A General Approach to Performance Analysis and Optimization of Asynchronous Circuits

Thesis by
Tak Kwan Lee

In Partial Fulfillment of the Requirements
for the Degree of
Doctor of Philosophy

California Institute of Technology
Pasadena, California

1995
(Submitted May 18, 1995)

Acknowledgements

I wish to express my gratitude

- to Alain Martin for being my mentor and guide;
- to Steve Burns for laying down the foundation;
- to Yaser Abu-Mostafa, Steve Burns, Mani Chandy, Rodney Goodman, and Alain Martin for being on my defense committee and giving me their valuable comments;
- to Chuck Seitz for teaching me how to design my very first chip;
- to ARPA and the Intel Foundation for generously funding my research;
- to Alain, Steve, Pieter Hazewindus, Dražen Borković, Marcel van der Goot, José Tierno, Christian Nielsen, Jessie Xu, James Cook, Peter Hofstee, Mika Nystroem, Andrew Lines, Rajit Manohar, and the other members of the Friday morning group for all the insightful discussions;
- to Arlene DesJardins, Wen-King Su, Chris Lee, Cindy Ferrini, Patty Renstrom, Gail Stowers, and Diane Goodfellow for always being so helpful;
- to the rest of the Caltech community, especially my ex-officemate Jakov Seizović, for making my stay as enjoyable and rewarding as it has been;
- to Kathleen, Claire, Denise, Geraldine, Davis, Rachelle, Steven, Jaime, Bê, E.T., Brandon, Huong, Minh, Huy, the CSULA social dance club, and many others for providing me with a "life" outside campus;

- to Tammy and the Chuchen family for their support and encouragement;
- to Becky, Tom, and Ashley for putting up with my escapades over the years;
- and, most importantly, to my parents for making it all possible.

Abstract

A systematic approach for evaluating and optimizing the performance of asynchronous VLSI circuits is presented. *Index-priority* simulation is introduced to efficiently find *minimal cycles* in the state graph of a given circuit. These minimal cycles are used to determine the causality relationships between all signal transitions in the circuit. Once these relationships are known, the circuit is then modeled as an *extended event-rule system*, which can be used to describe many circuits, including ones that are *inherently disjunctive*. An accurate indication of the performance of the circuit is obtained by analytically computing the *period* of the corresponding extended event-rule system.

Contents

List of Figures

Chapter 1

Introduction

1.1 Asynchronous VLSI Circuits

Asynchronous VLSI circuits are those that do not use global clocks. Instead, synchronization among components is achieved through the generation and detection of request and acknowledgement signals. Asynchronous circuits have many advantages over traditional synchronous systems [41, 30]. Besides the elimination of the clock skew and synchronization failure problems [34], asynchronous circuits also are more tolerant to variations in physical parameters, can be more easily synthesized using systematic and modular approaches [31], have a higher potential for low-energy computation [42], and yield average-case instead of worst-case performance [24].

The concept of asynchronous circuits has been around since the fifties [18]. However, it has not gained popularity until recently because of the difficulties involved in removing hazards from early designs [44]. Since then, several methodologies that generate functional asynchronous circuits under various timing assumptions have been developed (for example, [10, 35, 12, 45, 39]). In particular, the Caltech approach, invented by A. J. Martin [30], has produced many successful CMOS circuits such as stacks, arbiters [27], routers, a $3x + 1$ special-purpose processor [19], a multiply-accumulator [40], a memory management unit [38], and, in 1988, the first asynchronous microprocessor [32]. The favorable statistics of the microprocessor [33] and its portability to gallium arsenide technology [43] have contributed to the renewed interest in asynchronous designs.

The Martin synthesis method (which will be outlined in Chapter 2) systematically transforms a high-level specification, through a series of semantics-preserving steps, into a network of circuit elements. By construction, the circuits produced by the method are hazard-free and operate correctly regardless of the delays in the elements and wires, provided delays along different branches of certain forks, known as *isochronic forks*, are negligible. Such a circuit is said to be *quasi-delay-insensitive (QDI)* [28]. As we shall see, if each branch of a *non-isochronic* fork is explicitly modeled by a "wire operator" with arbitrary delay, then a QDI circuit is equivalent to a *speed-independent* circuit [37], where the delays on elements are arbitrary and those on wires are negligible. Due to the weak assumption on delays, QDI circuits are very robust; of the designs mentioned above, those that were fabricated functioned correctly on "first silicon" and could be operated over wide ranges of supply voltages and temperatures. QDI circuits are also relatively easy to test as demonstrated in [16].

1.2 Performance of Asynchronous Circuits

Though the delays of the elements in a QDI circuit do not affect its functionality, they do have a direct bearing on the speed at which it operates. This thesis presents a method to evaluate and optimize the performance of a QDI circuit by finding appropriate sizes for its transistors. As explained below, the approach taken is fundamentally different from the one for synchronous circuits.

Given a synchronous circuit, the speed at which it operates depends mainly on its clock rate. Registers are used to save data from one clock period to the next and, as long as the sub-circuits between the registers can complete their computations faster than the clock allowance, the system operates successfully. Consequently, optimization of synchronous systems is achieved by selecting an appropriate placement of the registers [20] and limiting the delays needed to transfer and manipulate data from one register to the next [23]. The analysis is further simplified by the fact that most of these stages are purely combinational and that there is no feedback [22].

In a QDI circuit, however, the occurrences of signal transitions are not regulated by a clock. Instead, each signal transition occurs as soon as an appropriate set of other signal transitions — either produced internally or supplied by the environment — have occurred and a sufficient amount of delay has elapsed. Because of the absence of clocked registers to act as separators, each particular signal transition can have an effect, directly or indirectly, on many other signal transitions in the system. Therefore, evaluating the performance of a small group of elements in isolation, as is done for synchronous circuits, is no longer sufficient. Instead, in general, it is necessary to determine the causality and delay relationships between all signal transitions in an asynchronous circuit and its environment.

The first successful attempt to address this problem is by Burns in [6]. There, he develops the concept of *Event-Rule Systems (ER-systems)* where "events" represents occurrences of signal transitions and "rules" are used to describe their causality and delay relationships. For an ER-system that is *repetitive*, he is able to compute its *period* and shows that it is a good indicator of the performance of the underlying circuit.

Though ER-systems are very useful for representing and evaluating the performance of *conjunctive* asynchronous systems, they cannot describe *inherently disjunctive* systems where an event has more than one set of causes. Also, though Burns recognizes the need to simulate a *data-dependent* circuit to extract the causality relationships between its signal transitions, no explicit algorithm has been given on how to systematically transform such a circuit into a repetitive ER-system.

The purpose of this thesis is to address these two short-comings. First, *Extended ER-systems (XER-system)* are introduced and shown to retain many of the properties of ER-systems. In particular, we will demonstrate how to compute the *period* of an XER-system and how this value reflects its performance. Next, we will present an algorithm for converting any QDI circuit (without *arbiters*) into a repetitive XER-system. The algorithm makes use of *index-priority simulation* which guarantees that the XER-system it produces has the minimal number of transitions.

1.3 Outline of Thesis

This thesis is organized as follows:

- Chapter 1 serves as a general introduction.
- Chapter 2 gives a brief outline of Martin's synthesis method.

- Chapter 3 explains the limitations of ER-system in detail.
- Chapter 4 describes XER-systems and their properties.
- Chapter 5 presents the theoretical framework for analyzing the states of a QDI circuit.
- Chapter 6 describes "index-priority simulation" for finding the minimal periodic behavior of a QDI circuit.
- Chapter 7 explains how to use this behavior to convert a QDI circuit into an XER-system.
- Chapter 8 serves as a summary and points out some directions for future work.

1.4 Notation and Conventions

The order of precedence for logical operators, from highest to lowest, is *not* ($\neg$), *and* ($\wedge$), *or* ($\vee$), *implies* ($\Rightarrow$), and *if-and-only-if* ($\Leftrightarrow$). Set difference is denoted by "$\backslash$" and has the same precedence as union ($\cup$) and intersection ($\cap$). Set inclusion is denoted by "$\subseteq$" and proper set inclusion by "$\subset$." Also, "$\uplus$" is sometimes used to denote the union of two disjoint sets.

The existential quantification "there exists x and y, with $x < y$, such that $x + y = 5$" is expressed as

$$\exists x, y : x < y : x + y = 5.$$

The same convention holds for universal quantification ($\forall$). Also, sets are sometimes denoted with similar notation; for example, the set of perfect squares is

$$\{i : i \in \mathbb{Z} : i^2\}.$$

Finally, i, j, k, l, m and n are always integer variables unless stated otherwise.

Chapter 2

Compilation Method

In this chapter, we give a brief outline of Martin's synthesis method and show its application to two specific examples: a one-place buffer and a zero-checker. Interested readers should refer to [30] for more details. Also, it should be pointed out that the transformation steps described below can be bypassed by using a syntax-directed compiler [5], though the results are usually too large for practical use.

2.1 CSP

At the top-most level, the specification of the circuit to be synthesized is written as a concurrent program, using a language that is based on Hoare's model of *Communicating Sequential Processes (CSP)* [17]. A CSP program consists of one or more processes which operate in parallel and communicate with each other through *channels*. A channel connects two processes and the two ends of a channel are referred to as *ports*[1].

Example 2.1: Figure 2.1 shows a set of three processes. Each process contains an L port and an R port. Process $p[0]$ communicates with $p[1]$ through the channel $\langle p[0].R, p[1].L\rangle$ and so forth. □

The operations performed by a process are described in the following notation. An assignment of an expression e to a variable x is "$x := e$."

[1]In some of the larger designs, for efficiency reasons, the language has been extended to allow shared variables and buses.

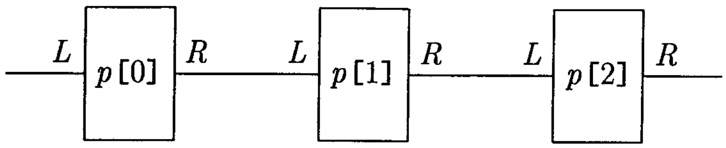

Figure 2.1: Communicating sequential processes

For brevity, if x is a Boolean variable, then "$x\uparrow$" and "$x\downarrow$" are equivalent to $x :=$ **true** and $x :=$ **false**, respectively.

A selection statement is of the form "$[G_0 \rightarrow S_0 \ [] \ \ldots \ [] \ G_n \rightarrow S_n]$." Each $G_j \rightarrow S_j$ is a *guarded command* [11] where G_j is a Boolean expression (the *guard* of the command) and S_j is a program part. The operational semantics of the selection statement is: "Wait until one of the G_j's is **true**, then non-deterministically choose a guarded command with a **true** guard and execute the corresponding program part." The notation "$[G]$" is a shorthand for "$[G \rightarrow$ **skip**$]$" and amounts to "wait until G is **true**."

A loop statement is of the form "$*[G_0 \rightarrow S_0 \ [] \ \ldots \ [] \ G_n \rightarrow S_n]$" and has the operational semantics "Choose a guarded command with a **true** guard, execute the corresponding program part, and repeat; if all G_j's are **false**, then exit loop." The notation "$*[S]$" is an abbreviation for "$*[$**true** $\rightarrow S]$" and means "execute S forever."

For a channel $\langle p.R, q.L\rangle$, "$R!e$" in process p denotes the communication action of sending the value of the expression e to the channel, and "$L?x$" in process q denotes the communication action of receiving the value from the channel and storing it in the variable x. Thus, the combined effect of the two statements is to assign to the variable x in q the value of e in p. Note that channels in CSP have no slack [25]: $R!e$ in p cannot complete and the process suspends unless q executes the corresponding $L?x$, and vice versa. Thus, dataless channels can be used to enforce synchronization between processes. A communication action on such a channel is expressed by naming the corresponding port. Also, the *probe* of a port L, denoted $\overline{L}$, is a Boolean value that is **true** only if the communication action L can be completed without suspension [29].

Finally, sequential composition is represented by ";" and concurrent composition — which is *weakly fair*, i.e., every non-terminating component is executed infinitely often — is represented by "$\|$." In addition, if A and B are two communication actions, then $A \bullet B$ is their *coincident execution* which means that A and B are to complete together [32].

Example 2.2: A one-place buffer that receives a data value (say, an n-bit integer) from a port named L and sends it to a port named R can be described as $*[L?x; R!x]$. □

2.1.1 Process Decomposition

The first step of the synthesis method is process decomposition whereby attempts are made to convert each process into smaller sub-processes and to extract, if possible, common program parts. Compiling smaller processes facilitates the rest of the synthesis procedure and sharing common program parts reduces the area of the final circuit. Also, parts of the program that cannot be compiled into stable production rules (see Section 2.3) are "factored out." These program parts deal with arbitration and synchronization of negated probes and are implemented directly as standard networks of transistors. The one-place buffer is already simple enough so that no process decomposition is necessary; see [26] for a larger example where this procedure is applied.

2.1.2 Separation of Control and Datapath

As we shall see, the datapath of a process can be implemented in a fairly standard way. In contrast, its control needs to be systematically transformed from one level of description to the next. Hence, for the next stage of the synthesis method, the datapath of a process is temporary removed and only the control part is compiled.

Example 2.3: After the removal of the datapath, the control for the one-place buffer is $*[L; R]$. □

2.2 Handshaking Expansion

The next step in the synthesis method represents each communication action with operations on Boolean variables. In order to maintain correctness, the two ends of a channel need to obey some given *protocol*. The two most

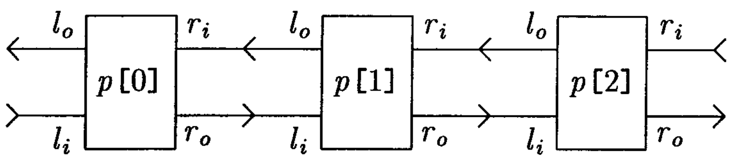

Figure 2.2: Handshaking variables

common protocols are *two-phase handshaking* and *four-phase handshaking*. Due to space limitation, only the latter will be discussed.

For the four-phase handshaking protocol, one communication action on a channel is chosen to be *active* and the corresponding one *passive*. If L is an active communication action, then it is transformed into[2]

$$l_o\uparrow;\ \ [l_i];\ \ l_o\downarrow;\ \ [\neg l_i]$$

and if R is a passive communication action, then it is transformed into

$$[r_i];\ \ r_o\uparrow;\ \ [\neg r_i];\ \ r_o\downarrow.$$

The channel $\langle p.L, q.R\rangle$ is represented by connecting the output variable of $p.L$ with the input variable of $q.R$, and vice versa. See Figure 2.2 for the transformation of Figure 2.1. In general, only passive communications can be probed and $\overline{R}$ is compiled into r_i.

Example 2.4: It turns out that it is easier to implement active input ports (data arrive only when requested) and, therefore, the communication action L in the one-place buffer is chosen to be active. Since we want to compose one instance of this buffer with another, the corresponding action R is required to be passive. The handshaking expansion for the active-passive buffer is

$$ap \;=\; *[\ l_o\uparrow;\ \ [l_i];\ \ l_o\downarrow;\ \ [\neg l_i];\ \ [r_i];\ \ r_o\uparrow;\ \ [\neg r_i];\ \ r_o\downarrow\ ].$$

□

2.2.1 Reshuffling

The last half of a four-phase handshaking protocol ($l_o\downarrow$; $[\neg l_i]$ or $[\neg r_i]; r_o\downarrow$) is not needed for synchronization and optimizations can often be made by

[2]By convention, input variables are subscripted with i and output variables with o.

postponing it or part of it. Such a procedure is known as *reshuffling*. However, care must be taken so that no deadlock ensues and data integrity is maintained (see Chapter 6 in [6]).

Example 2.5: In the one-place buffer, there is no need to wait for l_i to become **false** before starting the R communication. Hence, we can postpone the wait $[\neg l_i]$ so that it occurs as late as possible, i.e., just before $l_o\uparrow$. The resulting protocol for L becomes

$$[\neg l_i];\ l_o\uparrow;\ [l_i];\ l_o\downarrow$$

and is called the *lazy-active* protocol [6]. The lazy-active-passive buffer is

$$lap \ = \ *[\ [\neg l_i];\ l_o\uparrow;\ [l_i];\ l_o\downarrow;\ [r_i];\ r_o\uparrow;\ [\neg r_i];\ r_o\downarrow\ ],$$

and, in general, it outperforms the active-passive buffer due to the postponed wait. □

2.2.2 State Variable Insertion

If there are two states in a reshuffled handshaking expansion that are indistinguishable, then state variables need to be introduced to differentiate them.

Example 2.6: In *lap*, each variable in the state after $l_o\downarrow$ may have the same value as in the state after $r_o\downarrow$. Hence, a state variable is needed. We have chosen, among many others, the following state variable assignment:

$$lap' \ = \ *[\ [\neg l_i];\ l_o\uparrow;\ [l_i];\ s\uparrow;\ l_o\downarrow;\ [r_i];\ r_o\uparrow;\ [\neg r_i];\ s\downarrow;\ r_o\downarrow\ ].$$

□

2.3 Production Rules

Once a handshaking expansion with all states distinguishable has been obtained, the explicit sequential operators (the "semi-colons") are removed by transforming it into a set of *production rules*. A production rule (PR) is of the form "$G \to S$," where G, a Boolean expression, is its *guard* and S, an assignment of **true** or **false** to a Boolean variable, is its *assignment* or *output transition*. A PR is *enabled* if its guard is **true**. A PR *fires* when its assignment is executed — the firing is *effective* if it causes a state change; else, it is *vacuous*. The operational semantics of a PR is that it may fire if it is enabled. The operational semantics of a set of PR's is the weakly fair concurrent composition of the PR's in the set.

A PR is *stable* if once it is enabled, it remains enabled until it fires. For any variable x, a PR for $x\uparrow$ and a PR for $x\downarrow$ are *complementary* and two complementary PR's are *non-interfering* if they are never both enabled. Many of the later results rely on the following observation made by Martin [30]:

> Under the stability of each PR and non-interference among complementary PR's, the concurrent execution of the PR's of a set is equivalent to the following sequential execution:
>
> $$*[\ \textit{select a PR with a true guard; fire the PR}\ ]$$
>
> where the selection is weakly fair.

Ignoring arbiters and synchronizers, all PR's generated by the synthesis method satisfy the stability and non-interference conditions above. So, from now on, we will assume that there is only one PR for each transition since $G_0 \rightarrow x\uparrow$ and $G_1 \rightarrow x\uparrow$ can be replaced by $G_0 \vee G_1 \rightarrow x\uparrow$.

Example 2.7: The *lap* buffer is compiled into the following PR set:

$$\begin{array}{lcllcl}
\neg l_i \wedge \neg s \wedge \neg r_o & \rightarrow & l_o\uparrow & r_i \wedge s \wedge \neg l_o & \rightarrow & r_o\uparrow \\
l_i \wedge l_o & \rightarrow & s\uparrow & \neg r_i \wedge r_o & \rightarrow & s\downarrow \\
s & \rightarrow & l_o\downarrow & \neg s & \rightarrow & r_o\downarrow
\end{array}$$

The PR set is *minimal* in the sense that the set of the literals in each of the guards cannot be replaced by a smaller subset. □

Before continuing with the compilation method, several properties of PR sets need to be listed for future reference. A *self-invalidating* PR is one whose firing falsifies its own guard. An example of a self-invalidating PR is $\neg x \rightarrow x\uparrow$. Since they never occur in actual circuits, self-invalidating PR's will be disallowed in this paper.

Next, occasionally, it will be convenient to regard the guard of a PR as written in disjunctive-normal-form (DNF)

$$B_0 \vee B_1 \vee \ldots \vee B_m$$

with each B_i being a conjunction of the form

$$l_{i,0} \wedge l_{i,1} \wedge \ldots \wedge l_{i,n_i}$$

and each $l_{i,j}$ is a literal (a variable or its negation). B_i is said to contain a literal l if there exists j such that $l = l_{i,j}$. A *disjunct* of a PR refers to a conjunction in the DNF of the guard of the PR. The disjuncts of a PR are *mutually exclusive (mutex)* if at most one of them is **true** in any state. A disjunct B_i of a PR is a *stable disjunct* if, whenever B_i is **true**, it remains **true** until the PR fires. A PR set *has only stable disjuncts* if all disjuncts in all of its PR's are stable disjuncts.

2.3.1 Reset Signal

In order to put the circuit in the proper state upon power-up, a *reset* signal is added to the PR set.

Example 2.8: Adding the reset signal to the PR set above yields

$$\begin{array}{lcl}
\neg Reset \wedge \neg l_i \wedge \neg s \wedge \neg r_o & \rightarrow & l_o\uparrow \\
l_i \wedge l_o & \rightarrow & s\uparrow \\
Reset \vee s & \rightarrow & l_o\downarrow \\
r_i \wedge s \wedge \neg l_o & \rightarrow & r_o\uparrow \\
Reset \vee \neg r_i \wedge r_o & \rightarrow & s\downarrow \\
\neg s & \rightarrow & r_o\downarrow.
\end{array}$$

To avoid complicating the PR sets, in the sequel, the reset signal will no longer be explicitly mentioned. □

2.3.2 Symmetrization and Operator Reduction

The next step of the synthesis method forms the *operator* for each non-input variable y, by grouping together the two complementary PR's with output transitions on y. Occasionally, it is possible to change some of the guards to yield standard operators such as OR-gates and Muller C-elements [36] — this process is known as *symmetrization.*

Example 2.9: Consider the Q-element

$$*[\ [l_i];\ r_o\uparrow;\ [r_i];\ s\uparrow;\ r_o\downarrow;\ [\neg r_i];\ l_o\uparrow;\ [\neg l_i];\ s\downarrow;\ l_o\downarrow\ ].$$

If it is implemented as the following minimal PR set

$$\begin{array}{lcllcl} \neg s \wedge l_i & \rightarrow & r_o\uparrow & \qquad s \wedge \neg r_i & \rightarrow & l_o\uparrow \\ r_i & \rightarrow & s\uparrow & \qquad \neg l_i & \rightarrow & s\downarrow \\ s & \rightarrow & r_o\downarrow & \qquad \neg s & \rightarrow & l_o\downarrow, \end{array}$$

then the operator for r_o is state-holding. Note, however, that the PR set still implements the Q-element if the guard for $r_o\downarrow$ is *weakened* to $s \vee \neg l_i$. In this new PR set, the operator for r_o is a two-input AND-gate, which, in CMOS, can be more cheaply implemented than the original state-holding element. A similar improvement results from weakening the guard for $l_o\downarrow$ to $\neg s \vee r_i$. □

The actual implementation of the operators depends on the target technology; the rest of this chapter assumes that CMOS has been chosen.

2.3.3 Isochronic Forks and Bubble Shuffling

Given a PR set $\mathcal{P}$, let x be a variable appearing in either guard of the operator for y. Consider the new PR set $\widetilde{\mathcal{P}}$ obtained by adding the *wire* operator

$$x \;\rightarrow\; x'\uparrow \qquad\qquad \neg x \;\rightarrow\; x'\downarrow$$

to $\mathcal{P}$ and replacing each occurrence of x in the guards of the operator for y with x'. If, when x' is ignored, $\mathcal{P}$ and $\widetilde{\mathcal{P}}$ behave differently, then the variable x and operator for y are said to form an *isochronic branch.* Note that this situation implies that the delay modeled by the wire operator affects the functionality of $\mathcal{P}$. Hence, $\mathcal{P}$ operates as specified only if the delays on its isochronic branches are negligible. If variable x forms an isochronic branch with any operator, then x is an *isochronic fork.*

Example 2.10: Let $\langle x, y\rangle$ denote the branch from variable x to the operator for y. Then, in Example 2.7, $\langle s, l_o\rangle$ and $\langle s, r_o\rangle$ are non-isochronic branches and all other branches are isochronic. So, l_i, l_o, r_i, and r_o are isochronic forks. □

For CMOS implementation, a given PR set needs to be converted to one that is *CMOS-mappable (CM)* where the literals in the guards of the PR's with down-going transitions are positive and the literals in the guards of the PR's with up-going transitions are negative. To effect this transformation, inverters may be needed to produce x_-, the negative sense of a variable x. However, to avoid adding assumptions on the speed of these inverters, it is preferable that no inverter be added on any isochronic branch. This requirement, unfortunately, cannot always be satisfied. When this situation arises, one can either alter the original PR set or analyze the circuit to make as weak a timing assumption as possible.

Example 2.11: Consider the PR set of Example 2.7. A CM operator for l_o is

$$\neg l_i \wedge \neg s \wedge \neg r_o \quad \rightarrow \quad l_o\uparrow \qquad\qquad s \quad \rightarrow \quad l_o\downarrow.$$

For s, there are two choices:

$$\neg l_{i-} \wedge \neg l_{o-} \quad \rightarrow \quad s\uparrow \qquad\qquad r_{i-} \wedge r_o \quad \rightarrow \quad s\downarrow$$

or

$$l_i \wedge l_o \quad \rightarrow \quad s_-\downarrow \qquad\qquad \neg r_i \wedge \neg r_{o-} \quad \rightarrow \quad s_-\uparrow.$$

The first choice requires an inverter on the branch $\langle l_i, l_o\rangle$ or $\langle l_{i-}, s\rangle$; the second requires an inverter on $\langle r_o, l_o\rangle$ or $\langle r_{o-}, s\rangle$. Similar observations hold if we had chosen a CM operator for l_{o-} instead of l_o. Thus, in order to transform the PR set into a CM one, an inverter needs to be placed on one of the isochronic branches.

Note, however, that the original handshaking expansion, *lap*, can also be implemented by

$$\begin{array}{lcl@{\qquad\qquad}lcl}
\neg l_i \wedge \neg s \wedge \neg r_o & \rightarrow & l_o\uparrow & r_i \wedge s \wedge \neg l_o & \rightarrow & r_o\uparrow \\
l_i \wedge l_o & \rightarrow & s\uparrow & \neg r_i \wedge r_o & \rightarrow & s\downarrow \\
l_i \wedge s & \rightarrow & l_o\downarrow & \neg r_i \wedge \neg s & \rightarrow & r_o\downarrow
\end{array}$$

where two of the guards (for $l_o\downarrow$ and $r_o\downarrow$) have been *strengthened.* The branches $\langle l_i, l_o\rangle$ and $\langle r_i, r_o\rangle$ are now non-isochronic. This new PR set can be transformed into the following CM one without introducing any inverters on isochronic branches:

$$
\begin{array}{lll@{\qquad}lll}
\neg l_i & \rightarrow & l_{i-}\uparrow & \neg r_{i-} \wedge \neg s_- \wedge \neg l_o & \rightarrow & \_r_o\uparrow \\
l_{i-} \wedge s_- \wedge r_{o-} & \rightarrow & l_{o-}\downarrow & \_r_o & \rightarrow & r_{o-}\downarrow \\
\neg l_{o-} & \rightarrow & l_o\uparrow & \neg r_{o-} & \rightarrow & r_o\uparrow \\
l_i \wedge l_o & \rightarrow & s_-\downarrow & \neg r_i \wedge \neg r_{o-} & \rightarrow & s_-\uparrow \\
l_i & \rightarrow & l_{i-}\downarrow & \neg r_i & \rightarrow & r_{i-}\uparrow \\
\neg l_{i-} \wedge \neg s_- & \rightarrow & l_{o-}\uparrow & r_{i-} \wedge s_- & \rightarrow & \_r_o\downarrow \\
l_{o-} & \rightarrow & l_o\downarrow & \neg\_r_o & \rightarrow & r_{o-}\uparrow \\
r_i & \rightarrow & r_{i-}\downarrow & r_{o-} & \rightarrow & r_o\downarrow.
\end{array}
$$

The signal $\_r_o$ is an internal version of r_o; $\_r_o$ is needed to generate r_{o-} which, in turn, generates r_o, the output to the environment. □

As can be seen by the previous example, a CM PR set can be significantly larger than its counterpart before bubble shuffling. Hence, for conciseness, PR sets that are not CM will be used for illustrative purposes, even though it should be pointed out that transistor sizing, as described in the next section, is relevant only for CM PR sets.

2.4 CMOS Circuit

Each CM operator for y is realized as a CMOS element so that there is a connecting path from *VDD* to the output when the guard for $y\uparrow$ is **true** and one from *GND* to the output if the guard for $y\downarrow$ is **true**. A staticizer is added on the output if it is needed to maintain the charge when both guards are **false**. Due to electrical considerations, there is an upper limit on how many transistors can be in series on each connecting path; this limit translates into a bound on the number of literals in each conjunction of the guards.

Example 2.12: The transistor network for the previous example, without staticizers, is shown in Figure 2.3, where every four-way intersection represents an overlap and not a connection. □

2.4.1 Transistor Sizing

For a CMOS element, the time it takes for its output node to change value once a connecting path to *VDD* or *GND* is established depends, to a large degree, on R, the effective resistance of the transistors in the path, and C,

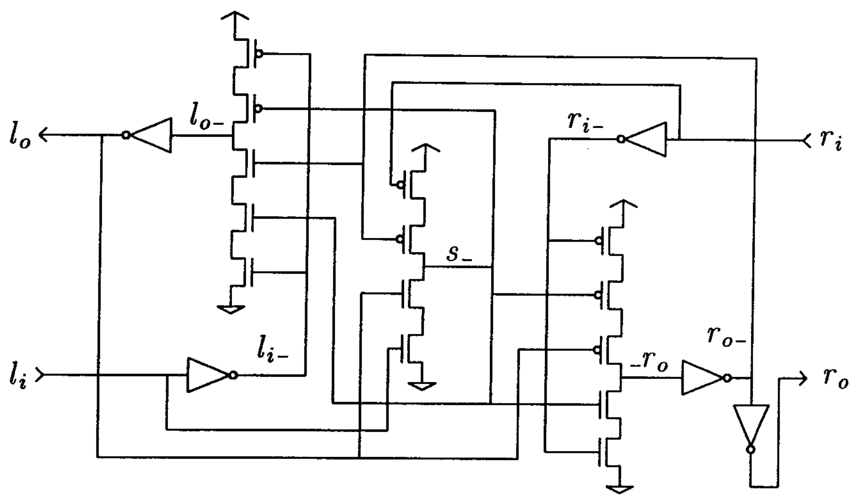

Figure 2.3: CMOS implementation

the capacitive load on the output node. Both R and C are functions of the transistor sizes in the network, plus parasitic wiring capacitances. One of the goals of this paper is to develop a method to determine the appropriate transistor sizes so that the network can operate at optimal speed. Because of the close correspondence between a CM PR set and the CMOS circuit that implements it, the analysis will be performed on the former using the implicit assumption that the delays between occurrences of transitions can be expressed as functions of the appropriate transistor sizes. Some schemes for computing these delays are given in [13, 21, 7, 4, 9, 8].

2.5 Datapaths

In contrast to the control part, the datapath of a process can usually be implemented efficiently by combining members from a standard set of components such as registers, adders, completion trees, etc. This section describes the nature of some of these components and how they can be put together to form a datapath.

2.5.1 Registers

The input part of a binary register, *ireg* $= *[P?x]$, can be implemented by the handshaking expansion

$$
\begin{array}{l}
*[[\ pT_i \longrightarrow x\uparrow;\ p_o\uparrow;\ [\neg pT_i];\ p_o\downarrow \\
\ \ []\ \ pF_i \longrightarrow x\downarrow;\ p_o\uparrow;\ [\neg pF_i];\ p_o\downarrow \\
\ \ ]].
\end{array}
$$

Note that a Boolean value is communicated in a delay-insensitive manner by using *dual-rail encoding* where one *data signal* (pT_i) is raised if a 1 is sent, and a different data signal (pF_i) is raised if a 0 is sent. The raised signal is then lowered during the second half of the handshaking protocol. A dual-rail port is said to have a *valid* value if exactly one of the two data signals is high; it has a a *neutral* value if both signals are low [24]. For other encoding schemes, see [2, 46].

As written above, *ireg* cannot be implemented without adding an inverter on an isochronic branch. However, this problem can be removed if x is stored in true-complement form as shown below:

$$
\begin{array}{l}
*[[\ pT_i \longrightarrow x0\downarrow;\ x1\uparrow;\ p_o\uparrow;\ [\neg pT_i];\ p_o\downarrow \\
\ \ []\ \ pF_i \longrightarrow x1\downarrow;\ x0\uparrow;\ p_o\uparrow;\ [\neg pF_i];\ p_o\downarrow \\
\ \ ]].
\end{array}
$$

The corresponding PR set is

$$
\begin{array}{lcl@{\qquad}lcl}
pT_i & \rightarrow & x0\downarrow & pT_i \wedge x1 \vee pF_i \wedge x0 & \rightarrow & p_{o\text{-}}\downarrow \\
\neg x0 \wedge \neg pF_i & \rightarrow & x1\uparrow & \neg p_{o\text{-}} & \rightarrow & p_o\uparrow \\
pF_i & \rightarrow & x1\downarrow & \neg pT_i \wedge \neg pF_i & \rightarrow & p_{o\text{-}}\uparrow \\
\neg x1 \wedge \neg pT_i & \rightarrow & x0\uparrow & p_{o\text{-}} & \rightarrow & p_o\downarrow.
\end{array}
$$

Analogously, the output part of a binary register, *oreg* $= *[Q!x]$, can be written as

$$
\begin{array}{l}
*[[\ q_i \wedge x1 \longrightarrow qT_o\uparrow;\ [\neg q_i];\ qT_o\downarrow \\
\ \ []\ \ q_i \wedge x0 \longrightarrow qF_o\uparrow;\ [\neg q_i];\ qF_o\downarrow \\
\ \ ]]
\end{array}
$$

which is compiled into

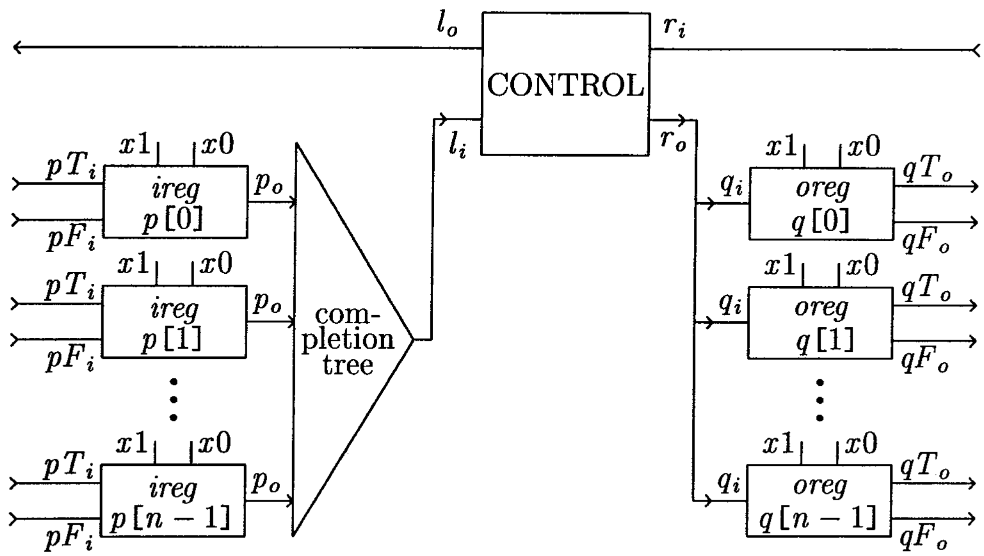

Figure 2.4: Transferring data

$$
\begin{array}{llllll}
q_i \wedge x1 & \to & qT_{o\text{-}}\downarrow & \qquad q_i \wedge x0 & \to & qF_{o\text{-}}\downarrow \\
\neg qT_{o\text{-}} & \to & qT_o\uparrow & \qquad \neg qF_{o\text{-}} & \to & qF_o\uparrow \\
\neg q_i & \to & qT_{o\text{-}}\uparrow & \qquad \neg q_i & \to & qF_{o\text{-}}\uparrow \\
qT_{o\text{-}} & \to & qT_o\downarrow & \qquad qF_{o\text{-}} & \to & qF_o\downarrow .
\end{array}
$$

2.5.2 Completion Trees

In *ireg*, a completion signal p_o is generated after the bit has been stored. If n registers of this form are used to store an n-bit integer, then the completion signals from these n registers need to be combined to form a single acknowledgement signal. This task is accomplished by the C-element

$$
\begin{array}{lll}
p\,[0].p_o \wedge p\,[1].p_o \wedge \ldots \wedge p\,[n-1].p_o & \to & p_o\uparrow \\
\neg p\,[0].p_o \wedge \neg p\,[1].p_o \wedge \ldots \wedge \neg p\,[n-1].p_o & \to & p_o\downarrow
\end{array}
$$

where $p\,[j].p_o$ is the completion signal from bit register $p\,[j]$ and p_o is the combined completion signal. If n is large, this n-input C-element can be implemented by a tree of C-elements with fewer inputs; hence, the term *completion tree* is used.

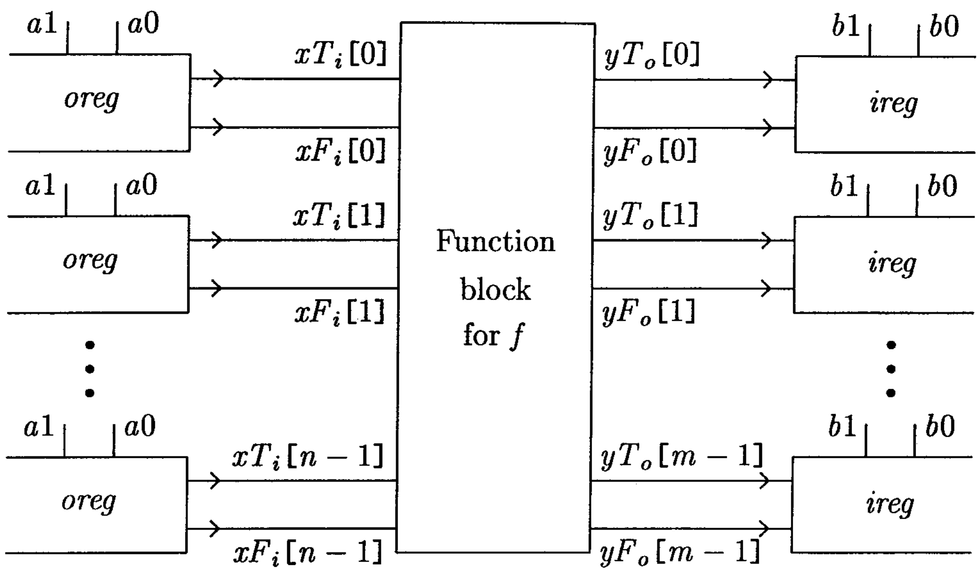

Figure 2.5: Implementation for $\mathbf{b} := f(\mathbf{a})$

2.5.3 Register Transfers

Figure 2.4 illustrates the most common scheme for transferring data. On the right side of the figure, where data are sent using a passive output communication on port R, the output signal of the control part, r_o, is used to cause the registers to send out their values in dual-rail form. And, as shown on the left side of the figure, for an active input communication on port L, data are latched into registers and a combined completion signal is then generated to serve as l_i, the acknowledgement for the control part.

Example 2.13: The schematics for the one-place buffer $*[L?x; R!x]$ is shown in Figure 2.4 where the control part is the network of Figure 2.3 and, for each j, $0 \leq j < n$, $p[j].x0$ is connected to $q[j].x0$ and $p[j].x1$ is connected to $q[j].x1$. $\square$

2.5.4 Function Blocks

A *function block* for a given function f repeatedly accepts an argument and produces its image under f. The assignment $\mathbf{b} := f(\mathbf{a})$, where $\mathbf{a}$ and $\mathbf{b}$

are lists of registers, is usually implemented by sending the values of **a** to a function block for f and storing the result in **b**, as Figure 2.5 illustrates[3]. A possible protocol [24] for a function block is

$$*[\ [v(X)];\ \ Y\Uparrow;\ \ [n(X)];\ \ Y\Downarrow\ ]$$

where $v(X)$ means that input port X has a *valid* value, $n(X)$ means that X has a *neutral* value, $Y\Uparrow$ means that output port Y is set to a valid value, and $Y\Downarrow$ means that Y is reset to a neutral value. Using dual-rail encoding, a non-binary port has a valid (or, alternatively, neutral) value when all the ports for communicating the constituent bits have valid (neutral) values. For the implementation of an adder function block, the reader is referred to [24].

2.5.5 Zero-Checkers

The zero-checker is an interesting function block in that it illustrates the use of guards with disjuncts that are not mutex. Its input is an n-bit integer X and its output is a Boolean variable Y that is **false** if and only if X is identically zero. An obvious approach is to implement the n-bit zero-checker as a tree of zero-checkers of smaller size. For example, one possible implementation of a two-bit zero-checker is *zeroA* below:

$$\begin{array}{lcl}
aT_i \wedge bT_i \vee aT_i \wedge bF_i \vee aF_i \wedge bT_i & \rightarrow & cT_o\uparrow \\
aF_i \wedge bF_i & \rightarrow & cF_o\uparrow \\
\neg aT_i \wedge \neg aF_i \wedge \neg bT_i \wedge \neg bF_i & \rightarrow & cT_o\downarrow \\
\neg aF_i \wedge \neg bF_i & \rightarrow & cF_o\downarrow
\end{array}$$

where aT_i and aF_i are the dual-rail signals of one of the input bits, bT_i and bF_i are those for the other bit, and cT_o and cF_o are those of the output. Note that all disjuncts in the guards are mutex. Also, the three disjuncts in the guard for $cT_o\uparrow$ are necessary in order to test for the validity of the input. Similarly, if cT_o is ever raised, one needs to make sure that the appropriate input signals have been reset before lowering cT_o.

An alternative implementation of a two-bit zero-checker is to perform the input validity and neutrality checks explicitly as done in *zeroB* below:

[3] At times, performance may be improved if the function block is combined with some of the registers; to simplify the presentation, no such optimization is applied.

$$
\begin{array}{lll}
aT_i \vee aF_i & \rightarrow & a\uparrow \\
bT_i \vee bF_i & \rightarrow & b\uparrow \\
a \wedge b & \rightarrow & g\uparrow
\end{array}
\qquad
\begin{array}{lll}
\neg aT_i \wedge \neg aF_i & \rightarrow & a\downarrow \\
\neg bT_i \wedge \neg bF_i & \rightarrow & b\downarrow \\
\neg a \wedge \neg b & \rightarrow & g\downarrow
\end{array}
$$

$$
\begin{array}{lll}
g \wedge (aT_i \vee bT_i) & \rightarrow & cT_o\uparrow \\
g \wedge (aF_i \wedge bF_i) & \rightarrow & cF_o\uparrow
\end{array}
\qquad
\begin{array}{lll}
\neg g & \rightarrow & cT_o\downarrow \\
\neg g & \rightarrow & cF_o\downarrow .
\end{array}
$$

Note that the disjuncts in the guard for $cT_o\uparrow$ are no longer mutex since both aT_i and bT_i may be **true**. Even though *zeroA* has 14 literals in the guards versus 20 for *zeroB*, all guards in the latter have no more than three literals in any conjunction, most of them have fewer. Furthermore, a direct extension of *zeroA* to four bits requires 76 literals, 8 of which belonging to one conjunction, whereas similar extension to *zeroB* yields 36 literals, at most 5 in a conjunction. If the four-bit zero-checker is implemented by a binary tree of three two-bit zero-checkers of type *zeroA*, then 42 literals, at most 4 in a conjunction, are needed. Consequently, since each literal corresponds to a transistor, on the bases of area and power consumption, the best way to implement a four-bit zero-checker is with the extension to *zeroB*. Thus, it is sometimes profitable to have guards with disjuncts that are not mutex.

2.5.6 Quick-Decision Zero-Checkers

Even if an input bit has a valid non-zero value, a zero-checker of the previous sub-section still waits for all other input input bits to have valid values before issuing **true** as output. There are practical situations where this wait is a serious drawback. For instance, for a memory access, one may need to add a base address *base* to an offset *offset* and compare the sum to *tag*, the tag of a cache line. Using bit-wise exclusive-or ($\veebar$) as the comparison operator, a cache miss occurs if $(base+offset)\veebar tag$ is non-zero. If the adder of [24] is used, then there is a variance in the times at which the bits of the above expression become valid, due to the rippling effect of the carry-chains. Since a fetch from main memory upon a cache miss should be initiated as soon as possible, it is highly advantageous to use a *quick-decision* zero-checker that raises a signal whenever *one* of the input bits is non-zero, without waiting for the other bits to become valid. In order to satisfy the delay-insensitive protocol, one still needs to wait for the validity of all input bits; however, this wait can now proceed concurrently with operations that would otherwise be postponed.

The PR set for a two-bit quick-decision zero-checker is *zeroQ* below:

$$
\begin{array}{lcl@{\qquad\qquad}lcl}
aT_i \vee aF_i & \rightarrow & a\uparrow & \neg aT_i \wedge \neg aF_i & \rightarrow & a\downarrow \\
bT_i \vee bF_i & \rightarrow & b\uparrow & \neg bT_i \wedge \neg bF_i & \rightarrow & b\downarrow \\
a \wedge b & \rightarrow & g\uparrow & \neg a \wedge \neg b & \rightarrow & g\downarrow \\
\\
aQ_i \vee bQ_i & \rightarrow & cQ_o\uparrow & \neg aQ_i \wedge \neg bQ_i & \rightarrow & cQ_o\downarrow \\
g \wedge cQ_o & \rightarrow & cT_o\uparrow & \neg g \wedge \neg cQ_o & \rightarrow & cT_o\downarrow \\
g \wedge aF_i \wedge bF_i & \rightarrow & cF_o\uparrow & \neg g & \rightarrow & cF_o\downarrow.
\end{array}
$$

Each data bit is now represented by three signals: cQ_o is **true** if any of the input bits is non-zero; cT_o is **true** if any of the input bits is non-zero and they are all valid; and cF_o is **true** if all input bits are zeros. Similar conventions hold for the input signals if they are generated from another quick-decision zero-checker. If the input is from a binary register, then aQ_i is the same signal as aT_i.

Note that, in a tree of *zeroQ*'s, the latency between one of the input bits assuming a non-zero value and the final cQ_o becoming **true** is simply the delay of a tree of OR-gates. So, *zeroQ* serves as another example of a useful PR set containing a guard with disjuncts (namely, aQ_i and bQ_i) that are not mutex.

Chapter 3

Event-Rule Systems

We begin this chapter by describing the Event-Rule Systems (ER-systems) invented by Burns [6] and showing how they can be used to model simple systems. We then point out some of the difficulties involved when ER-systems are used to model *data-dependent* systems or ones with *multiple-occurrences*. Finally, we will argue the need of a new abstraction to describe systems that are *inherently disjunctive*.

3.1 Event-Rule Systems

Definition: A *(general) event-rule system (ER-system)* is a pair $\mathcal{Y} = \langle E, R \rangle$ where

- E is a (possibly infinite) set of *events*; and
- R is a (possibly infinite) set of *rules* where each rule r is a triple $\langle e, f, \alpha \rangle$, written as $e \overset{\alpha}{\mapsto} f$, and $e \in E$ is the *source* of r, $f \in E$ is the *target* of r, and $\alpha \in [0, \infty)$ is the *delay* of r.

Definition: A *timing function* for an ER-system $\mathcal{Y} = \langle E, R \rangle$ is a function t from E to $[0, \infty)$ such that

$$\forall e, f, \alpha : e \overset{\alpha}{\mapsto} f \in R : t(f) \geq t(e) + \alpha. \tag{3.1}$$

Intuitively, $t(f)$ is the time at which event f occurs. If we let

$$C = \{e : (\exists\alpha :: e \overset{\alpha}{\mapsto} f \in R) : e\},$$

then (3.1) specifies that f cannot occur until every event in C has occurred; hence, C will be referred to, in this paper, as the *cause set* of f.

For periodic systems, the concept of *repetitive ER-systems* is introduced.

Definition: A *repetitive ER-system* is a pair $\mathcal{Y}' = \langle E', R' \rangle$ where

- E' is a finite set of *transitions*; and
- R' is a finite set of *rule templates* where each rule template r' is a tuple $\langle u, v, \alpha, \varepsilon \rangle$ and $R' \subseteq E' \times E' \times [0, \infty) \times \mathbb{Z}$.

Each repetitive ER-system $\mathcal{Y}' = \langle E', R' \rangle$ induces a general ER-system $\mathcal{Y} = \langle E, R \rangle$ where

- $E = E' \times \mathbb{N}$; and
- $R = \{u, v, i, \alpha, \varepsilon : \langle u, v, \alpha, \varepsilon \rangle \in R' \wedge i \geq \mathbf{max}\{0, \varepsilon\} : \langle u, i - \varepsilon \rangle \overset{\alpha}{\mapsto} \langle v, i \rangle\}$.

Thus, each event is an indexed occurrence of a transition. Furthermore, the rule template $r' = \langle u, v, \alpha, \varepsilon \rangle$ specifies that every occurrence of v has an occurrence of u in its cause set, with ε as a constant *occurrence-index offset* between the two transitions.

Many of the properties and definitions associated with ER-systems will be extended in the next chapter and their presentation will be postponed until then. For the rest of this chapter, we will address the issues involved in using ER-systems to model PR sets.

3.2 Closed Systems

ER-systems are used to model *closed* systems, i.e., ones where every signal is both an input and an output. For a PR set, this requirement implies that the behavior of the environment needs to be included (see next section).

3.3 Simple Straightline Programs

Without symmetrization, the guards of a PR set derived from a simple straightline program — a non-terminating repetition of a fixed sequence of different waits and transitions — are conjunctions. Hence, the conversion of this PR set to a repetitive ER-system is straightforward as the following example illustrates. Furthermore, a literal added by weakening a guard is always **false** when the PR fires; hence, this literal can be ignored since it does not introduce any new causality relationship.

Example 3.1: The most liberal environment for the Q-element $*[[l_i];\ r_o\uparrow;\ [r_i];\ s\uparrow;\ r_o\downarrow;\ [\neg r_i];\ l_o\uparrow;\ [\neg l_i];\ s\downarrow;\ l_o\downarrow]$ is

$$\begin{array}{lcl@{\qquad\qquad}lcl} \neg l_o & \rightarrow & l_i\uparrow & r_o & \rightarrow & r_i\uparrow \\ l_o & \rightarrow & l_i\downarrow & \neg r_o & \rightarrow & r_i\downarrow. \end{array}$$

The combination of this PR set and the unsymmetrized PR set in Example 2.9 is modeled by the repetitive ER-system $\mathcal{Y}' = \langle E', R' \rangle$ where

- $E' = \{l_i\uparrow, l_i\downarrow, l_o\uparrow, l_o\downarrow, r_i\uparrow, r_i\downarrow, r_o\uparrow, r_o\downarrow, s\uparrow, s\downarrow\}$; and
- $$\begin{array}{llll} R' = \{ & \langle l_i\uparrow, r_o\uparrow, \alpha_0, 0\rangle, & \langle s\downarrow, r_o\uparrow, \alpha_1, 1\rangle, & \langle r_i\uparrow, s\uparrow, \alpha_2, 0\rangle, \\ & \langle s\uparrow, r_o\downarrow, \alpha_3, 0\rangle, & \langle r_i\downarrow, l_o\uparrow, \alpha_4, 0\rangle, & \langle s\uparrow, l_o\uparrow, \alpha_5, 0\rangle, \\ & \langle l_i\downarrow, s\downarrow, \alpha_6, 0\rangle, & \langle s\downarrow, l_o\downarrow, \alpha_7, 0\rangle, & \langle l_o\downarrow, l_i\uparrow, \alpha_8, 1\rangle, \\ & \langle l_o\uparrow, l_i\downarrow, \alpha_9, 0\rangle, & \langle r_o\uparrow, r_i\uparrow, \alpha_{10}, 0\rangle, & \langle r_o\downarrow, r_i\downarrow, \alpha_{11}, 0\rangle \ \} \end{array}$$

with α's being the delays prescribed by the timing model under use. (For CM PR sets, these delays may be functions of transistor sizes.) Note that the second occurrence of $r_o\uparrow$ is caused by the first occurrence of $s\downarrow$ and so forth. Hence, an occurrence-index offset of 1 is needed in the rule template for the two transitions. The same observation holds for $\langle l_o\downarrow, l_i\uparrow, \alpha_8, 1\rangle$. Note also that the same ER-system is used even if the guard of $r_o\downarrow$ is symmetrized to $\neg l_i \vee s$ since l_i is **true** whenever $r_o\downarrow$ occurs and, so, $l_i\downarrow$ is not a cause of $r_o\downarrow$. □

3.4 Multiple Occurrences

Consider the toggle $*[X; X;\ Y]$. A handshaking expansion for it is

$$*[[x_i];\ x_o\uparrow;\ [\neg x_i];\ u\uparrow;\ v\uparrow;\ x_o\downarrow;\ [x_i];\ u\downarrow;\ x_o\uparrow;$$
$$[\neg x_i];\ y_o\uparrow;\ x_o\downarrow;\ [y_i];\ v\downarrow;\ y_o\downarrow;\ [\neg y_i]\ ].$$

A possible PR rule for $x_o\uparrow$ is

$$x_i \wedge \neg y_i \wedge \neg v \vee \neg y_o \wedge \neg u \wedge v \quad \rightarrow \quad x_o\uparrow.$$

One of the requirement of a repetitive ER-system is that the i-th occurrence of a transition t must be caused by the $(i - \varepsilon)$-th occurrence of another transition s, where ε is independent of i. In this example, $x_o\uparrow$ occurs twice for each occurrence of $y_i\downarrow$ and, therefore, the system cannot be modeled directly as a repetitive ER-system.

