Available online at www.sciencedirect.com

SciVerse ScienceDirect

Electronic Notes in Theoretical Computer Science

Electronic Notes in Theoretical Computer Science 283 (2012) 159–177

www.elsevier.com/locate/entcs

Asynchronous Logic Circuits and Sheaf Obstructions

Michael Robinson[1,2]

Mathematics Department
University of Pennsylvania
209 S. 33rd. Street
Philadelphia, PA 19104

Abstract

This article exhibits a particular encoding of logic circuits into a sheaf formalism. The central result of this article is that there exists strictly more information available to a circuit designer in this setting than exists in static truth tables, but less than exists in event-level simulation. This information is related to the timing behavior of the logic circuits, and thereby provides a "bridge" between static logic analysis and detailed simulation.

Keywords: logic circuits, asynchronous systems, sheaf cohomology, circuit structural analysis

1 Introduction

Verification of asynchronous logic circuits usually requires extensive simulation and appropriate test coverage. This article presents a novel technique for detecting certain behavioral properties of a logic circuit using a less exhaustive structural analysis. In this analysis, wire delays are unknown and finite, but unlike the work of others in this situation, the wire delays are implicit. We do not need to assume that they have a fixed value over time, and we never specify them even as variables. We show how potentially hazardous race conditions (which often cause glitches or unwanted latching) correspond to nontrivial first cohomology classes of a particular sheaf that encodes this implicit timing model of the circuit.

[1] This work was supported under AFOSR FA9550-09-1-0643

[2] Email: robim@math.upenn.edu

doi:10.1016/j.entcs.2012.05.010

1.1 Historical discussion

The synthesis of asynchronous logic circuits is an old subject, having been studied in the earliest days of computing by Huffman [19]. Although the benefits of using asynchronous over synchronous hardware are substantial (better composibility of modules, lower power usage, lower electromagnetic interference, faster speed), its challenges have generally precluded its widespread acceptance. Most of the difficulty of asynchronous design involves careful control of delays within the circuit, and the avoidance of race conditions [21]. However, even correct timing is insufficient to ensure correct operation, due to subtleties involving switching thresholds [38]. That said, an increasing number of organizations use designs that incorporate some asynchronous portions [12]. A few processors, for instance the ILLIAC [33], the Caltech Asynchronous Microprocessor [23], and the ACT11 [15] have been constructed without global clocks.

The challenges of asynchronous design revolve around timing instabilities and sensitivities, which usually mean that verification requires exhaustive simulation. As a result of both the benefits and the challenges, a lively literature has grown up around the design and verification of asynchronous logic. There are essentially three major threads of inquiry:

(i) specification of a semantic or behavioral model of the circuit,

(ii) synthesis of the gate-level circuitry from this specification, and

(iii) simulation of the circuit to verify its correct performance.

Semantic specifications are often given using process algebras like the π-calculus [31] or CSP [17]. The latter gained some traction when Martin [22] described how to compile a version of CSP into a gate-level logic circuit. He later refined this compilation to generate quasi-delay insensitive circuits only [24], a class of circuits that was later shown to be Turing-complete [21]. Quasi-delay insensitivity requires that the circuit be insensitive to wire delays, except for certain pairs of signals that are assumed to arrive nearly simultaneously. It would seem that complete delay insensitivity would be more desirable, but Brzozowksi [5] showed that this unduly limits the circuits that can be constructed. After these initial efforts, the underlying theory of specifications for asynchronous circuits has continued to develop, often drawing upon methods in formal proof for mathematics [16], logic [39], and computer science [10].

In addition to Martin's work, other researchers have pursued asynchronous logic synthesis approaches. A detailed survey of some of them is given in [11]. Readers familiar with Petri nets will find the unified synthesis treatment in [14] particularly satisfying. In some cases, researchers have succeeded in more exhaustive *ad hoc* approaches, such as [9]. Other methods have focused on robustness against faults [30] or hierarchical design [32].

Once a circuit has been synthesized, its behavior should be verified against the original specification. The most straightforward simulation involves generating accurate timeseries of the voltage or logic signals in the circuit. Since there are

manufacturing variations, it is helpful to propagate ambiguous logic values during switching transitions [7]. Most modern approaches for verification generally involve symbolic event-level simulation that is motivated by temporal logic [4]. However, this approach usually suffers from a state explosion unless the original specifications are taken into account [8]. Using algebraic manipulations [20] also cuts back on the computational load of such a simulation. A different way to address this aspect of the problem is to switch to an intuitionist logic model, in which delays are explicitly uncertain. This approach has been exploited in a very elegant work by Mendler [27] and his subsequent propositional stablization theory [28].

Verification tools have been unified into the hardware description languages used for design [13], and address hierarchical design workflows [37]. We refer the reader to surveys [26] and [1] for more extensive treatment of asynchronous simulation.

In contrast to verification by temporal logic or timeseries simulation, the approach taken in this article suggests that the topological structure of a logic circuit plays an important role in determining its behavior. This appears to be related to the recent approach in [29], but topological analysis of electrical circuits is not new. Indeed, algebraic topology can be used to show that the usual formulation of electrical circuit laws results in a solution for voltages and currents [34]. Branin [2] showed how a topological approach can be extended to address a wide class of network-related problems. Moreover, Smale [36] showed that the differential equations describing electrical network behavior can be derived from homology theory. Smale's results were subsequently extended for more general circuit elements by Calvert [6]. This dynamical viewpoint can also be understood using the topology of manifolds [25].

1.2 Our approach

In order to shorten the design cycle for asynchronous circuits, it is desirable to bridge the gap between static logic state computation and event-level simulation. Ideally, such a technique would avoid both the level of detail and the computational burden of exhaustive simulation, while providing coarse semantic properties that static logic computation cannot address. This article describes a way to encode slightly more information than the netlist (the gates and connections) and logic values on the wires.

Specifically, we assume that consistent logic states may extend over portions of the circuit or over the entire circuit. Consistent logic states that cannot be extended consistently over the entire circuit correspond to transient states: inconsistency occurs along wires where the logic value is changing. *In this way, we are able to examine circuit behavior that involves unknown delays along wires in an implicit fashion: we never need to specify the delays as variables as in [20] nor do we make any assumption about delays remaining fixed during the operation of the circuit.* Since this information is local in the usual topology induced on the directed graph describing circuit connections, the natural computational framework is that of *sheaf theory*. From the outset, a direct application of sheaf theory to logic circuits results in significant computational difficulties since the natural sheaf is not one of abelian

groups.

