On Breakable Cyclic Definitions[*]

Jie-Hong R. Jiang, Alan Mishchenko, and Robert K. Brayton

Department of Electrical Engineering and Computer Sciences
University of California, Berkeley

ABSTRACT

In the course of hardware system design or real-time process control, high-level specifications may contain simultaneous definitions of concurrent modules whose information flow forms cyclic dependencies without the separation of state-holding elements. The temporal behavior of these cyclic definitions may be meant to be combinational rather than sequential. Most prior approaches to analyzing cyclic combinational circuits were built upon the formulation of ternary-valued simulation at the circuit level. This paper shows the limitation of this formulation and investigates, at the functional level, the most general condition where cyclic definitions are semantically combinational. It turns out that the prior formulation is a special case of our treatment. Our result admits strictly more flexible high-level specifications. Furthermore, it allows a higher-level analysis of combinationality, and, thus, no costly synthesis of a high-level description into a circuit netlist before combinationality analysis can be performed. With our formulation, when the target is software implementations, combinational cycles need not be broken as long as the execution of the underlying system obeys a sequencing execution rule. For hardware implementations, combinational cycles are broken and replaced with acyclic equivalents at the functional level to avoid malfunctioning in the final physical realization.

1. INTRODUCTION

Cyclic definitions occur commonly in high-level system descriptions (e.g., due to resource sharing, module composition, etc.) as was observed in [22]. Checking if cyclic definitions are semantically combinational is crucial in both hardware and software synthesis for two reasons. First, the analysis certifies the legitimacy of cyclic definitions. Second, if the cyclic definitions of a system are inappropriate but breakable, the analysis provides a means of rewriting the system which breaks the cycles.

The analysis of cyclic combinational circuits was first formulated by Malik [15], based on ternary-valued simulation [3, 4], at the circuit (or gate) level. Subsequent efforts [9, 21, 20, 16] were built upon this formulation and bore much the same foundation. However, the quest for solutions to analyzing combinationality remains because the analysis at the functional level was left open.

Combinationality analysis for cyclic circuits is an essential step in the compilation of synchronous languages [8], such as Esterel [1, 2], which allow simultaneous cyclic definitions. Before applying Malik's approach for static analysis[1], a synchronous program typically needs to be translated into a circuit netlist. Depending on how a program is written, the same specification can be translated into different netlists. Because the analysis based on ternary-valued simulation heavily depends on circuit structures (more precisely, on the arrangement of delay elements over circuit netlists) as observed in [21, 20], a netlist may be declared not combinational even though there exists a functionally equivalent one that behaves combinationally. This phenomenon corresponds to the so-called *schizophrenia problem* in the compilation of Esterel programs [2].

This paper proposes a functional-level analysis that avoids the complication of translating programs into circuit netlists and eliminates the discrepancy problem of analyzing equivalent different netlists. Essentially, our formulation of combinationality is extended to an extreme at the functional level. That is, there exists a combinational implementation (where cyclic definitions might need to be broken) if and only if the system in consideration passes our combinationality test. As will be clear, ternary-valued simulation, when extended to the functional level, is a limited special case of our formulation.

Although combinational circuits with feedback have their potential savings in area [13], they are hard to manipulate including timing analysis, logic minimization, etc. Breaking cyclic dependencies is sometimes necessary to avoid later complication since manipulating such circuits needs special care to prevent destroying well-behaved combinationality. Earlier efforts [9, 5] on breaking combinational cycles were done for circuit netlists. In fact, to break combinational cycles, there is no need to wait until circuit structures are derived. In addition, analyzing combinationality at the functional level broadens the generality.

Our results are established upon the following principle. When cyclic definitions are to be broken or the synthesis target is software, the combinationality analysis should be generalized to an extreme and be performed at the highest level possible (i.e., the functional level). On the other hand, when the target is hardware synthesis and combinational cycles are allowed to exist, then the analysis should be conservative enough to tolerate undesirable physical effects but general enough to abstract away unnecessary details at the appropriate level (i.e., the circuit, or gate level). We emphasize this asymmetry in analyzing combinationality, which

[*]This work was supported in part by NSF grant CCR-0312676, California Micro program, and our industrial sponsors, Fujitsu, Intel and Synplicity.

[1]For dynamic or runtime analysis, translating a program to a netlist may be avoided, e.g., see [6].

was overlooked in prior work.

For combinationality at the circuit level, we comment on a recent development in the synthesis of cyclic combinational circuits. Targeting area minimization, an attempt was made in [17, 18] to synthesize cyclic combinational circuits by extending the formulation of ternary-valued simulation at the functional level. Unfortunately, it was ignored that functional-level analysis is not sufficient in guaranteeing well-behaved circuitry. This paper shows the pitfall and suggests two cures. Essentially, additional conditions other than purely functional ones need to be applied in order to guarantee well-behaved circuitry.

The paper is organized as follows. After preliminaries and notations are given in Section 2, our formulation of combinationality at the functional level will be introduced in Section 3. Section 4 discusses some issues about combinationality at the circuit level. Section 5 compares our formalism with other related work. Finally, Section 6 concludes this paper and lists some future research directions.

2. PRELIMINARIES

Unless otherwise noted, this paper assumes that variables and functions in consideration are of type Boolean, $\mathbb{B}$. Moreover, we concentrate on output-deterministic systems, whose output valuation is uniquely determined under any input assignment and under any current state designated by the systems' state-holding elements. As a notational convention, $[\![S]\!]$ denotes the set of all possible valuations of the set S of variables. Also, $|S|$ denotes the cardinality (or size) of S.

A functional-level description of a system $\mathcal{M}$ consists of a set $D_{\mathcal{M}}$ of **atomic definitions**. Each atomic definition is of the form $a_j := \phi_j$, where a_j and ϕ_j are a Boolean variable and formula, respectively. In particular, for a circuit-level description of a system, ϕ_j could be a formula of an identity function representing a wire (with delay), or a formula of an elementary Boolean function representing a primitive logic gate (with delay) in the gate library for technology mapping. At first glance, the distinctions between functional- and circuit-level descriptions are obscure; it seems to be a matter of granularity. However, we distinguish these two levels by saying that the valuations of atomic definitions take no time at the functional level, but take time at the circuit level. (We assume that every definition $a_j := \phi_j$ in $D_{\mathcal{M}}$ is deterministic, that is, variable a_j valuates to a definite value under any assignment on variables in formula ϕ_j. Thus, ϕ_j is a total Boolean function.) Associated to $D_{\mathcal{M}}$ is a **definition graph** $\Gamma^d_{\mathcal{M}} = (V^d, E^d)$ characterizing the information flow among atomic definitions. A vertex $v_j \in V^d$ represents a left-hand variable a_j of an atomic definition $a_j := \phi_j$ in $D_{\mathcal{M}}$. An directed edge $(v_j, v_k) \in E^d$ indicates that variable a_j appears in the right-hand formula ϕ_k of atomic definition $a_k := \phi_k$. The set $D_{\mathcal{M}}$ of atomic definitions is of **cyclic definition** if $\Gamma^d_{\mathcal{M}}$ is a cyclic graph.

