus 1_T \oplus 1_x)\right) \text{ (C.1)}
$$

$$
\iff \overline{T} = g((u \oplus 1_T) \oplus 1_x) \neq g((u \oplus 1_T) \oplus 1_T \oplus 1_x) \text{ (C.2)}
$$

$$
\iff c = g[z/\overline{c}]((u \oplus 1_z) \oplus 1_T) \neq g[z/\overline{c}](u \oplus 1_Z) \oplus 1_T \oplus 1_x \text{ (C.3)}
$$

(C.4) Note that for all $T \subseteq X$, by (A.3),

$$
x \not\in T, \quad c = g(u \oplus 1_T) \neq g(u \oplus 1_T \oplus 1_x)
$$

$$
\iff \sigma(x) \not\in \sigma(T), \quad c = \sigma(g(u \oplus 1_{\sigma(T)}) \neq \sigma(g(u \oplus 1_{\sigma(T)} \oplus 1_x)))
$$

Since $|T| = |\sigma(T)|$, the values $s_{scs}^u(g, c)$ and $s_{scs}^{\sigma \overline{u}}(\sigma g, c)$ must be equal.

(C.5) Let us show

$$
s_{scs}^u(f, c) = s_{scs}^u(g, c) + d_u^u(f_{g/1} \overline{f_{g/0}})
$$

for $x \in \text{dep}(g)$ by considering both directions “$\leq$” and “$\geq$”. First however, note that due to $f$’s modularity, for all $G \subseteq \text{dep}(g)$ and $F \subseteq X \setminus \text{dep}(g)$,

$$
f(u \oplus 1_{G \cup F}) = g(u \oplus 1_G) f_{g/1}(u \oplus 1_F) \lor \overline{g}(u \oplus 1_G) f_{g/0}(u \oplus 1_F), \quad \text{(*)}
$$

$$
f(u \oplus 1_{G \cup F} \oplus 1_x) = g(u \oplus 1_G \oplus 1_x) f_{g/1}(u \oplus 1_F) \lor \overline{g}(u \oplus 1_G \oplus 1_x) f_{g/0}(u \oplus 1_F). \quad \text{(**)}
$$
Theorem 2. \( B^\rho \) is an unbiased IVF for any share function \( \rho \).

**Proof.**

\( B^\rho \) is unbiased  
Note that (C.1) also implies \( \text{scs}^u_x(g, c) = \text{scs}^u_x(g, \mathbb{T}) \) for all Boolean functions \( g \) and variables \( x \). Then it is easy to see that \( B^\rho_x(g) = B^\rho_x(\mathbb{T}) \).

\[ 0 \leq B^\rho_x(f) \leq 1 \]

follows from the fact that \( B^\rho \) is a convex combination of terms limited by 0 and 1.

**DIC**  
Let \( x \) be a variable. We want to show \( B^\rho_x(x) = 1 \). \( B^\rho_x(\mathbb{T}) = 1 \) then follows from TYPE. Note that \( \text{scs}^u_x(x) = 0 \) and thus \( \rho(\text{scs}^u_x(x)) = 1 \) for all assignments \( u \), so \( B^\rho_x(x) = 1 \) follows.

**DUM**  
Let \( f \) be a Boolean function and \( x \notin \text{dep}(f) \) be a variable. We want to show \( B^\rho_x(f) = 0 \). Note that \( \text{scs}^u_x(f) = \infty \) for all assignments \( u \) since flipping \( x \) can never make a difference – the set of candidate solutions to \( \text{scs}^u_x(f) \) must therefore be empty. But then \( \rho(\text{scs}^u_x(f)) = 0 \) for all assignments \( u \) and thus \( B^\rho_x(f) = 0 \).

**TYPE**  
Suppose \( f \) is a Boolean function, \( \sigma \) a permutation and \( x, y \) variables. We want to show (i) \( B^\rho_x(f) = B^\rho_x(\sigma f) \) and (ii) \( B^\rho_x(f) = B^\rho_y(f[y/y]) \). To see (i), note that (C.4) also implies \( \text{scs}^u_x(f) = \text{scs}^u_x(\sigma f) \) due to the fact that \( f(u) = \sigma f(\sigma u) \). Hence,

\[ B^\rho_x(f) = \mathbb{E}_{u \in \{0,1\}^X} [\rho(\text{scs}^u_x(\sigma f))]. \]

Using the fact that the mapping \( u \mapsto \sigma u \) is a permutation of assignments over \( X \) and that all assignments are equally likely, we get

\[ \mathbb{E}_{u \in \{0,1\}^X} [\rho(\text{scs}^u_x(\sigma f))] = \mathbb{E}_{u \in \{0,1\}^X} [\rho(\text{scs}^u_x(\sigma f))] = B^\rho_x(\sigma f). \]

For (ii), apply (C.3) and \( f(u) = f[y/y](u \oplus 1_y) \) to get \( \text{scs}^u_x(f) = \text{scs}^u_x(1_y) \). Then argue analogously as for (i) and use the fact that \( u \mapsto u \oplus 1_y \) is a permutation.
for these functions in order to derive statements about $\rho$.

Proposition 1.

\[ (ii) \quad (i) \]

\[
\begin{array}{cccccccccccccccc}
 x & y & z & f_1(u) & f_2(u) & f_3(u) & f_4(u) & h(u) & \text{scs}^u_x(f_1) & \text{scs}^u_x(f_2) & \text{scs}^u_x(f_3) & \text{scs}^u_x(f_4) & \text{scs}^u_x(h) \\
 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\
 0 & 0 & 1 & 1 & 1 & 0 & 0 & 1 & 0 & 2 & \infty & 0 & 0 & 0 & 0 \\
 0 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & \infty & \infty & 0 & 0 & 0 & 0 \\
 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & \infty & \infty & 0 & 1 & 1 & 1 \\
 1 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & \infty \\
 1 & 0 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\
 1 & 1 & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 0 \\
 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 2 & 2 & 1 & 0 & 1 & 1 \\
\end{array}
\]

Table 2: Functions and scs-values from Proposition 1.

**MODEC** Let us assume that $f, h, g$ are Boolean functions such that (i) $f$ and $h$ are monotonically modular in $g$, (ii) $f_{g/1} \geq h_{g/1}$ and $h_{g/0} \geq f_{g/0}$ and (iii) $x \in \text{dep}(g)$. We want to show $B^\rho_x(f) \geq B^\rho_x(h)$. We first show this assuming the following statement:

\[
\rho(\text{scs}^u_x(f)) + \rho(\text{scs}^u_x(\oplus 1_x(f))) \geq \rho(\text{scs}^u_x(h)) + \rho(\text{scs}^u_x(\oplus 1_x(h))) \quad \text{for all } u \in \{0,1\}^X \tag{C.6}
\]

Since $u \mapsto u \oplus 1_x$ is a permutation and we're using the uniform distribution, we have:

\[
B^\rho_x(f) = \frac{1}{2}\mathbb{E}_{u \in \{0,1\}^X}[\rho(\text{scs}^u_x(f))] + \frac{1}{2}\mathbb{E}_{u \in \{0,1\}^X}[\rho(\text{scs}^u_x(\oplus 1_x(f)))]
\]

By linearity of $\mathbb{E}$ and inequality (C.6) we can conclude $B^\rho_x(f) \geq B^\rho_x(h)$.

**Proof of (C.6):** If $f(u) \neq f(u \oplus 1_x)$, then the left side is maximal and the inequality holds trivially. So let us assume $f(u) = f(u \oplus 1_x)$. Furthermore, it must then be the case that $h(u) = h(u \oplus 1_x)$: If $h(u) \neq h(u \oplus 1_x)$, then $g(u) \neq g(u \oplus 1_x)$, $h_{g/1}(u) = 1$ and $h_{g/0}(u) = 0$ by (ii) and (iii), which implies $f_{g/1}(u) = 1$ and $f_{g/0}(u) = 0$ by (ii). But then $f(u) \neq f(u \oplus 1_x)$, contradicting our assumption.

Now that we have established $h(u) = h(u \oplus 1_x)$, let us distinguish into two cases: $c = f(u) = h(u)$ and $c = f(u) \neq h(u)$. For the first case, using (ii) and (C.5),

\[
\text{scs}^u_x(f) = \text{scs}^u_x(g, c) + d^u(f_{g/1}f_{g/0}).
\]

By (ii), note that $f_{g/1}f_{g/0} \geq h_{g/1}h_{g/0}$, which implies

\[
\text{scs}^u_x(f) \leq \text{scs}^u_x(g, c) + d^u(h_{g/1}h_{g/0}) = \text{scs}^u_x(h)
\]

due to $h(u) = c$. The same reasoning can be applied to get $\text{scs}^u_x(\oplus 1_x(f)) \leq \text{scs}^u_x(\oplus 1_x(h))$. This already suffices to obtain (C.6) by noting that $\rho$ is monotonically decreasing. For the case $c = f(u) \neq h(u)$, let us again apply decomposition (C.5) to get

\[
\text{scs}^u_x(f) = \text{scs}^u_x(g, c) + d^u(f_{g/1}f_{g/0}) \\
= \text{scs}^u_x(g, \overline{x}) + d^u(c, (f_{g/1}f_{g/0}) = (C.2) \text{ and } x \notin \text{dep}(f_{g/1}f_{g/0}) \\
\leq \text{scs}^u_x(g, \overline{x}) + d^u(1_x, (h_{g/1}h_{g/0}) = (C.2) \text{ and } x \notin \text{dep}(f_{g/1}f_{g/0}) \\
= \text{scs}^u_x(h) = \overline{c}
\]

We can derive $\text{scs}^u_x(\oplus 1_x(f)) \leq \text{scs}^u_x(h)$ analogously. Using again the fact that $\rho$ is monotonically decreasing, we can conclude that (C.6) holds.

Proposition 1 shows that some properties are satisfied only if one chooses a very specific share function. Proofs utilize Table 2, which contains scs-values for six different functions. We combine implications of properties such as derivative dependence for these functions in order to derive statements about $\rho$.

**Proposition 1.** Let $\rho$ be a share function. Then the following statements are equivalent:

(i) $B^\rho$ is weakly chain-rule decomposable,
(ii) $B^\rho$ is derivative dependent, and
(iii) $\rho = \rho_{\text{step}}$

Further, neither $B^{\text{strac}}$ nor $B^{\text{exp}}$ are weakly rank preserving.

Proof. For the equivalence of (i), (ii) and (iii), note that if $\rho = \rho_{\text{step}}$, then $B^\rho$ is simply the influence, which is both chain rule decomposable and derivative dependent as we will see later on (cf. Proposition 2). Therefore, let us focus on the directions from (i) to (iii) and (ii) to (iii), i.e. we assume that $B^\rho$ is chain rule decomposable (resp. derivative dependent) and we want to show that this implies $\rho = \rho_{\text{step}}$.

