Delay-Insensitivity and Semi-Modularity*

J. A. BRZOZOWSKI H. ZHANG

Department of Computer Science
University of Waterloo
Waterloo, Ontario, Canada N2L 3G1
email: {brzozo, hzhang}@maveric.uwaterloo.ca
ftp://cs-archive.uwaterloo.ca/cs-archive/CS-97-11/CS-97-11.ps.Z

March 14, 1997

Abstract

A network is delay-dense if, of each pair of adjacent components in the network, at least one is a delay. In this paper, we prove that a delay-dense asynchronous network is delay-insensitive if and only if its behavior is semi-modular. We consider *autonomous* networks, i.e., networks without external inputs, whose components can be any sequential machines of the Moore type. A formal model for this type of networks and their behaviors is established. We define delay-insensitivity of networks using the notion of bisimulation. The definition of semi-modularity is generalized to reflect non-determinism in network behaviors.

Keywords: asynchronous, bisimulation, delay-dense, delay-insensitive, delay extension, isochronic, module, network, semi-modular, speed-independent

1 Introduction

Although much of today's digital design is based on a synchronous approach, there has been a considerable interest in asynchronous circuits [2], more so during the past decade. In contrast to a synchronous circuit, whose operation is under the control of a global clock signal, an asynchronous circuit uses local handshaking between its components. Potential advantages in using asynchronous circuits include lower energy consumption, higher speed, and avoidance of clock distribution problems [2].

*This research was supported by the Natural Sciences and Engineering Research Council of Canada under grant No. OGP0000871.

Among asynchronous designs, the class of so-called delay-insensitive networks is receiving special attention. Roughly speaking, a network is delay-insensitive if it continues to operate correctly, even if the delays in its components and wires change arbitrarily. When such networks are designed in a modular fashion, it is possible to replace their components by faster or slower ones, without changing the correctness of the network operation, although, performance may be affected.

C. E. Molnar et al. introduced the "foam-rubber wrapper" postulate [6] to describe delay-insensitive specifications of network modules. A module is viewed as being surrounded by a "foam-rubber wrapper." The inner surface of the wrapper corresponds to the module interface, whereas the outer surface defines the environment interface. The foam-rubber analogy suggests that the distance between the inner and outer surfaces along one wire may be different from that along a different wire, representing different (and possibly time-varying) delays. Following this informal idea, J. T. Udding [9] gave the first formal definition of delay-insensitivity of network modules, based on trace theory.

Semi-modularity of circuit behaviors was studied by Muller in his theory of "speed-independent" circuits [4]. Roughly speaking, semi-modularity requires that no state transition can change the implied value of any unchanged state variable which is unstable. Muller considered autonomous circuits. He assumed that components could have arbitrary delays, but that wire delays were negligible. Intuitively, a circuit is speed-independent if it operates correctly independently of the speeds of its components. Muller's formal definition of speed-independence concerns the terminal classes (or outcome [2]) of so-called allowed state sequences for the circuit. Muller showed that circuits exhibiting semi-modular behaviors form a *proper* subclass of speed-independent circuits. For a recent survey of various definitions of delay-insensitivity, speed-independence, and related concepts, see [1].

In this paper, we consider autonomous networks whose components can be any sequential machines of the Moore type [7]. A formal model for this type of networks and their behaviors is established. An asynchronous circuit is characterized by a so-called delay-dense network. A network is delay-dense if, of each pair of adjacent network components, at least one is a delay. We define delay-insensitivity of networks using the notion of bisimulation [5]. A network N is delay-insensitive if the behavior of any network $\hat{N}$ derived from N by the addition of delays is bisimilar to the behavior of N. A similar approach has been used by N. Shintel and M. Yoeli [8]. We generalize the definition of semi-modularity to reflect nondeterminism in network behaviors, so that we can model arbitration devices. We prove that a delay-dense network is delay-insensitive if and only if its behavior is semi-modular.

The paper is structured as follows. In Sections 2 and 3, we present our model for modules, networks, and network behaviors. Sections 4 and 5 concentrate on the definitions of delay-insensitivity of networks and semi-modularity of network behaviors. In Section 6, we give the proof of our main result. Section 7 concludes the paper; in particular, we relate our result to Muller's theory of speed-independent circuits.

2 Modules

We now introduce our model of asynchronous component, which we call "module." To hide the details of the internal design of a module, we represent it only by an abstract internal state. This also helps to keep the model simple. We do not introduce any delays in the input and output wires of the module for several reasons. First, wire delays are frequently ignored in modules designed on a small area of a chip ("equipotential region") [3]. Second, we want to have the ability to model isochronic forks and similar components, since they are used in many practical designs. And finally, if we do wish to have modules with input and output delays, we can model them within a network, which is defined in the next section.

Definition 2.1 A *module* is a pair $M^i = \langle G^i, C^i \rangle$, where

- $G^i = \langle \mathcal{V}^i, \mathcal{E}^i \rangle$ is a directed graph, the *module graph*, where
  - $\mathcal{V}^i = \mathcal{X}^i \cup \{y^i\} \cup \mathcal{Z}^i$ is the set of *module vertices*, where $\mathcal{X}^i$, $\{y^i\}$, and $\mathcal{Z}^i$ are pairwise disjoint sets, and
    - $\mathcal{X}^i = \{x^i_1, \ldots, x^i_{m^i}\}$, $m^i \geq 1$, is the set of *module input vertices*;
    - y^i is the *module internal state vertex*;
    - $\mathcal{Z}^i = \{z^i_1, \ldots, z^i_{p^i}\}$, $p^i \geq 1$, is the set of *module output vertices*;
  - $\mathcal{E}^i = (\mathcal{X}^i \times \{y^i\}) \cup (\{y^i\} \times \mathcal{Z}^i)$ is the set of *module edges*;
- $C^i = \langle \mathcal{S}^i, \mathcal{X}^i, y^i, \mathcal{Z}^i, \delta^i, \lambda^i \rangle$ is a *sequential machine*, where
  - $\mathcal{S}^i$ is a finite set of *module internal states*;
  - $\mathcal{X}^i$ is the set of *module input variables*; also, $x^i = (x^i_1, \ldots, x^i_{m^i})$ is the m^i-tuple of module input variables;
  - y^i is the *module internal state variable*;
  - $\mathcal{Z}^i$ is the set of *module output variables*; also, $z^i = (z^i_1, \ldots, z^i_{p^i})$ is the p^i-tuple of module output variables;
  - δ^i is the *module excitation function*, $\delta^i : \{0,1\}^{m^i} \times \mathcal{S}^i \to 2^{\mathcal{S}^i}$, and for any $a \in \{0,1\}^{m^i}$ and $b \in \mathcal{S}^i$, $\delta^i(a,b) \neq \emptyset$ and either $\delta^i(a,b) = \{b\}$ or $b \notin \delta^i(a,b)$;
  - $\lambda^i = (\lambda^i_1, \ldots, \lambda^i_{p^i})$ is the *module output function*, $\lambda^i : \mathcal{S}^i \to \{0,1\}^{p^i}$. □

An *input state* of the module is a tuple $a \in \{0,1\}^{m^i}$, an *internal state* of the module is an element $b \in \mathcal{S}^i$, and a *total state* is an ordered pair $(a,b) \in \{0,1\}^{m^i} \times \mathcal{S}^i$. Let $t = (a,b)$ be a total state. The set $T = \delta^i(t) \subseteq \mathcal{S}^i$ is the *excitation state set*, or simply, the *excitation* of the module in state t. If $T = \{b\}$, then the total state t and the module are said to be *stable*, and the internal state

cannot change. If $T \neq \{b\}$, then the total state and the module are *unstable* and the internal state may change to any state $b' \in T$ at any time. The state b' is non-deterministically selected by the module. If the cardinality of T is always 1, the module is said to be *deterministic*. In that case, we view δ^i as a function from the set of total states to $\mathcal{S}^i$. The *output state* of the module, which is a binary p^i-tuple, is determined solely by the present internal state. If the internal state changes, and the output state computed by the output function differs from the output state before the change, the new output value appears *instantaneously* together with the internal state change.

Definition 2.1 permits us to treat any sequential machine of the Moore type [7] as a module. In particular, delays (or wires with delays), forks, logic gates, latches, counters, C-elements, and arbiters are included. Note that in this model, forks are *isochronic*, meaning that all the output branches change at exactly the same time, simultaneously with internal state change. *Anisochronic* forks can be modelled within a network.

Now let us consider some examples of modules. If only one module is being discussed, we often drop the superscript i, for convenience. The modules of the first six examples are deterministic.

Example 2.1 A delay is a module $M^d = \langle G^d, C^d \rangle$, with $C^d = \langle \{0,1\}, \{x_1^d\}, y^d, \{z_1^d\}, \delta^d, \lambda^d \rangle$, and δ^d and λ^d defined as in Table 2.1. For future applications, we often use the superscript d to identify a delay module.

