Homology of Boolean functions and the complexity of simplicial homology

Erick J. Chastain[1] and Nicholas A. Scoville[2]

1 **Computer Science Department**
Rutgers University
erickc@cs.rutgers.edu

2 **Department of Mathematics and Computer Science**
Ursinus College
nscoville@ursinus.edu

Abstract

We study the topology of Boolean functions from the perspective of Simplicial Homology, and characterize Simplicial Homology in turn by using Monotone Boolean functions. In so doing, we analyze to what extent topological invariants of Boolean functions (for instance the Euler characteristic) change under binary operations over Boolean functions and other operations, such as permutations over the input variables. We apply these tools to proving the Δ_2^p-hardness of calculating the Euler characteristic of general Boolean functions (as defined by Kulkarni and Santha[17]) and the co-NP-hardness of calculating the Euler characteristic for a simplicial complex of arbitrary dimension. We also show that calculating the Betti numbers for Simplicial Homology is co-NP hard.

1998 ACM Subject Classification F.1.3 Complexity Measures and Classes, F.2.2 Nonnumerical Algorithms and Problems, G.2.m Discrete Mathematics (Miscellaneous)

Keywords and phrases Simplicial Homology, Computational Topology, Analysis of Boolean functions, Boolean function isomorphism, Computational Complexity

Digital Object Identifier 10.4230/LIPIcs.xxx.yyy.p

Computational biology in recent years has tried to understand how the following quote attributed to Aristotle is true for biological networks: "The whole is greater than the sum of its parts" [3]. In particular the focus has been on (typically Boolean) functions or properties of objects that are irreducible in some strong sense to the inputs of the functions or parts of the objects [25, 12, 13, 9, 4]. The applications of this are fairly broad, from analysis of gene expression [2], to even proposing theories of what the neural correlates for consciousness are [25]. We make a new contribution to this field by analyzing the qualitative properties of Boolean functions by using tools from Topology, specifically Simplicial Homology.

Surprisingly, we find that there exists a bijection between simplicial complexes and Monotone Boolean functions (building on work by Saks et al. [15]), and that Equivalence classes based on topological invariants for Simplicial complexes give us Isomorphism classes for Monotone Boolean functions. In addition, we find that common binary operations used in building up Boolean formulas ($\wedge$ and $\vee$) correspond to operations on simplicial complexes ($\cap$ and $\cup$). These connections and others allow us to use tools in the analysis of Boolean functions to characterize simplicial complexes and vice-versa.

The study of Boolean functions from a Topological perspective gives us a few new results/tools for computational topology and Boolean functions as well, including

- A new representation (called the union-intersection representation) for simplicial complexes using unions and intersections (Definition 1.1.11)

Conference title on which this volume is based on.
Editors: Billy Editor and Bill Editors; pp. 1–14

Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

- The co-NP-hardness of calculating Betti numbers of a simplicial complex in the union-intersection representation (Corollary 2.0.6)
- The Δ_2^p-hardness of calculating the Euler characteristic of a Boolean function as defined in [17] (Theorem 2.0.3)
- The co-NP-hardness of calculating Betti numbers of a simplicial complex of arbitrary dimension in the union-intersection representation (Corollary 2.0.5)
- An algorithm for Boolean function isomorphism based on calculating the Euler characteristic of a simplicial complex of arbitrary dimension (Theorem 2.0.4)

Finally, for analyzing the "irreducibility" of Boolean functions to their parts, we study the Euler characteristic [17] of a Boolean function as a measure of how "rugged" f is, and show it has some interesting properties. For instance, for Monotone Boolean functions f, g:

- $\chi(f \wedge g) = \chi(f) + \chi(g) - \chi(f \vee g)$ (Proposition 1.1.18)
- $\chi(f \vee g) = \chi(f) + \chi(g) - \chi(f \wedge g)$ (Proposition 1.1.17)
- There exists a non-trivial Boolean function g such that for all f, $\chi(f * g) = 1$ where $f * g$ is the point-wise product of f and g (Example 1.1.8)
- We introduce (in Definition 1.1.9) a natural binary operator for Boolean functions $f \times g$ which satisfies $\chi(f \times g) = \chi(f) \cdot \chi(g)$ (Proposition 1.1.15)

And thus for all of these different instances, the Euler characteristic of the combined Boolean functions is different from the sum of the Euler characteristic of the parts (except when $f \wedge g$ or $f \vee g$ are non-evasive). We also prove a number of novel results about Decision Tree complexity (Proposition 1.1.21). To prove these results, we give a Topological foundation for the choice made by Kulkarni & Santha [17] for the Euler characteristic of a Boolean function (Proposition 1.0.8).

1 The Homology of Boolean functions

In this section we introduce a relation between topological spaces of a certain kind and Boolean functions. We both show that there exist topological invariants that characterize a class of Boolean functions up to isomorphism, and also that the same topological invariant behaves nicely with respect to different binary operators on Boolean functions. For the remaining binary operators, the same topological invariant is defined, and shows non-trivial behavior with respect to the Boolean functions combined using the operators. Moreover, we show that a complexity measure introduced by Kulkarni and Santha [17] is related to this topological invariant, and prove various results about their measure accordingly. Finally, we introduce a new representation for simplicial complexes, and show how this representation is related to Monotone boolean formulas.

When considering decision tree complexity and measures of how "sparse" a Boolean function is, Kulkarni and Santha [17] proposed a new measure of complexity for Boolean functions, the Euler Characteristic:

▶ **Definition 1.0.1.** [17] The Euler characteristic for a Boolean function f is $\chi(f) = \sum_{x \in \{0,1\}^n} (-1)^{wt(x)} f(x)$ where $wt(x)$ is the number of ones in x.

The definition of $\chi(f)$ was given without much motivation, but below we show that a very natural topological space can be made for a Boolean function to justify calling the measure the Euler characteristic (in Proposition 1.0.8). In order to characterize $\chi(f)$, we must first discuss how a special class of Boolean functions has a natural associated topological space. The class of Boolean functions we consider is called the Monotone Boolean functions:

▶ **Definition 1.0.2.** A Monotone Boolean function f is a Boolean function for which $f(x) \leq f(x')$ when $x \leq x'$ with respect to the lexicographical ordering of bit strings.