(i) implies (iii): Define

$$
\begin{align*}
    f_1 &= (x \oplus y) \lor z \\
    f_2 &= x \lor y \lor z \\
    f_3 &= x \lor y \\
    f_4 &= x \oplus y
\end{align*}
$$

And we get by $\rho(0) = 1$ and $\rho(\infty) = 0$, using Table 2,

$$
\begin{align*}
    B^\rho_x(f_1) &= \frac{1}{2^3}(4 + 2\rho(1) + 2\rho(2)), \\
    B^\rho_x(f_2) &= \frac{1}{2^2}(2 + 2\rho(1) + \rho(2)), \\
    B^\rho_x(f_3) &= \frac{1}{2^2}(4 + 2\rho(1)), \\
    B^\rho_x(f_4) &= 1.
\end{align*}
$$

Since $\rho$ is weakly chain rule decomposable and satisfies TYPE, we can see that $B^\rho_x(f_1) = B^\rho_x(f_4) B^\rho_x(f_3)$ and $B^\rho_x(f_2) = B^\rho_x(f_3)^2$. Solving the first equation implies $\rho(2) = 0$. Observe that we have:

$$
B^\rho_x(f_3)^2 = \frac{(4 + 2\rho(1))^2}{64} = \frac{16 + 16\rho(1) + 4\rho(1)^2}{64},
$$

Now $B^\rho_x(f_3)^2 = B^\rho_x(f_2)$ together with $\rho(2) = 0$ yields:

$$
\begin{align*}
B^\rho_x(f_3)^2 &= \frac{16 + 16\rho(1) + 4\rho(1)^2}{64} = \frac{1 + \rho(1)}{4} = B^\rho_x(f_2) \\
\iff 16 + 16\rho(1) + 4\rho(1)^2 &= 16 + 16\rho(1) \\
\iff \rho(1) &= 0
\end{align*}
$$

Since $\rho$ is monotonically decreasing but bounded by zero, $\rho(n) = 0$ for $n \geq 1$. But then $\rho = \rho_{\text{step}}$.

(ii) implies (iii): Define

$$
\begin{align*}
    f &= xy \oplus \overline{xy} \\
    h &= x(y \lor z) \oplus \overline{xyz}.
\end{align*}
$$

Note that $D_x f = D_x h$. Furthermore, using Table 2,

$$
\begin{align*}
    B^\rho_x(f) &= \frac{1}{2^4}(4 + 4\rho(1)) \\
    B^\rho_x(h) &= \frac{1}{2^3}(4 + 2\rho(1)).
\end{align*}
$$

Since $\rho$ is derivative dependent, we can see that $B^\rho_x(f) = B^\rho_x(h)$ must hold, which implies $\rho(1) = 0$ after resolving both equations. But then $\rho(n) = 0$ for all $n \geq 1$ since $\rho$ is monotonically decreasing. Hence, $\rho = \rho_{\text{step}}$.

For the last part, we have automatically generated counterexamples for $B^{\text{exp}}$ and $B^{\text{strac}}$. For $B^{\text{exp}}$, let us consider the following functions,

$$
\begin{align*}
    g &= \overline{x_1} \overline{x_0} x_2 \lor x_1 x_0 x_2 \lor x_3 \overline{x_0} x_2 \lor x_3 x_0 x_2 \lor x_3 x_1, \\
    f &= g \lor z.
\end{align*}
$$

We can clearly see that $f$ is monotonically modular in $g$. Furthermore, $B^{\text{exp}}$ results in the following ranking,

$$
\begin{align*}
    g : \quad & x_3 : 0.6250 < x_0 : 0.7188 = x_2 : 0.7188 < x_1 : 0.7266, \\
    f : \quad & x_3 : 0.4062 < x_1 : 0.4980 < x_0 : 0.5000 = x_2 : 0.5000 < z : 0.6172.
\end{align*}
$$
Note how the positions of 1 and 0 are flipped when extending \( g \) to \( f \), so \( B^{\rho \text{dep}} \) is not weakly rank preserving. A counterexample for \( B^{\rho \text{rsv}} \) is given by

\[
\begin{align*}
g &= x_1 \overline{x}_0 \overline{x}_2 \lor \overline{x}_1 x_0 \lor x_3, \\
f &= g \lor z,
\end{align*}
\]

which results in the ranking

\[
\begin{align*}
g &: x_2: 0.2969 < x_1: 0.6302 = x_0: 0.6302 < x_3: 0.7188 \\
f &: x_2: 0.2250 < z: 0.4688 = x_3: 0.4688 < x_1: 0.4802 = x_0: 0.4802
\end{align*}
\]

and reverses the positions of 0 and 3.

**Modified Blame**

Like (C.5), there is a way to decompose the value \( msccs^u_x(f) \) if \( f \) is modular in \( g \) and \( x \in \text{dep}(g) \). However, we can express this decomposition using only \( msccs \):

**Lemma 7.** If \( f \) is modular in \( g \), \( u \) an assignment over \( X \) and \( x \in \text{dep}(g) \), then

\[
msccs^u_x(f) = msccs^u_x(g) + msccs^u_y(f).
\]

**Proof.** Suppose \( S \) is an optimal solution to \( msccs^u_x(f) \), and define \( G = S \cap \text{dep}(g) \) and \( F = S \cap (X \setminus \text{dep}(g)) \). By Lemma 1,

\[
D_x f(u \oplus 1_S) = 1 \Rightarrow D_x g(u \oplus 1_G) = D_x f(u \oplus 1_F) = 1,
\]

so \( G \) is a candidate solution to \( msccs^u_x(g) \) and \( F \) a candidate solution to \( msccs^u_y(f) \). But then

\[
msccs^u_x(f) = |S| = |G| + |F| \geq msccs^u_x(g) + msccs^u_y(f).
\]

For the other direction, suppose that \( G \) resp. \( F \) are optimal solutions to \( msccs^u_x(g) \) and \( msccs^u_y(f) \). Define \( S = G \cup F \). By Lemma 1,

\[
D_x g(u \oplus 1_G) = D_x f(u \oplus 1_F) = 1 \Rightarrow D_x f(u \oplus 1_S) = 1,
\]

which shows that \( S \) is a candidate solution to \( msccs^u_x(f) \) and thus

\[
msccs^u_x(f) \leq |S| \leq |G| + |F| = msccs^u_x(g) + msccs^u_y(f).
\]

By Proposition 7, if \( MB^\rho \) satisfies the properties DIC, DUM, derivative dependence and TYPE, then it must be an unbiased IVF. For the chain-rule property, we rely on Lemma 7:

**Theorem 3.** \( MB^\rho \) is an unbiased, derivative-dependent IVF for any share function \( \rho \). If there is \( 0 \leq \lambda < 1 \) so that \( \rho(k) = \lambda^k \) for all \( k \geq 1 \), then \( MB^\rho \) is chain-rule decomposable.

**Proof.** Let us first show that \( MB^\rho \) is indeed an unbiased importance value function.

**DIC** Let \( x \) be a variable. We want to show \( MB^\rho_x(x) = 1. \) \( MB^\rho_x(\overline{x}) = 1 \) then follows from TYPE. Note that since \( D_x x = 1 \), we have \( msccs^u_x(x) = 0 \) for all assignments \( u \). Thus, since \( MB^\rho_x(x) \) is a convex combination of \( \rho(0) = 1 \)'s, we get \( MB^\rho_x(x) = 1 \).

**DUM** Let \( f \) be a Boolean function and \( x \notin \text{dep}(f) \) be a variable. We want to show \( MB^\rho_x(f) = 0 \). Note that \( D_x f = 0 \), which implies \( msccs^u_x(f) = \infty \) for all assignments \( u \). Since \( MB^\rho_x(f) \) is a convex combination of \( \rho(\infty) = 0 \)'s we get \( MB^\rho_x(f) = 0 \).

**deriv. dep.** Let \( f, h \) be Boolean functions and \( x \) a variable such that \( D_x f \geq D_x h \). We want to show \( MB^\rho_x(f) \geq MB^\rho_x(h) \). First mind that the inequality \( msccs^u_x(f) \leq msccs^u_x(h) \) holds for all assignments \( u \), which follows from the fact that \( msccs^u_x(f) = d^u(D_x f) \) and \( msccs^u_x(h) = d^u(D_x h) \) and by noting that \( \ell \geq r \) implies \( d^u(\ell) \leq d^u(r) \) (every candidate solution to \( d^u(\ell) \) is automatically a candidate solution to \( d^u(r) \).) Turning back to derivative dependence, we use the fact that \( \rho \) is monotonically non-increasing to obtain

\[
MB^\rho_x(f) = \mathbb{E}_{u \in \{0,1\}^x}[\rho(msccs^u_x(f))] \geq \mathbb{E}_{u \in \{0,1\}^x}[\rho(msccs^u_x(h))] = MB^\rho_x(h).
\]
Theorem 4. A value function $\mathcal{I}$ satisfies DIC, DUM, and cofactor-additivity if and only if $\mathcal{I} = I$. 

**Proof.** Follows from the fact that $D_x f(u) = \rho_{step}(\text{mscs}_x^u(f)) = \rho_{step}(\text{scs}_x^u(f))$ for all Boolean functions $f$, variables $x$ and assignments $u$. 

The additional additivity property states that the importance of a variable to a function should always be the average importance with respect to the cofactors of another variable. We can applying this property repeatedly to obtain special cases which are covered by DUM and DIC:

**Theorem 4.** A value function $\mathcal{I}$ satisfies DIC, DUM, and cofactor-additivity if and only if $\mathcal{I} = I$. 

**Proof.** Follows from the fact that $D_x f(u) = \rho_{step}(\text{mscs}_x^u(f)) = \rho_{step}(\text{scs}_x^u(f))$ for all Boolean functions $f$, variables $x$ and assignments $u$. 

The additional additivity property states that the importance of a variable to a function should always be the average importance with respect to the cofactors of another variable. We can applying this property repeatedly to obtain special cases which are covered by DUM and DIC:
Proof. Let us first establish that \( I \) indeed satisfies all of these properties. Proposition 2 suffices to show that \( I \) satisfies DIC and DUM. For cofactor-additivity, note that

\[
I_x(f) = \frac{1}{2^m} \sum_{u \in \{0,1\}^m} D_x f(u)
\]

definition

\[
= \frac{1}{2^m} \sum_{u \in \{0,1\}^{m \setminus \{x\}}} D_x f(z/0; u) + D_x f(z/1; u)
\]

expanding w.r.t. \( z \)

\[
= \frac{1}{2^m} \sum_{u \in \{0,1\}^{m \setminus \{x\}}} D_x (f_{z/0})(u) + D_x (f_{z/1})(u)
\]

\[
= \frac{1}{2} I_x(f_{z/0}) + \frac{1}{2} I_x(f_{z/1}).
\]

For the other direction, let us assume that \( \mathcal{J} \) is cofactor-additive and that it satisfies DIC and DUM. Use cofactor-additivity repeatedly on every variable except for \( x \) to get

\[
\mathcal{J}_x(f) = \frac{1}{2} \mathcal{J}_x(f_{z/0}) + \frac{1}{2} \mathcal{J}_x(f_{z/1})
\]

\[
= \frac{1}{2} \mathcal{J}_x(f_{z/0})/0 + \frac{1}{2} \mathcal{J}_x(f_{z/0}/0/1) + \frac{1}{2} \mathcal{J}_x(f_{z/1/1/2}) + \frac{1}{2} \mathcal{J}_x(f_{z/1/1/2/1})
\]

\[
= \ldots
\]

\[
= \frac{1}{2^m} \sum_{u \in \{0,1\}^{m \setminus \{x\}}} \mathcal{J}_x(f_u).
\]

Since \( u \) assigns all variables except for \( x \) a value, \( f_u \) can only depend on \( x \). So there are four possible functions that \( f_u \) can represent: \( f = x, f = \overline{x}, f = 0 \) or \( f = 1 \). In the latter two cases, use DIC to get \( \mathcal{J}_x(f_u) = 0 \). In the former two cases, we can apply DIC to obtain \( \mathcal{J}_x(f_u) = 1 \). But then \( \mathcal{J}_x(f_u) = D_x f(u) \) and therefore \( \mathcal{J}_x(f) = I_x(f) \) after writing \( \mathcal{J}_x(f) \) as a sum of assignments over \( X \) instead of \( X \setminus \{x\} \).