Table 2.1: Functions of a delay

	x		
y	0	1	$\lambda^d(y)$
0	0	1	0
1	0	1	1
	$\delta^d(x,y)$		

Table 2.2: Functions of a fork

	x		
y	0	1	$\lambda(y)$
0	0	1	00
1	0	1	11
	$\delta(x,y)$		

Example 2.2 An (isochronic) fork has $\mathcal{S} = \{0,1\}$, $m = 1$, $p = 2$, and δ and λ defined as in Table 2.2.

Example 2.3 A three-input majority element has $\mathcal{S} = \{0,1\}$, $m = 3$, and $p = 1$.

Table 2.3: Functions of a three-input majority element

	x								
y	000	001	010	011	100	101	110	111	$\lambda(y)$
0	0	0	0	1	0	1	1	1	0
1	0	0	0	1	0	1	1	1	1
	$\delta(x,y)$								

Example 2.4 A set-dominant set-reset latch has $\mathcal{S} = \{0, 1\}$, $m = 2$, and $p = 2$.

Table 2.4: Functions of a set-dominant set-reset latch

	x				
y	00	01	10	11	$\lambda(y)$
0	0	0	1	1	01
1	1	0	1	1	10
	$\delta(x, y)$				

Example 2.5 A modulo-4 counter has $\mathcal{S} = \{0, 1, 2, 3\}$, $m = 1$, and $p = 2$.

Table 2.5: Functions of a modulo-4 counter

	x		
y	0	1	$\lambda(y)$
0	0	1	00
1	2	1	01
2	2	3	10
3	0	3	11
	$\delta(x, y)$		

Example 2.6 A Muller C-element has $\mathcal{S} = \{0, 1\}$, $m = 2$, and $p = 1$.

Table 2.6: Functions of a Muller C-element

	x				
y	00	01	10	11	$\lambda(y)$
0	0	0	0	1	0
1	0	1	1	1	1
	$\delta(x, y)$				

Example 2.7 An arbiter has $\mathcal{S} = \{0, 1, 2\}$, $m = 2$, and $p = 2$.

Table 2.7: Functions of an arbiter

	x				
y	00	01	10	11	$\lambda(y)$
0	$\{0\}$	$\{2\}$	$\{1\}$	$\{1, 2\}$	00
1	$\{0\}$	$\{0\}$	$\{1\}$	$\{1\}$	10
2	$\{0\}$	$\{2\}$	$\{0\}$	$\{2\}$	01
	$\delta(x, y)$				

Arbiters are non-deterministic; when both inputs are 1 and the module is in state 0, the next state is chosen between 1 and 2 arbitrarily.

Definition 2.2 A module M^i, as in Definition 2.1, is said to be *one-step* if, for all $a \in \{0,1\}^{m^i}$ and $b \in \mathcal{S}^i$, whenever $b' \in \delta^i(a,b)$, then $\delta^i(a,b') = \{b'\}$.

All the modules we have seen so far are one-step. Here is an example of a module $M^{\updownarrow}$ which is not one-step. Its functions are defined as in Table 2.8. When the input is fixed at 1, the state of $M^{\updownarrow}$ will eventually oscillate between 1 and 2.

Table 2.8: Functions of $M^{\updownarrow}$

	x		
y	0	1	$\lambda(y)$
0	$\{2\}$	$\{1,2\}$	0
1	$\{0\}$	$\{2\}$	0
2	$\{2\}$	$\{1\}$	1
	$\delta(x,y)$		

3 Networks

We now introduce our circuit model, which we call "network." Networks are composed of modules. Also, following Muller's approach, we consider only autonomous networks.

Definition 3.1 A *network* is a pair $N = \langle \mathcal{M}, G \rangle$, where

- $\mathcal{M} = \{M^1, \ldots, M^n\}$, $n \geq 1$, is a set of modules, where M^i is as defined in Definition 2.1 for $1 \leq i \leq n$;
- $G = \langle \mathcal{V}, \mathcal{E} \rangle$ is a connected directed graph, the *network graph*, where
  - $\mathcal{V} = \bigcup_{i=1}^{n} \mathcal{V}^i$ is the set of *network vertices*;
  - $\mathcal{E} \supseteq \bigcup_{i=1}^{n} \mathcal{E}^i$ is the set of *network edges*, such that

$$\mathcal{E} \subseteq (\bigcup_{i=1}^{n} \mathcal{Z}^i \times \bigcup_{i=1}^{n} \mathcal{X}^i) \cup \bigcup_{i=1}^{n} \mathcal{E}^i,$$

    and

    1. for each module input vertex $x_l^j \in \mathcal{V}$, there exists exactly one module output vertex $z_k^i \in \mathcal{V}$ such that $(z_k^i, x_l^j) \in \mathcal{E}$;
    2. for each module output vertex $z_k^i \in \mathcal{V}$, there exists exactly one module input vertex $x_l^j \in \mathcal{V}$ such that $(z_k^i, x_l^j) \in \mathcal{E}$. □

We denote the set of network edges that are not internal module edges by $\mathcal{K}$, and refer to these edges as *connections*. The set of *state variables* of the network N is $\mathcal{Y} = \{y^1, \ldots, y^n\}$. The set of *states* of N is $\mathcal{S} = \mathcal{S}^1 \times \ldots \times \mathcal{S}^n$. If $s \in \mathcal{S}$, the i-th

component of s is denoted by s_i. In general, we adopt the convention that specific components of a tuple are indexed using subscripts.

Definition 3.2 Let $y^i \in \mathcal{Y}$. The *network excitation function* $\Delta_i : \mathcal{S} \to 2^{\mathcal{S}^i}$ of y^i, is the module excitation function δ^i of M^i with arguments changed as follows: If (z_k^h, x_l^i) is a connection, then the l-th input argument of δ^i is $\lambda_k^h(y^h)$.

That is, the output function λ^h performs an instantaneous encoding of the state of module M^h and provides the k-th bit of the encoded value as an input to module M^i. Since λ^h depends solely on y^h, the excitation function Δ_i becomes a function of y^h.

Definition 3.3 For $y^i \in \mathcal{Y}$ and $s \in \mathcal{S}$, the *excitation* of y^i in state s, denoted by S_i, is $S_i = \Delta_i(s)$.

Definition 3.4 A state s is *stable* if $S_i = \{s_i\}$ for all i; otherwise, s is *unstable*. We denote the set of unstable state variables in state s by

$$\mathcal{U}(s) = \{y^i \in \mathcal{Y} \mid S_i \neq \{s_i\}\}.$$

We now define the behavior of a network. Like most authors, for simplicity, we assume that only one signal changes at a time. This is essentially the *general single-winner* (GSW) model [2].

Definition 3.5 The *GSW relation* over the set of states of a network N is a binary relation $\mathcal{R}$, such that $(s,t) \in \mathcal{R}$, or $s\mathcal{R}t$, if and only if s differs from t in exactly one component i, $s_i \neq t_i$, $y^i \in \mathcal{U}(s)$, and $t_i \in S_i$.

Definition 3.6 A (GSW) *behavior* of a network N is an initialized directed graph $B = \langle q, \mathcal{Q}, \mathcal{R} \rangle$, where

- $q \in \mathcal{S}$ is the initial state;
- $\mathcal{Q}$, specifying the vertices of B, is the set of states reachable from q via the GSW relation $\mathcal{R}$,
$$\mathcal{Q} = \{s \in \mathcal{S} \mid q\mathcal{R}^* s\};$$
- $\mathcal{R}$, specifying the edges of B, is the GSW relation of Definition 3.5 restricted to $\mathcal{Q}$.

If $(s,t) \in \mathcal{R}$, we attach to edge (s,t) a *tag* $\tau(s,t) \in \mathcal{Y}$, which denotes the state variable in which s and t differ. □

Definition 3.2 reflects a functional dependency among state variables (or corresponding modules). This is captured by the "feed" relation, defined as follows:

Definition 3.7 A module M^i is said to *feed* a module M^j, denoted by $(M^i, M^j) \in \mathcal{F}$, or $M^i \mathcal{F} M^j$, if there exist $z_k^i \in \mathcal{Z}^i$ and $x_l^j \in \mathcal{X}^j$ such that $(z_k^i, x_l^j) \in \mathcal{K}$.

The relation $\mathcal{F}$ can also be viewed as being defined over the set of state variables. That is, $(y^i, y^j) \in \mathcal{F}$ if and only if $(M^i, M^j) \in \mathcal{F}$.

Proposition 3.1 *Let $(s,t) \in \mathcal{R}$ and $\tau(s,t) = y^i$. If $S_j \neq T_j$ for some $j \neq i$, then $(y^i, y^j) \in \mathcal{F}$.*

Proof: Since $j \neq i$, we have $s_j = t_j$. Thus, if $S_j \neq T_j$, the value of some input argument of Δ_j must have changed along with the state transition. As the only state variable changed is y^i, by Definitions 3.2 and 3.7, $(y^i, y^j) \in \mathcal{F}$. □

A behavior $B = \langle q, \mathcal{Q}, \mathcal{R} \rangle$ of a network N can be viewed as a non-deterministic finite automaton $\mathcal{B} = \langle \mathcal{Y}, \mathcal{Q}, q, \rho, \mathcal{Q} \rangle$, where

- $\mathcal{Y}$ is the input alphabet;
- $\mathcal{Q}$ is the state set;
- q is the initial state;
- ρ is the state transition function, $\rho : \mathcal{Q} \times \mathcal{Y} \to 2^{\mathcal{Q}}$, such that

  $$\rho(s, y^i) = \{t \in \mathcal{Q} \mid s \mathcal{R} t \text{ and } \tau(s,t) = y^i\}$$

  for $s \in \mathcal{Q}$ and $y^i \in \mathcal{Y}$;
- $\mathcal{Q}$ is the set of accepting states.