The remedy is obvious. For now, let us define a *cycle* as a sequence of transitions whose occurrences return the system back to the state it was before these occurrences have taken place. So, in this example, a cycle is

$$\langle x_i\uparrow, x_o\uparrow, x_i\downarrow, u\uparrow, v\uparrow, x_o\downarrow, x_i\uparrow, u\downarrow, x_o\uparrow, x_i\downarrow, y_o\uparrow, x_o\downarrow, y_i\uparrow, v\downarrow, y_o\downarrow, y_i\downarrow\rangle.$$

It is then sufficient to rename the transitions so that each transition in a cycle has a distinct name. For the toggle above, let each odd occurrence of $x_o\uparrow$ be renamed $x1_o\uparrow$ and each even occurrence $x2_o\uparrow$. Then, the PR above can be written without a disjunction as

$$x_i \wedge \neg y_i \wedge \neg v \quad \rightarrow \quad x1_o\uparrow \qquad\qquad \neg y_o \wedge \neg u \wedge v \quad \rightarrow \quad x2_o\uparrow.$$

Moreover, the i-th occurrence of $x1_o\uparrow$ is caused (in part) by the $(i - 1)$-th occurrence of $y_i\downarrow$, etc. Hence, the system can be modeled as a repetitive ER-system.

For this simple example, finding a cycle and determining how many times each transition occurs is relatively easy. However, for general cases involving programs with vacuous firings or initial transient behavior, such a task is no longer trivial. The situation becomes even more complicated when there are data-dependencies as described below.

3.5 Data-Dependent Systems

Consider the following PR from *ireg* of Sub-section 2.5.1:

$$pT_i \wedge x1 \vee pF_i \wedge x0 \quad \rightarrow \quad p_o\text{-}\downarrow.$$

Suppose a fixed environment which never sets both pTi and pFi **true** simultaneously has been included. Then, a particular occurrence of $p_{o}\text{-}\downarrow$ is caused by the occurrences of either $pT_i\uparrow$ and $x1\uparrow$ or by the occurrences of $pF_i\uparrow$ and $x0\uparrow$ but never both alternatives. Thus, the PR set can be modeled by a repetitive ER-system *provided that we know which occurrences of* $p_{o}\text{-}\downarrow$ *are caused by occurrences of* $pT_i\uparrow$ *and* $x1\uparrow$ *and which other occurrences of* $p_{o}\text{-}\downarrow$ *are caused by occurrences of* $pF_i\uparrow$ *and* $x0\uparrow$. As another example of complications that arise from data-dependencies, the PR for $x0\downarrow$ in *ireg* is

$$pT_i \quad \rightarrow \quad x0\downarrow.$$

However, depending on the sequence of actions issued by the environment, $x0$ may already be **false** when a particular occurrence of $pT_i\uparrow$ takes place. Hence, this occurrence does not cause any real occurrence of $x0\downarrow$ in spite of the PR above.

In general, the causality relationships between transitions in a PR set can be ascertained only through simulation. However, exhaustive simulation is usually too costly and, as the following example shows, depth-first simulation may result in a cycle that contains more transitions than necessary. Since the complexity in finding the performance of the corresponding repetitive ER-system increases dramatically with the length of this cycle, in Chapter 6, we have developed a simple algorithm which guarantees that the cycles it finds are "minimal." To establish this algorithm, the theoretical background on the properties of cycles will be presented in Chapter 5.

Example 3.2: Consider the following PR set:

$$\begin{array}{lcl}
\neg a3 \wedge \neg c & \rightarrow & a1\uparrow \\
a1 \wedge \neg a3 & \rightarrow & a2\uparrow \\
a2 \wedge (\neg b3 \vee b2 \vee c) & \rightarrow & a3\uparrow \\
a3 \wedge c & \rightarrow & a2\downarrow \\
\neg a2 \wedge a3 & \rightarrow & a1\downarrow \\
\neg a1 \wedge (b3 \vee \neg b1 \vee \neg c) & \rightarrow & a3\downarrow \\
\neg b3 \wedge \neg c & \rightarrow & b1\uparrow \\
b1 \wedge \neg b3 & \rightarrow & b2\uparrow \\
b2 \wedge (\neg a3 \vee a1 \vee c) & \rightarrow & b3\uparrow \\
b3 \wedge c & \rightarrow & b2\downarrow \\
\neg b2 \wedge b3 & \rightarrow & b1\downarrow \\
\neg b1 \wedge (a3 \vee \neg a2 \vee \neg c) & \rightarrow & b3\downarrow
\end{array}$$

$$
\begin{array}{lcl}
a2 \wedge b2 & \rightarrow & c\uparrow \\
\neg a1 \wedge \neg b1 & \rightarrow & c\downarrow .
\end{array}
$$

Its state graph is shown in Figure 3.1. Each circle represents a state and is labeled with three digits ABC where A is the value of the binary vector $\langle a3\ \ a2\ \ a1\rangle$, B is the value of $\langle b3\ \ b2\ \ b1\rangle$, and C is the binary value of c. Each edge represents the transition that causes the corresponding state change. To avoid cluttering up the graph, only some of the edges are labeled. It can be shown that the graph contains all states reachable from the initial state (marked 000) where every variable is **false**. Furthermore, no two circles in the graph represent the same state.

Now, in depth-first simulation, at every state, the most recently enabled transition is fired and the algorithm proceeds to the new state that results from that firing. When the new state is one that has been encountered before, the algorithm terminates and reports the cycle found. So, starting at the initial state 000, the bold edges in Figure 3.1 shows a possible cycle found by depth-first simulation. Note that all states on the cycle are different and yet it should be clear that there is a smaller cycle, half as long, that contains the same set of transitions. □

3.5.1 Environmental Scenarios

As illustrated by the discussion on *ireg* above, the causality relationships between occurrences in a data-dependent system depend on the particular choices made by the environment. Hence, in general, one can evaluate the performance of such a system only under a fixed environmental scenario. To get a better indication of the overall performance of the system, one can combine the performance evaluations from different environmental scenarios by weighting each according to its probability. As we shall see, the performance of a circuit can be expressed as a function of its transistor sizes; hence, by optimizing the combination of these functions from different scenarios, appropriate transistor sizes can be determined. Furthermore, by adding constraints — such as one requiring that the transistors used in setting a bit **true** have the same sizes as those for setting it **false** — it is possible to obtain proper sizes even for transistors that are not exercised under a given scenario. Consequently, in the sequel, it will be assumed that a "typical" environmental scenario has been selected for the PR set under consideration.

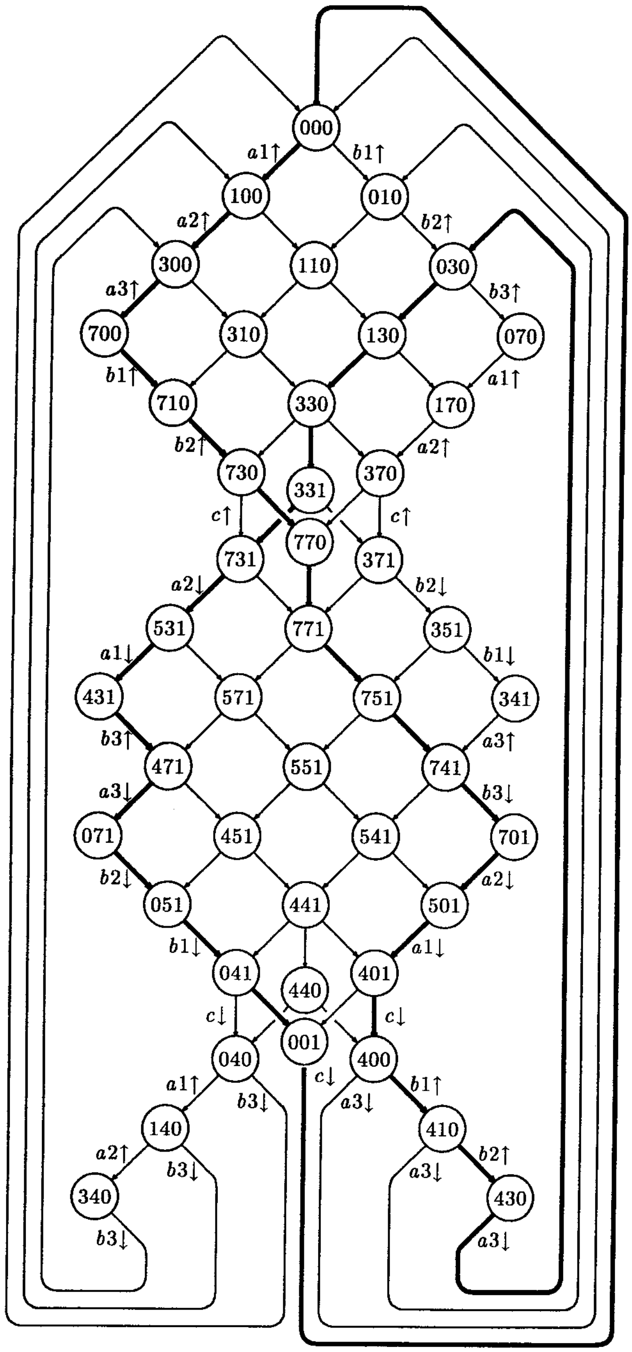

Figure 3.1: Depth-first simulation

3.6 Inherently Disjunctive Systems

Consider *zeroQ*, the quick-decision zero-checker from Sub-section 2.5.6. If the environment sets both aQ_i and bQ_i to **true**, then the subsequent occurrence of $cQ_o\uparrow$ is caused by either $aQ_i\uparrow$, or $bQ_i\uparrow$, or both, *depending on the delays between these transitions and* $cQ_o\uparrow$. Note that this is a fundamentally different situation from a data-dependent system, where every occurrence has a unique set of causes independent of the delays. Here, an event has more than one set of causes and, consequently, cannot be modeled by an ER-system. Such a system is said to be *inherently disjunctive* [6]. The generalization of ER-systems to describe such a system is the topic of the next chapter.

3.7 Arbiters and Synchronizers

As mentioned before, program parts dealing with arbitration and synchronization of negated probes cannot be described by stable PR's. In our approach to performance analysis, these *arbiters* and *synchronizers* are regarded as belonging to the environment and the user is required to specify their non-deterministic behavior by selecting the appropriate environmental scenarios.

Chapter 4

Extended Event-Rule Systems

In an *extended ER-system (XER-system)*, an event may have more than one set of causes. Hence, XER-systems can be used to model inherently disjunctive systems. In this chapter, we will demonstrate how to compute the *period* of a *repetitive* XER-system and show that this period gives a good indication of the performance of the system. Most of the definitions and results for XER-systems have counterparts in ER-systems. Moreover, most of the proofs in this chapter, with the notable exception of those in Section 4.4 and Section 4.7, are extensions to those given in [6].

4.1 General Extended Event-Rule Systems

The source of a rule in an XER-system is a set of events. Furthermore, having more than one rule with f as target specifies that f has more than one possible set of causes. Since there may be more than one delay associated with each rule, an explicit function Δ is introduced to specify the delay between two events when one is a cause of the other. These concepts are formalized in the following definitions.

Definition: A *(general) extended event-rule system (XER-system)* is a triple $\mathcal{X} = \langle E, R, \Delta \rangle$ where

- E is a (possibly infinite) set of *events*;
- R is a (possibly infinite) set of *rules* where each rule is a pair $\langle C, f \rangle$, written as $C \mapsto f$, with $f \in E \wedge C \subseteq E$ and, for every f in E, there

exists at least one rule $C \mapsto f$ in R (C may be empty); and

- Δ is a *delay function* such that $\Delta : \mathcal{D} \rightarrow [0, \infty)$ with

$$\mathcal{D} = \{e, f, C : C \mapsto f \in R \wedge e \in C : \langle e, f, C \mapsto f \rangle\}.$$

The *initial event set* of $\mathcal{X}$ is

$$\mathbf{init}(\mathcal{X}) \stackrel{\text{def}}{=} \{f : \emptyset \mapsto f \in R : f\}.$$

For a rule $r = C \mapsto f$, the *source set* of r is $\mathbf{src}(r) \stackrel{\text{def}}{=} C$, and the *target* of r is $\mathbf{tar}(r) \stackrel{\text{def}}{=} f$. A rule is *empty* if its source set is empty. Also, C is called a *set of causes* (or a *cause set*) of f if $C \mapsto f$ is a rule. The *set of all cause sets* of f is $\{C : C \mapsto f \in R : C\}$. An event f is said to be *disjunctively caused* if it has more than one set of causes. An XER-system is said to be *conjunctive* if none of its events is disjunctively caused.

Example 4.1: As an example of an XER-system, consider $\mathcal{X} = \langle E, R, \Delta \rangle$ where

- $E = \{a, b, c, d, e\}$;
- $R = \{\emptyset \mapsto a\, ,\ \{a\} \mapsto b\, ,\ \{a\} \mapsto c\, ,\ \{a\} \mapsto d\, ,\ \{b, c\} \mapsto e\, ,\ \{d\} \mapsto e\}$; and
- $\Delta(a, b, \{a\} \mapsto b) = 1\, ,\ \Delta(a, c, \{a\} \mapsto c) = 1\, ,\ \Delta(a, d, \{a\} \mapsto d) = 1\, ,$
$\Delta(b, e, \{b, c\} \mapsto e) = 2\, ,\ \Delta(c, e, \{b, c\} \mapsto e) = 8\, ,\ \Delta(d, e, \{d\} \mapsto e) = 3.$

Then, $\mathbf{init}(\mathcal{X}) = \{a\}$ and $\emptyset \mapsto a$ is the only empty rule. For the rule $\{b, c\} \mapsto e$, $\mathbf{src}(\{b, c\} \mapsto e) = \{b, c\}$, and $\mathbf{tar}(\{b, c\} \mapsto e) = e$. Also, $\{b, c\}$ and $\{d\}$ are sets of causes of e; thus, e is disjunctively caused and $\mathcal{X}$ is not conjunctive.
$\square$

Intuitively, a rule $C \mapsto f$ specifies the timing constraint that the event f can occur only if *all* events in C have occurred and a proper amount of delay (as specified by Δ) has elapsed. When there are two or more rules with target f, then f can occur if the timing constraint imposed by *any* of these rules can be satisfied. So, in the previous example, e can occur only if either both b and c have occurred, or d has occurred. These concepts are formalized by the following definition.

Definition: A *timing function* for $\mathcal{X} = \langle E, R, \Delta \rangle$ is a function $t : E \rightarrow [0, \infty)$ such that[1]

$$\begin{array}{l} \forall f : f \in E : (\ \exists r : r \in R \wedge f = \mathbf{tar}(r) : \\ \qquad (\forall e : e \in \mathbf{src}(r) : t(f) \geq t(e) + \Delta(e, f, r))\ ). \end{array} \tag{4.1}$$

An XER-system is *feasible* if there exists a timing function for it.

Example 4.2: The XER-system in the previous example is feasible since t, where $t(a) = 0, t(b) = 1, t(c) = 1, t(d) = 1, t(e) = 4$, is a timing function for it. $\Box$

Note that in this example, the values of Δ are numerically given and it can be shown that the rule $\{b, c\} \mapsto e$ can be removed without affecting the corresponding set of timing functions. However, when one wants to optimize the performance of an XER-system, the values of Δ are themselves functions of other variables and, consequently, it cannot be decided in advance which of the possible sets of causes for an event can be removed. Thus, in general, an XER-system cannot be reduced to one that is conjunctive.

4.1.1 Conjunctive General XER-Systems

As the following lemma shows, any conjunctive XER-system can be transformed into an equivalent ER-system.

Lemma 4.1 *Let $\mathcal{X} = \langle E, R, \Delta \rangle$ be a conjunctive XER-system. Then, there exists an ER-system such that the two systems have the same set of timing functions.*

Proof: Let $\mathcal{Y} = \langle E_{\mathcal{Y}}, R_{\mathcal{Y}} \rangle$ be the ER-system where $E_{\mathcal{Y}} = E$ and

$$\begin{array}{l} R_{\mathcal{Y}} = \{e, f, r, \alpha : \\ \qquad r \in R \wedge e \in \mathbf{src}(r) \wedge f = \mathbf{tar}(r) \wedge \Delta(e, f, r) = \alpha : e \overset{\alpha}{\mapsto} f\}. \end{array} \tag{4.2}$$

Now, since $\mathcal{X}$ is conjunctive, for every $\hat{f}$ in E, there exists a unique rule $\hat{r}$ in R with target $\hat{f}$. By the construction of $R_{\mathcal{Y}}$, $e \overset{\alpha}{\mapsto} \hat{f} \in R_{\mathcal{Y}}$ if and only if $e \in \mathbf{src}(\hat{r}) \wedge \Delta(e, \hat{f}, \hat{r}) = \alpha$. So, (4.1) and (3.1) are equivalent and the lemma is established. *Q.E.D.*

[1] For succinctness, a minor liberty with the functional notation has been taken and $\Delta(e, f, r)$ is used as an abbreviation for $\Delta(\langle e, f, r \rangle)$ and similarly for other functions.

4.1.2 Constraint Graphs

Definition: The *constraint graph* for an XER-system $\mathcal{X} = \langle E, R, \Delta \rangle$ is the directed labeled bipartite graph $G = \langle V, A \rangle$ with $V = V_E \uplus V_R$ and $A = A_\wedge \uplus A_\vee$ where

- V_E is the set of *event vertices*, one corresponding[2] to each event in E,
- V_R is the set of *rule vertices*, one corresponding to each rule in R,
- $A_\wedge = \{e, f, r : r \in R \wedge e \in \mathbf{src}(r) \wedge f = \mathbf{tar}(r) : \langle e, r, \Delta(e, f, r) \rangle\}$ is the set of *conjunctive arcs*, and
- $A_\vee = \{r, f : r \in R \wedge f = \mathbf{tar}(r) : \langle r, f, 0 \rangle\}$ is the set of *disjunctive arcs*.

For an arc $\langle e, r, \alpha \rangle$, α is called the *weight* of the arc.

Example 4.3: The constraint graph for the XER-system in Example 4.1 is shown in Figure 4.1. The following conventions have been adopted: event vertices are drawn as circles, rule vertices as boxes, conjunctive arcs are drawn smooth, and disjunctive arcs are drawn with wiggles. Note that since the labels on all disjunctive edges are zeros, they are not included in the picture. □

Ancestors at Infinite Distances

Example 4.4: Consider the XER-system $\mathcal{X} = \langle E, R, \Delta \rangle$ where

- $E = \{i : i \in \mathbb{N} : a_i\}$,
- $R = \{i : i \in \mathbb{N} : \{a_{i+1}\} \mapsto a_i\}$, and
- $\Delta(a_{i+1}, a_i, \{a_{i+1}\} \mapsto a_i) = \alpha$.

[2]For succinctness, the same name is used for an event and the event vertex to which it corresponds; the context in which the name is used should resolve any ambiguity. The same convention is used for a rule and the corresponding rule vertex.

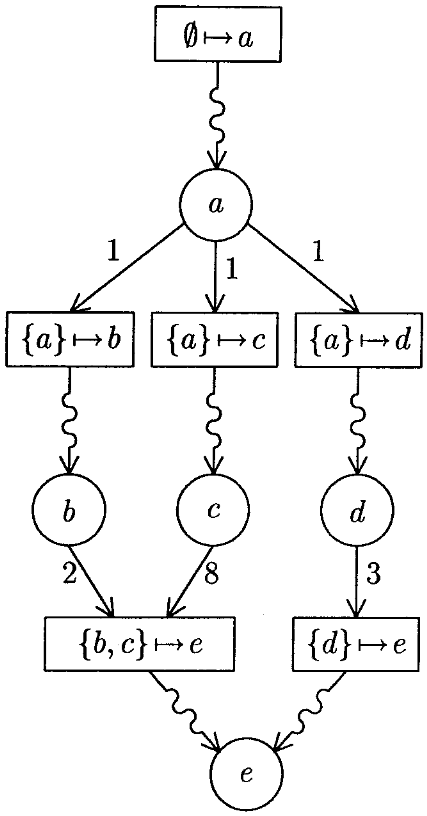

Figure 4.1: Constraint graph for Example 4.1

Figure 4.2a shows its constraint graph. If there is a path from vertex v to vertex w in the graph, then v is an *ancestor* of w and the *distance* between them is the number of arcs in the path. Note that every vertex in Figure 4.2a has an ancestor at an infinite distance from it and that $\mathcal{X}$ is feasible only if $\alpha = 0$. $\square$

If v and w are event vertices and w is an ancestor of v at an infinite distance, then the occurrence of v depends on the occurrence of w, which took place *infinitely* long ago. Since this situation is unrealistic, in the sequel, *XER-systems whose constraint graphs contain vertices with ancestors at infinite distances will be excluded from consideration.* Note that the graph itself may be infinite provided the distance between every vertex and any of its ancestors is finite (but not necessarily bounded).

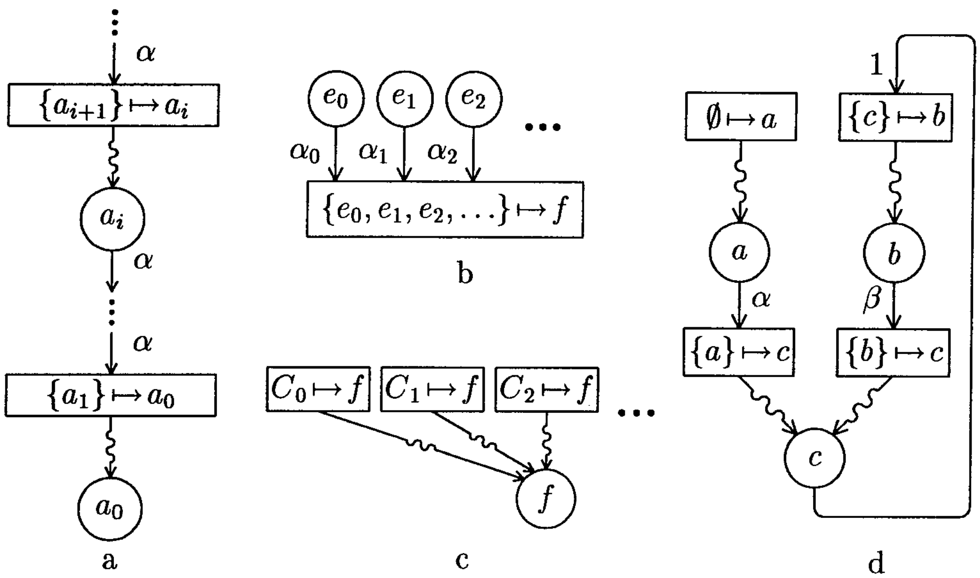

Figure 4.2: Pathological constraint graphs

Vertices with Infinite In-degrees

In a constraint graph, if a rule vertex $r = C \mapsto f$ has an infinite in-degree (see Figure 4.2b), then $|C| = \infty$ and the event f can occur due to this set of causes only after the infinite number of events in C have already occurred. Similarly, if an event vertex f has an infinite in-degree (Figure 4.2c), then f have infinite number of possible sets of causes. To avoid these pathological situations that do not correspond to realistic systems, *only XER-systems whose constraint graphs contain no vertices with infinite in-degrees are considered.*

Cyclic Constraint Graphs

In [6], Burns shows the analog of the fact that if an XER-system is conjunctive and feasible, then the sum of the weights along the arcs in any cycle in its constraint graph is zero. Furthermore, the vertices in that cycle can then be "merged" together and be treated as a single vertex for the purpose of defining timing simulation as presented in the next sub-section. As the following example shows, however, these results do not hold if the XER-system is not conjunctive.

Example 4.5: Consider the XER-system $\mathcal{X} = \langle E, R, \Delta \rangle$ where

- $E = \{a, b, c\}$,
- $R = \{\emptyset \mapsto a\,,\ \{a\} \mapsto c\,,\ \{b\} \mapsto c\,,\ \{c\} \mapsto b\}$, and
- $\Delta(a, c, \{a\} \mapsto c) = \alpha\,,\ \Delta(b, c, \{b\} \mapsto c) = \beta\,,\ \Delta(c, b, \{c\} \mapsto b) = 1.$

Its constraint graph is shown in Figure 4.2d and a timing function for it is

$$t(a) = 0\,,\ t(b) = \alpha + 1\,,\ t(c) = \alpha.$$

In fact, regardless of the values of α and β, $t(c) < t(b)$. Thus, this system is equivalent to one where the rule $\{b\} \mapsto c$ is removed. □

As this example demonstrates, if an XER-system is feasible, then any cycle with non-zero weight contains an arc that arises from an unnecessary rule. Also, an XER-system with a cyclic constraint graph implies there is an event (like c above) which, indirectly, can cause itself to occur. Since such systems occur rarely, if at all, in practice, it is assumed that *in the sequel, all constraint graphs are acyclic.*

4.1.3 Timing Simulation

Since we consider only XER-systems with acyclic constraint graphs that do not contain vertices with ancestors at infinite distances or vertices with infinite in-degrees, the following function is well-defined. Note that the minimization is over all rules with target f and the maximization is over all events in the source set of each of these rules.

Definition: For an XER-system $\mathcal{X} = \langle E, R, \Delta \rangle$, $\hat{t} : E \to [0, \infty)$ defined below is called the *timing simulation* of $\mathcal{X}$:

$$\hat{t}(f) \stackrel{\text{def}}{=} \begin{cases} 0 & \text{if } f \in \mathbf{init}(\mathcal{X}) \\ \mathbf{min}\{r : (r \in R) \wedge (f = \mathbf{tar}(r)) : & \\ \quad \mathbf{max}\{e : e \in \mathbf{src}(r) : \hat{t}(e) + \Delta(e, f, r)\}\} & \text{if } f \notin \mathbf{init}(\mathcal{X}). \end{cases} \tag{4.3}$$

Example 4.6: The timing function given in Example 4.2 is the timing simulation of the corresponding XER-system. □

Lemma 4.2 *The timing simulation of $\mathcal{X}$ is a timing function of $\mathcal{X}$ and for any timing function t of $\mathcal{X}$,*

$$\forall e : e \in E : \hat{t}(e) \leq t(e).$$

Proof: If f is in $\mathbf{init}(\mathcal{X})$, then (4.1) holds since there exists an empty rule with target f. If f is not in $\mathbf{init}(\mathcal{X})$, then there exists a r such that

$$\hat{t}(f) = \mathbf{max}\{e : e \in \mathbf{src}(r) : \hat{t}(e) + \Delta(e, f, r)\}$$

and this equality implies

$$\forall e : e \in \mathbf{src}(r) : \hat{t}(f) \geq \hat{t}(e) + \Delta(e, f, r) \tag{4.4}$$

and, so, $\hat{t}$ is a timing function of $\mathcal{X}$.

For a given timing function t, let

$$\mathcal{Z} \stackrel{\text{def}}{=} \{e : \hat{t}(e) > t(e) : e\}.$$

If $\mathcal{Z}$ is not empty then let $\hat{f}$ be an event in $\mathcal{Z}$ such that for any event e and rule r

$$e \in \mathbf{src}(r) \wedge \hat{f} = \mathbf{tar}(r) \Rightarrow e \notin \mathcal{Z}. \tag{4.5}$$

Such an event exists since the constraint graph is acyclic.

If $\hat{f} \in \mathbf{init}(\mathcal{X})$, then $t(\hat{f}) < \hat{t}(\hat{f}) = 0$ which is a contradiction. Alternatively, if $\hat{f} \notin \mathbf{init}(\mathcal{X})$, then by the definition of a timing function, there exists a rule $\hat{r}$ such that

$$t(\hat{f}) \geq \mathbf{max}\{e : e \in \mathbf{src}(\hat{r}) : t(e) + \Delta(e, \hat{f}, \hat{r})\}.$$

But, by (4.5), the right side of the above inequality is at least

$$\mathbf{max}\{e : e \in \mathbf{src}(\hat{r}) : \hat{t}(e) + \Delta(e, \hat{f}, \hat{r})\} \geq \hat{t}(\hat{f})$$

where the last inequality is due to the minimality of $\hat{t}(\hat{f})$ in (4.3). Thus, $t(\hat{f}) \geq \hat{t}(\hat{f})$ in contradiction to the assumption that $\hat{f} \in \mathcal{Z}$. Thus, $\mathcal{Z}$ is empty and the lemma is established. *Q.E.D.*

4.2 Repetitive XER-Systems

A general XER-system is difficult to analyze due to the fact that it may have a large number of arbitrary timing constraints. Fortunately, the systems that we are interested in exhibit regular behavior and can be modeled by a "repetitive" XER-system as defined below.

Definition: A *repetitive XER-system* $\mathcal{X}'$ is a quadruple $\langle E', R', \delta, \theta \rangle$ where

- E' is a finite set of *transitions*;
- R' is a finite set of *templates* where each template is a pair $\langle C', v \rangle$, written as $C' \mapsto v$, with $v \in E' \land C' \subseteq E'$ and, for every v in E', there exists at least one template $\langle C', v \rangle$ in R';
- δ is a *delay function* such that $\delta : \mathcal{D}' \rightarrow [0, \infty)$ with

$$\mathcal{D}' = \{u, v, C' : C' \mapsto v \in R' \land u \in C' : \langle u, v, C' \mapsto v \rangle\}; \text{ and}$$

- θ is an *occurrence-index offset function* such that $\theta : \mathcal{D}' \rightarrow \mathbb{Z}$.

The *maximum occurrence-index offset* of $\mathcal{X}'$ is

$$\theta_{\max} = \mathbf{max}(\{u, v, q : \langle u, v, q \rangle \in \mathcal{D}' : \theta(u, v, q)\} \cup \{0\}). \tag{4.6}$$

For a template $q = C' \mapsto v$, the *source set* of q is $\mathbf{src}(q) \stackrel{\text{def}}{=} C'$, and the *target* of q is $\mathbf{tar}(q) \stackrel{\text{def}}{=} v$. Also, C' is called *a set of causes* for v, if $C' \mapsto v$ is a template. A transition v is said to be *disjunctively caused* if it has more than one set of causes. A repetitive XER-system is *conjunctive* if it has no transition that is disjunctively caused.

Example 4.7: Consider the following program which waits for either of two input channels to be activated, performs an output communication C, and then complete both input communications:

$$*[[\overline{A} \lor \overline{B} \longrightarrow C;\ A \bullet B\ ]].$$

One reshuffled handshaking expansion for this program is

$$*[[a_i \lor b_i];\ c_o\uparrow;\ [c_i \land a_i \land b_i];\ s\uparrow;\ (a_o\uparrow \parallel b_o\uparrow \parallel c_o\downarrow);$$
$$[\neg c_i \land \neg a_i \land \neg b_i];\ s\downarrow;\ (a_o\downarrow \parallel b_o\downarrow)]$$

which compiles into the following PR set:

$$
\begin{array}{lcllcl}
\neg s \wedge (a_i \vee b_i) & \to & c_o\uparrow & s & \to & c_o\downarrow \\
c_i \wedge a_i \wedge b_i & \to & s\uparrow & \neg c_i \wedge \neg a_i \wedge \neg b_i & \to & s\downarrow \\
s & \to & a_o\uparrow & \neg s & \to & a_o\downarrow \\
s & \to & b_o\uparrow & \neg s & \to & b_o\downarrow .
\end{array}
$$

Assuming that the environment is just independent processes completing the communications on A, B, and C, and that initially all variables are **false**, this PR set can be described by the repetitive XER-system $\mathcal{X}' = \langle E', R', \delta, \theta \rangle$ where

- $E' = \{a_i\uparrow, a_i\downarrow, a_o\uparrow, a_o\downarrow, b_i\uparrow, b_i\downarrow, b_o\uparrow, b_o\downarrow, c_i\uparrow, c_i\downarrow, c_o\uparrow, c_o\downarrow, s\uparrow, s\downarrow\}$;

- $$
\begin{array}{llll}
R' = \{ & \{s\downarrow, a_i\uparrow\} \mapsto c_o\uparrow, & \{s\downarrow, b_i\uparrow\} \mapsto c_o\uparrow, & \{c_i\uparrow, a_i\uparrow, b_i\uparrow\} \mapsto s\uparrow, \\
& \{s\uparrow\} \mapsto a_o\uparrow, & \{s\uparrow\} \mapsto b_o\uparrow, & \{s\uparrow\} \mapsto c_o\downarrow, \\
& \{c_i\downarrow, a_i\downarrow, b_i\downarrow\} \mapsto s\downarrow, & \{s\downarrow\} \mapsto a_o\downarrow, & \{s\downarrow\} \mapsto b_o\downarrow, \\
& \{a_o\downarrow\} \mapsto a_i\uparrow, & \{a_o\uparrow\} \mapsto a_i\downarrow, & \{b_o\downarrow\} \mapsto b_i\uparrow, \\
& \{b_o\uparrow\} \mapsto b_i\downarrow, & \{c_o\uparrow\} \mapsto c_i\uparrow, & \{c_o\downarrow\} \mapsto c_i\downarrow \ \};
\end{array}
$$

- δ is some function depending on the timing model being used — for concreteness, we will assume $\delta(u, v, q) = \delta_v$ in this example; and

- $\theta(s\downarrow, c_o\uparrow, \{s\downarrow, a_i\uparrow\} \mapsto c_o\uparrow) = 1$, $\theta(s\downarrow, c_o\uparrow, \{s\downarrow, b_i\uparrow\} \mapsto c_o\uparrow) = 1$, $\theta(a_o\downarrow, a_i\uparrow, \{a_o\downarrow\} \mapsto a_i\uparrow) = 1$, $\theta(b_o\downarrow, b_i\uparrow, \{b_o\downarrow\} \mapsto b_i\uparrow) = 1$, and $\theta(u, v, q) = 0$ for any other combination of u, v, and q in the domain of θ.

Note that if $\theta(u, v, q) = 1$, then it is the occurrence of transition u in the *previous* iteration that causes the occurrence of transition v in the current iteration. Since $c_o\uparrow$ is disjunctively caused, $\mathcal{X}'$ is not conjunctive. Also, $\theta_{\max} = 1$. □

A repetitive XER-system $\mathcal{X}'$ can be viewed as a specification of a (general) XER-system $\mathcal{X}$. The events of $\mathcal{X}$ are of the form $\langle u, i \rangle$ where u is a transition of $\mathcal{X}'$ and i is a natural number called the *occurrence index* of the event. The rules of $\mathcal{X}$ are to be generated by the templates of $\mathcal{X}'$. Intuitively, if v is the target of a template q and u is a transition in the source set of q, then there is a rule r such that $\langle v, i \rangle$ is the target of r and an event $\langle u, j \rangle$ is in the source set of r. The difference between i and j is specified by the function θ and hence its name. The following definitions formalize these notions.

Definition: For a template $q = C' \mapsto v$ and a natural number i, the *i-th instantiation* of q is the rule $q\lceil i = C \mapsto \langle v, i \rangle$ where

$$C = \{u : u \in C' \wedge i \geq \theta(u, v, q) : \langle u, i - \theta(u, v, q) \rangle\}. \qquad (4.7)$$

Note that if $i \geq \theta_{\max}$, then

$$u \in \mathbf{src}(q) \Leftrightarrow \langle u, i - \theta(u, v, q) \rangle \in \mathbf{src}(q\lceil i). \qquad (4.8)$$

Example 4.8: Let q be the first template listed in the previous example. Then,

$$q\lceil i = \begin{cases} \{\langle a_i\uparrow, 0 \rangle\} \mapsto \langle c_o\uparrow, 0 \rangle & \text{if } i = 0 \\ \{\langle s\downarrow, i-1 \rangle, \langle a_i\uparrow, i \rangle\} \mapsto \langle c_o\uparrow, i \rangle & \text{if } i > 0. \end{cases}$$

□

Definition: A repetitive XER-system $\mathcal{X}' = \langle E', R', \delta, \theta \rangle$ *induces* or *generates* the XER-system $\mathcal{X} = \langle E, R, \Delta \rangle$ where

- $E = \{u, i : u \in E' \wedge i \in \mathbb{N} : \langle u, i \rangle\}$;
- $R = \{q, i : q \in R' \wedge \ i \in \mathbb{N} : q\lceil i\}$; and
- $\Delta(\langle u, j \rangle, \langle v, i \rangle, q\lceil i) = \delta(u, v, q)$ with the domain of Δ being

$$\mathcal{D} = \{u, j, v, i, q : \langle v, i \rangle = \mathbf{tar}(q\lceil i) \wedge \langle u, j \rangle \in \mathbf{src}(q\lceil i) : \\ \langle \langle u, j \rangle, \langle v, i \rangle, q\lceil i \rangle\}.$$

Note that this construction is well-defined since

$$\langle \langle u, j \rangle, \langle v, i \rangle, q\lceil i \rangle \in \mathcal{D} \Rightarrow \langle u, v, q \rangle \in \mathcal{D}'.$$

Also, by the construction described above, a conjunctive repetitive XER-system (one set of causes per transition) induces a conjunctive general XER-system (one set of causes per event). For brevity, a timing function (or the timing simulation) of $\mathcal{X}$ is also referred to as a timing function (or the timing simulation) of $\mathcal{X}'$.

4.2.1 Conjunctive Repetitive XER-Systems

A repetitive XER-system is the analog of a repetitive ER-system defined in the previous chapter. As expected, there is a natural correspondence between a conjunctive repetitive XER-system and a repetitive ER-system.

Lemma 4.3 *Let $\mathcal{X}' = \langle E', R', \delta, \theta \rangle$ be a conjunctive repetitive XER-system. Then, there exists a repetitive ER-system such that the two systems have the same set of timing functions.*

Proof: Consider the repetitive ER-system $\mathcal{Y}' = \langle E'_{\mathcal{Y}}, R'_{\mathcal{Y}} \rangle$ where $E'_{\mathcal{Y}} = E'$ and

$$R'_{\mathcal{Y}} = \{u, v, \alpha, \varepsilon, q : q \in R' \wedge u \in \mathbf{src}(q) \wedge v = \mathbf{tar}(q) \wedge \\ \delta(u, v, q) = \alpha \wedge \theta(u, v, q) = \varepsilon : \langle u, v, \alpha, \varepsilon \rangle\}. \tag{4.9}$$

Let $\mathcal{X} = \langle E, R, \Delta \rangle$ be the XER-system induced by $\mathcal{X}'$ and $\mathcal{Y} = \langle E_{\mathcal{Y}}, R_{\mathcal{Y}} \rangle$ be the ER-system induced by $\mathcal{Y}'$. Then, $E'_{\mathcal{Y}} = E'$ implies $E_{\mathcal{Y}} = E$. Let $\langle \hat{v}, \hat{\imath} \rangle$ be an event in E. Since $\mathcal{X}'$ is conjunctive, there exists a unique template $\hat{q}$ such that $\hat{q} = C' \mapsto \hat{v}$. Now, by construction, the only rule in R with target $\langle \hat{v}, \hat{\imath} \rangle$ is $\hat{q}\lceil \hat{\imath} = \langle C, \langle \hat{v}, \hat{\imath} \rangle \rangle$ where

$$C = \{u : u \in C' \wedge \hat{\imath} \geq \theta(u, \hat{v}, \hat{q}) : \langle u, \hat{\imath} - \theta(u, \hat{v}, \hat{q}) \rangle\}.$$

Next, by (4.9), $\langle u, \hat{v}, \alpha, \varepsilon \rangle$ is in $R'_{\mathcal{Y}}$ if and only if

$$u \in C' \wedge \alpha = \delta(u, \hat{v}, \hat{q}) \wedge \varepsilon = \theta(u, \hat{v}, \hat{q}).$$

This relationship implies that the subset of rules in $R_{\mathcal{Y}}$ with target $\langle \hat{v}, \hat{\imath} \rangle$ is precisely

$$\{u : u \in C' \wedge \hat{\imath} \geq \theta(u, \hat{v}, \hat{q}) : \langle u, \hat{\imath} - \theta(u, \hat{v}, \hat{q}) \rangle \overset{\delta(u, \hat{v}, \hat{q})}{\mapsto} \langle \hat{v}, \hat{\imath} \rangle\}.$$

Since $\delta(u, \hat{v}, \hat{q}) = \Delta(\langle u, j \rangle, \langle \hat{v}, \hat{\imath} \rangle, \hat{q}\lceil \hat{\imath})$, $\langle u, j \rangle \overset{\alpha}{\mapsto} \langle \hat{v}, \hat{\imath} \rangle$ is a rule in $R_{\mathcal{Y}}$ if and only if there exists $\hat{q}\lceil \hat{\imath}$ such that

$$\langle u, j \rangle \in \mathbf{src}(\hat{q}\lceil \hat{\imath}) \wedge \langle v, i \rangle = \mathbf{tar}(\hat{q}\lceil \hat{\imath}) \wedge \alpha = \Delta(\langle u, j \rangle, \langle \hat{v}, \hat{\imath} \rangle, \hat{q}\lceil \hat{\imath}).$$

Since this analysis holds for any $\hat{v}$, $R_{\mathcal{Y}}$ is related to R by (4.2). The lemma then follows from the proof of Lemma 4.1. *Q.E.D.*

4.2.2 Collapsed-Constraint Graphs

Definition: The *collapsed-constraint graph* for a repetitive XER-system $\mathcal{X}' = \langle E', R', \delta, \theta \rangle$ is the directed doubly-labeled bipartite graph $G' = \langle V', A' \rangle$ with $V' = V'_E \uplus V'_R$ and $A' = A'_\wedge \uplus A'_\vee$ where

- V'_E is the set of *transition vertices*, one corresponding to each transition u in E',
- V'_R is the set of *template vertices*, one corresponding to each template q in R',
- $A'_\wedge = \{u, v, q : q \in R' \wedge u \in \mathbf{src}(q) \wedge v = \mathbf{tar}(q) : \langle u,\ q,\ \Delta(u, v, q), \theta(u, v, q)\rangle\}$ is the set of *conjunctive arcs*, and
- $A_\vee = \{q, v : q \in R' \wedge v = \mathbf{tar}(q) : \langle q, v, 0, 0\rangle\}$ is the set of *disjunctive arcs*.

For an arc $\langle w, w', \alpha, \varepsilon\rangle$, α and ε are called the *weight* and the *occurrence-index offset* of the arc, respectively.

Example 4.9: The collapsed-constraint graph for the $\mathcal{X}'$ of Example 4.7 is shown in Figure 4.3 where transition vertices are drawn as circles, template vertices as boxes, conjunctive arcs are drawn smooth, and disjunctive arcs with wiggles. For clarity, we have adopted the convention that for the conjunctive arc $\langle u, q, \alpha, \varepsilon\rangle$, if $\varepsilon > 0$, then the corresponding arc is drawn with a number of slashes equal to ε. Also, the zero labels on the disjunctive arcs have been left out. $\square$

For a path

$$\rho' = \langle w_0, w_1, \ldots, w_l\rangle$$

in a collapsed-constraint graph, the sum of the weights on the arcs in ρ' will be denoted as

$$\delta(\rho') \stackrel{\text{def}}{=} \sum_{j=0}^{l-1} (\alpha \text{ in } \langle w_j, w_{j+1}, \alpha, \varepsilon\rangle),$$

and the sum of the occurrence-index offsets of the arcs as

$$\theta(\rho') \stackrel{\text{def}}{=} \sum_{j=0}^{l-1} (\varepsilon \text{ in } \langle w_j, w_{j+1}, \alpha, \varepsilon\rangle).$$

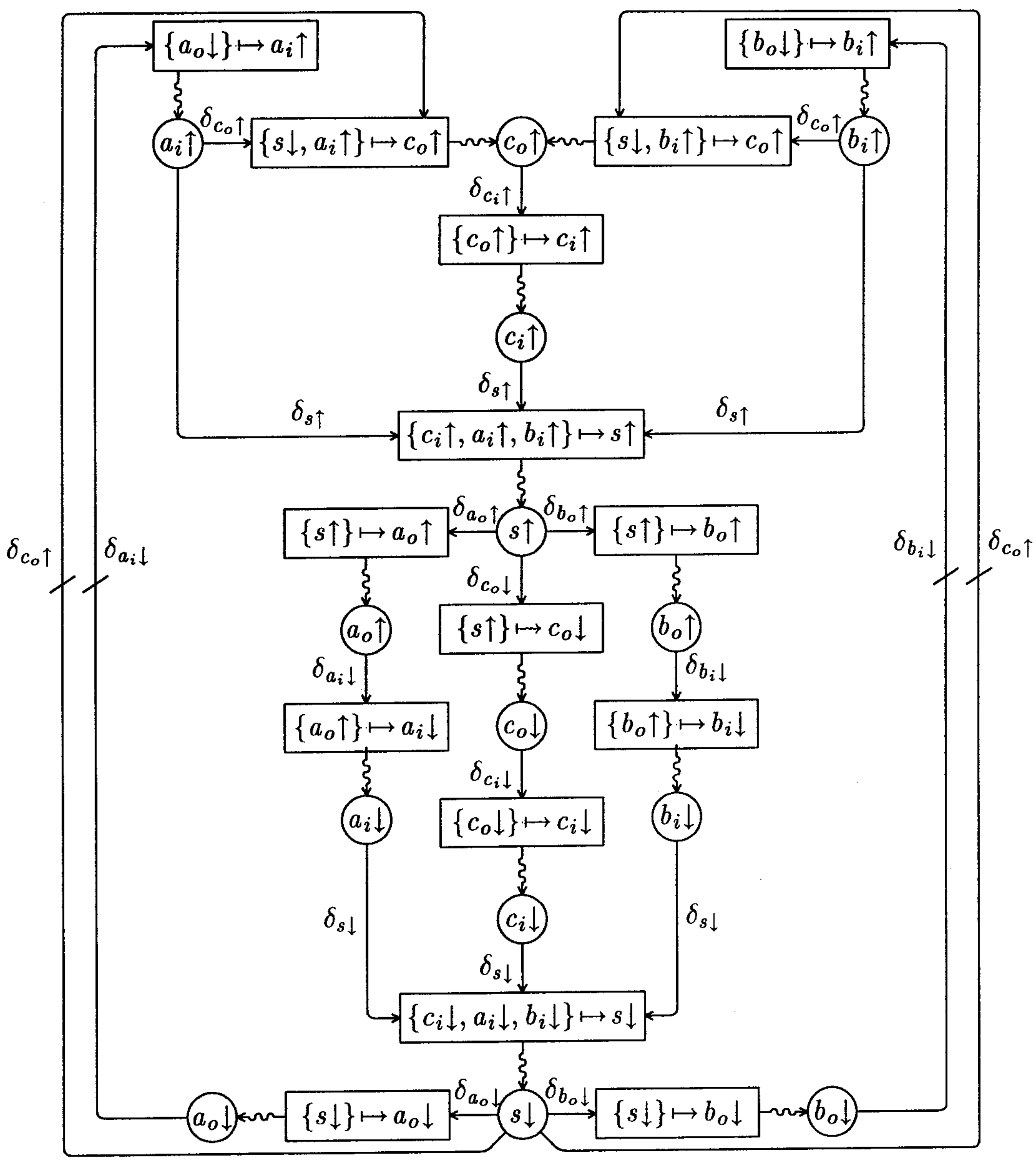

Figure 4.3: Collapsed constraint graph for Example 4.7

The following results relate the collapsed-constraint graph of a repetitive XER-system $\mathcal{X}'$ to the constraint graph of the general XER-system induced by $\mathcal{X}'$.

Lemma 4.4 *Let G' be the collapsed-constraint graph of $\mathcal{X}'$ and G be the constraint graph of $\mathcal{X}$, the XER-system induced by $\mathcal{X}'$. If there exists a path ρ in G from event vertex $\langle u, j\rangle$ to $\langle v, i\rangle$, then there exists a path ρ' in G' from transition vertex u to transition vertex v.*

Proof: If $\langle\langle u, j\rangle, r, \langle v, i\rangle\rangle$ is a path of length 2 in G, then, by construction, there exists $q \in R'$ such that $r = q\lceil i$, $v = \mathbf{tar}(q)$, $u \in \mathbf{src}(q)$, and $j = i - \theta(u, v, q)$. Consequently, $\langle u, q, v\rangle$ is a path in G'. Since every non-empty path from an event vertex to an event vertex is a concatenation of paths with length 2, the lemma is established. *Q.E.D.*

Lemma 4.5 *Let G and G' be as in Lemma 4.4. If there is a path ρ' from transition vertex u to transition vertex v in G' then there exists an I such that, for all $i \geq I$, there exists a path ρ from event vertex $\langle u, i - \theta(\rho')\rangle$ to event vertex $\langle v, i\rangle$ in G.*

Proof: Use induction on l, the length of ρ'.

Base Case: $(l = 0)$ In this case, $u = v$; so, let $\rho = \langle\langle u, i\rangle\rangle$.

Inductive Step: Assume that the lemma holds for all ρ' with length less than or equal to l. Consider

$$\sigma' = \langle (u = w_0), w_1, \ldots, (w_{l+1} = v)\rangle.$$

Since G' is bipartite, w_l is a template vertex q, and w_{l-1} is a transition vertex $\tilde{u}$. So, σ' be the concatenation of a path ρ' with length $l - 1$ and the path $\langle \tilde{u}, q, v\rangle$.

Now, by definition of G', $\tilde{u} \in \mathbf{src}(q)$ and $v = \mathbf{tar}(q)$. But, by the construction of $\mathcal{X}$, for all natural number i such that $i \geq \theta(\tilde{u}, v, q)$, there exists rule $r = q\lceil i$ in $\mathcal{X}$ such that $\langle \tilde{u}, i - \theta(\tilde{u}, v, q)\rangle \in \mathbf{src}(r)$ and $\langle v, i\rangle = \mathbf{tar}(r)$. Letting ε be $\theta(\tilde{u}, v, q)$ implies $\langle\langle \tilde{u}, i - \varepsilon\rangle, r, \langle v, i\rangle\rangle$ is a path in G.

By the inductive hypothesis, there exists I such that for all i with $(i - \varepsilon) \geq I$, there exists a path ρ from $\langle u, i - \varepsilon - \theta(\rho')\rangle$ to $\langle \tilde{u}, i - \varepsilon\rangle$. The concatenation of this path and $\langle\langle\tilde{u}, i-\varepsilon\rangle, r, \langle v, i\rangle\rangle$ for sufficiently large i yields a path from $\langle u, i-\varepsilon-\theta(\rho')\rangle$ to $\langle v, i\rangle$. Since

$$\theta(\sigma') = \theta(\rho') + \theta(\tilde{u}, v, q) + 0 = \theta(\rho') + \varepsilon,$$

the lemma is established. *Q.E.D.*

4.3 Pseudorepetitive XER-Systems

Sometimes, a system being modeled exhibits initial transient behavior prior to entering its steady state. For such a system, "pseudorepetitive" XER-systems are introduced.

Definition: A *pseudorepetitive XER-system* $\mathcal{X}''$ is a pair $\langle \mathcal{X}_0, \mathcal{X}'_1\rangle$ where

- $\mathcal{X}_0 = \langle E_0, R_0, \Delta_0\rangle$ is a (general) XER-system with E_0 and R_0 finite and is called the *initial part* of $\mathcal{X}''$; and
- $\mathcal{X}'_1 = \langle E'_1, R'_1, \delta_1, \theta_1\rangle$ is a repetitive XER-system and is called the *repeated part* of $\mathcal{X}''$.

Example 4.10: Consider a system where there is an initial occurrence of b, followed the endless repetition of the sequence (a, b, c). Then, this system can be described by a pseudorepetitive XER-system $\mathcal{X}'' = \langle \mathcal{X}_0, \mathcal{X}'_1\rangle$ where

- $\mathcal{X}_0 = \langle \ \{ \langle b, 0\rangle, \langle a, 0\rangle \}, \{ \emptyset \mapsto \langle b, 0\rangle, \{\langle b, 0\rangle\} \mapsto \langle a, 0\rangle \}, \Delta_0 \ \rangle$; and
- $\mathcal{X}'_1 = \langle \ \{ a, b, c \}, \{ \{a\} \mapsto b, \{b\} \mapsto c, \{c\} \mapsto a \}, \delta_1, \theta_1 \ \rangle$.

Since the i-th occurrence of c is caused by the $(i+1)$-th occurrence of b, $\theta_1(b, c, \{b\} \mapsto c) = -1$. By similar analyses, $\theta_1(a, b, \{a\} \mapsto b) = 1$ and $\theta_1(c, a, \{c\} \mapsto a) = 1$. Again, Δ_0 and δ_1 depend on the timing model. The way in which $\mathcal{X}''$ specifies the original system will be discussed after the next definition. □

Intuitively, the XER-system generated by a pseudorepetitive XER-system $\mathcal{X}''$ is the combination of its initial part and the XER-system generated by its repeated part. If an event appears in both parts, then its constraints as specified in the initial part take precedence over the ones specified in the repeated part. The construction below formalizes these notions.

Definition: Let $\mathcal{X}''$ be a pseudorepetitive XER-system as defined above. Let $\mathcal{X}_2 = \langle E_2, R_2, \Delta_2 \rangle$ be the XER-system induced by $\mathcal{X}'_1$. Then, the XER-system *induced* or *generated* by $\mathcal{X}''$ is $\mathcal{X} = \langle E, R, \Delta \rangle$ where

- $E = E_0 \cup E_2$;
- $R = R_0 \cup (R_2 \lfloor E_0)$ with

$$R_2 \lfloor E_0 \stackrel{\text{def}}{=} \{C, f : C \mapsto f \in R_2 \wedge f \notin E_0 : C \mapsto f\}; \text{ and} \qquad (4.10)$$

- Δ is defined by

$$\Delta(e, f, r) = \begin{cases} \Delta_0(e, f, r) & \text{if } f \in E_0 \\ \Delta_2(e, f, r) & \text{if } f \notin E_0. \end{cases} \qquad (4.11)$$

Note that an event in E_0 can be in the source set of a rule in R_2; hence, events of this type serve as a link between the initial part and the repeated part of the pseudorepetitive XER-system.