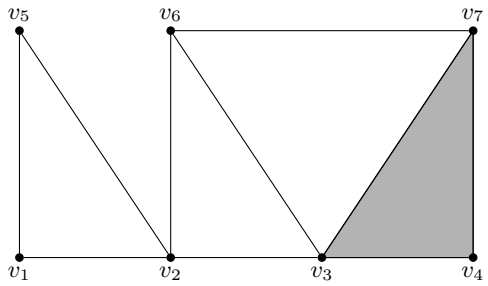
To each Monotone Boolean function, we may associate a unique simplicial complex (See Proposition 1.0.9). Simplicial complexes are a combinatorial object of study in topology, and are given by a collection of k-dimensional building blocks or simplices. They are used to approximate "smooth" topological spaces and play a key role in the classical Invariance of domain problem [20]. One of the most fundamental ways to distinguish these simplicial complexes is by computing their Euler characteristic, an invariant which is defined as the alternating sum of the number of the k-dimensional simplices (Definition 1.0.5). The Euler characteristic relates to the number of k-dimensional "holes" in a simplicial complex by the well-known Euler–Poincaré formula and completely determines the genus of a surface. It played a crucial role in the classification of compact surfaces [19, Chapter 1.8]. Moreover, as it is a topological invariant, $\chi(K) = \chi(S)$ if K and S are isomorphic as simplicial complexes. Results about Euler characteristic may be found in [23, Chapter 1.6]. For more details and the basics of simplicial complexes, see [10].

▶ **Definition 1.0.3.** Let $[n] := \{1, 2, \ldots, n\}$ and $\mathcal{P}([n])$ denote the power set of $[n]$. A **simplicial complex** on $[n]$ is a collection K of subsets of $[n]$ such that if $\sigma \in K$ and $\tau \subseteq \sigma$, then $\tau \in K$. If $\sigma = \{v_1, v_2, \ldots, v_k\} \in K$, we sometimes write $v_1 v_2 \ldots v_k$ as shorthand for σ. We write $V(K) := [n]$ for the **vertex set** of K. An element of K is a **simplex** (plural: **simplices**). If $\sigma \in K$ is any simplex, we call $\dim(\sigma) := |\sigma|$ the **dimension** of σ, and σ is referred to as an **i-simplex**. The largest dimension over all simplices on K is the **dimension** of K. A simplex $\sigma \in K$ which is not contained in any other simplex is called a **facet**. The **n-simplex** is defined by $\Delta^n := \mathcal{P}([n+1])$.

▶ Example 1.0.4. We define a simplicial complex on $\{v_1, v_2, \ldots, v_7\}$ by

$$K = \{\emptyset, v_1, v_2, v_3, v_4, v_5, v_6, v_7, v_1v_2, v_2v_3, v_3v_4, v_1v_5, v_2v_5, v_2v_6, v_3v_6, v_3v_7, v_4v_7, v_6v_7, v_3v_4v_7\}.$$

We may view K geometrically below:



▶ **Definition 1.0.5.** Let K be a simplicial complex of dimension n, and let c_i denote the number of sets in K if cardinality i. The **Euler characteristic** of K is defined by $\chi(K) := \sum_{i=0}^{n} (-1)^i c_i$.

Homology is a technical tool used to count the number and kind of holes in a simplicial complex K on $[n]$. For example, a 0-dimensional hole is the number of connected components of a simplicial complex while an i-dimensional hole is somehow "enclosing" $i+1$-dimensional space. In order to define homology, we begin by constructing a sequence of vector spaces $\{k^{c_i}\}_{i=0}^{\infty}$ with coefficients in $\mathbb{Z}_2$ along with linear transformations $\partial_{i+1} \colon k^{c_{i+1}} \to k^{c_i}$. Recall that given any set X, we can construct the vector space $k^{|X|}$ with basis X and coefficients

in $\mathbb{Z}_2$ by taking all linear combinations of elements in X with coefficients in $\mathbb{Z}_2$. Denote by $C_i(K)$ the set of i-simplices of K and define $c_i := |C_i(K)|$. In order to differentiate between an element of $C_i(K)$ vs the corresponding element in the vector space k^{c_i} generated by $C_i(K)$, we associate to each $\sigma \in C_i(K)$ the symbol e_σ. Hence e_σ is a basis element in the $\mathbb{Z}_2$-vector space k^{c_i} generated by all the elements of $C_i(K)$.

For any $\sigma \in C_i(K)$, the **boundary** of e_σ is given by $\partial_i(e_\sigma) := \sum_{j\in\sigma} e_{\sigma-\{j\}}$. This extends by linearity to a linear transformation on all of k^{c_i}, denoted $\partial_i \colon k^{c_i} \to k^{c_{i-1}}$ called the **boundary operator**. Recall that for any linear transformation $\partial \colon V \to W$, the **image** of ∂ defined by $\mathrm{im}(\partial) := \{w \in W : \partial(v) = w \text{ for some } v \in V\}$ is a vector subspace of W and the **kernal** of ∂ defined by $\ker(\partial) := \{v \in V : \partial(v) = 0\}$ is a vector subspace of V. The dimension of $\mathrm{im}(\partial)$ and $\ker(\partial)$ is the **rank**, $\mathrm{rank}(\partial)$ and the **nullity**, $\mathrm{nul}(\partial)$, respectively. It can be shown that $\partial_{i+1} \circ \partial_i = 0$, whence $\mathrm{im}(\partial_{i+1}) \subseteq \ker(\partial_i)$. We define the i^{th} **(unreduced) homology** of K to be the vector space

$$H_i(K) := k^{\mathrm{nul}\partial_i - \mathrm{rank}\partial_{i+1}}.$$

The i^{th} **Betti number** of K is defined to be $b_i(K) := \mathrm{nul}\partial_i - \mathrm{rank}\partial_{i+1}$.

▶ Example 1.0.6. To illustrate a homology computation, let K be the simplicial complex defined in Example 1.0.4. We expect $b_0(K) = 1$ since K is connected and $b_1(K) = 3$ since we count 3 holes. All the higher Betti numbers we expect to be 0.

We have

$$\begin{aligned} C_0 &= \{v_1, v_2, v_3, v_4, v_5, v_6, v_7\} \\ C_1 &= \{v_1v_2, v_2v_3, v_3v_4, v_1v_5, v_2v_5, v_2v_6, v_3v_6, v_3v_7, v_4v_7, v_6v_7\} \\ C_2 &= \{v_3v_4v_7\} \\ \ldots = C_4 = C_3 &= \emptyset \end{aligned}$$

and $c_0 = 7, c_1 = 10, c_2 = 1$. These in turn generate vector spaces k^7, k^{10}, and k^1.