We extend the state transition function ρ to $\rho^* : \mathcal{Q} \times \mathcal{Y}^* \to 2^{\mathcal{Q}}$, which can be defined inductively as follows. For $s \in \mathcal{Q}$, $y^i \in \mathcal{Y}$, and $w \in \mathcal{Y}^*$,

- $\rho^*(s, \epsilon) = \{s\}$;
- $\rho^*(s, wy^i) = \{t \in \mathcal{Q} \mid t \in \rho(s', y^i) \text{ for some } s' \in \rho^*(s, w)\}$.

We often represent the state set $\rho^*(s, w)$ by the short-hand form sw. The language $L(\mathcal{B})$ accepted by $\mathcal{B}$ is defined in the usual way,

$$L(\mathcal{B}) = \{w \in \mathcal{Y}^* \mid \rho^*(q, w) \neq \emptyset\}.$$

It is worth noting that $L(\mathcal{B})$ is always prefix-closed.

Example 3.1 Figure 3.1 shows a network N composed of a C-element, an inverter, and a fork. The set of state variables is $\mathcal{Y} = \{y^1, y^2, y^3\}$. The behavior of N with initial state $(0,0,0)$ is shown in Figure 3.2[1]. We see that this behavior is *sequential*.

[1]From now on, when showing particular state tuples, we omit parentheses and commas, for brevity; also, unstable state components are in boldface.

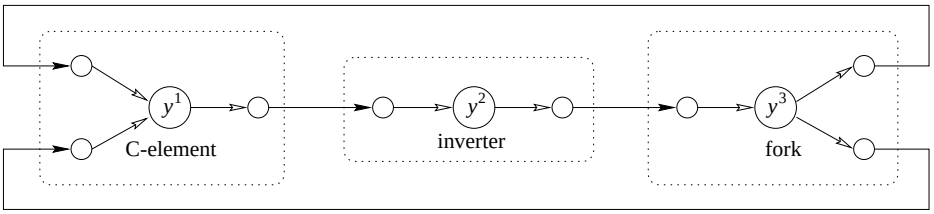

Figure 3.1: Network N.

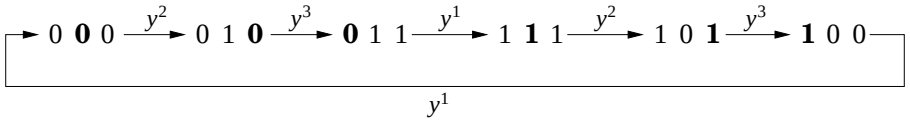

Figure 3.2: A behavior of network N.

Example 3.2 Figure 3.3 shows a network N' composed of a two-output C-element and two inverters. The behavior of N' with initial state 0**00** is shown in Figure 3.4. Note the existence of races in the behavior.

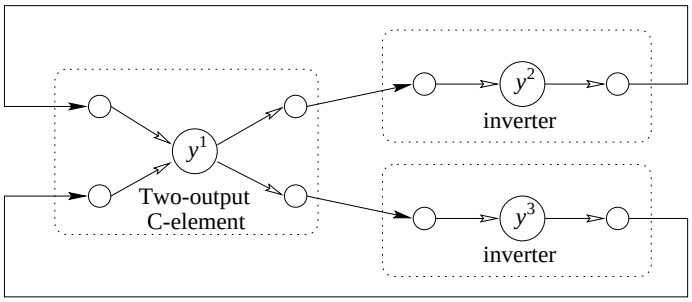

Figure 3.3: Network N'.

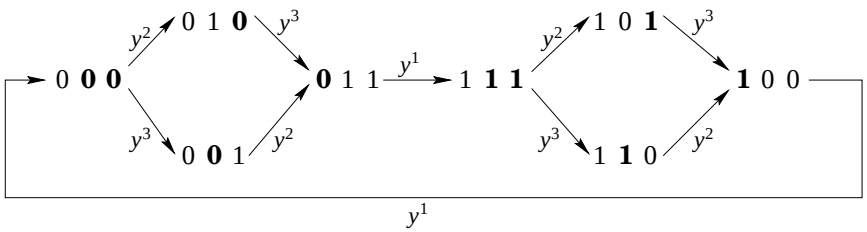

Figure 3.4: A behavior of network N'.

4 Delay-insensitivity of networks

In this section, we first model asynchronous circuits using networks. An asynchronous circuit is characterized by a so-called delay-dense network. Then we describe how we model all possible delay distributions in an asynchronous circuit. Finally, we give a formal definition for delay-insensitivity of networks.

4.1 Delay-dense networks

An asynchronous circuit consists of a set of components interconnected by wires, where each component can be modelled by a module. The distinction between the current state and the current excitation of a module allows us to associate a delay with the module. To model wire delays, we treat wires as delay modules. Intuitively, an asynchronous circuit is delay-insensitive if the correctness of its operation is independent of the delays in its components and in the wires connecting the components. We only assume that the delays are finite, but arbitrary. As both component and wire delays are to be taken into account, any asynchronous circuit can be modelled precisely by a *delay-dense* network, defined as follows:

Definition 4.1 A network N, as in Definition 3.1, is *delay-dense* if, whenever $(M^i, M^j) \in \mathcal{F}$, then either M^i or M^j is a delay module.

For example, the networks of Examples 3.1 and 3.2 are not delay-dense. They become delay-dense if we "insert" a delay in each of the network connections.

Proposition 4.1 *If N is delay-dense and $(M^i, M^i) \in \mathcal{F}$ for some M^i, then N consists of a single delay module.*

Proof: If N is delay-dense and $(M^i, M^i) \in \mathcal{F}$, then M^i must be a delay and $(z_1^i, x_1^i) \in \mathcal{K}$. Thus, the vertices of M^i cannot be connected with vertices of other modules. Since the network graph is connected, N consists of a single delay module, namely, M^i. □

4.2 Delay extensions

To determine whether a network is delay-insensitive, we wish to examine how the distribution of delays in the modules affects the behavior of the network. The motivation behind our approach to this problem is Charles Molnar's "foam-rubber wrapper" principle [6], where a module is viewed as being surrounded by a flexible boundary to reflect the fact that communication between its terminals and its environment takes place through a medium involving unknown delays. This intuitive idea corresponds to surrounding each module in a network by delays. However, instead of "wrapping" all the modules by delays at once, we choose to proceed inductively using the notion of *delay extension*. This facilitates the mathematical treatment of the problem.

Definition 4.2 Let N be a network. A *delay successor* of N is a network $\hat{N} = \langle \hat{\mathcal{M}}, \hat{G} \rangle$ obtained from N by inserting a delay M^{n+1} in a connection $e = (z_k^i, x_l^j)$ of N, where

- $\hat{\mathcal{M}} = \mathcal{M} \cup \{M^{n+1}\}$;
- $\hat{G} = \langle \hat{\mathcal{V}}, \hat{\mathcal{E}} \rangle$, where
  - $\hat{\mathcal{V}} = \mathcal{V} \cup \mathcal{V}^{n+1}$;
  - $\hat{\mathcal{E}} = (\mathcal{E} - \{e\}) \cup \mathcal{E}^{n+1} \cup \{(z_k^i, x_1^{n+1}), (z_1^{n+1}, x_l^j)\}$.

Definition 4.3 The set $\mathcal{D}(N)$ of *delay extensions of a network* N is defined inductively, as follows,

- $N \in \mathcal{D}(N)$;
- if $\hat{N} \in \mathcal{D}(N)$, then each delay successor of $\hat{N}$ is also in $\mathcal{D}(N)$.

As a convention, we denote a delay extension of a network N by $\hat{N}$. Furthermore, all the components of $\hat{N}$, any object associated with $\hat{N}$, e.g., a behavior, and the components of that object are all marked by $\hat{}$.

In order to compare the behavior of a network N with the behavior of a delay extension of N, it is essential that we relate the corresponding states and state transition sequences in the two behaviors.

Definition 4.4 Let $\hat{N} \in \mathcal{D}(N)$. The *projection of a state* $\hat{s}$ of $\hat{N}$ with respect to N is the $|\mathcal{Y}|$-tuple $s = \hat{s} \downarrow \mathcal{Y}$ obtained from $\hat{s}$ by removing all components corresponding to variables not in $\mathcal{Y}$. The *projection of a word* $\hat{w} \in \hat{\mathcal{Y}}^*$ with respect to N is the word $w = \hat{w} \downarrow \mathcal{Y}$ obtained from $\hat{w}$ by erasing all the symbols not in $\mathcal{Y}$.