Example 4.11: Continuing with the previous example, $\mathcal{X}'_1$ induces the rules $\emptyset \mapsto \langle a, 0 \rangle$. This rule is superseded by $\{\langle b, 0 \rangle\} \mapsto \langle a, 0 \rangle$ in $\mathcal{X}_0$, reflecting the transient behavior of the original system. Note that even though $\mathcal{X}'_1$ also induces $\emptyset \mapsto \langle b, 0 \rangle$, this rule in needed in $\mathcal{X}_0$ so that $\mathcal{X}_0$ is an XER-system. $\square$

4.3.1 Approximating Timing Simulation

Though they can be used to model a large class of systems, pseudorepetitive XER-systems are cumbersome to work with. Fortunately, Lemma 4.8 shows that to get a good indication of its timing simulation, it is sufficient to deal with the repeated part of any given pseudorepetitive XER-system. To establish the lemma, the following properties of numbers are needed.

Lemma 4.6 *For $N > 0$ and any number B, if*

$$\forall i : 0 \leq i < N : |x_i - y_i| \leq B,$$

then

$$|\mathbf{min}\{i : 0 \le i < N : x_i\} - \mathbf{min}\{i : 0 \le i < N : y_i\}| \le B; \qquad (4.12)$$
$$|\mathbf{max}\{i : 0 \le i < N : x_i\} - \mathbf{max}\{i : 0 \le i < N : y_i\}| \le B. \qquad (4.13)$$

Proof: Let $x_j = \mathbf{min}\{x_i\}$ and $y_k = \mathbf{min}\{y_i\}$. If $x_j \ge y_k$, then the minimality of x_j implies

$$|\mathbf{min}\{x_i\} - \mathbf{min}\{y_i\}| = (x_j - y_k) \le (x_k - y_k) \le B.$$

Alternatively, if $x_j < y_k$, then the minimality of y_k implies

$$|\mathbf{min}\{x_i\} - \mathbf{min}\{y_i\}| = (y_k - x_j) \le (y_j - x_j) \le B.$$

Thus, (4.12) is established. Equation (4.13) can be verified by similar arguments. *Q.E.D.*

Corollary 4.7 *Let $N > 0$ and $\forall i : 0 \le i < N : M_i > 0$. Let*

$$\begin{aligned} X &\stackrel{\text{def}}{=} \mathbf{min}\{i : 0 \le i < N : \mathbf{max}\{j : \le j < M_i : x_{ij}\}\}, \\ Y &\stackrel{\text{def}}{=} \mathbf{min}\{i : 0 \le i < N : \mathbf{max}\{j : \le j < M_i : y_{ij}\}\}. \end{aligned}$$

If $\forall i, j : 0 \le i < N \wedge 0 \le j < M_i : |x_{ij} - y_{ij}| \le B$, then $|X - Y| \le B$.

Proof: For any i, $0 \le i < N$, let $X_i = \mathbf{max}\{j : 0 \le j < M_i : x_{ij}\}$ and $Y_i = \mathbf{max}\{j : 0 \le j < M_i : y_{ij}\}$. By (4.13) of Lemma 4.6, $|X_i - Y_i| \le B$. The corollary is established by using (4.12) of Lemma 4.6 for the sets $\{X_i\}$ and $\{Y_i\}$. *Q.E.D.*

Lemma 4.8 *Let $\mathcal{X} = \langle E, R, \Delta\rangle$ be the XER-system induced by a pseudorepetitive XER-system $\mathcal{X}'' = \langle \mathcal{X}_0, \mathcal{X}'_1\rangle$. Let $\mathcal{X}_2 = \langle E_2, R_2, \Delta_2\rangle$ be the XER-system induced by $\mathcal{X}'_1$. If $\hat{t}$ and $\hat{t}_2$ are, respectively, the timing simulations of $\mathcal{X}$ and $\mathcal{X}_2$, then, there exists B such that*

$$\forall e : e \in E_2 : |\hat{t}(e) - \hat{t}_2(e)| \le B. \qquad (4.14)$$

Proof: Let $\mathcal{X}_0 = \langle E_0, R_0, \Delta_0 \rangle$ and $\mathcal{X}'_1 = \langle E'_1, R'_1, \delta_1, \theta_1 \rangle$. Let $\mathcal{E} = E_2 \cap E_0$. Since E_0 is finite, so is $\mathcal{E}$. Also, for any $\hat{f} \in (E_2 \setminus \mathcal{E})$, $\hat{f} \notin E_0$. So, by (4.10)

$$\begin{aligned} \{C : C \mapsto \hat{f} \in R_2 : C \mapsto \hat{f}\} &= \{C : C \mapsto \hat{f} \in (R_2 \lfloor E_0) : C \mapsto \hat{f}\} \\ &= \{C : C \mapsto \hat{f} \in R : C \mapsto \hat{f}\} \end{aligned} \tag{4.15}$$

with the last equality due to $R = R_0 \cup (R_2 \lfloor E_0)$ and $\hat{f} \notin E_0$.

Next, if $\mathcal{E}$ is empty, then let B be 0; else,

$$B \stackrel{\text{def}}{=} \mathbf{max}\{e : e \in \mathcal{E} : |\hat{t}(e) - \hat{t}_2(e)|\}. \tag{4.16}$$

Suppose that

$$\mathcal{Z} \stackrel{\text{def}}{=} \{e : e \in E_2 \wedge |\hat{t}(e) - \hat{t}_2(e)| > B : e\}$$

is not empty. Then, since only XER-systems with acyclic constraint graphs are considered, there exists an element $\hat{f}$ in $\mathcal{Z}$ such that

$$\forall C : C \mapsto \hat{f} \in R_2 : (\forall e : e \in C : e \notin \mathcal{Z}). \tag{4.17}$$

Clearly, by (4.16), $\hat{f} \notin \mathcal{E}$, which implies $\hat{f} \notin E_0$. So, if $\hat{f} \in \mathbf{init}(\mathcal{X}_2)$ then, by (4.15), $\hat{f} \in \mathbf{init}(\mathcal{X})$ and $\hat{t}(\hat{f}) = \hat{t}_2(\hat{f}) = 0$. Alternatively, since $\hat{f} \notin E_0$, (4.17) and (4.11) imply

$$\begin{aligned} &\forall C : C \mapsto \hat{f} \in R_2 : (\forall e : e \in C : \\ &\quad |(\hat{t}(e) + \Delta(e, \hat{f}, r)) - (\hat{t}_2(e) + \Delta(e, \hat{f}, r))| \leq B. \end{aligned} \tag{4.18}$$

By (4.15), $\hat{t}(\hat{f})$ and $\hat{t}_2(\hat{f})$ are obtained by taking the maximum and minimum over the same sets of events and rules. So, by Corollary 4.7, (4.18) implies $|\hat{t}(\hat{f}) - \hat{t}_2(\hat{f})| \leq B$ which contradicts $\hat{f} \in \mathcal{Z}$. Therefore, $\mathcal{Z}$ is empty and the lemma is established. *Q.E.D.*

4.4 Scenarios

Definition: For a repetitive XER-system $\mathcal{X}' = \langle E', R', \delta, \theta \rangle$, a *scenario* of $\mathcal{X}'$ is a conjunctive repetitive XER-system $\bar{\mathcal{X}}' = \langle E', \bar{R}', \bar{\delta}, \bar{\theta} \rangle$ where

- $\bar{R}' \subseteq R'$;

- $\overline{\delta}$ is the same[3] as δ but defined only over the restricted domain

$$\overline{\mathcal{D}}' = \{u, v, q : q \in \overline{R}' \wedge u \in \mathbf{src}(q) \wedge v = \mathbf{tar}(q) : \langle u, v, q \rangle\};$$

- $\overline{\theta}$ is the same as θ but defined only over $\overline{\mathcal{D}}'$.

Thus, a scenario corresponds to an XER-system where only one of the possible sets of causes of a transition is included. If, in $\mathcal{X}'$, the number of sets of causes for transition u is $n(u)$, then $\mathcal{X}'$ has precisely $\prod_{(u \in E')} n(u)$ scenarios.

Example 4.12: Consider the repetitive XER-system $\mathcal{X}' = \langle E', R', \delta, \theta \rangle$ where

- $E' = \{a, b, c, d, e\}$;
- $R' = \{\{d\} \mapsto a\,,\ \{a\} \mapsto b\,,\ \{b, e\} \mapsto c\,,\ \{c\} \mapsto d\,,\ \{a, b\} \mapsto e\,,\ \{b, d\} \mapsto e\}$;
- $\delta(d, e, \{b, d\} \mapsto e) = 4$, $\delta(a, e, \{a, b\} \mapsto e) = 2$, and $\delta(u, v, q) = 1$ for all other $\langle u, v, q \rangle$ in the domain of δ; and
- $\theta(d, a, \{d\} \mapsto a) = 1$, $\theta(a, b, \{a\} \mapsto b) = 1$, $\theta(d, e, \{b, d\} \mapsto e) = 1$, and $\theta(u, v, q) = 0$ for all other $\langle u, v, q \rangle$ in the domain of θ.

Because of the two possible sets of causes for e, $\mathcal{X}'$ has two scenarios: $\overline{\mathcal{X}}'_0$ with template set $\overline{R}'_0 = (R' \setminus \{\{b, d\} \mapsto e\})$ and $\overline{\mathcal{X}}'_1$ with template set $\overline{R}'_1 = (R' \setminus \{\{a, d\} \mapsto e\})$. The collapsed-constraint graphs of $\mathcal{X}'$, $\overline{\mathcal{X}}'_0$ and $\overline{\mathcal{X}}'_1$ are shown in Figure 4.4. □

4.4.1 Strongly Connected Scenarios

A scenario is said to be *strongly connected* if its collapsed-constraint graph is strongly connected[4]. To facilitate some of the proofs given in the sequel, *only XER-systems with strongly connected scenarios are considered.* The

[3] To reduce the cluttering of notation, when the domain of the function is obvious or irrelevant, δ and θ will be used, respectively, in place of $\overline{\delta}$ and $\overline{\theta}$.

[4] For the purpose of this work, the graph containing a single vertex with no arc is *not* considered to be strongly connected. This exception is needed so that all strongly connected graphs contain at least one cycle.

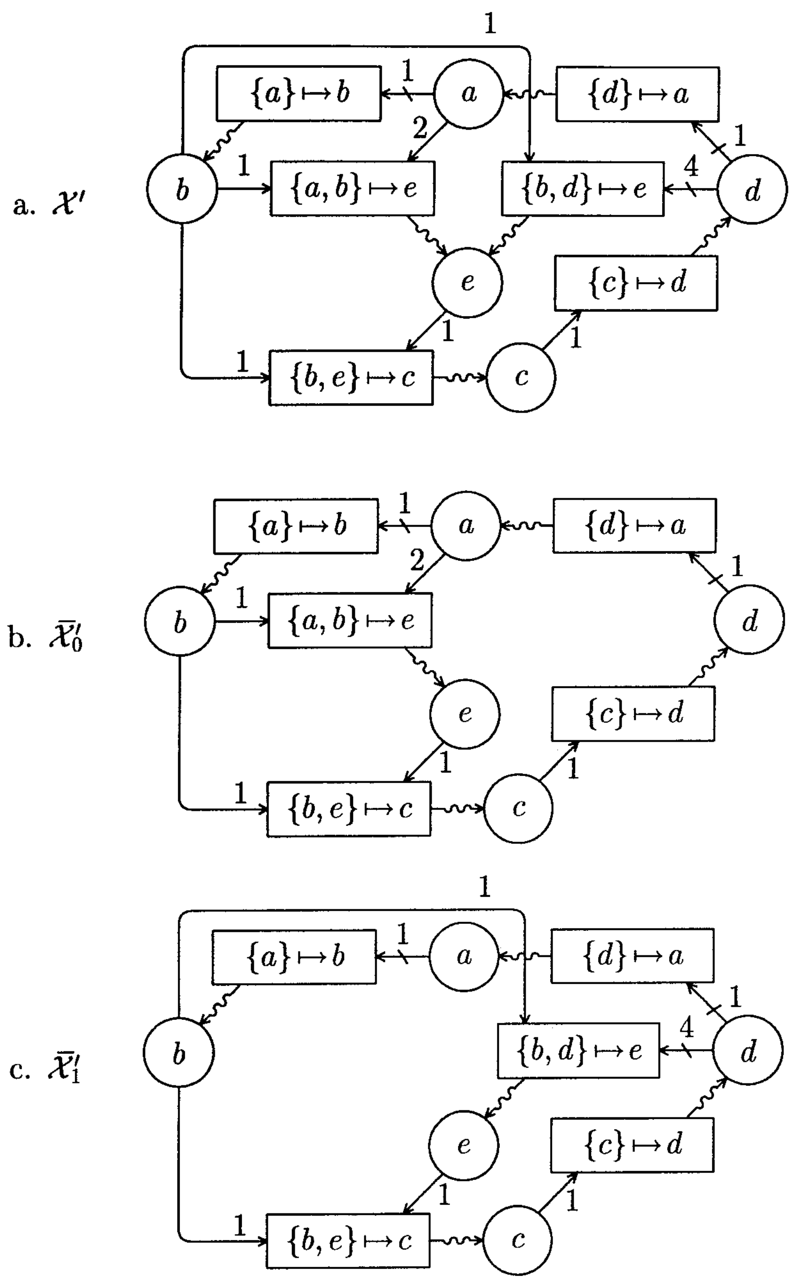

Figure 4.4: Scenarios of Example 4.12

justification for this restriction is that, for our purpose, XER-systems are used to model delay-insensitive circuits that function properly even if there are arbitrary delays between the occurrences of transitions. In other words, the delay function of a repetitive XER-system specifies the relative delays among the transitions only under a given set of circumstances. In contrast, the causality relationships embodied in the set of templates are valid under all possible delay functions. The rest of the section shows how strongly connected scenarios are necessary for delay-insensitive systems. The arguments hinge on the observation that if, in the collapsed-constraint graph, transition vertex $\hat{w}$ does not lead to transition vertex $\hat{v}$ (i.e., there is no path from $\hat{w}$ to $\hat{v}$), then, under an appropriate set of delays, $\hat{v}$ can occur an arbitrary number of times before a particular occurrence of $\hat{w}$ takes place. This observation is formalized and proved in the following lemma.

Lemma 4.9 *Let $\mathcal{X}' = \langle E', R', \delta, \theta \rangle$ be a conjunctive repetitive XER-system with collapsed-constraint graph G' and maximum occurrence-index offset θ_{max}. Let $\hat{w}$ be an element in E' such that $\emptyset \mapsto \hat{w} \notin R'$. Let $\hat{v}$ be a transition such that $\hat{w}$ does not lead to $\hat{v}$ in G'. For any event $\langle \hat{v}, \hat{\imath} \rangle$, there exists a conjunctive repetitive XER-system $\mathcal{X}'_1 = \langle E', R', \delta_1, \theta \rangle$ such that $\hat{t}_1(\hat{v}, \hat{\imath}) \leq \hat{t}_1(\hat{w}, \theta_{max})$, where $\hat{t}_1$ is the timing simulation of $\mathcal{X}'_1$.*

Proof: Let $\hat{q} \in R'$, $\mathbf{tar}(\hat{q}) = \hat{w}$, and $\hat{y} \in \mathbf{src}(\hat{q})$. Let $\hat{t}$ be the timing simulation of $\mathcal{X}'$. Define δ_1 to be the same as δ except that $\delta_1(\hat{y}, \hat{w}, \hat{q}) = \hat{t}(\hat{v}, \hat{\imath})$. Recall that we are dealing with conjunctive systems. So, by (4.8),

$$\hat{t}_1(\hat{w}, \theta_{\max}) \geq \hat{t}_1(\hat{y}, \theta_{\max} - \theta(\hat{y}, \hat{w}, \hat{q})) + \delta_1(\hat{y}, \hat{w}, \hat{q}) \geq \hat{t}(\hat{v}, \hat{\imath}). \tag{4.19}$$

Next, let $\mathcal{X}$ be the XER-system induced by $\mathcal{X}'$. Let G be the constraint graph of $\mathcal{X}$. Let

$$\mathcal{U} = \{v, i : (\forall k :: \langle \hat{w}, k \rangle \text{ does not lead to } \langle v, i \rangle \text{ in } G) : \langle v, i \rangle\}.$$

Suppose $\mathcal{Z} \stackrel{\text{def}}{=} \{v, i : \langle v, i \rangle \in \mathcal{U} \wedge \hat{t}(v, i) \neq \hat{t}_1(v, i) : \langle v, i \rangle\}$ is not empty. Then, since G is acyclic, there exists $\langle v, i \rangle \in \mathcal{Z}$ such that for any $\langle u, j \rangle$ and r,

$$\langle u, j \rangle \in \mathbf{src}(r) \wedge \langle v, i \rangle = \mathbf{tar}(r) \Rightarrow \langle u, j \rangle \notin \mathcal{Z}. \tag{4.20}$$

If $\langle v, i \rangle \in \mathbf{init}(\mathcal{X})$, then $\hat{t}(v, i) = \hat{t}_1(v, i) = 0$. Else, let $\langle u, j \rangle$ and r satisfy the premise of (4.20). If $\langle u, j \rangle \notin \mathcal{U}$, then there exists k such that $\langle \hat{w}, k \rangle$ leads to $\langle u, j\rangle$ which implies $\langle \hat{w}, k\rangle$ leads to $\langle v, i\rangle$, contradicting $\langle v, i\rangle \in \mathcal{Z}$. Hence, (4.20) is equivalent to

$$\langle u, j\rangle \in \mathbf{src}(r) \wedge \langle v, i\rangle = \mathbf{tar}(r) \Rightarrow \hat{t}(u, j) = \hat{t}_1(u, j). \tag{4.21}$$

Let $r = q\lceil i$. Then, since $v \neq \hat{w}$ due to $\langle v, i\rangle \in \mathcal{U}$, $\delta(u, v, q) = \delta_1(u, v, q)$. So, by the definition of timing simulations, (4.21) implies $\hat{t}(v, i) = \hat{t}_1(v, i)$ which is a contradiction. Hence, $\mathcal{Z}$ is empty. But, by Lemma 4.4, $\langle \hat{v}, \hat{\imath}\rangle \in \mathcal{U}$; so, $\hat{t}(\hat{v}, \hat{\imath}) = \hat{t}_1(\hat{v}, \hat{\imath})$ and the lemma follows from (4.19). *Q.E.D.*

With this result, we can now argue that delay-insensitive circuits are modeled by XER-systems with strongly connected scenarios. First, consider a conjunctive repetitive XER-systems $\mathcal{X}'$. Suppose G', the collapsed-constraint graph of $\mathcal{X}'$, has two strongly connected components H_0 and H_1. If a transition $\hat{v}$ in H_0 is in the cause set of a transition $\hat{w}$ in H_1, then $\hat{w}$ does not lead to $\hat{v}$ by the definition of components. But then, by Lemma 4.9, there exists a set of delays such that $\hat{v}$ occurs arbitrarily often before $\langle \hat{w}, \theta_{\max}\rangle$ takes place. Since there is only one cause set for $\hat{w}$, this situation implies instability in the circuit being modeled. Hence, for a stable system, transitions in one strongly connected component do not interact with those in another. Consequently, it is sufficient to analyze each component independently and the requirement that a conjunctive XER-system be strongly connected is justified.

If $\mathcal{X}'$ is not conjunctive, then let $\bar{\mathcal{X}}'$ be one of its scenarios. For each $q = C' \mapsto v$ where q is a template in $\mathcal{X}'$ but not in $\bar{\mathcal{X}}'$, define the delay function of $\mathcal{X}'$ so that the delay between any transition in C' and v is very large. In the timing simulation of $\mathcal{X}'$, each event occurs as soon as the timing constraints imposed by one of its cause sets have been fulfilled. Thus, under the new delay function, this cause set is always induced from $\bar{\mathcal{X}}'$ and, therefore, the timing simulations of $\mathcal{X}'$ and $\bar{\mathcal{X}}'$ are identical. To avoid the instabilities previously mentioned for conjunctive XER-systems, $\bar{\mathcal{X}}'$ needs to be strongly connected. Since the choice of $\bar{\mathcal{X}}'$ is arbitrary, if an XER-system models a QDI circuit, then all of its scenarios are strongly connected.

4.5 Linear Timing Function

Definition: A timing function $\bar{t}$ of a repetitive XER-system $\mathcal{X}'$ is said to be *linear* if it has the following form:

$$\bar{t}(u,i) = h(u) + p\,i \tag{4.22}$$

where h is a function of u and p does not depend on u or i. The *period* of the timing function is p.

Note that this definition differs slightly from the one given in [6] where a linear timing function has the form:

$$\bar{t}(u,i) = h(u) + p_u\,i. \tag{4.23}$$

Burns then shows that if u and v belong to the same strongly connected component of the corresponding collapsed-constraint graph G', then p_u and p_v are identical. He then restricts his attention only to strongly connected systems. Since this restriction has already been made in this paper, it is more convenient to use (4.22) as the definition of linear timing function.

4.5.1 Linear Offset Functions

As Lemma 4.10 below shows, the timing constraints of (4.1) can be equivalently expressed in terms of the function h as given in (4.22). In fact, the latter expression is more convenient to work with since it is independent of i. Thus, the following definition is made.

Definition: Let $\mathcal{X}'$ be a repetitive XER-system with transition set E'. Given a number p, a *linear offset function* of $\mathcal{X}'$ with period p is any function $h : E' \rightarrow [0,\infty)$ such that $\bar{t}$ defined by (4.22) is a linear timing function of $\mathcal{X}'$. The *size* of h is $|h| = \sum_{(u \in E')} h(u)$.

Lemma 4.10 *Let $\mathcal{X}' = \langle E', R', \delta, \theta\rangle$ be a repetitive XER-system. Let p be any positive number and h be any function mapping elements of E' to $[0,\infty)$. Then, h is a linear offset function of $\mathcal{X}'$ with period p if and only if*

$$\begin{aligned}&\forall v : v \in E' : (\ \exists q : q \in R' \wedge \mathbf{tar}(q) = v : \\ &\qquad (\forall u : u \in \mathbf{src}(q) : h(v) \geq h(u) - p\theta(u,v,q) + \delta(u,v,q))\ ).\end{aligned} \tag{4.24}$$

Proof: Let $\mathcal{X} = \langle E, R, \Delta \rangle$ be the XER-system induced by $\mathcal{X}'$. Let $\bar{t}$ be defined by (4.22). Then, by the construction of R, $\bar{t}$ is a linear timing function if and only if

$$\begin{array}{l}\forall \langle v,i\rangle : \langle v,i\rangle \in E : (\ \exists q\lceil i : q\lceil i \in R \wedge \mathbf{tar}(q\lceil i) = \langle v,i\rangle : \\ \quad (\forall \langle u, i-\theta(u,v,q)\rangle : \langle u, i-\theta(u,v,q)\rangle \in \mathbf{src}(q\lceil i) : \\ \quad\quad \bar{t}(v,i) \geq \bar{t}(u, i-\theta(u,v,q)) + \Delta(\langle u, i-\theta(u,v,q)\rangle, \langle v,i\rangle, q\lceil i))).\end{array} \tag{4.25}$$

Now, $\langle u, i-\theta(u,v,q)\rangle \in \mathbf{src}(q\lceil i) \Rightarrow u \in \mathbf{src}(q)$. Hence, by the definition of Δ, if (4.24) holds, then (4.25) holds and h is a linear offset function. Conversely, if h is a linear offset function, then (4.25) holds. In particular, if $i = \theta_{\max}$, then (4.25) implies (4.24) by (4.8). Hence, the lemma is established. *Q.E.D.*

4.6 Minimum-Period Linear Timing Functions

Definition: A *minimum-period linear timing function (MPLTF)* of a repetitive XER-system $\mathcal{X}'$ is a linear timing function $\bar{t}$ of $\mathcal{X}'$ whose period is minimum — i.e., $\mathcal{X}'$ has no linear timing function with smaller period. The *minimum period* of $\mathcal{X}'$, denoted $\mathbf{period}(\mathcal{X}')$, is the period of its MPLTF.

The following lemma is a rephrasing of one of the main results of [6].

Lemma 4.11 *If $\mathcal{X}'$ is a conjunctive repetitive XER-system whose collapsed-constraint graph G' is strongly connected, then a MPLTF of $\mathcal{X}'$ exists and* $\mathbf{period}(\mathcal{X}')$ *is*

$$p = \mathbf{max}\{\rho' : \rho' \text{ is a cycle in } G' : \frac{\delta(\rho')}{\theta(\rho')}\}. \tag{4.26}$$

Proof: Convert $\mathcal{X}'$ into the equivalent repetitive ER-system $\mathcal{Y}' = \langle E', R'_{\mathcal{Y}} \rangle$ as shown in Lemma 4.3. Then, see Section 2.4 of [6] for the rest of the proof. Below, a brief outline of that proof is given to illustrate the approach used and to present some intermediate results which will be needed in the sequel.

First, note that, by construction, (4.10) is equivalent to

$$\forall v : v \in E' : (\forall u, \alpha, \varepsilon : \langle u, v, \alpha, \varepsilon \rangle \in R'_{\mathcal{Y}} : h(v) \geq h(u) - p\varepsilon + \alpha). \tag{4.27}$$

Next, in $\mathcal{Y}'$, order the elements of E' as $\{u_1, u_2, \ldots, u_n\}$ and those of $R'_{\mathcal{Y}}$ as $\{r_1, r_2, \ldots, r_m\}$. Let template r_j be $\langle u_k, u_l, \alpha_j, \varepsilon_j \rangle$, with u_k and u_l being, respectively, the *source* and *target* of r_j. Let ε be the transpose of $\langle \varepsilon_1, \varepsilon_2, \ldots, \varepsilon_m \rangle$ and α be the transpose of $\langle \alpha_1, \alpha_2, \ldots, \alpha_m \rangle$. Construct A', the *arc-node incidence matrix* of $\mathcal{Y}'$, by

$$a'_{jk} = \begin{cases} -1 & \text{if } u_k \text{ is the source of } r_j \\ 1 & \text{if } u_k \text{ is the target of } r_j \\ 0 & \text{otherwise.} \end{cases}$$

Let x represents the vector $\langle h(u_1), h(u_2), \ldots, h(u_n) \rangle$. Then, the constraint expressed in (4.27) is equivalent to (4.29) below and, therefore, finding the MPLTF for $\mathcal{Y}'$ is equivalent to solving

$$z = \mathbf{min}\, p \tag{4.28}$$

$$[A' \ \varepsilon] \begin{bmatrix} x \\ p \end{bmatrix} \geq \alpha \tag{4.29}$$

$$x, p \geq 0. \tag{4.30}$$

The fact that this solution exists and is related to the cycles by (4.26) is given in [6]. *Q.E.D.*

Note that p in (4.26) is well-defined because of the following reasons. Let u be a transition in a cycle ρ'. Then, by Lemma 4.5, for all i sufficiently large, there is a path from $\langle u, i - \theta(\rho') \rangle$ to $\langle u, i \rangle$ in G, the constraint graph of the XER-system induced by $\mathcal{X}'$. If $\theta(\rho') = 0$, then G is cyclic. If $\theta(\rho') < 0$, then G has a vertex with an ancestor at an infinite distance (See Example 4.4). Since both these cases have been excluded, $\theta(\rho')$ is positive and (4.26) is consistent.

Example 4.13: Consider $\bar{\mathcal{X}}'_0$ of Example 4.12. As shown in Figure 4.4b, there are the following three simple cycles in the collapsed-constraint graph of $\bar{\mathcal{X}}'_0$:

- $\rho_0 = \langle a, \{a, b\} \mapsto e, e, \{b, e\} \mapsto c, c, \{c\} \mapsto d, d, \{d\} \mapsto a, a \rangle$,
- $\rho_1 = \langle a, \{a\} \mapsto b, b, \{b, e\} \mapsto c, c, \{c\} \mapsto d, d, \{d\} \mapsto a, a \rangle$, and
- $\rho_2 = \langle a, \{a\} \mapsto b, b, \{a, b\} \mapsto e, e, \{b, e\} \mapsto c, c, \{c\} \mapsto d, d, \{d\} \mapsto a, a \rangle$.

Since

$$\frac{\delta(\rho_0)}{\theta(\rho_0)} = \tfrac{5}{1} = 5, \quad \frac{\delta(\rho_1)}{\theta(\rho_1)} = \tfrac{4}{2} = 2, \quad \text{and} \quad \frac{\delta(\rho_2)}{\theta(\rho_2)} = \tfrac{5}{2} = 2.5,$$

$\mathbf{period}(\bar{\mathcal{X}}'_0) = 5$. Indeed, $\bar{t}$ defined by

$$\begin{array}{lll} \bar{t}(a,i) = 5i, & \bar{t}(b,i) = 5i, & \bar{t}(c,i) = 3 + 5i, \\ \bar{t}(d,i) = 4 + 5i, & \bar{t}(e,i) = 2 + 5i, & \end{array} \tag{4.31}$$

is a MPLTF of $\bar{\mathcal{X}}'_0$. Similar analysis shows that the minimum period of $\bar{\mathcal{X}}'_1$ in Example 4.12 is 6. □

To facilitate the generalization of Lemma 4.11 to non-conjunctive systems, the following definition is made.

Definition: A *minimum-period linear offset function (MPLOF)* of $\mathcal{X}'$ is a linear offset function h of $\mathcal{X}'$ with period p such that $\bar{t}$ defined by

$$\bar{t}(u,i) = h(u) + pi \tag{4.32}$$

is a MPLTF of $\mathcal{X}'$.

By the relationship dictated by (4.32), the MPLOF of $\mathcal{X}'$ is the linear offset function whose period is minimum.

Theorem 4.1 *Let $\mathcal{X}'$ be a repetitive XER-system whose scenarios are strongly connected. Then, a MPLTF of $\mathcal{X}'$ exists and* $\mathbf{period}(\mathcal{X}')$ *is*

$$p = \mathbf{min}\{\bar{\mathcal{X}}' : \bar{\mathcal{X}}' \text{ is a scenario of } \mathcal{X}' : \mathbf{period}(\bar{\mathcal{X}}')\}. \tag{4.33}$$

Proof: Let the scenarios of $\mathcal{X}'$ be $\bar{\mathcal{X}}'_0$, $\bar{\mathcal{X}}'_1$, ..., $\bar{\mathcal{X}}'_M$. For any positive number $\tilde{p}$, let $T_{\tilde{p}}(\bar{\mathcal{X}}'_m)$ denote the set of linear offset functions for $\bar{\mathcal{X}}'_m$ with period $\tilde{p}$ and likewise for $T_{\tilde{p}}(\mathcal{X}')$. If $h \in T_{\tilde{p}}(\bar{\mathcal{X}}'_m)$, then, by Lemma 4.10, for any v, there exists a unique template q_v in $\bar{\mathcal{X}}'_m$ such that

$$\forall u : u \in \mathbf{src}(q_v) : h(v) \geq h(u) - \tilde{p}\theta(u, \hat{v}, q_v) + \delta(u, \hat{v}, q_v). \tag{4.34}$$

But q_v is also a template in $\mathcal{X}'$; so, $h \in T_{\tilde{p}}(\mathcal{X}')$. Conversely, if h is in $T_{\tilde{p}}(\mathcal{X}')$, then for any v, there exists a template q_v in $\mathcal{X}'$ such that (4.34) holds. Let

$\bar{\mathcal{X}}'_m$ be the scenario whose set of templates is precisely $\{v : v \in E' : q_v\}$. Then, $h \in T_{\tilde{p}}(\bar{\mathcal{X}}'_m)$. So, for any positive $\tilde{p}$,

$$T_{\tilde{p}}(\mathcal{X}') = \bigcup_{m=0}^{M} T_{\tilde{p}}(\bar{\mathcal{X}}'_m). \tag{4.35}$$

By the definition of MPLOF, p is the smallest $\tilde{p}$ such that the RHS of (4.35) is non-empty. Hence, it is also the smallest $\tilde{p}$ such that the LHS is non-empty. So, by the remarks after the definition of MPLOF, p is the period of a MPLOF of $\mathcal{X}'$ and the theorem is established. *Q.E.D.*

Example 4.14: By the analysis done in Example 4.13, the minimum period of $\mathcal{X}'$ of Example 4.12 is 5. In fact, (4.31) is a MPLTF of $\mathcal{X}'$. □

4.6.1 Critical Scenarios, Cycles, and Transitions

Theorem 4.1 states that the minimum period of a repetitive XER-system is determined by a particular cycle in the collapsed-constraint graph of a particular scenario. For future reference, we introduce the following definition.

Definition: A scenario $\bar{\mathcal{X}}'$ of a repetitive XER-system $\mathcal{X}'$ is *critical* if $\mathbf{period}(\bar{\mathcal{X}}') = \mathbf{period}(\mathcal{X}')$. A *critical cycle* in a scenario $\bar{\mathcal{X}}'$ is a cycle $\hat{\rho}'$ in the collapsed-constraint graph of $\bar{\mathcal{X}}'$ such that

$$\frac{\delta(\hat{\rho}')}{\theta(\hat{\rho}')} = \mathbf{period}(\bar{\mathcal{X}}').$$

A *critical cycle* in a repetitive XER-system $\mathcal{X}'$ is any critical cycle in any critical scenario of $\mathcal{X}'$. Also, the *set of critical transitions* of $\bar{\mathcal{X}}'$ is

$$\begin{aligned} \mathcal{C}(\bar{\mathcal{X}}') \stackrel{\text{def}}{=} \{&w, \rho' : \\ &\rho' \text{ is a critical cycle of } \bar{\mathcal{X}}' \wedge w \text{ is a transition vertex in } \rho' : w\}. \end{aligned} \tag{4.36}$$

Example 4.15: For $\mathcal{X}'$ of Example 4.12, $\bar{\mathcal{X}}'_0$ is a critical scenario. A critical cycle of $\bar{\mathcal{X}}'_0$ is

$$\langle a, \{a, b\} \mapsto e, e, \{b, e\} \mapsto c, c, \{c\} \mapsto d, d, \{d\} \mapsto a, a\rangle,$$

and a critical cycle of $\bar{\mathcal{X}}'_1$ is

$$\langle e, \{b, e\} \mapsto c, c, \{c\} \mapsto d, d, \{b, d\} \mapsto e, e \rangle.$$

The former is also a critical cycle of $\mathcal{X}'$. Furthermore, $\mathcal{C}(\bar{\mathcal{X}}'_0) = \{a, c, d, e\}$ and $\mathcal{C}(\bar{\mathcal{X}}'_1) = \{c, d, e\}$. $\square$

Let $\rho' = \langle u_0, q_0, u_1, q_1, \ldots, u_{l-1}, q_{l-1}, u_0 \rangle$ be a critical cycle of a scenario $\bar{\mathcal{X}}'_m$. Note that ρ' may not be a cycle in the collapsed-constraint graph of another scenario $\bar{\mathcal{X}}'$, if one of the q_j's belongs to the scenario $\bar{\mathcal{X}}'_m$ but not to $\bar{\mathcal{X}}'$. In determining the minimum period of an XER-system, the following lemma shows that once the critical cycles of a scenario have been found, it is only necessary to analyze other scenarios for which none of these critical cycles are present. By repeatedly applying this observation, the amount of computation required may be greatly reduced.

Lemma 4.12 *Let $\bar{\mathcal{X}}'_m$ be any scenario of a repetitive XER-system $\mathcal{X}'$. Let $\mathcal{R}_m$ be the set of critical cycles of $\bar{\mathcal{X}}'_m$. Then,* **period**$(\mathcal{X}')$ = **period**$(\bar{\mathcal{X}}'_m)$ *unless there exists a critical scenario $\bar{\mathcal{X}}'$, such that none of the elements in $\mathcal{R}_m$ is a cycle in G', the collapsed-constraint graph of $\bar{\mathcal{X}}'$.*

Proof: If any element of $\mathcal{R}_m$ is a cycle in G', then, by Lemma 4.11, **period**$(\bar{\mathcal{X}}') \geq$ **period**$(\bar{\mathcal{X}}'_m)$. *Q.E.D.*

4.7 MPLTF's and Timing Simulations

Throughout this section, except in the examples or when stated otherwise, $\mathcal{X}' = \langle E', R', \delta, \theta \rangle$ is a repetitive XER-system with period p and $\bar{\mathcal{X}}' = \langle E', \bar{R}', \delta, \theta \rangle$ is any of its critical scenarios. The purpose of this section is to prove that a MPLTF of $\mathcal{X}'$ is a "good" approximation to its timing simulation. The following example illustrates some of the issues involved.

Example 4.16: A linear timing function $\bar{t}$ (as defined in (4.31)) and the timing simulation $\hat{t}$ of the repetitive XER-system $\mathcal{X}'$ in Example 4.12 are shown in Figure 4.5.

Note that the difference between $\bar{t}(b, i)$ and $\hat{t}(b, i)$ starts at 0, then increases to 4, and then remains at 5, which is **period**$(\mathcal{X}')$, for all $i \geq 2$. Thus, for large i, $\hat{t}(b, i+1) - \hat{t}(b, i) = \bar{t}(b, i+1) - \bar{t}(b, i) =$ **period**$(\mathcal{X}')$. Hence,

$$
\begin{array}{ll}
\bar{t}(a,0) = 0, & \hat{t}(a,0) = 0, \\
\bar{t}(b,0) = 0, & \hat{t}(b,0) = 0, \\
\bar{t}(e,0) = 2, & \hat{t}(e,0) = \mathbf{min}\{\mathbf{max}\{\hat{t}(b,0)+1\}, \\
 & \qquad\qquad \mathbf{max}\{\hat{t}(a,0)+2, \hat{t}(b,0)+1\}\} = 1, \\
\bar{t}(c,0) = 3, & \hat{t}(c,0) = \mathbf{max}\{\hat{t}(b,0)+1, \hat{t}(e,0)+1\} = 2, \\
\bar{t}(d,0) = 4, & \hat{t}(d,0) = \hat{t}(c,0)+1 = 3, \\
\bar{t}(a,1) = 5, & \hat{t}(a,1) = \hat{t}(d,0)+1 = 4, \\
\bar{t}(b,1) = 5, & \hat{t}(b,1) = \hat{t}(a,0)+1 = 1, \\
\bar{t}(e,1) = 7, & \hat{t}(e,1) = \mathbf{min}\{\mathbf{max}\{\hat{t}(b,1)+1, \hat{t}(d,0)+4\}, \\
 & \qquad\qquad \mathbf{max}\{\hat{t}(a,1)+2, \hat{t}(b,1)+1\}\} = 6, \\
\bar{t}(c,1) = 8, & \hat{t}(c,1) = \mathbf{max}\{\hat{t}(b,1)+1, \hat{t}(e,1)+1\} = 7, \\
\bar{t}(d,1) = 9, & \hat{t}(d,1) = \hat{t}(c,1)+1 = 8, \\
\bar{t}(a,2) = 10, & \hat{t}(a,2) = \hat{t}(d,1)+1 = 9, \\
\bar{t}(b,2) = 10, & \hat{t}(b,2) = \hat{t}(a,1)+1 = 5, \\
\bar{t}(e,2) = 12, & \hat{t}(e,2) = \mathbf{min}\{\mathbf{max}\{\hat{t}(b,2)+1, \hat{t}(d,1)+4\}, \\
 & \qquad\qquad \mathbf{max}\{\hat{t}(a,2)+2, \hat{t}(b,2)+1\}\} = 11, \\
\bar{t}(c,2) = 13, & \hat{t}(c,2) = \mathbf{max}\{\hat{t}(b,2)+1, \hat{t}(e,2)+1\} = 12, \\
\bar{t}(d,2) = 14, & \hat{t}(d,2) = \hat{t}(c,2)+1 = 13, \\
\bar{t}(a,3) = 15, & \hat{t}(a,3) = \hat{t}(d,3)+1 = 14, \\
\bar{t}(b,3) = 15, & \hat{t}(b,3) = \hat{t}(a,1)+1 = 10, \\
\ldots &
\end{array}
$$

Figure 4.5: Timing functions for Example 4.16

at least in this example, **period**($\mathcal{X}'$) serves as an indicator of the periodic performance of $\mathcal{X}'$.

Also, observe that $q_0 = \{b,d\} \mapsto e$ is the template such that

$$\hat{t}(e,0) = \mathbf{max}\{\langle u,j\rangle : \langle u,j\rangle \in q_0\lceil 0 : \hat{t}(u,j) + \Delta(\langle u,j\rangle, \langle e,i\rangle, q_0\lceil 0)\} \quad (4.37)$$

whereas $q_1 = \{a,b\} \mapsto e$ is the template for which

$$\hat{t}(e,i) = \mathbf{max}\{\langle u,j\rangle : \langle u,j\rangle \in q_1\lceil i : \hat{t}(u,j) + \Delta(\langle u,j\rangle, \langle e,i\rangle, q_1\lceil i)\} \quad (4.38)$$

holds for all $i > 0$. Thus, in this example, except for some initial occurrences, the timing simulation of $\mathcal{X}'$ is dictated by the constraints of a single scenario.
□

Our approach is to prove the existence of a critical scenario $\bar{\mathcal{X}}'$ — like the one containing template q_1 in the previous example — such that, eventually, the behavior of $\mathcal{X}'$ is "close" to that of $\bar{\mathcal{X}}'$.

4.7.1 The "Smallest" MPLOF of a Critical Scenario

As an intermediate step in proving that a MPLTF is a good approximation of the timing simulation, we will show that there exists a MPLOF h of $\bar{\mathcal{X}}'$ such that, for any transition v and the unique template q_v in $\bar{\mathcal{X}}'$ with $\mathbf{tar}(q_v) = v$,

$$h(v) = \mathbf{max}\{u : u \in \mathbf{src}(q_v) : h(u) - p\theta(u, v, q_v) + \delta(u, v, q_v)\}. \tag{4.39}$$

The example below illustrates that it is not trivial to prove that such a MPLOF exists.

Example 4.17: Consider again $\bar{\mathcal{X}}'_0$ of Example 4.13. For any u in $\mathbf{src}(q_v)$, define the *slack* of the pair $\langle u, v\rangle$ as

$$s(u, v) = h(v) - (h(u) - p\theta(u, v, q_v) + \delta(u, v, q_v)).$$

Figure 4.6a depicts a MPLOF h that results from minimizing the sum of all slacks in $\bar{\mathcal{X}}'_0$: Each transition vertex u is labeled with $h(u)$ and the edges in $\langle u, q_v, v\rangle$ is shown in bold if $u \in \mathbf{src}(q_v) \wedge s(u, v) = 0$. Note that (4.39) is not satisfied for $v = b$ (b is the leftmost transition vertex).

Alternatively, if we find the smallest MPLOF h satisfying $h(b) = 5$, then the MPLOF depicts in Figure 4.6b results; if we find the smallest MPLOF h satisfying $h(a) = 0$, then the MPLOF depicts in Figure 4.6c results. In both cases, (4.39) is violated by $v = b$.

Note, however, that if we find the smallest MPLOF h that satisfies $h(a) = 5$, then, as shown in Figure 4.6d, (4.39) is satisfied for all v. $\square$

In Lemma 4.17, the existence of a MPLOF h of $\bar{\mathcal{X}}'$ satisfying (4.39) will be demonstrated[5]. However, some results need to be established beforehand.

Lemma 4.13 *Let G' be the collapsed-constraint graph of $\bar{\mathcal{X}}'$. Let h be a MPLOF of $\bar{\mathcal{X}}'$. If*

$$\rho' = \langle u_0, q_2, u_1, q_1, \ldots, u_{l-1}, q_{l-1}, u_l\rangle$$

[5]There are simpler methods to prove that such a MPLOF exists (e.g. Lemma 4.18); however, the MPLOF guaranteed by Lemma 4.17 satisfies several properties which are needed in the subsequent arguments.

a

b

c

d

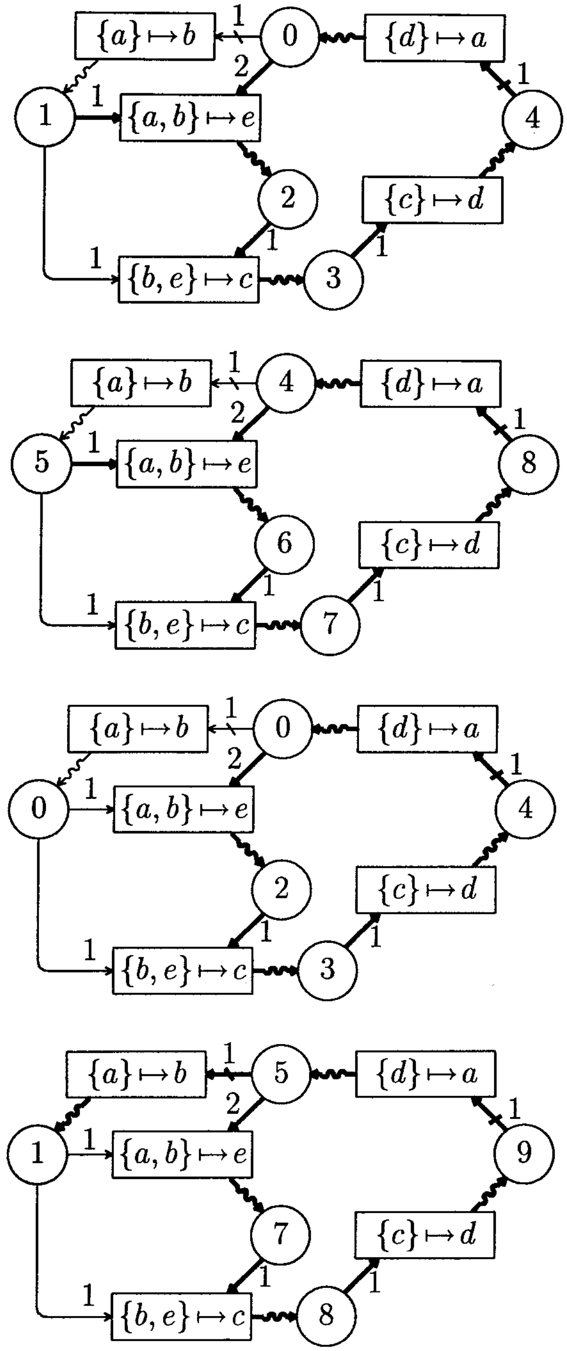

Figure 4.6: MPLOF of Example 4.17

is a path in G' where u_0 and u_l are transition vertices, then

$$h(u_l) \geq h(u_0) - p\theta(\rho') + \delta(\rho'). \tag{4.40}$$

Proof: Since G' is bipartite, each u_j a transition vertex and each q_j a template vertex. Note that since $\mathcal{X}'$ is conjunctive, q_j is the only template with target u_{j+1}. So, Lemma 4.10 implies that for all j, $0 \leq j < l$,

$$h(u_{j+1}) \geq h(u_j) - p\theta(u_j, u_{j+1}, q_j) + \delta(u_j, u_{j+1}, q_j).$$

Summing along the conjunctive edges along the path ρ' and recalling that the weight and the occurrence-index offset of the disjunctive edge $\langle q_j, u_{j+1}\rangle$ are zeros establish (4.40). *Q.E.D.*

Lemma 4.14 *For any MPLOF h of $\mathcal{X}'$, there exists a constant $\mathcal{B}$, independent of h, such that*

$$\forall u, v : (u \in E') \wedge (v \in E') : h(u) - h(v) \leq (\mathcal{B} - 1). \tag{4.41}$$

Proof: For now, assume $\mathcal{X}'$ is conjunctive and let G' be its collapsed-constraint graph. Let w be a fixed transition vertex. Since only strongly connected graphs are considered, for every v in E', there exists in G' a path $\rho'_{w,v}$ from w to v. So, by Lemma 4.13, $h(w) - h(v)$ is bounded above by a value that depends only on p and the sums of weights and occurrence-index offsets along a fixed path $\rho'_{w,v}$. By setting $\mathcal{B}_w$ to the maximum of these bounds over all v, and $\mathcal{B}$ to the maximum of the $\mathcal{B}_w$'s over all transitions w, the lemma is established if $\mathcal{X}'$ is conjunctive.

If $\mathcal{X}'$ is not conjunctive, then, by the arguments above, for each critical scenario $\bar{\mathcal{X}}'_m$ of $\mathcal{X}'$, there exists $\mathcal{B}_m$ such that (4.41) with $\mathcal{B}_m$ in place of $\mathcal{B}$ holds. Since, by (4.35), h is a MPLOF of $\mathcal{X}'$ implies h is a MPLOF of a critical scenario of $\mathcal{X}'$, setting $\mathcal{B}$ to the maximum of the $\mathcal{B}_m$'s establishes the lemma. *Q.E.D.*

Lemma 4.15 *In a scenario $\bar{\mathcal{X}}'$ with period p, let*

$$\hat{\rho}' = \langle u_0, q_0, u_1, q_1, \ldots, u_{l-1}, q_{l-1}, u_l\rangle \tag{4.42}$$

be a critical cycle with $u_l = u_0$. Then, for any MPLOF h of $\bar{\mathcal{X}}'$ and any j such that $0 \leq j < l$,

$$h(u_{j+1}) = h(u_j) - p\theta(u_j, u_{j+1}, q_j) + \delta(u_j, u_{j+1}, q_j). \tag{4.43}$$

Proof: Let σ' be the subpath of $\hat{\rho}'$ from u_1 to $u_l = u_0$. By Lemma 4.13,

$$h(u_0) \geq h(u_1) - p\theta(\sigma') + \delta(\sigma'). \tag{4.44}$$

By definition, $0 = p\theta(\hat{\rho}') - \delta(\hat{\rho}')$. So, adding this identity to (4.44) yields

$$h(u_0) \geq h(u_1) + p(\theta(\hat{\rho}') - \theta(\sigma')) - (\delta(\hat{\rho}') - \delta(\sigma')). \tag{4.45}$$

But $\langle u_0, q_0, u_1 \rangle$ is the only difference between $\hat{\rho}'$ and σ'; so,

$$h(u_0) \geq h(u_1) + p\theta(u_0, u_1, q_0) - \delta(u_0, u_1, q_0). \tag{4.46}$$

Now, since $\mathcal{X}'$ is conjunctive, q_0 is the only template whose target is u_1. Therefore, since $u_0 \in \mathbf{src}(q_0)$, by Lemma 4.10, (4.46) is an equality. Hence, (4.43) holds for $j = 0$. Since the critical cycle can starts with any u_j, the lemma is established. *Q.E.D.*

Lemma 4.16 *Let $\hat{v}$ be a transition of $\mathcal{X}'$. Let $h^\star$ be a MPLOF of $\mathcal{X}'$ and ν a positive number such that $h^\star(\hat{v}) > \nu$. If*

$$\begin{array}{l} \exists q : q \in R' \wedge \mathbf{tar}(q) = \hat{v} : \\ \qquad (\forall u : u \in \mathbf{src}(q) : \nu \geq h^\star(u) - p\theta(u, \hat{v}, q) + \delta(u, \hat{v}, q)), \end{array} \tag{4.47}$$

then h' defined by

$$h'(u) = \begin{cases} h^\star(u) & \text{if } u \neq \hat{v} \\ \nu & \text{if } u = \hat{v}. \end{cases} \tag{4.48}$$

is a MPLOF of $\mathcal{X}'$.

Proof: Let v be a transition other than $\hat{v}$. For any u and q such that $u \in \mathbf{src}(q)$ and $v = \mathbf{tar}(q)$,

$$h^\star(v) \geq h^\star(u) - p\theta(u, v, q) + \delta(u, v, q) \tag{4.49}$$

implies

$$h'(v) \geq h'(u) - p\theta(u, v, q) + \delta(u, v, q) \tag{4.50}$$

since the LHS of (4.49) equal the LHS of (4.50) and the RHS of (4.49) is not less than the RHS of (4.50). This observation and (4.47) establish the fact that (4.24) holds if h is replaced by h'. Hence, by Lemma 4.10, h' is a MPLOF of $\mathcal{X}'$. *Q.E.D.*

We are now ready to prove the main result of this sub-section.