However, by lifting the logic values from binary values into an abstract vector space spanned by logic 0 and logic 1, we obtain new information from the sheaf (at the level of its global sections, rather than just in its relative cohomology) and computations become straightforward exercises in linear algebra. This seemingly abstract trick corresponds to using one-hot signaling [11], which is used to provide error detection in existing asynchronous interfaces. Mathematically, using this encoding gives the resulting *switching sheaf* enough freedom to describe global logic states that are the superposition of two transitional states; essentially by capturing undefined signals and signal collisions. Therefore, by examining the cohomology of switching sheaves, certain behaviors can be detected in addition to the static logic states. The main result is that the first cohomology group of switching sheaves is generated by all the feedback loops that have the potential to latch or cause glitches. On the other hand, we show that combinational logic circuits in which each input is used exactly once (and therefore cannot glitch) have trivial first cohomology. Therefore, it appears that the first cohomology group contains certificates of truly asynchronous behavior.

As an aside, we note that without one-hot signalling, a sheaf theoretic approach to this problem could still proceed by looking for obstructions to extending local logic states. We hold out some hope that a coarser obstruction theory exists (for switching sheaves, using one-hot signalling) that is more refined than the one presented in this article yet less exhaustive than a complete simulation.

1.3 Outline of the paper

In Section 2 we give the basic definitions and highlight the relevant results from sheaf theory. Section 3 describes our encoding of a logic circuit as a switching sheaf. In Section 4, we show how the cohomology group of a switching sheaf captures the logic states that arise from sustained feedback. We give three examples of switching sheaves and computation of their cohomology in Section 5, culminating in a demonstration that the cohomology of a switching sheaf carries more information than the list of logic states. Finally, the results are discussed in Section 6.

2 Highlights from sheaf theory

A sheaf is a mathematical tool for storing local information over a domain. It assigns some algebraic object, a vector space in our case, to each open set, subject to certain compatibility conditions. These conditions are of two kinds: (1) those that pertain to restricting the information from a larger to a smaller open set, and (2) those that pertain to assembling information on small open sets into information on larger ones. What is of particular interest is the relationship of the global information, which is valid over the entire graph, to the topology of that graph. This is captured by the cohomology of the sheaf, in the way we summarize here.

2.1 Elementary definitions for sheaves

In this section, we follow the introduction to sheaves given in Appendix 7 of [18], largely for its direct treatment of sheaves over tame spaces. For more a more general, and more traditional approach, compare our discussion with [3].

In this article, we will work with graphs as *one-dimensional topological spaces* instead of finite sets of vertices and edges. Specifically, we will use a geometric realization associated to a directed graph $G = (V, E)$. As is usual, each edge $e \in E$ is an ordered pair (v_1, v_2) where $v_1, v_2 \in V \sqcup \perp$. We use $\perp$ to indicate that an edge e is not connected to any vertex in the graph, or briefly that e has an *external connection*. Let Y be the disjoint union of points indexed by V and closed intervals $[0, 1]$ indexed by E. (We'll usually abuse notation by identifying the points with the vertices in V and intervals with the edges in E.) We let X be the space formed from Y by attaching the intervals to the points as follows: for the interval indexed by $e = (v_1, v_2) \in E$,

(i) if the left endpoint of the interval is not $\perp$, attach this endpoint to the vertex v_1. We call e an *outgoing edge* for v_1.

(ii) If the right endpoint of the interval is not $\perp$, attach this endpoint to the vertex v_2. We call e an *incoming edge* for v_2.

The resulting topological space captures the graph structure, and the labels *incoming* and *outgoing* capture directional structure of the graph.

A *presheaf* F on a topological space X is the assignment of a vector space $F(U)$ to each open set U and the assignment of a linear map $\rho_U^V : F(U) \to F(V)$ for each inclusion $V \subseteq U$. This assignment is required to be *functorial*, by which we mean that if $U \subseteq V \subseteq W$, then $\rho_W^U = \rho_V^U \circ \rho_W^V$, and that ρ_U^U is the identity. We call the map ρ_U^V the *restriction map* from U to V. Elements of $F(U)$ are called *sections* of F *defined over* U.

A *sheaf* $\mathcal{F}$ is a presheaf F that satisfies the gluing axioms:

- (Monopresheaf) Suppose that $u \in F(U)$ and that $\{U_1, U_2, ...\}$ is an open cover of U. If $\rho_U^{U_i} u = 0$ for each i, then $u = 0$ in $F(U)$. Simply: sections that agree everywhere locally also agree globally.
- (Conjunctivity) Suppose $u \in F(U)$ and $v \in F(V)$ are sections such that $\rho_U^{U \cap V} u = \rho_V^{U \cap V} v$. Then there exists a $w \in F(U \cup V)$ such that $\rho_{U \cup V}^U w = u$ and $\rho_{U \cup V}^V w = v$. In other words, sections that agree on the intersection of their domains can be "glued together" into a section that is defined over the union of their domains of definition.

Standard examples of sheaves are

- The collection of continous real-valued functions $C(X, \mathbb{R})$ over a topological space X.
- The collection of locally constant functions, which essentially assigns a constant to each connected component of each open set.

In contrast, the collection of *constant functions* does not form a sheaf.

Sheaves are often most easily constructed by the process of *sheafification*, taking a presheaf to a unique sheaf that preserves its structure. Intuitively, given a presheaf F on a space X, one defines the sheaf $\mathcal{F}$ by its sections on an arbitrary open set $U \subseteq X$ by gluing all sections in U that agree on their overlaps. To define this precisely, we need the notion of a *stalk* A_x of a presheaf at a point $x \in X$, which is the direct limit over all $A(U)$ for which U contains x. We then define the sheafification $\mathcal{F}(U)$ to be the set of all functions $f \in \prod_{x \in U} A_x$ such that for each $x \in U$ there exists an open subset $V \subseteq U$ containing x and a $g \in F(V)$ for which $f|_V = g$.

There are six famous operations on sheaves that are important in the general theory, but only one of them (cohomology) play a role in this article.

2.2 Cohomology

We can recast the conjunctivity axiom as measuring the kernel of the linear map $d : \mathcal{F}(U) \oplus \mathcal{F}(V) \to \mathcal{F}(U \cap V)$ given by $d(x, y) = \rho_U^{U \cap V} x - \rho_V^{U \cap V} y$. Indeed, all of the elements of the kernel of such a linear map correspond to the agreement of sections on $U \cap V$. On the other hand, the monopresheaf axiom indicates that the preimage of zero under the map d corresponds to the restriction of these glued sections onto each of U and V. Indeed, any nonzero element of the *image* of d cannot be a section over $U \cup V$.