Since $\emptyset = C_3 = C_4 = \ldots$, we have $0 = k^{c_3} = k^{c_4} = \ldots$ whence $\partial_i = 0$ for $i = 3, 4, \ldots$. We then need to only compute ∂_2 and ∂_1. Now $\partial_2 \colon k^1 \to k^{10}$ and by the definition given above, $\partial_2(e_{v_3v_4v_7}) = \sum_{j\in e_{v_3v_4v_7}} e_{v_3v_4v_7} - \{j\} = e_{v_4v_7} + e_{v_3v_7} + e_{v_3v_4}$. The matrix corresponding to this is then

$$\partial_2 = \begin{array}{c} \\ e_{v_1v_2} \\ e_{v_2v_3} \\ e_{v_3v_4} \\ e_{v_1v_5} \\ e_{v_2v_5} \\ e_{v_2v_6} \\ e_{v_3v_6} \\ e_{v_3v_7} \\ e_{v_4v_7} \\ e_{v_6v_7} \end{array} \begin{array}{c} \begin{array}{c} e_{v_3v_4v_7} \end{array} \\ \begin{pmatrix} 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \\ 0 \end{pmatrix} \end{array}$$

Next we compute $\partial_1 \colon k^{10} \to k^7$:

$$\begin{aligned} \partial_1(e_{v_1v_2}) &= e_{v_2} + e_{v_1} \\ \partial_1(e_{v_2v_3}) &= e_{v_3} + e_{v_2} \end{aligned}$$

$$\begin{aligned}
\partial_1(e_{v_3v_4}) &= e_{v_4} + e_{v_3} \\
\partial_1(e_{v_1v_5}) &= e_{v_5} + e_{v_1} \\
\partial_1(e_{v_2v_5}) &= e_{v_5} + e_{v_2} \\
\partial_1(e_{v_2v_6}) &= e_{v_6} + e_{v_2} \\
\partial_1(e_{v_3v_6}) &= e_{v_6} + e_{v_3} \\
\partial_1(e_{v_3v_7}) &= e_{v_7} + e_{v_3} \\
\partial_1(e_{v_4v_7}) &= e_{v_7} + e_{v_4} \\
\partial_1(e_{v_6v_7}) &= e_{v_7} + e_{v_6}
\end{aligned}$$

This yields the matrix

$$\partial_1 = \begin{array}{c} \\ e_{v_1} \\ e_{v_2} \\ e_{v_3} \\ e_{v_4} \\ e_{v_5} \\ e_{v_6} \\ e_{v_7} \end{array} \begin{array}{c} \begin{array}{cccccccccc} e_{v_1v_2} & e_{v_2v_3} & e_{v_3v_4} & e_{v_1v_5} & e_{v_2v_5} & e_{v_2v_6} & e_{v_3v_6} & e_{v_3v_7} & e_{v_4v_7} & e_{v_6v_7} \end{array} \\ \begin{pmatrix} 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 \end{pmatrix} \end{array}$$

We see that $\text{rank}(\partial_2) = 1, \text{nul}(\partial_2) = 0, \text{rank}(\partial_1) = 6, \text{nul}(\partial_1) = 4, \text{rnk}(\partial_0) = 0$, and $\text{nul}(\partial_0) = 7$. Hence we obtain $H_2(K) = k^0, H_1(K) = k^{4-1} = k^3$, and $H_0(K) = k^{7-6} = k^1$, whence $b_0(K) = 1, b_1(K) = 3$, and $b_i(K) = 0$ for all $i \geq 2$, as predicted above.

The Euler–Poincaré formula relates the Betti numbers b_i to the number of faces c_i:

$$\sum_i (-1)^i b_i = \sum_i (-1)^i c_i = \chi(K) \tag{1}$$

Now we review Saks et al's [15] way of constructing a simplicial complex from a Monotone Boolean function:

▶ **Definition 1.0.7.** Let f be a Monotone Boolean function. The **simplicial complex Γ_f induced by** f is the set of all sets of coordinates such that if $\vec{x} \in \{0,1\}^n$ is 0 exactly on these coordinates, then $f(\vec{x}) = 1$. We adopt the convention that if $x_{i_1} = \ldots = x_{i_k} = 0$ are the 0 coordinates of an input, then the corresponding simplex in Γ_f is denoted by $\sigma_{\vec{x}} := \{x_1, \ldots, x_k\}$.

Using this construction and properties of the Euler characteristic, we give a topological foundation for Kulkarni & Santha's measure $\chi(f)$:

▶ Proposition 1.0.8. For Monotone Boolean functions f, $\chi(f) = \chi(\Gamma_f)$.

Proof. $\chi(f) = \sum_x (-1)^{wt(x)} f(x) = \sum_{x:wt(x)=k} (-1)^{wt(x)} f(x)$. Then it is apparent that $\sum_{x:wt(x)=k} (-1)^{wt(x)} f(x) = \sum_{i=0}^n (-1)^i c_i$. ◀

Now we show that Monotone Boolean functions and simplicial complexes are interchangeable:

▶ Proposition 1.0.9. There is a bijection between Monotone Boolean functions $f\colon \{0,1\}^n \to \{0,1\}$ and simplicial complexes on $[n]$.

Proof. Let $f\colon \{0,1\}^n \to \{0,1\}$ be a Monotone Boolean function. If $f(\vec{x}) = 1$ with 0 coordinates $x_1, \ldots, x_k$, write as above $\sigma_{\vec{x}} := \{x_1, \ldots, x_k\}$ and let Γ_f be the collection of all such $\sigma_{\vec{x}}$. To see that Γ_f is a simplicial complex, let $\sigma_{\vec{x}} \in \Gamma_f$ and suppose $\sigma_{\vec{y}} := \{y_{j_1}, y_{j_2}, \ldots, y_{j_\ell}\}$ is a subset of $\sigma_{\vec{x}}$. We need to show that the value $\vec{y}$ which is 0 exactly on the coordinates of $\sigma_{\vec{y}}$ satisfies $f(\vec{y}) = 1$. Suppose by contradiction that $f(\vec{y}) = 0$. Since $\sigma_{\vec{x}} \in \Gamma_f$, $f(\vec{x}) = 1$. Observe that since $\sigma_{\vec{y}} \subset \sigma_{\vec{x}}$, $\vec{y}$ can be obtained from $\vec{x}$ by switching all the coordinates in $\sigma_{\vec{x}} - \sigma_{\vec{y}}$ from 0 to 1. But then if $f(\vec{y}) = 0$, we have switched our inputs from 0 to 1 while our output has switched from 1 to 0, contradicting the assumption that f is monotonic.

Now let K be a simplicial complex on $[n]$. Let $\sigma \in K$ with $\sigma = \{x_1, \ldots, x_k\}$. Define $\vec{x}_\sigma \in \{0,1\}^n$ to be 0 on coordinates $x_1, \ldots, x_k$ and 1 on all other coordinates. Define $f\colon \{0,1\}^n \to \{0,1\}$ by $f(\vec{x}_\sigma) = 1$ and 0 otherwise. To see that f is monotone, let $f(\vec{x}_\sigma) = 1$ with 0 coordinates $x_1, \ldots, x_k$ and suppose $\vec{x'}$ has 0 coordinates on a subset of S of $\{x_1, \ldots, x_k\}$. Since K is a simplicial complex, $S \in K$. Hence the vector with 0 coordinates S has output 1. But this vector is precisely $\vec{x'}$.