Definition 4.5 Let $\hat{N} \in \mathcal{D}(N)$, and let s be a state of N. An *extension* of s with respect to $\hat{N}$ is a state $\hat{s}$ of $\hat{N}$ for which $\hat{s} \downarrow \mathcal{Y} = s$. Let $\hat{s}$ be an extension of s; if $\mathcal{U}(\hat{s}) \subseteq \mathcal{Y}$, in other words, if all the inserted delays are stable in $\hat{s}$, then we say that $\hat{s}$ is the *stable-delay extension* of s.

We now prove two useful propositions concerning state extensions. In the following, $\hat{N} \in \mathcal{D}(N)$, and s and $\hat{s}$ are states of N and $\hat{N}$, respectively. We call a module of $\hat{N}$ which is also present in N an *original* module.

Proposition 4.2 *If $\hat{s}$ is the stable-delay extension of s, then $S_i = \hat{S}_i$ for all i such that M^i is an original module.*

Proof: By Definition 3.3, it suffices to prove that in state s, the value of each argument of Δ_i agrees with that of the corresponding argument of $\hat{\Delta}_i$ in state $\hat{s}$. Since $s_i = \hat{s}_i$, we only need to worry about the input arguments. Let the k-th input argument of $\hat{\Delta}_i$ be $\lambda_l^h(\hat{s}_h)$. If M^h is original, then we are done. Otherwise, M^h is an inserted delay. Then there exists a "chain" of inserted delays of which

M^h is the tail, and some original module M^g feeds the head of the chain. As $\hat{s}$ is the stable-delay extension of s, all the delays in the chain are stable in $\hat{s}$. Thus, $\lambda_j^g(\hat{s}_g) = \hat{s}_h$, where we assume that the j-th output of M^g feeds the head of the delay chain. Furthermore, $\lambda_l^h(\hat{s}_h) = \hat{s}_h = \lambda_j^g(\hat{s}_g)$. On the other hand, it is clear that $(M^g, M^i) \in \mathcal{F}$ and the value of the k-th input argument of Δ_i is $\lambda_j^g(s_g) = \lambda_j^g(\hat{s}_g)$, same as that of $\hat{\Delta}_i$. We are done. □

Corollary 4.2.1 *If $\hat{s}$ is the stable-delay extension of s, then $\mathcal{U}(\hat{s}) = \mathcal{U}(s)$.*

Proposition 4.3 *Let $\hat{s}$ be an extension of s. Then there exists $\hat{w} \in (\hat{\mathcal{Y}} - \mathcal{Y})^*$ such that $\hat{s}\hat{w} = \{\hat{s}'\}$, where $\hat{s}'$ is the stable-delay extension of s.*

Proof: Let us assign "levels" to the inserted delays in $\hat{N}$ inductively as follows: a) A delay is of level 1 if it is fed by an original module; b) a delay is of level $k+1$ if it is fed by a delay of level k. Clearly, there are no cycles of inserted delays in $\hat{N}$. Also, every delay has a unique input and output. Therefore, each inserted delay is assigned a unique level. Starting in state $\hat{s}$, we change inserted delays which are unstable in such a way that at each step, we always choose one which has the least level. We stop when all the inserted delays are stabilized. One easily verifies that the process will terminate when the stable-delay extension $\hat{s}'$ of s is reached. Moreover, since delays are deterministic, $\hat{s}'$ is the only state reachable. □

4.3 Formal definition of network delay-insensitivity

The delay extensions of a network give a description of all possible delay distributions within the network. For a network to be delay-insensitive, the behavior of any delay extension of the network should satisfy a certain correctness concern with respect to the behavior of the original network. We define this correctness concern to correspond to *weak bisimulation* [5] (or *observation equivalence*), one of the strongest forms of behavioral equivalence. Transitions occurring on the inserted delays are treated as non-observable "internal" actions; in fact, they can be viewed simply as a passing of time.

We will give a formal definition of network delay-insensitivity shortly. But first, in order to compare the behavior of a network with the behavior of a delay extension of the network, we have to choose an initial state for the latter behavior. It only makes sense to choose an initial state which "imitates" the initial state of the behavior of the original network.

Definition 4.6 Let $\hat{N} \in \mathcal{D}(N)$, and let $B = \langle q, \mathcal{Q}, \mathcal{R} \rangle$ be a behavior of N. A behavior $\hat{B} = \langle \hat{q}, \hat{\mathcal{Q}}, \hat{\mathcal{R}} \rangle$ of $\hat{N}$ is said to be *initial-state compatible with B* if $\hat{q}$ is the stable-delay extension of q.

We now define delay-insensitivity of networks formally. Note that in addition to observation equivalence, we require livelock-freedom.

Definition 4.7 Let $\hat{N} \in \mathcal{D}(N)$. Let $B = \langle q, \mathcal{Q}, \mathcal{R} \rangle$ be a behavior of N, and let $\hat{B} = \langle \hat{q}, \hat{\mathcal{Q}}, \hat{\mathcal{R}} \rangle$ be the behavior of $\hat{N}$ which is initial-state compatible with B.

- The behavior $\hat{B}$ is said to be *safe with respect to* B if, whenever $\hat{s} \in \hat{\mathcal{Q}}$ is an extension of $s \in \mathcal{Q}$, then for all $\hat{t} \in \hat{\mathcal{Q}}$ and $\hat{w} \in \hat{\mathcal{Y}}^*$, if $\hat{t} \in \hat{s}\hat{w}$ and $\hat{w} \downarrow \mathcal{Y} = y^i$ for some $y^i \in \mathcal{Y}$, then there exists $t \in \mathcal{Q}$ such that $t \in sy^i$ and t is the projection of $\hat{t}$ with respect to N.
- The behavior $\hat{B}$ is said to be *faithful with respect to* B if, whenever $\hat{s} \in \hat{\mathcal{Q}}$ is an extension of $s \in \mathcal{Q}$, then for all $t \in sy^i$, there exist $\hat{w} \in \hat{\mathcal{Y}}^*$ and $\hat{t} \in \hat{\mathcal{Q}}$ such that $\hat{t} \in \hat{s}\hat{w}$, $\hat{w} \downarrow \mathcal{Y} = y^i$, and $\hat{t}$ is an extension of t with respect to $\hat{N}$.
- The behavior $\hat{B}$ is said to be *livelock-free with respect to* B if, for all $\hat{s} \in \hat{\mathcal{Q}}$ and $\hat{w} \in (\hat{\mathcal{Y}} - \mathcal{Y})^+$, $\hat{s} \notin \hat{s}\hat{w}$.

Definition 4.8 A network N is *delay-insensitive with respect to a state* q if, for *any* delay extension $\hat{N}$ of N, $\hat{B}$ is safe, faithful, and livelock-free with respect to B, where $B = \langle q, \mathcal{Q}, \mathcal{R} \rangle$ is the behavior of N originating from q, and $\hat{B}$ is the behavior of $\hat{N}$ which is initial-state compatible with B.

We now show that faithfulness and livelock-freedom are redundant properties, since they are satisfied by all delay extensions. Consequently, only safety is required.

Proposition 4.4 *The behavior $\hat{B}$ of Definition 4.8 is always livelock-free with respect to B.*

Proof: Suppose that there exist $\hat{s} \in \hat{\mathcal{Q}}$ and $\hat{w} \in (\hat{\mathcal{Y}} - \mathcal{Y})^+$ such that $\hat{s} \in \hat{s}\hat{w}$. Let y^i be an inserted delay which appears in $\hat{w}$. Clearly, y^i must appear at least twice in $\hat{w}$. Thus, the (unique) state variable y^h that feeds y^i must have changed and appears in $\hat{w}$. It follows that y^h is an inserted delay. Applying the same argument (this time, on y^h, to start with) repeatedly will lead to a contradiction since there are no cycles of inserted delays in $\hat{N}$. We are done. □

Proposition 4.5 *The behavior $\hat{B}$ of Definition 4.8 is always faithful with respect to B.*

Proof: Let $\hat{s} \in \hat{\mathcal{Q}}$ be an extension of $s \in \mathcal{Q}$, and let $t \in sy^i$. By Proposition 4.3, there exists $\hat{w} \in (\hat{\mathcal{Y}} - \mathcal{Y})^*$ such that $\hat{s}\hat{w} = \{\hat{s}'\}$, where $\hat{s}'$ is the stable-delay extension of s. By Proposition 4.2, $S_i = \hat{S}'_i$. Therefore, there exists $\hat{t} \in \hat{s}'y^i$ such that $\hat{t}$ is an extension of t with respect to N. Thus, we have $\hat{t} \in \hat{s}'y^i \subseteq \hat{s}\hat{w}y^i$, $\hat{w}y^i \downarrow \mathcal{Y} = y^i$, and $\hat{t}$ is an extension of t with respect to N. By Definition 4.7, $\hat{B}$ is faithful with respect to B. □

In view of Proposition 4.4 and 4.5, we have the following:

Theorem 4.1 *A network N is delay-insensitive with respect to a state q if, for any delay extension $\hat{N}$ of N, $\hat{B}$ is safe with respect to B, where B and $\hat{B}$ are as defined in Definition 4.8.*

Example 4.1 Consider part of a network N shown in Figure 4.1(a) (for brevity, we hide the internal structures of modules), where two delays are connected to an arbiter (as defined in Example 2.7). Figure 4.1(c) shows the corresponding part of a delay successor $\hat{N}$ of N, where the inserted delay is shaded. Let the current values and excitations of the modules be as shown, where excitations are shown in brackets.