A different axiomatic characteristic of the influence uses the relaxed cofactor-additvity property, which is defined as follows. Say that a value function \( \mathcal{J} \) is relaxed cofactor-additive if there are \( \alpha, \beta \in \mathbb{R} \) for every \( z \in X \) such that for all \( x \neq z \):

\[
\mathcal{J}_x(f) = \alpha_x \mathcal{J}_x(f_{z/0}) + \beta_x \mathcal{J}_x(f_{z/1}).
\]

If we additionally assume that \( \mathcal{J} \) satisfies TYPE, DIC and DUM, then \( \alpha = \beta = 1/2 \), and so it must be the influence:

Corollary 1. A value function \( \mathcal{J} \) satisfies TYPE, DIC, DUM, and relaxed cofactor-additivity if and only if \( \mathcal{J} = I \).

Proof. The direction from right to left is clear since the influence is a cofactor-additive IVF (cf. Theorem 4). For the direction from left to right, we only need to show that \( \alpha = \beta = 1/2 \) holds: since \( \mathcal{J} \) is then cofactor-additive, Theorem 4 implies that it must be the influence.

Pick an arbitrary \( z \in X \) and choose \( x \neq z \). Apply relaxed cofactor-additivity and DIC to the dictator function:

\[
\mathcal{J}_x(x) = \alpha \mathcal{J}_x(x) + \beta \mathcal{J}_x(x) \implies \beta = \alpha = 1.
\]

Moreover, note that due to relaxed cofactor-additivity, DIC and DUM,

\[
\mathcal{J}_x(xz) = \alpha \mathcal{J}_x(x) + \beta \mathcal{J}_x(0) = \alpha_z,
\]

\[
\mathcal{J}_x(x\overline{z}) = \alpha \mathcal{J}_x(0) + \beta \mathcal{J}_x(x) = \beta_z.
\]

The application of TYPE yields \( \mathcal{J}_x(xz) = \mathcal{J}_x(x\overline{z}) \), and so: \( \alpha_z = \beta_z = 1/2 \).

The following connection to the two-sided Jeroslav-Wang heuristic is established by counting the number of satisfying assignments. This heavily relies on the fact that every pair of clauses is distinguished by a literal that occurs negatively in one and positively in the other:

Theorem 5. Let \( D \) be a family of sets of literals such that all of its elements are non-trivial, and let \( x \) be variable such that \( D \) is \( x \)-orthogonal. Then:

\[
I_x(\bigvee_{C \in D} \bigwedge_{\eta \in C} \eta) = 2 \cdot JW_x(D).
\]

Proof. For a set of literals \( C \), denote their conjunction as \( \pi_C = \bigwedge_{\eta \in C} \eta \). We can write \( f = \bigvee_{C \in D} \pi_C \) as

\[
f = h_1 \lor xh_2 \lor \overline{x} h_3 = x(h_1 \lor h_2) \lor \overline{x}(h_1 \lor h_3),
\]

where

\[
h_1 = \bigvee_{C \in D} x \pi_C \pi_C \backslash \overline{x},
\]

\[
h_2 = \bigvee_{C \in D} x \pi_C \pi_C(\overline{x}),
\]

\[
h_3 = \bigvee_{C \in D} \pi_C \pi_C \backslash x.
\]
Let us first define

\[ I_x(f) = E[(h_1 \land h_2) \lor (h_1 \land h_3)], \]

which can be written as

\[ I_x(f) = E[h_1 h_2 h_3] + E[h_1 h_2 h_3]. \]

The goal is now to show that \( h_i h_j \) is equal to \( h_i \) for \( i, j \in \{1, 2, 3\}, i \neq j \). Using the distributivity law, write the conjunction of \( h_i \) and \( h_j \) as

\[ h_i h_j = \bigvee_{C \in I} \bigwedge_{C' \in J} \pi C \pi C', \]

where \( I \) and \( J \) are chosen accordingly from \( \{C \in D: x, \overline{x} \notin C\}, \{C \setminus \{x\}: x \in C \in D\} \) and \( \{\{x\}: x \in C \in D\} \). Since \( D \) is \( x \)-orthogonal, for every \( C \in I \) and \( C' \in J \), there is a literal \( \eta \) such that \( \eta \in C \) and \( \overline{\eta} \in C' \). Thus, every inner term \( \pi C \pi C' \) must evaluate to zero. But then \( h_i h_j = 0 \), and we can see that \( h_i h_j = h_i \) is implied. This allows the simplification

\[ I_x(f) = E[h_3] + E[h_2]. \]

Let us now compute the expected value of \( h_2 \), and, by a symmetric argument, that of \( h_3 \). Again, note that for \( C, C' \in D \) with \( x \in C \) and \( x \in C' \) and \( C \neq C' \), it must be the case that \( \pi C \setminus \{x\} \pi C' \setminus \{x\} = 0 \), thus implying \( \pi C \setminus \{x\} \pi C' \setminus \{x\} = \pi C \setminus \{x\} \).

Therefore, for every \( C' \in D \) with \( x \in C' \),

\[ E[h_2] = E[\bigvee_{C \in D, x \in C} \pi C \setminus \{x\}] = E[\pi C \setminus \{x\}] + E[\bigvee_{C \in D \setminus \{C\}} x \in C \pi C \setminus \{x\}] = E[\pi C \setminus \{x\}] + E[\bigvee_{C \in D \setminus \{C\}} x \in C \pi C \setminus \{x\}] = \cdots = E[\pi C \setminus \{x\}]. \]

For \( C \in D \) with \( x \in C \), the number of satisfying assignments of \( \pi C \setminus \{x\} \) is simply \( 2^{n-(|C|-1)} \). Thus, \( E[\pi C \setminus \{x\}] = 2^{-|C|+1} \).

We can now apply a symmetric argument to \( h_3 \), resolve the above equation of \( I_x(f) \) and then apply the fact that all \( C \in D \) are non-trivial to get

\[ I_x(f) = 2 \cdot \sum_{C \in D, x \in C} \pi C \setminus \{x\} 2^{-|C|} = 2 \cdot JW_x(D). \]

The following proposition shows that every Boolean function \( f \) can be represented by DNF made up of a family of sets of literals \( D \) that satisfies the premise in Theorem 5. Say that a family of sets of literals \( A \) is a DNF representation of \( f \) if \( f = \bigvee_{C \in A} \bigwedge_{\eta \in C} \eta \). Then:

**Proposition 8.** Suppose \( f \) is a Boolean function and \( x \) a variable. Then there exists a family of sets of literals \( D \) such that (i) \( D \) is a DNF representation of \( f \), (ii) all \( C \in D \) are non-trivial and (iii) \( D \) is \( x \)-orthogonal.

**Proof.** Let us first define \( R \) as the family of sets of literals that corresponds to the canonical DNF of \( f \), i.e.

\[ R = \{\{z: u(z) = 1\} \cup \{\overline{x}: u(z) = 0\} : u \in \{0, 1\}^X, f(u) = 1\}. \]

We now resolve all sets of literals \( C \) for which \( C \cup \{x\} \in R \) and \( C \cup \{\overline{x}\} \in R \) holds:

\[ D = \{C \in R: x, \overline{x} \notin C \} \cup \{C \setminus \{x\}, C \setminus \{\overline{x}\} \} \in R \}. \]

For example, if \( f = y \land (x \lor z) \), then \( R = \{\{y, x, \overline{z}\}, \{y, x, z\}, \{y, \overline{x}, z\}\} \) and \( D = \{\{y, x, \overline{z}\}, \{y, z\}\} \).

If \( A \) is a DNF representation of \( f \) and \( C \) a set of literals such that \( C \cup \{x\}, C \cup \{\overline{x}\} \in A \), then note that both \( C \cup \{x\} \) and \( C \cup \{\overline{x}\} \) can be resolved to \( C \setminus \{x\} \) without changing the fact that the resulting family is a DNF representation of \( f \). More precisely, \( (A \setminus \{C \cup \{x\}, C \cup \{\overline{x}\}\}) \cup (C \setminus \{x\}) \) must also be a DNF representation of \( f \). Repeatedly applying this argument for all \( C \) such that \( C \cup \{x\}, C \cup \{\overline{x}\} \in R \) shows that \( D \) must indeed be a DNF representation of \( f \), i.e. satisfy (i). For (ii), note that there is no \( C \in R \) that is trivial. Since every set in \( D \) is also a subset of a set in \( R \), no element of \( D \) can be trivial. Finally, for (iii), note that \( R \) is orthogonal, i.e. for all \( C, C' \in R, C \neq C' \) there exists a literal \( \eta \) such that \( \eta \in C \) and \( \overline{\eta} \in C' \). After resolving all sets that differ only in \( \eta \in \{x, \overline{x}\} \), all resulting pairs of sets must differ in at least one literal not in \( \{x, \overline{x}\} \). Thus, \( D \) must be \( x \)-orthogonal.
Cooperative Game Mappings

Lemma 2. If $\tau$ is an importance-inducing CGM and $E$ an expectation of contributions, then $E \circ \tau$ is an IVF. If $\tau$ is unbiased/derivative dependent, then so is $E \circ \tau$. Finally, if $\tau$ is (weakly) chain-rule decomposable, then so is $Bz \circ \tau$.

Proof. Let $c(0), \ldots, c(n-1) \in \mathbb{R}_{\geq 0}$ be $E$’s weights. Let us first show that $E \circ \tau$ is indeed an importance value function.

**BOUND** Suppose $f$ is a Boolean function and $x$ a variable. We want to show $0 \leq (E \circ \tau)_x(f) \leq 1$. First, $0 \leq (E \circ \tau)_x(f)$ follows from the fact that $E \circ \tau$ is a linear combination of non-negative terms due to $c$’s non-negativity and $0 \leq \partial_x \tau_I$. Since $\partial_x \tau_f \leq 1$,

$$(E \circ \tau)_x(f) \leq \sum_{S \subseteq X \setminus \{x\}} c(|S|) = 1.$$

**DUM** Suppose $x \not\in \text{dep}(f)$. We want to show $(E \circ \tau)_x(f) = 0$. Since in that case $\partial_x \tau_f(S) = 0$ for all $S \subseteq X$ due to DUMCG, mind that $(E \circ \tau)_x(f)$ must also be zero since it is a linear combination of $\partial_x \tau_f(S)$ for $S \subseteq X \setminus \{x\}$.

**DIG** Suppose $x$ is a variable. We want to show $(E \circ \tau)_x(S) = 1$. $(E \circ \tau)_x(S) = 1$ then follows from TYPE. Since $\tau$ satisfies DIG, note that $\partial_x \tau_x(S) = 1$ for all $S \subseteq X$. Now use the fact that $(E \circ \tau)_x(f)$ is a convex combination of $\partial_x \tau_x(S) = 1$’s, which implies $(E \circ \tau)_x(f) = 1$.