It is clear that these constructions are inverses of each other whence the result.

◀

Now we show that within the class of Monotone Boolean functions, one can characterize exactly isomorphism by looking at isomorphism of simplicial complexes, and vice-versa. This shows that $\chi(f)$ characterizes isomorphism for Monotone Boolean functions since $\chi(K) = \chi(L)$ when K and L are isomorphic simplicial complexes (which is an easy consequence of $\chi(K)$ being a topological invariant).

▶ **Definition 1.0.10.** Let K, L be simplicial complexes. We say that K is **isomorphic** to L, $K \cong L$, if there exists a bijection $\phi\colon V(K) \to V(L)$ such that $v_1 \ldots v_m \in K$ if and only if $\phi(v_1) \ldots \phi(v_m) \in L$.

▶ **Definition 1.0.11.** Two Boolean functions $f, g\colon \{0,1\}^n \to \{0,1\}$ are isomorphic if there exists a permutation σ such that $f(x_1, \ldots, x_n) = g(x_{\sigma(1)}, \ldots, x_{\sigma(n)})$.

▶ Theorem 1.0.12. Let $f, g\colon \{0,1\}^n \to \{0,1\}$ be Monotone Boolean functions. Then $\Gamma_f \cong \Gamma_g$ if and only if f is isomorphic to g.

Proof. Suppose that $\Gamma_f \cong \Gamma_g$. Then there is a bijection $\phi\colon V(\Gamma_f) \to V(\Gamma_g)$ satisfying $\{i_1, \ldots, i_k\} \in \Gamma_f$ if and only if $\{\phi(i_1) \ldots \phi(i_k)\} \in \Gamma_g$. Define a permutation

$$s(i) = \begin{cases} \phi(i) & \text{if } i \in V(\Gamma_f) \\ i & \text{otherwise.} \end{cases}$$

Since ϕ is a bijection, s is a permutation. We claim that $f(x_1, \ldots, x_n) = g(x_{s(1)}, \ldots, x_{s(n)})$. Write $s(\vec{x}) := (x_{s(1)}, \ldots, x_{s(n)})$. Suppose $f(x_1, \ldots, x_n) = 1$ and let $x_{i_1}, \ldots, x_{i_k}$ be the 0 coordinates. Then $\{i_1, \ldots, i_k\} \in \Gamma_f$ and since $\Gamma_f \cong \Gamma_g$, we have $\{\phi(i_1), \ldots, \phi(i_k)\} \in \Gamma_g$. It follows that the vector with exactly $x_{s(i_1)}, \ldots, x_{s(i_k)}$ as 0 coordinates is equal to 1 under g i.e. $g(s(\vec{x})) = 1$. The same argument holds assuming $g(x_1, \ldots, x_n) = 1$, hence $f(\vec{x}) = g(s(\vec{x}))$.

Now suppose that there exists a permutation s such that $f(x_1, \ldots, x_n) = g(x_{s(i_1)}, \ldots, x_{s(i_n)})$. Define a bijection $\phi\colon V(\Gamma_f) \to V(\Gamma_g)$ by $\phi(i) = s(i)$. We have

$$\begin{aligned} \sigma := \{i_1, \ldots, i_k\} \in \Gamma_f &\Leftrightarrow f(\vec{x}_\sigma) = 1 \\ &\Leftrightarrow g(s(\vec{x}_\sigma)) = 1 \end{aligned}$$

$$\Leftrightarrow \quad \{\phi(i_1)\dots\phi(i_k)\} \in \Gamma_g.$$

Thus $\Gamma_f \cong \Gamma_g$. ◀

1.1 Binary operations on functions and simplicial complexes

Now we turn to natural binary operations on simplicial complexes and how they correspond to natural binary operations on Boolean functions. For our first binary operation, we look at the OR of two Boolean functions, and show that this corresponds to the union of the corresponding simplicial complexes for each function.

▶ **Definition 1.1.1.** Let $f, g\colon \{0,1\}^n \to \{0,1\}$ be Boolean functions. Define $f \vee g\colon \{0,1\}^n \to \{0,1\}$ by

$$f \vee g(\vec{x}) = \begin{cases} 1 & \text{if } f(\vec{x}) = 1 \text{ or } g(\vec{x}) = 1 \\ 0 & \text{otherwise.} \end{cases}$$

▶ Proposition 1.1.2. Let $f, g\colon \{0,1\}^n \to \{0,1\}$ be Boolean functions. Then $\Gamma_{f\vee g} = \Gamma_f \cup \Gamma_g$

Proof. Let $\sigma \in \Gamma_{f\vee g}$. Then $\sigma = \{i_1, \dots, i_k\}$ corresponds to $\vec{x}_\sigma$ satisfying $f \vee g(\vec{x}_\sigma) = 1$. Hence either $f(\vec{x}_\sigma) = 1$ or $g(\vec{x}_\sigma) = 1$. Without loss of generality, suppose $f(\vec{x}_\sigma) = 1$. It follows that $\sigma \in \Gamma_f$ whence $\sigma \in \Gamma_f \cup \Gamma_g$.

Now suppose that $\sigma \in \Gamma_f \cup \Gamma_g$. Without loss of generality, assume $\sigma \in \Gamma_f$. Then $f(\vec{x}_\sigma) = 1$ so clearly $f \vee g(\vec{x}_\sigma) = 1$ whence $\sigma \in \Gamma_{f\vee g}$, and the proof is complete.

◀

For our next binary operation, we look at the AND of two Boolean functions, and show that this corresponds to the intersection of the corresponding simplicial complexes for each function.

▶ **Definition 1.1.3.** Let $f, g\colon \{0,1\}^n \to \{0,1\}$ be Boolean functions. Define $f \wedge g\colon \{0,1\}^n \to \{0,1\}$ by

$$f \wedge g(\vec{x}) = \begin{cases} 1 & \text{if } f(\vec{x}) = 1 \text{ and } g(\vec{x}) = 1 \\ 0 & \text{otherwise.} \end{cases}$$

▶ Proposition 1.1.4. Let $f, g\colon \{0,1\}^n \to \{0,1\}$ be Boolean functions. Then $\Gamma_{f\wedge g} = \Gamma_f \cap \Gamma_g$

Proof. Let $\sigma \in \Gamma_{f\wedge g}$. Then $\sigma = \{i_1, \dots, i_k\}$ corresponds to $\vec{x}_\sigma$ satisfying $f \wedge g(\vec{x}_\sigma) = 1$. Hence $f(\vec{x}_\sigma) = 1$ and $g(\vec{x}_\sigma) = 1$. Without loss of generality, suppose $f(\vec{x}_\sigma) = 1$ and $g(\vec{x}_\sigma) = 1$. It follows that $\sigma \in \Gamma_f$ and $\sigma \in \Gamma_g$ whence $\sigma \in \Gamma_f \cap \Gamma_g$.