These two points motivate a computational framework for working with sheaves, called the Čech construction.

Suppose $\mathcal{F}$ is a sheaf on X, and that $\mathcal{U} = \{U_1, U_2, ...\}$ is a cover of X. We define the *Čech cochain spaces* $\check{C}^k(\mathcal{U}; \mathcal{F})$ to be the direct sum of the spaces of sections over each k-wise intersection of elements in $\mathcal{U}$. That is $\check{C}^k(\mathcal{U}; \mathcal{F}) = \bigoplus \mathcal{F}(U_{i_1} \cap ... \cap U_{i_k})$.

We define a sequence of linear maps $d^k : \check{C}^k(\mathcal{U}; \mathcal{F}) \to \check{C}^{k+1}(\mathcal{U}; \mathcal{F})$ by

$$d^k(\alpha)(U_1, U_2, ..., U_{k+1}) = \sum_{i=0}^{k+1} (-1)^i \rho_{U_0 \cap ... \cap U_{k+1}}^{U_0 \cap ... \hat{U}_i ... \cap U_{k+1}} \alpha(U_0 \cap ... \hat{U}_i ... \cap U_{k+1}),$$

where the hat means that an element is omitted from the list. Note that these fit together into a sequence, called the *Čech cochain complex*:

$$0 \to \check{C}^0(\mathcal{U}; \mathcal{F}) \xrightarrow{d^0} \check{C}^1(\mathcal{U}; \mathcal{F}) \xrightarrow{d^1} ...$$

A standard computation shows that $d_k \circ d_{k-1} = 0$, so that we can define the *k-th Čech cohomology space* $\check{H}^k(\mathcal{U}; \mathcal{F}) = \ker d_k / \text{image } d_{k-1}$.

The $\check{H}^k$ apparently depend on the choice of cover $\mathcal{U}$, but for good covers [3] (much as in the Nerve Lemma), this dependence vanishes. Leray's theorem for sheaves states that $\check{H}^k(\mathcal{U}; \mathcal{F})$ is the same for each good cover. So we write $H^k(X; \mathcal{F}) = \check{H}^k(\mathcal{U}; \mathcal{F})$, the sheaf's *cohomology* in the case that $\mathcal{U}$ is a good cover.

A little thought about good covers on graphs reveals two important facts:

[3] A *good cover* $\mathcal{U}$ is one that consists of contractible sets, for which all intersections of finitely many elements of $\mathcal{U}$ are also contractible

- if X is a geometric realization of a graph, $H^k(X;\mathcal{F}) = 0$ for $k > 1$, and
- $H^0(X;\mathcal{F})$ is isomorphic to the space of global sections $\mathcal{F}(X)$.

By analogy with the Mayer-Vietoris sequence for homology, there is a Mayer-Vietoris sequence for sheaf cohomology [3]. Suppose that A, B are two open subspaces of a graph X that cover X, and that F is a sheaf over X. Then the following *Mayer-Vietoris sequence* is an exact sequence:

$$... \to H^k(X;\mathcal{F}) \xrightarrow{r} H^k(A;\mathcal{F}) \oplus H^k(B;\mathcal{F}) \xrightarrow{d} H^k(A \cap B;\mathcal{F}) \xrightarrow{\delta} \xrightarrow{\delta} H^{k+1}(X;\mathcal{F}) \to ...$$

In this sequence, r comes from restriction maps in the obvious way, d is the composition of restriction maps and a difference: $d(x,y) = \rho_A^{A\cap B}x - \rho_B^{A\cap B}y$, and δ is the connecting homomorphism. Notation has been abused above slightly: by $H^k(A;\mathcal{F})$ we mean the k-th cohomology of the sheaf F restricted to subsets lying in A.

The Mayer-Vietoris exact sequence simplifies if A and B are disjoint. In this case, one simply obtains that $H^k(X;\mathcal{F}) = H^k(A;\mathcal{F}) \oplus H^k(B;\mathcal{F})$ for each k.

3 Construction of a switching sheaf from a circuit

This section describes a way to associate a sheaf structure to a directed graph that encodes a logic circuit. Each vertex represents a logic gate, where the in-degree represents the number of inputs. Each edge of the graph corresponds to a 1-bit signal connecting the the input of one gate to the output of another. We allow edges to be self-loops (connecting the input of a gate to an output of the same gate) and external connections. As existing logic circuits contain finitely many gates, we assume that the underlying graph is finite, but not necessarily connected.

3.1 Quiescent logic states, one-hot encoding, and categorification

We begin with a brief description of the circuit model to be encoded. As the sheaf structure will require logic functions to be *linear* functions, we categorify them. This is accomplished by the relatively standard one-hot encoding of logical values.

Suppose that X is a directed graph in which each vertex has finite degree. A *logic circuit* is the assignment of a function $f_v : \mathbb{F}_2^{m(v)} \to \mathbb{F}_2^{n(v)}$ to each vertex v, where $m(v)$ is the in-degree of v and $n(v)$ is the out-degree of v. We call f_v the *logic gate* at v.

Given a logic circuit, a *quiescent logic state* (QLS) is an assignment $s : E \to \mathbb{F}_2$ of a binary value to each edge, such that for each vertex v, $f_v(s(e_1^+), s(e_2^+), ...) = (s(e_1^-), s(e_2^-), ...)$ where $\{e_i^+\}$ are the incoming edges at v and $\{e_i^-\}$ are the outgoing edges at v.

In this article, we examine *one-hot* encoding T of binary values in a logic circuit. That is, we consider the function $T : \mathbb{F}_2 \to \mathbb{F}_2^2$ where

$$T(0) = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$$

$$T(1) = \begin{pmatrix} 0 \\ 1 \end{pmatrix}.$$

In this way, the elements of $\mathbb{F}_2$ become the elements of the standard vector space basis for $\mathbb{F}_2^2$.

Applying this replacement to each occurance of $\mathbb{F}_2$ in the definition of a logic circuit and logic state results in a particular *categorification* of logic circuits. Indeed, each of the logic gates f_v become *linear* functions Tf_v between $\mathbb{F}_2$ vector spaces.