**TYPE** Suppose $f$ is a Boolean function, $x, y$ variables and $\sigma$ is a permutation. We want to show that (i) $(E \circ \tau)_x(f) = (E \circ \tau)_{\sigma(x)}(\sigma(f))$ and (ii) $(E \circ \tau)_x(f) = (E \circ \tau)_x(f[y/g])$.

For (i), note that $\partial_x \tau_f(S) = \partial_{\sigma(x)} \tau_{\sigma f}(\sigma(S))$ due to TYPECG. Thus, since $|S| = |\sigma(S)|$,

$$(E \circ \tau)_x(f) = \sum_{S \subseteq X \setminus \{x\}} c(|\sigma(S)|) \partial_{\sigma(x)} \tau_{\sigma f}(\sigma(S))$$

as the mapping $S \mapsto \sigma(S)$ constitutes a bijection that takes $2^{X \setminus \{x\}}$ to $2^{S \setminus \{x\}}$.

For (ii), note that $\tau_f = \tau_f[y/g]$ implies $\partial_x \tau_f = \partial_x \tau_f[y/g]$, and thus $(E \circ \tau)_x(f) = (E \circ \tau)_x(f[y/g])$.

**MODEC** We can see that the premise, i.e. the requirements that $f$ encapsulates $h$ on $g$ and $x \in \text{dep}(g)$, is exactly the same as in MODECG. Therefore, it suffices to pick such $f, g, h$ and $x$ and then apply MODECG to obtain $\partial_x \tau_f \geq \partial_x \tau_h$. Using non-negativity of $c$, we can see that $(E \circ \tau)_x(f) \geq (E \circ \tau)_x(h)$ must indeed hold.

For the second part,

- If $\tau$ is unbiased, then so is $E \circ \tau$: Suppose $g$ is a Boolean function and $x$ a variable. We can clearly see that $(E \circ \tau)_x(g) = (E \circ \tau)_x(g)$ must hold due to the fact that $\partial_x \tau_f = \partial_x \tau_f$ is implied by $\tau_g = \tau_g$.

- If $\tau$ is derivative dependent, then so is $E \circ \tau$: Let $f, h$ be Boolean functions such that $D_x f \geq D_y h$. Since $\tau$ is derivative dependent, we can see that $\partial_x \tau_f \geq \partial_x \tau_h$ must hold. Use $c$’s non-negativity to see that $c(|S|) \partial_x \tau_f(S) \geq c(|S|) \partial_x \tau_h(S)$ is the case for all $S \subseteq X \setminus \{x\}$. Application of $E$’s definition shows $(E \circ \tau)_x(f) \geq (E \circ \tau)_x(h)$.

- If $\tau$ is (weakly) chain-rule decomposable, then so is $Bz \circ \tau$: First note that if $v$ is a cooperative game and $\text{dep}(v) \subseteq R$, then $E_{S \subseteq X}[v(S)] = E_{S \subseteq R}[v(S)]$. This follows from Lemma 5, when viewing $2^X$ as $(\{0, 1\})^X$. Denote this by $(*).$ Let us now show the original claim. Suppose that $f$ is (monotonically) modular in $g$ and $x \in \text{dep}(g)$. Then

$$(Bz \circ \tau)_x(f) = E_{S \subseteq X \setminus \{x\}}[(\partial_x \tau_f)(S)]$$

by definition

$$(weak\ chain\ rule\ decomposability, \ (*)$$

DUMCG

$$(uniform\ distribution)$$

$$(\text{independent\ random\ variables})$$

The fourth equality is true since

\[
\begin{align*}
\text{dep}(g) \cap T &= \emptyset \implies \text{dep}(\tau_g) \cap T = \emptyset \\
&\implies \text{dep}(\partial_x \tau_g) \cap T = \emptyset \\
&\implies \partial_x \tau_g(R \cup \{z\}) = \partial_x \tau_g(R \setminus \{z\}) \quad \forall R \subseteq X, z \in T \\
&\implies \partial_x \tau_g(S \cup T) = \partial_x \tau_g(S).
\end{align*}
\]

$\text{dep}(\tau_g) \subseteq \text{dep}(g)$ by DUMCG

$\text{dep}(\partial_x \tau_g) \subseteq \text{dep}(\tau_g)$
The identity $\partial_y \tau_f(S \cup T) = \partial_y \tau_f(T)$ applies for a similar reason. Set $\ell = f[g/x_g]$. Then:

$$(X \setminus \text{dep}(g)) \cap S = \emptyset \implies \text{dep}(\ell) \cap S = \emptyset$$

$$(X \setminus \text{dep}(g)) \cap S = \emptyset \implies \text{dep}(\tau_\ell) \cap S = \emptyset$$

For the seventh equality, note that we can apply (\ast) due to $x \not\in \text{dep}(\partial_x \tau_g)$ and $x_g \not\in \text{dep}(\partial_x \tau_\ell)$.

**Simple Satisfiability-Biased Cooperative Game Mappings**

If $\omega$ is the dominating cooperative game mapping (cf. Definition 12), then we will also sometimes use the following inequalities, which are easy to show: If $s$ and $t$ are Boolean functions such that $s \geq t$, then $\omega_s \geq \omega_t$. Furthermore, one can then see that $\omega_s \lor \omega_t = \omega_s$. The following decompositions are helpful as well:

**Lemma 8.** Suppose $f$ is a Boolean function. Then for all $S \subseteq X$ and variables $z$,

$$\omega_f(S \cup \{z\}) = (\omega_{f_{z/1}} \lor \omega_{f_{z/0}})(S) \quad (C.7)$$

$$\omega_f(S \setminus \{z\}) = \omega_{f_{z/1} \land f_{z/0}}(S) \quad (C.8)$$

**Proof.** For (C.7), note that

$$\omega_f(S \cup \{z\}) = \exists c \in \{0, 1\}. \exists u \in \{0, 1\}^S \forall v \in \{0, 1\}^{X \setminus (S \cup \{z\})}. f(z/c; u; v).$$

$$= \exists c \in \{0, 1\}. \exists u \in \{0, 1\}^S \forall v \in \{0, 1\}^{X \setminus S}. f_z/c(u; v).$$

$$= \omega_{f_{z/1}}(S) \lor \omega_{f_{z/0}}(S).$$

And for (C.8),

$$\omega_f(S \setminus \{z\}) = \exists u \in \{0, 1\}^S \forall v \in \{0, 1\}^{X \setminus (S \cup \{z\})}. \forall c \in \{0, 1\}. f(z/c; u; v).$$

$$= \exists u \in \{0, 1\}^S \forall v \in \{0, 1\}^{X \setminus S}. f_{z/1}(u; v) \land f_{z/0}(u; v).$$

$$= \omega_{f_{z/1} \land f_{z/0}}(S).$$

The following decomposition of $\omega$ might be useful even in other contexts. (For example, for solving problems in $\Sigma^g_p$.) Note that the decomposition provided in Lemma 9 is essentially a case-distinction that distinguishes between all four values that $\omega_g(S)$ and $\omega_g(S)$ can take.

**Lemma 9.** If $f$ is modular in $g$ with positive and negative cofactor $s$ and $t$, then

$$\omega_f = \omega_g \omega_s \lor \omega_t \lor \omega_g \omega_s \lor \omega_g \omega_t \lor \omega_g \omega_s \lor \omega_g \omega_t.$$

Moreover, if $f$ is monotonically modular in $g$, then this simplifies to $\omega_g \omega_s \lor \omega_g \omega_t$.

**Proof.** Choose $S \subseteq X$ arbitrarily and define $U = S \cap (X \setminus \text{dep}(g))$ and $V = S \cap \text{dep}(g)$. This partitions $S$ into $S = U \cup V$ such that $s$ and $t$ only depend on variables in $U$ and $g$ only depends on variables in $V$. Thus,

$$\omega_f(S) = 1 \iff \text{there are } u \in \{0, 1\}^U, v \in \{0, 1\}^V \text{ such that } f(u; v) = g_u s_u \lor \overline{g_u} t_u = 1.$$

Furthermore, let us define:

$$v = \omega_g \omega_s \lor \omega_t \lor \omega_g \omega_s \lor \omega_g \omega_t \lor \omega_g \omega_s \lor \omega_g \omega t.$$

We want to establish $\omega_f = v$ by showing both directions $\omega_f(S) \leq v(S)$ and $\omega_f(S) \geq v(S)$:

For “$\leq$,” assume $\omega_f(S) = 1$, i.e. the existence of $u \in \{0, 1\}^U, v \in \{0, 1\}^V$ such that $g_u s_u \lor \overline{g_u} t_u = 1$. Let us distinguish into the following cases,

$g_u$ is one everywhere: Then $s_u$ must be one everywhere. But this implies $\omega_g \omega_s(S) = 1$ and therefore $v(S) = 1$.

$g_u$ is zero everywhere: Then $t_u$ and $\overline{g_u}$ must be one everywhere. But this implies $\omega_g \omega_t(S) = 1$ and therefore $v(S) = 1$. 

...
\( g_v \) is not constant: Then there are \( w_1, w_2 \) over \( \text{dep}(g) \setminus V \) such that \( w_1 \) satisfies \( w_2 \) falsifies \( g_v \). However, due to \( f_{u:v:w_1} = f_{u:v:w_2} = 1 \), note that the first case implies \( s_u = 1 \) and the second \( t_u = 1 \). Thus, \( \omega_{st}(U) = 1 \), which also establishes \( \omega_s(U) = \omega_l(U) = (\omega_s \lor \omega_l)(U) = 1 \) due to \( st \leq s \), \( st \leq t \) and \( st \leq v \). This implies \( v(S) = 1 \) as well.

For \( \geq \), assume \( v(S) = 1 \), and let us distinguish into the following cases,
\[
\omega_g \overrightarrow{\omega}(V) = 1: \text{Then } (\omega_s \lor \omega_l)(U) = 1. \text{ If } \omega_s(U) = 1, \text{ then } \omega_{gs}(S) = 1 \text{ and } g \leq f \text{ implies } \omega_f(S) = 1. \text{ If } \omega_l(U) = 1, \text{ then } \omega_{gl}(S) = 1 \text{ and } \overrightarrow{gt} \leq f \text{ implies } \omega_f(S) = 1.
\]
\[
\omega_g \overleftarrow{\omega}(V) = 1: \text{Symmetric.}
\]
\[
\omega_g \overrightarrow{\omega}(V) = 1: \text{Then } \omega_{st}(U) = 1. \text{ Note that } st \leq f, \text{ which implies } \omega_f(S) = 1.
\]

For the simplified formula, suppose that \( f \) is monotonically modular in \( g \). Mind that \( s \geq t \) implies \( \omega_s \lor \omega_l = \omega_s \text{ and } \omega_{st} = \omega_l \). But then \( v \) reduces to \( \omega_g \omega_s \lor \overleftarrow{\omega_g \omega_l} \).

We prove the following proposition using Lemma 8, by checking that every property is satisfied.