Now suppose that $\sigma \in \Gamma_f \cap \Gamma_g$. Without loss of generality, assume $\sigma \in \Gamma_f$ and $\sigma \in \Gamma_g$. Then $f(\vec{x}_\sigma) = 1$ and $g(\vec{x}_\sigma) = 1$. So clearly $f \wedge g(\vec{x}_\sigma) = 1$ whence $\sigma \in \Gamma_{f\wedge g}$, and the proof is complete.

◀

If we take the point-wise product of two Boolean functions (also called the tensor product), we show that this corresponds to taking the join of their simplicial complexes:

▶ **Definition 1.1.5.** Let $f\colon \{0,1\}^n \to \{0,1\}$ and $g\colon \{0,1\}^m \to \{0,1\}$ be Boolean. Define the **join** $f * g\colon \{0,1\}^{n+m} \to \{0,1\}$ of f and g by

$$f * g(v_1, \dots, v_n, v_{n+1}, \dots, v_{n+m}) = f(v_1, \dots, v_n) \cdot g(v_{n+1}, \dots, v_{n+m})$$

the point-wise product of f and g.

▶ **Definition 1.1.6.** Let K, L be simplicial complexes. The **join** of K and L, denotes $K * L$, is defined by $K * L := \{\sigma \sqcup \tau : \sigma \in K, \tau \in L\}$ where $\sqcup$ is the disjoint union.

▶ Proposition 1.1.7. Let $f\colon \{0,1\}^n \to \{0,1\}$ and $g\colon \{0,1\}^m \to \{0,1\}$ be Monotone Boolean functions. Then $\Gamma_f * \Gamma_g = \Gamma_{f*g}$.

Proof. Let $\gamma \in \Gamma_f * \Gamma_g$. Then $\sigma = \sigma \sqcup \tau$ where $\sigma = \{i_1, \ldots, i_k\} \in \Gamma_f$ and $\tau = \{j_1, \ldots, j_\ell\} \in \Gamma_g$. It follows that $f(\vec{x}_\sigma) = 1$ and $g(\vec{y}_\tau) = 1$ so that $f * g(\vec{x}_\sigma, \vec{y}_\tau) = 1$, whence $\sigma \sqcup \tau = \{i_1, \ldots, i_k, j_1, \ldots j_\ell\} \in \Gamma_{f*g}$. These steps reversed shows the other direction so that $\Gamma_f * \Gamma_g = \Gamma_{f*g}$. ◀

Here is an example of how radically different the topology of a Monotone Boolean function can change if we take the point-wise product of it with another function:

▶ Example 1.1.8. In the special case where $L = \{v\}$ is a single vertex, $CK := K * L$ is called the **cone** on K. It is well known that the cone is collapsible and hence $\chi(CK) = 1$ for all complexes K. The "cone" over any Monotone Boolean function $f\colon \{0,1\}^n \to \{0,1\}$ can be realized by Γ_{f*g} where $g\colon \{0,1\}^1 \to \{0,1\}$ defined by $g(0) = g(1) = 1$.

So then there always exists a g such that for any Monotone Boolean function $\chi(f * g) = 1$ (since if K is collapsible $\chi(K) = 1$). Now we can define a binary operator over Boolean functions that acts very much like the AND of two Boolean functions, but which only outputs 1 if the satisfying assignments of both f and g being considered are lexicographically increasing. We then relate this operator over Boolean functions to the product of their respective simplicial complexes.

▶ **Definition 1.1.9.** Let $f\colon \{0,1\}^n \to \{0,1\}$ and $g\colon \{0,1\}^m \to \{0,1\}$ be Boolean. Define $f \times g\colon \{0,1\}^{n \cdot m} \to \{0,1\}$ as follows: For any input

$$\vec{v} := (v_{1,1}, v_{1,2}, \ldots, v_{1,m}, v_{2,1}, \ldots, v_{n,m}) \in \{0,1\}^{n \cdot m}$$

suppose $v_{i_1,j_1} = \ldots = v_{i_k,j_k} = 0$ are precisely the 0 coordinates of $\vec{v}$. Then $f \times g(\vec{v}) = 1$ if and only if the following conditions holds:

1. $i_1 \leq \ldots \leq i_k$
2. $j_1 \leq \ldots \leq j_k$
3. f has output 1 when exactly the coordinates with index $i_1, \ldots, i_k$ are equal to 0
4. g has output 1 when exactly the coordinates with index $j_1, \ldots, j_k$ are equal to 0

▶ Proposition 1.1.10. Let f and g be Monotone Boolean functions. Then $\Gamma_{f \times g} = \Gamma_f \times \Gamma_g$.

Proof. Let $\sigma = \{(x_1, y_1), \ldots, (x_k, y_k)\} \in \Gamma_{f \times g}$. Passing to the Boolean function $f \times g$ and applying Definition 1.1.9, we have that $x_1 \leq \ldots \leq x_k, y_1 \leq \ldots \leq y_k, f(\vec{x}_\sigma) = 1$ and $g(\vec{y}_\sigma) = 1$. Hence $\{x_1, \ldots, x_k\} \in \Gamma_f$ and $\{y_1, \ldots, y_k\} \in \Gamma_g$ so $\sigma \in \Gamma_f \times \Gamma_g$. The same argument yields the reverse direction so that $\Gamma_{f \times g} = \Gamma_f \times \Gamma_g$. ◀

Motivated by the operations above, we introduce a new representation for simplicial complexes. We specify basis subsimplices, and then combine them with intersections and unions. We call this the union-intersection representation:

Before we begin, we would like to store the union or intersection of functions which correspond to x_i when translated to formulas. To do this we make Boolean functions $f(x)$ such that $f(x) = x_i$. These are called dictator fuctions. Then we encode these functions as simplicial complexes. The resulting simplicial complex is $P([n] \setminus i)$. Rather than storing the whole simplicial complex, we use the list of facets that make up the simplicial complex as

featured in [5]. It turns out that $P([n] \setminus i)$ has only one facet, $[n] \setminus i$. So this will stand in for the simplicial complex $P([n] \setminus i)$, which we then take to build up the encoded simplicial complex using the unions and intersections given in the representation.

▶ **Definition 1.1.11.** Let Σ be the alphabet $\{1, ..., n, \cup, \cap, (,), \{, \}\}$. The simplicial complexes in the union-intersection representation (UIR) are given the following inductive definition:

- $[n] \setminus i$ with $i \in [n]$ is in the UIR
- If α and β are in the UIR, then so are $(\alpha \cup \beta)$ and $(\alpha \cap \beta)$.