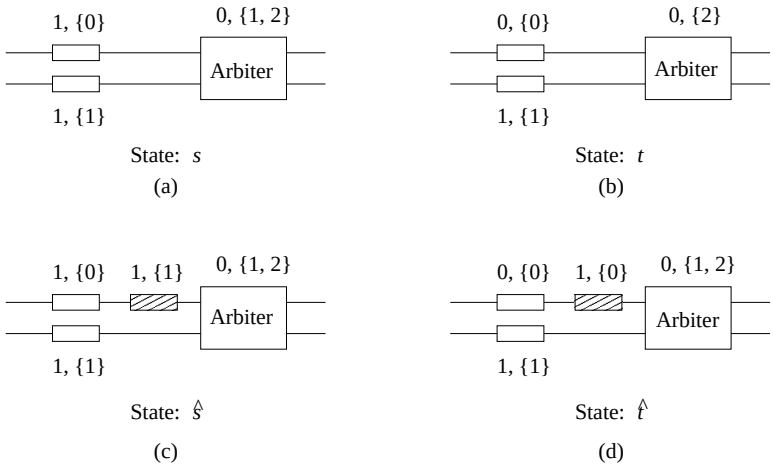

Figure 4.1: Example of safety violation.

Each figure represents a network state: State t of N, as shown in Figure 4.1(b), results from state s (Figure 4.1(a)) by a delay change; state $\hat{t}$ of $\hat{N}$ (Figure 4.1(d)) results from state $\hat{s}$ (Figure 4.1(c)) by the same delay change; also, $\hat{s}$ is the stable-delay extension of s. We assume that the states of network modules not shown are the same in all four states.

Clearly, $\hat{t}$ is an extension of t. In state $\hat{t}$, the arbiter may change to state 1 or 2, whereas in state t, the arbiter can only change to 2. This demonstrates a violation of safety. Therefore, N is not delay-insensitive with respect to state s. □

5 Semi-modularity of network behaviors

Semi-modularity was initially defined in [4] for deterministic circuit behaviors. We generalize the definition of semi-modularity to permit non-determinism, so that we can include arbiters.

Definition 5.1 Let $B = \langle q, \mathcal{Q}, \mathcal{R} \rangle$ be a behavior of a network N. A state $s \in \mathcal{Q}$ is said to be *semi-modular* if, for all $t \in \mathcal{Q}$, if $(s,t) \in \mathcal{R}$ with $\tau(s,t) = y^i$, then $S_j \subseteq T_j$ for all $j \neq i$ such that $y^j \in \mathcal{U}(s)$.

Definition 5.2 A behavior $B = \langle q, \mathcal{Q}, \mathcal{R} \rangle$ of a network N is *semi-modular* if, for all $s \in \mathcal{Q}$, s is semi-modular.

Intuitively, semi-modularity requires that once a state variable y^i becomes unstable and b is in its excitation, b should remain in the excitation until y^i changes.

Example 5.1 For a violation of semi-modularity, refer to Figure 4.1(a) and (b). The delay change involved alters the excitation of the arbiter and violates semi-modularity.

Example 5.2 Figure 5.1 shows a network N consisting of an arbiter, a $\overline{C}$-element, and two delays. A $\overline{C}$-element functions like a Muller C-element with an inverted output.

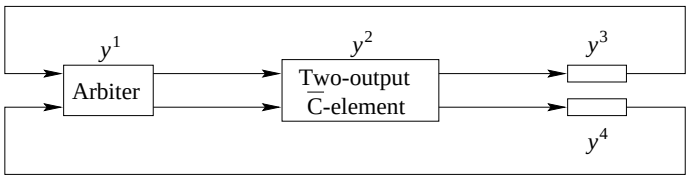

Figure 5.1 Network N.

Figure 5.2 shows a semi-modular behavior of N. This example illustrates that the set inclusion in Definition 5.1 can be proper. For example, in state **0**01**0**, the arbiter can change to state 1 only; in a subsequent state **0**011, resulting from a change on a delay, the arbiter can change to either 1 or 2.

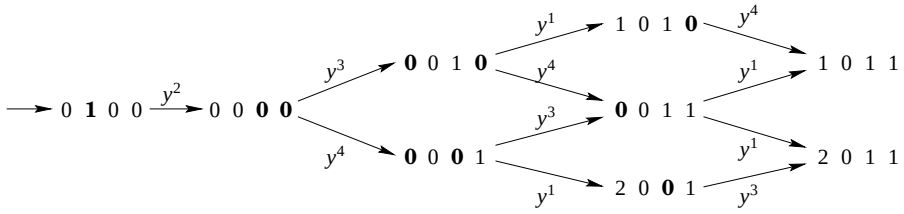

Figure 5.2 A semi-modular behavior of N.

Proposition 5.1 *A behavior $B = \langle q, \mathcal{Q}, \mathcal{R} \rangle$ is semi-modular if and only if, whenever $s, t \in \mathcal{Q}$, $t \in sw$ for some $w \in \mathcal{Y}^*$, $y^i \in \mathcal{U}(s)$, and $S_i \not\subseteq T_i$, then y^i appears in w.*

Proof: We prove the "if" part by contradiction. Assume that B is not semi-modular. Then there exists $(s, t) \in \mathcal{R}$ with $\tau(s, t) = y^j$ such that $S_i \not\subseteq T_i$ for some $i \neq j$ with $y^i \in \mathcal{U}(s)$. Let $w = y^j$. Then we have $t \in sw$, $S_i \not\subseteq T_i$, but y^i does

not appear in w. Now assume that B is semi-modular. Let $s, t \in \mathcal{Q}$ be such that $t \in sw$ for some $w \in \mathcal{Y}^*$ and $S_i \not\subseteq T_i$. Clearly, $w \neq \epsilon$. If y^i does not appear in w, then by semi-modularity and transitivity of set inclusions, we would have $S_i \subseteq T_i$, which is a contradiction. Hence, the proposition is proved. □

6 Delay-insensitivity and semi-modularity

6.1 Semi-modularity is preserved under delay extensions

In this section, we prove that semi-modularity of behaviors of delay-dense networks is preserved under delay extensions. This result is essential in the proof of the main result of the paper, which is presented in the next section. The first theorem concerns delay successors. The second theorem extends this result to delay extensions.

Theorem 6.1 *Let N be a network, and let $B = \langle q, \mathcal{Q}, \mathcal{R} \rangle$ be a semi-modular behavior of N. Let $\hat{N}$ be a delay successor of N, where the extra delay M^d is inserted in a connection between two distinct modules. Without loss of generality, assume that the connection is from M^1 to M^2. If M^2 is a delay, then $\hat{B} = \langle \hat{q}, \hat{\mathcal{Q}}, \hat{\mathcal{R}} \rangle$, the behavior of $\hat{N}$ which is initial-state compatible with B, is also semi-modular.*

Proof: By the construction of $\hat{N}$, we have $(y^1, y^2) \in \mathcal{F}$ and $(y^1, y^d), (y^d, y^2) \in \hat{\mathcal{F}}$. The result follows from a series of four lemmata. In the following, we assume that $B = \langle q, \mathcal{Q}, \mathcal{R} \rangle$ is semi-modular. Also, let $\hat{s} \in \hat{\mathcal{Q}}$ be any state of $\hat{N}$. The state $\hat{s}$ is said to be *normal* if $\hat{s} \downarrow \mathcal{Y} \in \mathcal{Q}$; in that case, we let $s = \hat{s} \downarrow \mathcal{Y}$.

Lemma 6.1.1 *Let $\hat{s}$ be normal. If $i \neq d$ and $i \neq 2$, then $\hat{S}_i = S_i$ and $\mathcal{U}(\hat{s}) - \{y^d, y^2\} \subseteq \mathcal{U}(s)$.*

Proof: Clearly, M^i is an original module. One easily verifies that the input connections to M^i are identical in the two networks N and $\hat{N}$. Thus, the value of each argument of Δ_i in s is equal to the value of the corresponding argument of $\hat{\Delta}_i$ in $\hat{s}$. Hence, $\hat{S}_i = S_i$. The second result follows immediately. □

Lemma 6.1.2 *If $\hat{s}$ is normal, but not semi-modular, then $y^d \in \mathcal{U}(\hat{s})$.*

Proof: Let $s = \hat{s} \downarrow \mathcal{Y} \in \mathcal{Q}$. Suppose that $\hat{s}$ is not semi-modular. Then there exists $\hat{t} \in \hat{\mathcal{Q}}$ such that

1. $(\hat{s}, \hat{t}) \in \hat{\mathcal{R}}$ with $\tau(\hat{s}, \hat{t}) = y^i$;
2. there exists $j \neq i$ such that $y^j \in \mathcal{U}(\hat{s})$ and $\hat{S}_j \not\subseteq \hat{T}_j$.