Given a system $\mathcal{M}$ (without state-holding elements, i.e., registers or latches), three sets of variables are distinguished: the set I of primary-input variables, O of primary-output variables, and X of all the other (internal) variables. Notice that the primary-input variables are the definition-free variables, and vice versa. A subset $C \subseteq X \cup O$ is selected as a **cutset** such that the information flow among $D_{\mathcal{M}}$ becomes acyclic if C were exposed as primary-input variables in addition to the original ones. That is, the corresponding vertices of the cutset variables form a feedback vertex set in $\Gamma^d_{\mathcal{M}}$. It turns out that any such C out of $X \cup O$ provides enough information in analyzing the combinationality (its precise definition will be given later) of $\mathcal{M}$ at the functional level. Selecting a minimal[2] cutset helps simplify the analysis. (Previous studies, e.g. [7], on computing minimum feedback vertex sets [12] can be applied.) Furthermore, as will be proved, the analysis is independent of the choice of C as long as C is minimal. With a chosen cutset C, the behavior of $\mathcal{M}$ can be captured by two sets of definitions: the definitions of $c_j \in C$, i.e., $c_j := \xi_j$, and the definitions of $o_k \in O$, i.e., $o_k := \omega_k$. Here, I and C are the only variable occurrences in formulae ξ_j's and ω_k's. These formulae are obtained by a sequence of recursive substitutions of the definitions in $D_{\mathcal{M}}$ until the formulae for variables in $C \cup O$ have $I \cup C$ as the only variable occurrences. Thus, the original intermediate definitions of $\mathcal{M}$ are collapsed away. We call ξ_j the **excitation function** of $c_j \in C$, and ω_k the **observation function** of $o_k \in O$. (Cutset variables here are analogous to state variables of a state transition system, while excitation functions are analogous to transition functions.)

EXAMPLE 1. *Let $D_{\mathcal{M}} = \{a := \neg xa \vee c,\ b := \neg x(a \vee \neg b) \vee c,\ c := xb,\ y := \neg x(\neg a \vee ab) \vee x(a\neg c \vee \neg ac)\}$, $I = \{x\}$, and $O = \{y\}$. Suppose we choose C to be $\{a, b\}$. Then, rewriting $D_{\mathcal{M}}$ with respect to C, we have*

$$\begin{aligned} a &:= \neg xa \vee xb \\ b &:= \neg x(a \vee \neg b) \vee xb \\ y &:= \neg x(\neg a \vee ab) \vee x(a\neg b \vee \neg ab) \end{aligned}$$

The above right-hand formulae for a and b are the excitation functions; the formula for y is the observation function.

Combinationality analysis for cyclic definitions of systems with state-holding elements can be approximated as follows. Expose the outputs of state-holding elements as the primary inputs; expose the inputs of state-holding elements as the primary outputs. If the unreachable state set of the system is available, it can be used as the don't care condition in the combinationality analysis. Also, if the state equivalence relation is known, it can be used as a nondeterministic flexibility in the valuation of state-holding elements. Therefore, we mainly focus on systems without state-holding elements. The exact analysis for systems with state-holding elements is postponed to Section 3.6. Unless otherwise noted, we shall assume systems in consideration are without state-holding elements.

If a system consists of acyclic definitions, then it is combinational. However, the converse is not true: A combinational system may have *breakable cyclic definitions*. Hence, only systems with cyclic definitions are of our interest. Let $\mathcal{M}$ be such a system with cutset C. Given an input assignment for $\mathcal{M}$, then the valuation of the cutset variables evolves with time. The evolution can be captured by **state evolution graphs** (SEGs), analogous to state transition graphs for state transition systems. However, unlike a state transition graph, an SEG $\Gamma^e_{\mathcal{M},C,\tilde{\imath}} = (V^e_{\tilde{\imath}}, E^e_{\tilde{\imath}})$ exists with respect to a particular fixed input assignment $\tilde{\imath} \in [\![I]\!]$. A vertex $v_{\tilde{s}} \in V^e_{\tilde{\imath}}$ corresponds to an intermediate valuation (or, a state) $\tilde{s} \in [\![C]\!]$ of the cutset variables (in the sequel,

[2]A cutset C is *minimal* if removing any element from C makes the resultant information flow cyclic.

we shall not distinguish between a vertex and the state it represents); each directed edge $(v_{\tilde{s}_1}, v_{\tilde{s}_2}) \in E^e_{\tilde{\imath}}$ corresponds to an evolution of intermediate valuations from $\tilde{s}_1$ to $\tilde{s}_2$. That is, $\tilde{s}_2 = \vec{\xi}(\tilde{\imath}, \tilde{s}_1)$, where $\vec{\xi} : [\![I]\!] \times [\![C]\!] \to [\![C]\!]$ is the vector of excitation functions of C. Therefore, the evolution is deterministic at the functional level. (By contrast, the evolution may be nondeterministic due to races, hazards, glitches, etc., at the circuit level.) On the other hand, the vector $\vec{\omega} : [\![I]\!] \times [\![C]\!] \to [\![O]\!]$ of observation functions imposes a labelling over $[\![C]\!]$ with respect to some $\tilde{\imath} \in [\![I]\!]$.

Below we define and explore some basics about SEGs.

Definition 1. *A* **walk** *W of length k, denoted as $len(W) = k$, on an SEG $\Gamma^e_{\mathcal{M},C,\tilde{\imath}} = (V^e_{\tilde{\imath}}, E^e_{\tilde{\imath}})$ is a sequence $v_{\tilde{s}_0}, v_{\tilde{s}_1}, \ldots, v_{\tilde{s}_k}$ of vertices with $(v_{\tilde{s}_{j-1}}, v_{\tilde{s}_j}) \in E^e_{\tilde{\imath}}$. A* **path** *is a walk without repeated vertices. A* **loop** *of length k is a walk of length k without repeated edges and with $v_{\tilde{s}_0} = v_{\tilde{s}_k}$.*

In this paper, we use the term "loops" for SEGs and preserve "cycles" for definition graphs.