By Propositions 1.1.3 and 1.1.2, a simplicial complex constructed according to the union-intersection representation is the simplicial complex of a monotone Boolean formula (a boolean formula composed only of $\wedge$ and $\vee$ gates [14]). Just replace the instances of $\cup$ with $\vee$ and $\cap$ with $\wedge$, replace every instance of $[n] \setminus i$ by x_i. Using standard pattern matching algorithms, this can be done in time linear in the size of the representation (each pattern is constant size compared to the size of the representation). This gives us the following Proposition:

▶ Proposition 1.1.12. A simplicial complex Γ constructed by using a union-intersection representation gives exactly a corresponding monotone Boolean formula f_Γ, with $\cap$ and $\cup$ corresponding to $\wedge$ and $\vee$ respectively, $[n] \setminus i$ should be replaced by x_i. Constructing a monotone Boolean formula from a simplicial complex in union-intersection form can be done in time linear in the size of the representation.

Here is an example of how a simplicial complex in the union-intersection representation corresponds to a Boolean formula:

▶ Example 1.1.13. In the shorthand established above ($v_1v_2v_3 = \{v_1, v_2, v_3\}$), $(\{23\} \cap \{13\}) \cup \{12\} \cup (\{23\} \cap \{12\})$ is a simplicial complex in the union-intersection representation. The corresponding Boolean formula is $(x_1 \wedge x_2) \vee x_3 \vee (x_1 \wedge x_3)$.

Based on Propositions 1.1.12 and 1.0.9 we know that a union-intersection representation exists for an arbitrary simplicial complex, since a monotone formula exists for any Monotone Boolean function [14].

Now we consider some simple relations between binary operators on Boolean functions and how they change topologically under those operators. In particular, we study how the Euler characteristic changes under these binary operations.

The Euler characteristic behaves very nicely with respect to the binary operators we used for simplicial complexes. The product of two simplicial complexes has as many "holes" as the product of their respective Euler characteristics:

▶ Proposition 1.1.14. [23, p. 52]For any simplicial complexes K and L, we have $\chi(K \times L) = \chi(K) \cdot \chi(L)$.

and so we can say that for the product of two Boolean functions, the Euler characteristic of the whole also grows in a non-linear way as a function of the parts:

▶ Proposition 1.1.15. $\chi(f \times g) = \chi(f) \cdot \chi(g)$ for Monotone Boolean functions f and g.

Proof. Because f and g are monotone, $\chi(f \times g) = \chi(\Gamma_{f \times g})$. The RHS becomes $\chi(\Gamma_f)\chi(\Gamma_g)$, whence the result. ◀

The union of simplicial complexes has as many "holes" as the sum of the respective Euler characteristics of the parts, minus that of their intersection:

▶ Proposition 1.1.16. [23](Inclusion/Exclusion Formula)For any simplicial complexes K and L, we have $\chi(K \cup L) = \chi(K) + \chi(L) - \chi(K \cap L)$.

So for the OR of two Boolean functions, the Euler characteristic of the whole also grows in a sub-linear way as a function of the parts:

▶ Proposition 1.1.17. $\chi(f \vee g) = \chi(f) + \chi(g) - \chi(f \wedge g)$ for Monotone Boolean functions f and g

Proof. Because f and g are monotone, $\chi(f \vee g) = \chi(\Gamma_{f \vee g}) = \chi(\Gamma_f \cup \Gamma_g)$. By the Inclusion/Exclusion Formula, the RHS becomes $\chi(\Gamma_f) + \chi(\Gamma_g) - \chi(\Gamma_{f \wedge g})$ whence the result. ◀

The same is true for the AND of two functions:

▶ Proposition 1.1.18. $\chi(f \wedge g) = \chi(f) + \chi(g) - \chi(f \vee g)$ for Monotone Boolean functions f and g

Proof. Because f and g are monotone, $\chi(f \wedge g) = \chi(\Gamma_{f \wedge g}) = \chi(\Gamma_f \cap \Gamma_g)$. By the Inclusion/Exclusion Formula, the RHS becomes $\chi(\Gamma_f) + \chi(\Gamma_g) - \chi(\Gamma_{f \vee g})$ whence the result. ◀

The results are least interesting when $f \wedge g$ or $f \vee g$ are non-evasive (have Decision tree complexity less than n [14]), as then $\chi(f \wedge g) = 1$ or $\chi(f \vee g) = 1$ respectively by the collapsibility of the function [15]. Otherwise, if there is a non-trivial interaction between the two functions, then the whole function's Euler characteristic is truly different from the sum of the parts, and it could have less "holes."

A corollary of Proposition 1.1.17 is a direct sum-type theorem for $\chi(f)$ with respect to OR, and also for AND:

▶ Corollary 1.1.19. If f is a Monotone Boolean function then $\chi(\vee_{i=1}^{k} f) = k\chi(f)$ and $\chi(\wedge_{i=1}^{k} f) = k\chi(f)$

Proof. First note that by Proposition 1.1.17 $\chi(f \vee f) = 2\chi(f) - \chi(f \wedge f)$. Now $f \wedge f = f$, so $\chi(f \vee f) = \chi(f)$. An easy induction argument gives the result.

Now $\chi(f \wedge f) = 2\chi(f) - \chi(f \vee f)$ by a similar argument with Proposition 1.1.18, and $f \vee f = f$, so again by a similar argument, $\chi(\wedge_{i=1}^{k} f) = k\chi(f)$. ◀

In three cases of the above binary operations over Monotone Boolean functions, product, AND, and OR, we get that the whole is quantitatively different topologically from the sum of its parts. We saw an example as well when we discussed the join, of how the point-wise product could arbitrarily decrease the Euler characteristic of the whole. So in all of these cases (except when $f \wedge g$ or $f \vee g$ is non-evasive) the sum really is "more" (beyond) the parts.