Casual examination of the categorification procedure suggests that very little has changed, except the algebraic structure has been slightly enhanced. However, two new things have occured:

- Problems of logic can now be addressed computationally using the framework of linear algebra. This can result in gains in asymptotic computational complexity. Rather than being forced to enumerate states, one may instead perform standard polynomial-time linear algebra (over the finite field $\mathbb{F}_2$).
- It is possible to superpose two logic states, and thereby study certain kinds of transitions between logic states. This is subtle and somewhat surprising: we have not explicitly described anything about time evolution of circuits, and indeed the usual way of examining the QLS of a logic circuit does not concern itself with time. However, by permitting superposed states, we are able to study the circuit's response to both *simultaneously* and thereby discern the way that one might transition to the other.

3.2 Switching sheaves

Suppose that X is a directed graph with the usual topology, let $\mathcal{U} = \{U_\alpha, V_\beta\}$ be a base for the topology of X where each U_α is connected and contains exactly one vertex and each V_β is contained in the interior of a single edge. A *switching sheaf* $\mathcal{S}$ on X is the sheafification of the following presheaf S, defined on $\mathcal{U}$:

- $S(U_\alpha)$ is the tensor product of copies $\mathbb{F}_2^2$, one for each incoming edge into the unique vertex contained in U_α,
- $S(V_\beta) = \mathbb{F}_2^2$,
- if $V_\beta \subset U_\alpha$ and V_β is contained in the n-th incoming edge, the restriction map $S(U_\alpha) \to S(V_\beta)$ is the contraction onto the n-th $\mathbb{F}_2^2$ factor of $S(U_\alpha)$,
- if $V_\beta \subset U_\alpha$ and V_β is contained in the n-th outgoing edge, then there is a fixed $\mathbb{F}_2$-linear map $\phi_v : (U_\alpha) \to S(V_\beta)$ depending only on the vertex v contained in U_α and n (the outgoing edge). This collection of maps $\{\phi_v\}$ for vertices v, is the one-hot encoding $\phi_v = Tf_v$ of the logic function f_v located at the vertex v as described in the previous section.

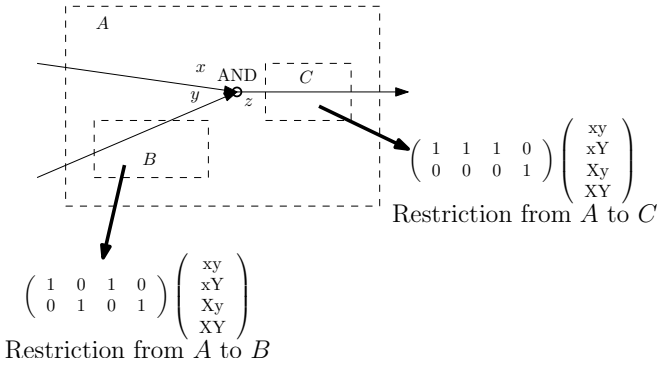

Fig. 1. An example of a switching sheaf (in this figure xY denotes $\overline{x} \otimes y$)

Figure 1 gives an example of a switching sheaf over a graph with one vertex, which represents an AND gate. Notice that the dimension of the sheaf over B and C is 2, while it has dimension 4 over A.

When we treat Čech cohomology with respect to the cover $\mathcal{U}$, we will use the notation $\check{H}^k(X;\mathcal{S})$, rather than $\check{H}^k(\mathcal{U};\mathcal{S})$, to emphasize that this choice of cover is being used.

Proposition 3.1 *Suppose $\mathcal{S}$ is a switching sheaf over a logic circuit X. Every QLS of this logic circuit lifts via T to an element of $H^0(X;\mathcal{S})$. Conversely, every element of $H^0(X;\mathcal{S})$ that restricts to $\binom{1}{0}$ or $\binom{0}{1}$ on each edge is the image of a QLS through this lift. Any such section is nonvanishing.*

Proof. Given a QLS s, it is clear that $\sigma = T \circ s$ defines a section of $\mathcal{S}$ over the edges of X. We only need to address the value of this lifted section at the vertices. The correct answer is easy to obtain. Suppose v is a vertex with incoming edges $\{e_1^+, e_2^+, ..., e_k^+\}$. Then the appropriate definition for $\sigma(v)$ is $(T\circ s)(e_1^+)\otimes(T\circ s)(e_2^+)\otimes ... \otimes (T \circ s)(e_k^+)$. The definition of $\phi_v = Tf_v$ ensures that the lifts of each outgoing edge through T agrees with our choice for $\sigma(v)$. Therefore, σ lifts to an element of $H^0(X;\mathcal{S})$, which we define as Ts.

Conversely, suppose we have a global section τ of $\mathcal{S}$ that restricts to $\binom{1}{0}$ or $\binom{0}{1}$ on each edge. Suppose that U is a contractible open set containing a single vertex v, with incoming edges $\{e_1^+, e_2^+, ..., e_k^+\}$. Since the map $\mathcal{S}(U) \to \mathcal{S}(e_i^+)$ for each incoming edge e_i^+ is a contraction onto the i-th factor, we have that $\tau(v) = (Ta_1) \otimes (Ta_2) \otimes \cdots \otimes (Ta_n)$ where (Ta_i) is the value of τ on the i-th incoming edge. Clearly, this is well-defined since the image of T on $\mathbb{F}_2$ is the two-element set $\{\binom{1}{0}, \binom{0}{1}\}$.

If V contains the interiors of all outgoing edges for v, then by the definition of the switching sheaf $\mathcal{S}$, $\mathcal{S}(U) \to \mathcal{S}(U \cap V)$ is a linear map

$$\begin{aligned} \phi_v(\tau(v)) &= \phi_v((Ta_1) \otimes (Ta_2) \otimes \cdots \otimes (Ta_n)) \\ &= (Tf_v)((Ta_1) \otimes (Ta_2) \otimes \cdots \otimes (Ta_n)) \\ &= T(f_v(a_1, a_2, ...a_n)), \end{aligned}$$

which indicates that τ is the image of some QLS, whose incoming edges at v have values $a_1, a_2, ...a_n$. This computation makes use of the commutative diagram

$$\begin{array}{ccc} \mathbb{F}_2^{2^{2n}} & \xrightarrow{Tf_v=\phi_v} & \mathbb{F}_2^{2^{2m}} \\ {\scriptstyle T}\uparrow & & {\scriptstyle T}\uparrow \\ \mathbb{F}_2^{2n} & \xrightarrow{f_v} & \mathbb{F}_2^{2m} \end{array}$$