By Proposition 3.1, $(y^i, y^j) \in \hat{\mathcal{F}}$. Also, $y^i \in \mathcal{U}(\hat{s})$, by definition. Assume that $y^d \notin \mathcal{U}(\hat{s})$. Then, $i \neq d$ and $j \neq d$. Since M^2 (a delay) has exactly one input, $i \neq d$, and $(y^i, y^j), (y^d, y^2) \in \hat{\mathcal{F}}$, we must have $j \neq 2$.

Since $y^d \notin \mathcal{U}(\hat{s})$, $\hat{s}$ is the stable-delay extension of s. Also, M^i and M^j are original. By Proposition 4.2, $\hat{S}_j = S_j$ and $\hat{S}_i = S_i$. It follows that there exists $t \in \mathcal{Q}$ such that $s\mathcal{R}t$, $\tau(s,t) = y^i$, and $t = \hat{t} \downarrow \mathcal{Y}$. Thus, $\hat{t}$ is normal. Recall that $j \neq d$ and $j \neq 2$. By Lemma 6.1.1, $\hat{T}_j = T_j$. On the other hand, $y^j \in \mathcal{U}(\hat{s}) = \mathcal{U}(s)$. Also, B is semi-modular and $j \neq i$. Therefore, $S_j \subseteq T_j$. It follows that $\hat{S}_j = S_j \subseteq T_j = \hat{T}_j$—a contradiction. Hence, if $\hat{s}$ is not semi-modular, then $y^d \in \mathcal{U}(\hat{s})$. □

Lemma 6.1.3 *If $\hat{s}$ is normal, but not semi-modular, then either $y^2 \in \mathcal{U}(\hat{s})$ or there exists $\hat{t} \in \hat{\mathcal{Q}}$ such that $(\hat{s},\hat{t}) \in \hat{\mathcal{R}}$, $\tau(\hat{s},\hat{t}) = y^1$, and $\hat{S}_d \not\subseteq \hat{T}_d$.*

Proof: Let $s = \hat{s} \downarrow \mathcal{Y} \in \mathcal{Q}$. Suppose that $\hat{s}$ is not semi-modular. Let i, j, and $\hat{t}$ be as defined in the proof of Lemma 6.1.2. Again, we have $(y^i, y^j) \in \hat{\mathcal{F}}$ and $y^i \in \mathcal{U}(\hat{s})$. Assume that $y^2 \notin \mathcal{U}(\hat{s})$. Then, $i \neq 2$ and $j \neq 2$. Since M^d (a delay) has exactly one output, $j \neq 2$, and (y^i, y^j), $(y^d, y^2) \in \hat{\mathcal{F}}$, we must have $i \neq d$.

Suppose that $j \neq d$. By Lemma 6.1.1, $\hat{S}_j = S_j$. Similarly, $\hat{S}_i = S_i$. It follows that there exists $t \in \mathcal{Q}$ such that $s\mathcal{R}t$, $\tau(s,t) = y^i$, and $t = \hat{t} \downarrow \mathcal{Y}$. Thus, $\hat{t}$ is normal. By Lemma 6.1.1 again, $\hat{T}_j = T_j$. On the other hand, by Lemma 6.1.1, $y^j \in \mathcal{U}(\hat{s}) - \{y^d, y^2\} \subseteq \mathcal{U}(s)$. Also, B is semi-modular and $j \neq i$. Therefore, $S_j \subseteq T_j$. It follows that $\hat{S}_j = S_j \subseteq T_j = \hat{T}_j$—a contradiction. Hence, $j = d$. Since $(y^i, y^j) \in \hat{\mathcal{F}}$, we must have $i = 1$. Thus, $(\hat{s},\hat{t}) \in \hat{\mathcal{R}}$, $\tau(\hat{s},\hat{t}) = y^1$, and $\hat{S}_d \not\subseteq \hat{T}_d$, as in the lemma. □

Lemma 6.1.4 *For all $\hat{s} \in \hat{\mathcal{Q}}$, $\hat{s}$ is semi-modular and normal.*

Proof: We proceed by induction. For the base case, let $\hat{s} = \hat{q}$. Clearly, $\hat{s}$ is normal and $y^d \notin \mathcal{U}(\hat{s})$. By Lemma 6.1.2, $\hat{s}$ is semi-modular. Now, let $(\hat{r},\hat{s}) \in \hat{\mathcal{R}}$ with $\tau(\hat{r},\hat{s}) = y^i$. Assume that $\hat{r}$ is semi-modular and normal. We shall prove that $\hat{s}$ is semi-modular and normal.

Let $r = \hat{r} \downarrow \mathcal{Y} \in \mathcal{Q}$. If $y^d \notin \mathcal{U}(\hat{r})$, then $i \neq d$ and $\hat{R}_i = R_i$. It follows that $\hat{s} \downarrow \mathcal{Y} \in \mathcal{Q}$. Now assume that $y^d \in \mathcal{U}(\hat{r})$. If $y^2 \in \mathcal{U}(\hat{r})$, then by changing y^d in state $\hat{r}$, we would have a violation of semi-modularity, contradicting the semi-modularity of $\hat{r}$. Therefore, $y^2 \notin \mathcal{U}(\hat{r})$ and $i \neq 2$. If $i = d$, then $\hat{s} \downarrow \mathcal{Y} = \hat{r} \downarrow \mathcal{Y} \in \mathcal{Q}$; otherwise, by Lemma 6.1.1, we have $\hat{R}_i = R_i$, which implies that $\hat{s} \downarrow \mathcal{Y} \in \mathcal{Q}$. Hence, $\hat{s}$ is normal.

Suppose that $\hat{s}$ is not semi-modular. Let the output vertex of M^1 which is connected to M^d be z_k^1. Clearly, z_k^1 is connected to M^2 in N. By Lemma 6.1.2, $y^d \in \mathcal{U}(\hat{s})$. It follows that $i \neq d$. By Lemma 6.1.3, there are two cases to consider.

- $y^2 \in \mathcal{U}(\hat{s})$:

  Note that y^2 is a delay. Since $i \neq d$, $y^2 \in \mathcal{U}(\hat{r})$. By a similar argument given above, we have $y^d \notin \mathcal{U}(\hat{r})$. Thus, $i = 1$ and $y^1 \in \mathcal{U}(\hat{r})$. On the other hand, since $y^d \notin \mathcal{U}(\hat{r})$, $\hat{r}$ is the stable-delay extension of r. It follows that $y^1, y^2 \in \mathcal{U}(\hat{r}) = \mathcal{U}(r)$ and there exists $s \in \mathcal{Q}$ such that $r\mathcal{R}s$, $\tau(r,s) = y^1$, and $s = \hat{s} \downarrow \mathcal{Y}$. We have $R_2 = \lambda_k^1(r_1) = \lambda_k^1(\hat{r}_1) = \hat{R}_d = \hat{r}_d$ and $S_2 = \lambda_k^1(s_1) = \lambda_k^1(\hat{s}_1) = \hat{S}_d$. Since $y^d \in \mathcal{U}(\hat{s})$, $\hat{s}_d \neq \hat{S}_d$. But $\hat{r}_d = \hat{s}_d$. Therefore, $R_2 \neq S_2$, which violates semi-modularity of B—a contradiction.

- There exists $\hat{t} \in \hat{\mathcal{Q}}$ such that $(\hat{s}, \hat{t}) \in \hat{\mathcal{R}}$, $\tau(\hat{s}, \hat{t}) = y^1$, and $\hat{S}_d \not\subseteq \hat{T}_d$:
  Since $y^2 \notin \mathcal{U}(\hat{s})$, $\hat{s}_2 = \hat{s}_d$. Also, $\hat{s}_d \neq \lambda_k^1(\hat{s}_1)$, since $y^d \in \mathcal{U}(\hat{s})$. Let $s = \hat{s} \downarrow \mathcal{Y} \in \mathcal{Q}$. Then, $s_2 \neq \lambda_k^1(s_1)$, which implies that $y^2 \in \mathcal{U}(s)$. On the other hand, by Lemma 6.1.1, we have $\hat{S}_1 = S_1$. Therefore, there exists $t \in \mathcal{Q}$ such that $s\mathcal{R}t$, $\tau(s,t) = y^1$, and $t = \hat{t} \downarrow \mathcal{Y}$. We have $S_2 = \lambda_k^1(s_1) = \lambda_k^1(\hat{s}_1) = \hat{S}_d$, and similarly, $T_2 = \hat{T}_d$. Since $\hat{S}_d \not\subseteq \hat{T}_d$, i.e., $\hat{S}_d \neq \hat{T}_d$, we have $S_2 \neq T_2$, which violates semi-modularity of B—a contradiction again.