Proposition 1. *Any vertex of an SEG is in a loop and/or on a path leading to a loop.*

Proof. Since every state of an SEG has at least one outgoing edge, any vertex is in a loop and/or on a path leading to a loop. □

Proposition 2. *Any two loops of an SEG with deterministic evolution are disjoint.*

Proof. Since every vertex of an SEG with deterministic evolution has exactly one outgoing edge, any two loops of the SEG must be disjoint. □

In the sequel, we shall assume SEGs are deterministic unless otherwise stated.

Definition 2. *A loop L is* **stable** *if $len(L) = 1$; L is* **unstable** *if $len(L) > 1$.*

It will be clear later why a loop's stability is determined by its length.

Definition 3. *An* **equilibrium loop** *L of an SEG $\Gamma^e_{\mathcal{M},C,\tilde{\imath}}$ is a loop whose vertices all have the same observation label, i.e., $\forall v_{\tilde{s}_j}, v_{\tilde{s}_k} \in L.\ \vec{\omega}(\tilde{\imath}, \tilde{s}_j) = \vec{\omega}(\tilde{\imath}, \tilde{s}_k)$.*

Let $\mathcal{S}^\ell_{\mathcal{M},C,\tilde{\imath}}$ denote the set $\{\tilde{s} \in [\![C]\!] \mid v_{\tilde{s}}$ is a vertex in the loops of $\Gamma^e_{\mathcal{M},C,\tilde{\imath}}\}$. For $S \subseteq [\![C]\!]$, let $\mathcal{L}^o(S)$ denote the set $\{\vec{\omega}(\tilde{\imath}, \tilde{s}) \in [\![O]\!] \mid \tilde{s} \in S\}$ of observation labels.

Example 2. *Continue the set $D_{\mathcal{M}}$ of definitions and cutset C of Example 1. Figure 1 shows the state evolution graphs. Vertices are indexed with states, i.e., valuations of (a, b). States are distinguished by solid and dotted circles to reflect different observation labels induced by the observation function. The SEG of $x = 0$ has two loops, one stable and the other unstable; the SEG of $x = 1$ has two stable loops. All of the loops are equilibrium loops.*

3. COMBINATIONALITY AT FUNCTIONAL LEVEL

Given the functional-level description of a system $\mathcal{M}$ with cyclic definitions, we study the condition when $\mathcal{M}$ is combinational.

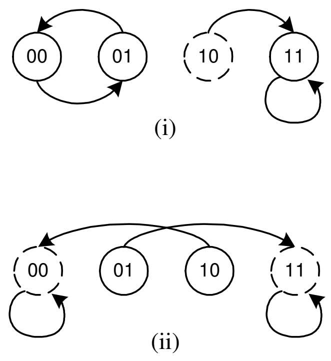

Figure 1: (i) SEG for $x = 0$. (ii) SEG for $x = 1$.

3.1 Formulation of Combinationality

In the functional-level formulation of combinationality, physical timing effects are abstracted away by assuming that all valuations of functions are instantaneous. However, the order of valuations matters.

A system $\mathcal{M}$ is said to be **combinational** at the functional level if, under any input assignment, $\mathcal{M}$ would *eventually* (i.e., within bounded steps) evolve into a status in which the observation labelling settles to a definite value[3] independent of the initial internal state in $[\![C]\!]$. Here, the dynamics of $\mathcal{M}$'s evolution is with respect to a cutset C. More precisely,

Theorem 1. *A system $\mathcal{M}$ with cutset C is combinational at the functional level if and only if, for any input assignment $\tilde{\imath} \in [\![I]\!]$, all states $\tilde{s} \in \mathcal{S}^\ell_{\mathcal{M},C,\tilde{\imath}}$ have the same observation label $\vec{\omega}(\tilde{\imath}, \tilde{s})$, i.e., $|\mathcal{L}^o(\mathcal{S}^\ell_{\mathcal{M},C,\tilde{\imath}})| = 1$.*

Proof. ($\longrightarrow$) Suppose not. $\mathcal{M}$ may produce outputs depending on the initial state in $[\![C]\!]$. In these cases, $\mathcal{M}$ is not combinational.

($\longleftarrow$) Since every vertex of $\Gamma^e_{\mathcal{M},C,\tilde{\imath}} = (V^e_{\tilde{\imath}}, E^e_{\tilde{\imath}})$ is either in a loop or on a path leading to a loop, any possible initial state $\tilde{s}_j$ evolves into a state $\tilde{s}_k \in \mathcal{S}^\ell_{\mathcal{M},C,\tilde{\imath}}$ after $|V^e_{\tilde{\imath}}| - 1$ steps of evolution. Since all $\tilde{s} \in \mathcal{S}^\ell_{\mathcal{M},C,\tilde{\imath}}$ have the same observation label, $\mathcal{M}$ eventually produces a unique output under input $\tilde{\imath}$. Because this is true for any input assignment, the proof follows. □

Example 3. *Continue Example 2. The system described by $D_{\mathcal{M}}$ is functionally combinational because, for any SEG, states in loops have the same observation label. Under input assignment $x = 0$, output y valuates to 1; under $x = 1$, y valuates to 0.*

3.2 Computation Algorithms

3.2.1 Combinationality Test.

From Theorem 1, we conduct a combinationality test on $\mathcal{M}$ using a symbolic computation (e.g. BDD-based computation) as follows. First, derive the set $\mathcal{S}^\ell_{\mathcal{M},C,\tilde{\imath}}$ for all $\tilde{\imath} \in [\![I]\!]$ by a greatest fixed-point computation. (It corresponds to:

[3]For simplicity, here we focus on the case where there is a unique output valuation in $[\![O]\!]$ for any input assignment. Our results can be straightforwardly generalized to a set of possible output valuations.