Lemma 4.17 *Let $\mathcal{C}(\bar{\mathcal{X}}')$ be as defined by (4.36). Let $\mathcal{B}$ satisfy (4.41). Let $h(\mathcal{C}(\bar{\mathcal{X}}')) \geq \mathcal{B}$ denote the predicate*

$$\forall w : w \in \mathcal{C}(\bar{\mathcal{X}}') : h(w) \geq \mathcal{B}. \tag{4.51}$$

Then, there exists a MPLOF $h^\star$ of $\bar{\mathcal{X}}'$ such that

$$h^\star(\mathcal{C}(\bar{\mathcal{X}}')) \geq \mathcal{B}, \qquad \text{and} \tag{4.52}$$

$$|h^\star| = \mathbf{min}\{h : h \text{ is a MPLOF of } \bar{\mathcal{X}}' \text{ and } h(\mathcal{C}(\bar{\mathcal{X}}')) \geq \mathcal{B} \ : |h|\}. \tag{4.53}$$

Furthermore, for any transition $\hat{v}$, let $q_{\hat{v}}$ denote the unique template in $\bar{\mathcal{X}}'$ with target $\hat{v}$. Then, for all $\hat{v}$,

$$h^\star(\hat{v}) = \mathbf{max}\{u : u \in \mathbf{src}(q_{\hat{v}}) : h^\star(u) - p\theta(u, \hat{v}, q_{\hat{v}}) + \delta(u, \hat{v}, q_{\hat{v}})\}. \tag{4.54}$$

Proof: Convert $\bar{\mathcal{X}}'$ into the equivalent repetitive ER-system $\mathcal{Y}' = \langle E', R'_{\mathcal{Y}} \rangle$ as shown in Lemma 4.3. Let $|E'| = n$ and $|R'_{\mathcal{Y}}| = m$. For $\mathcal{Y}'$, define the arc-node incidence matrix A', the vector x, and the column-vectors ε and α as done in the proof of Lemma 4.11. Let β be the n-dimensional column-vector defined by

$$\beta_k = \begin{cases} \mathcal{B} & \text{if } u_k \in \mathcal{C}(\bar{\mathcal{X}}') \\ 0 & \text{if } u_k \notin \mathcal{C}(\bar{\mathcal{X}}'). \end{cases}$$

Let $c^{\mathbf{T}} = \langle 1, 1, \ldots, 1 \rangle$. Then, since $\mathbf{period}(\bar{\mathcal{X}}') = p$ is given, finding $h^\star$ that satisfies (4.52) and (4.53) is equivalent to solving

$$\begin{aligned} z &= \mathbf{min}\ c^{\mathbf{T}} x && (4.55)\\ A'x &\geq \alpha - p\varepsilon && (4.56)\\ x &\geq \beta. && (4.57) \end{aligned}$$

Now, $\mathbf{period}(\bar{\mathcal{X}}') = p$; so, there exists MPLOF h of $\bar{\mathcal{X}}'$. Let $\hat{w}$ be a transition in $\mathcal{C}(\bar{\mathcal{X}}')$ such that $h(\hat{w})$ is the minimum in the set $\{w : w \in \mathcal{C}(\bar{\mathcal{X}}') : h(w)\}$. Let $h^\star = h - h(\hat{w}) + \mathcal{B}$. If $h^\star(v) < 0$, then $h(v) - h(\hat{w}) + \mathcal{B} < 0$ which contradicts Lemma 4.14. Thus, $h^\star : E' \to [0, \infty)$. Also, in Lemma 4.10, for any u and v, adding the same amount to both sides of (4.24) maintains the inequality; consequently, $h^\star$ is a MPLOF of $\bar{\mathcal{X}}'$. Moreover, for any $w \in \mathcal{C}(\bar{\mathcal{X}}')$,

$$h^\star(w) = h(w) - h(\hat{w}) + \mathcal{B} \geq \mathcal{B}$$

by the minimality of $h(\hat{w})$. Hence, a feasible solution exists for the linear program above. Furthermore, since $c^{\mathbf{T}}x$ is non-negative, by the Duality Theorem [14], an optimal solution exists. Therefore, there exists a MPLOF $h^\star$ such that (4.52) and (4.53) are satisfied.

Next, by Lemma 4.10, $h^\star$ is a MPLOF implies

$$h^\star(\hat{v}) \geq \mathbf{max}\{u : u \in \mathbf{src}(q_{\hat{v}}) : h^\star(u) - p\theta(u, \hat{v}, q_{\hat{v}}) + \delta(u, \hat{v}, q_{\hat{v}})\}. \quad (4.58)$$

Consider the following two cases:

Case 1: ($\hat{v} \in \mathcal{C}(\bar{\mathcal{X}}')$) Let $\hat{u}$ be the transition immediately preceding $\hat{v}$ in a critical cycle $\hat{\rho}'$. Then, $\hat{u} \in \mathbf{src}(q_{\hat{v}})$ and, consequently, by Lemma 4.15, (4.58) is an equality.

Case 2: ($\hat{v} \notin \mathcal{C}(\bar{\mathcal{X}}')$) Let μ be the value of the RHS of (4.58). Assume, toward a contradiction that $h^\star(\hat{v}) > \mu$. Let $\nu = \mathbf{max}\{h^\star(\hat{v}) - 1, \mu\}$ and define h' as

$$h'(u) = \begin{cases} h^\star(u) & \text{if } u \neq \hat{v} \\ \nu & \text{if } u = \hat{v}. \end{cases} \quad (4.59)$$

Let w be any transition in $\mathcal{C}(\bar{\mathcal{X}}')$; then, $h^\star(w) - \mathcal{B} \geq 0$ by (4.52). So, for any $\hat{v}$, since $\mathcal{B} - 1 \geq h^\star(w) - h^\star(\hat{v})$ by Lemma 4.14,

$$h'(\hat{v}) = \nu \geq h^\star(\hat{v}) - 1 \geq (h^\star(w) - \mathcal{B} + 1) - 1 \geq 0.$$

Also, by (4.58) and $\nu \geq \mu$, (4.47) is satisfied. Thus, applying Lemma 4.16 implies that h' is MPLOF such that (4.52) is satisfied. But, $|h'| < |h^\star|$ which contradicts the definition of $h^\star$. *Q.E.D.*

Before ending this sub-section, it should be pointed out that though defining $h^\star$ as the MPLOF with the smallest size that satisfies (4.52) leads to the proof of its existence, it is necessary, in Lemma 4.21, to use the fact that $h^\star$ is also the MPLOF with the smallest "norm" as defined below.

Definition: Let $\bar{R}'$ be the set of templates for $\bar{\mathcal{X}}'$. For a MPLOF h of $\bar{\mathcal{X}}'$, the *norm of h in $\bar{\mathcal{X}}'$*, denoted $[\![h, \bar{\mathcal{X}}']\!]$, is defined to be

$$|h| + \sum_{q \in \bar{R}'} \mathbf{max}\{u : u \in \mathbf{src}(q) : h(u) - p\theta(u, \mathbf{tar}(q), q) + \delta(u, \mathbf{tar}(q), q)\}.$$

As a mean to showing that $h^\star$ has the smallest norm, the following definition is introduced.

Definition: For a scenario $\bar{\mathcal{X}}'$, a MPLOF h *saturates* a transition v if either $v \in \mathcal{C}(\bar{\mathcal{X}}')$ or

$$\begin{aligned}&\exists u, q : u \in \mathbf{src}(q) \wedge v = \mathbf{tar}(q) : \\ &\qquad h(v) = h(u) - p\theta(u, v, q) + \delta(u, v, q) \wedge h \text{ saturates } u.\end{aligned} \tag{4.60}$$

Lemma 4.18 *If h is a MPLOF of $\bar{\mathcal{X}}'$ with $h(\mathcal{C}(\bar{\mathcal{X}}')) \geq \mathcal{B}$, then there exists a MPLOF h'' such that $h''(\mathcal{C}(\bar{\mathcal{X}}')) \geq \mathcal{B}$, h'' saturates every transition in $\bar{\mathcal{X}}'$, and $[\![h'', \bar{\mathcal{X}}']\!] \leq [\![h, \bar{\mathcal{X}}']\!]$.*

Proof: Let $\mathcal{S}$ be the set of transitions saturated by h. Let $\mathcal{T}$ be the set of transitions not saturated by h. Let q_v denote the unique template such that $\mathbf{tar}(q_v) = v$. Then, since h is a MPLOF, by Lemma 4.10, for any v and $u \in \mathbf{src}(q_v)$,

$$h(v) \geq h(u) - p\theta(u, v, q_v) + \delta(u, v, q_v). \tag{4.61}$$

Let $\mathcal{Q}$ be defined as $\{u, v : u \in \mathbf{src}(q_v) \wedge u \in \mathcal{S} \wedge v \in \mathcal{T} : \langle u, v \rangle\}$. By the fact that $\bar{\mathcal{X}}'$ is strongly connected, $\mathcal{Q}$ is empty only if $\mathcal{T}$ is empty; in which case, the lemma obviously holds. So, assume $\mathcal{Q}$ is not empty and define

$$\begin{aligned}\hat{s} = \mathbf{min}(&\{1\} \cup \\ &\{u, v : \langle u, v \rangle \in \mathcal{Q} : h(v) - (h(u) - p\theta(u, v, q_v) + \delta(u, v, q_v))\}.\end{aligned} \tag{4.62}$$

Note that $\hat{s} \geq 0$ by (4.61). So, define h' by

$$h'(v) = \begin{cases} h(v) & \text{if } v \in \mathcal{S} \\ h(v) - \hat{s} & \text{if } v \in \mathcal{T}. \end{cases} \tag{4.63}$$

Since $\mathcal{C}(\bar{\mathcal{X}}') \subseteq \mathcal{S}$, $h'(\mathcal{C}(\bar{\mathcal{X}}')) \geq \mathcal{B}$. Also, by Lemma 4.14, $\hat{s} \leq 1$ implies $h(v) - \hat{s} \geq 0$.

Next, let v be an arbitrary transition and u be in $\mathbf{src}(q_v)$. If $u \in \mathcal{T}$, then (4.61) implies

$$h'(v) \geq h'(u) - p\theta(u, v, q_v) + \delta(u, v, q_v) \tag{4.64}$$

since $\hat{s}$ is subtracted from the RHS of (4.61) and at most $\hat{s}$ is subtracted from its LHS. If $u \in \mathcal{S}$ and $v \in \mathcal{T}$, then

$$h'(v) = h(v) - \hat{s} \geq h(v) - (h(v) - (h(u) - p\theta(u, v, q_v) + \delta(u, v, q_v)))$$

by the minimality of $\hat{s}$ and, so, once again (4.64) holds. Finally, if $u \in \mathcal{S}$ and $v \in \mathcal{S}$, then (4.61) implies (4.64) since the values on each side of the two equations are the same. So, h' is a MPLOF of $\bar{\mathcal{X}}'$. Moreover, since h' is the same as h for transitions saturated by h, these transitions are also saturated by h'.

Next, assume, for now, that there exist $\langle \hat{u}, \hat{v} \rangle \in \mathcal{Q}$ such that

$$\hat{s} = h(\hat{v}) - (h(\hat{u}) - p\theta(\hat{u}, \hat{v}, q_{\hat{v}}) + \delta(\hat{u}, \hat{v}, q_{\hat{v}})). \tag{4.65}$$

Then, since $\hat{v} \in \mathcal{T}$, $\hat{s} > 0$. Also, (4.65) and (4.63) imply

$$h'(\hat{v}) = h(\hat{v}) - \hat{s} = h'(\hat{u}) - p\theta(\hat{u}, \hat{v}, q_{\hat{v}}) + \delta(\hat{u}, \hat{v}, q_{\hat{v}}).$$

Hence, $\hat{v}$ is saturated by h'. So, $h'(\mathcal{C}(\bar{\mathcal{X}}')) \geq \mathcal{B}$, h' saturates more transitions than h, and $[\![h', \bar{\mathcal{X}}']\!] \leq [\![h, \bar{\mathcal{X}}']\!]$ by (4.63). Alternatively, suppose that there exists no $\langle \hat{u}, \hat{v} \rangle$ that satisfies (4.65). Then, by (4.62), $\hat{s} = 1$, and, since $\mathcal{T}$ is not empty, (4.63) implies $|h'| \leq |h| - 1$ and $[\![h', \bar{\mathcal{X}}']\!] \leq [\![h, \bar{\mathcal{X}}']\!]$.

So, regardless of whether $\langle \hat{u}, \hat{v} \rangle$ for (4.65) exists, the sum of $|h'|$ and the number of transitions not saturated by h' is at least one less than the corresponding sum for h. Since this sum is bounded below by zero, repeatedly applying this result yields h'' which saturates all transitions and the lemma is therefore established. *Q.E.D.*

Lemma 4.19 *The MPLOF $h^\star$ defined by (4.52) and (4.53) also satisfies*

$$\begin{aligned}[\![h^\star, \bar{\mathcal{X}}']\!] = \mathbf{min}\{h :& \\ & h \text{ is a MPLOF of } \bar{\mathcal{X}}' \wedge h(\mathcal{C}(\bar{\mathcal{X}}')) \geq \mathcal{B} \ : [\![h, \bar{\mathcal{X}}']\!]\}.\end{aligned} \tag{4.66}$$

Proof: By Lemma 4.18, it suffices to consider only MPLOF's that saturate all transitions of $\bar{\mathcal{X}}'$. But for such a MPLOF h,

$$\mathbf{max}\{u : u \in \mathbf{src}(q) : h(u) - p\theta(u, \mathbf{tar}(q), q) + \delta(u, \mathbf{tar}(q), q)\} = h(\mathbf{tar}(q))$$

holds for any template q by (4.43), (4.60), and Lemma 4.10. Since every transition is the target of a unique template, $[\![h, \bar{\mathcal{X}}']\!] = 2|h|$. The validity of this lemma then follows from (4.53). *Q.E.D.*

4.7.2 The "Smallest" MPLOF of an XER-System

In this sub-section, we will show that there is a MPLOF $h^\star$ for any repetitive XER-system $\mathcal{X}'$ such that (4.68) holds. The proof relies on the following result.

Lemma 4.20 *Let $\hat{v}$ be a transition that is disjunctively caused. Let $\hat{q}_0$ and $\hat{q}_1$ be two templates with target $\hat{v}$. Let $\bar{\mathcal{X}}_0'$ and $\bar{\mathcal{X}}_1'$ be two scenarios that are identical except that $\hat{q}_0$ is a template in the former and $\hat{q}_1$ is a template in the latter. Suppose that $\bar{\mathcal{X}}_0'$ is a critical scenario with MPLOF h and*

$$h(\hat{v}) > \mathbf{max}\{u : u \in \mathbf{src}(\hat{q}_1) : h(u) - p\theta(u, \hat{v}, \hat{q}_1) + \delta(u, \hat{v}, \hat{q}_1)\}. \qquad (4.67)$$

Then, h is a MPLOF of $\bar{\mathcal{X}}_1'$ and $\mathcal{C}(\bar{\mathcal{X}}_1') \subseteq \mathcal{C}(\bar{\mathcal{X}}_0')$.

Proof: For $v \neq \hat{v}$, let q_v be the unique template in $\bar{\mathcal{X}}_0'$ with target v. Then, q_v is also the unique template in $\bar{\mathcal{X}}_1'$ with target v. Thus, h is a MPLOF implies

$$h(v) \geq \mathbf{max}\{u : u \in \mathbf{src}(q_v) : h(u) - p\theta(u, v, q_v) + \delta(u, v, q_v)\}$$

for all $v \neq \hat{v}$. This observation and (4.67) imply h is MPLOF of $\bar{\mathcal{X}}_1'$ since $\mathbf{period}(\bar{\mathcal{X}}_1')$ cannot be less than p by Theorem 4.1.

Next, suppose w is a transition in $\mathcal{C}(\bar{\mathcal{X}}_1') \setminus \mathcal{C}(\bar{\mathcal{X}}_0')$. Let ρ' be the critical cycle of $\bar{\mathcal{X}}_1'$ which contains w. Since w is not in $\mathcal{C}(\bar{\mathcal{X}}_0')$, ρ' contains a template not in $\bar{\mathcal{X}}_0'$, namely, $\hat{q}_1$. So, there exists $\hat{u} \in \mathbf{src}(\hat{q}_1)$ such that ρ' can be written as the concatenation of $\langle \hat{u}, \hat{q}_1, \hat{v} \rangle$ and σ' that is a path from $\hat{v}$ back to $\hat{u}$.

Now, by Lemma 4.13,

$$h(\hat{u}) \geq h(\hat{v}) - p\theta(\sigma') + \delta(\sigma').$$

By (4.67),

$$h(\hat{v}) > h(\hat{u}) - p\theta(\hat{u}, \hat{v}, \hat{q}_1) + \delta(\hat{u}, \hat{v}, \hat{q}_1) \geq h(\hat{v}) - p\theta(\rho') + \delta(\rho').$$

But then $p > \dfrac{\delta(\rho')}{\theta(\rho')}$ contradicting the assumption that ρ' is a critical cycle of a critical scenario. This contradiction establishes the lemma. *Q.E.D.*

Lemma 4.21 *There exists a MPLOF $h^\star$ of $\mathcal{X}'$ such that, for any transition $\hat{v}$,*

$$\begin{aligned} h^\star(\hat{v}) = \mathbf{min}\{q : q \in R' \wedge \hat{v} = \mathbf{tar}(q) :& \\ \mathbf{max}\{u : u \in \mathbf{src}(q) : h^\star(u) - p\theta(u, \hat{v}, q) + \delta(u, \hat{v}, q)\}\}.& \end{aligned} \tag{4.68}$$

Proof: Let the critical scenarios of $\mathcal{X}'$ be $\bar{\mathcal{X}}'_0$, $\bar{\mathcal{X}}'_1$, ..., $\bar{\mathcal{X}}'_M$, where $\bar{\mathcal{X}}'_m = \langle E', \bar{R}'_m, \delta, \theta\rangle$. Let $\mathcal{B}$ satisfy (4.41). By Lemma 4.17 and Lemma 4.19, let $h^\star_m$ be the MPLOF satisfying

$$h^\star_m(\mathcal{C}(\bar{\mathcal{X}}'_m)) \geq \mathcal{B}, \quad \text{and} \tag{4.69}$$

$$\begin{aligned} [\![h^\star_m, \bar{\mathcal{X}}'_m]\!] = \mathbf{min}\{h :& \\ h \text{ is a MPLOF of } \bar{\mathcal{X}}'_m \wedge h(\mathcal{C}(\bar{\mathcal{X}}'_m)) \geq \mathcal{B} \ :& \ [\![h, \bar{\mathcal{X}}'_m]\!]\}. \end{aligned} \tag{4.70}$$

Let $\mu = \mathbf{min}\{m :: [\![h^\star_m, \bar{\mathcal{X}}'_m]\!]\}$. W.l.g., assume $[\![h^\star_0, \bar{\mathcal{X}}'_0]\!] = \mu$ and let $h^\star$ be $h^\star_0$.

We will now show that (4.68) is satisfied. Let $\hat{v}$ be an arbitrary transition. Let $\hat{q}_0$ be the unique template with target $\hat{v}$ in $\bar{\mathcal{X}}'_0$. Since $h^\star$ is $h^\star_0$, by (4.54) of Lemma 4.17,

$$h^\star(\hat{v}) = \mathbf{max}\{u : u \in \mathbf{src}(\hat{q}_0) : h^\star(u) - p\theta(u, \hat{v}, \hat{q}_0) + \delta(u, \hat{v}, \hat{q}_0)\}. \tag{4.71}$$

Suppose, toward a contradiction, that (4.68) does not hold. Then, there exists a template $\hat{q}_1$ in $\mathcal{X}'$ with target $\hat{v}$ such that

$$h^\star(\hat{v}) > \mathbf{max}\{u : u \in \mathbf{src}(\hat{q}_1) : h^\star(u) - p\theta(u, \hat{v}, \hat{q}_1) + \delta(u, \hat{v}, \hat{q}_1)\}. \tag{4.72}$$

Let $\bar{\mathcal{X}}'$ be the scenario that is identical to $\bar{\mathcal{X}}'_0$ except that $\hat{q}_0$ is replaced by $\hat{q}_1$. By Lemma 4.20, $h^\star$ is a MPLOF of $\bar{\mathcal{X}}'$. Thus, $\bar{\mathcal{X}}'$ is a critical scenario and, w.l.g., let $\bar{\mathcal{X}}'$ be $\bar{\mathcal{X}}'_1$. Also, by the same lemma, $\mathcal{C}(\bar{\mathcal{X}}'_1) \subseteq \mathcal{C}(\bar{\mathcal{X}}'_0)$. So, since (4.69) holds with $m = 0$, $h^\star(\mathcal{C}(\bar{\mathcal{X}}'_1)) \geq \mathcal{B}$. Therefore, by (4.70) with $m = 1$,

$$[\![h^\star, \bar{\mathcal{X}}'_1]\!] \geq [\![h^\star_1, \bar{\mathcal{X}}'_1]\!]. \tag{4.73}$$

Now, since $\bar{\mathcal{X}}'_0$ and $\bar{\mathcal{X}}'_1$ are identical except for the templates involving $\hat{v}$, (4.71) and (4.72) imply

$$[\![h^\star, \bar{\mathcal{X}}'_0]\!] > [\![h^\star, \bar{\mathcal{X}}'_1]\!]. \tag{4.74}$$

This inequality and (4.73) contradict the minimality of $\mu = [\![h^\star, \bar{\mathcal{X}}'_0]\!]$. Hence, (4.68) holds for $\hat{v}$ and the lemma is established. *Q.E.D.*

4.7.3 Closeness of Approximation

Theorem 4.2 *Let $\hat{t}$ be the timing simulation of a repetitive XER-system $\mathcal{X}' = \langle E', R', \delta, \theta \rangle$ whose scenarios are all strongly connected. Let $\bar{t}$ be any MPLTF of $\mathcal{X}'$. Then, there exists a constant B such that*

$$\forall u, i : u \in E' \wedge i \in \mathbb{N} : \bar{t}(u,i) - \hat{t}(u,i) \leq B \tag{4.75}$$

and B does not depend on u or i.

Proof: Let $\mathcal{X}$ be the XER-system induced by $\mathcal{X}'$. Let the period of $\mathcal{X}'$ be p and let $h^\star$ be a MPLOF of $\mathcal{X}'$ that satisfies (4.68). Define

$$\bar{t}^\star(u,i) = h^\star(u) + pi.$$

Recall the definition of $\theta_{\max}$ and the fact that if $v = \mathbf{tar}(q)$, then for all i, $i \geq \theta_{\max}$,

$$\langle u, j \rangle \in \mathbf{src}(q \lceil i) \Leftrightarrow \langle u, i - \theta(u, v, q) \rangle \in \mathbf{src}(q). \tag{4.76}$$

So, for $i \geq \theta_{\max}$, (4.68) implies

$$\begin{aligned} \bar{t}^\star(\hat{v}, i) = \mathbf{min}\{&q : q \in R' \wedge \hat{v} = \mathbf{tar}(q) : \\ &\mathbf{max}\{u : u \in \mathbf{src}(q) : \bar{t}^\star(u, i - \theta(u, \hat{v}, q)) + \delta(u, \hat{v}, q)\}\}. \end{aligned} \tag{4.77}$$

Let

$$B^\star \stackrel{\text{def}}{=} \mathbf{max}\{u, i : u \in E' \wedge i < \theta_{\max} : \bar{t}^\star(u,i) - \hat{t}(u,i)\}. \tag{4.78}$$

Suppose that

$$\mathcal{Z} \stackrel{\text{def}}{=} \{v, i : v \in E' \wedge i \in \mathbb{N} \wedge \bar{t}^\star(v,i) - \hat{t}(v,i) > B^\star : \langle v, i \rangle\}$$

is not empty. Then, since only XER-systems with acyclic constraint graphs are considered, there exists an element $\langle \hat{v}, \hat{\imath} \rangle$ in $\mathcal{Z}$ such that for any event $\langle u, j \rangle$ in E and any rule $q \lceil \hat{\imath}$ in R,

$$\langle u, j \rangle \in \mathbf{src}(q \lceil i) \wedge \langle \hat{v}, \hat{\imath} \rangle = \mathbf{tar}(q \lceil i) \Rightarrow \langle u, j \rangle \notin \mathcal{Z}. \tag{4.79}$$

Clearly, $\hat{\imath} \geq \theta_{\max}$ by (4.78). If $\langle \hat{v}, \hat{\imath} \rangle$ is in $\mathbf{init}(\mathcal{X})$, then (4.76) implies any templates with $\hat{v}$ as target has an empty set of sources. This relationship implies that there is no edge leading into $\hat{v}$ in the collapsed-constraint graph of $\mathcal{X}'$, contradicting the strong connectivity of the graph.

Alternatively, if $\langle \hat{v}, \hat{\imath} \rangle \notin \mathbf{init}(\mathcal{X})$, then, by (4.79),

$$\begin{aligned} \forall \langle u, j \rangle, \hat{q} : \langle u, j \rangle \in \mathbf{src}(\hat{q}\lceil \hat{\imath}) \wedge \langle \hat{v}, \hat{\imath} \rangle = \mathbf{tar}(\hat{q}\lceil \hat{\imath}) : \\ \bar{t}^\star(u, j) - \hat{t}(u, j) \leq B^\star. \end{aligned} \tag{4.80}$$

So, by the definition of timing simulation and (4.77), $\bar{t}^\star(\hat{v}, \hat{\imath})$ and $\hat{t}(\hat{v}, \hat{\imath})$ are obtained by taking the minimum and maximum over the same sets of templates and source transitions. Thus, Corollary 4.7 implies

$$\bar{t}^\star(\hat{v}, \hat{\imath}) - \hat{t}(\hat{v}, \hat{\imath}) \leq B^\star,$$

which contradicts $\langle \hat{v}, \hat{\imath} \rangle \in \mathcal{Z}$. Therefore, $\mathcal{Z}$ is empty and (4.75) with $\bar{t}^\star$ and $B^\star$ in place of $\bar{t}$ and B holds.

Let $\bar{t}$ be any other MPLTF of $\mathcal{X}'$ and let h be the associated MPLOF. Define

$$B_0 = \mathbf{max}\{u : u \in E' : h(u) - h^\star(u)\}.$$

Then, for all $\langle v, i \rangle \in (E' \times \mathbb{N})$,

$$\bar{t}(v, i) - \hat{t}(v, i) = \bar{t}(v, i) - \bar{t}^\star(v, i) + \bar{t}^\star(v, i) - \hat{t}(v, i) \leq B_0 + B^\star.$$

So, setting B to $B_0 + B^\star$ establishes the theorem. *Q.E.D.*

Note that (4.75) may not hold if a critical scenario $\bar{\mathcal{X}}'$ is not strongly connected. In [15], there is no requirement of strongly connected scenarios and Gunawardena is able to give necessary and sufficient criteria for a condition that would imply (4.75); however, the results are valid only if there are at most two initial events. Since we allow arbitrary number of initial events and have already argued that the scenarios of many practical systems are strongly connected, we believe Theorem 4.2 represents a significant contribution to the theory of timing analysis.

4.8 Summary

We have extended the concept of ER-systems so that inherently disjunctive systems, which arise from PR's with guards that are not mutex, can be modeled. Furthermore, we have shown that the period of a repetitive

XER-system is a good indication of its performance. Thus, to determine the performance of a circuit, it is sufficient to represent the circuit as a repetitive XER-system. Chapter 7 describes how this representation can be achieved systematically. Furthermore, if the delays between transitions in the circuit are specified as functions of transistor sizes, then the performance of circuit can be optimized by finding sizes that minimize the period of the corresponding repetitive XER-system.

To compute this period, one can use the methods described in [6] to find the periods of the scenarios and then select the minimum. Since, in practice, the number of transitions with more than one set of causes and the number of alternative sets of causes for a particular transition are both relatively small, this approach is usually acceptable. Alternatively, when numerical values have been given for the delays, one can start with an arbitrary scenario and use Lemma 4.12 to selectively add and remove templates so that not all scenarios have to be analyzed. Finally, when the delays are functions of transistor sizes, heuristics can be used to search for the optimum period for the entire XER-system instead of doing it for each individual scenario.

Chapter 5

Cumulative State Graphs

We have chosen *cumulative state graphs* as our basic framework for analyzing the behavior of a PR set. Many properties of these graphs will be presented in this chapter. In particular, the notions of *minimal cycles*, *minimal periods*, and *separable graphs* will play a major role in subsequent chapters.

5.1 Definitions

5.1.1 Events and States

Let $\mathcal{P}$ be a closed PR set with K variables. Then,

- $X(\mathcal{P}) = x_0, x_1, \ldots x_{K-1}$ is the *set of variables* of $\mathcal{P}$;
- $\mathcal{E}(\mathcal{P}) = X(\mathcal{P}) \times \mathbb{N}$ is the *set of events* of $\mathcal{P}$;
- $\Sigma(\mathcal{P}) = \mathbb{N}^K$ is the *set of cumulative states*[1] of $\mathcal{P}$.

For an event $\alpha = \langle x_k, l \rangle$, the *variable* of the event is $\mathbf{var}(\alpha) = x_k$, and the *occurrence number* of the event is $\mathbf{oc}(\alpha) = l$.

Intuitively, event $\langle x_k, l \rangle$ represents (approximately) the l-th occurrence of a transition on the variable x_k. For a state σ, its k-th component[2] being

[1]For brevity, in the sequel, a *state* is taken to mean a *cumulative state* unless stated otherwise.

[2]For a vector such as $\sigma \in \Sigma(\mathcal{P})$, we use $\sigma[k]$ to denote the k-th component of the vector.

l implies that $\langle x_k, l\rangle$ has taken place in that state but $\langle x_k, l+1\rangle$ has not. These notions will be formally defined below.

The concept of cumulative states (but not indexed events) has been used in [37] whose authors, using more abstract techniques, have established results similar to some of those given in this chapter. However, developing these results under our approach corresponds more closely to the operational nature of PR sets and allows for extensions that will be presented in the subsequent chapters.

Returning to our model, in order to easily determine the value of a variable at a given state, we have adopted the following convention:

> The occurrence of the event $\langle x_k, l\rangle$ causes x_k in the new state to be **true** if l is odd and causes x_k in the new state to be **false** if l is even.

So, for an event $\gamma = \langle x_k, l\rangle$, *the transition corresponding to* γ is

$$\mathbf{tran}(\gamma) = \begin{cases} x_k\uparrow & \text{if } l \text{ is odd} \\ x_k\downarrow & \text{if } l \text{ is even} \end{cases} \tag{5.1}$$

and *the literal corresponding to* γ is

$$\mathbf{lit}(\gamma) = \begin{cases} x_k & \text{if } l \text{ is odd} \\ \neg x_k & \text{if } l \text{ is even.} \end{cases} \tag{5.2}$$

Also, we define the *Boolean value* of a state σ, denoted by $\mathbf{bool}(\sigma)$, as the element in $\{\mathbf{true}, \mathbf{false}\}^K$ whose components satisfy

$$(\mathbf{bool}(\sigma))[k] = \begin{cases} \mathbf{false} & \text{if } \sigma[k] \text{ is even} \\ \mathbf{true} & \text{if } \sigma[k] \text{ is odd.} \end{cases} \tag{5.3}$$

The value of x_k in state σ is then $(\mathbf{bool}(\sigma))[k]$, and, by extrapolation, we can define the value of any Boolean expression involving the variables x_0, x_1, $\ldots$, x_{K-1} in state σ.

Example 5.1: As an illustration of the concepts of this chapter, let $\mathcal{P}$ be the PR set shown in Figure 5.1. Then, at state[3] $\sigma = 2211$, $\mathbf{bool}(\sigma) = \langle\mathbf{false}, \mathbf{false}, \mathbf{true}, \mathbf{true}\rangle$. Thus, the values of x_0 and x_1 are both **false** and the value of $\neg x_0 \wedge \neg x_1$ is **true** at σ. $\square$

[3]The numerical value of a state σ is written as the juxtaposition $\sigma[0]\sigma[1]\ldots\sigma[K-1]$.

$$
\begin{array}{lcl}
\neg x_3 & \rightarrow & x_0\!\uparrow \\
\neg x_3 & \rightarrow & x_1\!\uparrow \\
x_0 \vee x_1 & \rightarrow & x_2\!\uparrow \\
x_0 \wedge x_1 \wedge x_2 & \rightarrow & x_3\!\uparrow \\
x_3 & \rightarrow & x_0\!\downarrow \\
x_3 & \rightarrow & x_1\!\downarrow \\
\neg x_0 \wedge \neg x_1 & \rightarrow & x_2\!\downarrow \\
\neg x_2 & \rightarrow & x_3\!\downarrow
\end{array}
$$

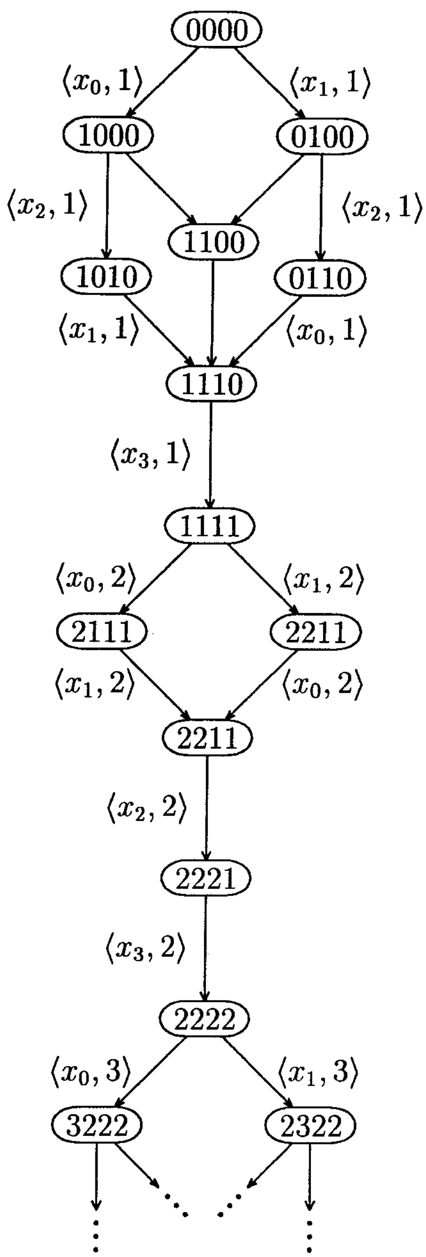

Figure 5.1: A state graph with initial state $\sigma_{\text{init}} = 0000$

5.1.2 State Changes

Definition: An event $\alpha = \langle x_k, l\rangle$ is said to be *enabled* in state σ, denoted $\mathbf{enb}(\alpha, \sigma)$, if $\sigma[k] = l - 1$ and the guard for $\mathbf{tran}(\alpha)$ is **true** in σ.

Example 5.2: In the above example, since $\sigma[2] = 1$ and the value of $\neg x_0 \wedge \neg x_1$, the guard for $x_2{\downarrow}$, is **true** at σ, $\mathbf{enb}(\langle x_2, 2\rangle, \sigma)$. □

Definition: A state σ_a *changes to* σ_b, denoted $\sigma_a \longrightarrow \sigma_b$, if there exists an event $\alpha = \langle x_k, l\rangle$ such that

$$\mathbf{enb}(\alpha, \sigma_a) \wedge \sigma_b[k] = \sigma_a[k] + 1 \wedge \forall\ \tilde{k} : \tilde{k} \neq k : \sigma_b[\tilde{k}] = \sigma_a[\tilde{k}].$$

The event α is said to *effect* a state change between σ_a and σ_b and this relationship is represented by $\sigma_a \xrightarrow{\alpha} \sigma_b$.

Note that under this definition, a transition on a variable is an atomic action. To model the situation where there is an arbitrary delay along a non-isochronic branch, a wire operator and a new variable need to be added explicitly as done in Sub-section 2.3.3.

Example 5.3: Since $\mathbf{enb}(\langle x_2, 2\rangle, 2211)$, $2211 \xrightarrow{\langle x_2, 2\rangle} 2221$. □

Definition: The relationship *leads to*, denoted $\overset{*}{\longrightarrow}$, is the reflexive, transitive closure of *changes to*. In other words, it is the smallest relation defined recursively by

1. $\sigma_a \overset{*}{\longrightarrow} \sigma_a$.
2. $\sigma_a \overset{*}{\longrightarrow} \sigma_b \wedge \sigma_b \longrightarrow \sigma_c \Rightarrow \sigma_a \overset{*}{\longrightarrow} \sigma_c$.

In the definition below, σ_{init} can be any member of $\Sigma(\mathcal{P})$. In particular, one can assume that

$$\sigma_{\text{init}}[k] = \begin{cases} 0 & \text{if the initial value of } x_k \text{ is } \mathbf{false} \\ 1 & \text{if the initial value of } x_k \text{ is } \mathbf{true}. \end{cases}$$

Definition: The *(cumulative) state graph* of $\mathcal{P}$ for a given initial state σ_{init} is a labeled directed graph $\Gamma(\mathcal{P}, \sigma_{\text{init}}) = \langle \mathbf{S}, \mathbf{C}\rangle$ where

- $\mathbf{S} = \{\sigma : \sigma \in \Sigma(\mathcal{P}) \wedge \sigma_{\text{init}} \overset{*}{\longrightarrow} \sigma : \sigma\}$;
- $\mathbf{C} = \{\sigma_a, \sigma_b, \alpha : \sigma_a \in \mathbf{S} \wedge \sigma_b \in \mathbf{S} \wedge \alpha \in \mathcal{E}(\mathcal{P}) \wedge \sigma_a \xrightarrow{\alpha} \sigma_b : \langle \sigma_a, \sigma_b, \alpha \rangle\}$.

S is the set of states of Γ and, in the sequel, *unless specifically stated otherwise, all states are those that are reachable from the initial state* and will be represented by σ, τ, ϕ, or ρ. Events will be represented by α, β, γ, or δ. Furthermore,

$$\sigma_0 \xrightarrow{\alpha_0} \sigma_1 \xrightarrow{\alpha_1} \cdots \xrightarrow{\alpha_{n-1}} \sigma_n$$

denotes the fact that

$$\forall i : 0 \leq i < n : \langle \sigma_i, \sigma_{i+1}, \alpha_i \rangle \in \mathbf{C}$$

and such a set of connected edges will be referred to as a *path* in the graph. The *length* of the path is n and α_i, for i such that $0 \leq i < n$, is said to *occur in* the path. Also, we will use $\sigma_a \overset{*}{\longrightarrow} \sigma_b$ to denote any path from σ_a to σ_b (including the one with zero length), if the identities of the events and intermediate states in the path are immaterial.

Example 5.4: Figure 5.1 shows a PR set and the beginning part of its state graph starting at initial state $\sigma_{\text{init}} = 0000$. To avoid cluttering up the picture, the labels of some of the edges are not given; however, they should be obvious from context. □

5.2 Basic Properties

5.2.1 Weights and Paths

In this section, we list some basic properties of state graphs for future reference.

Definition: The *weight* for a state σ is $\mathbf{wt}(\sigma) = \sum_{k=0}^{K-1} \sigma[k]$.

Lemma 5.1 *If* $\sigma_0 \xrightarrow{\alpha_0} \sigma_1 \xrightarrow{\alpha_1} \cdots \xrightarrow{\alpha_{n-1}} \sigma_n$, *then* $\mathbf{wt}(\sigma_n) = \mathbf{wt}(\sigma_0) + n$.

Proof: Use the definition of state change and induction on n. *Q.E.D.*

Lemma 5.2 *If* $\sigma_0 \xrightarrow{\alpha_0} \sigma_1 \xrightarrow{\alpha_1} \cdots \xrightarrow{\alpha_{n-1}} \sigma_n$ *and* $\mathcal{A} = \{i : 0 \le i < n : \alpha_i\}$, *then for any* k, $0 \le k < K$, $\sigma_0[k] \le \sigma_n[k]$ *and*

$$\forall l :: (\sigma_0[k] < l \le \sigma_n[k] \Leftrightarrow \langle x_k, l\rangle \in \mathcal{A}). \tag{5.4}$$

Proof: Use the definition of state change and induction on n. *Q.E.D.*

Lemma 5.3 *Suppose there exist states and events such that*

$$\sigma \xrightarrow{\alpha_0} \tau_1 \xrightarrow{\alpha_1} \cdots \xrightarrow{\alpha_{n-2}} \tau_{n-1} \xrightarrow{\alpha_{n-1}} \sigma_a, \tag{5.5}$$

and

$$\sigma \xrightarrow{\beta_0} \phi_1 \xrightarrow{\beta_1} \cdots \xrightarrow{\beta_{m-2}} \phi_{m-1} \xrightarrow{\beta_{m-1}} \sigma_b. \tag{5.6}$$

Let $\mathcal{A} = \{i : 0 \le i < n : \alpha_i\}$ *and* $\mathcal{B} = \{i : 0 \le i < m : \beta_i\}$. *Then,* $\mathcal{A} = \mathcal{B}$ *if and only if* $\sigma_a = \sigma_b$.

Proof: By Lemma 5.2, $\mathcal{A}$ is determined uniquely by σ_a, and vice versa. So, $\mathcal{A} = \mathcal{B} \Leftrightarrow \sigma_a = \sigma_b$. *Q.E.D.*

As a notational shorthand, we will use $\sigma_a \overset{\mathcal{A}}{\twoheadrightarrow} \sigma_b$ to denote the facts that there is a path from σ_a to σ_b and $\mathcal{A}$ is a set of events occurring along that path. By Lemma 5.3, $\mathcal{A}$ is unique. In the sequel, $\mathcal{A}$, $\mathcal{B}$, $\mathcal{C}$, and $\mathcal{D}$ will be used to represent sets of events.

5.2.2 Stable Graphs

In a stable PR set $\mathcal{P}$, if two transitions are enabled, then firing one does not cause the other to become not enabled. This property is reflected in the state graph of $\mathcal{P}$ by the following definition which is illustrated in Figure 5.2

Definition: A state graph Γ is *stable* if for any $\alpha \ne \beta$,

$$(\sigma_a \xrightarrow{\alpha} \sigma_b \wedge \sigma_a \xrightarrow{\beta} \sigma_c) \Rightarrow (\exists\ \sigma_d :: \sigma_b \xrightarrow{\beta} \sigma_d \wedge \sigma_c \xrightarrow{\alpha} \sigma_d).$$

Since only stable PR sets are produced by the compilation method, from now on, *all state graphs are assumed to be stable unless stated otherwise.* As an aside, it should be pointed out that stability in a state graph is analogous to *semi-modularity* in a lattice [3]. However, since some of the later results

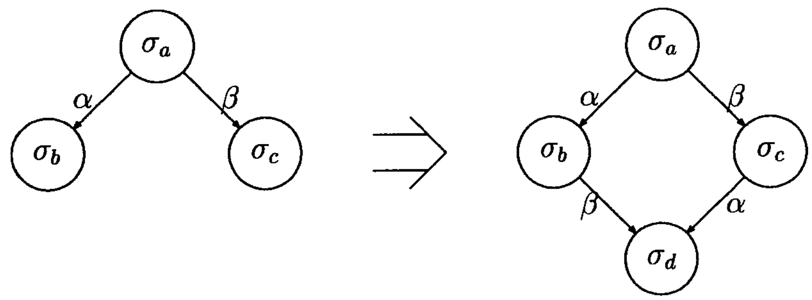

Figure 5.2: Stability in a state graph

apply only to state graphs and not to lattices in general, we have decided not to employ any lattice theory and, instead, start from first principles as given above.

The next two results apply the stability property to paths in which no common event occurs. Lemma 5.6 and Lemma 5.7 then investigate the more general situation.

Lemma 5.4 *If*

$$\sigma_0 \xrightarrow{\alpha_0} \sigma_1 \xrightarrow{\alpha_1} \cdots \xrightarrow{\alpha_{n-1}} \sigma_n, \tag{5.7}$$

$\sigma_0 \xrightarrow{\beta} \tau_0$, *and* $\forall\ i : 0 \leq i < n : \alpha_i \neq \beta$, *then there exists* $\{i : 1 \leq i \leq n : \tau_i\}$ *such that*

$$\tau_0 \xrightarrow{\alpha_0} \tau_1 \xrightarrow{\alpha_1} \cdots \xrightarrow{\alpha_{n-1}} \tau_n$$

and $\forall\ i : 1 \leq i \leq n : \mathbf{enb}(\beta, \sigma_i) \wedge \sigma_i \xrightarrow{\beta} \tau_i$.

Proof: Use stability and induction on n. *Q.E.D.*

Lemma 5.5 *If*

$$\sigma_{0,0} \xrightarrow{\alpha_0} \sigma_{1,0} \xrightarrow{\alpha_1} \cdots \xrightarrow{\alpha_{n-2}} \sigma_{n-1,0} \xrightarrow{\alpha_{n-1}} \sigma_{n,0}$$

and

$$\sigma_{0,0} \xrightarrow{\beta_0} \sigma_{0,1} \xrightarrow{\beta_1} \cdots \xrightarrow{\beta_{m-2}} \sigma_{0,m-1} \xrightarrow{\beta_{m-1}} \sigma_{0,m}$$

and $\{i : 0 \leq i < n : \alpha_i\} \cap \{i : 0 \leq i < m : \beta_i\} = \emptyset$, *then there exists* $\{i, j : 1 \leq i \leq n \wedge 1 \leq j \leq m : \sigma_{i,j}\}$ *such that*

$$\forall\ i, j : 0 \leq i < n \wedge 0 \leq j < m : \sigma_{i,j} \xrightarrow{\alpha_i} \sigma_{i+1,j} \wedge \sigma_{i,j} \xrightarrow{\beta_j} \sigma_{i,j+1}. \tag{5.8}$$

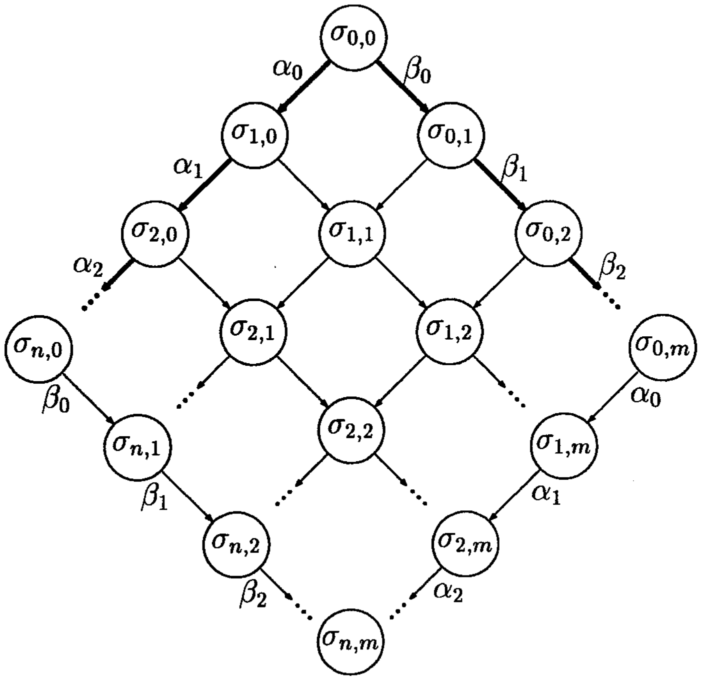

Figure 5.3: Stability for paths with no common event

Proof: Use Lemma 5.4 and induction on m. See Figure 5.3 where the state changes stated in the hypothesis are shown in bold. *Q.E.D.*

Lemma 5.6 *If (5.7) holds, and there exists $\hat{\imath}$ such that $0 \leq \hat{\imath} < n$ and* $\mathbf{enb}(\alpha_{\hat{\imath}}, \sigma_0)$*, then there exists $\{i : 0 \leq i < \hat{\imath} : \tau_i\}$ such that*

$$\tau_0 \xrightarrow{\alpha_0} \tau_1 \xrightarrow{\alpha_1} \cdots \xrightarrow{\alpha_{\hat{\imath}-2}} \tau_{\hat{\imath}-1} \xrightarrow{\alpha_{\hat{\imath}-1}} \sigma_{\hat{\imath}+1} \xrightarrow{\alpha_{\hat{\imath}+1}} \sigma_{\hat{\imath}+2} \cdots \xrightarrow{\alpha_{n-1}} \sigma_n. \tag{5.9}$$

Proof: By definition of a state graph, $\mathbf{enb}(\alpha_{\hat{\imath}}, \sigma_0)$ implies there exists τ_0 such that $\sigma_0 \xrightarrow{\alpha_{\hat{\imath}}} \tau_0$. Apply Lemma 5.4 to this relationship and

$$\sigma_0 \xrightarrow{\alpha_0} \sigma_1 \xrightarrow{\alpha_1} \cdots \xrightarrow{\alpha_{\hat{\imath}-2}} \sigma_{\hat{\imath}-1} \xrightarrow{\alpha_{\hat{\imath}-1}} \sigma_{\hat{\imath}}$$

to get the existence of $\{i : 0 \leq i \leq \hat{\imath} : \tau_i\}$ such that

$$\sigma_0 \xrightarrow{\alpha_{\hat{\imath}}} \tau_0 \xrightarrow{\alpha_0} \tau_1 \xrightarrow{\alpha_1} \cdots \xrightarrow{\alpha_{\hat{\imath}-2}} \tau_{\hat{\imath}-1} \xrightarrow{\alpha_{\hat{\imath}-1}} \tau_{\hat{\imath}}. \tag{5.10}$$

But since the set of events occurring in $\sigma_0 \overset{*}{\longrightarrow} \tau_i$ is the same as the set of events occurring in $\sigma_0 \overset{*}{\longrightarrow} \sigma_{i+1}$, $\tau_i = \sigma_{i+1}$ by Lemma 5.3 and (5.9) holds. *Q.E.D.*

Lemma 5.7 *If*

$$\sigma \overset{\mathcal{A}}{\overset{*}{\longrightarrow}} \sigma_a \wedge \sigma \overset{\mathcal{B}}{\overset{*}{\longrightarrow}} \sigma_b, \tag{5.11}$$

then there exists state τ *such that*

$$\sigma_a \overset{\mathcal{B} \setminus \mathcal{A}}{\overset{*}{\longrightarrow}} \tau \wedge \sigma_b \overset{\mathcal{A} \setminus \mathcal{B}}{\overset{*}{\longrightarrow}} \tau. \tag{5.12}$$

Proof: Use induction on the size of $\mathcal{B}$.

Base Case: $(|\mathcal{B}| = 0)$ In this case, $\sigma_b = \sigma$. So, let $\tau = \sigma_a$.

Inductive Step: Suppose $|\mathcal{B}| > 0$. This situation implies there exist $\widehat{\sigma}_b$, $\widehat{\mathcal{B}}$, and β such that

$$\sigma \overset{\widehat{\mathcal{B}}}{\overset{*}{\longrightarrow}} \widehat{\sigma}_b \overset{\beta}{\longrightarrow} \sigma_b \tag{5.13}$$

and $\mathcal{B} = \widehat{\mathcal{B}} \uplus \{\beta\}$. By the inductive hypothesis, there exists state $\widehat{\tau}$ such that

$$\sigma_a \overset{\widehat{\mathcal{B}} \setminus \mathcal{A}}{\overset{*}{\longrightarrow}} \widehat{\tau} \wedge \widehat{\sigma}_b \overset{\mathcal{A} \setminus \widehat{\mathcal{B}}}{\overset{*}{\longrightarrow}} \widehat{\tau}. \tag{5.14}$$

Consider the following two cases:

Case 1: $(\beta \notin (\mathcal{A} \setminus \widehat{\mathcal{B}}))$ This case implies $\beta \notin \mathcal{A}$ or $\beta \in \widehat{\mathcal{B}}$. But $\beta \notin \widehat{\mathcal{B}}$ by (5.13); so, $\beta \notin \mathcal{A}$. Furthermore, by Lemma 5.4 and the second half of (5.14), there exists τ such that

$$\widehat{\tau} \overset{\beta}{\longrightarrow} \tau \wedge \sigma_b \overset{\mathcal{A} \setminus \widehat{\mathcal{B}}}{\overset{*}{\longrightarrow}} \tau.$$

So, because $\beta \notin \mathcal{A}$, the set of events occurring in $\sigma_a \overset{*}{\longrightarrow} \widehat{\tau} \overset{*}{\longrightarrow} \tau$ is $((\widehat{\mathcal{B}} \setminus \mathcal{A}) \cup \{\beta\}) = (\mathcal{B} \setminus \mathcal{A})$ and the set of events occurring in $\sigma_b \overset{*}{\longrightarrow} \tau$ is $(\mathcal{A} \setminus \widehat{\mathcal{B}}) = (\mathcal{A} \setminus \mathcal{B})$.

Case 2: $(\beta \in (\mathcal{A} \setminus \widehat{\mathcal{B}}))$ This case implies $\beta \in \mathcal{A}$. Also, by Lemma 5.6,

$$\sigma_b \overset{(\mathcal{A} \setminus \widehat{\mathcal{B}}) \setminus \{\beta\}}{\overset{*}{\longrightarrow}} \widehat{\tau}.$$

Let $\tau = \widehat{\tau}$. Then, due to $\beta \in \mathcal{A}$, the set of events occurring in $\sigma_a \overset{*}{\longrightarrow} \tau$ is $(\widehat{\mathcal{B}} \setminus \mathcal{A}) = (\mathcal{B} \setminus \mathcal{A})$ and the set of events occurring in $\sigma_b \overset{*}{\longrightarrow} \tau$ is $((\mathcal{A} \setminus \widehat{\mathcal{B}}) \setminus \{\beta\}) = (\mathcal{A} \setminus \mathcal{B})$. *Q.E.D.*

5.2.3 Descendents and Ancestors

Definition: A state ϕ is a *descendent* of σ if $\sigma \overset{*}{\rightarrow} \phi$. A state ϕ is an *ancestor* of σ if $\phi \overset{*}{\rightarrow} \sigma$. A state ϕ is a *common descendent* (or, alternatively, *common ancestor*) of σ_a and σ_b if ϕ is a descendent (ancestor) of both σ_a and σ_b. A state ϕ is a *closest common descendent (c.c.d.)* of σ_a and σ_b if it is a common ancestor of σ_a and σ_b and

$$\forall \tilde{\phi} : \tilde{\phi} \text{ is a common descendent of } \sigma_a \text{ and } \sigma_b : \mathbf{wt}(\tilde{\phi}) \geq \mathbf{wt}(\phi). \tag{5.15}$$

A state ϕ is a *closest common ancestor (c.c.a.)* of σ_a and σ_b if it is a common ancestor of σ_a and σ_b and

$$\forall \tilde{\phi} : \tilde{\phi} \text{ is a common ancestor of } \sigma_a \text{ and } \sigma_b : \mathbf{wt}(\tilde{\phi}) \leq \mathbf{wt}(\phi). \tag{5.16}$$

Example 5.5: In Example 5.4, let $\sigma_a = 1010$ and $\sigma_b = 0110$. Then, 1110, 1111, 2111, etc. are their common descendents. Since 1110 has the least weight, it is their c.c.d. This observation can be generalized by the following lemma. □

Lemma 5.8 *Any two states σ_a and σ_b have a unique c.c.d. τ defined by*

$$\forall k : 0 \leq k < K : \tau[k] = \mathbf{max}\{\sigma_a[k], \sigma_b[k]\}. \tag{5.17}$$

Proof: Let k be an arbitrary variable index. By Lemma 5.2, if τ is a common descendent of σ_a and σ_b, then

$$\tau[k] \geq \mathbf{max}\{\sigma_a[k], \sigma_b[k]\}. \tag{5.18}$$

Thus, if any common descendent τ satisfies (5.17), then τ is the unique c.c.d.