**Theorem 6.** The dominating CGM is weakly chain-rule decomposable and importance inducing.

**Proof.**

**BOUNDCG** Let \( f \) be a Boolean function and \( x \) be a variable. We want to show \( 0 \leq \partial_x \omega_f \leq 1 \). Since \( \omega_f(S) \in \{0, 1\} \) for all \( S \subseteq X \), we can easily show that \( \partial_x \omega_f \leq 1 \). Furthermore, note that for all \( S \subseteq X \), by Lemma 8,
\[
\omega_f(S \cup \{x\}) = (\omega_{fx/1} \lor \omega_{fx/0})(S) \geq \omega_{fx/1}(S) \geq \omega_{fx/0}(S) = \omega_f(S \setminus \{x\}),
\]
which implies \( 0 \leq \partial_x \omega_f \).

**DIACC** Let \( x \) be a variable. We want to show \( \partial_x \omega_x = 1, \partial_x \omega_\tau = 1 \) then follows from \( \text{TYPECG} \). Note that \( \omega_x(S) = 1 \)
if and only if \( x \in S \) for \( S \subseteq X \). Thus, \( \partial_x \omega_x(S) = 1 \) for all \( S \subseteq X \).

**DUMCG** Let \( f \) be a Boolean function and \( x \notin \text{dep}(f) \) be a variable. We want to show \( \partial_x \omega_f = 0 \). Since \( f = f_{x/1} = f_{x/0} \), we simply apply Lemma 8 and get \( \omega_f(S \cup \{x\}) = \omega_f(S \setminus \{x\}) \) and thus \( \partial_x \omega_f(S) = 0 \) for all \( S \subseteq X \).

**TYPECG** Suppose \( f \) is a Boolean function, \( \sigma \) a permutation and \( x, y \) variables. Note that for \( \omega_f(S) = \omega_{\sigma f}(\sigma(S)) \),
\[
\omega_{\sigma f}(S) = \exists u \in \{0, 1\}^S, \forall v \in \{0, 1\}^{X \setminus S}. f(u; v)
\]
\[
= \exists u \in \{0, 1\}^S, \forall v \in \{0, 1\}^{X \setminus S}. f(\sigma^{-1} u; \sigma^{-1} v)
\]
\[
= \exists u \in \{0, 1\}^{\sigma^{-1} S}, \forall v \in \{0, 1\}^{X \setminus \sigma^{-1} S}. f(u; v)
\]
\[
= \omega_f(\sigma^{-1}(S)) \quad (A.1)
\]
This is the same as asking for \( \omega_f(S) = \omega_{\sigma f}(\sigma(S)) \) for all \( S \subseteq X \). For \( \omega_f(S) = \omega_{f_{y/1}}(S) \), note that we can apply (A.1) for \( u \mapsto u \oplus 1_y \) in an analogous way.

**MODECG** Suppose that (i) \( f, h \) are monotonically modular in \( g \) such that (ii) \( f_{g/1} \geq h_{g/1} \) and \( h_{g/0} \geq f_{g/0} \) and (iii) \( x \in \text{dep}(g) \). We want to show \( \partial_x \omega_f \geq \partial_x \omega_{h_l} \). By Lemma 9,
\[
\omega_f = \omega_g \omega_{f_{g/1}} + (1 - \omega_g) \omega_{f_{g/0}}
\]
Therefore,
\[
\partial_x \omega_f = (\partial_x \omega_g)(\omega_{f_{g/1}} - \omega_{f_{g/0}}),
\]
\[
\partial_x \omega_{h_l} = (\partial_x \omega_g)(\omega_{h_{g/1}} - \omega_{h_{g/0}}).
\]
By using \( \omega_{f_{g/1}} \geq h_{g/1} \) and \( \omega_{h_{g/0}} \geq f_{g/0} \) and \( \omega_{f_{g/1}} \geq f_{g/0} \) and \( \omega_{h_{g/0}} \geq h_{g/0} \) (due to (i) and (ii)) and \( \partial_x \omega_g \geq 0 \), we obtain \( \partial_x \omega_f \geq \partial_x \omega_{h_l} \).

Finally, to see that \( \omega \) is weakly chain-rule decomposable, suppose that \( f \) is monotonically modular in \( g \) and \( x \in \text{dep}(g) \). The same decomposition as in the proof for \( \text{MODECG} \) applies, i.e.
\[
\partial_x \omega_f = (\partial_x \omega_g)(\omega_{f_{g/1}} - \omega_{f_{g/0}}).
\]
Denote \( \ell = f_{g/x_g} \) and note that for all \( S \subseteq X \), since \( x_{g/1} \geq f_{g/1} \geq f_{g/0} = \ell_{x_g/0} \) and due to Lemma 8,
\[
\omega_{f_{g/1}}(S) = (\omega_{\ell_{x_g/1}} \lor \omega_{\ell_{x_g/0}})(S) = \omega_{f_{g/1}}(S \cup \{x_g\})
\]
\[
\omega_{f_{g/0}}(S) = \omega_{\ell_{x_g/1} \lor \ell_{x_g/0}}(S) = \omega_{f_{g/0}}(S \setminus \{x_g\}),
\]
which suffices to show \( \omega_{f_{g/1}} - \omega_{f_{g/0}} = \partial_x \omega_{\ell} = \partial_x \omega_f \). Thus, \( \partial_x \omega_f = (\partial_x \omega_g)(\partial_x \omega_f) \).
The proof of Proposition 3 uses the fact that \( \omega \) can be expressed in terms of \( \nu \). The additional condition \( c(k) = c(n-1-k) \) then allows us to rearrange the terms of the sum of the expectation of contributions:

**Proposition 3.** Let \( \mathcal{E} \) be an expectation of contributions with \( c(k) = c(n-1-k) \) for all \( k \in \{0, \ldots, n-1\} \). Then for all \( g \in \mathbb{B}(X) \) and \( x \in X \):

\[
(\mathcal{E} \circ \omega)_x(g) = (\mathcal{E} \circ \nu)_x(\mathcal{F})
\]

**Proof.** Note that \( \omega_x(S) = 1 - \nu_x(X \setminus S) \). Thus, \( \partial_x \omega_x(S) = \partial_x \nu_x(X \setminus S) \) and \( \partial_x \nu_x(S \setminus \{x\}) \) since \( \partial_x \nu_x \) does not depend on \( x \). Therefore,

\[
(\mathcal{E} \circ \omega)_x(g) = \sum_{S \subseteq X \setminus \{x\}} c(|S|) \partial_x \omega_x(S)
= \sum_{S \subseteq X \setminus \{x\}} c(|S|) \partial_x \nu_x(X \setminus S)
= \sum_{S \subseteq X \setminus \{x\}} c(|S \setminus \{x\}|) \partial_x \nu_x((X \setminus \{x\}) \setminus S)
= \sum_{S \subseteq X \setminus \{x\}} c(|S|) \partial_x \nu_x(S)
= (\mathcal{E} \circ \nu)_x(\mathcal{F}),
\]

due to the fact that \( n-1 - |S| = |(X \setminus \{x\}) \setminus S| \) for \( x \notin S \) and that the mapping \( S \mapsto (X \setminus \{x\}) \setminus S \) constitutes a bijection from \( 2^X \setminus \{x\} \) to itself. (Note that \( (X \setminus \{x\}) \setminus S \) is just the complement of \( S \) w.r.t. \( X \setminus \{x\} \)).

**Proposition 4.** Let \( f \) be a monotone Boolean function and \( x \) a variable. Then \( (Bz \circ \omega)_x(f) = (Bz \circ \nu)_x(f) = I_x(f) \).

**Proof.** Recall that if \( f \) is monotone in \( z \), then by Lemma 8,

\[
\omega_f(S) = (\omega_{f_{z/1}} \lor \omega_{f_{z/0}})(S) = \omega_{f_{z/1}}(S) = \omega_{f_{z/0}}(S \setminus \{z\}) \quad \text{if } z \in S
\]

\[
\omega_f(S) = \omega_{f_{z/1} \land f_{z/0}}(S) = \omega_{f_{z/0}}(S) \quad \text{if } z \notin S
\]

due to \( z \notin \text{dep}(f_{z/1}) \). Since \( f \) is monotone in all of its variables, we repeatedly apply this argument for all variables except for \( x \) to get

\[
(Bz \circ \omega)_x(f) = \frac{1}{2^{n-1}} \sum_{S \subseteq X \setminus \{x\}} \omega_f(S \cup \{x\}) - \omega_f(S)
= \frac{1}{2^{n-1}} \sum_{S \subseteq X \setminus \{x\}} \omega_{f_{z/0}}(\{x\}) - \omega_{f_{z/0}}(\emptyset)
= \frac{1}{2^{n-1}} \sum_{u \in \{0,1\}^{X \setminus \{x\}}} \omega_{f_{z/0}}(\{x\}) - \omega_{f_{z/0}}(\emptyset).
\]

By monotonicity of \( f \), we have three options for \( f_u \) if \( u \) is an assignment over \( X \setminus \{x\} \): either \( f_u(x) = x, f_u = 0 \) or \( f_u = 1 \). In the first case, \( \omega_{f_u}(\{x\}) = 1 \) and \( \omega_{f_u}(\emptyset) = 0 \). In the last two, \( \omega_{f_u}(\{x\}) = \omega_{f_u}(\emptyset) = 0 \). Therefore, \( \omega_{f_u}(\{x\}) - \omega_{f_u}(\emptyset) = D_x f(u) \), and we get \( (Bz \circ \omega)_x(f) = I_x(f) \). Finally, since (i) the influence does not distinguish between a function and its dual by Theorem 1, (ii) the dual of a monotone boolean function is itself monotone, (iii) TYPE and (iv) Proposition 3,

\[
I_x(f) = \left( I_x(f) \right)_{y \in X} \quad \text{and} \quad \left( Bz \circ \omega \right)_x(f) = \left( Bz \circ \nu \right)_x(f).
\]

**A Constancy-Based Cooperative Game Mapping**

The first four properties of importance inducing CGMs hold for Hammer, Kogan and Rothblum’s CGM under all constancy measures \( \kappa \), since we only have to rely on general properties of constancy measures. The “trick” for \( \kappa = \kappa_{\text{quad}} \) is that the quantity \( \frac{1}{2} \kappa(a) + \frac{1}{2} \kappa(b) - \kappa(a+b) + \frac{1}{2} b \) can be expressed as \( (a-b)^2 \), which in turn implies properties such as \( \text{ModEC} \) for \( \text{Ham}^n \):

**Theorem 7.** Suppose \( \kappa \) is a constancy measure. Then \( \text{Ham}^n \) is an unbiased CGM that satisfies \( \text{BOUND} \), \( \text{DIC} \), \( \text{DUM} \), and \( \text{TYPE} \). Further, \( \text{Ham}^n_{\text{quad}} \) is chain-rule decomposable and satisfies \( \text{ModEC} \).
Proof.

**Unbiasedness.** Suppose that $g$ is a Boolean function and $x$ a variable. We want to show $H_g^\kappa = H_{\overline{g}}^\kappa$. Note that for all $S \subseteq X$ and assignments $\alpha$ over $S$,
\[ \kappa(E_g[\alpha]) = \kappa(E[1 - \overline{g}\alpha]) = \kappa(1 - E_g[\alpha]) = \kappa(E_{\overline{g}}[\alpha]) \]
due to $\kappa(\alpha) = \kappa(1 - \alpha)$. But then it is easy to see that $H_g^\kappa(S) = H_{\overline{g}}^\kappa(S)$ for all $S \subseteq X$.