In the initial step, let $\mathcal{S}^{\ell}_{\mathcal{M},C,\tilde{\imath}} = [\![C]\!]$ initially, for any $\tilde{\imath} \in [\![I]\!]$. In the iterative steps, states without predecessors are successively removed from $\mathcal{S}^{\ell}_{\mathcal{M},C,\tilde{\imath}}$ by forward image computation with the characteristic function $\Xi = \bigwedge_j (c'_j \equiv \xi_j)$ of evolution relation over $\mathcal{S}^{\ell}_{\mathcal{M},C,\tilde{\imath}}$, where $\{c'_j \mid c_j \in C\}$ are newly introduced auxiliary variables, i.e., the "next-state" cutset variables. The process terminates when no more states can be removed from $\mathcal{S}^{\ell}_{\mathcal{M},C,\tilde{\imath}}$.) Notice that, with symbolic computation, $\mathcal{S}^{\ell}_{\mathcal{M},C,\tilde{\imath}}$ can be derived simultaneously for all $\tilde{\imath} \in [\![I]\!]$ since variables in I are not quantified out in the fixed-point computation. Second, we derive the characteristic function Λ of $\mathcal{L}^o(\mathcal{S}^{\ell}_{\mathcal{M},C,\tilde{\imath}})$ by setting $\Lambda = \exists c \in C.\Omega \wedge \Sigma$, where $\Omega : [\![I]\!] \times [\![C]\!] \times [\![O]\!] \to \mathbb{B}$ is the characteristic function $\Omega = \bigwedge_j (o_j \equiv \omega_j)$ of the output relation, and $\Sigma : [\![I]\!] \times [\![C]\!] \to \mathbb{B}$ is the characteristic function of $\mathcal{S}^{\ell}_{\mathcal{M},C,\tilde{\imath}}$ for all $\tilde{\imath} \in [\![I]\!]$. Again, this can be computed simultaneously for all $\tilde{\imath} \in [\![I]\!]$ since primary-input variables are not quantified out in the computation. Finally, we check if there exists an $\tilde{\imath}$ such that $|\mathcal{L}^o(\mathcal{S}^{\ell}_{\mathcal{M},C,\tilde{\imath}})| > 1$. If the answer is positive, then $\mathcal{M}$ is not combinational. Otherwise, it is. The computation can be done with a SAT-solving formulation, or with a BDD-based formulation. For the latter, the computation can be performed effectively using the *compatible projection operator* [14], **cprojection**. That is, $\mathcal{M}$ is combinational if and only if Λ equals **cprojection**$(\Lambda, \tilde{o})$, where $\tilde{o}$ is an arbitrary minterm in $[\![O]\!]$.

The computational complexity of the combinationality test is the same as that of state traversal on the space spanned by the cutset variables. That is, the complexity is PSPACE-complete in the size of the selected cutset.

THEOREM 2. *The problem of analyzing combinationality at the functional level is in the complexity class of PSPACE-complete with respect to the selected cutset size.*

PROOF. The problem of combinationality analysis can be done in nondeterministic PSPACE. To determine if a state $\tilde{s}$ is in a loop under some input assignment, one can record any consecutive two states in the state evolution trace starting from $\tilde{s}$. As the "window" slides along the trace, the recurrence of $\tilde{s}$ can be checked in at most $|[\![C]\!]|$ steps. In addition, one can test if different output observation labels ever appear in the sliding windows. Hence the combinationality analysis can be achieved within space bounded by a polynomial in the cutset size.

On the other hand, we need to reduce a PSPACE-complete problem to the problem of combinationality analysis. The following problem can be used.

> Given a total function $f : \{1, \ldots, n\} \to \{1, \ldots, n\}$, is there a k such that $f^k(1) = n$?

It was shown [11] to be deterministic[4] LOGSPACE-complete in n and, thus, PSPACE-complete in $\log n$. We establish that the answer to the PSPACE-complete problem is positive if and only if the answer to the corresponding problem of combinationality analysis (to be constructed) is negative. Since the complexity class of nondeterministic space is closed under complementation [10], the theorem follows.

To complete the proof, given $f : \{1, \ldots, n\} \to \{1, \ldots, n\}$, an excitation function $\xi : \{1, \ldots, n\} \to \{1, \ldots, n\}$ and observation function $\omega : \{1, \ldots, n\} \to \mathbb{B}$ are constructed as follows. Let ξ have the same mapping as f but with $\xi(n) = 1$. Also, let $\omega(j) =$ FALSE for $1 \le j \le n-1$, and $\omega(n) =$ TRUE. With the above construction, n is reachable from 1 under f if and only if the system defined by ξ and ω is not combinational. (Note that, since an n-valued variable can be encoded with $O(\log n)$ binary variables, multiple-valued representations fit our framework.) $\square$

[4]It is a well-known fact, proved by Savitch in [19], that deterministic and nondeterministic space complexities coincide.

3.2.2 Cycle Breaking.

Suppose $\mathcal{M}$ is combinational. From the above combinationality test, we can derive a set of equivalent acyclic definitions for $\mathcal{M}$. In fact, there are two ways of doing so: One is to rewrite definitions of primary-output variables as functions of primary-input variables. The other is to rewrite definitions of cutset variables as functions of primary-input variables. An advantage of the latter would be that the original definitions of $\mathcal{M}$ can be reused except for the definitions $c_j := \phi_j$, for $c_j \in C$, and the resultant unused ones. The derivations of the new definitions are as follows.

For the former rewriting, the new definitions for the primary-output variables can be inferred from the input-output relation $\exists c \in C.\Omega \wedge \Sigma$, which has been computed in the combinationality test. For the later rewriting, for every $\tilde{\imath} \in [\![I]\!]$, some state $\tilde{s}_{\tilde{\imath}} \in \mathcal{S}^{\ell}_{\mathcal{M},C,\tilde{\imath}}$ is selected as the representative for $\mathcal{S}^{\ell}_{\mathcal{M},C,\tilde{\imath}}$. Then the new definitions for the cutset variables can be inferred from the relation $\bigwedge_{\tilde{\imath}} (\wedge_j (i_j \equiv \tilde{\imath}[j])) \wedge (\wedge_k (c_k \equiv \tilde{s}_{\tilde{\imath}}[k]))$, where $i_j \in I$, $c_k \in C$, and $\tilde{\imath}[j]$ (resp. $\tilde{s}_{\tilde{\imath}}[k]$), which denotes the jth (resp. kth) bit of $\tilde{\imath}$ (resp. $\tilde{s}_{\tilde{\imath}}$), is a Boolean constant of value either TRUE or FALSE.

3.3 Generality Analysis

THEOREM 3. *Let C_1 and C_2 be two choices of minimal cutsets for a system $\mathcal{M}$ with cyclic definitions. Then, under any input assignment $\tilde{\imath} \in [\![I]\!]$, there exists a bijection between the loops of $\Gamma^e_{\mathcal{M},C_1,\tilde{\imath}}$ and those of $\Gamma^e_{\mathcal{M},C_2,\tilde{\imath}}$.*

PROOF. First observe that, since both C_1 and C_2 are cutsets, under a specific input assignment $\tilde{\imath} \in [\![I]\!]$, the variables in C_1 can be expressed as functions of variables in C_2, and vice versa. Thus, there exist a mapping $f_{21} : [\![C_1]\!] \to [\![C_2]\!]$ (resp. $f_{12} : [\![C_2]\!] \to [\![C_1]\!]$) such that, for a valuation $\tilde{s}$ of C_1 (resp. $\tilde{t}$ of C_2), $f_{21}(\tilde{s})$ (resp. $f_{12}(\tilde{t})$) is the corresponding valuation of C_2 (resp. C_1) variables. In addition, since C_1 and C_2 are minimal, we have $f_{12}(f_{21}(\tilde{s})) = \vec{\xi}_1(\tilde{\imath}, \tilde{s})$ and $f_{21}(f_{12}(\tilde{t})) = \vec{\xi}_2(\tilde{\imath}, \tilde{t})$, where $\vec{\xi}_1$ and $\vec{\xi}_2$ are the vectors of excitation functions of $\mathcal{M}$ with cutsets C_1 and C_2, respectively.