Next, in Lemma 5.7, let $\sigma = \sigma_{\text{init}}$ and consider the state τ guaranteed by that lemma. First, τ is a common descendent of σ_a and σ_b. Also, by Lemma 5.2, $\langle x_k, \tau[k] \rangle$ is in $\mathcal{A} \cup (\mathcal{B} \setminus \mathcal{A})$ and $\mathcal{B} \cup (\mathcal{A} \setminus \mathcal{B})$. So, $\langle x_k, \tau[k] \rangle \in \mathcal{A} \cup \mathcal{B}$ which implies, by Lemma 5.2,

$$\tau[k] \leq \sigma_a[k] \vee \tau[k] \leq \sigma_b[k]. \tag{5.19}$$

By (5.18), at least one of these two inequalities is an equality and (5.17) holds. So, τ is the unique c.c.d. of σ_a and σ_b. *Q.E.D.*

Corollary 5.9 *For any two state σ and τ,*

$$\sigma \overset{\star}{\twoheadrightarrow} \tau \Leftrightarrow \forall k :: \sigma[k] \leq \tau[k]. \tag{5.20}$$

Proof: The "if" part is from Lemma 5.2. The "only-if" part is due to τ being the c.c.d. of σ and τ. *Q.E.D.*

Since σ_{init} is the common ancestor of any two states and there is an upper bound on the weight of any ancestor of a state, a c.c.a. exists for any two states. However, an analogy to (5.17) for defining the c.c.a. does not exist as the following example illustrates. Instead, we need to establish Lemma 5.10 in order to show that the c.c.a. of any two states is unique.

Example 5.6: Continuing with the previous example where $\sigma_a = 1010$ and $\sigma_b = 0110$. Then, $\rho = 0000$ is the only common ancestor of the two states and is therefore their c.c.a. However, $\rho[2] \neq \mathbf{min}\{\sigma_a[2], \sigma_b[2]\}$. □

Lemma 5.10 *Let ρ be a c.c.a. of σ_a and σ_b with $\rho \overset{\mathcal{A}}{\twoheadrightarrow} \sigma_a$ and $\rho \overset{\mathcal{B}}{\twoheadrightarrow} \sigma_b$. If $\mathbf{enb}(\gamma, \rho)$, then $\gamma \notin (\mathcal{A} \cap \mathcal{B})$.*

Proof: Assume, toward a contradiction, that $\mathbf{enb}(\gamma, \rho)$ and $\gamma \in (\mathcal{A} \cap \mathcal{B})$. By Lemma 5.6, there exist τ_a and τ_b such that

$$\rho \xrightarrow{\gamma} \tau_a \overset{\mathcal{A}\setminus\{\gamma\}}{\twoheadrightarrow} \sigma_a \wedge \rho \xrightarrow{\gamma} \tau_b \overset{\mathcal{B}\setminus\{\gamma\}}{\twoheadrightarrow} \sigma_b.$$

But then, by Lemma 5.3, $\tau_a = \tau_b$, and therefore τ_a is a common ancestor of σ_a and σ_b. Moreover, by Lemma 5.1, $\mathbf{wt}(\tau_a) = \mathbf{wt}(\rho) + 1$. This equation violates the hypothesis that ρ is a c.c.a. of σ_a and σ_b. *Q.E.D.*

Lemma 5.11 *Let ρ be a c.c.a. of σ_a and σ_b. If $\tilde{\rho}$ be a common ancestor of σ_a and σ_b, then $\tilde{\rho} \overset{\star}{\twoheadrightarrow} \rho$.*

Proof: Let ϕ be a c.c.a. of $\tilde{\rho}$ and ρ with $\phi \overset{\mathcal{C}}{\twoheadrightarrow} \tilde{\rho}$ and $\phi \overset{\mathcal{D}}{\twoheadrightarrow} \rho$. Suppose, toward a contradiction, that $\phi \neq \tilde{\rho}$. Then, there exists γ and $\hat{\phi}$ such that $\gamma \in \mathcal{C}$ and $\mathbf{enb}(\gamma, \phi)$. By Lemma 5.10, $\gamma \notin \mathcal{D}$ and therefore, by stability, $\mathbf{enb}(\gamma, \rho)$.

Now, γ occurs in $\phi \overset{\star}{\twoheadrightarrow} \tilde{\rho} \overset{\star}{\twoheadrightarrow} \sigma_a$; so, it occurs in $\phi \overset{\star}{\twoheadrightarrow} \rho \overset{\star}{\twoheadrightarrow} \sigma_a$. Since $\gamma \notin \mathcal{D}$, γ occurs in $\rho \overset{\star}{\twoheadrightarrow} \sigma_a$. Similarly, γ occurs in $\rho \overset{\star}{\twoheadrightarrow} \sigma_b$. By Lemma 5.10, these two relationships and $\mathbf{enb}(\gamma, \rho)$ contradict the hypothesis that ρ is a c.c.a. of σ_a and σ_b. Thus, $\phi = \tilde{\rho}$ and $\tilde{\rho} \overset{\star}{\twoheadrightarrow} \rho$. *Q.E.D.*

Corollary 5.12 *Two states have a unique c.c.a..*

Proof: Let ρ and $\widehat{\rho}$ both be c.c.a.s of the two state. By the previous lemma, $\rho \xrightarrow{*} \widehat{\rho}$ and $\widehat{\rho} \xrightarrow{*} \rho$. Thus, $\rho = \widehat{\rho}$. *Q.E.D.*

We conclude this section with two other consequences of Lemma 5.10 that will be useful later.

Lemma 5.13 *Let ρ be the c.c.a. of σ_a and σ_b and $\widehat{\rho}$ be any of their common ancestors. If α occurs in $\widehat{\rho} \xrightarrow{*} \sigma_a$ but not in $\widehat{\rho} \xrightarrow{*} \sigma_b$, then α occurs in $\rho \xrightarrow{*} \sigma_a$.*

Proof: Consider the two paths $\widehat{\rho} \xrightarrow{*} \rho \xrightarrow{*} \sigma_a$ and $\widehat{\rho} \xrightarrow{*} \rho \xrightarrow{*} \sigma_b$. Since α occurs in the first but not the second, it occurs in $\rho \xrightarrow{*} \sigma_a$. *Q.E.D.*

Lemma 5.14 *If $\sigma \xrightarrow[*]{\mathcal{A}} \tau$, $\phi \xrightarrow[*]{\mathcal{B}} \tau$, and $\mathcal{B} \subseteq \mathcal{A}$, then $\sigma \xrightarrow[*]{\mathcal{A} \setminus \mathcal{B}} \phi$.*

Proof: Let ρ be the c.c.a. of σ and ϕ. If $\rho \neq \sigma$, then let

$$\rho \xrightarrow{\alpha} \widehat{\rho} \xrightarrow{*} \sigma \xrightarrow[*]{\mathcal{A}} \tau. \tag{5.21}$$

By Lemma 5.10, α does not occur in $\rho \xrightarrow{*} \phi$. Also, by (5.21), $\alpha \notin \mathcal{A}$ which implies $\alpha \notin \mathcal{B}$. Therefore, α does not occur in $\rho \xrightarrow{*} \phi \xrightarrow[*]{\mathcal{B}} \tau$, contradicting (5.21) and Lemma 5.3. Therefore, $\rho = \sigma$ and $\sigma \xrightarrow{*} \phi$. The set of events on the path then follows from Lemma 5.3. *Q.E.D.*

5.3 Cycles and Periods

5.3.1 State Offsets

Besides being a state of $\Sigma(\mathcal{P})$, $\pi \in \mathbb{N}^K$ can also be considered as a "state offset" since, for any state σ and any state offset π, we can define $\tau = \sigma + \pi$ as a state in $\Sigma(\mathcal{P})$ by

$$\forall i : 0 \leq i < K : \tau[i] = \sigma[i] + \pi[i]. \tag{5.22}$$

If $\sigma_{\text{init}} \overset{}{\twoheadrightarrow} \tau$, then τ is also a state of $\Gamma(\mathcal{P}, \sigma_{\text{init}})$. The expression $\sigma - \pi$ is similarly defined provided $\forall i : 0 \le i < K : \sigma[i] \ge \pi[k]$. Likewise, for $q \ge 0$, $q\pi$ denotes the state offset such that $(q\pi)[k] = q\,(\pi[k])$. Also, for two state offsets, π and ω, $\pi \le \omega$ denotes $\forall k : 0 \le k < K : \pi[k] \le \omega[k]$. and $\pi < \omega$ is equivalent to $(\pi \le \omega) \wedge (\pi \ne \omega)$.

Next, we use the following definition to capture the notion of "adding" and "subtracting" a state offset to an event.

Definition: For an event $\alpha = \langle x_k, l\rangle$ and a state offset π, the *(positive) extension* of α by π is the event $(\alpha \oplus \pi) = \langle x_k, l + \pi[k]\rangle$. Similarly, provided $\sigma_{\text{init}}[k] < l - \pi[k]$, the *negative extension* of α by π is the event $(\alpha \ominus \pi) = \langle x_k, l - \pi[k]\rangle$. Also, we will use $(\mathcal{A} \oplus \pi)$ to denote the set $\{\alpha : \alpha \in \mathcal{A} : (\alpha \oplus \pi)\}$ and analogously for $(\mathcal{A} \ominus \pi)$.

By Lemma 5.2, for any path

$$\sigma \overset{\mathcal{A}}{\twoheadrightarrow} (\sigma + \pi), \tag{5.23}$$

the number of distinct α_i's with $\mathbf{var}(\alpha_i) = x_k$ is precisely $(\sigma_0 + \pi)[k] - \sigma_0[k] = \pi[k]$. So, the set of variables that occur in the path is a function of π and not of σ. Thus, for reference, we borrow the following definition from [37].

Definition: For a state offset π, the *spanning set* of π is $\mathbf{span}(\pi) = \{k : \pi[k] \ne 0 : x_k\}$.

Note that in (5.23), $\mathbf{span}(\pi)$ is determined by the events in $\mathcal{A}$. So, for convenience, we overload the meaning of $\mathbf{span}()$ and define the *spanning set* of a set of events $\mathcal{A}$ as

$$\mathbf{span}(\mathcal{A}) = \{k, l : \langle x_k, l\rangle \in \mathcal{A} : x_k\}.$$

With this definition, the path in (5.23) implies $\mathbf{span}(\pi) = \mathbf{span}(\mathcal{A})$.

5.3.2 Cycles

Definition: The path $\sigma_0 \overset{\mathcal{A}}{\twoheadrightarrow} \sigma_n$ is called a *cycle* if $\mathcal{A} \ne \emptyset$ and $\mathbf{bool}(\sigma_0) = \mathbf{bool}(\sigma_n)$. The *period* of the cycle is the state offset π such that $\sigma_n = \sigma_0 + \pi$. A state offset π is a *period* if it is the period of any cycle.

Since the length of any cycle is at least one, its period is not the zero-vector. Also, by the definition of the Boolean value of a state, if $\tau = \sigma + \pi$, then $\mathbf{bool}(\sigma) = \mathbf{bool}(\tau)$ if and only if π has only even components.

Example 5.7: Referring back to Figure 5.1, $0100 \xrightarrow{*} 2322$ is a cycle in the graph with period 2222. □

The following lemma verifies the intuitive notion of what it means for two states to have the same Boolean value. Some of its immediate consequences are listed afterward for future reference.

Lemma 5.15 *If there exist state offset π and states σ and τ such that*

$$\mathbf{bool}(\sigma) = \mathbf{bool}(\tau) \wedge \tau = \sigma + \pi, \tag{5.24}$$

then

$$\begin{aligned}(\forall \alpha :: \mathbf{enb}(\alpha, \sigma) \Rightarrow \mathbf{enb}(\alpha \oplus \pi, \tau)) \wedge \\ (\forall \beta :: \mathbf{enb}(\beta, \tau) \Rightarrow \mathbf{enb}(\beta \ominus \pi, \sigma)).\end{aligned} \tag{5.25}$$

Proof: Suppose $\mathbf{enb}(\beta, \tau)$. Let $\beta = \langle x_k, l \rangle$. Then, by definition, $\tau[k] = l - 1$ and the guard for $\mathbf{tran}(x_k)$ is **true** in state τ. But $\mathbf{bool}(\tau) = \mathbf{bool}(\sigma)$; so, the guard for $\mathbf{tran}(x_k)$ is **true** in state σ as well. Moreover,

$$\sigma[k] = (\tau - \pi)[k] = \tau[k] - \pi[k] = (l - 1) - \pi[k].$$

Thus, $(\beta \ominus \pi) = \langle x_k, l - \pi[k] \rangle$ is enabled in state σ and the second half of (5.25) is established. Alternatively, if $\mathbf{enb}(\alpha, \sigma)$, then by analogous arguments to the ones above, the first half of (5.25) holds. *Q.E.D.*

Lemma 5.16 *If there exist state offset π and states σ and τ such that (5.24) holds, then, for $n \geq 0$,*

$$(\sigma = \sigma_0) \xrightarrow{\alpha_0} \sigma_1 \xrightarrow{\alpha_1} \cdots \xrightarrow{\alpha_{n-1}} \sigma_n \tag{5.26}$$

if and only if

$$(\tau = (\sigma_0 + \pi)) \xrightarrow{\alpha_0 \oplus \pi} (\sigma_1 + \pi) \xrightarrow{\alpha_1 \oplus \pi} \cdots \xrightarrow{\alpha_{n-1} \oplus \pi} (\sigma_n + \pi). \tag{5.27}$$

Proof: Use Lemma 5.15 and induction on n. *Q.E.D.*

Corollary 5.17 *Given a cycle* $\sigma \overset{\mathcal{A}}{\twoheadrightarrow} (\sigma + \pi)$, *for any* q, $q \geq 0$, *there exists cycle*

$$\sigma \overset{\mathcal{A}}{\twoheadrightarrow} (\sigma + \pi) \overset{\mathcal{A} \oplus \pi}{\twoheadrightarrow} (\sigma + 2\pi) \overset{\mathcal{A} \oplus 2\pi}{\twoheadrightarrow} \cdots \overset{\mathcal{A} \oplus (q-1)\pi}{\twoheadrightarrow} (\sigma + q\pi). \quad (5.28)$$

Proof: Use induction on q and Lemma 5.16. *Q.E.D.*

Note that Lemma 5.16 pertains only to "exiting" from two states with the same Boolean value. The analog for going "backward" from two states with the same Boolean value is not valid. More precisely, even if (5.24) holds, $\sigma_a \xrightarrow{\alpha} \sigma$ does *not* imply $(\sigma_a + \pi) \xrightarrow{\alpha \oplus \pi} \tau$, nor vice versa. See the following example.

Example 5.8: Figure 5.4 shows the state graph if we start at the initial state of 0010 for the PR set shown in Figure 5.1. Notice that even though **bool**(1110) = **bool**(3332), 3322 $\longrightarrow$ 3332 but 1100, not being a state in the graph, does not change to 1110. Similarly, **bool**(1010) = **bool**(3232) but 0010 $\longrightarrow$ 1010 while 2232 does not change to 3232. □

If a state graph contains a cycle $\sigma \twoheadrightarrow (\sigma + \pi)$, then π specifies fully the set of transitions whose occurrences after state σ leads to another state where all the variables have the same values. By Corollary 5.17, the same set of transitions can then occur at the new state and be repeated over and over. Thus, this cycle, with its associated period, describes a possible steady-state behavior of the system. Furthermore, the following results show that once a state is reached where the transitions associated with the period π can occur, then this set of transitions (with perhaps different occurrence numbers) can occur at any subsequent state.

Lemma 5.18 *If* $\sigma \overset{\mathcal{A}}{\twoheadrightarrow} (\sigma + \pi)$ *is a cycle and* $\sigma \xrightarrow{\gamma} \tau$, *then there exists cycle* $\tau \overset{\mathcal{B}}{\twoheadrightarrow} (\tau + \pi)$ *where* $\mathcal{B} = ((\mathcal{A} \cup \{(\gamma \oplus \pi)\}) \setminus \{\gamma\})$.

Proof: Consider the two cases:

Case 1: $(\gamma \notin \mathcal{A})$ By Lemma 5.4, there exists $\hat{\tau}$ such that $\tau \overset{\mathcal{A}}{\twoheadrightarrow} \hat{\tau}$. By Lemma 5.2, $\hat{\tau} - \tau = (\sigma + \pi) - \sigma = \pi$. Thus, the lemma is valid in this case.

Case 2: $(\gamma \in \mathcal{A})$ By Lemma 5.6,

$$\sigma \xrightarrow{\gamma} \tau \overset{\mathcal{A} \setminus \{\gamma\}}{\twoheadrightarrow} (\sigma + \pi).$$

$\neg x_3$	$\longrightarrow$	$x_0\uparrow$
$\neg x_3$	$\longrightarrow$	$x_1\uparrow$
$x_0 \vee x_1$	$\longrightarrow$	$x_2\uparrow$
$x_0 \wedge x_1 \wedge x_2$	$\longrightarrow$	$x_3\uparrow$
x_3	$\longrightarrow$	$x_0\downarrow$
x_3	$\longrightarrow$	$x_1\downarrow$
$\neg x_0 \wedge \neg x_1$	$\longrightarrow$	$x_2\downarrow$
$\neg x_2$	$\longrightarrow$	$x_3\downarrow$

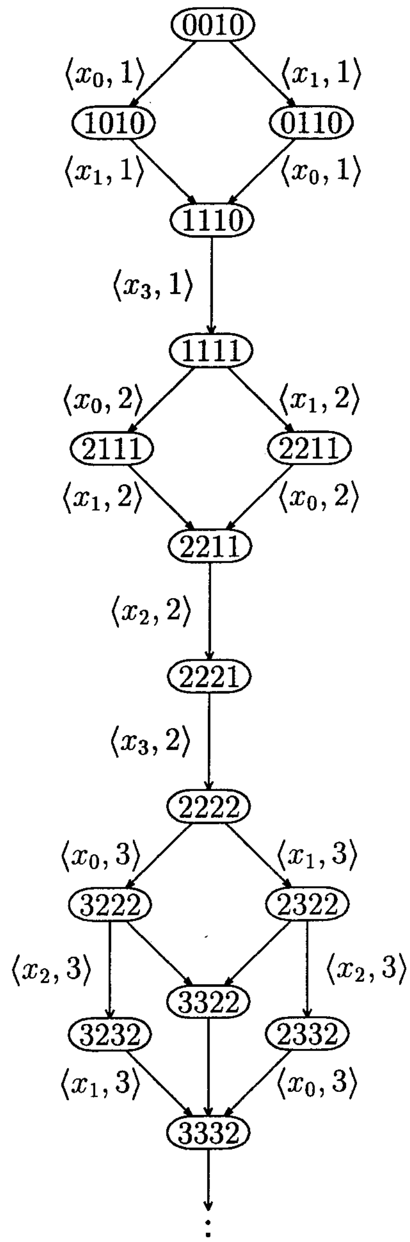

Figure 5.4: A state graph with initial state $\sigma_{\text{init}} = 0010$

Also, by Lemma 5.16, $\sigma \xrightarrow{\gamma} \tau$ implies $(\sigma+\pi) \xrightarrow{\gamma \oplus \pi} (\tau+\pi)$. Concatenating this edge to the path above establishes the claim. *Q.E.D.*

Lemma 5.19 *Given* $\sigma \overset{\mathcal{C}}{\twoheadrightarrow} \tau$ *and a cycle* $\sigma \twoheadrightarrow (\sigma+\pi)$, *there exists cycle* $\tau \twoheadrightarrow (\tau+\pi)$.

Proof: Use Lemma 5.18 and induction on the size of $\mathcal{C}$. *Q.E.D.*

Note that as the following example shows, in Lemma 5.19, $\tau \twoheadrightarrow (\tau+\pi)$ does not imply $\sigma \twoheadrightarrow (\sigma+\pi)$.

Example 5.9: In Figure 5.4, the cycle $1010 \twoheadrightarrow 3232$ implies the cycle $1110 \twoheadrightarrow 3332$ and so on. However, it does not imply there is a cycle starting from 0010 even though $0010 \longrightarrow 1010$. □

5.4 Sub-cycles and Minimal Periods

Definition: The cycle

$$\tilde{\sigma} \overset{\tilde{\mathcal{A}}}{\twoheadrightarrow} (\tilde{\sigma}+\tilde{\pi}) \tag{5.29}$$

is a *sub-cycle* of the cycle

$$\sigma \overset{\mathcal{A}}{\twoheadrightarrow} (\sigma+\pi) \tag{5.30}$$

if $\tilde{\mathcal{A}} \subset \mathcal{A}$. A cycle is *minimal* if it has no sub-cycles.

By the transitivity of $\subset$, any sub-cycle of (5.29) is also a sub-cycle of (5.30). Consequently, every non-minimal cycle contains a minimal sub-cycle. Unfortunately, a sub-cycle as defined above is too unrestrictive to be useful in many proofs. Hence, in the next subsection, we will show that any non-minimal cycle contains a sub-cycle that satisfies certain conditions.

5.4.1 Normal Sub-cycles

Definition: A sub-cycle (5.29) of the cycle (5.30) is *normal* if

$$\exists \mathcal{D} :: (\sigma \overset{\mathcal{D}}{\twoheadrightarrow} \tilde{\sigma}) \wedge (\mathcal{D} \cap \mathcal{A} = \emptyset). \tag{5.31}$$

One useful property of a normal sub-cycle is that there is a simple relationship between its period and that of its corresponding cycle.

Lemma 5.20 *Given cycles (5.29) and (5.30) that satisfy (5.31), (5.29) is a sub-cycle of (5.30) if and only if* $\tilde{\pi} < \pi$.

Proof: Since $\mathcal{A} \cap \mathcal{D} = \emptyset$, by Lemma 5.5, $\tilde{\sigma} \overset{\mathcal{A}}{\twoheadrightarrow} (\tilde{\sigma} + \pi)$. By Lemma 5.3, $\mathcal{A} = \tilde{\mathcal{A}} \Leftrightarrow \pi = \tilde{\pi}$; by Lemma 5.2, $\mathcal{A} \subseteq \tilde{\mathcal{A}} \Leftrightarrow \pi \leq \tilde{\pi}$. *Q.E.D.*

We will next show that any non-minimal cycle contains a normal sub-cycle.

Lemma 5.21 *Given a cycle* $\sigma \overset{\mathcal{A}}{\twoheadrightarrow} (\sigma + \pi)$ *and a path* $\sigma \overset{\tilde{\mathcal{D}}}{\twoheadrightarrow} \tau$, *there exist* $q \geq 0$, ϕ, *and* $\mathcal{D}$ *such that* $\mathbf{span}(\mathcal{D}) \cap \mathbf{span}(\pi) = \emptyset$ *and*

$$\sigma \overset{\mathcal{D}}{\twoheadrightarrow} \phi \twoheadrightarrow \tau \twoheadrightarrow (\phi + q\pi). \tag{5.32}$$

Proof: Partition $\tilde{\mathcal{D}}$ into the sets

$$\mathcal{D} = \{\alpha : \alpha \in \tilde{\mathcal{D}} \wedge \mathbf{var}(\alpha) \notin \mathbf{span}(\pi) : \alpha\} \tag{5.33}$$

and $\mathcal{C} = \tilde{\mathcal{D}} \setminus \mathcal{D}$. Note that $\mathcal{D} \cap \mathcal{A} = \emptyset$. Next, choose q large enough so that for any variable index k, $\langle x_k, l \rangle \in \mathcal{C}$ implies $l + \sigma[k] \leq q\pi[k]$. By Lemma 5.17, there exists cycle

$$\sigma \overset{\mathcal{B}}{\twoheadrightarrow} (\sigma + q\pi) \tag{5.34}$$

and $\mathcal{C} \subseteq \mathcal{B}$ by the choice of q. Applying Lemma 5.7 to (5.34) and $\sigma \twoheadrightarrow \tau$ implies there exists $\tilde{\phi}$ such that

$$(\sigma + q\pi) \overset{\tilde{\mathcal{D}} \setminus \mathcal{B}}{\twoheadrightarrow} \tilde{\phi} \wedge \tau \overset{\mathcal{B} \setminus \tilde{\mathcal{D}}}{\twoheadrightarrow} \tilde{\phi}. \tag{5.35}$$

By (5.33) and $\mathcal{C} \subseteq \mathcal{B}$, $(\tilde{\mathcal{D}} \setminus \mathcal{B}) = ((\mathcal{D} \cup \mathcal{C}) \setminus \mathcal{B}) = \mathcal{D}$ and $(\mathcal{D} \ominus q\pi) = \mathcal{D}$. So, the first half of (5.35) implies $\sigma \overset{\mathcal{D}}{\twoheadrightarrow} (\tilde{\phi} - q\pi)$ by Lemma 5.16. Let $\phi = \tilde{\phi} - q\pi$; then the other paths in (5.32) follows from $\mathcal{D} \subseteq \tilde{\mathcal{D}}$ and the second half of (5.35). *Q.E.D.*

Lemma 5.22 *If*

$$\sigma \overset{\mathcal{A}}{\twoheadrightarrow} (\sigma + \pi) \tag{5.36}$$

has a sub-cycle of period $\widetilde{\pi}$, then it has a sub-cycle

$$\tau \overset{\widetilde{\mathcal{A}}}{\twoheadrightarrow} (\tau + \widetilde{\pi}) \tag{5.37}$$

such that $\sigma \twoheadrightarrow \tau$.

Proof: Let

$$\widetilde{\sigma} \overset{\widetilde{\mathcal{A}}}{\twoheadrightarrow} (\widetilde{\sigma} + \widetilde{\pi}) \tag{5.38}$$

be a sub-cycle of (5.36). Let τ be the c.c.d. of σ and $\widetilde{\sigma}$ with $\widetilde{\sigma} \overset{\mathcal{C}}{\twoheadrightarrow} \tau$. Applying Lemma 5.7 to $\widetilde{\sigma} \overset{\mathcal{C}}{\twoheadrightarrow} \tau$ and (5.38) yields the existence of $\widehat{\tau}$ such that

$$\tau \overset{\widetilde{\mathcal{A}} \setminus \mathcal{C}}{\twoheadrightarrow} \widehat{\tau}.$$

Now, if $\langle x_k, l \rangle \in \widetilde{\mathcal{A}}$, then $\sigma[k] < l$ and $\widetilde{\sigma}[k] < l$. So, $\tau[k] < l$ by Lemma 5.8 and $\langle x_k, l \rangle \notin \mathcal{C}$. Hence, $\widetilde{\mathcal{A}} \cap \mathcal{C} = \emptyset$ and so $\widehat{\tau} = (\tau + \widetilde{\pi})$. *Q.E.D.*

Lemma 5.23 *If a cycle has a sub-cycle of period $\widetilde{\pi}$, then it has a normal subcycle of period $\widetilde{\pi}$.*

Proof: Let the cycle be (5.36) and, by Lemma 5.22, we can assume that it has a sub-cycle (5.37) with $\sigma \overset{\widetilde{\mathcal{D}}}{\twoheadrightarrow} \tau$. Then, by Lemma 5.2, $\widetilde{\mathcal{A}} \subset \mathcal{A}$ implies $\widetilde{\pi} < \pi$. Next, apply Lemma 5.21 to obtain (5.32). By Lemma 5.19, $\tau \twoheadrightarrow (\phi + q\pi)$ and (5.37) imply $(\phi + q\pi) \twoheadrightarrow (\phi + q\pi + \widetilde{\pi})$. By Lemma 5.16, $\phi \twoheadrightarrow (\phi + \widetilde{\pi})$. Setting $\widetilde{\sigma}$ to ϕ implies there exists the cycle $\widetilde{\sigma} \twoheadrightarrow (\widetilde{\sigma} + \widetilde{\pi})$. By Lemma 5.20, this is a normal sub-cycle of (5.37) since $\mathcal{D} \cap \mathcal{A} = \emptyset$ and $\widetilde{\pi} < \pi$. *Q.E.D.*

5.4.2 Minimal Periods

Our next goal is to establish Theorem 5.1 which states that a cycle with period π is minimal implies all cycles with period π are minimal. Toward that end, we need to establish some preliminary results.

Lemma 5.24 *Let* $\tilde{\sigma} \overset{\tilde{\mathcal{A}}}{\twoheadrightarrow} (\tilde{\sigma} + \tilde{\pi})$ *be a normal subcycle of* $\sigma \overset{\mathcal{A}}{\twoheadrightarrow} (\sigma + \pi)$ *with* $\sigma \overset{\mathcal{D}}{\twoheadrightarrow} \tilde{\sigma}$*. Then, the followings hold:*

$$(\tilde{\sigma} + \tilde{\pi}) \overset{\mathcal{A} \setminus \tilde{\mathcal{A}}}{\twoheadrightarrow} (\tilde{\sigma} + \pi) \wedge (\sigma + \pi) \overset{\mathcal{D}}{\twoheadrightarrow} (\tilde{\sigma} + \pi); \tag{5.39}$$

and

$$\tilde{\sigma} \twoheadrightarrow (\tilde{\sigma} + \pi - \tilde{\pi}) \text{ is a subcycle of } \sigma \twoheadrightarrow (\sigma + \pi). \tag{5.40}$$

Proof: Since $\mathcal{A} \cap \mathcal{D} = \emptyset$, $(\tilde{\sigma} + \pi)$ is the c.c.d. of $(\tilde{\sigma} + \tilde{\pi})$ and $(\sigma + \pi)$ and, so, (5.39) holds. By the first part of (5.39) and $\mathbf{bool}(\tilde{\sigma}) = \mathbf{bool}(\tilde{\sigma} + \tilde{\pi})$, Lemma 5.16 implies $\tilde{\sigma} \twoheadrightarrow (\tilde{\sigma} + \pi - \tilde{\pi})$. But $(\tilde{\pi} \neq 0) \wedge (\tilde{\pi} < \pi)$ is equivalent to $((\pi - \tilde{\pi}) \neq 0) \wedge ((\pi - \tilde{\pi}) < \pi)$. So, by Lemma 5.20, (5.40) holds. *Q.E.D.*

Lemma 5.25 *If the cycle*

$$\sigma \overset{\mathcal{A} \setminus \{\alpha\}}{\twoheadrightarrow} \sigma_a \overset{\alpha}{\longrightarrow} (\sigma + \pi) \tag{5.41}$$

is not minimal, then it has a normal sub-cycle such that α *does not occur in the normal sub-cycle.*

Proof: Let $\alpha = \langle x_k, (\sigma + \pi)[k] \rangle$. Let $\tilde{\sigma} \twoheadrightarrow (\tilde{\sigma} + \tilde{\pi})$ be a sub-cycle of (5.41). If α occurs in the sub-cycle, then $(\tilde{\sigma} + \tilde{\pi})[k] = (\sigma + \pi)[k]$. So, $(\tilde{\sigma} + \pi - \tilde{\pi})[k] = 0$ and, so, α does not occur in $\tilde{\sigma} \twoheadrightarrow (\tilde{\sigma} + \pi - \tilde{\pi})$ which, by (5.40), is another sub-cycle of (5.41). *Q.E.D.*

Lemma 5.26 *Given cycles*

$$\sigma \overset{\mathcal{A}}{\twoheadrightarrow} (\sigma + \pi) \tag{5.42}$$

and

$$\tau \overset{\mathcal{B}}{\twoheadrightarrow} (\tau + \pi), \tag{5.43}$$

and $\sigma \overset{\gamma}{\longrightarrow} \tau$*, (5.42) is minimal if and only if (5.43) is minimal.*

Proof: By Lemma 5.18,

$$\mathcal{B} = (\mathcal{A} \cup \{\gamma \oplus \pi\}) \setminus \{\gamma\}. \tag{5.44}$$

<u>$(\Leftarrow)$</u> Suppose (5.42) is not minimal. Then, there exists a normal sub-cycle

$$\tilde{\sigma} \xrightarrow[*]{\tilde{\mathcal{A}}} (\tilde{\sigma} + \tilde{\pi}). \tag{5.45}$$

Let $\sigma \xrightarrow[*]{\mathcal{D}} \tilde{\sigma}$ and $\mathcal{A} \cap \mathcal{D} = \emptyset$. Also, note that $\tilde{\pi} < \pi$ by Lemma 5.20.

Next, by Lemma 5.7, there exists $\tilde{\tau}$ such that

$$\tau \xrightarrow[*]{\mathcal{D} \setminus \{\gamma\}} \tilde{\tau} \wedge \tilde{\sigma} \xrightarrow[*]{\{\gamma\} \setminus \mathcal{D}} \tilde{\tau}. \tag{5.46}$$

By Lemma 5.19, (5.45) and the second part of (5.46) imply the existence of

$$\tilde{\tau} \xrightarrow[*]{\tilde{\mathcal{B}}} (\tilde{\tau} + \tilde{\pi}). \tag{5.47}$$

Now, if $\langle x_k, l \rangle \in \mathcal{D} \setminus \{\gamma\}$, then by Lemma 5.2, $\langle x_k, l \rangle$ is in $\mathcal{D}$ and therefore not in $\mathcal{A}$ since $\mathcal{D} \cap \mathcal{A} = \emptyset$. Thus, $\pi[k] = 0$. So, since $\textbf{span}(\mathcal{B}) = \textbf{span}(\pi)$, $(\mathcal{D} \setminus \{\gamma\}) \cap \mathcal{B} = \emptyset$ and the hypothesis of Lemma 5.20 is satisfied for (5.47) and (5.43). Since $\tilde{\pi} < \pi$, (5.47) is a sub-cycle of (5.43). Consequently, we have demonstrated (5.42) is not minimal implies (5.43) is not minimal.

<u>$(\Rightarrow)$</u> Suppose (5.43) is not minimal. Let (5.47) be one of its normal sub-cycle. Then there exists $\tilde{\mathcal{D}}$ such that

$$\sigma \xrightarrow{\gamma} \tau \xrightarrow[*]{\tilde{\mathcal{D}}} \tilde{\tau} \tag{5.48}$$

with $\tilde{\mathcal{D}} \cap \mathcal{B} = \emptyset$. The last equality implies, by (5.44) and $\gamma \notin \tilde{\mathcal{D}}$, $\tilde{\mathcal{D}} \cap \mathcal{A} = \emptyset$.

There are two cases to considered:

<u>Case 1</u>: $(\gamma \notin \mathcal{A})$ In this case, (5.44) implies $\mathcal{B} = \mathcal{A}$. So, $\tilde{\mathcal{D}} \cap \mathcal{B} = \emptyset$ implies $(\tilde{\mathcal{D}} \cup \{\gamma\}) \cap \mathcal{A} = \emptyset$. Thus, by (5.48) and Lemma 5.20, (5.47) is a sub-cycle of (5.42) which is therefore not minimal.

<u>Case 2</u>: $(\gamma \in \mathcal{A})$ In this case, by Lemma 5.6, we have $\sigma \xrightarrow{\gamma} \tau$ and

$$\tau \xrightarrow[*]{\mathcal{A} \setminus \{\gamma\}} (\sigma + \pi) \xrightarrow{\gamma \oplus \pi} (\tau + \pi). \tag{5.49}$$

Now, by definition,

$$(\tilde{\mathcal{B}} \subset \mathcal{B}) \wedge (\mathcal{B} = (\mathcal{A} \setminus \{\gamma\}) \cup \{\gamma \oplus \pi\}).$$

But, by Lemma 5.25, we can assume $(\gamma \oplus \pi) \notin \tilde{\mathcal{B}}$. Consequently,

$$\tilde{\mathcal{B}} \subseteq (\mathcal{A} \setminus \{\gamma\}) \subset \mathcal{A}$$

with the last inclusion due to $\gamma \in \mathcal{A}$. These relationships imply (5.47) is a sub-cycle of (5.42) which is therefore not minimal. Thus, we have also demonstrated that (5.42) is minimal implies (5.43) is minimal. *Q.E.D.*

Lemma 5.27 *Given cycles (5.42) and (5.43) and $\sigma \twoheadrightarrow\!\!\!\!\star\;\tau$, (5.42) is minimal if and only if (5.43) is minimal.*

Proof: Use Lemma 5.26 and induction on the length of the path $\sigma \twoheadrightarrow\!\!\!\!\star\;\tau$. *Q.E.D.*

Theorem 5.1 *A cycle with period π is minimal implies all cycles with period π are minimal.*

Proof: Let (5.42) be a minimal cycle of period π. Let (5.43) be any other cycle of period π. Let ϕ be the c.c.d. of σ and τ. By Lemma 5.19, there exists cycle

$$\phi \twoheadrightarrow\!\!\!\!\star\;(\phi + \pi). \tag{5.50}$$

By Lemma 5.27, (5.42) is minimal implies (5.50) is minimal which, in turn, implies (5.43) is minimal. *Q.E.D.*

This result allows us to make the following definition:

Definition: The state offset π is a *minimal period* of a state graph, if there exists a minimal cycle with that period.

5.5 Non-separable Graphs

Theorem 5.2 *If there exist minimal cycles with periods π_a and π_b, then either $\pi_a = \pi_b$ or $\mathbf{span}(\pi_a) \cap \mathbf{span}(\pi_b) = \emptyset$.*

Proof: Let the two cycles be $\sigma_a \xrightarrow{*} (\sigma_a + \pi_a)$ and $\sigma_b \xrightarrow{*} (\sigma_b + \pi_b)$. Let τ be the c.c.d. of σ_a and σ_b. Then, by Lemma 5.19,

$$\tau \xrightarrow{*} (\tau + \pi_a) \wedge \tau \xrightarrow{*} (\tau + \pi_b). \tag{5.51}$$

Suppose $\pi_a \neq \pi_b$. Let ϕ be the c.c.d. of $(\tau + \pi_a)$ and $(\tau + \pi_b)$. Then, by Lemma 5.8,

$$\begin{aligned} \forall k :: \phi[k] &= \mathbf{max}\{(\tau + \pi_a)[k], (\tau + \pi_b)[k]\} \\ &= \tau[k] + \mathbf{max}\{\pi_a[k], \pi_b[k]\}. \end{aligned} \tag{5.52}$$

Also, by Lemma 5.16, (5.51) implies $(\tau + \pi_a + \pi_b)$ is a state. Since the maximum of two non-negative numbers is no greater than their sum, by Lemma 5.9,

$$(\tau + \pi_a) \xrightarrow{*} \phi \xrightarrow{*} (\tau + \pi_a + \pi_b). \tag{5.53}$$

By Lemma 5.16, this path implies

$$\tau \xrightarrow{*} (\phi - \pi_a) \xrightarrow{*} (\tau + \pi_b). \tag{5.54}$$

Now, since $\mathbf{bool}(\tau + \pi_a) = \mathbf{bool}(\tau) = \mathbf{bool}(\tau + \pi_b)$, by (5.52), $\mathbf{bool}(\phi) = \mathbf{bool}(\tau)$. Consequently, $\tau \xrightarrow{*} (\phi - \pi_a)$ is a sub-cycle of $\tau \xrightarrow{*} (\tau + \pi_b)$, which contradicts the hypothesis that π_b is a minimal period, unless $\tau = \phi - \pi_a$ or $\phi - \pi_a = \tau + \pi_b$. If $(\phi - \pi_a) = \tau$, then due to (5.52), $\pi_b < \pi_a$ and, by Lemma 5.9, $\tau \xrightarrow{*} (\tau + \pi_b) \xrightarrow{*} (\tau + \pi_a)$ and π_a is not a minimal period. So, $(\phi - \pi_a) = (\tau + \pi_b)$ which implies, for any variable index k, $\mathbf{max}\{\pi_a[k], \pi_b[k]\} = \pi_a[k] + \pi_b[k]$. This equality implies $\pi_a[k] = 0 \vee \pi_b[k] = 0$ and the theorem is established.
Q.E.D.

Theorem 5.2 states that if a state graph has different minimal periods, then the sets of variables spanned by these periods are disjoint[4]. Note that by Lemma 5.16, transitions on the variables in each of these sets can occur repeatedly. Furthermore, since the two sets have no variables in common, if the PR set ever reached a state where both cycles are possible, then, by Lemma 5.5, the occurrences of transitions in one set are independent of the occurrences of transitions in another set. These observations mean that the

[4]In [37], it has only been shown that the sets of variables spanned by the periods of cycles starting from a given state are disjoint.

original PR set contains two or more sub-components which operate independently of each other, except, perhaps, for some initial transient interactions. Consequently, we make the following definition.

Definition: A state graph is *separable* if it contains minimal cyles with different periods. It is *non-separable* if it is not separable.

Chapter 6

Index-Priority Simulation

In this chapter, we will present and prove the correctness of the *index-priority simulation* algorithm which finds all minimal periods in a state graph. It turns out that the number of simulation steps can be greatly reduced if it is known that the input graphs are *uniform*. Hence, the first part of the chapter will study these graphs and present some criteria for identifying them. The correctness of the actual algorithm is demonstrated in Section 6.4.

6.1 Uniform Graphs

Definition: A cycle

$$\sigma \xrightarrow{\mathcal{A}}\!\!\!\!\!\!{}^{*}\; (\sigma + \pi) \tag{6.1}$$

is *uniform* if it is minimal or it has a sub-cycle

$$\sigma \xrightarrow{\widetilde{\mathcal{A}}}\!\!\!\!\!\!{}^{*}\; (\sigma + \widetilde{\pi}). \tag{6.2}$$

A graph is *uniform* if all its cycles are uniform.

Before giving the motivation for this definition, the following useful property of uniform cycles should be established.

Lemma 6.1 *If cycle (6.1) is uniform that there exist an integer $p > 0$ such that (6.1) can be written as a concatenation of p minimal cycles. In other*

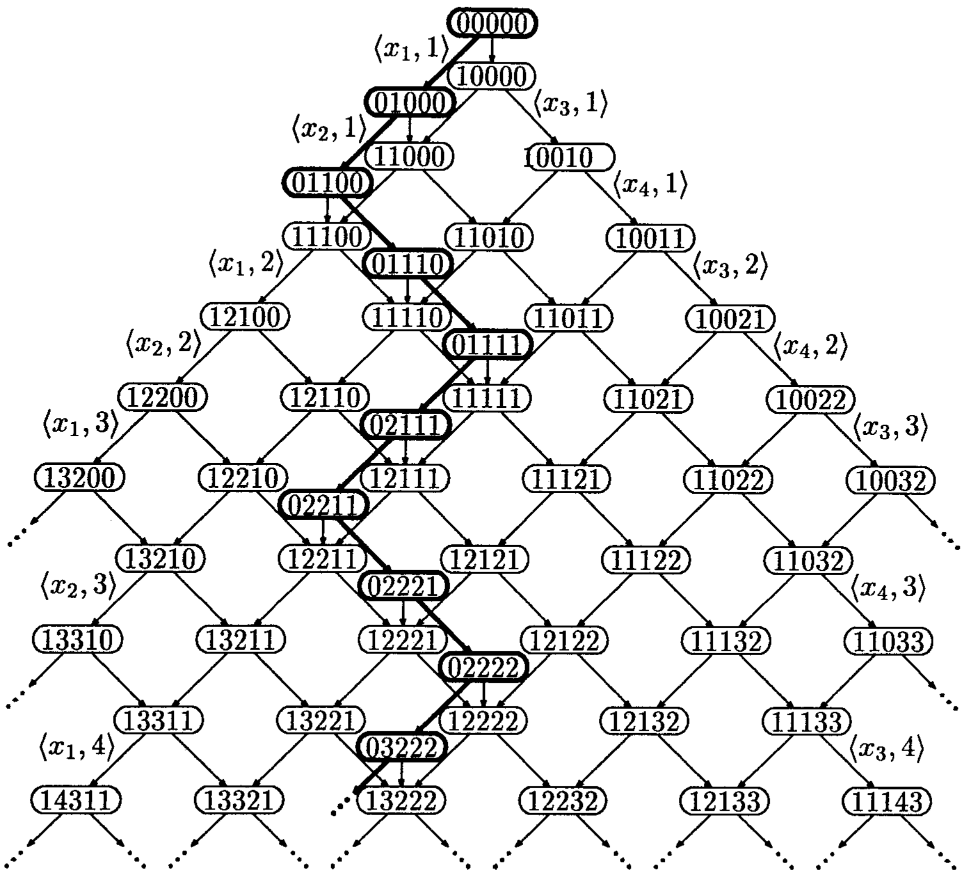

Figure 6.1: Non-uniform separable graph

words, there exists a set of minimal cycles

$$\{i : 0 \leq i < p : \sigma_i \overset{\star}{\rightarrow} \sigma_{i+1}\} \tag{6.3}$$

such that $\sigma_0 = \sigma$ *and* $\sigma_p = (\sigma + \pi)$.

Proof: Use induction on the length of the uniform cycle. Since there are no cycles with length zero, the base case holds. Next, assume the lemma holds for all cycles with length less than that of (6.1). If (6.1) is minimal, then set $p = 1$, and we are done. Otherwise, by the above definition and Lemma 5.24, $\sigma \overset{\star}{\rightarrow} (\sigma + \widetilde{\pi})$ and $(\sigma + \widetilde{\pi}) \overset{\star}{\rightarrow} (\sigma + \pi)$ are cycles. By the inductive hypothesis, each of these cycles can be written as a concatenation of minimal cycles; therefore, (6.1) can also be so written. *Q.E.D.*

The following example gives the motivation for the need of introducing uniform graphs.

Example 6.1: Consider the following PR set:

$$\begin{array}{lcl}
\neg x_0 \wedge \neg x_4 \vee x_0 \wedge \neg x_2 & \rightarrow & x_1\uparrow \\
\neg x_0 \wedge x_1 \vee x_0 \wedge x_1 & \rightarrow & x_2\uparrow \\
\neg x_0 \wedge x_2 \vee x_0 \wedge \neg x_4 & \rightarrow & x_3\uparrow \\
\neg x_0 \wedge x_3 \vee x_0 \wedge x_3 & \rightarrow & x_4\uparrow \\
 & & \\
\neg x_0 \wedge x_4 \vee x_0 \wedge x_2 & \rightarrow & x_1\downarrow \\
\neg x_0 \wedge \neg x_1 \vee x_0 \wedge \neg x_1 & \rightarrow & x_2\downarrow \\
\neg x_0 \wedge \neg x_2 \vee x_0 \wedge x_4 & \rightarrow & x_3\downarrow \\
\neg x_0 \wedge \neg x_3 \vee x_0 \wedge \neg x_3 & \rightarrow & x_4\downarrow
\end{array}$$

$$\begin{array}{lcl}
\neg x_4 \vee \neg x_2 \vee x_3 & \rightarrow & x_0\uparrow \\
\textbf{false} & \rightarrow & x_0\downarrow .
\end{array}$$

Its state graph is shown in Figure 6.1 where the vertical edges represent the occurrence of $\langle x_0, 1\rangle$, the bold states are those with x_0 **false**, and the bold edges are the events effecting changes among the bold states. (It may help to visualize the bold states and events as being "above" the rest of the graph.)

Some of the guards in the text above, such as the one for $x_2\uparrow$, have been written redundantly to show that the PR set operates in either of two modes. If x_0 is **false**, then only the state changes marked in bold can occur. Once x_0 becomes **true**, only those (non-vertical) state changes not marked in bold can occur.

The graph is separable with minimal periods 02200 and 00022. Since $00000 \overset{*}{\rightarrow} 02222$ is a cycle that contains the sub-cycle $10000 \overset{*}{\rightarrow} 12200$, it is not minimal. It is also not uniform since it contains no sub-cycle starting at 00000. Note that if x_0 never becomes **true** — i.e., only the bold part of the graph is valid — then $00000 \overset{*}{\rightarrow} 02222$ would be a minimal cycle. □

As illustrated above, it is difficult to decide if a cycle is minimal for this example since the PR set has two "modes of operation" — one when it is in the bolds states and another when it is not. In fact, unless extra care has been taken to examine all possible modes, any algorithm could, conceivably, err in regarding a non-minimal cycle as minimal because no sub-cycles can be found in the current mode. As shown in a later section, all non-uniform

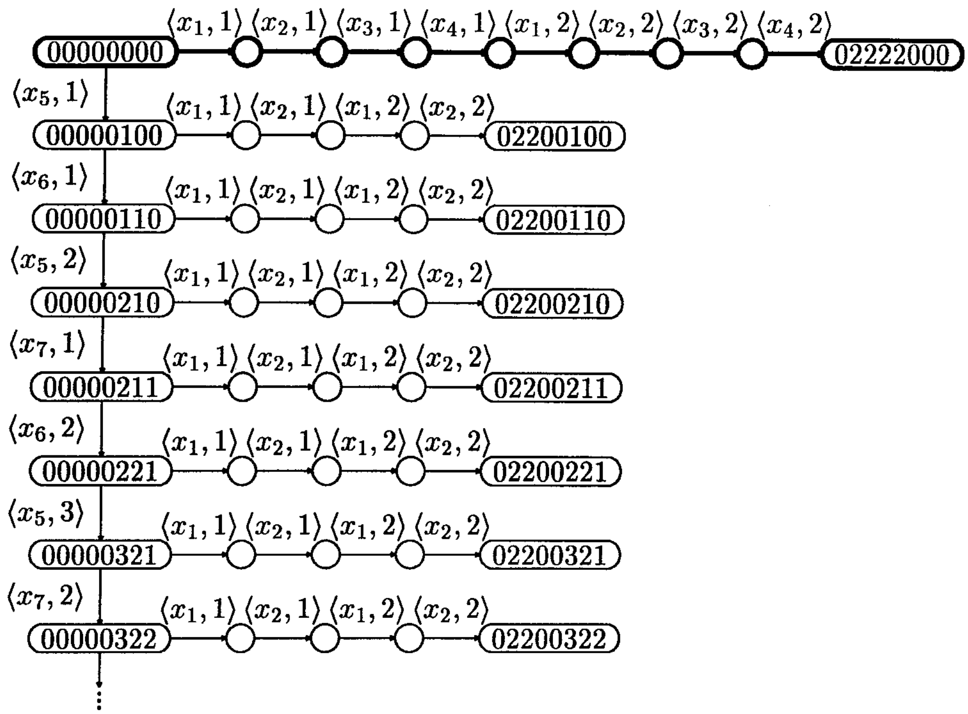

Figure 6.2: Non-uniform separable graph with no terminating events

graphs have more than one mode of operations; thus, it is important to investigate the properties of these graphs.

In Example 6.1, the variable associated with the "mode-switching" event, x_0, changes value only once. However, as the next example illustrates, even if all transitions occur infinitely often, the graph may still be non-uniform.

Example 6.2: In the PR set of Example 6.1, remove the PR's for x_0, replace each occurrence of x_0 in a guard with $(x_5 \vee x_6 \vee x_7)$ and add the following PR's:

$$\begin{array}{lcllcl}
\neg x_7 \wedge x_5 & \rightarrow & x_6\uparrow & \quad x_7 & \rightarrow & x_6\downarrow \\
x_6 & \rightarrow & x_5\downarrow & \quad \neg x_6 & \rightarrow & x_5\uparrow \\
\neg x_5 \wedge x_6 & \rightarrow & x_7\uparrow & \quad x_5 & \rightarrow & x_7\downarrow .
\end{array}$$

Now, all transitions (except those for x_0) occur infinitely often. But there is still a mode-switching event which is $\langle x_5, 1\rangle$ as Figure 6.2 attempts to illustrate. Imagine the vertical axis of Figure 6.2 as a vertical axis superimposed on Figure 6.1 and each horizontal line in Figure 6.2 as a path in the cross-sectional plane of Figure 6.1. Before $\langle x_5, 1\rangle$ occurs, the only possible path corresponds to the bold edges of Figure 6.1. Once $\langle x_5, 1\rangle$ have occurred, then the PR set switches mode and cycles with minimal periods are possible. □

As our final example, we present a non-uniform non-separable graph.