Now we can use the relations we derived above to derive novel identities for $\chi(f)$. Let $\text{supp}(x) = \{x_i | x_i \neq 0\}$. We recall that a Boolean function has a representation in an orthonormal Fourier basis (with each term S standing for a subset of $\{0,1\}^n$), and that $\hat{f}(S)$ is the coefficient for one term in that expansion [22]. It was proven in [17] that when $\chi(f) \neq 0$,

$$\log |\text{supp}(\hat{f})| = \Omega(n - log|\chi(f)|)$$

With the above identities and the others in this section, we derive immediately the following Proposition:

▶ Proposition 1.1.20. Let f and g be Monotone Boolean functions, then it holds that when $\chi(f \vee g) \neq 0$ or $\chi(f \wedge g) \neq 0$ respectively

$$\log |\text{supp}(\widehat{f \vee g})| = \Omega(n - log|\chi(f) + \chi(g) - \chi(f \wedge g)|)$$

and

$$\log |\text{supp}(\widehat{f \wedge g})| = \Omega(n - log|\chi(f) + \chi(g) - \chi(f \vee g)|)$$

Using this proposition, we can prove the following proposition about the decision tree [14] $D(f \vee g)$ and parity decision tree [26] complexities $D_{\oplus}(f \vee g)$ of $f \vee g$:

▶ Proposition 1.1.21. Let f and g be Monotone Boolean functions, then it holds that when $\chi(f \vee g) \neq 0$ or $\chi(f \wedge g) \neq 0$ respectively

$$D(f \vee g) \geq D_{\oplus}(f \vee g) = \Omega(n - log|\chi(f) + \chi(g) - \chi(f \wedge g)|)$$

and

$$D(f \wedge g) \geq D_{\oplus}(f \wedge g) = \Omega(n - log|\chi(f) + \chi(g) - \chi(f \vee g)|)$$

Proof. By results in [26, 21]

$$D_{\oplus}(f \vee g) = \Omega(\log |\text{supp}(\widehat{f \vee g})|)$$

and by Proposition 1.1.20, therefore

$$D_{\oplus}(f \vee g) = \Omega(n - log|\chi(f) + \chi(g) - \chi(f \wedge g)|)$$

The result follows since $D(f) \geq D_{\oplus}(f)$. A similar argument using the second part of Proposition 1.1.20 gives us the other bound. ◀

2 The Computational Complexity of Simplicial Homology and Topological invariants of Boolean functions

Using the results from the previous section, we can prove some interesting results in the computational complexity of calculating some classic quantities of interest in computational topology and some other natural problems in the analysis of Boolean functions. The computational topology results focus on the computational complexity of Betti numbers and Euler characteristic for arbitrary dimension for simplicial complexes encoded in the union-intersection representation. For these two problems, previous results have shown that the Euler characteristic is $\#P$-complete when representing the complex by its maximal facets in the same setting [24], and that there are reasonably efficient algorithms for fixed dimension[7] or simplicial complexes with few simplices [8, 7].

We begin by defining problems in computational complexity that we will use in our reductions:

▶ **Definition 2.0.1.** An instance of the LexMaxSAT problem gives as input the Boolean formula f and asks to find the lexicographically-maximal $x \in \{0,1\}^n$ such that $f(x) = 1$.

▶ **Definition 2.0.2.** An instance of the Boolean function isomorphism problem gives as input the pair of Boolean formulas f and g and asks whether f and g are isomorphic. The Monotone Boolean function isomorphism problem is the same but with f and g Monotone boolean formulas.

Now we show that calculating the Euler characteristic of a Boolean formula is Δ_2^p-hard when we place no restrictions on the type of formula. We do this by reducing LexMaxSAT to the Euler characteristic, exploiting that the latter will not change for inputs x with $wt(x)$ higher than the weight of the lexicographically-maximal satisfying assignment of the Boolean formula to do a binary search for the solution:

▶ Theorem 2.0.3. Calculating $\chi(f)$ with formula inputs is Δ_2^p-hard.

Proof. We will prove a polynomial-time reduction from LexMaxSAT to calculating $\chi(f)$. This then shows that calculating $\chi(f)$ is Δ_2^p-Hard because LexMaxSAT is Δ_2^p-Complete [16]. Let n_i be the number of inputs to the Boolean formula. Let f_k be a function identical to f on all the inputs of weight at most k, and 0 otherwise. Then by the definition of $\chi(f)$ if there are no satisfying assignments of weight k, $\chi(f_k) = \chi(f_{k-1})$. So to find the lexicographically maximum satisfying assignment, we want to find a $k*$ for which $\chi(f_{k'}) = \chi(f_{k*})$ when $k' \geq k*$. Let us calculate $\chi(f_k)$ for all k, finding a $k*$ for which $\chi(f_{k'}) = \chi(f_{k*})$ when $k' > k*$ in n_i steps. To calculate f_k we just set $f_k(x) = f(x) \wedge g(x)$ where $g(x) = 0$ if $wt(x) > k$ and $g(x) = 1$ otherwise (one can encode $g(x)$ as the negation of a DNF formula, with each clause having a conjunction of k un-negated literals). Now looking through all sets of bitstrings with weight $k*$, we are guaranteed that the lexicographically maximum satisfying assignment in such a set will be lexicographically maximum for all satisfying assignments. We can do this using binary search using the lexicographical ordering, and since there are at most $\binom{n_i}{\frac{n_i}{2}}$ bitstrings for which the weight is equal, the binary search will finish in at most $O(n_i \log n_i)$ steps. So overall the algorithm will take at most $n_i + O(n_i \log n_i) = O(n_i \log n_i)$ steps. Note that since n_i is the number of inputs for the Boolean formula, this implies that the reduction is polynomial time in the input size (for formula inputs) ◀

In contrast, when we restrict calculating the Euler characteristic to Boolean formulas that are monotone, it is only co-NP-hard. We show this by reducing Boolean function isomorphism to calculating $\chi(f)$ and $\chi(g)$ for both Boolean formulas and checking whether they are equal. We also show that if we calculate the Euler characteristic of a simplicial complex Γ in the union-intersection representation, this too is co-NP-Hard, again by the same kind of reduction.

▶ Theorem 2.0.4. Boolean function isomorphism is polynomial-time reducible to calculating $\chi(f)$ from formula inputs for monotone f.

Proof. Using Theorem 1.0.12 and Proposition 1.0.8, for f and g monotone, $\chi(f) = \chi(\Gamma_f)$ and $\chi(g) = \chi(\Gamma_g)$, so $\chi(f) = \chi(g)$ iff f and g are isomorphic. Therefore we can simply check that $\chi(f) = \chi(g)$ and we will find out whether f and g are isomorphic. Finally then, there exists a polynomial time reduction from Boolean function isomorphism to Monotone Boolean function isomorphism [11].So this shows that there exists a reduction from Boolean function isomorphism to calculating $\chi(f)$ from formula inputs. ◀

▶ Corollary 2.0.5. Calculating $\chi(f)$ for monotone f with formula inputs or $\chi(\Gamma)$ with Γ in the union-intersection representation is co-NP-hard.

Proof. Boolean function isomorphism is co-NP-hard [1], so there exists a polynomial-time reduction from a co-NP-complete problem to it. Therefore by Theorem 2.0.4, there exists a polynomial-time reduction from a co-NP-complete problem to calculating the Euler characteristic of monotone f with a formula as input. Now the same holds for calculating the Euler characteristic of a simplicial complex Γ in the union-intersection representation by Proposition 1.1.12, since one can therefore calculate the Euler characteristic of any Monotone formula, and thus one can reduce calculating $\chi(f)$ (with f a formula) to calculating $\chi(\Gamma_f)$ with Γ_f in the union-intersection representation (and by Proposition 1.1.12 the translation from the formula representation to the union-intersection representation takes linear time). ◀

Finally, we show that calculating the Betti numbers for a simplicial complex in the union-intersection representation is co-NP-hard. To do this, we use the Euler-Poincare

formula (Equation 1), and reduce calculating the Euler characteristic of f to calculating all the Betti numbers of the associated simplicial complex Γ_f.