To see the relation between the loops of $\Gamma^e_{\mathcal{M},C_1,\tilde{\imath}}$ and those of $\Gamma^e_{\mathcal{M},C_2,\tilde{\imath}}$, consider a state evolution sequence $\sigma_1 = \tilde{s}_1, \ldots, \tilde{s}_j, \ldots, \tilde{s}_k$ of $[\![C_1]\!]$ such that $\tilde{s}_k$ is the first recurrent state in σ_1 with $\tilde{s}_k = \tilde{s}_j$. Clearly, $\sigma_2 = f_{21}(\tilde{s}_1), \ldots, f_{21}(\tilde{s}_j), \ldots, f_{21}(\tilde{s}_k)$ is a state evolution sequence over $[\![C_2]\!]$ because $f_{21}(f_{12}(f_{21}(\tilde{s}))) = \vec{\xi}_2(\tilde{\imath}, f_{21}(\tilde{s}))$. Now, we need to show that $f_{21}(\tilde{s}_k)$ is the only recurrent state in σ_2 with $f_{21}(\tilde{s}_k) = f_{21}(\tilde{s}_j)$. By contradiction, suppose there exists another recurrent state in σ_2 such that $f_{21}(\tilde{s}_m) = f_{21}(\tilde{s}_l)$, $l < m < k$. However, this implies $\tilde{s}_{m+1} = \tilde{s}_{l+1}$ in σ_1 because $f_{12}(f_{21}(\tilde{s}_m)) = f_{12}(f_{21}(\tilde{s}_l))$. It contradicts with the assumption that $\tilde{s}_k$ is the first recurrent state in σ_1 unless $m = k-1$ and $l = j-1$. Similarly, one can show that, given a state evolution sequence of $[\![C_2]\!]$ with a loop, there exists a corresponding sequence of $[\![C_1]\!]$ with a loop. Also, by Propositions 1 and 2, there exists a bijection between the loops of $\Gamma^e_{\mathcal{M},C_1,\tilde{\imath}}$ and those of $\Gamma^e_{\mathcal{M},C_2,\tilde{\imath}}$. $\square$

COROLLARY 1. *A system's combinationality at the functional level is independent of the choice of minimal cutset in the analysis.*

PROOF. Let C_1 and C_2 be two choices of minimal cutsets, and $\vec{\omega}_1 : [\![I]\!] \times [\![C_1]\!] \to [\![O]\!]$ and $\vec{\omega}_2 : [\![I]\!] \times [\![C_2]\!] \to [\![O]\!]$ be the resultant vectors of observation functions. Let f_{21} and f_{12} be the mappings as defined in the proof of Theorem 3. Then, $\vec{\omega}_1(\tilde{\imath}, \tilde{s}_1) = \vec{\omega}_2(\tilde{\imath}, f_{21}(\tilde{s}_1))$ and, similarly, $\vec{\omega}_1(\tilde{\imath}, f_{12}(\tilde{s}_2)) = \vec{\omega}_2(\tilde{\imath}, \tilde{s}_2)$, for any $\tilde{\imath} \in [\![I]\!]$, $\tilde{s}_1 \in [\![C_1]\!]$, and $\tilde{s}_2 \in [\![C_2]\!]$. In addition to the result of Theorem 3, we need to show that all corresponding loops of $\Gamma^e_{\mathcal{M},C_1,\tilde{\imath}}$ and $\Gamma^e_{\mathcal{M},C_2,\tilde{\imath}}$ must have the same output observation for all $\tilde{\imath} \in [\![I]\!]$.

Suppose $\mathcal{M}$ is combinational under an analysis with cutset C_1. Then, all the states in $\mathcal{S}^\ell_{\mathcal{M},C_1,\tilde{\imath}}$ must have the same output observation label, say $\tilde{o}_1 \in [\![O]\!]$. For the sake of contradiction, assume there exists a state $\tilde{s}_2 \in \mathcal{S}^\ell_{\mathcal{M},C_2,\tilde{\imath}}$ with $\vec{\omega}_2(\tilde{\imath}, \tilde{s}_2) \neq \tilde{o}_1$. It implies that $\vec{\omega}_1(\tilde{\imath}, f_{12}(\tilde{s}_2)) \neq \tilde{o}_1$. Since $f_{12}(\tilde{s}_2)$ is in $\mathcal{S}^\ell_{\mathcal{M},C_1,\tilde{\imath}}$, it contradicts with the assumption that all the states in $\mathcal{S}^\ell_{\mathcal{M},C_1,\tilde{\imath}}$ have observation label $\tilde{o}_1$. Hence, all the states in $\mathcal{S}^\ell_{\mathcal{M},C_2,\tilde{\imath}}$ must have the same observation label $\tilde{o}_1$ as well. The corollary follows. □

Notice that the result holds even when the cutset changes dynamically.

Assuming a system $\mathcal{M}$ operates without a special pre-initialization, our combinationality analysis at the functional level is the most general formulation that one can hope for in the sense that

THEOREM 4. *There exists a feasible combinational implementation of $\mathcal{M}$ if and only if $\mathcal{M}$ satisfies our combinationality test.*

PROOF. ($\longrightarrow$) Suppose that $\mathcal{M}$ fails our combinationality test. It implies that there exists some input assignment such that the corresponding output valuation cannot settle to an unique value. This violates the definition of combinationality.

($\longleftarrow$) Trivial. □

3.4 Conditions of Legitimacy

The legitimacy of our combinationality formulation is confirmed if a system's cyclic definitions are to be broken in the final realization. However, if some cyclic definitions are to be maintained in the final realization, then the certification of combinationality at the functional level of abstraction is not sufficient to guarantee correctness. Essentially, two restrictions need to be imposed to ensure the correctness. First, all excitation functions should be valuated synchronously such that state evolutions follow the combinationality analysis. Second, the time interval between two consecutive input assignments should be much larger than the time spent on internal valuations such that the state of the system has enough time to evolve to an equilibrium loop. Certainly, the first restriction is inadequate for hardware realization of cyclic definitions due to undesirable defects, such as races, hazards, glitches, etc. In contrast, software realization is more adequate since the above defects can be eliminated. A possible application domain could be software synthesis for reactive systems, where the common assumption is that internal computations are much faster than environmental responses. Hence, the second restriction is satisfied under this assumption.