Example 6.3: Consider the following PR set:

$$
\begin{array}{lcl}
\neg x \wedge \neg a3 \wedge \neg c \vee x \wedge \neg a3 \wedge \neg c & \rightarrow & a1\uparrow \\
\neg x \wedge a1 \wedge \neg a3 \vee x \wedge a1 \wedge \neg a3 & \rightarrow & a2\uparrow \\
\neg x \wedge a2 \wedge (b1 \wedge \neg b2 \vee c) \vee x \wedge a2 \wedge (\neg b3 \vee b2 \vee c) & \rightarrow & a3\uparrow \\
\neg x \wedge a3 \wedge c \vee x \wedge a3 \wedge c & \rightarrow & a2\downarrow \\
\neg x \wedge \neg a2 \wedge a3 \wedge \neg b3 \vee x \wedge \neg a2 \wedge a3 & \rightarrow & a1\downarrow \\
\neg x \wedge \neg a1 \wedge (c \wedge b3 \vee b2 \wedge \neg c) \vee & & \\
\qquad\qquad x \wedge \neg a1 \wedge (b3 \vee \neg b1 \vee \neg c) & \rightarrow & a3\downarrow \\
\\
\neg x \wedge \neg b3 \wedge \neg c \vee x \wedge \neg b3 \wedge \neg c & \rightarrow & b1\uparrow \\
\neg x \wedge b1 \wedge \neg b3 \wedge a3 \vee x \wedge b1 \wedge \neg b3 & \rightarrow & b2\uparrow \\
\neg x \wedge b2 \wedge (a3 \wedge \neg c \wedge a1 \vee c \wedge \neg a1) \vee & & \\
\qquad\qquad x \wedge b2 \wedge (\neg a3 \vee a1 \vee c) & \rightarrow & b3\uparrow \\
\neg x \wedge b3 \wedge c \vee x \wedge b3 \wedge c & \rightarrow & b2\downarrow \\
\neg x \wedge \neg b2 \wedge b3 \vee x \wedge \neg b2 \wedge b3 & \rightarrow & b1\downarrow \\
\neg x \wedge \neg b1 \wedge (\neg a3 \vee a1) \vee x \wedge \neg b1 \wedge (a3 \vee \neg a2 \vee \neg c) & \rightarrow & b3\downarrow \\
\\
\neg x \wedge (a3 \wedge b3 \vee a2 \wedge b2 \wedge \neg a3) \vee x \wedge a2 \wedge b2 & \rightarrow & c\uparrow \\
\neg x \wedge \neg a1 \wedge \neg b1 \wedge \neg b3 \vee x \wedge \neg a1 \wedge \neg b1 & \rightarrow & c\downarrow \\
\textbf{true} & \rightarrow & x\uparrow \\
\textbf{false} & \rightarrow & x\downarrow.
\end{array}
$$

Figure 6.3 shows its state graph with the initial state of all variables **false**. To reduce cluttering, each state with x **false** is merged with the state that has the same value but with x **true** and is indicated as a bold circle. The bold edges represent all the events that are possible when x is **false**. Note that there is only one transition on x and $\langle x, 1\rangle$ serves as a mode-switching event whose occurrence causes the PR set to behave like the one in Example 3.2. All minimal cycles, like the one between the initial state and the state marked with a cross, have the same period. However, prior to $\langle x, 1\rangle$ occurring, there

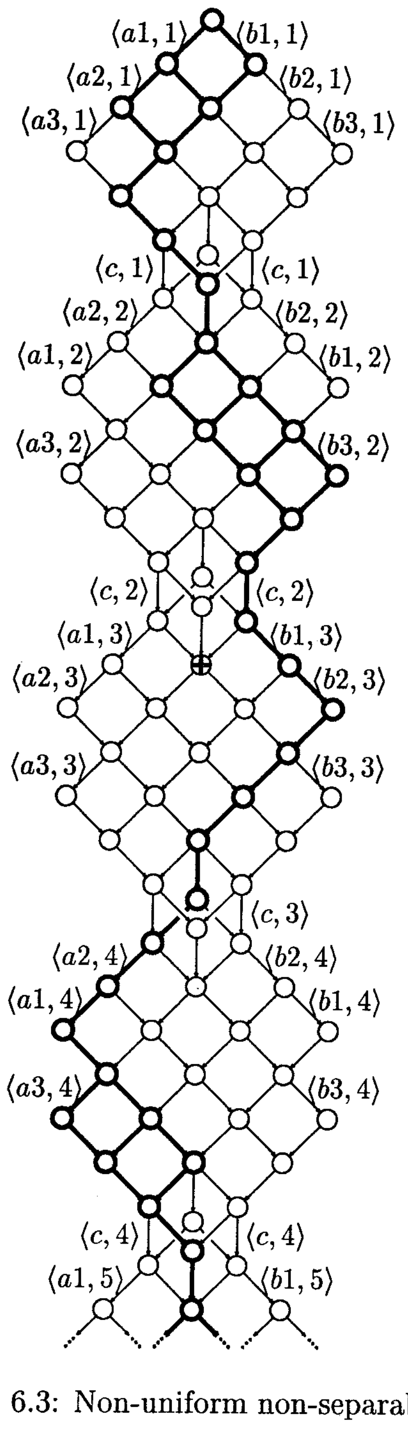

Figure 6.3: Non-uniform non-separable graph

is no minimal cycle and any cycle containing only bold edges is non-uniform or contains a non-uniform sub-cycle. □

6.2 Non-transitory States

In each of the previous examples, if we assume the firings of the PR's are weakly fair, then, eventually, the "mode-switching" event would occur and the original PR set would exhibit its steady-state behavior. This section shows that any graph contains a "non-transitory" state σ such that all cycles starting from a state reachable from σ are uniform. Furthermore, in Chapter 7, it will be shown that the behavior of the PR set after reaching a non-transitory state σ can be modeled as a set of repetitive XER-systems, one for each minimal period. Thus, even if a graph is non-uniform, to evaluate its steady-state performance, it is sufficient to find and analyze its minimal cycles.

Definition: A state σ is called a *non-transitory* state if

$$\forall \alpha : \mathbf{enb}(\alpha, \sigma) : (\ \exists \hat{\pi} : \hat{\pi} \text{ is a period spanning } \mathbf{var}(\alpha) : \sigma \xrightarrow{*} (\sigma + \hat{\pi})\ ). \quad (6.4)$$

The algorithm in Section 6.4 will show how to find a non-transitory state. Below we have established some of its properties.

Lemma 6.2 *If σ is a non-transitory state and $\sigma \overset{\mathcal{B}}{\xrightarrow{*}} \tau$, then there exists period π such that $\tau \xrightarrow{*} (\sigma + \pi)$.*

Proof: Use induction on the size of $\mathcal{B}$. If $|\mathcal{B}| = 0$, then let π be the zero-vector. Suppose $|\mathcal{B}| > 0$, then let

$$\sigma \overset{\tilde{\mathcal{B}}}{\xrightarrow{*}} \tilde{\tau} \xrightarrow{\beta} \tau.$$

By the inductive hypothesis, there exists $\tilde{\pi}$ such that $\tilde{\tau} \xrightarrow{*} (\sigma + \tilde{\pi})$ and $\mathbf{bool}(\sigma + \tilde{\pi}) = \mathbf{bool}(\sigma)$. If β occurs in the path $\tilde{\tau} \xrightarrow{*} (\sigma + \tilde{\pi})$, then let π be $\tilde{\pi}$ and we are done by Lemma 5.6. If β does not occur in the path, then, by stability, $\mathbf{enb}(\beta, \sigma + \tilde{\pi})$. By Lemma 5.16, $\mathbf{enb}(\beta \ominus \tilde{\pi}, \sigma)$, and, by the definition of non-transitory state, there exists period $\hat{\pi}$ such that

$$\sigma \overset{\mathcal{C}}{\xrightarrow{*}} (\sigma + \hat{\pi}) \wedge (\beta \ominus \tilde{\pi}) \in \mathcal{C}.$$

By Lemma 5.16,

$$(\sigma + \tilde{\pi}) \overset{\mathcal{C} \oplus \tilde{\pi}}{\twoheadrightarrow} (\sigma + \hat{\pi} + \tilde{\pi}) \wedge \beta \in (\mathcal{C} \oplus \tilde{\pi}).$$

Setting π to $\tilde{\pi} + \hat{\pi}$ establishes the lemma. *Q.E.D.*

Corollary 6.3 *If σ is a non-transitory state and $\sigma \twoheadrightarrow \tau$, then for any period $\hat{\pi}$, $\sigma \twoheadrightarrow (\sigma + \hat{\pi})$ if and only if $\tau \twoheadrightarrow (\tau + \hat{\pi})$.*

Proof: The "only-if" part is due to Lemma 5.19; the "if" part follows from applying that lemma to $\tau \twoheadrightarrow (\sigma + \pi)$, which is guaranteed by Lemma 6.2, and then using Lemma 5.16. *Q.E.D.*

Corollary 6.4 *If σ is a non-transitory state, then all cycles starting at σ are uniform.*

Proof: Suppose $\sigma \twoheadrightarrow (\sigma + \pi)$ has sub-cycle $\tau \twoheadrightarrow (\tau + \tilde{\pi})$. By Lemma 5.22, we can assume w.l.g. that $\sigma \twoheadrightarrow \tau$. Then, by Corollary 6.3, $\sigma \twoheadrightarrow (\sigma + \tilde{\pi})$ is also a sub-cycle. *Q.E.D.*

Corollary 6.5 *A state σ is a non-transitory state if and only if*

$$\begin{array}{l} \forall \alpha : \mathbf{enb}(\alpha, \sigma) : (\ \exists \hat{\pi} : \\ \qquad \hat{\pi} \textit{ is a minimum period spanning } \mathbf{var}(\alpha) : \sigma \twoheadrightarrow (\sigma + \hat{\pi})\ ). \end{array} \tag{6.5}$$

Proof: Follows directly from Corollary 6.4 and Lemma 6.1. *Q.E.D.*

Corollary 6.6 *If σ is a non-transitory state, then the existence of a cycle with period π implies $\sigma \twoheadrightarrow (\sigma + \pi)$.*

Proof: Let $\tau \twoheadrightarrow (\tau + \pi)$ be the cycle. Let ϕ be the c.c.d. of σ and τ. The result then follows from Corollary 6.3. *Q.E.D.*

Lemma 6.7 *If $\sigma \twoheadrightarrow \tau$ and σ is a non-transitory state, then, so is τ.*

Proof: By Lemma 6.2 and Lemma 5.19, there exists a period π such that

$$\sigma \overset{\mathcal{B}}{\twoheadrightarrow} \tau \overset{\mathcal{C}}{\twoheadrightarrow} (\sigma + \pi) \overset{\mathcal{B} \oplus \pi}{\twoheadrightarrow} (\tau + \pi).$$

Suppose $\langle x_k, l \rangle$ is enabled at τ. If $x_k \notin \mathbf{span}(\pi)$, then, by stability, $\mathbf{enb}(\langle x_k, l \rangle, \sigma + \pi)$ which implies $\mathbf{enb}(\langle x_k, l \rangle, \sigma)$. By Corollary 6.5, there exists a minimal period $\hat{\pi}$ such that $x_k \in \mathbf{span}(\hat{\pi})$ and $\sigma \twoheadrightarrow (\sigma + \hat{\pi})$. Then, by Lemma 5.19, $\tau \twoheadrightarrow (\tau + \hat{\pi})$. So, τ is a non-transitory state. *Q.E.D.*

Note that as the following example shows, even if every cycle starting from a state σ is uniform, σ may still not be a non-transitory state.

Example 6.4: Figure 6.4 shows a PR set and its state graph for the initial state where every variable is **false**. The short near-vertical edges are for $\langle x, 1 \rangle$ and states with x **false** and their connecting edges are shown in bold. All cycles starting at the initial state are uniform. However, the initial state is not a non-transitory state because there is no cycle containing $\langle x, 1 \rangle$.

Note that for any $i > 0$, event $\langle c, 2i+1 \rangle$ is caused by either $\langle b, 2i+1 \rangle$ or $\langle d, 2i \rangle$ and $\langle x, 1 \rangle$. Since there is no periodic index (i.e., i) in $\langle x, 1 \rangle$, the PR set cannot be modeled as a repetitive XER-system. However, if we assume that the PR set, due to fairness, has entered a non-transitory state where x has gone up, then x has no further bearing on the performance of the system. Consequently, we can ignore $\langle x, 1 \rangle$ and regard $\langle b, 2i+1 \rangle$ and $\langle d, 2i \rangle$ as possible causes of $\langle c, 2i+1 \rangle$. The details of these arguments will become clearer in Chapter 7; for now, it is sufficient to realize that a non-transitory state serves a more important purpose than the implication of uniform cycles. □

6.3 Detecting Non-Uniform Graphs

The algorithm presented in the next section guarantees to return minimal periods only if the graph is uniform. If the graph is not uniform, then the algorithm has to be re-run starting at the non-transitory state determined by the algorithm. Thus, for correctness, it is not *necessary* to determine if a graph is uniform and this section can be skipped with little loss of continuity. However, by recognizing a uniform graph, one can cut in half the simulation steps needed to find its minimal periods and, thus, it maybe worthwhile to

$\neg d$	$\longrightarrow$	$a\uparrow$
$\neg d \wedge c \vee a$	$\longrightarrow$	$b\uparrow$
$\neg d \wedge x \vee b$	$\longrightarrow$	$c\uparrow$
$a \wedge b \wedge c$	$\longrightarrow$	$d\uparrow$
d	$\longrightarrow$	$a\downarrow$
$d \wedge \neg a$	$\longrightarrow$	$b\downarrow$
$d \wedge \neg b$	$\longrightarrow$	$c\downarrow$
$\neg c$	$\longrightarrow$	$d\downarrow$
true	$\longrightarrow$	$x\uparrow$
false	$\longrightarrow$	$x\downarrow$

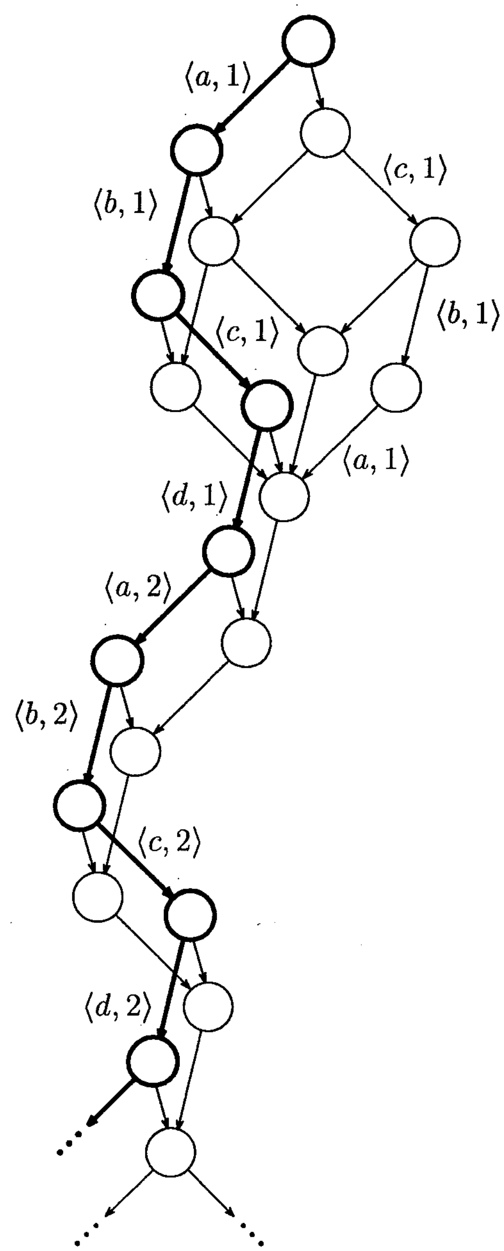

Figure 6.4: Non-transitory state in a compact graph

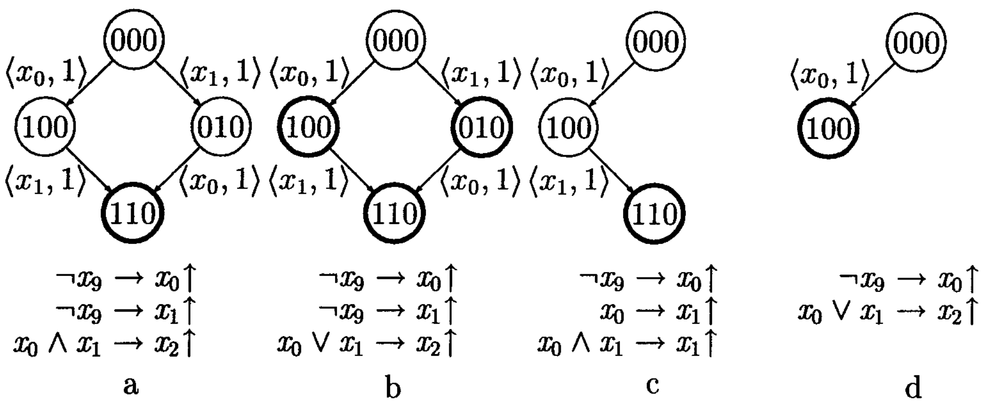

Figure 6.5: Examples of triggers for an event

ascertain whether this is the case. As can be seen from the previous examples, non-uniform graphs are fairly hard to construct and the following results give some easy-to-check sufficient conditions (Corollaries 6.16 to 6.18) for a graph to be uniform. To facilitate the discussion, we make the following definitions.

6.3.1 Disjunctively Enabled Events

Definition: An event α is a *trigger* for another event γ at state σ if

$$\sigma \xrightarrow{\alpha} \tau \wedge \neg\mathbf{enb}(\gamma, \sigma) \wedge \mathbf{enb}(\gamma, \tau).$$

Definition: An event γ is *disjunctively enabled* if there exists a state σ such that there are two distinct triggers of γ at σ. Each of these triggers is called a *disjunctive trigger* of γ.

Example 6.5: Figure 6.5 gives some examples of how the triggers of the event $\langle x_2, 1\rangle$ can behave. In each example, only the relevant PR's and state values are given (x_9 is just some arbitrary variable) and all states at which $\langle x_2, 1\rangle$ is enabled is marked in bold. In Figure 6.5a, both $\langle x_0, 1\rangle$ and $\langle x_1, 1\rangle$

can occur in the initial state and they are both triggers for $\langle x_2, 1\rangle$. The same observation holds in Figure 6.5b. Moreover, due to the disjunction in the guard for $x_2\uparrow$, $\langle x_0, 1\rangle$ and $\langle x_1, 1\rangle$ are also disjunctive triggers of $\langle x_2, 1\rangle$ at state 000. Note that the number of states at which $\langle x_2, 1\rangle$ is enabled is larger for the second case. In Figure 6.5c, only $\langle x_0, 1\rangle$ can occur in the initial state and so $\langle x_1, 1\rangle$ is the only trigger of $\langle x_2, 1\rangle$. In Figure 6.5d, $\langle x_0, 1\rangle$ is the only trigger of $\langle x_2, 1\rangle$ and there need not be any disjunctive trigger in spite of the disjunction in the guard for $\mathbf{tran}(\langle x_2, 1\rangle)$.

All four scenarios occur in practice. The first two are typical behavior for conjunctive and disjunctive guards. The third arises from strengthening a guard to prevent its misfiring at some undesirable state (i.e., state where $\neg x_0 \wedge x_1$ is **true**). Finally, the last example can be due to data-dependency or the effect of symmetrization caused by weakening the original guard x_0 with x_1. $\square$

As one may suspect from these examples, there is a correlation between the trigger of an event and its guard.

Lemma 6.8 *If α is a trigger of γ and the guard of* $\mathbf{tran}(\gamma)$ *is $B_0 \vee B_1 \vee \ldots \vee B_m$, then there exists at least one disjunct B_i such that B_i contains the literal* $\mathbf{lit}(\alpha)$.

Proof: By definition, there exist σ and τ such that $\sigma \xrightarrow{\alpha} \tau$ with $\neg\mathbf{enb}(\gamma, \sigma)$ and $\mathbf{enb}(\gamma, \tau)$. Now, $\mathbf{enb}(\gamma, \tau)$ implies there exists an i such that B_i is **true** at state τ. But, $\neg\mathbf{enb}(\gamma, \sigma)$ implies B_i is **false** at σ. By the definition of $\longrightarrow$, the only difference between σ and τ is that $\mathbf{lit}(\alpha)$ is **false** at σ and is **true** at τ. Thus, B_i contains $\mathbf{lit}(\alpha)$. *Q.E.D.*

Lemma 6.9 *If α and β are disjunctive triggers of γ at some state σ, and the guard of* $\mathbf{tran}(\gamma)$ *is $B_0 \vee B_1 \vee \ldots B_m$, then, there exist i and j, such that B_i contains* $\mathbf{lit}(\alpha)$ *but not* $\mathbf{lit}(\beta)$ *and B_j contains* $\mathbf{lit}(\beta)$ *but not* $\mathbf{lit}(\alpha)$.

Proof: By definition, there exist τ_a and τ_b such that $\sigma \xrightarrow{\alpha} \tau_a$, $\sigma \xrightarrow{\beta} \tau_b$, $\neg\mathbf{enb}(\gamma, \sigma)$, $\mathbf{enb}(\gamma, \tau_a)$, and $\mathbf{enb}(\gamma, \tau_b)$. By the proof of Lemma 6.8, there exists i such that B_i contains $\mathbf{lit}(\alpha)$, B_i is **false** at σ, and B_i is **true** at τ_a. If B_i contains $\mathbf{lit}(\beta)$, then B_i can be written as $\mathbf{lit}(\alpha) \wedge \mathbf{lit}(\beta) \wedge C$ for some Boolean expression C not containing $\mathbf{lit}(\alpha)$ or $\mathbf{lit}(\beta)$. Now, by definition of

$\sigma \xrightarrow{\beta} \tau_b$, the value of $\mathbf{lit}(\beta)$ is **false** at σ. So, the value of $\mathbf{lit}(\beta)$ remains **false** at τ_a which contradicts the fact that B_i is **true** at τ_a. Symmetric arguments for B_j establish the lemma. *Q.E.D.*

The following two results concerning disjunctively enabled events will be needed in Sub-section 6.3.3.

Lemma 6.10 *Given $n, m > 0$, and states and events such that*

$$\sigma_{0,0} \xrightarrow{\alpha_0} \sigma_{1,0} \xrightarrow{\alpha_1} \cdots \xrightarrow{\alpha_{n-1}} \sigma_{n,0},$$

$$\sigma_{0,0} \xrightarrow{\beta_0} \sigma_{0,1} \xrightarrow{\beta_1} \cdots \xrightarrow{\beta_{m-1}} \sigma_{0,m},$$

and $\{i : 0 \leq i < n : \alpha_i\} \cap \{j : 0 \leq j < m : \beta_j\} = \emptyset$, for any γ, if

$$\neg\mathbf{enb}(\gamma, \sigma_{n-1,0}) \wedge \mathbf{enb}(\gamma, \sigma_{n,0}) \wedge \neg\mathbf{enb}(\gamma, \sigma_{0,m-1}) \wedge \mathbf{enb}(\gamma, \sigma_{0,m}), \qquad (6.6)$$

then there exists $\hat{\jmath}$ and τ such that $0 \leq \hat{\jmath} < m$, $\sigma_{0,0} \xrightarrow{*} \tau$, and α_{n-1} and $\beta_{\hat{\jmath}}$ are disjunctive triggers of γ at τ.

Proof: Applying Lemma 5.5 to the two paths above yields the existence of $\{i, j : 1 \leq i \leq n \wedge 1 \leq j \leq m : \sigma_{i,j}\}$ such that (5.8) holds. By stability, $\mathbf{enb}(\gamma, \sigma_{i,j})$ if $i = n$ or $j = m$. Since $\neg\mathbf{enb}(\gamma, \sigma_{n-1,0})$ and $\mathbf{enb}(\gamma, \sigma_{n-1,m})$, let $\hat{\jmath}$ be the largest index such that $\neg\mathbf{enb}(\gamma, \sigma_{n-1,\hat{\jmath}})$. Setting τ to $\sigma_{n-1,\hat{\jmath}}$ establishes the lemma. *Q.E.D.*

Lemma 6.11 *Let ρ be the c.c.a. of σ_a and σ_b with $\rho \xrightarrow{\beta} \phi \xrightarrow[*]{\widehat{\mathcal{B}}} \sigma_b$ and $\rho \xrightarrow{\mathcal{A}} \sigma_a$. Then, for any γ, $\gamma \in (\mathcal{A} \cap \widehat{\mathcal{B}}) \wedge \mathbf{enb}(\gamma, \phi)$ implies there exists an event $\alpha \in \mathcal{A}$ and a state τ such that $\rho \xrightarrow{*} \tau \xrightarrow{*} \sigma_a$, and β and α are disjunctive triggers of γ at τ.*

Proof: By Lemma 5.10, $\beta \notin \mathcal{A}$ and $\gamma \in \mathcal{A} \cap \mathcal{B}$ implies $\neg\mathbf{enb}(\gamma, \rho)$. But $\gamma \in \mathcal{A}$; so, let $\tilde{\sigma}$ be the first intermediate state in the path $\rho \xrightarrow{*} \sigma_a$ such that $\mathbf{enb}(\gamma, \tilde{\sigma})$ and let

$$(\rho = \tau_0) \xrightarrow{\alpha_0} \tau_1 \xrightarrow{\alpha_1} \cdots \xrightarrow{\alpha_{n-2}} \tau_{n-1} \xrightarrow{\alpha_{n-1}} (\tau_n = \tilde{\sigma}).$$

Then, it can be verified that β and α_{n-1} are disjunctive triggers of γ at τ_{n-1}. *Q.E.D.*

6.3.2 Terminating Events

Definition: Event $\langle x_k, l\rangle$ is a *terminating event* if

$$\exists\sigma :: \mathbf{enb}(\langle x_k, l\rangle, \sigma) \wedge \forall\tau :: \neg\mathbf{enb}(\langle x_k, l+1\rangle, \tau).$$

A variable x_k *has a terminating event* if there exists l such that $\langle x_k, l\rangle$ is a terminating event.

Lemma 6.12 *Suppose there exists a cycle with period* π. *If* $\mathbf{var}(\delta) \in \mathbf{span}(\pi)$, *then there exists a minimal period* $\widehat{\pi}$ *such that* $\mathbf{var}(\delta) \in \mathbf{span}(\widehat{\pi})$.

Proof: Use induction on the length of the cycle. There is no period with zero length. For any cycle with non-zero length, if it is minimal then set $\widehat{\pi}$ to π. Else, by Lemma 5.24, there exist two sub-cycles with periods $\widehat{\pi}$ and $\pi - \widehat{\pi}$. If $\mathbf{var}(\delta) \in \mathbf{span}(\pi)$ then $\mathbf{var}(\delta) \in \mathbf{span}(\widehat{\pi})$ or $\mathbf{var}(\delta) \in \mathbf{span}(\pi - \widehat{\pi})$. So, applying the inductive hypothesis on one of these sub-cycles establishes the lemma. *Q.E.D.*

Lemma 6.13 *If there exist event* $\langle x_k, l\rangle$ *and state* σ *such that* $\mathbf{enb}(\langle x_k, l\rangle, \sigma)$ *and* x_k *has no terminating event, then then there exists a minimal period* $\mathbf{span}(\widehat{\pi})$ *such that* $x_k \in \mathbf{span}(\widehat{\pi})$.

Proof: Pick q large enough so that $q - \sigma_{\text{init}}[k] > 2^K$, where K is the number of variables. Since x_k has no terminating event, there exists τ such that $\mathbf{enb}(\langle x_k, q\rangle, \tau)$. W.l.g., assume τ is a state with the minimum weight such that $\mathbf{enb}(\langle x_k, q\rangle, \tau)$ is satisfied. Now, since the range of the $\mathbf{bool}()$ function has at most 2^K elements, by the Pigeonhole Principle, there exist intermediate states ϕ and $(\phi + \pi)$ in the path from σ_{init} to τ such that $\phi \xrightarrow{*} (\phi + \pi)$ is a cycle. By Lemma 5.16, $\mathbf{enb}(\langle x_k, q\rangle \ominus \pi, \tau - \pi)$. So, by the choice of τ, $\pi[k] \neq 0$. The existence of $\widehat{\pi}$ then follows from Lemma 6.12. *Q.E.D.*

6.3.3 Criteria for Uniform Graphs

Definition: The event δ is called a *mode-switching* event if there exists a non-uniform cycle $\sigma \xrightarrow{*} (\sigma + \pi)$ that has a uniform sub-cycle $\widetilde{\sigma} \xrightarrow{*} (\widetilde{\sigma} + \widetilde{\pi})$ and $\sigma \xrightarrow{\delta} \widetilde{\sigma}$.

Lemma 6.14 *If a state graph is non-uniform, then there exists a mode-switching event.*

Proof: Of all the non-uniform cycles, let

$$\sigma \overset{\mathcal{A}}{\twoheadrightarrow} (\sigma + \pi) \tag{6.7}$$

be one with the minimal length. By definition, it is not minimal; of all of its normal sub-cycles, let

$$\tilde{\sigma} \overset{\tilde{\mathcal{A}}}{\twoheadrightarrow} (\tilde{\sigma} + \tilde{\pi}) \tag{6.8}$$

be the one such that $\sigma \overset{\mathcal{D}}{\twoheadrightarrow} \tilde{\sigma}$ has the shortest length. Since $\tilde{\mathcal{A}} \subset \mathcal{A}$, (6.8) is uniform since its length is less than that of (6.7). Now, $\mathcal{D}$ is not empty or else (6.7) is uniform; so, let

$$\sigma \overset{\hat{\mathcal{D}}}{\twoheadrightarrow} \tau \xrightarrow{\delta} \tilde{\sigma}. \tag{6.9}$$

By Lemma 5.19 and Theorem 5.1, $\tau \twoheadrightarrow (\tau + \pi)$ exists and is not minimal. If it is uniform, then it has a sub-cycle $\tilde{\tau} \twoheadrightarrow (\tilde{\tau} + \tilde{\pi})$ which would then be a normal sub-cycle of (6.7) and thereby contradict the definition of (6.8). Thus, $\tau \twoheadrightarrow (\tau + \pi)$ is non-uniform and has (6.8) as a sub-cycle. Therefore, δ is a mode-switching event. *Q.E.D.*

Lemma 6.15 *Suppose* $\sigma \overset{\mathcal{A}}{\twoheadrightarrow} (\sigma + \pi)$ *is a cycle and* $\tilde{\sigma} \overset{\tilde{\mathcal{A}}}{\twoheadrightarrow} (\tilde{\sigma} + \hat{\pi})$ *is a path such that* $\sigma \xrightarrow{\delta} \tilde{\sigma}$, $\delta \notin \mathcal{A}$, *and* $\tilde{\mathcal{A}} \subseteq \mathcal{A}$. *Let* ρ *be the c.c.a. of* $(\sigma + \pi)$ *and* $(\tilde{\sigma} + \hat{\pi})$. *Then, there exist* ϕ *and* $\mathcal{C} \subseteq \tilde{\mathcal{A}}$ *such that*

$$\rho \xrightarrow{\delta} \phi \overset{\mathcal{C}}{\twoheadrightarrow} (\tilde{\sigma} + \hat{\pi}), \tag{6.10}$$

and, for any $\gamma \in \mathcal{C}$ *such that* $\mathbf{enb}(\gamma, \phi)$,

$$\begin{aligned} &\exists \beta, \tau : \beta \in \mathcal{A} \wedge \sigma \twoheadrightarrow \tau \twoheadrightarrow (\sigma + \pi) : \\ &\qquad \delta \text{ and } \beta \text{ are disjunctive triggers of } \gamma \text{ at } \tau. \end{aligned} \tag{6.11}$$

Furthermore, either δ *is a terminating event or, for all* $\gamma \in \mathcal{C}$ *such that* $\mathbf{enb}(\gamma, \phi)$, *if the guard of* $\mathbf{tran}(\gamma)$ *is* $B_0 \vee B_1 \vee \ldots \vee B_m$, *then there are at least two* B_i*'s that do not contain the literal* $\mathbf{lit}(\delta)$ *and at least one* B_i *that contains the literal* $\mathbf{lit}(\delta)$ *but is not a stable disjunct.*

Proof: Since σ is a common ancestor of $(\sigma + \pi)$ and $(\tilde{\sigma} + \tilde{\pi})$ and $\delta \notin \mathcal{A}$, by Corollary 5.13, δ occurs in $\rho \xrightarrow{*} (\tilde{\sigma} + \tilde{\pi})$ and so, by stability, (6.10) holds. Moreover, every event γ in $\mathcal{C}$ occurs in $\sigma \xrightarrow{*} \rho \xrightarrow{*} (\tilde{\sigma} + \tilde{\pi})$; so, $\mathcal{C} \subseteq \tilde{\mathcal{A}}$. Let γ be any event such that $\mathbf{enb}(\gamma, \phi) \wedge (\gamma \in \mathcal{C} \subseteq \mathcal{A})$. Then, by Lemma 6.11, (6.11) is valid.

For the second half of the lemma, let β and τ be the witnesses to (6.11). Write the cycle $\sigma \xrightarrow{*} (\sigma + \pi)$ as

$$\sigma_0 \xrightarrow{\alpha_0} \sigma_1 \xrightarrow{\alpha_1} \cdots \xrightarrow{\alpha_{r-1}} \sigma_r \xrightarrow{\alpha_r} \sigma_{r+1} \xrightarrow{\alpha_{r+1}} \cdots \xrightarrow{\alpha_{n-1}} \sigma_n \tag{6.12}$$

with $\sigma_r = \tau$ and $\alpha_r = \beta$. Since δ is a disjunctive trigger of γ at σ_r, there exists $\tilde{\sigma}_r$ such that $\sigma_r \xrightarrow{\delta} \tilde{\sigma}_r$ and $\mathbf{enb}(\gamma, \tilde{\sigma}_r)$.

Next, suppose δ is not a terminating event. Let $\delta = \langle x_k, l \rangle$ and $\tilde{\delta} = \langle x_k, l+1 \rangle$. Then, there exists $\hat{\tau}$ such that $\mathbf{enb}(\tilde{\delta}, \hat{\tau})$. Let $\tilde{\tau}$ be the c.c.d. of σ_r and $\hat{\tau}$; then, $\mathbf{enb}(\tilde{\delta}, \tilde{\tau})$. By Lemma 5.19, $\sigma_r \xrightarrow{*} (\sigma_r + \pi)$. So, by Lemma 5.21, there exist $q > 0$ and a state $\tilde{\phi}$ such that

$$\sigma_r \overset{\mathcal{D}}{\xrightarrow{*}} \tilde{\phi} \xrightarrow{*} \tilde{\tau} \xrightarrow{*} (\tilde{\phi} + q\pi),$$

and $\mathbf{span}(\mathcal{D}) \cap \mathbf{span}(\pi) = \emptyset$. Since $\delta \notin \mathcal{A}$, $\pi[k] = 0$ and so, by stability and Lemma 5.16, $\mathbf{enb}(\tilde{\delta}, \tilde{\tau}) \Rightarrow \mathbf{enb}(\tilde{\delta}, \tilde{\phi} + q\pi) \Rightarrow \mathbf{enb}(\tilde{\delta}, \tilde{\phi})$. Letting $\phi_a = \tilde{\phi}$ implies there exists ϕ_b such that

$$\sigma_r \xrightarrow{\delta} \tilde{\sigma}_r \overset{\mathcal{D} \setminus \{\delta\}}{\xrightarrow{*}} \phi_a \xrightarrow{\tilde{\delta}} \phi_b. \tag{6.13}$$

Since $\mathbf{enb}(\gamma, \tilde{\sigma}_r)$, let B_0 be a disjunct that is **true** in state $\tilde{\sigma}_r$. By $\neg\mathbf{enb}(\gamma, \sigma_r)$, B_0 contains $\mathbf{lit}(\delta)$. Note, however, that the value of $\mathbf{lit}(\delta)$ is **false** at ϕ_b and therefore B_0 is not a stable disjunct. Also, by Lemma 6.9, let B_1 be the disjunct that contains the literal $\mathbf{lit}(\alpha_r)$ but not $\mathbf{lit}(\delta)$. The value of $\mathbf{lit}(\alpha_r)$ is **false** in σ_r since $\mathbf{enb}(\alpha_r, \sigma_r)$; therefore, it remains **false** at state ϕ_b since α_r does not occur in (6.13). Also, at state ϕ_b, the value of $\mathbf{lit}(\delta)$ is **false.** Thus, the value of any disjunct containing the literal $\mathbf{lit}(\delta)$ or $\mathbf{lit}(\alpha_r)$ is **false**. So, stability requires that there exists yet another disjunct, B_2, that contains neither literals. *Q.E.D.*

Theorem 6.1 *If a graph is not uniform, then there exists a mode-switching event δ such that δ is a disjunctive trigger for two events α and β with*

$\mathbf{var}(\alpha) \neq \mathbf{var}(\beta)$. *Furthermore, either δ is a terminating event or, if the guard for* $\mathbf{tran}(\alpha)$ *or* $\mathbf{tran}(\beta)$ *is $B_0 \vee B_1 \vee \ldots \vee B_m$, then there are at least two B_i's that do not contain* $\mathbf{lit}(\delta)$ *and at least one B_i that contains* $\mathbf{lit}(\delta)$ *but is not a stable disjunct.*

Proof: By Lemma 6.14, there exists a mode-switching event δ. So, let

$$\sigma \overset{\mathcal{A}}{\twoheadrightarrow} (\sigma + \pi) \tag{6.14}$$

be a non-uniform cycle with uniform sub-cycle $\widetilde{\sigma} \twoheadrightarrow (\widetilde{\sigma} + \widetilde{\pi})$ and $\sigma \xrightarrow{\delta} \widetilde{\sigma}$. By Lemma 5.24, there also exists cycle $(\widetilde{\sigma} + \widetilde{\pi}) \twoheadrightarrow (\widetilde{\sigma} + \pi)$. Since the lengths of these last two cycles are less than (6.14), they are uniform and, by Lemma 6.1, can be written a concatenation of minimal cycles. Thus, there exist an integer $p > 1$ and a set of minimal cycles

$$\{i : 0 \leq i < p : \widehat{\tau}_i \twoheadrightarrow \widehat{\tau}_{i+1}\}$$

such that $\widehat{\tau}_0 = \widetilde{\sigma}$ and $\widehat{\tau}_m = (\widetilde{\sigma} + \pi)$. For all i, $0 \leq i < p$, let $\pi_i = \widehat{\tau}_{i+1} - \widehat{\tau}_i$ and observe that, by Lemma 5.2, $\pi_i < \pi$.

Next, let $\hat{\imath}$ be an arbitrary integer with $0 \leq \hat{\imath} < p$. Since $\mathbf{bool}(\widehat{\tau}_{\hat{\imath}}) = \mathbf{bool}(\widetilde{\sigma})$, by Lemma 5.16, $\widetilde{\sigma} \twoheadrightarrow (\widetilde{\sigma} + \pi_{\hat{\imath}})$. By Lemma 5.20, this is a sub-cycle of (6.14). Let $\widetilde{\mathcal{A}}$ be the set of events occurring in that sub-cycle and so $\widetilde{\mathcal{A}} \subset \mathcal{A}$. Let ρ be the c.c.a. of $(\sigma + \pi)$ and $(\widetilde{\sigma} + \pi_{\hat{\imath}})$ and let $\widehat{\pi} = \pi_{\hat{\imath}}$. Then, applying Lemma 6.15 yields the existence of ϕ and $\mathcal{C} \subseteq \widetilde{\mathcal{A}}$ such that (6.10) is satisfied and, for any $\gamma \in \mathcal{C}$ such that $\mathbf{enb}(\gamma, \phi)$, (6.11) holds.

If $\mathcal{C}$ is empty, then by (6.10) and $\sigma \xrightarrow{\delta} \widetilde{\sigma}$, $(\widetilde{\sigma} + \pi_{\hat{\imath}}) - \rho = \widetilde{\sigma} - \sigma$. So, $\rho = \sigma + \pi_{\hat{\imath}}$ and $\sigma \twoheadrightarrow \rho$ is a sub-cycle of (6.14), contradicting the fact that it is non-uniform. Thus, there exists $\gamma \in \mathcal{C}$ such that δ is a disjunctive trigger for γ by (6.11). Moreover, $\mathcal{C} \subseteq \widetilde{\mathcal{A}}$ implies $\mathbf{var}(\gamma)$ is in $\mathbf{span}(\pi_{\hat{\imath}})$.

Now, if there exist $\hat{\imath}$ and $\hat{\jmath}$ such that $\pi_{\hat{\imath}} \neq \pi_{\hat{\jmath}}$, then δ is a disjunctive trigger for two events whose variables are in the spanning sets of different minimal periods. By Theorem 5.2, these variables are different and the first part of the theorem is established in this case.

Alternatively, if for all $\hat{\imath}$ and $\hat{\jmath}$, $\pi_{\hat{\imath}} = \pi_{\hat{\jmath}}$, then $\pi = p\pi_0$. Let $\widehat{\pi} = \pi_0$ and let $\widetilde{\mathcal{A}}$ be the set of events occurring in $\widetilde{\sigma} \twoheadrightarrow (\widetilde{\sigma} + \widehat{\pi})$. If ρ is the c.c.a. of $(\sigma + \pi)$ and $(\widetilde{\sigma} + \widehat{\pi})$, then, by Lemma 6.15, there exist ϕ, and $\mathcal{C} \subseteq \widetilde{\mathcal{A}}$ such that (6.10) is satisfied and for all $\gamma \in \mathcal{C}$ with $\mathbf{enb}(\gamma, \phi)$, (6.11) holds.

Now, $\sigma \xrightarrow{\delta} \widetilde{\sigma}$, $\sigma \overset{*}{\longrightarrow} \rho$, and (6.10) imply

$$\widetilde{\sigma} \overset{*}{\longrightarrow} \phi \overset{\mathcal{C}}{\overset{*}{\longrightarrow}} (\widetilde{\sigma} + \widehat{\pi}).$$

So, by Lemma 5.17 and Lemma 5.16,

$$\widetilde{\sigma} \overset{*}{\longrightarrow} (\widetilde{\sigma} + (p-1)\widehat{\pi}) \overset{*}{\longrightarrow} (\phi + (p-1)\widehat{\pi}) \overset{\mathcal{C} \oplus (p-1)\widehat{\pi}}{\overset{*}{\longrightarrow}} (\widetilde{\sigma} + p\widehat{\pi}). \tag{6.15}$$

Since $(\sigma + \pi) \xrightarrow{\delta} (\widetilde{\sigma} + p\widehat{\pi})$, the set of events occurring in (6.15) is exactly $\mathcal{A}$. So, if $\widetilde{\rho}$ is the c.c.a. of $(\sigma + \pi)$ and $(\phi + (p-1)\widehat{\pi})$, then, by Lemma 6.15, there exist $\widetilde{\phi}$ and $\widetilde{\mathcal{C}} \subseteq \mathcal{A}$ such that

$$\widetilde{\rho} \xrightarrow{\delta} \widetilde{\phi} \overset{\widetilde{\mathcal{C}}}{\overset{*}{\longrightarrow}} (\phi + (p-1)\widehat{\pi}) \tag{6.16}$$

and for all $\widetilde{\gamma} \in \widetilde{\mathcal{C}}$ such that $\mathbf{enb}(\widetilde{\gamma}, \widetilde{\phi})$,

$$\begin{gathered} \exists \beta, \tau : (\beta \in \mathcal{A}) \wedge (\sigma \overset{*}{\longrightarrow} \tau \overset{*}{\longrightarrow} (\sigma + \pi)) : \\ \delta \text{ and } \beta \text{ are disjunctive triggers of } \widetilde{\gamma} \text{ at } \tau. \end{gathered} \tag{6.17}$$

Now, if $\widetilde{\mathcal{C}}$ is empty, then by (6.16) and (6.10), $(\phi + (p-1)\widehat{\pi}) - \widetilde{\rho} = \phi - \rho$ and, therefore, $\widetilde{\rho} = \rho + (p-1)\widehat{\pi}$. This equality and $\widetilde{\rho} \overset{*}{\longrightarrow} (\sigma + \pi)$ implies

$$\sigma \overset{*}{\longrightarrow} \rho \overset{*}{\longrightarrow} (\sigma + \pi - (p-1)\widehat{\pi})$$

is a sub-cycle of (6.14), contradicting the fact that it is non-uniform. Therefore, $\widetilde{\mathcal{C}}$ is not empty and contains $\widetilde{\gamma}$ such that δ is a disjunctive trigger of $\widetilde{\gamma}$.

Now, $\widetilde{\mathcal{C}}$ is not empty implies $\widetilde{\rho} \neq (\sigma + \pi)$ and there exists $\widetilde{\beta}$ such that $\mathbf{enb}(\widetilde{\beta}, \widetilde{\rho})$ and $\widetilde{\beta}$ occurs in $\widetilde{\rho} \overset{*}{\longrightarrow} (\sigma + \pi)$. By Lemma 5.10, $\widetilde{\beta} \notin \widetilde{\mathcal{C}}$; so, by stability, $\mathbf{enb}(\widetilde{\beta}, \phi + (p-1)\widehat{\pi})$. Also, since the set of events occurring in (6.15) is $\mathcal{A}$, $\widetilde{\beta} \in (\mathcal{C} \oplus (p-1)\widehat{\pi})$. Consequently, there exists $\widehat{\gamma} = (\widetilde{\beta} \ominus (p-1)\widehat{\pi})$ such that $\widehat{\gamma} \in \mathcal{C}$ and $\mathbf{enb}(\widehat{\gamma}, \phi)$. Thus, by (6.11), δ is a disjunctive trigger of $\widehat{\gamma}$. Now, if $\mathbf{var}(\widetilde{\beta}) = \mathbf{var}(\widetilde{\gamma})$, then $\widetilde{\beta} = \widetilde{\gamma}$ since they are both enabled at $\widetilde{\phi}$. However, $\mathbf{enb}(\widetilde{\beta}, \widetilde{\rho})$ but $\neg\mathbf{enb}(\widetilde{\gamma}, \widetilde{\rho})$ due to $\widetilde{\gamma} \in \widetilde{\mathcal{C}} \subseteq \mathcal{A}$ and Lemma 5.10. Thus, a contradiction can be avoided only if $\mathbf{var}(\widehat{\gamma})$, which is the same as $\mathbf{var}(\widetilde{\beta})$, is different from $\mathbf{var}(\widetilde{\gamma})$. Therefore, the first part of the theorem is established.

The second part of the Theorem follows directly from the second part of Lemma 6.15. *Q.E.D.*

Corollary 6.16 *A state graph that has no disjunctively enabled events is uniform.*

Proof: Obvious from Theorem 6.1. *Q.E.D.*

Corollary 6.17 *A state graph where all variables have no terminating events and all PR's have stable disjuncts is uniform.*

Proof: Obvious from Theorem 6.1. *Q.E.D.*

Corollary 6.18 *All mode-switching events in a non-separable state graph are terminating.*

Proof: Suppose the graph is not uniform. Then, let σ, π, $\mathcal{A}$, $\widetilde{\sigma}$, $\widetilde{\pi}$, and δ be defined as in Theorem 6.1. By the remarks following the definition of a minimal cycle, (6.14) contains a sub-cycle with minimal period $\widehat{\pi}_a$. Since $\delta \notin \mathcal{A}$, $\mathbf{var}(\delta) \notin \mathbf{span}(\pi)$ which implies $\mathbf{var}(\delta) \notin \mathbf{span}(\widehat{\pi}_a)$.

Next, suppose $\mathbf{var}(\delta)$ has no terminating event. Then, by Lemma 6.13, there exists a minimal period $\widehat{\pi}_b$ such that $\mathbf{var}(\delta) \in \mathbf{span}(\widehat{\pi}_b)$. Thus, the graph contains at least two minimal cycles with different periods and is therefore separable. *Q.E.D.*

Theorem 6.1 and its corollaries give some of the conditions that would guarantee a uniform graph. As can be seen, a non-uniform graph can only result from the presence of a mode-switching event that is a disjunctive trigger for two transitions with distinct variables. By Lemma 6.9, this condition can be checked syntactically. Most of the practical PR sets are non-separable and most (if not all) of their disjuncts are stable. Furthermore, the user is usually aware of which variables have terminating events and can then check specifically whether it is a potential mode-switching event. Lastly, the additional conditions that a non-terminating mode-switching event has to satisfy are quite restrictive, making it uncommon and fairly easy to spot.

6.4 Index-Priority Simulation

The algorithm for index-priority simulation is given in Section A.1. Algorithm 1, with its associated procedure *find_cycle*(), takes a PR set with a stable state graph and returns a non-transitory state S[n] and a list of p cycles, in array C[ ], that satisfies the conditions described below. In the remainder of this chapter, for any i, $0 \leq i < \mathrm{p}$, let $\mathtt{C}[i] = \sigma_i \xrightarrow{*} (\sigma_i + \pi_i)$. Then, the following two conditions hold for the list of cycles returned by the algorithm:

$$\forall \widehat{\pi} : \widehat{\pi} \text{ is a period} : \mathbf{span}(\widehat{\pi}) \subseteq \bigcup_{i=0}^{\mathrm{p}-1} \mathbf{span}(\pi_i), \tag{6.18}$$

and

$$\forall i, j : 0 \leq i < j < \mathrm{p} : (\exists k :: x_k \notin \mathbf{span}(\pi_i) \wedge x_k \in \mathbf{span}(\pi_j)). \tag{6.19}$$

In addition, if the associated graph is uniform, then all of the periods are minimal and different from each other.

Below is an illustration of how the algorithm works. The proof of its correctness will follow.

Example 6.6: Below is a simplified version of the PR set for the zero-checker *zeroB* described in Sub-section 2.5.5 — the intermediate variables a and b have been removed:

$$\begin{array}{lcl}
(aT_i \vee aF_i) \wedge (bT_i \vee bF_i) & \rightarrow & g\uparrow \\
g \wedge (aT_i \vee bT_i) & \rightarrow & cT_o\uparrow \\
g \wedge (aF_i \wedge bF_i) & \rightarrow & cF_o\uparrow \\
(\neg aT_i \wedge \neg aF_i) \wedge (\neg bT_i \wedge \neg bF_i) & \rightarrow & g\downarrow \\
\neg g & \rightarrow & cT_o\downarrow \\
\neg g & \rightarrow & cF_o\downarrow .
\end{array}$$

Suppose we want to analyze its performance with respect to the following environment:

$$\begin{array}{ll}
 & *[aT_i\uparrow;\ [cT_o \vee cF_o];\ aT_i\downarrow;\ [\neg cT_o \wedge \neg cF_o]; \\
 & \quad aF_i\uparrow;\ [cT_o \vee cF_o];\ aF_i\downarrow;\ [\neg cT_o \wedge \neg cF_o]] \\
\| & *[bT_i\uparrow;\ [cT_o \vee cF_o];\ bT_i\downarrow;\ [\neg cT_o \wedge \neg cF_o]; \\
 & \quad bF_i\uparrow;\ [cT_o \vee cF_o];\ bF_i\downarrow;\ [\neg cT_o \wedge \neg cF_o]].
\end{array}$$

First, the handshaking expansions above need to be converted into PR sets. Since we are not interested in the precise implementation of the environment, this conversion need not be performed optimally. In fact, by using a "program counter," there exists a syntactic translator from arbitrary handshaking expansions to PR sets. In this example, the state variables $d0$ and $d1$ are introduced in the first handshaking expansion which is then compiled into

$$\begin{array}{lcl@{\qquad}lcl}
\neg d1 \wedge \neg d0 & \rightarrow & aT_i\uparrow & \neg d1 \wedge (cT_o \vee cF_o) & \rightarrow & d0\uparrow \\
d0 \wedge \neg d1 & \rightarrow & aT_i\downarrow & d0 \wedge \neg cT_o \wedge \neg cF_o & \rightarrow & d1\uparrow \\
d1 \wedge d0 & \rightarrow & aF_i\uparrow & d1 \wedge (cT_o \vee cF_o) & \rightarrow & d0\downarrow \\
d1 \wedge \neg d0 & \rightarrow & aF_i\downarrow & \neg d0 \wedge \neg cT_o \wedge \neg cF_o & \rightarrow & d1\downarrow
\end{array}$$

Similarly, the second handshaking expansion is implemented in a symmetric fashion using state variables $e0$ and $e1$.

So, let $\mathcal{P}$ be the union of the PR set for the zero-checker and those for its environment; its state graph is shown in Figure 6.6. Next, suppose the variables are indexed in reverse-alphabetical order, i.e, $x_0 = g$, $x_1 = e1$, ..., $x_{10} = aF_i$. Then, at every state when there are more than one events enabled, the event correspond to the alphabetically first variable will be selected in *find_cycle*(). Hence, when first called from Algorithm 1, *find_cycle*() traces out the path shown in bold in Figure 6.6. Since the two states marked with crosses are the first pair to have the same Boolean value, the following cycle, with period 42222222222, will be returned:

$$\begin{array}{l}
\sigma_0 \xrightarrow{\langle bT_i,1\rangle} \sigma_1 \xrightarrow{\langle g,1\rangle} \sigma_2 \xrightarrow{\langle cT_o,1\rangle} \sigma_3 \xrightarrow{\langle d0,1\rangle} \sigma_4 \xrightarrow{\langle aT_i,2\rangle} \\
\quad \sigma_5 \xrightarrow{\langle e0,1\rangle} \sigma_6 \xrightarrow{\langle bT_i,2\rangle} \sigma_7 \xrightarrow{\langle g,2\rangle} \sigma_8 \xrightarrow{\langle cT_o,2\rangle} \sigma_9 \xrightarrow{\langle d1,1\rangle} \\
\quad \sigma_{10} \xrightarrow{\langle aF_i,1\rangle} \sigma_{11} \xrightarrow{\langle e1,1\rangle} \sigma_{12} \xrightarrow{\langle bF_i,1\rangle} \sigma_{13} \xrightarrow{\langle g,3\rangle} \sigma_{14} \xrightarrow{\langle cF_o,1\rangle} \\
\quad \sigma_{15} \xrightarrow{\langle d0,2\rangle} \sigma_{16} \xrightarrow{\langle aF_i,2\rangle} \sigma_{17} \xrightarrow{\langle e0,2\rangle} \sigma_{18} \xrightarrow{\langle bF_i,2\rangle} \sigma_{19} \xrightarrow{\langle g,4\rangle} \\
\quad \sigma_{20} \xrightarrow{\langle cF_o,2\rangle} \sigma_{21} \xrightarrow{\langle d1,2\rangle} \sigma_{22} \xrightarrow{\langle aT_i,3\rangle} \sigma_{23} \xrightarrow{\langle e1,2\rangle} \sigma_{24}.
\end{array} \tag{6.20}$$

Since all variables have events appearing in the cycle, $\mathtt{V}$ will become the set of all variables when *find_cycle*() returns for the first time. Hence, the second call of *find_cycle*() results in *empty_cycle* being returned and the algorithm terminates with (6.20) as the only cycle found and $\sigma_0 = 00000000010$ as a non-transitory state.