▶ Corollary 2.0.6. Calculating the Betti numbers b_i for all i for a simplicial complex in the union-intersection representation is co-NP-hard.

Proof. This follows from the Euler-Poincare formula (Equation 1), since this shows that by calculating all Betti numbers of a simplicial complex Γ in the union-intersection representation, we can calculate the Euler characteristic of Γ. So there is a reduction from calculating $\chi(\Gamma_f)$ to calculating the Betti numbers of Γ_f, and we can use Proposition 1.0.8 and Proposition 1.1.12 to show that there is a reduction from calculating $\chi(f)$ for monotone f to calculating all Betti numbers of Γ_f for a simplicial complex in the union-intersection representation (use the mapping given in Proposition 1.1.12 to get Γ_f in union-intersection representation from the formula f, and this only requires linear time). Then by Corollary 2.0.5, there exists a reduction from a co-NP-complete problem to calculating all the Betti numbers of Γ_f. ◀

3 Conclusion

The connection between Simplicial Homology and the analysis of Boolean functions has given us interesting new methods for the analysis of Boolean functions and for representing simplicial complexes. Moreover, we were able to derive computational complexity results using this connection. Some natural questions are whether calculating the Euler characteristic for Boolean functions is reducible to calculating the same for Monotone Boolean functions. Such a result would prove that Boolean function isomorphism is Δ_2^p-hard (the best upper-bound known is that it is in Σ_2^p but it isn't Σ_2^p-complete, and it is co-NP-hard [6]). Also, could one use random Monotone Boolean functions to characterize random simplicial complexes in some way? What properties of commonly-used simplicial complexes can be related to the union-intersection representation? For instance, is there a relation between the Euler characteristic of the Boolean formula and formula size?

References

1. Manindra Agrawal and Thomas Thierauf. The boolean isomorphism problem. In *Foundations of Computer Science, 1996. Proceedings., 37th Annual Symposium on*, pages 422–430. IEEE, 1996.
2. Dimitris Anastassiou. Computational analysis of the synergy among multiple interacting genes. *Molecular systems biology*, 3(83):83, 2007.
3. Aristotle. *Metaphysics*, volume VIII:6 (line 1045a9).
4. Nihat Ay, Eckehard Olbrich, Nils Bertschinger, and Jurgen Jost. A geometric approach to complexity. *Chaos (Woodbury, N.Y.)*, 21(3):037103, Sep 2011.
5. Bruno Benedetti and Frank H Lutz. Random discrete morse theory and a new library of triangulations. *Experimental Mathematics*, 23(1):66–94, 2014.
6. Bernd Borchert, Desh Ranjan, and Frank Stephan. On the computational complexity of some classical equivalence relations on boolean functions. *Theory of Computing Systems*, 31(6):679–693, 1998.
7. Martin Cadek, Marek Krcal, Jiri Matousek, Lukas Vokrinek, and Uli Wagner. Polynomial-time computation of homotopy groups and postnikov systems in fixed dimension. *Siam Journal on Computing*, 43(5):1728–1780, 2014.
8. Cecil Jose A Delfinado and Herbert Edelsbrunner. An incremental algorithm for betti numbers of simplicial complexes. In *Proceedings of the ninth annual symposium on Computational geometry*, pages 232–239. ACM, 1993.

9 Lee DeVille and Eugene Lerman. Dynamics on networks of manifolds. *arXiv*, page 15, Aug 2012.

10 D. L. Ferrario and R. A. Piccinini. *Simplicial structures in topology*. CMS Books in Mathematics/Ouvrages de Mathématiques de la SMC. Springer, New York, 2011. Translated from the 2009 Italian original by Maria Nair Piccinini.

11 Judy Goldsmith, Matthias Hagen, and Martin Mundhenk. Complexity of dnf and isomorphism of monotone formulas. In *Mathematical Foundations of Computer Science 2005*, pages 410–421. Springer, 2005.

12 Virgil Griffith and Christof Koch. Quantifying synergistic mutual information. In *Guided Self-Organization: Inception*, pages 159–190. Springer, 2014.

13 Nikhil J Joshi, Giulio Tononi, and Christof Koch. The minimal complexity of adapting agents increases with fitness. *PLoS Comput Biol*, 9(7):e1003111, 2013.

14 Stasys Jukna. *Boolean function complexity: advances and frontiers*, volume 27. Springer Science & Business Media, 2012.

15 J. Kahn, M. Saks, and D. Sturtevant. A topological approach to evasiveness. *Combinatorica*, 4(4):297–306, 1984.

16 Mark W Krentel. The complexity of optimization problems. In *Proceedings of the eighteenth annual ACM symposium on Theory of computing*, pages 69–76. ACM, 1986.

17 Raghav Kulkarni and Miklos Santha. Query complexity of matroids. In *Algorithms and Complexity*, pages 300–311. Springer, 2013.

18 OB Lupanov. On the realization of functions of logical algebra by formulae of finite classes (formulae of limited depth) in the basis $\wedge$,$\vee$,$\neg$. *Probl. Cybern*, 6:1–14, 1965.

19 W. S. Massey. *A basic course in algebraic topology*, volume 127 of *Graduate Texts in Mathematics*. Springer-Verlag, New York, 1991.

20 J. McCleary. *A first course in topology*, volume 31 of *Student Mathematical Library*. American Mathematical Society, Providence, RI, 2006. Continuity and dimension.

21 Ashley Montanaro and Tobias Osborne. On the communication complexity of xor functions. *arXiv preprint arXiv:0909.3392*, 2009.

22 Ryan O'Donnell. *Analysis of boolean functions*. Cambridge University Press, 2014.

23 V. V. Prasolov. *Elements of homology theory*, volume 81 of *Graduate Studies in Mathematics*. American Mathematical Society, Providence, RI, 2007. Translated from the 2005 Russian original by Olga Sipacheva.

24 Bjarke Hammersholt Roune and Eduardo Sáenz-de Cabezón. Complexity and algorithms for euler characteristic of simplicial complexes. *Journal of Symbolic Computation*, 50:170–196, 2013.

25 Giulio Tononi. An information integration theory of consciousness. *BMC neuroscience*, 5(1):42, 2004.

26 Zhiqiang Zhang and Yaoyun Shi. On the parity complexity measures of boolean functions. *Theoretical Computer Science*, 411(26):2612–2618, 2010.