**BoundCG.** Suppose $f$ is a Boolean function and $x$ a variable. We want to show $0 \leq \partial_x H_f^\kappa \leq 1$. For $S \subseteq X$, note that $H_f^\kappa(S)$ is a convex combination of terms limited by 0 and 1, so $\partial_x H_f^\kappa(S) \leq 1$. The requirement $\partial_x H_f^\kappa \geq 0$ states that $H_f^\kappa$ is a monotone cooperative game, which was already established by [HKR00].

**DiCG.** Suppose that $x$ is a variable. We want to show $\partial_x H_f^\kappa = 1$. Then follows from **TypeCG**. Note that if $S \subseteq X$ and $\alpha$ is an assignment over $S$ and $f = x$, it is the case that $E[f] = \frac{1}{2}$ if $x \notin S$ and $E[f] \in \{0, 1\}$ if $x \in S$. Thus, due to $\kappa(\frac{1}{2}) = 0$ and $\kappa(0) = \kappa(1) = 1$, we have $\partial_x H_f^\kappa(S) = 1$.

**DumCG.** Suppose that $f$ is a Boolean function and $x \notin \text{dep}(f)$ a variable. Then for all $S \subseteq X$,
\[ \partial_x H_f^\kappa(S) = E_{\alpha \in \{0, 1\}^S \text{\ s.t.} \ \kappa(E[f]_{\alpha})} \left[ \frac{1}{2} \kappa(E[f_{x/1}\alpha]) + \frac{1}{2} \kappa(E[f_{x/0}\alpha]) - \kappa(E[f\alpha]) \right] \]
where $E[f_{x/1}\alpha] = E[f_{x/0}\alpha] = E[f\alpha]$ due to $f_{x/1} = f_{x/0}$. But then $\partial_x H_f^\kappa(S) = 0$.

**TypeCG.** Suppose that $f$ is a Boolean function, $\sigma$ a permutation and $x, y$ variables. We want to show (i) $H_f^\kappa(S) = H_{\sigma f}^\kappa(\sigma(S))$ and (ii) $H_f^\kappa(S) = H_{f[y/\overline{y}]}^\kappa(S)$ for all $S \subseteq X$.

For (i), take an arbitrary $S \subseteq X$ and assignment $\alpha$ over $S$. Then
\[ E_{\alpha \in \{0, 1\}^S} \left[ \kappa(\sigma f)[\sigma\alpha] \right] = E_{\alpha \in \{0, 1\}^S} \left[ \kappa(f)[\sigma^{-1}\alpha] \right] \]
Thus,
\[ H_f^\kappa(S) = E_{\alpha \in \{0, 1\}^S} \left[ \kappa(E[f\alpha]) \right] = E_{\alpha \in \{0, 1\}^S} \left[ \kappa(E[(\sigma f)\alpha]) \right] \]
\[ = E_{\alpha \in \{0, 1\}^S} \left[ \kappa(E[(\sigma^{-1} f)\alpha]) \right] = H_{\sigma f}^\kappa(\sigma(S)). \]

For (ii), take again $S \subseteq X$. Then, if $y \in S$,
\[ H_f^\kappa(S) = \frac{1}{2} H_{f[y/1]}^\kappa(S) + \frac{1}{2} H_{f[y/0]}^\kappa(S) \]
\[ = \frac{1}{2} H_{f[y/\overline{y}]}^\kappa(S) + \frac{1}{2} H_{f[y/\overline{y}]}^\kappa(S) \]
\[ = H_{f[y/\overline{y}]}^\kappa(S) \]
And if $y \notin S$, note that for all assignments $\alpha$ over $S$,
\[ E[f\alpha] = \frac{1}{2} E[f_{y/0}\alpha] + \frac{1}{2} E[f_{y/1}\alpha] \]
\[ = \frac{1}{2} E[(f[y/\overline{y}])_{y/1}\alpha] + \frac{1}{2} E[(f[y/\overline{y}])_{y/0}\alpha] \]
\[ = E[(f[y/\overline{y}])\alpha]. \]
This however implies
\[ H_f^\kappa(S) = E_{\alpha \in \{0, 1\}^S} \left[ \kappa(E[f\alpha]) \right] = E_{\alpha \in \{0, 1\}^S} \left[ \kappa(E[(f[y/\overline{y}])\alpha]) \right] \]
\[ = H_{f[y/\overline{y}]}^\kappa(S). \]

Let us now focus on the case where $\kappa = \kappa_{\text{quad}}$, i.e. $\kappa(\alpha) = 4(a - \frac{1}{2})^2$. We can then see that
\[ \frac{1}{2} \kappa(a) + \frac{1}{2} \kappa(b) - \kappa(\frac{1}{2}a + \frac{1}{2}b) = (a - b)^2. \]
Thus,
\[ \partial_x H_f^\kappa(S) = E_{\alpha \in \{0, 1\}^S} \left[ \kappa(E[f\alpha]) \right] - E_{\alpha \in \{0, 1\}^S} \left[ \kappa(E[f_1\alpha]) \right] \]
\[ = E_{\alpha \in \{0, 1\}^S} \left[ \frac{1}{2} \kappa(E[f_{x/1}\alpha]) + \frac{1}{2} \kappa(E[f_{x/0}\alpha]) - \kappa(E[f\alpha]) \right] \]
unwrapping $E$ with respect to $x$
\[ = E_{\alpha \in \{0, 1\}^S} \left[ \kappa(E[f_{x/1}\alpha]) - \kappa(E[f_{x/0}\alpha]) \right] \]
above fact and $E[f\alpha] = \frac{1}{2} E[f_{x/1}\alpha] + \frac{1}{2} E[f_{x/0}\alpha]$

independence of $x$

This allows us to derive the remaining properties, i.e. the fact that $H^\kappa$ is chain-rule decomposable and that it satisfies **MODECG**:.
D.1 Computation of Blame Values

Let us first show how one can compute blame values. We want to show \( \partial_{g} H^f_{\mathcal{J}} = (\partial_{g} H^f_{\mathcal{S}})(\partial_{g} H^f_{\mathcal{J}}) \). First note that for \( S \subseteq X \) and \( a;b \in \{0, 1\}^{S} \) with \( A = S \cap \text{dep}(g) \) and \( B = S \cap (X \setminus \text{dep}(g)) \), due to Lemma 5,

\[
E[f_{a;b}] = E[g_{a}] E[(f_{g/1})_{b}] + (1 - E[g_{a}]) E[(f_{g/0})_{b}]
\]

and therefore

\[
E[(f_{x/1} - f_{x/0})_{a;b}] = E[(g_{x/1} + (1 - g_{x/0}) f_{g/0} - (g_{x/0} f_{g/1} + (1 - g_{x/0}) f_{g/0}))_{a;b}] \quad f \text{ modular in } g
\]

\[
= E[(g_{x/1} - g_{x/0}) (f_{g/1} - f_{g/0})]_{a;b} \quad \text{simplifying}
\]

\[
= E[(g_{x/1} - g_{x/0}) a (f_{g/1} - f_{g/0})]_{b} \quad \text{independence}
\]

\[
= E[(g_{x/1} - g_{x/0}) a] \cdot E[(f_{g/1} - f_{g/0})]_{b} \quad \text{Lemma 5}
\]

which implies, using \( \ell = [f_{g/x}] \),

\[
\partial_{g} H^f_{\mathcal{J}}(S) = \sum_{a \in \{0, 1\}^{S \cap \text{dep}(g)}, b \in \{0, 1\}^{B \setminus \text{dep}(g)}} \left[ E \left[ (g_{x/1} - g_{x/0}) | a \right] \cdot E \left[ (f_{g/1} - f_{g/0}) | b \right] \right] \quad \text{uniform distribution}
\]

\[
= \sum_{a \in \{0, 1\}^{S \cap \text{dep}(g)}} \left[ E \left[ (g_{x/1} - g_{x/0}) | a \right] \cdot E \left[ (f_{g/1} - f_{g/0}) | a \right] \right] \cdot \text{independence}
\]

\[
= \sum_{a \in \{0, 1\}^{S \cap \text{dep}(g)}} \left[ E \left[ (g_{x/1} - g_{x/0}) | a \right] \cdot E \left[ \ell_{x/1} - \ell_{x/0} | a \right] \right] \quad \text{Lemma 5}
\]

\[
= \partial_{g} H^f_{\mathcal{S}}(S) \cdot \partial_{g} H^f_{\mathcal{J}}(S).
\]

**ModEC**

Suppose that (i) \( f, h \) are monotonically modular in \( g \), (ii) \( f_{g/1} \geq h_{g/1} \) and \( h_{g/0} \geq f_{g/0} \) and (iii) \( x \in \text{dep}(g) \). We want to show \( \partial_{g} H^f_{\mathcal{J}} \geq \partial_{g} H^h_{\mathcal{J}} \). Note that since \( \text{BOUNDCG} \) is satisfied and because \( H^f \) is chain-rule decomposable. It suffices to show \( \partial_{g} H^f_{\mathcal{J}}(S) \geq \partial_{g} H^h_{\mathcal{J}}(S) \) for all \( S \subseteq X \). We have seen that

\[
\partial_{g} H^f_{\mathcal{J}}(S) = \sum_{a \in \{0, 1\}^{S \cap \text{dep}(g)}} \left[ E \left[ (f_{g/1} - f_{g/0}) | a \right] \right].
\]

Note that due to \( f_{g/1} \geq h_{g/1} \) and \( h_{g/0} \geq f_{g/0} \) and \( h_{g/1} \geq h_{g/0} \),

\[
(f_{g/1} - f_{g/0})_{a} \geq (h_{g/1} - h_{g/0})_{a} \geq 0
\]

for every assignment \( a \) over \( S \). This shows \( E[(f_{g/1} - f_{g/0})_{a}] \geq E[(h_{g/1} - h_{g/0})_{a}] \), so \( \partial_{g} H^f_{\mathcal{J}}(S) \geq \partial_{g} H^h_{\mathcal{J}}(S) \) is true as well.

**D. Section 5, Computing Importance Values**

**D.1 Computation of Blame Values**

Suppose \( f \) is a Boolean function and \( x \) a variable. We are interested in computing the quantity \( B^f_x(f) \) for any share function \( \rho \). First mind that \( \text{scs}^m_x(f) \) can only take values in \( \{0, \ldots, n-1, \infty\} \) since there are not more than \( n-1 \) elements a candidate solution to \( \text{scs}^m_x(f) \) can contain. Counting the number of times \( \text{scs}^m_x(f) = k \) occurs, we obtain

\[
B^f_x(f) = \frac{1}{2^n} \sum_{u \in \{0, 1\}^{X}} \rho(\text{scs}^m_x(f))
\]

\[
= \frac{1}{2^n} \sum_{k=0}^{n-1} \left[ \rho(k) \cdot \left| \{u \in \{0, 1\}^X : \text{scs}^m_x(f) = k \} \right| \right]
\]

\[
= E[g_{f,x,0}] + \sum_{k=1}^{n-1} \rho(k) (E[g_{f,x,k}] - E[g_{f,x,k-1}]),
\]

where \( g_{f,x,k} \) is a Boolean function such that \( g_{f,x,k}(u) = 1 \) holds if and only if \( \text{scs}^m_x(f) \leq k \); so the number of its models must agree with the number of elements in the set \( \{u \in \{0, 1\}^X : \text{scs}^m_x(f) \leq k \} \).