Hence, $\hat{s}$ is semi-modular. The lemma is proven by induction. □

By Lemma 6.1.4, $\hat{s}$ is semi-modular for all $\hat{s} \in \hat{\mathcal{Q}}$. Hence, $\hat{B}$ is semi-modular. We are done. □

Theorem 6.2 *Let N be a delay-dense network, and let $\hat{N} \in \mathcal{D}(N)$. Let $B = \langle q, \mathcal{Q}, \mathcal{R} \rangle$ be a behavior of N, and let $\hat{B} = \langle \hat{q}, \hat{\mathcal{Q}}, \hat{\mathcal{R}} \rangle$ be the behavior of $\hat{N}$ which is initial-state compatible with B. If B is semi-modular, then so is $\hat{B}$.*

Proof: We prove the theorem by induction on the number of inserted delays. The base case, where $\hat{N} = N$, is trivial. Now assume that $B = \langle q, \mathcal{Q}, \mathcal{R} \rangle$ is semi-modular. We insert a delay M^d into a connection e of N. If e connects a module to itself, then by Proposition 4.1, N consists of a single delay module; the result holds trivially in that case. Thus we assume otherwise, i.e., that M^d is inserted between two distinct modules M^i and M^j. Denote the resulting delay successor by $\hat{N}$. Thus, (M^i, M^d), $(M^d, M^j) \in \hat{\mathcal{F}}$. Let $\hat{B} = \langle \hat{q}, \hat{\mathcal{Q}}, \hat{\mathcal{R}} \rangle$ be the behavior of $\hat{N}$ which is initial-state compatible with B. We shall prove that $\hat{B}$ is semi-modular.

If M^j is a delay, then the result follows from Theorem 6.1; otherwise, since N is delay-dense, M^i must be a delay. Let $(M^h, M^i) \in \mathcal{F}$. Clearly, M^h is unique. Imagine that in the network N, we change the index of M^i to d, and call the new network $\tilde{N}$. Obviously, B is a semi-modular behavior of $\tilde{N}$ as well. Now $\hat{N}$ can be viewed as being obtained from $\tilde{N}$ by inserting M^i in the connection between M^h and M^d, in which case the new delay feeds the delay M^d. So the result we wish to prove again follows from Theorem 6.1. The theorem is proven by induction. □

It is worth noting that Theorem 6.2 does not necessarily hold for networks which are not delay-dense. For example, consider the network consisting of a two-output inverter and an AND-gate, as shown in Figure 6.1(a). Let the initial state be **0**0. The corresponding behavior is semi-modular, as shown in Figure 6.1(b). With a delay y^3 inserted, as shown in Figure 6.1(a), the corresponding behavior (Figure 6.1(c)) is no longer semi-modular; more specifically, the states 10**0** and **0**01 are not semi-modular.

6.2 Main theorem

In this section, the proof of the main theorem of the paper is presented. We first prove that network delay-insensitivity implies semi-modularity, where delay-density

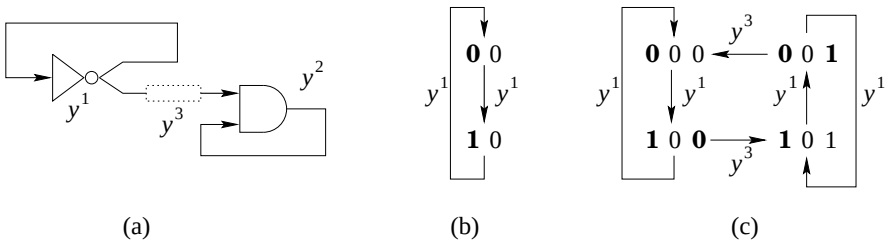

Figure 6.1: Theorem 6.2 fails to hold for networks which are not delay-dense.

of networks is not required. We then prove that semi-modularity implies delay-insensitivity for delay-dense networks. As any asynchronous circuit can be modelled precisely by a delay-dense network, we conclude that an asynchronous circuit is delay-insensitive if and only if its behavior is semi-modular.

Theorem 6.3 *If a network is delay-insensitive with respect to a state, then its behavior originating from that state is semi-modular.*

Proof: Let N be a network, which is delay-insensitive with respect to a state q. Let $B = \langle q, \mathcal{Q}, \mathcal{R} \rangle$ be the corresponding behavior. Let $\hat{N} \in \mathcal{D}(N)$ be obtained from N by inserting one delay in *each* connection of N. Let $\hat{B} = \langle \hat{q}, \hat{\mathcal{Q}}, \hat{\mathcal{R}} \rangle$ be the behavior of $\hat{N}$ which is initial-state compatible with B. By Theorem 4.1, $\hat{B}$ is safe with respect to B. Let us first prove a lemma, which will be used later.

Lemma 6.3.1 *Let $s \in \mathcal{Q}$. There exists $\hat{s} \in \hat{\mathcal{Q}}$ such that $\hat{s}$ is the stable-delay extension of s.*

Proof: We prove the result by induction. The base case, where $s = q$, is trivial. Let $s \in \mathcal{Q}$ and $t \in sy^i$, for some $y^i \in \mathcal{Y}$. Assume that there exists $\hat{s} \in \hat{\mathcal{Q}}$ such that $\hat{s}$ is the stable-delay extension of s. By Proposition 4.2, $\hat{S}_i = S_i$. Thus, there exists $\hat{t} \in \hat{\mathcal{Q}}$ such that $\hat{t} \in \hat{s}y^i$ and $\hat{t} \downarrow \mathcal{Y} = t$. By Proposition 4.3, there exists $w \in (\hat{\mathcal{Y}} - \mathcal{Y})^*$ such that $\hat{t}w = \{\hat{t}'\}$, where $\hat{t}' \in \hat{\mathcal{Q}}$ is the stable-delay extension of t. Thus, the lemma is proven. □

Now suppose that B is not semi-modular. Then there exists $(r, s) \in \mathcal{R}$ with $\tau(r, s) = y^i$ such that $R_j \not\subseteq S_j$ for some $j \neq i$ with $y^j \in \mathcal{U}(r)$. Let $b \in R_j$ such that $b \notin S_j$.

By Lemma 6.3.1, there exists $\hat{r} \in \hat{\mathcal{Q}}$ such that $\hat{r}$ is the stable-delay extension of r. By Proposition 4.2, $R_i = \hat{R}_i$ and $R_j = \hat{R}_j$. Thus, there exists $\hat{s} \in \hat{\mathcal{Q}}$ such that $\hat{s} \in \hat{r}y^i$ and $\hat{s} \downarrow \mathcal{Y} = s$. By our construction of the network $\hat{N}$, $(y^i, y^j) \notin \hat{\mathcal{F}}$. By Proposition 3.1, we have $\hat{R}_j = \hat{S}_j$. Note that $y^j \in \mathcal{U}(r) = \mathcal{U}(\hat{r})$. It follows that $y^j \in \mathcal{U}(\hat{s})$. Also, $b \in R_j = \hat{R}_j = \hat{S}_j$. Thus, there exists $\hat{t} \in \hat{\mathcal{Q}}$ such that $\hat{t} \in \hat{s}y^j$ and $\hat{t}_j = b$. Recall that $\hat{s}$ is an extension of s and $\hat{B}$ is safe with respect to B. Therefore, there exists $t \in \mathcal{Q}$ such that $t \in sy^j$ and $t = \hat{t} \downarrow \mathcal{Y}$. Thus, $t_j = b$, which implies that $b \in S_j$—a contradiction. Hence, B must be semi-modular. We are done. □

Theorem 6.4 *If a behavior of a delay-dense network is semi-modular, then the network is delay-insensitive with respect to the initial state of that behavior.*

Proof: Let N be a delay-dense network. Let $B = \langle q, \mathcal{Q}, \mathcal{R} \rangle$ be a semi-modular behavior of N. Let $\hat{N} \in \mathcal{D}(N)$, and let $\hat{B}$ be the behavior of $\hat{N}$ which is initial-state compatible with B. We shall prove that $\hat{B}$ is safe with respect to B.

Let $\hat{s} \in \hat{\mathcal{Q}}$ be an extension of $s \in \mathcal{Q}$. Let $\hat{t} \in \hat{\mathcal{Q}}$ and $\hat{w} \in \hat{\mathcal{Y}}^*$ be such that $\hat{t} \in \hat{s}\hat{w}$ and $\hat{w} \downarrow \mathcal{Y} = y^i$ for some $y^i \in \mathcal{Y}$. Let $\hat{t}_i = b$. Clearly, all we need to show is that $b \in S_i$. Let us write $\hat{w}$ as $\hat{w} = \hat{u}y^i\hat{u}'$, where $\hat{u}, \hat{u}' \in (\hat{\mathcal{Y}} - \mathcal{Y})^*$. Denote the state reached after $\hat{u}$ by $\hat{s}'$, $\hat{s}' \in \hat{s}\hat{u}$. Clearly, $y^i \in \mathcal{U}(\hat{s}')$ and $b \in \hat{S}'_i$. Also, $\hat{s}'$ is an extension of s. By Proposition 4.3, there exists $\hat{v} \in (\hat{\mathcal{Y}} - \mathcal{Y})^*$ such that $\hat{s}'\hat{v} = \{\hat{s}''\}$, where $\hat{s}''$ is the stable-delay extension of s. By Theorem 6.2, $\hat{B}$ is semi-modular. Furthermore, by Proposition 5.1, we have $\hat{S}'_i \subseteq \hat{S}''_i$, which implies that $b \in \hat{S}''_i$. On the other hand, by Proposition 4.2, we have $S_i = \hat{S}''_i$. Hence, $b \in S_i$. By definition, $\hat{B}$ is safe with respect to B. Since $\hat{N}$ was chosen arbitrarily, by Theorem 4.1, we conclude that N is delay-insensitive with respect to q. □

Note that Theorem 6.4 does not necessarily hold for networks which are not delay-dense. For example, consider the network N consisting of a single module $M^{\updownarrow}$ (as defined in Table 2.8), as shown in Figure 6.2(a).