3.5 Stable Cyclic Dependencies

For software synthesis with cyclic definitions to be maintained, although the combinationality formulation at the functional level can be legitimate, it may be undesirable for SEGs containing unstable loops. Since the existence of unstable loops results in persistent updates of state information (even though observation functions have settled to definite values), the updates consume dynamic power. To avoid the persistent power consumption, we require that all the loops in SEGs must be stable. To do so, the definitions of the system $\mathcal{M}$ in consideration should be rewritten. Such rewrites can be done in various ways. For instance, an unstable loop L is broken by redirecting the evolution of a state in L to itself or to another state not in an unstable loop. Recall that replacing cyclic definitions with acyclic equivalents is just a special case of such rewrites.

On the other hand, one can devise an algorithm to test if a system $\mathcal{M}$ with cutset C is stably combinational at the functional level. Essentially, $\mathcal{M}$ is stably combinational if and only if, for any input assignment, any state $\tilde{s} \in [\![C]\!]$ can reach (i.e., evolve to) a self-looped state. Let $\Sigma : [\![I]\!] \times [\![C]\!] \to \mathbb{B}$ be the characteristic function denoting the set of states which can reach self-looped states with respect to some input assignment. Then the algorithm can be outlined as follows. First, compute the set of self-looped states of $\Gamma^e_{\mathcal{M},C,\tilde{\imath}}$ for all $\tilde{\imath} \in [\![I]\!]$ by the characteristic function $\bigwedge_j (c_j \equiv \xi_j)$, where $c_j \in C$ is a cutset variable and $\xi_j : [\![I]\!] \times [\![C]\!] \to \mathbb{B}$ is an excitation function in $\vec{\xi}$. Second, let $\Sigma = \bigwedge_j (c_j \equiv \xi_j)$ initially. Perform the standard backward reachability analysis (however, variables in I are not quantified out). That is, with respect to some $\tilde{\imath} \in [\![I]\!]$, the set $S_{\Sigma_{\tilde{\imath}}}$ of states represented by $\Sigma_{\tilde{\imath}}$ is augmented iteratively by adding to it the set of its predecessor states, where $\Sigma_{\tilde{\imath}}$ denotes the partial valuation of Σ with variables I valuate to $\tilde{\imath}$. The iteration terminates when no more states can be added. Using a symbolic approach, the computation is done for all input assignments simultaneously since variables in I are not quantified out during the fixed-point computation. The system is stably combinational if and only if the final Σ is a tautology.

In addition to the stability requirement, one may want to bound the maximum length of evolution paths to equilibrium loops. The number of iterations spent in a combinationality test corresponds to the length of the longest evolution path(s). If the length is greater than the upper bound, say n, state evolutions need to be redirected to shorten long paths. One approach would be to memorize the newly removed state sets for every $n-1$ iterations in the combinationality test. After the test, redirect the evolutions of the memorized states to proper equilibrium loops.

3.6 Input-Output Determinism of State Transition Systems

We extend combinationality analysis on the set $D_{\mathcal{M}}$ of cyclic definitions of a system $\mathcal{M}$ with state-holding elements. Note that $D_{\mathcal{M}}$ contains only simultaneous definitions and, thus, excludes the delayed definitions of the state-holding elements. Let I and O be the sets of primary input and primary output variables, respectively. Also, let S (resp. S') be the set of output (resp. input) variables of the state-holding elements, and C be a cutset of $D_{\mathcal{M}}$. We specify two types of states: *External states* $[\![S]\!]$ are those designated by the state-holding elements; *internal states* $[\![C]\!]$ are those emerging from the cyclic definitions. Also, terms "transition" and "evolution" are used to differentiate the dynamics among external and internal states, respectively.

Our objective here is to analyze whether the cyclic definitions of $\mathcal{M}$ can be replaced with acyclic ones such that the sequential behavior of $\mathcal{M}$ remains unchanged. Essentially, such a substitution is possible if and only if $\mathcal{M}$ has deterministic input-output behavior[5]. As mentioned in Section 2, the inputs and outputs of the state-holding elements can be treated as the primary outputs and primary inputs, respectively, of the set of the cyclic definitions. However, a direct combinationality test on the cyclic definitions only yields an approximative analysis because it requires the valuations of S' to be deterministic.

To achieve an exact analysis, with the above input and output transformation, reachability analysis (for external states) and combinationality analysis (for internal states) should be performed alternately. Two conditions need to be satisfied: First, under any input assignment $\tilde{\imath} \in [\![I]\!]$ and any reachable state $\tilde{s} \in [\![S]\!]$, the set $\mathcal{S}^{\ell}_{\mathcal{M},C,(\tilde{\imath},\tilde{s})}$ of all internal states in loops of the corresponding SEG $\Gamma^{e}_{\mathcal{M},C,(\tilde{\imath},\tilde{s})}$ must have the same observation label induced by O, i.e., $|\mathcal{L}^{o}(\mathcal{S}^{\ell}_{\mathcal{M},C,(\tilde{\imath},\tilde{s})})| = 1$. Second, under any input assignment $\tilde{\imath}$ and any reachable state $\tilde{s}$, the corresponding next (external) states must be sequentially equivalent. This set of next states is derived from the set of observation labels induced by S' over $\mathcal{S}^{\ell}_{\mathcal{M},C,(\tilde{\imath},\tilde{s})}$.

A detailed computation is outlined as follows. Let $R^{(j)}$ be the reached state set for the state-holding elements at the jth iteration. Let $R^{(0)} \subseteq [\![S]\!]$ be the initial state set. In the jth iteration, we perform combinationality analysis detailed in Section 3.2.1 to certify that $D_{\mathcal{M}}$ is combinational with respect to O for any $\tilde{\imath} \in [\![I]\!]$ and $\tilde{s} \in R^{(j)}$. (If the certification is not established, $\mathcal{M}$ is not deterministic in its input-output behavior and the procedure aborts.) The combinationality analysis also gives us the set $\mathcal{S}^{\ell}_{\mathcal{M},C,(\tilde{\imath},\tilde{s})}$ for all $\tilde{\imath} \in [\![I]\!]$ and $\tilde{s} \in R^{(j)}$. From it, we obtain the set of next states under $\tilde{\imath}$ and $\tilde{s}$ by computing the set of observation labels induced by S' over $\mathcal{S}^{\ell}_{\mathcal{M},C,(\tilde{\imath},\tilde{s})}$. Denote the set of next states as $N^{(j)}_{\tilde{\imath},\tilde{s}}$. Then, $R^{(j+1)} = R^{(j)} \cup \{N^{(j)}_{\tilde{\imath},\tilde{s}} \mid \tilde{\imath} \in [\![I]\!], \tilde{s} \in R^{(j)}\}$. The iterations terminate when $R^{(k+1)} = R^{(k)}$ for some k. At this point, we need one more step to conclude whether $D_{\mathcal{M}}$ can be rewritten with acyclic definitions. The answer is affirmative if and only if $|\mathcal{L}^{o}(N^{(j)}_{\tilde{\imath},\tilde{s}})| = 1$, for $j = 0, \ldots, k-1$, and for any $\tilde{\imath} \in [\![I]\!]$, $\tilde{s} \in R^{(j)}$. The rewriting procedure is similar to what was described in Section 3.2.2.