Such a section of $\mathcal{S}$ is nonvanishing: for any QLS s, the function $(T \circ s)$ is clearly nonvanishing on the edges. At vertices, the lift takes values that are the tensor product of the incoming edge values, which are all nonzero. □

4 The content of the cohomology of a switching sheaf

In this section, we use the Mayer-Vietoris sequence to examine how switching sheaf cohomology changes as a circuit is progressively assembled. In this way, we describe an incremental method for computing switching sheaf cohomology that mimics the way a prototype circuit could be "soldered" together. The effect of adding an unconnected gate is straightforward, but adding a single connection wire reveals the meaning of H^1: nontrivial elements of H^1 correspond to sustained feedback states. Their presence is therefore an indication of possible latching (stable feedback) or glitches (unstable feedback, usually caused by race conditions). Currently, we do not know how to use the switching sheaves to discriminate between the two kinds of feedback, as the formalism apparently corresponds to an unbounded wire delay model.

We first consider the effect of adding a new gate G to a circuit A, but not connecting the two. We therefore consider the switching sheaf $\mathcal{S}$ on $X = A \sqcup G$, where G is a single vertex. The Mayer-Vietoris sequence in this case consists only of the isomorphism $H^k(X;\mathcal{S}) \cong H^k(A;\mathcal{S}) \oplus H^k(G;\mathcal{S})$ for all k. However, we note immediately that $H^k(G;\mathcal{S})$ is trivial for $k > 0$ since the covering dimension of G is zero. Thus, H^1 is unchanged by adding an unconnected logic gate to a circuit, and the the dimension of H^0 increases by $2^{\#\text{ inputs of } G}$.

In order to explain the effect of attaching a wire W to an existing circuit A (see Figure 2), we construct the Mayer-Vietoris sequence for a switching sheaf $\mathcal{S}$ on $X = A \cup W$. In order to ensure the correct interpretation, we assume W is a connected subset of an edge and A is homotopy equivalent to $X - W$. The Mayer-Vietoris sequence in this case is (we suppress the sheaves from the notation)

$$0 \to H^0(X) \to H^0(A) \oplus \mathbb{F}_2^2 \xrightarrow{\Delta} \mathbb{F}_2^4 \to H^1(A \cup W) \to H^1(A) \to 0.$$

Note that exactness requires that $\dim H^1(A \cup W;\mathcal{S}) \geq \dim H^1(A;\mathcal{S})$. Observe that

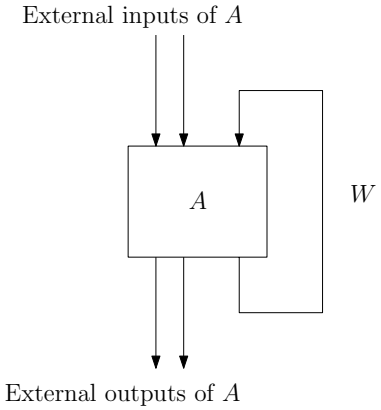

Fig. 2. Circuit A with a feedback wire W attached

the difference map takes the form

$$\Delta = \begin{pmatrix} P_{2\times k} & I_{2\times 2} \\ Q_{2\times k} & I_{2\times 2} \end{pmatrix},$$

where $I_{2\times k}$ is a 2 by 2 identity matrix, $P_{2\times k}$ and $Q_{2\times k}$ are 2 by k matrices, and k is the dimension of $H^0(A;\mathcal{S})$. The matrix $P_{2\times k}$ represents the restriction of sections over A to the output of the wire W, or equivalently to the particular input of A where the wire attaches. In much the same way, $Q_{2\times k}$ is the restriction from the sections of A to the particular output of A that is attached to the wire. Observe that a pair of row reductions on Δ results in the matrix

$$\begin{pmatrix} P_{2\times k} & I_{2\times 2} \\ Q_{2\times k} - P_{2\times k} & 0_{2\times 2} \end{pmatrix},$$

which has rank 2, 3, or 4. The rank of Δ depends how much the wire participates in the feedback of signals, so we assign names to the three possibilities:

- rank $\Delta = 2$: *complete feedback*, in which $Q_{2\times k} = P_{2\times k}$. This occurs when the input and output of A that the wire connects always agree.
- rank $\Delta = 3$: *partial feedback*.
- rank $\Delta = 4$: *no feedback*. This case occurs especially when the wire W connects two disconnected components of A, but more generally when the input and output connected by W are completely independent.

Therefore,

$$\dim H^0(X;\mathcal{S}) = \dim H^0(A;\mathcal{S}) - \begin{cases} 0 & \text{if complete feeback} \\ 1 & \text{if partial feedback} \\ 2 & \text{otherwise} \end{cases}$$

and

$$\begin{aligned}\dim H^1(X;\mathcal{S}) &= \dim H^1(A;\mathcal{S}) + 4 - rank\,\Delta \text{ (by exactness)} \\ &= \dim H^1(A;\mathcal{S}) + \begin{cases} 2 & \text{if complete feeback} \\ 1 & \text{if partial feeback} \\ 0 & \text{otherwise} \end{cases}\end{aligned}$$

The effect of attaching W is best described by the following slogan:

- Attaching a wire that does not participate in feedback suppresses logic states and leaves H^1 unchanged.
- Attaching a wire that participates in feedback leaves logic states unchanged and adds to the dimension of H^1.

This result indicates that the underlying space's Cech cohomology plays a role in the resulting switching sheaf cohomology. Race conditions and feedback *loops* that the switching sheaf detects are just that: they can only arise in the presence Cech cocycles in the underlying space that have a specific structure with respect to the logic gates in the circuit.

5 Examples of switching sheaves and their cohomology

In this section, we exhibit the cohomology of switching sheaves and its interpretation by way of three illustrative examples: combinational circuits with and without shared inputs and an RS flip-flop. These examples indicate that H^0 of a switching sheaf contains at least as many elements as the set of QLS. Additionally, as was shown in Section 4, H^1 of a switching sheaf captures information about the presence of feedback or race conditions. We give two explicit examples of this fact.