**BDD-Based Approach**

Let us first show how one can compute \( g_{f,x,k} \) for \( B^f_x(f) \) using BDD-based representations. So suppose that \( f \) is represented by a BDD with an arbitrary order of variables. Using this representation, we recursively define the following Boolean function for \( k \in \{0, \ldots, n-1\} \) and \( c \in \{0, 1\} \):

\[
\gamma_{f,x,k}(u) = 1 \text{ if and only if } \text{scs}^m_x(f) \leq k.
\]

\[
\text{BDD-Based Approach}
\]

Let us first show how one can compute \( \gamma_{f,x,k} \) for \( B^f_x(f) \) using BDD-based representations. So suppose that \( f \) is represented by a BDD with an arbitrary order of variables. Using this representation, we recursively define the following Boolean function for \( k \in \{0, \ldots, n-1\} \) and \( c \in \{0, 1\} \):

\[
\gamma_{f,x,k}(u) = 1 \text{ if and only if } \text{scs}^m_x(f) \leq k.
\]
\[ \theta_{f,x,c,k} = \begin{cases} f \land \overline{f}[x/\overline{x}] & k = 0, c = 1 \\ f \lor \overline{f}[x/\overline{x}] & k = 0, c = 0 \\ \theta_{f,x,c,k-1} \lor \bigvee_{y \in X \setminus \{x\}} \theta_{f,x,c,k-1[y/\overline{y}\big]} & k > 0 \end{cases} \]

Note that for BDDs, flipping variables and complementing and composing them once(!) using \( \land, \lor \), etc., increases BDD sizes only polynomially and sometimes even decreases them. The last fact is especially important for \( k > 0 \), as it is often times the case that the BDD’s size decreases as \( k \) grows, which is due to the fact that \( \theta_{f,x,c,k} \) becomes more and more satisfiable. Other approaches, such as the representation of \( \theta_{f,x,c,k} \) by propositional formulas, do not have this advantage and will always “explode”. We now show that for all assignments \( u \),

\[ \theta_{f,x,c,k}(u) = 1 \iff \text{scs}_x^u(f,c) \leq k. \]

In order to see this, we need the following technical lemma:

**Lemma 10.** For all Boolean functions \( f \), variables \( x, z \), constants \( c \in \{0, 1\} \) and \( k \geq 0 \),

\[ \theta_{f,[z/\overline{z}],x,c,k} = \theta_{f,x,c,k}[z/\overline{z}]. \]

**Proof.** Note that for all Boolean functions \( \ell \) and \( h \), variables \( y \) and component-wise operations \( \circ \) (such as \( \land, \lor \), etc.),

\[ (\ell \circ h)[z/\overline{z}] = \ell[z/\overline{z}] \circ h[z/\overline{z}], \]

\[ \ell[y/\overline{y}][z/\overline{z}] = \ell[z/\overline{z}][y/\overline{y}]. \]

Using induction over \( k \), we can see that these facts suffice to get us \( \theta_{f,[z/\overline{z}],x,c,0} = \theta_{f,x,c,0}[z/\overline{z}] \) in the base case. For \( k > 0 \), use the induction hypothesis, which implies \( \theta_{f,[z/\overline{z}],x,c,k-1} = \theta_{f,x,c,k-1}[z/\overline{z}] \), and simplify according to the rules given above to get \( \theta_{f,[z/\overline{z}],x,c,k} = \theta_{f,x,c,k}[z/\overline{z}] \).

**Theorem 8.** For all Boolean functions \( f \), variables \( x, c \in \{0, 1\}, k \geq 0 \) and assignments \( u \),

\[ \theta_{f,x,c,k}(u) = 1 \iff \text{scs}_x^u(f,c) \leq k. \]

**Proof.** This can be shown by induction over \( k \). If \( k = 0 \), then \( \text{scs}_x^u(f,c) = 0 \) if and only if \( c = f(u) \neq f(u \oplus \overline{1}_x) \), which can be seen to be identical to asking for \( \theta_{f,x,c,0}(u) = 1 \). For a fixed \( k > 0 \), let us assume for the induction hypothesis that the claim of Theorem 8 already holds for \( k-1 \):

- \( \text{scs}_x^u(f,c) \leq k \)
- \( \theta_{f,x,c,k-1}(u) = 1 \) or \( \exists y \in X \setminus \{x\}, \text{scs}_x^u(f[y/\overline{y}],c) \leq k-1 \)
- \( \theta_{f,x,c,k-1}[y/\overline{y}](u) = 1 \) (induction hypothesis)
- \( \text{Lemma 10} \)
- \( \theta_{f,x,c,k}(u) = 1 \) (definition)

Here \((\ast)\) is true since for all \( S \subseteq X \setminus \{x\} \) and \( y \in S \),

\[ c = f(u \oplus \overline{1}_S) \neq f(u \oplus \overline{1}_S \oplus \overline{1}_x) \]

iff \( c = f[y/\overline{y}][u \oplus \overline{1}_S \setminus \{y\}] \neq f[y/\overline{y}][u \oplus \overline{1}_S \setminus \{y\} \oplus \overline{1}_x] \).

Thus, if \( S \) is a candidate solution to \( \text{scs}_x^u(f,c) \) of size \( k \), then \( S \setminus \{y\} \) is a candidate solution to \( \text{scs}_x^u(f[y/\overline{y}],c) \) of size \( k-1 \) and vice versa.

Now, let us turn back to the computation of CHK’s blame. To obtain a BDD that models \( \text{scs}_x^u(f,c) \leq k \), we set

\[ \gamma_{f,x,k} = f \theta_{f,x,1,k} \lor \overline{f} \theta_{f,x,0,k}, \]

which introduces a case distinction on \( f \). For the modified blame, we can use a similar approach and simply have to redefine \( \gamma_{f,x,k} \) as

\[ \gamma_{f,x,k} = \begin{cases} f \oplus f[x/\overline{x}] & k = 0 \\ \gamma_{f,x,k-1} \lor \bigvee_{y \in X \setminus \{x\}} \gamma_{f,x,k-1}[y/\overline{y}] & k > 0 \end{cases} \]
Projected Model Counting Based Approach

Let \( \phi \) be a propositional formula that represents \( f \) and suppose \( x \in \text{var}(\phi) = \Phi \) occurs in \( \phi \). The basis for this approach lays in the observation that we can devise a sequence of “short” propositional formulas \( \{ \chi_k \}_{k=0}^{n-1} \) with some extra variables \( N \subseteq X \setminus \Phi \) such that for all assignments \( u \) over \( \Phi \) and \( k \in \{0, \ldots, n-1\} \),

\[
\exists v \in \{0,1\}^N \cdot \chi_k(u; v) \iff \text{scs}_k^u(f) \leq k.
\]

Counting the number of \( u \)'s that satisfy the left hand side is done through projected model counting. The general idea is that these new variables contained in \( N \) certify the existence of a candidate solution \( S \) of size \( k \) or less for the right side. The definition of \( \chi_k \) contains roughly two parts: one that states that \( S \) cannot have more than \( k \) elements, and one that makes sure that \( S \) is indeed a candidate solution, i.e. satisfies \( \phi(u) = \phi(u \uplus 1_S) \neq \phi(u \uplus 1_S \uplus 1_x) \). The former part is ensured by using a cardinality constraint and the latter part by introducing a set of new variables that model the action of adding elements to \( S \).

First, introduce \( |\Phi| - 1 \) new variables by setting \( V = \{v_y: y \in \Phi \setminus \{x\}\} \subseteq X \setminus \Phi \). These variables are used in the replacement

\[
\psi = \phi[y/(y \uplus v_y): v_y \in V],
\]

which is easy to carry out on formulas and doubles \( \phi \)'s size in the worst case. Furthermore, we use the totalizer encoding [BB03] to devise a formula \( \text{atmost}_k \) that introduces \( O(n \log n) \) new variables denoted by the set \( W \) such that \( \text{var}(\text{atmost}_k) = W \cup V \) and

\[
\exists w \in \{0,1\}^W \cdot \text{atmost}_k(v; w) \iff |v| \leq k.
\]

for all assignments \( v \) over \( V \). The length of \( \text{atmost}_k \) is in the range \( O(n^2) \). Finally, we set:

\[
\chi_k(u; w; v) = \text{atmost}_k(v; w) \land (\phi(u) \iff \psi(u; v)) \land (\phi(u) \oplus \psi(x/x)(u; v)),
\]

where \( \iff \) denotes the syntactic equivalence. Note that the latter part increases \( \phi \)'s size only linearly. Thus, if \( m \) represents the size of \( \phi \), then this transformation creates a formula of size \( O(n^2 + m) \) and \( O(n \log n) \) variables. The following theorem now shows our original claim:

**Theorem 9.** For all \( k \in \{0, \ldots, n-1\} \) and assignments \( u \) over \( \Phi \), we have

\[
\exists v \in \{0,1\}^V, w \in \{0,1\}^W \cdot \chi_k(u; w; v) \iff \text{scs}_k^u(f) \leq k.
\]

**Proof.** Define \( \sigma: V \rightarrow \Phi \setminus \{x\}, v_y \mapsto y \). Then note that

\[
\exists v \in \{0,1\}^V, w \in \{0,1\}^W \cdot \chi_k(u; w; v)
\]

\[
\iff \exists v \in \{0,1\}^V, (|v| \leq k) \land (\phi(u) = \psi(u; v) \neq \psi(u \uplus 1_x; v)) \quad \text{(D.1)}
\]

\[
\iff \exists v \in \{0,1\}^V, (|\sigma v| \leq k) \land (\phi(u) = \phi(u \uplus (\sigma v; x/0)) \neq \phi((u \uplus 1_x) \uplus (\sigma v; x/0))) \quad \text{(D.2)}
\]

\[
\iff \exists v \in \{0,1\}^V, (|v| \leq k) \land (\phi(u) = \phi(u \uplus (v; x/0)) \neq \phi((u \uplus 1_x) \uplus (v; x/0))) \quad \text{(D.3)}
\]

\[
\iff \exists S \subseteq \Phi \setminus \{x\}, (|S| \leq k) \land (\phi(u) = \phi(u \uplus 1_S) \neq \phi(u \uplus 1_S \uplus 1_x)) \quad \text{(D.4)}
\]

\[
\iff \text{scs}_k^u(f) \leq k. \quad \text{(D.5)}
\]

Equivalence (D.1) moves the existential quantification of \( w \) into \( \chi_k \), replaces \( \exists w \in \{0,1\}^W \cdot \text{atmost}_k(v; w) \) by the constraint \( |v| \leq k \), and swaps \( \psi(x/x)(u; v) \) by \( \psi((u \uplus 1_x); v) \). The next equivalence (D.2) uses the fact that for all \( v \) over \( V \),

\[
\psi(u; v) = \phi(u \uplus (\sigma v; x/0)).
\]

We show this fact as follows. Note that for a single element \( v_y \in V \) and \( c, e \in \{0,1\} \),

\[
\phi[y/(y \uplus v_y)]_{y/c,v_y/e} = (y \uplus v_y)_{y/c,v_y/e} = (y_{y/c} \lor \neg y_{y/c})_{y/c,e} = \phi((y_{y/c}) / (c \lor e)).
\]