The corresponding computational complexity is PSPACE-complete in the number of state-holding elements and the cutset size. The PSPACE-completeness is immediate from the fact that the input-output determinism problem of state transition systems is in PSPACE and is even harder than the PSPACE-complete problem of combinationality analysis shown in Theorem 2.

[5]Nevertheless, state transitions may be nondeterministic due to the cyclic definitions. Since internal states in loops of an SEG may have different observation labels induced by S', these observation labels constitute the possible next external states. Hence, state transitions are nondeterministic in general.

4. COMBINATIONALITY AT CIRCUIT LEVEL

Combinationality analysis at the functional level abstracts away timing information. Certainly, it does not guarantee the feasibility of maintaining cyclic definitions in final circuit implementations. On the other hand, Malik's formulation based on ternary-valued simulation turns out to be the right formulation at the circuit level. In his formation, effectively, all gates and wires are sources of uncertain delay[6]. Under the *up-bounded inertial delay model* [4], ternary-valued simulation can be treated as an operational definition of combinationality for cyclic circuits [21, 20].

4.1 Synthesis of Cyclic Circuits

A recent attempt [17, 18] of synthesizing cyclic circuits brings the formulation of ternary-valued simulation up to the functional level. Combinationality analysis was checked with recursive marginal operations. However, it was overlooked that functional-level analysis itself is not sufficient to guarantee the correctness of final circuit implementation. Consider the following cyclic definitions over primary-input variables a and b:

$$\begin{aligned} f &:= \neg a h \vee \neg b \neg h \\ g &:= \neg a \neg b f \\ h &:= ab \vee \neg g \end{aligned}$$

The reader can verify that the above definitions are *functionally* combinational under any input assignments. (Indeed, under the analysis with recursive marginal operations, the cyclic definitions are combinational.) However, functional analysis does not guarantee a well-behaved circuit implementation. Consider the circuit netlist in Figure 2 (i) as an implementation of the above cyclic definitions. The circuit may not be well-behaved. To see this, consider input assignment $a = 0$ and $b = 0$. Assume the resultant induced circuit is abstracted to that in Figure 2 (ii), where all the gates have one-unit delay and all the wires have zero delay. Now, suppose the previous input assignment is $a = 1$ and $b = 1$ before assignment $a = 0$ and $b = 0$. That is, internal signals x and y in Figure 2 (ii) are of value 0 initially. An examination shows that the circuit oscillates despite of its combinationality at the functional level. Essentially, the failure originates from the fact that some gates and wires are not fully characterized in the analysis. Hence, functional-level analysis is not sufficient to conclude the correctness of the gate-level implementation.

Two approaches can be applied to rectify the deficiency in the analysis proposed by [17, 18]. One is to remove axioms $x \vee \neg x = \text{TRUE}$ and $x \wedge \neg x = \text{FALSE}$ from the recursive marginal operations when x is not a primary-input variable. The other is to add more terms to functional expressions such that, for every input assignment, cyclic definitions are broken for some functions valuating to either TRUE or FALSE purely depending on the input assignment. For instance, in the previous example, product term $\neg a \neg b$ needs to be added to the definition of f, i.e, $f := \neg a h \vee \neg b \neg h \vee \neg a \neg b$. For the second rectification, one should be careful in any subsequent circuit optimization; the added terms should not be removed without special care. Note that the necessity of

[6]However, this timing assumption is very conservative in the sense that no asynchronous circuits can ever exist under this assumption if initialization is not allowed.

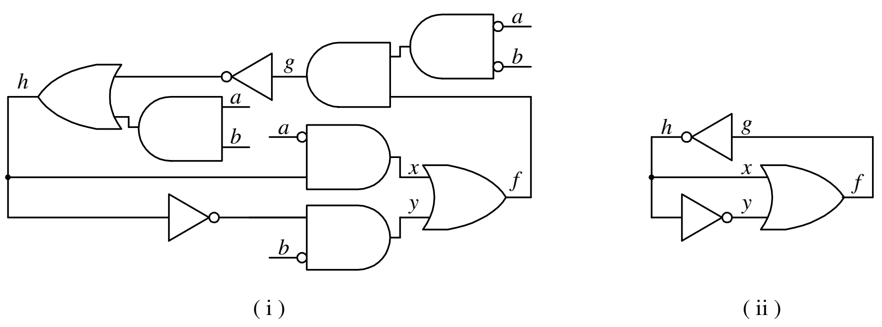

Figure 2: (i) The original circuit. (ii) The induced circuit under input assignment $a = 0$ and $b = 0$.

adding some rectification terms may nullify area gains due to allowing cyclic combinational circuits.

5. RELATED WORK

5.1 SEG vs. GMW

Our SEG formalism is closely related to the *general multiple winner* (GMW) analysis [4], which is commonly used in the analysis of asynchronous circuits. To reason about the behavior of asynchronous circuits under physical effects such as glitches, hazards, races, etc., the GMW analysis builds graphs similar to SEGs with additional nondeterministic evolutions. Depending on how the current state and next state are coded, an evolution branches out into several nondeterministic ones. Also, unlike an SEG existing for a fixed input assignment, the graph built by the GMW analysis is connected for different input assignments. These additional evolutions make GMW analysis a complicated procedure. Even worse, the GMW analysis declares a state variable for every delay element (possibly, a gate or wire). In comparison, only a minimal cutset needs to be chosen in our combinationality analysis. Therefore, the state space is substantially reduced. Under the legitimacy conditions in Section 3.4, all of the above complications in the GMW analysis can be avoided and simplified to our SEG formalism.

5.2 Combinationalities

At the functional level, we contrast our formulation of combinationality with prior work based on ternary-valued simulation. In the case where all valuations in the set of cyclic definitions must stabilize, the functional-level extension of Malik's formulation can be summarized as follows. For any input assignment $\tilde{\imath} \in [\![I]\!]$, there exists a set of definitions valuating to either TRUE or FALSE such that the cyclic definitions are broken. This requirement corresponds to that, for every input assignment, the corresponding SEG has a single loop, which is stable, i.e., of length one, and all states of the SEG evolve directly to this loop. (An SEG with multiple stable loops correspond to what was considered having nondeterministic multiple solutions; an SEG with an unstable loop, i.e., a loop with length greater than one, corresponds to what was considered having no consistent solution.) In comparison, our formulation is much more general because SEGs are allowed to have multiple loops, which can be stable or unstable, and to have long evolution paths.