5.1 Combinational circuits without shared inputs

Let us consider the case of a switching sheaf $\mathcal{S}$ on a connected, directed tree X. (The choice of directions on the edges of X does not effect the cohomology of $\mathcal{S}$.) This represents the situation in which each external input is used at most once in the production of each external output. In this case, $\{X\}$ by itself is a good cover, so we conclude that $H^1(X;\mathcal{S})$ is trivial. Observe that the combinatorial Euler characteristic of X is 1 by the same reasoning, so that the number of vertices of X is 1 more than the number of internal edges. Thus, if there are n vertices with in-degrees $\{m_1, ...m_n\}$,

$$\dim H^0(X;\mathcal{S}) = \dim \check{C}^0 - \dim \check{C}^1 = \sum_{i=1}^{n} 2^{m_i} - 2(n-1).$$

Looking at a basis for $H^0(X;\mathcal{S})$ is instructive, so consider the logic circuit shown in Figure 3. This circuit consists of a single m-input logic gate. One of the input edges (labeled with signals a and $\bar{a}$) is extended to include a single 1-input buffer

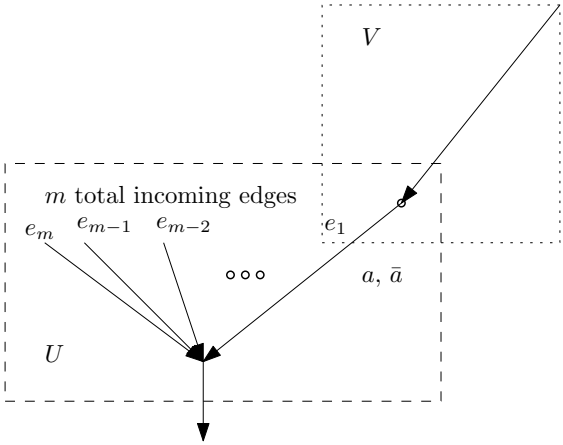

Fig. 3. A single logic gate with an extended input edge

gate (identity function). We compute the sheaf cohomology and a basis for this cohomology using a Čech complex. This complex has the form

$$0 \to \underbrace{\mathbb{F}_2^2 \otimes \cdots \otimes \mathbb{F}_2^2}_{m \text{ factors}} \xrightarrow{d^0} \mathbb{F}_2^2 \to 0.$$

The matrix form of d^0 is

$$d^0 = \begin{pmatrix} 1 \cdots \text{total of } 2^{m-1} \text{ ones} \cdots 1\ 0 & \cdots & 0\ 1\ 0 \\ 0 & \cdots & 0\ 1 \cdots \text{total of } 2^{m-1} \text{ ones} \cdots 1\ 0\ 1 \end{pmatrix}$$

which evidently has full rank. Hence, the dimension of $H^1(X;\mathcal{S})$ is zero. The dimension of $H^0(X;\mathcal{S})$ is 2^m, which is the same as the number of QLS for the logic circuit. We would therefore expect that $\check{H}^0(X;\mathcal{S})$ is spanned by images under T of QLS, and this is the case. A basis is

$$\overline{a} + \overline{e_1} \otimes \overline{e_2} \otimes \cdots \otimes \overline{e_m}$$

$$\overline{a} + \overline{e_1} \otimes e_2 \otimes \cdots \otimes \overline{e_m}$$

$$\cdots$$

$$\overline{a} + \overline{e_1} \otimes e_2 \otimes \cdots \otimes e_m$$

$$a + e_1 \otimes \overline{e_2} \otimes \cdots \otimes \overline{e_m}$$

$$a + e_1 \otimes e_2 \otimes \cdots \otimes \overline{e_m}$$

$$\cdots$$

$$a + e_1 \otimes e_2 \otimes \cdots \otimes e_m$$

where a is supported on V, and the other term is supported on U. Notice in particular that all sections are supported over the entirety of X, and all restrict to

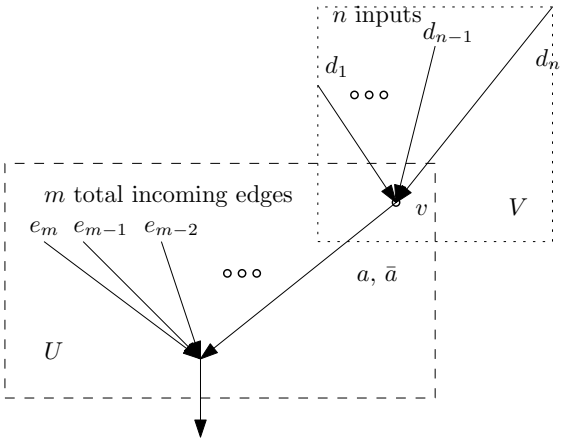

Fig. 4. Two logic gates composed

$\binom{0}{1}$ or $\binom{1}{0}$ on edges. Hence, this basis consists of images of QLS.

Given a situation like that shown for the graph Y in Figure 4, we observe that the number of QLS present is 2^{n+m-1}. However, the dimension of $H^0(Y;\mathcal{S})$ differs from this number. We construct the Čech coboundary map d^0 in matrix form

$$d^0 = \left(\begin{array}{cccccc|c} 1 & \cdots & 1 & 0 & \cdots & 0 & \overline{f_v} \\ 0 & \cdots & 0 & 1 & \cdots & 1 & f_v \end{array}\right),$$

where by f_v we mean a 1×2^n submatrix with zeros in the entries corresponding to gate v taking output value 0. The coboundary map is evidently of full rank, so that $H^0(Y;\mathcal{S})$ has dimension $2^n + 2^m - 2$, and $H^1(Y;\mathcal{S})$ is trivial.

Suppose that f_v has k nonzero entries. We note that $\check{H}^0(x;\mathcal{S})$ has a basis that consists of images of QLS under T. There are $k + 2^{n-1} - 1$ basis elements of the form (in particular, the first term is where f_v is nonzero and e_1 participates in the second term)

$$d_1 \otimes \cdots \otimes d_n + e_1 \otimes \cdots \otimes e_m.$$

There are additionally $2^m - k + 2^{n-1} - 1$ elements of the form (in which $\overline{e_1}$ participates in the second term)

$$d_1 \otimes \cdots \otimes d_n + \overline{e_1} \otimes \cdots \otimes e_m.$$

This proves the obvious fact that if no inputs are shared in a combinational circuit, then the entire circuit has no interesting asynchronous behavior. It should therefore be possible to prove the following conjecture, though we have not yet succeeded.

Conjecture 5.1 *If $\mathcal{S}$ is a switching sheaf over a directed tree X, then $H^0(X;\mathcal{S})$ has a basis that consists of lifted QLSs.*

This means that any section over X that vanishes anywhere must be the linear superposition of two or more QLS, and therefore describes uncertainty or transient states.