Now, since cTo and cFo are mutually exclusive due to dual-rail encoding, the disjuncts in the guards for $d0\uparrow$, $d0\downarrow$, $e0\uparrow$, and $e0\downarrow$ are stable. Also, if either of the disjuncts in the guard of $cT_o\uparrow$ is **true**, then it remains **true** until

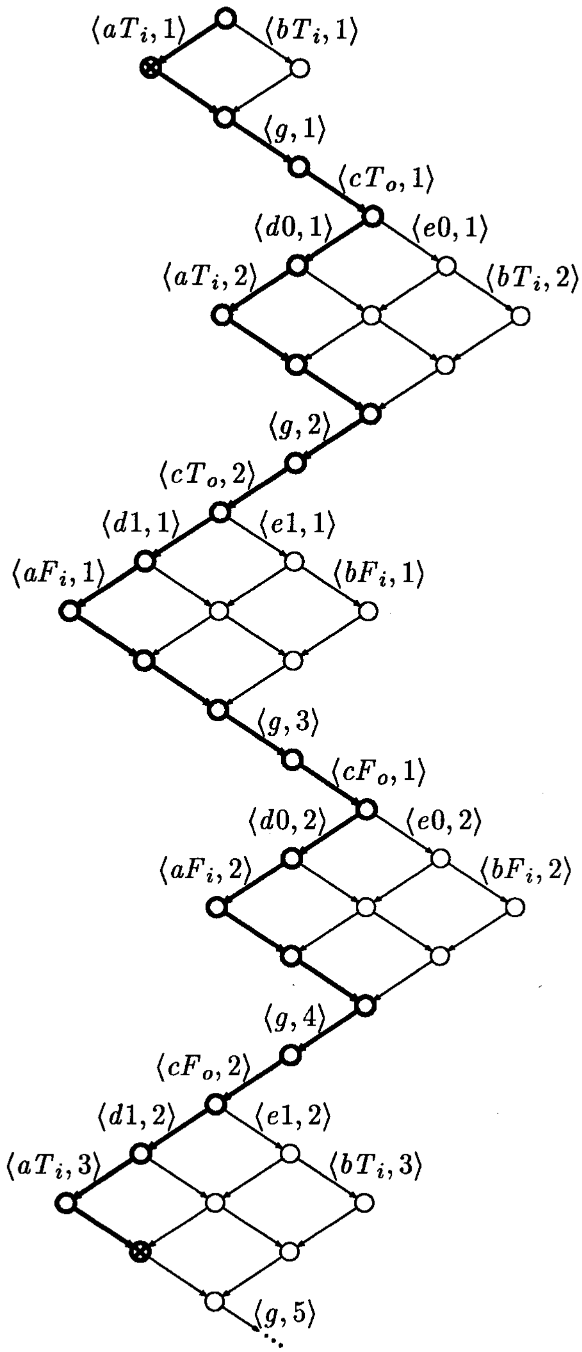

Figure 6.6: Cumulative state graph for a zero-checker cell

$cT_o\uparrow$ fires because of the handshake protocol in the environment. Finally, none of the literals in the guard of $g\uparrow$ can change from **true** to **false** without $g\uparrow$ occurring first; therefore, the disjuncts of $g\uparrow$ are stable. Note that it is possible to conclude that the disjuncts of a guard are stable without writing it in DNF.

Since all disjuncts are stable and there is no terminating event (each variable appears in a cycle), the state graph is uniform. Therefore, (6.20) is a minimal cycle and $\pi = 42222222222$ is the only minimal period. □

We will first establish the correctness of Algorithm 1 for the general case where the state graph may not be uniform. The function *next_state*(s, a) assumes the event a is enabled at state s and returns a new state σ such that

$$\mathtt{s} \xrightarrow{\mathtt{a}} \sigma. \tag{6.21}$$

Also, *empty_cycle* is a special return value which signifies that no cycle has been found.

Lemma 6.19 *The following two predicates are loop invariants of the* repeat-*loop in Algorithm 1:*

$$\mathtt{S[0]} \xrightarrow{\mathtt{A[0]}} \mathtt{S[1]} \xrightarrow{\mathtt{A[1]}} \cdots \xrightarrow{\mathtt{A[n-1]}} \mathtt{S[n]}, \tag{6.22}$$

and

$$\forall i, j : 0 \le i, j \le \mathrm{n} : \mathbf{bool}(\mathtt{S}[i]) = \mathbf{bool}(\mathtt{S}[j]) \Rightarrow i = j. \tag{6.23}$$

Proof: First, consider the procedure *find_cycle*(). By its topology,

$$\mathtt{E} = \{\alpha : \mathbf{enb}(\alpha, \mathtt{S[n]}) \wedge \mathbf{var}(\alpha) \notin \mathtt{V} : \alpha\} \tag{6.24}$$

is an invariant of the while-loop. Next, suppose that (6.22) and (6.23) hold when *find_cycle*() is called. Then, by (6.24), **enb**(A[n], S[n]). So, by (6.21),

$$\mathtt{S[0]} \xrightarrow{\mathtt{A[0]}} \mathtt{S[1]} \xrightarrow{\mathtt{A[1]}} \cdots \xrightarrow{\mathtt{A[n-1]}} \mathtt{S[n]} \xrightarrow{\mathtt{A[n]}} \mathtt{s} \tag{6.25}$$

holds just before the if-statement. If the condition in the if-statement is **false**, then $\forall \tilde{\imath} : 0 \le \tilde{\imath} \le \mathrm{n} : \mathbf{bool}(\mathtt{S}[\tilde{\imath}]) \ne \mathbf{bool}(\mathtt{s})$. This predicate, (6.25), and the assignments to n and S[n] imply (6.23) and (6.22) are invariants of the while-loop.

Alternatively, if the condition in the if-statement is **true**, then the procedure returns after setting n to a value that is less than its value before the if-statement. This assignment cannot invalidate (6.22) and (6.23). So, in either case, if (6.22) and (6.23) hold when *find_cycle*() is called, then they hold when the procedure returns. This conclusion establishes the lemma since (6.22) and (6.23) hold when the repeat-loop of Algorithm 1 is first entered. *Q.E.D.*

Lemma 6.20 *If find_cycle() does not return* empty_cycle, *then it returns a cycle whose period has a spanning set containing a variable not in* V.

Proof: If *find_cycle*() does not return *empty_cycle*, then it returns

$$\mathtt{S[i]} \xrightarrow{\mathtt{A[i]}} \mathtt{S[i+1]} \xrightarrow{\mathtt{A[i+1]}} \cdots \xrightarrow{\mathtt{A[m]}} \mathtt{s}. \tag{6.26}$$

By (6.25) and $\mathbf{bool}(\mathtt{S[i]}) = \mathbf{bool}(\mathtt{s})$, (6.26) is a cycle. Finally, by construction, $\mathbf{var}(\mathtt{A[m]}) \notin \mathtt{V}$; thus, the lemma is established. *Q.E.D.*

Lemma 6.21 *The following two predicates and (6.19) are invariants of the* repeat*-loop in Algorithm 1:*

$$\mathtt{V} = \bigcup_{i=0}^{\mathtt{p}-1} \mathbf{span}(\pi_i), \tag{6.27}$$

and

$$\forall i : 0 \le i < \mathtt{p} : \mathtt{S[n]} \xrightarrow{*} (\mathtt{S[n]} + \pi_i). \tag{6.28}$$

Proof: From the assignments to U and V, (6.27) is a loop invariant. Next, consider the situation immediately after the the assignment to C[p]. By Lemma 6.20, the spanning set of $\pi_{\mathtt{p}}$ contains a variable x_k not in V; so, by (6.27), x_k is not in the spanning set of any π_i with $0 \le i < \mathtt{p}$. Thus, (6.19) is established after the increment of p.

Finally, assume that (6.28) holds at the beginning of the repeat-loop and let $\widehat{n}$ be the value of n at that point so that

$$\forall i : 0 \le i < \mathtt{p} : \mathtt{S}[\widehat{n}] \xrightarrow{*} (\mathtt{S}[\widehat{n}] + \pi_i). \tag{6.29}$$

Consider the situation after *find_cycle*() returns. If c is *empty_cycle*, then $\mathtt{S}[\widehat{n}] \xrightarrow{\ast} \mathtt{S}[\mathtt{n}]$. By Lemma 5.19, (6.29) $\Rightarrow$ (6.28). Alternatively, if c is not *empty_cycle*, then there exists period $\widetilde{\pi}$ such that c is

$$\mathtt{S}[\mathtt{n}] \xrightarrow{\ast} (\mathtt{S}[\mathtt{n}] + \widetilde{\pi}). \tag{6.30}$$

If $\mathtt{n} \geq \widehat{n}$, then, as before, $\mathtt{S}[\widehat{n}] \xrightarrow{\ast} \mathtt{S}[\mathtt{n}]$ and (6.28) holds. Else, we have

$$\mathtt{S}[\mathtt{n}] \xrightarrow{\ast} \mathtt{S}[\widehat{n}] \xrightarrow{\ast} (\mathtt{S}[\mathtt{n}] + \widetilde{\pi}).$$

Again, (6.28) holds by applying Lemma 5.19 to the second half of the path above and then applying Lemma 5.16. Finally, (6.30) and (6.29) establish (6.28) as a loop invariant after the assignment to C[p] and the increment of p. *Q.E.D.*

Lemma 6.22 *Algorithm 1 returns with* S[n] *as a non-transitory state and* C *as a list of cycles such that (6.18) and (6.19) are satisfied.*

Proof: For now, suppose the algorithm terminates and consider the situation afterward. Then, by Lemma 6.21, (6.19), (6.27), and (6.28) hold. Furthermore, since *find_cycle*() returns *empty_cycle*, E is empty. Thus, $\mathbf{enb}(\alpha, \mathtt{S}[\mathtt{n}])$ implies $\mathbf{var}(\alpha) \in \mathtt{V}$. So, by (6.27) and (6.28), S[n] is a non-transitory state.

Next, let $\pi = \sum_{i=0}^{\mathtt{p}-1} \pi_i$. Then, by (6.28) and Lemma 5.16, $\mathtt{S}[\mathtt{n}] \xrightarrow{\ast} (\mathtt{S}[\mathtt{n}] + \pi)$. For any period $\widehat{\pi}$, Corollary 6.6 implies $\mathtt{S}[\mathtt{n}] \xrightarrow{\ast} (\mathtt{S}[\mathtt{n}] + \widehat{\pi})$. So, by Lemma 5.21, there exist q, $\mathcal{D}$, and ϕ such that

$$\mathtt{S}[\mathtt{n}] \overset{\mathcal{D}}{\xrightarrow{\ast}} (\phi - q\pi) \xrightarrow{\ast} (\mathtt{S}[\mathtt{n}] + \widehat{\pi}) \xrightarrow{\ast} \phi, \tag{6.31}$$

and $\mathbf{span}(\mathcal{D}) \cap \mathbf{span}(\pi) = \emptyset$. Suppose, toward a contradiction, that there exists a variable index k such that $x_k \in (\mathbf{span}(\widehat{\pi}) \setminus \mathbf{span}(\pi))$. Then, $\mathcal{D}$ is not empty since otherwise $\widehat{\pi} \leq q\pi$. Consequently, there exists an event δ such that $\mathbf{enb}(\delta, \mathtt{S}[\mathtt{n}])$ and $\delta \notin \mathbf{span}(\pi)$. By (6.27), δ is in E and the algorithm would not have terminated. This contradiction can be avoided only if (6.18) holds.

Since there are at most 2^K different values for **bool**(), (6.23) implies *find_cycle*() terminates. Also, since the size of V increases with each iteration through the repeat-loop and is bounded above by K, Algorithm 1 terminates and the lemma is established. *Q.E.D.*

6.4.1 Uniform Graphs

To show that only minimal periods are found by the algorithm if the graph is uniform, several preliminary results are needed.

Lemma 6.23 *Let $\widehat{\sigma} \xrightarrow{*} \sigma \xrightarrow{*} (\widehat{\sigma}+\pi)$ be a cycle in a uniform graph. If there exists an event α occurring in the cycle such that* $\mathbf{enb}(\alpha,\sigma)$ *and*

$$\forall\beta : \mathbf{enb}(\beta,\sigma) \wedge \mathbf{var}(\beta) \in \mathbf{span}(\pi) : \beta = \alpha, \tag{6.32}$$

then there exist $p > 0$ and minimal period $\widetilde{\pi}$ such that $\pi = p\widetilde{\pi}$.

Proof: By Lemma 5.19, $\sigma \xrightarrow{*} (\sigma + \pi)$. Since the graph is uniform, by Lemma 6.1, there exist $p > 0$ and a set of minimal cycles

$$\{i : 0 \leq i < p : \sigma_i \xrightarrow{*} (\sigma_i + \pi_i)\} \tag{6.33}$$

such that $\sigma_0 = \sigma$, $\sigma_{i+1} = (\sigma_i + \pi_i)$, and $\sigma_p = (\sigma + \pi)$. For any i, by Lemma 5.16 and $\mathbf{bool}(\sigma) = \mathbf{bool}(\sigma_i)$, (6.32) still holds if σ is replaced by σ_i and α is replaced by $(\alpha \oplus (\sigma_i - \sigma))$. Consequently, $(\alpha \oplus (\sigma_i - \sigma))$ occurs in $\sigma_i \xrightarrow{*} (\sigma_i + \pi_i)$ and so $\mathbf{var}(\alpha) \in \mathbf{span}(\pi_i)$. By Theorem 5.2, all of the π_i's are the same and π, being their sum, is $p\pi_0$. *Q.E.D.*

Lemma 6.24 *In a uniform graph, if there exist $p > 1$, minimal period $\widetilde{\pi}$, and a cycle*

$$\widehat{\sigma} \xrightarrow{*} \sigma_b \xrightarrow{*} \sigma_c \xrightarrow{*} (\widehat{\sigma} + p\widetilde{\pi}) \tag{6.34}$$

with event α and intermediate states σ_b and σ_c such that $\mathbf{enb}(\alpha,\sigma_b)$, $\mathbf{enb}(\alpha \oplus \widetilde{\pi}, \sigma_c)$,

$$\forall\beta : \mathbf{enb}(\beta,\sigma_b) \wedge \mathbf{var}(\beta) \in \mathbf{span}(\widetilde{\pi}) : \beta = \alpha, \tag{6.35}$$

and

$$\forall\gamma : \mathbf{enb}(\gamma,\sigma_c) \wedge \mathbf{var}(\gamma) \in \mathbf{span}(\widetilde{\pi}) : \gamma = (\alpha \oplus \widetilde{\pi}), \tag{6.36}$$

then $\mathbf{bool}(\sigma_b) = \mathbf{bool}(\sigma_c)$.

Proof: By Lemma 6.1 and Lemma 5.16, we have $\widehat{\sigma} \xrightarrow{*} (\widehat{\sigma} + \widetilde{\pi}) \xrightarrow{*} (\sigma_b + \widetilde{\pi})$. (See Figure 6.7.) Let ρ be the c.c.a. of σ_c and $(\sigma_b + \widetilde{\pi})$ with

$$\rho \overset{\mathcal{B}}{\xrightarrow{*}} \sigma_c \wedge \rho \overset{\mathcal{C}}{\xrightarrow{*}} (\sigma_b + \widetilde{\pi}).$$

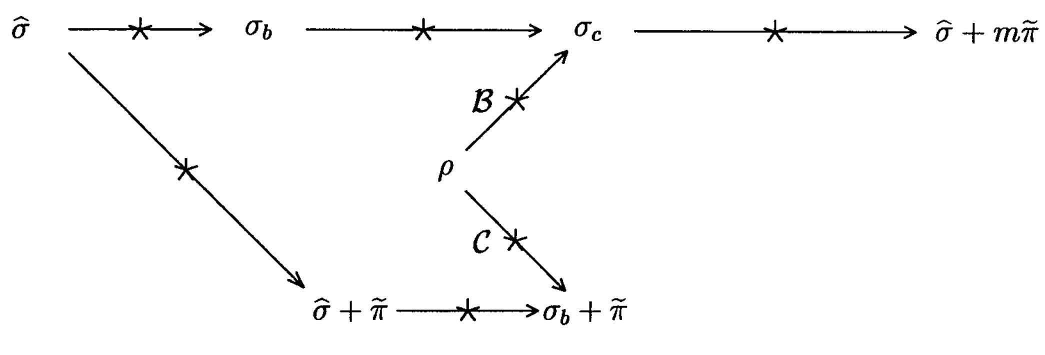

Figure 6.7: Proof of Lemma 6.24

Note that by Lemma 5.11, $\widehat{\sigma} \overset{*}{\longrightarrow} \rho$. Now, if $\mathcal{B}$ is not empty then there exists β such that $\beta \in \mathcal{B}$ and $\mathbf{enb}(\beta, \rho)$. By Lemma 5.10, $\beta \notin \mathcal{C}$ and therefore $\mathbf{enb}(\beta, \sigma_b + \widetilde{\pi})$ which implies $\mathbf{enb}(\beta \ominus \widetilde{\pi}, \sigma_b)$. Since β occurs in $\widehat{\sigma} \overset{*}{\longrightarrow} \rho \overset{*}{\longrightarrow} \sigma_c$, $\mathbf{var}(\beta) \in \mathbf{span}(\widetilde{\pi})$ and, by (6.35), $(\beta \ominus \widetilde{\pi}) = \alpha$ which means $\beta = (\alpha \oplus \widetilde{\pi})$. This equality and $\beta \in \mathcal{B}$ contradict $\mathbf{enb}(\alpha \oplus \widetilde{\pi}, \sigma_c)$; thus, $\mathcal{B}$ is empty. Similarly, $\mathcal{C}$ is empty due to (6.36). Therefore, $\sigma_c = (\sigma_b + \widetilde{\pi})$ and the lemma is proved. *Q.E.D.*

Theorem 6.2 *For a uniform graph, Algorithm 1 returns only minimal cycles and the periods of these cycles are all distinct and are the only minimal periods in the graph.*

Proof: By Lemma 6.22, it remains to show that *find_cycle*() returns only minimal cycles since then, by (6.19) and Theorem 5.2, the periods will be different from each other. Toward that end, let

$$\mathtt{S[i]} \xrightarrow{\mathtt{A[i]}} \mathtt{S[i+1]} \xrightarrow{\mathtt{A[i+1]}} \cdots \xrightarrow{\mathtt{A[m]}} \mathtt{s}. \tag{6.37}$$

be a cycle returned by *find_cycle*() and let $\mathtt{s} = \mathtt{S[i]} + \pi$. Let $\mathcal{A} = \{\hat{\imath} : \mathtt{i} \leq \hat{\imath} \leq \mathtt{m} : \mathtt{A}[\hat{\imath}]\}$. Let κ be the smallest index such that $x_K \in \mathbf{span}(\pi)$. Let λ be one more than the κ-th component of $\mathtt{S[i]}$. Then, by Lemma 5.2, $\alpha = \langle x_K, \lambda \rangle$ is in $\mathcal{A}$. Notice that by the choice of $\mathtt{k}$ in *find_cycle*() $\mathtt{A}[a] = \alpha$ only if no other event β, enabled at $\mathtt{S}[a]$, satisfies $\mathbf{var}(\beta) \in \mathbf{span}(\pi)$. Thus, by Lemma 6.23, there exists a minimal period $\widetilde{\pi}$ and an integer $p > 0$ such that $\pi = p\widetilde{\pi}$. If $p > 1$, then $(\alpha \oplus \widetilde{\pi}) \in \mathcal{A}$ by Lemma 5.2. Again, $\mathtt{A}[b] = (\alpha \oplus \widetilde{\pi})$ only if no other event γ enabled at $\mathtt{S}[b]$ satisfies $\mathbf{var}(\gamma) \in \mathbf{span}(\pi)$. So, by Lemma 6.24, $\mathbf{bool}(\mathtt{S}[a]) = \mathbf{bool}(\mathtt{S}[b])$ which violates the fact that (6.23) is a loop-invariant of *find_cycle*(). Thus, (6.37) is a minimal cycle and the theorem is proved *Q.E.D.*

6.4.2 Non-Uniform Graphs

First, note that the definition of state graphs allow for arbitrary initial state, i.e., $\Gamma(\mathcal{P}, \sigma)$ is defined to be the state graph consisting of all states in $\Sigma(\mathcal{P})$ that are reachable from σ under the state change relationship specified by $\mathcal{P}$. Then, by Lemma 6.4 and Lemma 6.7, $\Gamma(\mathcal{P}, \sigma)$ is a uniform graph whenever σ is a non-transitory state. Therefore, if it has not been determined that the state graph corresponding to the input PR set is uniform, then re-running Algorithm 1 starting at state $\mathtt{S}[\mathtt{n}]$ will return all minimal periods in the graph. As an extra computation-saving technique, if the input PR set is non-separable and there exists a cycle with period π such that the greatest common divisor of the elements in $\{k : 0 \leq k < K : \pi[k]\}$ is 2, then π is the unique minimal period by the following lemma.

Lemma 6.25 *In a non-separable graph with minimal period $\widehat{\pi}$, if π is a non-minimal period, there exists $p > 1$ such that $\pi = p\widehat{\pi}$.*

Proof: By Algorithm 1, there exists a non-transitory state σ. The rest follows from Lemma 6.6, Lemma 6.4, and Lemma 6.1. *Q.E.D.*

Note that the converse is not true. In Example 6.3, if x is set to be identically **false**, then, as indicated by the bold cycle in Figure 6.3, there are four transitions associated with every variable in the minimal period.

6.4.3 Implementation Issues

There are two observations that greatly simplify the implementation of Algorithm 1. First, though in the theoretical analysis, states are considered as vectors of integers, in the algorithm, only the Boolean values of these states are needed for comparisons. Hence, it is only necessary to represent states as Boolean vectors linked together by pointers. Similarly, an event can be identified by its transition without its occurrence number.

The second simplification arises from stability: once an event is enabled it remains enabled until the corresponding PR fires. Therefore, the current value of `E` can be updated incrementally. Every time a transition t occurs, add to `E` all transitions whose PR's become enabled because of the occurrence of t. When new variables are added to `V`, remove the transitions corresponding to these variables from `E`.

6.4.4 Complexity

In each simulation step, the enabled PR whose transition has the highest index is fired, the corresponding new state is computed and checked to see whether it has been encountered before, and, if it has not, all PR's that are enabled in the new state are determined. This step has similar complexity as one for any other selective simulation algorithm that attempts to find cycles by tracing out a single path. In particular, the amount of operations performed per step depends mainly on the number of states encountered so far, the number of guards affected by the firing, and the computation required to determine which of these guards change from **false** to **true**.

As for the number of simulation steps required by Algorithm 1, it is the sum of the lengths of the cycles found plus the steps needed to to reach a non-transitory state. If the graph is uniform and the initial state is a non-transitory state, then this number is optimal in the sense that any other algorithm needs this many steps just to trace out the minimal cycles. Therefore, as the following chapter demonstrates, Algorithm 1 provides a very simple and efficient means to determine the information that enables one to represent a PR set as a repetitive XER-system.

Chapter 7

Modeling PR Sets as XER-Systems

This chapter addresses the problem of generating an XER-system for a PR set once its minimal periods have been determined. First, as the following section shows, PR sets with separable graphs can be partitioned into independent components, each represented by its own XER-system. Then, the correspondence between the causality relationships of a PR set and those of an XER-system will be discussed. Finally, an algorithm for converting the former to the latter will be presented — it turns out that this conversion is much simpler if the PR set has only stable disjuncts.

7.1 Separable Graphs

In this section, we will show that if a state graph has more than one minimal period, then, after a non-transitory state has been reached, the corresponding PR set can be partitioned into independent components, one for each minimal period and each with its own set of variables. We need the following intermediate result.

Lemma 7.1 *Let σ_{nt} be a non-transitory state. Let $\tilde{\pi}$ be a minimal cycle. Then for any state τ reachable from σ_{nt}, there exist ϕ, $q \geq 0$, and period π such that*

$$\sigma_{\mathrm{nt}} \overset{\mathcal{A}}{\twoheadrightarrow} \phi \overset{\mathcal{B}}{\twoheadrightarrow} \tau \overset{\widetilde{\mathcal{B}}}{\twoheadrightarrow} (\phi + q\pi), \tag{7.1}$$

and

$$\mathbf{span}(\pi) \cap \mathbf{span}(\tilde{\pi}) = \emptyset \wedge \mathbf{span}(\mathcal{A}) \subseteq \mathbf{span}(\tilde{\pi}). \tag{7.2}$$

Proof: Let the minimal periods of the graph be $\tilde{\pi}_0$, $\tilde{\pi}_1$, ..., and $\tilde{\pi}_J$. W.l.g., let $\tilde{\pi} = \tilde{\pi}_0$. By Lemma 6.6 and Lemma 6.7,

$$\sigma_{\mathrm{nt}} \overset{*}{\longrightarrow} (\sigma_{\mathrm{nt}} + \tilde{\pi}_1) \overset{*}{\longrightarrow} (\sigma_{\mathrm{nt}} + \tilde{\pi}_1 + \tilde{\pi}_2) \overset{*}{\longrightarrow} \cdots \overset{*}{\longrightarrow} (\sigma_{\mathrm{nt}} + \pi)$$

where π is defined as $\tilde{\pi}_1+\tilde{\pi}_2+\ldots+\tilde{\pi}_J$. By Theorem 5.2, $\mathbf{span}(\pi)\cap\mathbf{span}(\tilde{\pi}) = \emptyset$. Also, by Lemma 5.21, (7.1) exists with $\mathbf{span}(\mathcal{A}) \cap \mathbf{span}(\pi) = \emptyset$.

Now, if α occurs in $\sigma_{\mathrm{nt}} \overset{*}{\longrightarrow} \tau$, then it occurs in a cycle starting from σ_{nt} by Lemma 6.2; so, it occurs in a minimal cycle by Lemma 6.12. Hence, $\mathbf{var}(\alpha) \in (\mathbf{span}(\pi) \cup \mathbf{span}(\tilde{\pi}))$. So, $\mathbf{span}(\mathcal{A}) \cap \mathbf{span}(\pi) = \emptyset$ implies the last conjunct in (7.2). *Q.E.D.*

The following lemma shows how variables that are not in the spanning set of a minimal period, can be removed from the guard of a transition whose variable *is* in the spanning set.

Lemma 7.2 *Let σ_{nt} be a non-transitory state. Let $\tilde{\pi}$ be a minimal period. Let α be an event such that* $\mathbf{var}(\alpha) \in \mathbf{span}(\tilde{\pi})$. *Let the guard of* $\mathbf{tran}(\alpha)$ *be*

$$G = B_0 \vee B_1 \vee \ldots \vee B_m.$$

For any j such that $0 \leq j \leq m$, let $B_j = C_j \wedge C'_j$ where

$$\begin{aligned} \mathbf{lit}(\beta) \in C_j &\Rightarrow \mathbf{var}(\beta) \in \mathbf{span}(\tilde{\pi}) \wedge \\ \mathbf{lit}(\beta) \in C'_j &\Rightarrow \mathbf{var}(\beta) \notin \mathbf{span}(\tilde{\pi}). \end{aligned} \tag{7.3}$$

Let

$$H = \begin{cases} G \textit{ with } B_j \textit{ replaced by } C_j & \textit{if } C'_j \textit{ is } \mathbf{true} \textit{ in } \sigma_{\mathrm{nt}} \\ G \textit{ with } B_j \textit{ removed} & \textit{if } C'_j \textit{ is } \mathbf{false} \textit{ in } \sigma_{\mathrm{nt}}. \end{cases} \tag{7.4}$$

Then, in any state τ reachable from σ_{nt}, the value of G is **true** *if and only if the value of H is* **true**.

Proof: Let τ be any state reachable from σ_{nt}. By Lemma 7.1, (7.1) exists and (7.2) holds. Since $\mathbf{span}(\mathcal{A}) \subseteq \mathbf{span}(\tilde{\pi})$, (7.3) implies

$$C'_j \text{ is } \mathbf{true} \text{ in } \sigma_{\mathrm{nt}} \Leftrightarrow C'_j \text{ is } \mathbf{true} \text{ in } \phi. \tag{7.5}$$

Next, assume that we are incrementally changing G to H by dealing with each disjunct in sequence. Consider the following two cases:

Case 1: (C_j' is **true** in σ_{nt}) Clearly, $G \Rightarrow H$. Suppose H is **true** in τ. If there exists $j' \neq j$ such that $B_{j'}$ is **true** in τ, then G is **true** in τ and we are done for this case. So, suppose for all $j' \neq j$, $B_{j'}$ is **false** in τ and C_j is **true** in τ. C_j remains **true** in $(\phi + q\pi)$ of (7.2) due to $\mathbf{span}(\widetilde{\pi}) \cap \mathbf{span}(\pi) = \emptyset$. Consequently, C_j is **true** in ϕ. Also, by (7.5), C_j' is **true** in ϕ. Therefore, B_j is **true** in ϕ and G, the guard for $\mathbf{tran}(\alpha)$ is **true** in ϕ. By stability, G is **true** in τ since α is not in $\mathcal{B}$. These observations establish the lemma for this case.

Case 2: (C_j' is **false** in σ_{nt}) By (7.4), $H \Rightarrow G$. Suppose G is **true** in τ. By stability and $\alpha \notin \widetilde{\mathcal{B}}$, G is **true** in $(\phi + q\pi)$. So, G is **true** in ϕ. Now, C_j' is **false** in σ_{nt} implies C_j' is **false** in ϕ; hence, B_j is **false** in ϕ. So, G is **true** in ϕ due to some disjunct $B_{j'}$, $j' \neq j$, being **true** in ϕ. Let $B_{j'} = C_{j'} \wedge C_{j'}'$ with $C_{j'}$ containing only literals whose variables are in $\mathbf{span}(\widetilde{\pi})$ and $C_{j'}'$ containing only literals whose variables are not. By (7.5), $B_{j'}$ is **true** in ϕ implies $C_{j'}'$ is **true** in σ_{nt}. So, by Case 1, $C_{j'}'$ can be removed. Consequently, we can assume that $B_{j'}$ contains only literals whose variables are in $\mathbf{span}(\widetilde{\pi})$. But then $B_{j'}$ is **true** in ϕ implies $B_{j'}$ is **true** in τ due to $\mathbf{span}(\widetilde{\pi}) \cap \mathbf{span}(\pi) = \emptyset$. Consequently, H is **true** in τ and the lemma is established. *Q.E.D.*

Example 7.1: Consider again the PR set of Example 6.1. A non-transitory state in its state graph is $\sigma_{\mathrm{nt}} = 10000$. In all states reachable from σ_{nt}, the behavior of the PR set is identical to the one below (See Figure 6.1):

$$\begin{array}{lcl@{\qquad\qquad}lcl}
\neg x_2 & \rightarrow & x_1\uparrow & \neg x_4 & \rightarrow & x_3\uparrow \\
x_1 & \rightarrow & x_2\uparrow & x_3 & \rightarrow & x_4\uparrow \\
x_2 & \rightarrow & x_1\downarrow & x_4 & \rightarrow & x_3\downarrow \\
\neg x_1 & \rightarrow & x_2\downarrow & \neg x_3 & \rightarrow & x_4\downarrow .
\end{array}$$

Note that once a non-transitory state has been reached, there is no further interaction between the variables in the spanning set of the minimal period 02200 and those in the spanning set of the minimal period 00022. □

As illustrated by the previous example, Lemma 7.2 implies that each minimal period $\widetilde{\pi}$ induces a PR set $\widetilde{\mathcal{P}}$ consisting only of variables in the spanning set of $\widetilde{\pi}$. Moreover, as far as those firings that involve variables in

$\mathbf{span}(\widetilde{\pi})$ are concerned, the behavior of $\widetilde{\mathcal{P}}$ is identical to the behavior of the original PR set $\mathcal{P}$ once a non-transitory state has been reached.

By Theorem 5.2, the PR sets induced by two different minimal periods do not share any variables; hence, each can be analyze independently. *So, for the rest of this chapter, all PR sets are assumed to be non-separable, each containing only variables in the spanning set of its unique minimal period, which will be denoted* π. *Moreover, unless stated otherwise, only states reachable from some fixed non-transitory state* σ_{nt} *are considered.*

7.2 Delay Insensitivity and Cause Sets

There are several technical issues concerning the definition of a "set of causes" (or cause set) for an event in a PR set. For convenience, we will say that an event $\langle x_k, l\rangle$ *has occurred* in a state σ if and only if $\sigma[k] \geq l$. Also, a set of events, $\mathcal{A}$, has occurred in σ if and only if every event in $\mathcal{A}$ has occurred in σ.

Intuitively, one criterion for $\mathcal{A}$ to be a cause set for α is that whenever all the events in $\mathcal{A}$ have occurred and α has not occurred, then α is enabled to occur. Conversely, each occurrence of α should be because one of its cause sets has occurred. However, these criteria are not sufficient to model delay-insensitivity as the following example illustrates.

Example 7.2: Consider the PR set and its associated state graph shown in Figure 7.1. First, note that there is no redundancy in the guard of $x_3\uparrow$ — x_0 is needed to avoid interference in state 21210, and x_2 is needed so that $\langle x_3, 3\rangle$ can fire in state 21321. Note also that at every state where $\langle x_1, 1\rangle$ has occurred, $\langle x_3, 1\rangle$ is enabled or has already occurred. Conversely, $\langle x_3, 1\rangle$ occurs only after $\langle x_1, 1\rangle$ has occurred. Thus, $\{\langle x_1, 1\rangle\}$ may appear to be a good candidate as the only cause set of $\langle x_3, 1\rangle$.

This analysis, however, is inadequate in that it ignores the delays between events. Recall that in the CMOS implementation of PR sets, if $\mathbf{lit}(\alpha)$ is a literal in the guard of $\mathbf{tran}(\beta)$, and β occurs due to an occurrence of α, then there is a delay associated with the two events which we can denote as $\Delta(\alpha, \beta)$. For instance, the event $\langle x_3, 1\rangle$ is enabled in state 11000 due to the disjunct $x_0 \wedge x_1$ being **true**. Under the XER-system model, $t(\langle x_3, 1\rangle)$, the

$$
\begin{array}{lcl}
\neg x_3 \wedge \neg x_1 & \rightarrow & x_0\uparrow \\
x_0 & \rightarrow & x_1\uparrow \\
x_4 \wedge x_1 \vee x_0 \wedge x_1 & \rightarrow & x_2\uparrow \\
x_0 \wedge x_1 \vee x_2 & \rightarrow & x_3\uparrow \\
x_2 \wedge x_3 & \rightarrow & x_0\downarrow \\
\neg x_4 \wedge \neg x_0 & \rightarrow & x_2\downarrow \\
\neg x_0 \wedge \neg x_2 & \rightarrow & x_3\downarrow \\
x_1 \wedge \neg x_3 \wedge \neg x_0 & \rightarrow & x_4\uparrow \\
x_4 \wedge x_3 & \rightarrow & x_1\downarrow \\
\neg x_1 & \rightarrow & x_4\downarrow
\end{array}
$$

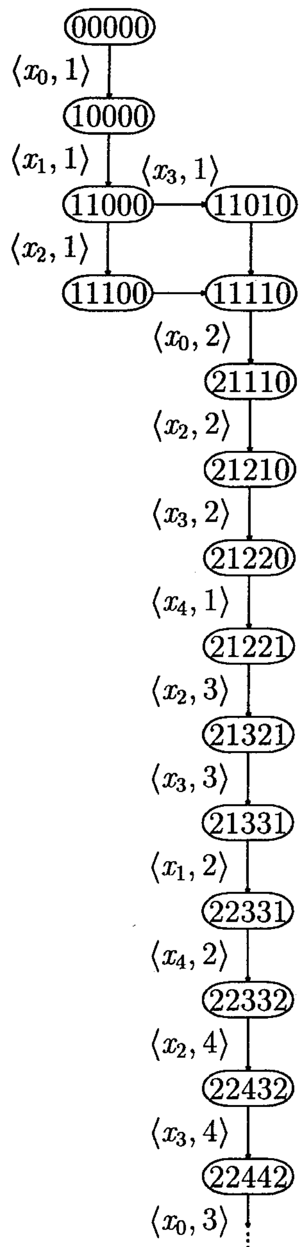

Figure 7.1: Example 7.2

time at which $\langle x_3, 1\rangle$ occurs, satisfies

$$\begin{aligned} t(\langle x_3, 1\rangle) \geq t(\langle x_0, 1\rangle) + \Delta(\langle x_0, 1\rangle, \langle x_3, 1\rangle) \wedge \\ t(\langle x_3, 1\rangle) \geq t(\langle x_1, 1\rangle) + \Delta(\langle x_1, 1\rangle, \langle x_3, 1\rangle). \end{aligned} \tag{7.6}$$

Similarly, the PR for $x_1\uparrow$ implies

$$t(\langle x_1, 1\rangle) \geq t(\langle x_0, 1\rangle) + \Delta(\langle x_0, 1\rangle, \langle x_1, 1\rangle).$$

Therefore, even though $\langle x_1, 1\rangle$ having occurred in a state implies $\langle x_0, 1\rangle$ has also occurred, *without further timing assumption*, it is possible that

$$\Delta(\langle x_0, 1\rangle, \langle x_1, 1\rangle) + \Delta(\langle x_1, 1\rangle, \langle x_3, 1\rangle) < \Delta(\langle x_0, 1\rangle, \langle x_3, 1\rangle).$$

Hence, using $\{\langle x_1, 1\rangle\}$ as a cause set for $\langle x_3, 1\rangle$ would ignore the possibility that it may be the timing constraint corresponding to $\langle x_0, 1\rangle$ that determines when $\langle x_3, 1\rangle$ can occur.

Next, suppose, because of the previous arguments, $\{\langle x_0, 1\rangle, \langle x_1, 1\rangle\}$ is chosen as the only cause set of $\langle x_3, 1\rangle$. Certainly, $\langle x_3, 1\rangle$ occurs or has occurred if and only if that set has occurred. Once again, however, this choice is inadequate. In state 11100, both disjuncts in the guard for $x_3\uparrow$ are **true**. Since x_2 is in the guard of $x_3\uparrow$, $\langle x_3, 1\rangle$ can occur after a sufficient delay has elapsed since the occurrence of $\langle x_2, 1\rangle$. Hence, $t(\langle x_3, 1\rangle)$ needs to satisfy *either* (7.6) *or*

$$t(\langle x_3, 1\rangle) > t(\langle x_2, 1\rangle) + \Delta(\langle x_2, 1\rangle, \langle x_3, 1\rangle). \tag{7.7}$$

So, once again, without further timing assumption, it is possible that $\Delta(\langle x_0, 1\rangle, \langle x_3, 1\rangle)$ and $\Delta(\langle x_1, 1\rangle, \langle x_3, 1\rangle)$ are sufficiently large so that $\langle x_3, 1\rangle$ occurs due to satisfying the timing constraint (7.7). Thus, when the delays between events can be arbitrary, the "complete set of cause sets" for $\langle x_3, 1\rangle$ in this example is $\{\{\langle x_0, 1\rangle, \langle x_1, 1\rangle\}, \{\langle x_2, 1\rangle\}\}$. $\Box$

As the previous example demonstrates, for arbitrary delays, it is necessary to consider the guard of $\mathbf{tran}(\alpha)$ to determine the set of cause sets for α. In particular, whenever there is a state where a disjunct of the guard is **true**, then a cause set containing the most recent events involving *all* of the variables in the disjunct needs to be included. This necessity arises from the fact that *any* one such event may be the event that determines when α can occur if the delay between them is large enough. Similarly, to have a complete set of cause sets, a cause set associated with *every* disjunct that is **true** in some state where α is enabled needs to be included since, under an appropriate set of delays, α may occur due to the timing constraint specified by that particular disjunct.

To formalize these notions, we have the definitions below. The first is to identify the most recent events that are responsible for the literals of a Boolean function being **true** in a particular state. The next two are the definitions of causes. Note that (7.9) and the first condition in (7.11) are the intuitive criteria mentioned in the beginning of the section, whereas (7.10) and the second condition in (7.11) are due to modeling arbitrary delays as discussed in the previous paragraph.

Definition: If B is a Boolean expression, then *the set of witnesses of B in σ* is

$$\mathbf{wit}(B,\sigma) = \{x_k, l : \ \mathbf{lit}(\langle x_k, l\rangle) \text{ is a literal in } B \wedge \sigma[k] = l : \langle x_k, l\rangle\}. \quad (7.8)$$

Definition: A set of events $\mathcal{A}$ is a *cause set* for an event α if

$$\forall \tau : \mathcal{A} \text{ has occurred in } \tau \text{ and } \alpha \text{ has not occurred in } \tau : \mathbf{enb}(\alpha, \tau), \quad (7.9)$$

and

$$\begin{aligned} &\exists \sigma, B : B \text{ is a disjunct in the guard of } \alpha : \\ &\qquad B \text{ is } \mathbf{true} \text{ in } \sigma \wedge \mathbf{wit}(B,\sigma) \subseteq \mathcal{A}. \end{aligned} \quad (7.10)$$

Definition: The set of L sets of events, $\{\mathcal{A}_0, \mathcal{A}_1, \ldots, \mathcal{A}_{L-1}\}$, is a *complete set of cause sets (CSCS)* for α if each $\mathcal{A}_i$ is a cause set of α, and, for any state σ such that $\mathbf{enb}(\alpha, \sigma)$ and for any disjunct B_j in the guard of α such that B_j is **true** in σ, there exists $\mathcal{A}_i$ such that

$$\mathcal{A}_i \text{ has occurred in } \sigma \wedge \mathbf{wit}(B_j, \sigma) \subseteq \mathcal{A}_i. \quad (7.11)$$

Definition: A cause set $\mathcal{A}$ for α is *minimal* if no proper subset of $\mathcal{A}$ is a cause set of α. A CSCS $\mathcal{S}$ for α is *minimal* if each member of $\mathcal{S}$ is a minimal cause set for α and no proper subset of $\mathcal{S}$ is a CSCS for α.

In the next three sub-sections, we will describe how to determine the CSCS of any event. Section 7.3 gives some general results. Section 7.4 deals only with PR's with stable disjuncts, whereas Section 7.5 includes those with unstable disjuncts.

7.3 Last-Enabled States

Definition: A *last-enabled state of an event* α is a state σ such that α is the only event enabled at σ.

Lemma 7.3 *Let σ be any last-enabled state for α. If τ is a state such that α has not occurred, then $\tau \xrightarrow{\star} \sigma$.*

Proof: Let ϕ be the c.c.d. for τ and σ. Since α has not occurred in τ and σ, α has not occurred in ϕ by Lemma 5.8. Consequently, $\sigma = \phi$ since $\sigma \xrightarrow{\star} \phi$ and the only event enabled at σ is α which has not occurred in ϕ. Hence, $\tau \xrightarrow{\star} \sigma$. *Q.E.D.*

Corollary 7.4 *For any event α, there is at most one last-enabled state for α. This state will be denoted* $\mathbf{last}(\alpha)$ *if it exists.*

Proof: Follows from the previous lemma and the fact that α has not occurred in any last-enabled state of α. *Q.E.D.*

Lemma 7.5 *Let B be a disjunct in the guard for* $\mathbf{tran}(\alpha)$*. For any σ such that* $\mathbf{enb}(\alpha, \sigma)$ *and B is* **true** *in σ, if* $\mathbf{wit}(B, \sigma) = \mathbf{wit}(B, \mathbf{last}(\alpha))$*, then* $\mathbf{wit}(B, \sigma)$ *is a minimal cause set of α.*

Proof: Let τ be any state such that $\mathbf{wit}(B, \sigma)$ has occurred and α has not occurred. By Lemma 7.3, $\tau \xrightarrow{\star} \mathbf{last}(\alpha)$. So, for any event $\langle x_k, l \rangle$ in $\mathbf{wit}(B, \sigma)$, $l = \sigma[k] \leq \tau[k] \leq \mathbf{last}(\alpha)[k] = l$. Hence, each literal in B has the same value in σ as in τ. Therefore, $\mathbf{enb}(\alpha, \tau)$, which validates (7.9). Also, the hypothesis implies (7.10) directly. So, $\mathbf{wit}(B, \sigma)$ is a cause set of α.

Now, if $\mathbf{wit}(B, \sigma)$ is not minimal then there exists another cause set $\mathcal{A}'$ such that $\mathcal{A}' \subset \mathbf{wit}(B, \sigma)$. But then, by definition, there exist σ' and a disjunct B' in the guard of α such that

$$\mathbf{wit}(B', \sigma') \subseteq \mathcal{A}' \subset \mathbf{wit}(B, \sigma).$$

So, every literal in B' is in B violating the fact that the guard of $\mathbf{tran}(\alpha)$ is in DNF. *Q.E.D.*

7.4 Stable Disjuncts

Lemma 7.6 *Let B be a stable disjunct of the PR for* $\mathbf{tran}(\alpha)$. *If B is* **true** *in σ, $\sigma \xrightarrow{*} \phi$,* $\mathbf{enb}(\alpha, \sigma)$, *and* $\mathbf{enb}(\alpha, \phi)$, *then B is* **true** *in ϕ and* $\mathbf{wit}(B, \sigma) = \mathbf{wit}(B, \phi)$.

Proof: Let $\langle x_k, l\rangle$ be an event in $\mathbf{wit}(B, \sigma)$. If $\langle x_k, l+1\rangle$ occurs in the path $\sigma \xrightarrow{*} \phi$, then B becomes **false** in an intermediate state along that path before the occurrence of α. Consequently, B is not a stable disjunct. So, to avoid a contradiction, for every $\langle x_k, l\rangle$ in $\mathbf{wit}(B, \sigma)$, $\phi[k] = \sigma[k]$ and the lemma follows. *Q.E.D.*

Corollary 7.7 *If B is a stable disjunct of the PR for* $\mathbf{tran}(\alpha)$, $\mathbf{enb}(\alpha, \sigma)$, *and B is* **true** *in σ, then* $\mathbf{wit}(B, \sigma)$ *is a minimal cause set of α.*

Proof: Follows directly from Lemma 7.5 and the previous lemma where ϕ is replaced by $\mathbf{last}(\alpha)$. *Q.E.D.*

Note that the condition of stable disjunct is necessary as Example 7.6 in Section 7.5 demonstrates. That example also illustrates an unstable disjunct that satisfies the hypothesis of Lemma 7.5.

Lemma 7.8 *Let the guard for* $\mathbf{tran}(\alpha)$ *be $B_0 \vee B_1 \vee \ldots \vee B_m$. If all the B_i's are mutex and each B_i is stable, then* $\mathbf{enb}(\alpha, \sigma) \wedge B_j$ *is* **true** *in σ implies* $\{\mathbf{wit}(B_j, \sigma)\}$ *is a minimal CSCS for α.*

Proof: By Corollary 7.7, $\mathbf{wit}(B_j, \sigma)$ is a minimal cause set. Let τ be a state such that $\mathbf{enb}(\alpha, \tau)$. Let $\sigma_{\mathrm{nt}} \overset{\mathcal{B}}{\xrightarrow{*}} \sigma$ and $\sigma_{\mathrm{nt}} \overset{\mathcal{C}}{\xrightarrow{*}} \tau$. Then by Lemma 5.7, there exists ϕ such that

$$(\sigma \overset{\mathcal{C} \setminus \mathcal{B}}{\xrightarrow{*}} \phi) \wedge (\tau \overset{\mathcal{B} \setminus \mathcal{C}}{\xrightarrow{*}} \phi).$$

Since $\alpha \notin (\mathcal{B} \cup \mathcal{C})$, $\mathbf{enb}(\alpha, \phi)$.

Now, $\mathbf{enb}(\alpha, \tau)$ implies there exists i such that B_i is **true** in τ. If $i \neq j$, then, by the fact that both B_i and B_j are stable disjuncts, $B_i \wedge B_j$ is **true** in ϕ which contradicts the hypothesis. So, $i = j$. Again, by the fact that B_j is a stable disjunct, $\mathbf{wit}(B_j, \tau) = \mathbf{wit}(B_j, \phi) = \mathbf{wit}(B_j, \sigma)$. Hence, $\mathbf{wit}(B_j, \sigma)$ has occurred in τ.

To establish the second condition in (7.11), note that if B_i is **true** in τ, then by the arguments above, $B_i = B_j$ and so $\{\mathbf{wit}(B_j, \sigma)\}$ is a CSCS. The fact that it is minimal follows from the fact that its only element is a minimal cause set. *Q.E.D.*

Corollary 7.9 *If the guard for* $\mathbf{tran}(\alpha)$ *is a conjunction* B *and* $\mathbf{enb}(\alpha, \sigma)$*, then a minimal CSCS for* α *is* $\{\mathbf{wit}(B, \sigma)\}$.

Proof: The claim follows directly from the lemma above and the fact that B is a stable disjunct. *Q.E.D.*

Lemma 7.10 *Let* $B_0 \vee B_1 \vee \ldots \vee B_m$ *be the guard for* $\mathbf{tran}(\alpha)$*. If all of the* B_i*'s are stable disjuncts, then*

$$\Omega(\alpha) = \{B_i : B_i \textit{ is } \mathbf{true} \textit{ in } \mathbf{last}(\alpha) : \mathbf{wit}(B_i, \mathbf{last}(\alpha))\} \tag{7.12}$$

is a minimal CSCS for α.

Proof: By Lemma 7.7, each $\mathbf{wit}(B_i, \mathbf{last}(\alpha))$ in $\Omega(\alpha)$ is a minimal cause set. Let τ be a state such that $\mathbf{enb}(\alpha, \tau)$. Then, by Lemma 7.3, $\tau \twoheadrightarrow \mathbf{last}(\alpha)$. Now, $\mathbf{enb}(\alpha, \tau)$ implies there exists B_i such that B_i is **true** in τ. By stability on B_i, B_i is **true** in $\mathbf{last}(\alpha)$ and $\mathbf{wit}(B_i, \tau) = \mathbf{wit}(B_i, \mathbf{last}(\alpha))$. Hence, $\mathbf{wit}(B_i, \mathbf{last}(\alpha))$ has occurred in τ.

To establish the second condition in (7.11), suppose B' is **true** in σ' and $\mathbf{enb}(\alpha, \sigma')$ for some disjunct B' in the guard of α. By the arguments above, $\sigma' \twoheadrightarrow \mathbf{last}(\alpha)$. Hence, by Lemma 7.6, B' is **true** in $\mathbf{last}(\alpha)$ and $\mathbf{wit}(B', \sigma') = \mathbf{wit}(B', \mathbf{last}(\alpha)) \in \Omega(\alpha)$; so, $\Omega(\alpha)$ is a CSCS. Furthermore, all its members are minimal cause sets. Also, if $\mathbf{wit}(B_j, \mathbf{last}(\alpha))$ is removed from the set, then there does not remain an element $\mathcal{A}_i$ in set such that

$$\mathcal{A}_i \text{ has occurred in } \mathbf{last}(\alpha) \wedge \mathbf{wit}(B_j, \mathbf{last}(\alpha)) \subseteq \mathcal{A}_i$$

because the guard of α is in DNF. Hence, $\Omega(\alpha)$ is a minimal CSCS for α. *Q.E.D.*

Corollary 7.11 *If* $\{ \mathcal{A}_0, \mathcal{A}_1, \ldots, \mathcal{A}_{L-1} \}$ *is a minimal CSCS of* α *as prescribed by Lemma 7.10, then for any* $i \geq 0$, $\{ \mathcal{A}_0 \oplus i\pi, \mathcal{A}_1 \oplus i\pi, \ldots, \mathcal{A}_{L-1} \oplus i\pi \}$ *is a minimal CSCS of* $\alpha \oplus i\pi$.

Proof: Since for any β and σ, $\mathbf{enb}(\beta, \sigma)$ if and only if $\mathbf{enb}(\beta \oplus i\pi, \sigma + i\pi)$, $\mathbf{last}(\alpha \oplus i\pi) = \mathbf{last}(\alpha) + i\pi$. The corollary then follows from Lemma 7.7 and the observation that $\mathbf{wit}(B, \sigma + i\pi) = (\mathbf{wit}(B, \sigma) \oplus i\pi)$. *Q.E.D.*

7.4.1 Conversion to XER-systems

Let $\mathcal{P}$ be a non-separable PR set with minimal period π. Further, let σ_{nt} be a non-transitory state and suppose that the transformation described in Section 7.1 have been performed so that all variables not in the spanning set of π have been removed. Let

$$\sigma_0 \xrightarrow{\alpha_0} \sigma_1 \xrightarrow{\alpha_1} \ldots \xrightarrow{\alpha_{n-1}} \sigma_n \tag{7.13}$$

with $\sigma_0 = \sigma_{\mathrm{nt}}$ be a minimal cycle. Then, as will be shown later, the causality and delay relationships of the PR set can be modeled by the repetitive XER-system $\mathcal{X}' = \langle E', R', \delta, \theta \rangle$ described below. (To avoid ambiguity, events and transitions in the state graph will be continued to be referred to as such, while events and transitions in the XER-system will be explicitly qualified.)