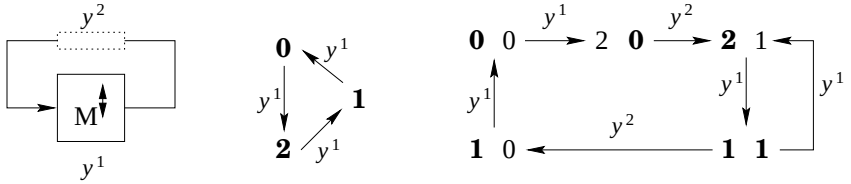

Figure 6.2: Theorem 6.4 fails to hold for networks which are not delay-dense.

Let the initial state be **0**. The corresponding behavior B is semi-modular, as shown in Figure 6.2(b). Now let us insert a delay y^2, as shown in Figure 6.2(a). The corresponding behavior $\hat{B}$ of the resulting delay extension $\hat{N}$ is given in Figure 6.2(c). Note that in state **11** of $\hat{B}$, $M^{\updownarrow}$ may change to 2; but in state **1** of B, $M^{\updownarrow}$ can only change to 0. Therefore, $\hat{B}$ is not safe with respect to B, and consequently, N is not delay-insensitive with respect to state **0**.

Theorem 6.3 holds for all networks. Combining Theorem 6.3 and 6.4, we have the main theorem of the paper:

Theorem 6.5 *A delay-dense network is delay-insensitive with respect to a state if and only if its behavior originating from that state is semi-modular.*

7 Concluding Remarks

We conclude the paper by classifying autonomous asynchronous networks using the criteria we discussed, i.e., delay-insensitivity, semi-modularity, and delay-density. Figure 7.1 shows a general picture. The universal set is really the set of *initialized* autonomous networks, i.e., autonomous networks with designated initial states.

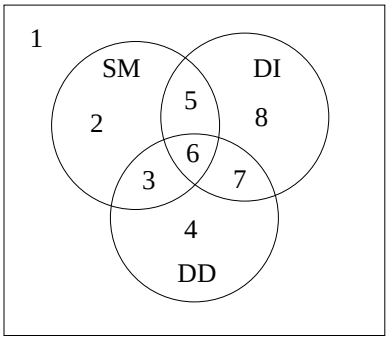

Figure 7.1: Classification of asynchronous networks.

We now describe each numbered region in Figure 7.1.

1. non-empty: a network belonging to this region is shown in Figure 7.2(a);
2. non-empty: an example was given in Figure 6.2(a) (without the delay y^2);
3. empty: by Theorem 6.4;
4. non-empty: an example is shown in Figure 7.2(b);
5. non-empty: an example is shown in Figure 7.2(c);
6. non-empty: an example is shown in Figure 7.2(d);
7. empty: by Theorem 6.3;
8. empty: by Theorem 6.3.

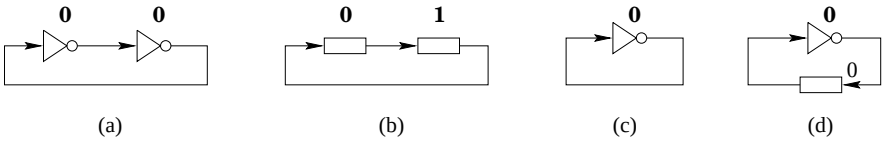

Figure 7.2: Network examples.

In addition, we also include speed-independent circuits, as defined by Muller [4], in our classification.

According to Muller, a circuit is speed-independent with respect to a state u if all *allowed state sequences* starting with u have the same *terminal class*. An allowed state sequence corresponds to a path in the behavior of the circuit; it is finite only if the last state of the sequence is stable; it cannot end in a *transient* cycle. A cycle of states is said to be transient if there is a state variable which is unstable and remains unchanged throughout the cycle. Let $\mathcal{R}^*$ be the reflexive and transitive closure of $\mathcal{R}$ (Definition 3.5). Two states a and b are equivalent if $a\mathcal{R}^*b$ and $b\mathcal{R}^*a$. Let $\mathcal{L}$ be a partial order defined over the induced equivalence classes: $A\mathcal{L}B$ if there exists $a \in A$ and $b \in B$ such that $a\mathcal{R}^*b$. For each allowed state sequence, there is a unique sequence of equivalence classes and a definite "last" class, the terminal class.

Muller considered only deterministic modules. In our network model, we call a network deterministic if all the modules in the network are deterministic; otherwise, it is non-deterministic. Muller showed that circuits exhibiting semi-modular behaviors form a *proper* subclass of speed-independent circuits. Using Muller's definition, we see that the circuit of Figure 5.2 is certainly not speed-independent; however, according to our definition, the circuit is semi-modular. This does not contradict Muller's result, for our definition of semi-modularity is less restrictive than that of Muller's. For deterministic networks, however, the two definitions are equivalent. Figure 7.3 shows the classification of deterministic initialized networks.

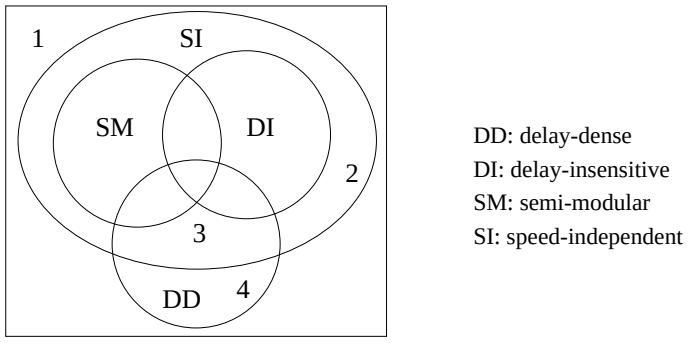

Figure 7.3: Classification of deterministic asynchronous networks.

We now describe each numbered region in Figure 7.3.

1. non-empty: a network belonging to this region was given in Figure 7.2(a);
2. non-empty: an example is shown in Figure 7.4(a) (from [4]);
3. non-empty: an example is shown in Figure 7.4(b);
4. non-empty: an example was given in Figure 7.2(b).

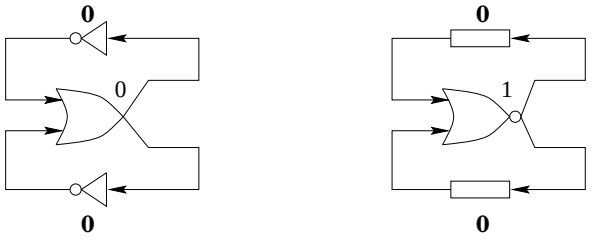

Figure 7.4: Network examples.

References

[1] J. A. Brzozowski, "Delay-Insensitivity and Ternary Simulation," *Proceedings of the First International Conference on Semigroups & Algebraic Engineering,*, Aizu-Wakamatsu City, Japan, March 24-38, 1997.

[2] J. A. Brzozowski and C-J. Seger, *Asynchronous Circuits,* Springer-Verlag, New York, NY, 1995.

[3] C. Mead and L. Conway, *Introduction to VLSI Systems,* Addison-Wesley, Reading, MA, 1980.

[4] R. E. Miller, *Switching Theory, Volume II: Sequential Circuits and Machines,* John Wiley & Sons, 1965.

[5] R. Milner, *Communication and Concurrency,* Prentice Hall, 1989.

[6] C. E. Molnar, T. P. Fang, and F. U. Rosenberger, "Synthesis of Delay-Insensitive Modules," *Proceedings of the 1985 Chapel Hill Conference on VLSI,* H. Fuchs, ed., Computer Science Press, Rockville, Maryland, pp. 67-86, 1985.

[7] E. F. Moore, "Gedanken Experiments on Sequential Machines," in C. E. Shannon and J. McCarthy, eds., *Automata Studies, Annals of Mathematics Study 34,* Princeton University Press, Princeton NJ, pp. 129–153, 1956.

[8] N. Shintel and M. Yoeli, "Synthesis of Modular Networks from Petri-Net Specifications," Technical Report #743, Israel Institute of Technology, Haifa 32000 Israel, 1992.

[9] J. T. Udding, "A Formal Model for Defining and Classifying Delay-Insensitive Circuits and Systems," *Distributed Computing,* vol. 1, no. 4, pp. 197–204, 1986.