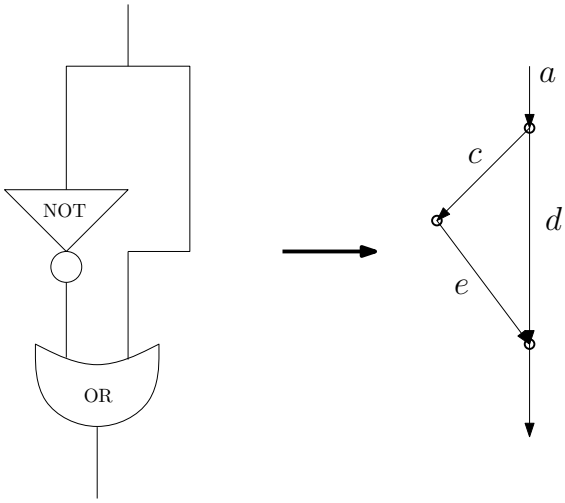

Fig. 5. A combinational logic circuit with a shared input signal

5.2 Combinational circuits with shared inputs

The circuit shown in Figure 5 does not satisfy the hypotheses of Conjecture 5.1. In particular, it contains two separate signal paths for the input a. It should be clear that as a logic circuit, this has two QLS: one for each binary input value. However, assuming that there is some delay in the circuit, the signal labeled e will be delayed from the ideal signal $\overline{a}$. This means that there is some time-sensitivity in the circuit, and it can therefore produce glitches (narrow pulses on its output) when the input is changed.

If we consider a switching sheaf over the logic circuit, we obtain a Čech coboundary matrix that has the form

$$d^0 = \left(\begin{array}{cc|cc|cccc} 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ \hline 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 \end{array}\right).$$

Row reduction of this matrix reveals a basis of sections supported over the entire graph:

$$\overline{a} + \overline{c} + \overline{d} \otimes e$$

$$a + c + d \otimes \overline{e}$$

$$a + \overline{a} + c + \overline{c} + d \otimes e + \overline{d} \otimes \overline{e}$$

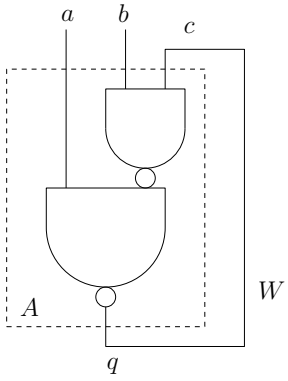

Fig. 6. An R-S flip-flop circuit

It is apparent that the first two basis elements are lifts of the QLS. However, the second is clearly neither a lift of a QLS, nor a linear combination of them. Indeed, it indicates that ambiguity in the input logic value (such as occurs during a transition) causes ambiguity throughout the rest of the circuit. It is therefore an *algebraic* indication of the presence of time-sensitivity of the circuit.

In addition to the presence of the additional basis element for $\check{H}^0(Y;\mathcal{S})$, there is another indication of additional information. $\check{H}^1(Y;\mathcal{S})$ is nontrivial in the case of this logic circuit, and is generated by $\overline{c}+c+\overline{d}+d+\overline{e}+e$, which indicates that the source of the time-sensitivity is the two separate signal paths for the input.

This calculation proves the following

Theorem 5.2 *The cohomology of a switching sheaf over a logic circuit contains different information than the set of its quiescent logic states.*

5.3 An R-S flip-flop

There are other switching sheaf structures that can be constructed over a graph with one (undirected) loop. While glitches are one kind of time sensitive behavior, another is the latching of a transient input. We therefore give a classic example of a circuit that exhibits latching, the R-S flip-flop.

Consider the circuit X shown in Figure 6, which we split into two pieces: a combinational circuit A with a 3-input gate, and a feedback wire W. The QLS for this circuit [35] are summarized in the following table:

a	b	c	q	Description
0	0	1	1	Danger
0	1	1	1	Set
1	0	0	0	Reset
1	1	0	0	Hold zero
1	1	1	1	Hold one

Looking at the difference map Δ for the Mayer-Vietoris sequence, we note that

$$P = \begin{pmatrix} 1\,0\,1\,0\,1\,0\,1\,0 \\ 0\,1\,0\,1\,0\,1\,0\,1 \end{pmatrix}$$

and

$$Q = \begin{pmatrix} 0\,0\,0\,0\,1\,1\,1\,0 \\ 1\,1\,1\,1\,0\,0\,0\,1 \end{pmatrix}.$$

The resulting matrix for Δ has rank 3, so that $H^0(X;\mathcal{S})$ has dimension 7 and $H^1(X;\mathcal{S})$ has dimension 1. Here is a basis for $\check{H}^0(X;\mathcal{S})$:

Element of $\check{H}^0(X;\mathcal{S})$	Description
$\overline{a} \otimes \overline{b} \otimes c$	Danger
$\overline{a} \otimes b \otimes c$	Set
$a \otimes \overline{b} \otimes \overline{c}$	Reset
$a \otimes b \otimes \overline{c}$	Hold zero
$a \otimes b \otimes c$	Hold one
$\overline{a} \otimes \overline{b} \otimes \overline{c} + a \otimes \overline{b} \otimes c$	Transition between Danger and Reset
$\overline{a} \otimes \overline{b} \otimes \overline{c} + \overline{a} \otimes b \otimes \overline{c}$	Transition between Danger and Set

Of most interest are the last two basis elements. These are linear combinations of two terms, neither of which is a lift of a QLS. The most suggestive interpretation is that they imply an uncertainty when exiting the Danger state. As the inputs a and b transition from both logic 0 to both logic 1, there is a race condition. Only one of them transitions first, so there is a brief transition into the Set or Reset states before entering a Hold state. If we add the last two basis elements, we obtain $a \otimes \overline{b} \otimes c + \overline{a} \otimes b \otimes \overline{c}$ which indicates that an uncertainty about which of a or b transitions has occured results in uncertainty in the signal c.

6 Discussion

The cohomology of switching sheaves is a new source of information about the behavior of logic circuits, especially those circuits that are asynchronous. Especially, the presence of nontrivial elements of H^1 indicates that a circuit has feedback or a race condition. This is a somewhat coarse descriptor of circuit behavior, as should be expected from such a global topological invariant as cohomology. However, there remain important questions regarding details at finer timescales. In particular, can the cohomology of switching sheaves discriminate between glitches and latching? If H^1 is trivial, H^0 does not contain the same information as the logic states. Indeed, a *basis* for H^0 often contains less information. (The set of QLS is contained in

H^0, *considered as a set.* See Proposition 3.1.) A sharper connection to one of the popular semantic models of asynchronous logic will likely be essential in answering these questions.