Now consider a more relaxed case where signals are allowed to oscillate for some input assignment as long as all output valuations are uniquely determined under this input assignment regardless of internal states. To see how Malik's formulation corresponds, we partition input assignments into two sets: one with outputs fully determined, and the other partially determined. Under the former set of input assignments, no restrictions need to be imposed on SEGs, just like in our formulation. Under the latter set of input assignments, however, the restrictions discussed in the case where all valuations must stabilize need to be imposed. Although the generality is enhanced in the relaxed case, the combinationality based on ternary-valued simulation is still a limited formulation. In comparison, our formulation is strictly more general. In fact, it is the most general formulation that one can hope for as stated in Theorem 4.

EXAMPLE 4. *Continue Example 1. The observation function is only partially determined under any input assignment. It is not hard to see that system $\mathcal{M}$ specified by $D_{\mathcal{M}}$ is not combinational under the functional-level extension of Malik's combinationality formulation, contrary to our combinationality analysis.*

5.3 Sequential Extensions

In his thesis [20], Shiple extended the analysis of combinational cycles for circuits with state-holding elements. He defined sequential output-stability, which allows a circuit to be initialized to some stable states and considers only initialized behavior. The GMW analysis was adopted to replace ternary-valued simulation such that nondeterministic internal states were admitted to exist as long as the observable behavior is unaffected. An equivalent acyclic circuit can be generated from the GMW analysis. Again, if the objective is software synthesis or to break cyclic definitions, such sequential extension can be generalized substantially and simplified to our computation outlined in Section 3.6 without resorting to the complicated GMW analysis.

6. CONCLUSIONS AND FUTURE WORK

Based on the observation that when cyclic definitions are to be broken in the final realization, the formulation of combinationality can be much more general than previous formulations. In addition, the analysis can be done at a higher abstraction level, i.e., the functional level. The combinationality formulation is extended to an extreme — a system is combinationally implementable if and only if it passes our combinationality test. When cyclic definitions are to be maintained, we examine the legitimacy condition of our formulation. It turns out that software synthesis of reactive systems may be an application domain, where cyclic definitions can be maintained in the final realization. In addition, we show that our analysis is independent of the choice of cutsets. Our results admit strictly more flexible high-level specifications in hardware/software system design. For combinationality at the circuit level, we comment on a pitfall in a recent attempt synthesizing cyclic circuits for area minimization. Two approaches are given to rectify the deficiency.

As for future work, although the choice of cutset does not affect the analysis of combinationality, it does influence the resultant system rewritten with acyclic definitions. It might be crucial to decide a good cutset with respect to various optimization objectives. Also, as shown in Sections 3.2, 3.5 and 3.6, there are many ways to rewrite cyclic definitions with acyclic equivalents. It would be interesting to explore such flexibilities for further optimization.

ACKNOWLEDGEMENTS

The authors would like to acknowledge the helpful comments of anonymous reviewers and clarifying discussions with Marc Riedel.

REFERENCES

[1] G. Berry. The foundations of Esterel. *Proof, Language, and Interaction: Essays in Honour of Robin Milner*, MIT Press, 2000.

[2] G. Berry. *The constructive semantics of pure Esterel.* Draft book, 1999.

[3] R. Bryant. Boolean analysis of MOS circuits. *IEEE Trans. Computer-Aided Design*, pages 634–649, July 1987.

[4] J. Brzozowski and C.-J. Seger. *Asynchronous Circuits.* Springer-Verlag, 1995.

[5] S. Edwards. Making cyclic circuits acyclic. In *Proc. Design Automation Conference*, pages 159–162, 2003.

[6] S. Edwards and E. Lee. The semantics and execution of a synchronous block-diagram language. *Science of Computer Programming*, vol. 48, pages 21-42, 2003.

[7] G. Even, J. Naor, B. Schieber, and M. Sudan. Approximating minimum feedback sets and multi-cuts in directed graphs. *Algorithmica*, vol. 20, pages 151–174, 1998.

[8] N. Halbwachs. *Synchronous Programming of Reactive Systems.* Kluwer Academic Publishers, 1993.

[9] N. Halbwachs and F. Maraninchi. On the symbolic analysis of combinational loops in circuits and synchronous programs. In *Proc. Euromicro*, 1995.

[10] N. Immerman. Nondeterministic space is closed under complementation. *SIAM Journal on Computing.* vol. 17, pages 935–938, 1988.

[11] N. Jones. Space-bounded reducibility among combinatorial problems. *Journal of Computer and System Sciences*, vol. 11, pages 68–85, 1975.

[12] R. Karp. Reducibility among combinatorial problems. *Complexity of Computer Computations*, pages 85–104, Plenum Press, 1972.

[13] W. Kautz. The necessity of closed circuit loops in minimal combinational circuits. *IEEE Trans. on Computers*, pages 162–164, 1970.

[14] B. Lin and A. R. Newton. Implicit manipulation of equivalence classes using binary decision diagrams. In *Proc. Int'l Conf. Computer Design*, pages 81–85, 1991.

[15] S. Malik. Analysis of cyclic combinational circuits. *IEEE Trans. on Computer-Aided Design*, vol. 13, no. 7, pages 950–956, July 1994.

[16] K. Namjoshi and R. Kurshan. Efficient analysis of cyclic definitions. In *Proc. Computer Aided Verification*, pages 394–405, 1999.

[17] M. Riedel and J. Bruck. The synthesis of cyclic combinational circuits. In *Proc. Design Automation Conference*, pages 163–168, 2003.

[18] M. Riedel and J. Bruck. Cyclic combinational circuits: analysis for synthesis. In *Proc. Int'l Workshop on Logic and Synthesis*, pages 105–112, 2003.

[19] W. Savitch. Relationships between nondeterministic and deterministic tape complexities. *Journal of Computer and System Sciences*, vol. 4, pages 177–192, 1970.

[20] T. Shiple. *Formal Analysis of Cyclic Circuits.* Ph.D. thesis, University of California at Berkeley, 1996.

[21] T. Shiple, G. Berry, and H. Touati. Constructive analysis of cyclic circuits. In *Proc. European Design and Test Conf.*, pages 328–333, 1996.

[22] L. Stok. False loops through resource sharing. In *Proc. Int'l Conf. on Computer-Aided Design*, pages 345–348, 1992.