Repeatedly applying this argument for all \( v_y \in V \) gives us for all \( w \) over \( V \) and \( \Phi \setminus \{x\} \),

\[
\psi_{w; v} = \phi_{w \uplus \sigma v}
\]

and therefore, for \( u = (w; x/c) \) over \( \Phi \) with \( W = \Phi \setminus \{x\} \),

\[
\psi(u; v) = \psi_{w; v}(x/c) = \phi_{w \uplus \sigma v}(x/c) = \phi((w \uplus (\sigma v); x/c) = \phi(u \uplus (\sigma v; x/0)).
\]

For (D.3), we use the fact that \( \sigma: v \mapsto \sigma v \) is a bijection from the set of assignments over \( V \) to the set of assignments over \( \Phi \setminus \{x\} \). For (D.4), we use again a bijection, but this time taking subsets of \( \Phi \setminus \{x\} \) to assignments over \( \Phi \setminus \{x\} \) through \( S \mapsto 1_s \). Finally, for (D.5), we simply apply the definition of \( \text{scs} \).
D.2 Cooperative Game Mappings

For the scope of this section, we interpret the simple cooperative games produced by $\omega$ and $\nu$ as Boolean functions, e.g., write $\omega_f(1_S) = \omega_f(S)$ or equivalently $\omega_f(u) = \omega_f(\{x \in X : u(x) = 1\})$ for all subsets $S \subseteq X$ resp. assignments $u$. Recall from Lemma 8 that for all variables $z$,

$$
\omega_f(S \cup \{z\}) = (\omega_{f_{z/1}} \lor \omega_{f_{z/0}})(S),
$$

$$
\omega_f(S \setminus \{z\}) = \omega_{f_{z/1} \land f_{z/0}}(S),
$$

which means for $\omega_f$’s cofactors w.r.t. $z$,

$$(\omega_f)_{z/1} = \omega_{f_{z/1}} \lor \omega_{f_{z/0}},$$

$$(\omega_f)_{z/0} = \omega_{f_{z/1} \land f_{z/0}}.$$

Using the Shannon decomposition, note that

$$
\omega_f = z(\omega_f)_{z/1} \lor \overline{z}(\omega_f)_{z/0} = z(\omega_{f_{z/1}} \lor \omega_{f_{z/0}}) \lor \overline{z}\omega_{f_{z/0} \land f_{z/1}},
$$

which corresponds to the decomposition as presented in the main part. The Banzhaf value for $\omega_f$ is simply $B\nu_{\omega}(\omega_f) = E(\nu_{\omega_{f_{z/1}}} - E(\{\omega_f\}_{z/0})$, which can easily be computed if $\omega_f$ is represented by a BDD. For the rectifying CGM, note that

$$
\nu_f(S \cup \{z\}) = \nu_{f_{z/1}} \lor \nu_{f_{z/0}}(S),
$$

$$
\nu_f(S \setminus \{z\}) = (\nu_{f_{z/1}} \lor \nu_{f_{z/0}})(S),
$$

which produces the decomposition

$$
\nu_f = z\nu_{f_{z/1}} \lor \nu_{f_{z/0}} \lor \overline{z}\nu_{f_{z/1}} \lor \nu_{f_{z/0}}.
$$

The remainder is analogous.

D.3 Experiments and Results

<table>
<thead>
<tr>
<th>Instance</th>
<th>#Variables</th>
<th>#Clauses</th>
<th>Influence (CNF)</th>
<th>Influence (formula)</th>
<th>Blame</th>
<th>Construction</th>
<th>Influence</th>
<th>DCGM</th>
<th>Blame</th>
</tr>
</thead>
<tbody>
<tr>
<td>b02</td>
<td>26</td>
<td>66</td>
<td>5 ms</td>
<td>49 ms</td>
<td>timeout</td>
<td>1 ms</td>
<td>&lt;1 ms</td>
<td>2 ms</td>
<td>3'649 ms</td>
</tr>
<tr>
<td>b06</td>
<td>44</td>
<td>122</td>
<td>7 ms</td>
<td>99 ms</td>
<td>timeout</td>
<td>3 ms</td>
<td>&lt;1 ms</td>
<td>6 ms</td>
<td>697'573 ms</td>
</tr>
<tr>
<td>b01</td>
<td>45</td>
<td>120</td>
<td>7 ms</td>
<td>110 ms</td>
<td>timeout</td>
<td>4 ms</td>
<td>&lt;1 ms</td>
<td>8 ms</td>
<td>3'086'667 ms</td>
</tr>
<tr>
<td>b03</td>
<td>156</td>
<td>376</td>
<td>11 ms</td>
<td>442 ms</td>
<td>timeout</td>
<td>53'934 ms</td>
<td>24 ms</td>
<td>1'776 ms</td>
<td>timeout</td>
</tr>
<tr>
<td>b09</td>
<td>169</td>
<td>417</td>
<td>22 ms</td>
<td>483 ms</td>
<td>timeout</td>
<td>23'373 ms</td>
<td>13 ms</td>
<td>1'157 ms</td>
<td>timeout</td>
</tr>
<tr>
<td>b08</td>
<td>180</td>
<td>455</td>
<td>26 ms</td>
<td>577 ms</td>
<td>timeout</td>
<td>11'703 ms</td>
<td>18 ms</td>
<td>1'729 ms</td>
<td>timeout</td>
</tr>
<tr>
<td>b10</td>
<td>201</td>
<td>525</td>
<td>27 ms</td>
<td>658 ms</td>
<td>timeout</td>
<td>19'079 ms</td>
<td>44 ms</td>
<td>3'221 ms</td>
<td>timeout</td>
</tr>
<tr>
<td>b13</td>
<td>352</td>
<td>847</td>
<td>34 ms</td>
<td>1'086 ms</td>
<td>timeout</td>
<td>timeout</td>
<td>timeout</td>
<td>timeout</td>
<td>timeout</td>
</tr>
<tr>
<td>b07</td>
<td>435</td>
<td>1'132</td>
<td>64 ms</td>
<td>2'058 ms</td>
<td>timeout</td>
<td>timeout</td>
<td>timeout</td>
<td>timeout</td>
<td>timeout</td>
</tr>
<tr>
<td>b04</td>
<td>730</td>
<td>1'919</td>
<td>324 ms</td>
<td>4'147 ms</td>
<td>timeout</td>
<td>timeout</td>
<td>timeout</td>
<td>timeout</td>
<td>timeout</td>
</tr>
<tr>
<td>b11</td>
<td>765</td>
<td>2'104</td>
<td>606 ms</td>
<td>5'872 ms</td>
<td>timeout</td>
<td>timeout</td>
<td>timeout</td>
<td>timeout</td>
<td>timeout</td>
</tr>
<tr>
<td>b05</td>
<td>966</td>
<td>2'798</td>
<td>173 ms</td>
<td>9'150 ms</td>
<td>timeout</td>
<td>timeout</td>
<td>timeout</td>
<td>timeout</td>
<td>timeout</td>
</tr>
<tr>
<td>b12</td>
<td>1'072</td>
<td>2'911</td>
<td>230 ms</td>
<td>8'555 ms</td>
<td>timeout</td>
<td>timeout</td>
<td>timeout</td>
<td>timeout</td>
<td>timeout</td>
</tr>
</tbody>
</table>

Table 3: Computation time for instances of the ISCAS’99 dataset, timeout set to one hour. BDD columns Influence, DCGM (construction of the BDD for the dominating CGM), and Blame are without the BDD construction time for the initial CNF (cf. column Construction).

The proposed algorithms were tested on two datasets of randomly generated CNFs and the ISCAS’99 benchmark. The former two are specified by tuples $(n, m, k)$ where $n$ is the number of variables, $m$ the number of clauses and $k$ the width of every clause. The first dataset contains random $(n, 3n, 7)$-CNFs which have expected values from at least 0.3 up to 0.9. These CNFs are very balanced; the probability that a random assignment satisfies a random CNF from this dataset is 0.54. The second dataset consists of random $(n, 7n, 4)$-CNFs which have expected values from $7 \cdot 10^{-14}$ up to 0.0195. Only small amounts of assignments satisfy CNFs of this dataset. For each $n$, we have sampled 10 CNFs and computed the mean running time of every algorithm. This includes parsing and constructing the BDDs with fixed but essentially random variable orders. The (modified) blame is computed for $\rho = \rho_{\text{max}}$. Blame and influence values are computed with respect to the first variable.

Let us compare the results of both datasets as shown in Figure 2:
(a) Performance trends for influence. *GPMC satcount (CNF)* is the time spent counting the number of satisfying assignments of the initial CNF and *GPMC influence (CNF)* is the time of the method from [Tra09].

(b) Performance trends for blame.

(c) Performance trends for the dominating CGM.

Figure 2: Performance results for random \((n, 3n, 7)\)-CNFs (left column) and random \((n, 7n, 4)\)-CNFs (right column). *BDD construction* is always the time used to compute the BDD representing the initial CNF and is included in all BDD-based methods.

- **Figure 2a**: For formula based approaches, computing the influence includes parsing the CNF as a formula \(\phi\), applying the transformation \(\phi \mapsto \phi _{x/0} \oplus \phi _{x/1}\) and then re-translating the result back into a CNF via a Tseytin transformation. This increases the original size only linearly, which is arguably the reason for why GPMC’s performance is close (or becomes close over time) to the approach of [Tra09] for both datasets. Our BDD-based implementation is faster for smaller numbers of variables, but is overtaken by GPMC eventually. As one can see on the right side, GPMC is much faster for random CNFs with low expected values. Note the influence is computed with little to no overhead once the BDD construction is done.

- **Figure 2b**: Recall that we have presented two possible ways of computing the (modified) blame. In our case, the projected model counting approach via GPMC performs worse than the BDD-based approach. Note that the modified blame is computed slightly faster than the blame in both cases. However, the computation is still limited to small amounts of variables.

- **Figure 2c**: Our proposed approach for the representation of the dominating cooperative game mapping through BDDs performs best in the second dataset. This is because most of these CNFs have a very low share of satisfied assignments, so one expects the BDDs to have small sizes, which in turn allows one to compute the dominating cooperative game mapping efficiently. (Only few subsets \(S \subseteq X\) satisfy \(\omega_f(S) = 1\).) Thus, the bottleneck for sparsely satisfied CNFs seems to be the BDD’s construction time.

We conclude that (i) the influence can be computed efficiently using modern model counting software, (ii) the blame can be computed efficiently for smaller amounts of variables using a BDD-based approach and (iii) given a BDD representation of a Boolean function, the dominating cooperative game mapping can usually be computed efficiently. For both influence and dominating cooperative game mapping, the BDD construction time is the biggest bottleneck.
Finally, let us consider the ISCAS’99 benchmark [Com23]. Table 3 shows how well our proposed approaches perform on average for 20 runs. The smallest instances of this benchmark still permit the computation of blame values, which was done for the instances b02, b06, and b01. When plotting the number of nodes in the BDDs representing the functions \( \{ \gamma_k \}_{k=0}^{n} \) (recall that for a fixed Boolean function \( f \) and variable \( x \), \( \gamma_k(u) = 1 \) if and only if \( scs^x_u(f) \leq k \)), we obtain Figure 3. Observe that as \( k \) grows, \( \gamma_k \) becomes more and more satisfiable. Consequently, we can see that the sizes of BDDs representing \( \gamma_k \) first increase but later on decrease until they hit zero nodes (i.e. are one everywhere).

References for the Appendix