References

[1] Baker, W. and A. Mahmood, *An analysis of parallel synchronous and conservative asynchronous logic simulation schemes*, in: *Sixth IEEE Symposium on Parallel and Distributed Processing*, 1994, pp. 92–99.

[2] Branin, F. H., *The algebraic-topological basis for network analogies and the vector calculus*, Technical report, IBM (1966).

[3] Bredon, G., "Sheaf theory," Springer, 1997.

[4] Browne, M., E. Clarke, D. Dill and B. Mishra, *Automatic verification of sequential circuits using temporal logic*, IEEE Transactions on Computers **C-35** (1986).

[5] Brzozowski, J. and J. Ebergen, *On the delay-sensitivity of gate networks*, IEEE Transactions on Computers **41** (1992).

[6] Calvert, B., *Unicursal resistive networks*, Circuits Systems Signal Processing **16** (1997), pp. 307–324.

[7] Chappell, S. G., *Simulation of large asynchronous logic circuits using an ambiguous gate model*, in: *Fall Joint Computer Conference*, 1971, pp. 651–661.

[8] Clarke, E. M. and O. Grümberg, *Avoiding the state explosion problem in temporal logic model checking algorithms*, in: *Proceedings of the sixth annual ACM Symposium on Principles of distributed computing*, 1987, pp. 294–303.

[9] Cox, D., *Complete synthesis method for D flip-flops with set and reset inputs*, in: *9th NASA Symposium on VLSI Design*, 2000.

[10] Dalrymple, D. A., "Asynchronous Logic Automata," Master's thesis, MIT (2008).

[11] Davis, A. and S. Nowick, *An introduction to asynchronous circuit design*, Technical Report UUCS-97-013, University of Utah Computer Science Department (1997).

[12] Edwards, D. A. and W. B. Toms, *The status of asynchronous design in industry*, Technical Report IST-1999-29119, Asynchronous Circuit Design Working Group (2004).

[13] Endecott, P. and S. Furber, *Modelling and simulation of asynchronous systems using the LARD hardware description language*, in: *Proceedings of the 12th European Simulation Multiconference on Simulation*, 1998, pp. 39–43.

[14] *et al.*, J. C., "Logic synthesis for asynchronous controllers and interfaces," Springer, 2002.

[15] *et al.*, N. K., *A flexible 8b asynchronous microprocessor based on low-temperature poly-silicon TFT technology*, in: *IEEE International Solid-State Circuits Conference*, 2005.

[16] Harrison, J., *Formal proof - theory and practice*, Notices of the AMS **55** (2008), pp. 1395–1406.

[17] Hoare, C. A. R., "Communicating Sequential Processes," 2004.

[18] Hubbard, J. H., "Teichmüller Theory, volume 1." Matrix Editions, 2006.

[19] Huffman, D. A., *The synthesis of sequential switching circuits*, Technical Report 274, Research Laboratory of Electronics, MIT (1954).

[20] Ishiura, N., M. Takahashi and S. Yajima, *Time-symbolic simulation for accurate timing verification of asynchronous behavior of logic circuits*, in: *26th ACM/IEEE Design Automation Conference*, 1989.

[21] Manohar, R. and A. Martin, *Quasi-delay-insensitive circuits are Turing-complete*, in: *Second International Symposium on Advanced Research in Asynchronous Circuits and Systems*, 1995.

[22] Martin, A., *Compiling communicating processes into delay-insensitive VLSI circuits*, Technical Report 5210:TR:86, Computer Science Department, California Institute of Technology (1986).

[23] Martin, A., *The design of an asynchronous microprocessor*, Technical Report Caltech-CS-TR-89-2, Computer Science Department, California Institute of Technology (1989).

[24] Martin, A., *The limitations to delay-insensitivity in asynchronous circuits*, Technical Report Caltech-CS-TR-90-02, Computer Science Department, California Institute of Technology (1990).

[25] Matsumato, T., *On several geometric aspects of nonlinear networks*, Journal of The Franklin Institute **301** (1976), pp. 203–225.

[26] Meister, G., *A survey on parallel logic simulation*, Technical report, University of Saarland, Department of Computer Science (1993).

[27] Mendler, M., *Timing refinement of intuitionistic proofs and its application to the timing analysis of combinational circuits*, in: P. Miglioli, U. Moscato, D. Mundici and M. Ornaghi, editors, *Proc. International Workshop on Theorem Proving with Analytic Tableaux and Related Methods (TABLEAUX'96)*, 1996 pp. 261–277.

[28] Mendler, M., *Propositional stablisation theory*, in: *Synchron 2005*, 2005.

[29] Meredith, L. G. and D. Snyder, *Knots as processes*, in: *Traced Monoidal Categories Workshop*, 2007.

[30] Monnet, Y., M. Renaudin and R. Leveugle, *Designing resistant circuits against malicious faults injection using asynchronous logic*, IEEE Transactions on Computers **55** (2006).

[31] Parrow, J., *An introduction to the π-calculus*, in: B. *et al.*, editor, *Handbook of Process algebra*, Elsevier, 2001 .

[32] Potop-Butucaru, D., *Correct-by-construction asynchronous implementation of modular synchronous specifications*, Fundamenta Informaticae **78** (2007), p. 131159.

[33] Robertson, J., *ILLIAC design techniques*, Technical report, Digital Computer Laboratory, University of Illinois (1955).

[34] Roth, J. P., *An application of algebraic topology to numerical analysis: on the existence of a solution to the network problem*, Proc. N. A. S. **41** (1955), pp. 518–521.

[35] Shannon, C., "A symbolic analysis of relay and switching circuits," Master's thesis, MIT (1940).

[36] Smale, S., *On the mathematical foundations of electrical circuit theory*, J. Differential Geometry **7** (1972), pp. 193–210.

[37] Vakilotojar, V., "Induced Hierarchical verification of asynchronous circuits using a partial order technique," Ph.D. thesis, University of Southern California (2000).

[38] van Berkel, K., *Beware the isochronic fork*, Integration **13** (1992), pp. 103–128.

[39] Vasyukevich, V. O., *Asynchronous logic elements: venjunction and sequention* (2009).