- E' is a set of n XER-system transitions, one associated with each α_i in (7.13). For reference, let $u(\alpha_i)$ denote the XER-system transition of $\mathcal{X}'$ that corresponds to α_i. Note that $u(\alpha_i)$ is different from $\mathbf{tran}(\alpha_i)$: If $\langle x_k, l \rangle$ and $\langle x_k, l+2 \rangle$ are both in the cycle, then $\mathbf{tran}(\langle x_k, l \rangle) = \mathbf{tran}(\langle x_k, l+2 \rangle)$. However, $u(\langle x_k, l \rangle)$ and $u(\langle x_k, l+2 \rangle)$ are different XER-system transitions in $\mathcal{X}'$. See Example 7.3.

- R' is the set of templates generated by Algorithm 2.

- θ is the occurrence-index offset function defined over the domain

  $$\mathcal{D} = \{\langle u, v, q \rangle : q \in R' \wedge u \in \mathbf{src}(q) \wedge v = \mathbf{tar}(q) : \langle u, v, q \rangle\}.$$

  The value of each $\theta(u, v, q)$ is determined by Algorithm 2.

- δ is the delay function between transitions, under some user-selected timing model, over the domain $\mathcal{D}$.

As described above, Algorithm 2, shown in Section A.2, is used to convert a PR set into an XER-system. Note that the procedure *fire_only*($\mathtt{D}, \mathtt{s}$) starts at state s and trace out a path firing only events in D. If and $\mathcal{E}(\mathcal{P})$ is the set of all possible events in PR set $\mathcal{P}$ and $\mathbf{enb}(\mathrm{a}, \mathrm{s})$, then *fire_only*$(\mathcal{E}(\mathcal{P}) \setminus \{\mathrm{a}\}, \mathrm{s})$, returns $\mathbf{last}(\mathrm{a})$. To see that this is the case, if the procedure does not terminate, then there exists a cycle that does not contain a. This existence implies the graph is either separable or s is not a non-transitory state — both are situations that having been excluded by our assumption. Hence, *fire_only*() terminates. Furthermore, let σ be the state returned by the procedure. By stability, $\mathbf{enb}(\mathrm{a}, \sigma)$; so, by construction, $\sigma = \mathbf{last}(\mathrm{a})$.

The transformation of a PR set into an XER-system is illustrated by the example below; arguments for its correctness will be provided afterward.

Example 7.3: Continuing with Example 6.6, since each variable transition, except for $g\uparrow$ and $g\downarrow$, appears only once in the cycle (6.20), we can define $u(\langle x_k, l\rangle)$ as

$$u(\langle x_k, l\rangle) = \begin{cases} \mathbf{tran}(\langle x_k, l\rangle) & \text{if } x_k \neq g \\ \text{`}g\uparrow\text{:0'} & \text{if } \langle x_k, l\rangle = \langle g, 1\rangle \\ \text{`}g\downarrow\text{:0'} & \text{if } \langle x_k, l\rangle = \langle g, 2\rangle \\ \text{`}g\uparrow\text{:1'} & \text{if } \langle x_k, l\rangle = \langle g, 3\rangle \\ \text{`}g\downarrow\text{:1'} & \text{if } \langle x_k, l\rangle = \langle g, 4\rangle. \end{cases}$$

Hence, the PR set can be described by the repetitive XER-system $\mathcal{X} = \langle E', R', \delta, \theta\rangle$ whose set of transitions is

$$\begin{aligned} E' = \{&aF_i\uparrow, aF_i\downarrow, aT_i\uparrow, aT_i\downarrow, bF_i\uparrow, bF_i\downarrow, bT_i\uparrow, bT_i\downarrow, \\ &cF_o\uparrow, cF_o\downarrow, cT_o\uparrow, cT_o\downarrow, d0\uparrow, d0\downarrow, d1\uparrow, d1\downarrow, \\ &e0\uparrow, e0\downarrow, e1\uparrow, e1\downarrow, \text{`}g\uparrow\text{:0'}, \text{`}g\downarrow\text{:0'}, \text{`}g\uparrow\text{:1'}, \text{`}g\downarrow\text{:1'}\}. \end{aligned}$$

To determine R' and θ, Algorithm 2 is applied with (6.20) as the minimal cycle. For the first event $\langle bT_i, 1\rangle$, its guard is conjunctive; so, *gen_template*() is called with $\mathbf{wit}(\text{guard of } bT_i\uparrow, \sigma_0)$ and $\langle bT_i, 1\rangle$ as arguments. Since $\mathbf{wit}(\neg e1 \wedge \neg e0, \sigma_0) = \{\langle e0, 0\rangle, \langle e1, 0\rangle\}$, the template $\{\{e0\downarrow, e1\downarrow\} \mapsto bT_i\uparrow\}$ is added to R'. Moreover, note that $\langle bT_i\uparrow, 1\rangle$ occurs in the cycle (6.20) but both $\langle e0, 0\rangle$ and $\langle e1, 0\rangle$ occur one period earlier. So, to reflect these differences in occurrence-indices, *gen_template*() defines

$$\begin{aligned} &\theta(e0\downarrow, bT_i\uparrow, \{e0\downarrow, e1\downarrow\} \mapsto bT_i\uparrow) = 1, \\ &\quad \theta(e1\downarrow, bT_i\uparrow, \{e0\downarrow, e1\downarrow\} \mapsto bT_i\uparrow) = 1. \end{aligned}$$

The second event in the cycle is $\langle g, 1\rangle$. All the disjuncts in its guard are stable, so *stab_disj*() is called. Since the disjuncts are mutually exclusive, an examination of the guard in σ_1 yields $aT_i \wedge bT_i$ as the only **true** disjunct in that state. Hence, *gen_template*() is called with $\{\langle aT_i, 1\rangle, \langle bT_i, 1\rangle\}$ and $\langle g, 1\rangle$ as arguments. The same call would have been made even if it had not been known that the disjuncts in the guard of $g\uparrow$ are mutually exclusive. So, the template

$$\{aT_i\uparrow, bT_i\uparrow\} \mapsto \text{`}g\uparrow\text{:0'}$$

and function values

$$\theta(aT_i\uparrow, \text{`}g\uparrow\text{:0'}, \{aT_i\uparrow, bT_i\uparrow\} \mapsto \text{`}g\uparrow\text{:0'}) = 1,$$
$$\theta(bT_i\uparrow, \text{`}g\uparrow\text{:0'}, \{aT_i\uparrow, bT_i\uparrow\} \mapsto \text{`}g\uparrow\text{:0'}) = 0$$

are added.

The third event in the cycle is $\langle cT_o\uparrow, 1\rangle$. All the disjuncts in its guard are stable but not mutually exclusive. So, *fire_only*() is called to find $\mathbf{last}(\langle cT_o\uparrow, 1\rangle)$ which, in this case, turns out to be σ_2. In that state, both the disjuncts $g \wedge aT_i$ and $g \wedge bT_i$ are **true**. Hence, the templates

$$\{\text{`}g\uparrow\text{:0'}, aT_i\uparrow\} \mapsto cT_o\uparrow\ ,\ \{\text{`}g\uparrow\text{:0'}, bT_i\uparrow\} \mapsto cT_o\uparrow$$

and function values

$$\theta(\text{`}g\uparrow\text{:0'}, cT_o\uparrow, \{\text{`}g\uparrow\text{:0'}, aT_i\uparrow\} \mapsto cT_o\uparrow) = 0,$$
$$\theta(aT_i\uparrow, cT_o\uparrow, \{\text{`}g\uparrow\text{:0'}, aT_i\uparrow\} \mapsto cT_o\uparrow) = 1,$$
$$\theta(\text{`}g\uparrow\text{:0'}, cT_o\uparrow, \{\text{`}g\uparrow\text{:0'}, bT_i\uparrow\} \mapsto cT_o\uparrow) = 0,$$
$$\theta(bT_i\uparrow, cT_o\uparrow, \{\text{`}g\uparrow\text{:0'}, bT_i\uparrow\} \mapsto cT_o\uparrow) = 0$$

are added.

By continuing the analysis for the other events in the cycle, R' and θ can be determined. □

We will now proceed to prove the correctness of Algorithm 2 in the case where the PR's have only stable disjuncts. The case of unstable disjuncts is very complicated and will be postponed until the next section.

First, we will say that the XER-system event $\langle u(\alpha), e\rangle$ "represents" the event $(\alpha \oplus e\pi)$. Analogously, we extend the definition to sets of events and to sets of sets of events.

Lemma 7.12 *Suppose $\mathcal{P}$ has only stable disjuncts. Let $\mathcal{X}'$ be the repetitive XER-system generated by Algorithm 2. Let $\mathcal{X}$ be the general XER-system induced by $\mathcal{X}'$. Then, for all $i \geq \theta_{max}$, if the XER-system event $\langle u, i \rangle$ represents the event α, then the set of all cause sets of $\langle u, i \rangle$ in $\mathcal{X}$ represents a minimal CSCS of α.*

Proof: By Corollary 7.7, whenever *gen_template*() is called from Algorithm 2 or from *stab_disj*(), its first argument is a minimal cause set of its second argument. Moreover, for any $\alpha_{\mathtt{i}}$, let $\mathcal{W}_{\mathtt{i}}$ be the set of all $\mathbf{wit}(B, \sigma)$ such that there is a call of *gen_template*($\mathbf{wit}(B, \sigma), \alpha_{\mathtt{i}}$) during the execution of the algorithm. From the topology of the program and Lemmas 7.9, 7.8, and 7.10, $\mathcal{W}_{\mathtt{i}}$ is a minimal CSCS for $\alpha_{\mathtt{i}}$.

Next, suppose the call *gen_template*($\{\gamma_0, \gamma_1, \ldots, \gamma_J\}, \alpha$) is made. By the assumption made on the input PR set, every variable occurs in the minimal cycle (7.13). Consequently, for each event γ_j, there exist an event β_j in the cycle and an integer e_j such that $\gamma_j = (\beta_j \oplus e_j\pi)$. So, let γ_j be represented by the pair $\langle u(\beta_j), e_j \rangle$ which is an XER-system event if $e_j \geq 0$.

The template generated by *gen_template*($\{\hat{\jmath} :: \gamma_{\hat{\jmath}}\}, \alpha$) is

$$q = \{\hat{\jmath} :: u(\beta_{\hat{\jmath}})\} \mapsto u(\alpha) \tag{7.14}$$

with

$$\theta(u(\beta_j), u(\alpha), \{\hat{\jmath} :: u(\beta_{\hat{\jmath}})\} \mapsto u(\alpha)) = -e_j. \tag{7.15}$$

By (4.7), this template induces the rule

$$q\lceil i = \{\hat{\jmath} :: \langle u(\beta_{\hat{\jmath}}), i + e_{\hat{\jmath}} \rangle\} \mapsto \langle u(\alpha), i \rangle$$

for $i \geq \theta_{\max}$. However, by Corollary 7.11, $\gamma_j = (\beta_j \oplus e_j\pi)$ is in a cause set of α implies $(\gamma_j \oplus i\pi) = (\beta_j \oplus (i + e_j)\pi)$ is in a cause set of $(\alpha \oplus i\pi)$, for any $i \geq 0$. So, for $i \geq \theta_{\max} \geq \mathbf{max}\{-e_j\}$, the source set of $q\lceil i$ represents the cause set $(\{\hat{\jmath} :: \gamma_{\hat{\jmath}}\} \oplus i\pi)$ of $(\alpha \oplus i\pi)$. By Corollary 7.11, this cause set is minimal. Moreover, since $\{\hat{\jmath} :: \gamma_{\hat{\jmath}}\}$ can be any minimal cause set of α, the same corollary implies that the set of all cause sets of $\langle u(\alpha), i \rangle$ in $\mathcal{X}$ represents a minimal CSCS of $(\alpha \oplus i\pi)$. *Q.E.D.*

The interpretation of this result is that $\mathcal{P}$ can be specified as a pseudorepetitive XER-system whose repeated part is $\mathcal{X}'$, as generated by Algorithm 2. Since, by Lemma 4.8, the period of a pseudorepetitive system is the same as its repeated part, it is therefore sufficient to analyze $\mathcal{X}'$.

$$
\begin{array}{lcl}
\neg x_3 & \rightarrow & x_0\uparrow \\
\neg x_2 \wedge \neg x_3 & \rightarrow & x_1\uparrow \\
x_1 & \rightarrow & x_2\uparrow \\
x_2 & \rightarrow & x_1\downarrow \\
x_0 \wedge x_1 \vee x_0 \wedge x_2 & \rightarrow & x_3\uparrow \\
\neg x_1 \wedge x_3 & \rightarrow & x_2\downarrow \\
\neg x_1 \wedge x_3 & \rightarrow & x_0\downarrow \\
\neg x_0 \wedge \neg x_2 & \rightarrow & x_3\downarrow
\end{array}
$$

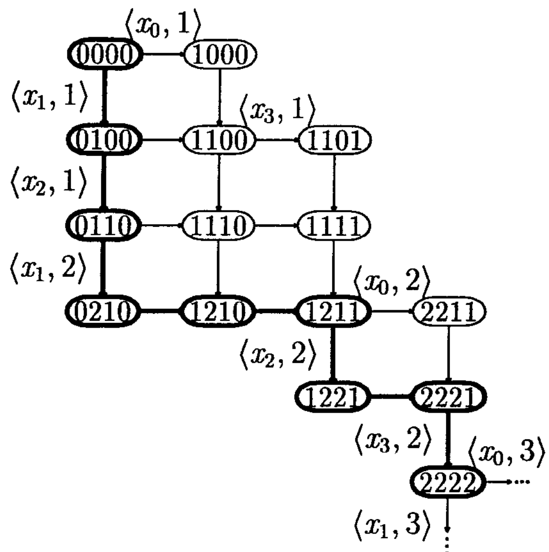

Figure 7.2: A state graph with an unstable disjunct

7.5 Unstable Disjuncts

If the PR of an event contains an unstable disjunct, then the analysis needed to determine its CSCS becomes very complicated. Consider the following simple example.

Example 7.4: In Figure 7.2, a PR set and its state graph are shown. Note that $x_0 \wedge x_1$ is an unstable disjunct for $x_3\uparrow$. Now, suppose that, in applying Algorithm 1 of the previous chapter, the cycle in bold is found. If only the disjuncts that are **true** in $\textbf{last}(\langle x_3, 1\rangle) = 1210$ are considered (as per Corollary 7.7 for stable disjuncts), then an erroneous conclusion that $\textbf{wit}(x_0 \wedge x_2, 1210) = \{\langle x_0, 1\rangle, \langle x_2, 1\rangle\}$ is the only cause set of $\langle x_3, 1\rangle$ will be made. Instead, it is necessary to "backtrack" from state 1210 to state 1100, which is not in the cycle, in order to determine that $\{\langle x_0, 1\rangle, \langle x_1, 1\rangle\}$ is also a cause set. $\square$

This section addresses the issues concerning unstable disjuncts. Because each unstable disjunct can switch between **true** and **false**, and vice versa, almost arbitrarily, no succinct result on how to determine the minimal CSCS of an event has been found. Instead, we will describe a method for finding a CSCS that, in practice, is often minimal. Unfortunately, the procedure, in the worst-case, has exponential complexity. Hence, since unstable disjuncts arise rarely in practice, for those readers who are willing to restrict their consideration to PR sets with only stable disjuncts, this section can be skipped with little loss of continuity.

7.5.1 Backtracking

As illustrated in the previous example, if the guard of an event has an unstable disjunct, then to find its CSCS, it is sometimes necessary to "trace backward" from a given state σ to another state τ such that there exists β with $\tau \xrightarrow{\beta} \sigma$. Now, the guard of **tran**(β) is **true** in τ, so it is **true** in σ since no self-invalidating PR's are allowed. Also, if $\beta = \langle x_k, l \rangle$, then by definition of state change, $\sigma[k] = l$. However, these two criteria are not sufficient to allow one to "backtrack" from a given state as the following example illustrates.

Example 7.5: Continuing with the previous example, let $\sigma = 2211$. Suppose we want to find all events $\langle x_k, l \rangle$ and states τ such that $\tau \xrightarrow{\langle x_k, l \rangle} \sigma$. In state σ, the guard for **tran**$(\langle x_0, 2 \rangle)$ is **true** and $\sigma[0] = 2$. As it turns out, $1211 \xrightarrow{\langle x_0, 2 \rangle} \sigma$. However, the guard for **tran**$(\langle x_1, 2 \rangle)$ is also **true** in σ and $\sigma[1] = 2$. But, there is not a state τ such that $\tau \xrightarrow{\langle x_1, 2 \rangle} \sigma$. $\square$

To fully describe the procedure for backtracking, we have the following definition.

Definition: The *set of incoming events* of σ, denoted **incoming**(σ), is

$$\{\alpha : (\exists \tau :: \tau \xrightarrow{\alpha} \sigma) : \alpha\}.$$

On page 166, the procedure *find_incoming*(s), which is used for determining **incoming**(s), is outlined. By the arguments given in the beginning of this sub-section, I, when initialized, is a superset of **incoming**(s). Note that if b ∈ I, then b occurs in the path from σ_{init} to s and therefore D exists. The rest of the procedure determines which of the original members of I can be removed to yield **incoming**(s). The test makes use of Lemma 5.14: If

$\phi \overset{\mathtt{D}}{\twoheadrightarrow} \mathtt{s}$ and $\mathtt{b} \in \mathtt{D}$, then

$$\forall \tau :: (\tau \xrightarrow{\mathtt{b}} \mathtt{s} \Leftrightarrow \phi \overset{\mathtt{D} \setminus \{\mathtt{b}\}}{\twoheadrightarrow} \tau).$$

Thus, any $\mathtt{b}$ whose corresponding $\mathtt{t}$ does not change to $\mathtt{s}$ via $\mathtt{b}$ is not an incoming event of $\mathtt{s}$. Hence, after the removal of all such $\mathtt{b}$'s, $\mathtt{I}$ is **incoming**($\mathtt{s}$).

If a list of events leading to $\mathtt{s}$ is maintained as, for instance,

$$\sigma_{\text{init}} \xrightarrow{\alpha_0} \sigma_1 \xrightarrow{\alpha_1} \sigma_2 \xrightarrow{\alpha_2} \cdots \xrightarrow{\alpha_{n-2}} \sigma_{n-1} \xrightarrow{\alpha_{n-1}} \mathtt{s},$$

then determining $\mathtt{D}$ in *find_incoming*($\mathtt{s}$) amounts to finding α_j such that $\alpha_j = \mathtt{b}$ and letting $\mathtt{D}$ be $\{i : j \leq i < n : \alpha_i\}$, which, typically, is a small set. Also, in cases where it is necessary to backtrack several steps from a state, $\mathtt{E}$ does not need to be evaluate anew each step; instead, it can be updated incrementally by determining which guards change values after each step.

7.5.2 Cause Sets

As an illustration of how the presence of unstable disjuncts in a guard can create situations that may seem counter-intuitive at first, consider the following example.

Example 7.6: Consider the following PR set:

$$\begin{array}{lcl}
\neg x_5 \wedge \neg x_4 \vee x_0 \wedge x_5 \wedge \neg x_4 & \rightarrow & x_3\uparrow \\
\neg x_5 & \rightarrow & x_0\uparrow \\
x_0 & \rightarrow & x_2\uparrow \\
\neg x_4 \wedge x_3 \vee \neg x_4 \wedge x_2 \wedge \neg x_5 \vee x_4 \wedge \neg x_3 & \rightarrow & x_1\uparrow \\
x_0 \wedge x_1 \wedge \neg x_3 \vee x_2 \wedge x_3 \wedge x_1 \vee x_2 \wedge x_4 & \rightarrow & x_5\uparrow \\
x_1 \wedge x_3 & \rightarrow & x_4\uparrow \\
x_4 \wedge \neg x_1 \wedge x_0 & \rightarrow & x_3\downarrow \\
x_2 \wedge x_5 \wedge x_1 \wedge x_4 \wedge \neg x_3 & \rightarrow & x_0\downarrow \\
x_5 \wedge \neg x_0 & \rightarrow & x_4\downarrow \\
\neg x_3 \wedge \neg x_4 \wedge \neg x_0 \vee x_3 \wedge x_4 & \rightarrow & x_1\downarrow \\
x_5 \wedge \neg x_0 \wedge \neg x_1 & \rightarrow & x_2\downarrow \\
x_5 \wedge \neg x_0 \wedge \neg x_2 & \rightarrow & x_5\downarrow .
\end{array}$$

Its state graph is depicted in Figure 7.3 where states at which $\langle x_5, 1\rangle$ is enabled are shown in bold.

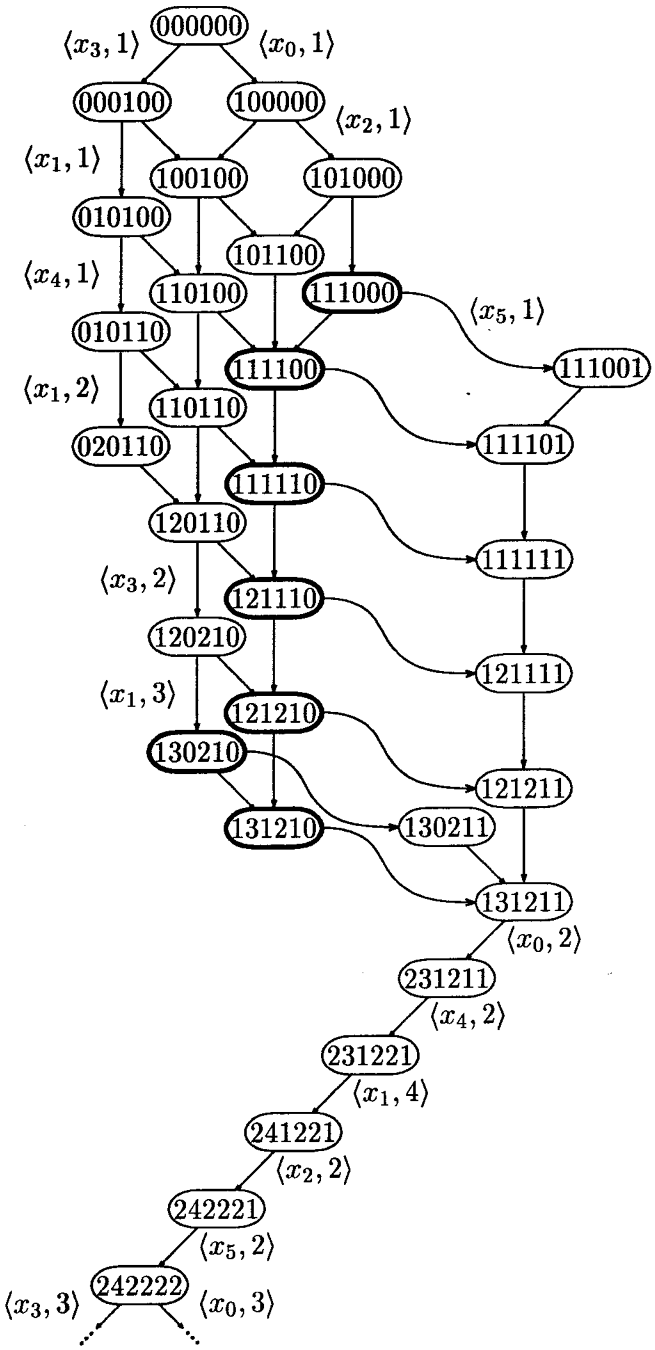

Figure 7.3: PR set with unstable disjuncts

Let α be $\langle x_5, 1\rangle$. Accordingly, let B_0 be $x_0 \wedge x_1 \wedge \neg x_3$, B_1 be $x_2 \wedge x_3 \wedge x_1$, and B_2 be $x_2 \wedge x_4$. The following are some interesting observations concerning the cause sets of α.

- B_0 is an unstable disjunct for **tran**(α) because it is **true** in 111000 and becomes **false** in 111100 before α has occurred. Moreover, **wit**$(B_0, 111000) = \{\langle x_0, 1\rangle, \langle x_1, 1\rangle, \langle x_3, 0\rangle\}$. So, at state 120210, **wit**$(B_0, 111000)$ has occurred. But, in that state α is not enabled and it has not yet occurred. Hence, **wit**$(B_0, 111000)$ is *not* a cause set for α despite of the fact that B_0 is **true** in 111000 and **enb**$(\alpha, 111000)$.

- B_1 is an unstable disjunct for **tran**(α) since it is **true** in 111110 and becomes **false** in 121110 before α has occurred. However, unlike the situation for B_0, **wit**$(B_1, 111110) = \{\langle x_1, 1\rangle, \langle x_2, 1\rangle, \langle x_3, 1\rangle\}$ *is* a cause set for α since α either is enabled or has occurred in every state where **wit**$(B_1, 111110)$ has occurred.

- B_2 is **true** in 111110. Note that it is **true** in **last**(α) also. In fact, it is a stable disjunct and, by Corollary 7.7, **wit**$(B_2, 111110)$ is a cause set of α.

- B_0 is **true** in 130210 also. Because it is **true** in **last**(α), by Lemma 7.5, **wit**$(B_0, 130210)$ is a cause set of α. Note, however, that **wit**$(B_0, 130210) = \{\langle x_0, 1\rangle, \langle x_1, 3\rangle, \langle x_3, 2\rangle\} \neq$ **wit**$(B_0, 111000)$. Thus, a single unstable disjunct B_0 can be **true** in two different states and give rise to two different sets of witnesses. This situation cannot occur if the disjunct is stable by virtue of Lemma 7.7.

□

A counterpart of Corollary 7.7 for unstable disjunct is given below.

Lemma 7.13 *If the guard for* **tran**(α) *is* $B_0 \vee B_1 \vee \ldots \vee B_m$, **enb**$(\alpha, \sigma)$, *and*

$$W_0(\alpha, \sigma) = \bigcup_{i=0}^{m} \mathbf{wit}(B_i, \sigma), \tag{7.16}$$

then $W_0(\alpha, \sigma)$ *is a cause set of* α.

Proof: Let τ be any state such that α has not occurred and $\neg\mathbf{enb}(\alpha,\tau)$. Let $\sigma_{\mathrm{nt}} \overset{\mathcal{B}}{\twoheadrightarrow} \sigma$ and $\sigma_{\mathrm{nt}} \overset{\mathcal{C}}{\twoheadrightarrow} \tau$. Then, by Lemma 5.7 there exists ϕ such that

$$(\sigma \overset{\mathcal{C}\setminus\mathcal{B}}{\twoheadrightarrow} \phi) \wedge (\tau \overset{\mathcal{B}\setminus\mathcal{C}}{\twoheadrightarrow} \phi).$$

Since α has not occurred in σ or τ, it has not occurred in ϕ. So, by stability, $\mathbf{enb}(\alpha,\sigma)$ implies $\mathbf{enb}(\alpha,\phi)$ and there exists a disjunct B_i such that B_i is **true** in ϕ.

Since $\neg\mathbf{enb}(\alpha,\tau)$, B_i is **false** in τ and B_i contains a literal that is **false** in τ but **true** in ϕ. Let the variable of that literal be x_k and let l be $\phi[k]$. Consider $\beta = \langle x_k, l\rangle$. Since $\mathbf{lit}(\beta)$ has different values in τ and ϕ, β has not occurred in τ and $\beta \in (\mathcal{B}\setminus\mathcal{C})$. So, $\beta \in \mathcal{B}$ and, consequently, $\sigma[k] \geq l$. But $\sigma[k] \leq \phi[k] = l$; so, $\sigma[k] = l$. By the choice of x_k, $\mathbf{lit}(\beta)$ is **true** in ϕ and so it is also **true** in σ since $\phi[k] = \sigma[k]$. Hence, β is in $\mathbf{wit}(B_i,\sigma)$ which is contained in $W_0(\alpha,\sigma)$. Consequently, if $W_0(\alpha,\sigma)$ has occurred in τ, then either $\mathbf{enb}(\alpha,\tau)$ or α has occurred in τ. *Q.E.D.*

Note that no claim is made as to whether the cause set defined by (7.16) is minimal. In fact, if B is a stable disjunct, this cause set is typically larger than $\mathbf{wit}(B,\sigma)$, which *is* minimal. Minimality is not guaranteed even for unstable disjuncts. In fact, when Lemma 7.13 is applied to B_1 and 111100 in Example 7.6, a cause set of $\{\langle x_0,1\rangle, \langle x_1,1\rangle, \langle x_2,1\rangle, \langle x_3,1\rangle\}$ is prescribed. However, $\{\langle x_1,1\rangle, \langle x_2,1\rangle, \langle x_3,1\rangle\}$ is sufficient. Currently, no efficient way to determine the minimal cause set due to an unstable disjunct has been discovered.

Lemma 7.14 *Let the guard for* $\mathbf{tran}(\alpha)$ *be* $B_0 \vee B_1 \vee \ldots \vee B_m$. *Suppose* $\mathbf{enb}(\alpha,\sigma)$ *and* B_j *is* **true** *in* σ. *Let* $W_0(\alpha,\sigma)$ *be as defined in (7.16). Let* $W_1(\alpha,B_j,\sigma)$ *be*

$$\begin{array}{l}\{\gamma : \gamma \in \mathbf{wit}(B_j,\sigma) \vee \\ \qquad \gamma \in W_0(\alpha,\sigma) \wedge\ \mathbf{wit}(B_j,\sigma) \textit{ has occurred in } \mathbf{last}(\gamma) : \gamma\}.\end{array} \tag{7.17}$$

Then,

$$\mathbf{cause}(\alpha,B_j,\sigma) = \begin{cases} \mathbf{wit}(B_j,\sigma) & \text{if } \mathbf{wit}(B_j,\sigma) = \mathbf{wit}(B_j,\mathbf{last}(\alpha)) \\ W_1(\alpha,B_j,\sigma) & \text{if } \mathbf{wit}(B_j,\sigma) \neq \mathbf{wit}(B_j,\mathbf{last}(\alpha)) \end{cases} \tag{7.18}$$

is a cause set of α.

Proof: By Lemma 7.5, it is sufficient to consider the case when $\mathbf{wit}(B_j, \sigma) \neq \mathbf{wit}(B_j, \mathbf{last}(\alpha))$. By construction, $\mathbf{wit}(B_j, \sigma) \subseteq \mathbf{cause}(\alpha, B_j, \sigma)$ and so (7.10) is satisfied. Next, let τ be any state such that $\mathbf{cause}(\alpha, B_j, \sigma)$ has occurred, α has not occurred, and $\neg\mathbf{enb}(\alpha, \tau)$. By the arguments in the proof of Lemma 7.13, there exists β in $W_0(\alpha, \sigma)$ such that β has not occurred in τ. So, by Lemma 7.3, $\tau \nrightarrow \mathbf{last}(\beta)$. But $\mathbf{wit}(B_j, \sigma) \subseteq \mathbf{cause}(\alpha, B_j, \sigma)$. Thus, $\mathbf{wit}(B_j, \sigma)$ has occurred in τ and, consequently, it has occurred in $\mathbf{last}(\beta)$. So, by (7.17), $\beta \in \mathbf{cause}(\alpha, B_j, \sigma)$ which is a direct contradiction to the prior conclusion that β has not occurred in τ. Thus, an absurdity can be avoided only if no such τ exists. *Q.E.D.*

Example 7.7: Consider again the cause sets of $\alpha = \langle x_5, 1\rangle$ in Example 7.6.

- Since $\mathbf{enb}(\alpha, 111000)$ and B_0 is **true** in 111000, $\mathbf{cause}(\alpha, B_0, 111000)$ is a cause set. Next, note that $W_0(\alpha, 111000) \setminus \mathbf{wit}(B_0, 111000) = \{\langle x_2, 1\rangle\}$. Since $\mathbf{wit}(B_0, 111000) = \{\langle x_0, 1\rangle, \langle x_1, 1\rangle, \langle x_3, 0\rangle\}$ has occurred in $\mathbf{last}(\langle x_2, 1\rangle)$ which has a value of 130210, $\langle x_2, 1\rangle$ is included in $\mathbf{cause}(\alpha, B_0, 111000)$. Hence, $\{\langle x_0, 1\rangle, \langle x_1, 1\rangle, \langle x_2, 1\rangle, \langle x_3, 0\rangle\}$ is a cause set of α.

- Since $\mathbf{enb}(\alpha, 111100)$ and B_1 is **true** in 111100, $\mathbf{cause}(\alpha, B_1, 111100)$ is a cause set. Next, $W_0(\alpha, 111100) \setminus \mathbf{wit}(B_1, 111100) = \{\langle x_0, 1\rangle\}$. Since $\mathbf{wit}(B_1, 111100) = \{\langle x_1, 1\rangle, \langle x_2, 1\rangle, \langle x_3, 1\rangle\}$ has *not* occurred in $\mathbf{last}(\langle x_0, 1\rangle) = 020110$, $\langle x_2, 1\rangle$ is excluded from $\mathbf{cause}(\alpha, B_1, 111100)$. Hence, $\{\langle x_1, 1\rangle, \langle x_2, 1\rangle, \langle x_3, 1\rangle\}$ is a cause set of α.

□

Though no claim to the minimality of (7.18) is made, in practice, as illustrated by the above example, the cause sets defined by (7.18) are often minimal.

In the previous example, we can continue to compute $\mathbf{cause}(\alpha, B_j, \sigma)$ for every disjunct B_j and every state σ such that $\mathbf{enb}(\alpha, \sigma)$ and B_i is **true** in σ. Then,

$$\{B_j, \sigma : \mathbf{enb}(\alpha, \sigma) \wedge B_j \text{ is } \mathbf{true} \text{ in } \sigma : \mathbf{cause}(\alpha, B_j, \sigma)\}$$

is a CSCS for α because setting $\mathcal{A}_i$ to $\mathbf{cause}(\alpha, B_j, \sigma)$ satisfies (7.11). Lemma 7.15, however, can be used to reduce the number of times a cause set is to be computed. The result makes use of the following definition.

Definition: A disjunct B is *critically* **true** in τ, denoted $\mathbf{ctrue}(B,\tau)$, if B is **true** in τ and $\forall\phi : \phi \longrightarrow \tau : B$ is **false** in ϕ.

Note that for any state such that B is **true** in σ, there exists a state τ such that $\tau \overset{*}{\longrightarrow} \sigma$ and $\mathbf{ctrue}(B,\tau)$ because the state graph is acyclic due to the partial order imposed on the states by the weight function.

Lemma 7.15 *Let $B_0 \vee B_1 \vee \ldots \vee B_m$ be the guard of α. Then,*

$$\{B_j, \tau : \mathbf{enb}(\alpha,\tau) \wedge \mathbf{ctrue}(B_j,\tau) : \mathbf{cause}(\alpha, B_j, \tau)\} \qquad (7.19)$$

is a CSCS of α.

Proof: By Lemma 7.14, each member of (7.19) is a cause set of α. Let B_j and σ be such that $\mathbf{enb}(\alpha,\sigma)$ and B_j is **true** in σ. Then, by previous arguments, there exists τ such that $\mathbf{ctrue}(B_j,\tau)$ and $\tau \overset{*}{\longrightarrow} \sigma$. Since B_j is **true** in τ, $\mathbf{enb}(\alpha,\tau)$. Let n be the length of the path $\tau \overset{*}{\longrightarrow} \sigma$ and let $\mathcal{A}_i = \mathbf{cause}(\alpha, B_j, \tau)$. We will show, by induction on n, that (7.11) is satisfied.

<u>Base Case</u>: $(n = 0)$ Here, $\mathbf{ctrue}(B_j,\sigma)$ and $\mathcal{A}_i = \mathbf{cause}(\alpha, B_j, \sigma)$. Hence, $\mathcal{A}_i$ has occurred in σ and $\mathbf{wit}(B_j,\sigma) \subseteq \mathcal{A}_i$.

<u>Inductive Step</u>: Assume (7.11) is satisfied if the length of $\tau \overset{*}{\longrightarrow} \sigma$ is n. Consider ϕ such that $\sigma \xrightarrow{\gamma} \phi$, B_j is **true** in ϕ, and $\mathbf{enb}(\alpha,\phi)$. Now, $\mathcal{A}_i$ has occurred in σ, so it has occurred in ϕ. Moreover, $\mathbf{wit}(B_j,\sigma) = \mathbf{wit}(B_j,\phi)$ since B_j is **true** in both states and they differ in only one event. So, by the inductive hypothesis, $\mathbf{wit}(B_j,\phi) = \mathbf{wit}(B_j,\sigma) \subseteq \mathcal{A}_i$. Hence, (7.11) is satisfied with ϕ in place of σ. *Q.E.D.*

Example 7.8: Suppose we want to determine the CSCS of $\alpha = \langle x_5, 1\rangle$ in Example 7.6. The set of all $\langle B_j, \tau\rangle$ such that $\mathbf{enb}(\alpha,\tau) \wedge \mathbf{ctrue}(B_j,\tau)$ is

$$\{\langle B_0, 111000\rangle, \langle B_1, 111100\rangle, \langle B_2, 111110\rangle, \langle B_0, 130210\rangle\}. \qquad (7.20)$$

So, by previous analysis, a CSCS of α is

$$\{\{\langle x_0,1\rangle,\langle x_1,1\rangle,\langle x_2,1\rangle,\langle x_3,0\rangle\},\{\langle x_1,1\rangle,\langle x_2,1\rangle,\langle x_3,1\rangle\},$$
$$\{\langle x_2,1\rangle,\langle x_4,1\rangle\},\{\langle x_0,1\rangle,\langle x_1,3\rangle,\langle x_3,2\rangle\}\}.$$

□

Though Lemma 7.15 makes no claim as to the minimality of (7.19), in practice, as illustrated in this example, a minimal CSCS often results. Note also that, in this example, there are two states where B_0 is critically **true**, though $\mathbf{wit}(B_0, 111000) \neq \mathbf{wit}(B_0, 130210)$. It is also possible that a single disjunct B_j is critically **true** in two different states σ_a and σ_b with $\mathbf{wit}(B_j, \sigma_a) = \mathbf{wit}(B_j, \sigma_b)$. In that case, if $\mathbf{cause}(\alpha, B_j, \sigma_a)$ is a subset of $\mathbf{cause}(\alpha, B_j, \sigma_b)$, then the latter can be removed from the CSCS since (7.11) with $\mathcal{A}_i = \mathbf{cause}(\alpha, B_j, \sigma_b)$ implies (7.11) with $\mathcal{A}_i = \mathbf{cause}(\alpha, B_j, \sigma_a)$. However, as the following example shows, for arbitrary cause sets $\mathcal{A}_a$ and $\mathcal{A}_b$, even if $\mathcal{A}_a \subset \mathcal{A}_b$, both $\mathcal{A}_a$ and $\mathcal{A}_b$ may be needed in the CSCS.

Example 7.9: Consider the following PR set whose state graph is shown in Figure 7.4:

$$\begin{array}{lcl}
\neg x_3 \wedge \neg x_4 & \rightarrow & x_5\uparrow \\
\neg x_3 & \rightarrow & x_2\uparrow \\
\neg x_3 & \rightarrow & x_1\uparrow \\
x_5 & \rightarrow & x_4\uparrow \\
x_4 & \rightarrow & x_5\downarrow \\
\neg x_3 \wedge x_4 \wedge \neg x_5 \vee x_1 \wedge x_2 \wedge x_4 & \rightarrow & x_0\uparrow \\
x_0 \wedge x_4 \wedge x_5 \vee x_1 \wedge x_2 \wedge x_4 & \rightarrow & x_3\uparrow \\
x_3 \wedge \neg x_5 \wedge x_0 & \rightarrow & x_2\downarrow \\
x_3 \wedge \neg x_5 \wedge x_0 & \rightarrow & x_1\downarrow \\
x_3 \wedge \neg x_2 \wedge \neg x_1 \wedge \neg x_5 & \rightarrow & x_4\downarrow \\
x_3 \wedge \neg x_5 \wedge \neg x_4 & \rightarrow & x_0\downarrow \\
\neg x_4 \wedge \neg x_2 \wedge \neg x_1 \wedge \neg x_0 & \rightarrow & x_3\downarrow.
\end{array}$$

The states in bold are those in which $\alpha = \langle x_3, 1\rangle$ is enabled. Let B_0 be $x_0 \wedge x_4 \wedge x_5$ and B_1 be $x_1 \wedge x_2 \wedge x_4$. Since $\mathbf{ctrue}(B_1, 011011)$ and $\mathbf{wit}(B_1, 011011) = \mathbf{wit}(B_1, \mathbf{last}(\alpha))$,

$$\mathcal{A}_a = \mathbf{wit}(B_1, 011011) = \{\langle x_1, 1\rangle, \langle x_2, 1\rangle, \langle x_4, 1\rangle\}$$

is a cause set of α. Also, since $\mathbf{ctrue}(B_0, 111011)$, by Lemma 7.14,

$$\mathcal{A}_b = \{\langle x_0, 1\rangle, \langle x_1, 1\rangle, \langle x_2, 1\rangle, \langle x_4, 1\rangle, \langle x_5, 1\rangle\}$$

is another cause set of α. Note that though $\mathcal{A}_a \subset \mathcal{A}_b$, $\mathcal{A}_b$ is needed since it is the only cause set that satisfies (7.11) for $B_j = B_0$ and $\sigma = 111011$. $\square$

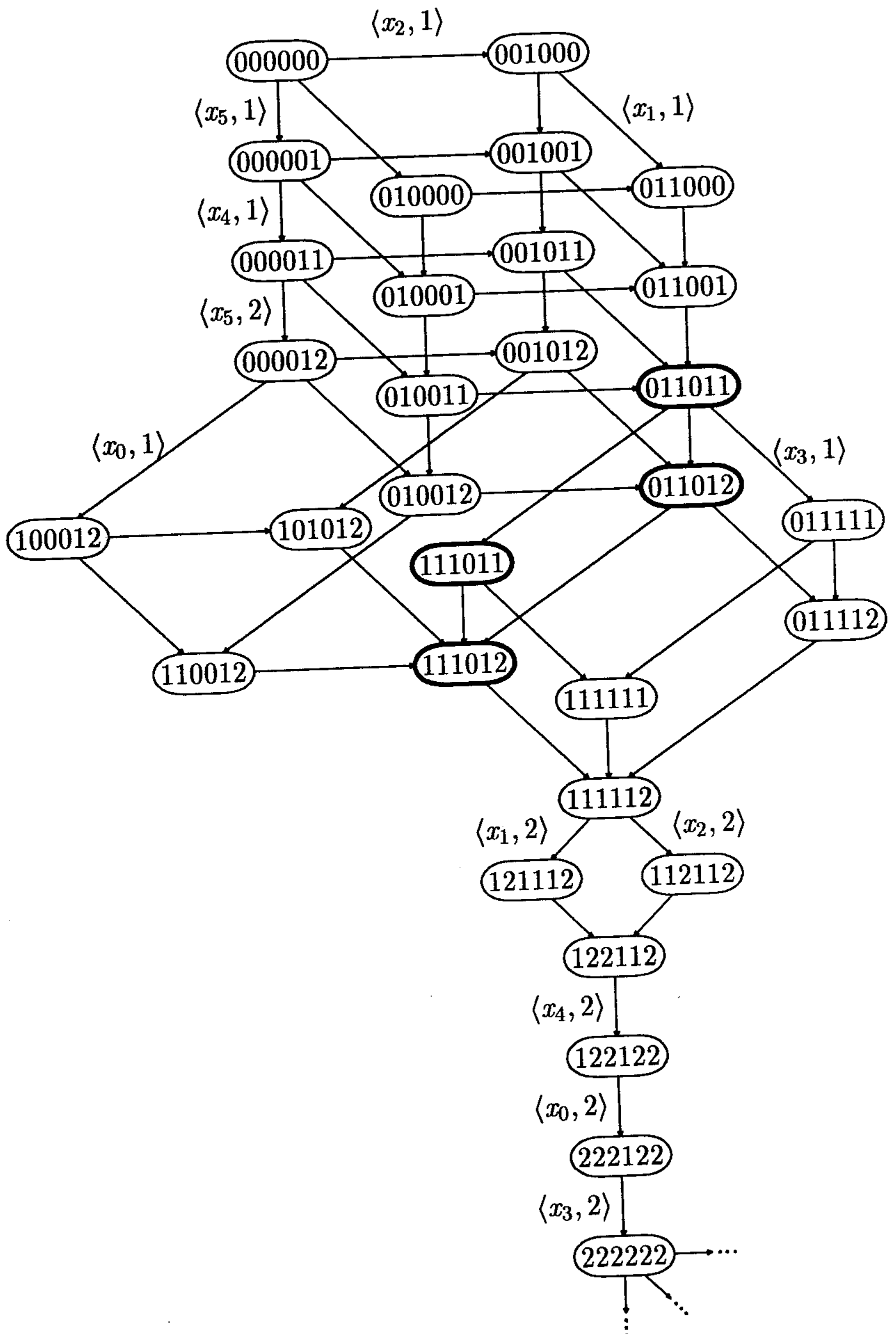

Figure 7.4: Cumulative state graph for Example 7.9

7.5.3 Finding Critically True States

By Lemma 7.15, to find a CSCS for a transition α, it is sufficient to find every disjunct B_j in the guard of $\mathbf{tran}(\alpha)$ and every state τ such that $\mathbf{ctrue}(B_j, \tau)$. The difficulty of this search can be illustrated by once again using the PR set in Example 7.9.

Example 7.10: The following cycle is found when Algorithm 1 is applied to Example 7.9:

$$000000 \xrightarrow{\langle x_5,1\rangle} 000001 \xrightarrow{\langle x_4,1\rangle} 000011 \xrightarrow{\langle x_5,2\rangle}$$
$$000012 \xrightarrow{\langle x_2,1\rangle} 001012 \xrightarrow{\langle x_1,1\rangle} 011012 \xrightarrow{\langle x_3,1\rangle}$$
$$011112 \xrightarrow{\langle x_0,1\rangle} \cdots \xrightarrow{\langle x_0,2\rangle} 222222.$$

At the state 011012 where $\alpha = \langle x_3, 1\rangle$ is enabled, only B_1 is **true**. Suppose, by backtracking from that state, we have determined that B_1 is critically **true** in state $\tau_1 = 011011$. At this point, there is no indication that there exists a state σ such that B_0 is **true**. Even if we had started at $\mathbf{last}(\alpha) = 111012$, to reach $\tau_1 = 011011$, it is possible that the following path is backtracked over:

$$011011 \xrightarrow{\langle x_5,2\rangle} 011012 \xrightarrow{\langle x_0,1\rangle} 111012.$$

Again, no state where B_0 is **true** is visited. □

It is our conjecture that in order not to miss any $\langle B_j, \sigma\rangle$ such that $\mathbf{ctrue}(B_j, \sigma)$ holds, it is necessary, in general, to check *every* state where α is enabled. The obvious penalty for this approach is that, in the worst-case, the number of states needed to be checked is exponential in the number of variables in the PR set. However, typically, the actual number of states where a particular event is enabled is significantly smaller. Moreover, this exhaustive check needs to be applied only for events corresponding to transitions whose PR's have unstable disjuncts.

7.5.4 Conversion to XER-systems

The procedure for determining a CSCS for an event a when the PR of $\mathbf{tran}(\mathrm{a})$ may contain unstable disjunct is *unstab_disj*() on page 164. The procedure is invoked from Algorithm 2 which assumes the minimal cycle (7.13) has been given. As discussed above, backtracking is used to find all states where an event α is enabled. Suppose, for instance, $\mathbf{enb}(\alpha_0 \oplus \pi, \sigma_{n-1})$. Then, the state $(\sigma_{n-1} - \pi)$, which has weight less than σ_0, may not exist if $\sigma_0 = \sigma_{\text{init}}$. Hence, the set of states where α_0 is enabled is smaller than the set of states where $(\alpha_0 \oplus \pi)$ is enabled. So, to determine the periodic behavior of a system, it may be necessary to compute the CSCS of $(\alpha \oplus i\pi)$ for some $i > 0$, instead of α. The following two results guarantees that such an i exists. Note that, as explained in the sub-section on implementation, the constant I introduced in the lemmas is only a proof device and its value needs not be determined.

Lemma 7.16 *Let (7.13) be a minimal cycle with σ_0 a non-transitory state. Then, there exists a constant I such that for any i, j, and σ satisfying $i \geq 0$ and $\mathbf{enb}(\alpha_j \oplus (i+I)\pi, \sigma)$, the following predicate holds:*

$$\forall k : 0 \leq k < K : \sigma[k] \geq \sigma_0[k] + \pi[k]. \tag{7.21}$$

Furthermore, for any state σ and $i \geq 0$ such that $\mathbf{enb}(\alpha_j \oplus (i+I)\pi, \sigma)$, the state $(\sigma - i\pi)$ exists.

Proof: The number of states σ for which (7.21) does not hold is at most $\prod_{k=0}^{K-1}(\sigma_0[k] + \pi[k])$, which is finite. Since $\mathbf{enb}(\alpha_j \oplus i_a\pi, \sigma)$ and $\mathbf{enb}(\alpha_j \oplus i_b\pi, \sigma)$ implies $i_a = i_b$, by the Pigeonhole Principle, there are finite number of i's for which there exists σ satisfying $\mathbf{enb}(\alpha_j \oplus i\pi, \sigma)$ but not (7.21). So, there exists I_j such that $i \geq 0$ and $\mathbf{enb}(\alpha_j \oplus (i + I_j)\pi, \sigma)$ imply (7.21). Setting I to be the maximum of the I_j's establishes the first part of the lemma.

Next, suppose, toward a contradiction, that the second part of the lemma does not hold. Let $\hat{\imath}$ be the smallest integer such that there exist $\hat{\jmath}$ and $\hat{\sigma}$ satisfying $\hat{\imath} \geq 0$, $\mathbf{enb}(\alpha_{\hat{\jmath}} \oplus (\hat{\imath} + I)\pi, \hat{\sigma})$, and the state $(\hat{\sigma} - \hat{\imath}\pi)$ does not exist. Obviously, $\hat{\imath} \neq 0$. By Lemma 5.9, (7.21) implies $(\sigma_0 + \pi) \overset{*}{\rightarrow} \hat{\sigma}$. So, by Lemma 5.16, $\sigma_0 \overset{*}{\rightarrow} (\hat{\sigma} - \pi)$. But $\mathbf{enb}(\alpha_{\hat{\jmath}} \oplus (\hat{\imath} + I)\pi, \hat{\sigma})$ implies $\mathbf{enb}(\alpha_{\hat{\jmath}} \oplus (\hat{\imath} - 1 + I)\pi, \hat{\sigma} - \pi)$ and $(\hat{\sigma} - \hat{\imath}\pi)$ does not exist implies $((\hat{\sigma} - \pi) - (\hat{\imath} - 1)\pi)$ does not exist. So, $\hat{\imath} - 1$ satisfies the conditions required of $\hat{\imath}$ and is also smaller. This contradiction proves the lemma. *Q.E.D.*

Lemma 7.17 *If $\{ \mathcal{A}_0, \mathcal{A}_1, \ldots, \mathcal{A}_{L-1} \}$ is a CSCS of $(\alpha \oplus I\pi)$ as prescribed by Lemma 7.15, then $\{ \mathcal{A}_0 \oplus i\pi, \mathcal{A}_1 \oplus i\pi, \ldots, \mathcal{A}_{L-1} \oplus i\pi \}$ is a CSCS of $(\alpha \oplus (i + I)\pi)$.*

Proof: Let B be any disjunct in the guard of $\mathbf{tran}(\alpha)$. By arguments similar to those in the proof of Corollary 7.11, $\mathbf{last}(\alpha \oplus (i+I)\pi) = (\mathbf{last}(\alpha \oplus I\pi) + i\pi)$ and $\mathbf{wit}(B, \sigma + i\pi) = (\mathbf{wit}(B, \sigma) \oplus i\pi)$. So,

$$\mathbf{cause}(\alpha \oplus (i+I)\pi, B, \sigma + i\pi) = (\mathbf{cause}(\alpha \oplus I\pi, B, \sigma) \oplus i\pi).$$

Therefore, from Lemma 7.14, it remains to show that for any τ and $\hat{\tau}$,

$$\begin{aligned} &\mathbf{enb}(\alpha \oplus I\pi, \tau) \wedge \mathbf{ctrue}(B, \tau) \Rightarrow \\ &\quad \mathbf{enb}(\alpha \oplus (i+I)\pi, \tau + i\pi) \wedge \mathbf{ctrue}(B, \tau + i\pi) \end{aligned} \tag{7.22}$$

and

$$\begin{aligned} &\mathbf{enb}(\alpha \oplus (i+I)\pi, \hat{\tau}) \wedge \mathbf{ctrue}(B, \hat{\tau}) \Rightarrow \\ &\quad \mathbf{enb}(\alpha \oplus I\pi, \hat{\tau} - i\pi) \wedge \mathbf{ctrue}(B, \hat{\tau} - i\pi). \end{aligned} \tag{7.23}$$

If the antecedent of (7.22) holds, then, $\mathbf{enb}(\alpha \oplus (i+I)\pi, \tau + i\pi)$ and B is **true** in $(\tau + i\pi)$ due to Lemma 5.15. Moreover, if there exists ϕ such that $\phi \longrightarrow (\tau + i\pi)$ and B is **true** in ϕ, then, by Lemma 7.16 and Lemma 5.16, state $(\phi - i\pi)$ exists, $(\phi - i\pi) \longrightarrow \tau$, and B is **true** in $\phi - i\pi$. This situation contradicts $\mathbf{ctrue}(B, \tau)$. Thus, B is critically **true** in $(\tau + i\pi)$ and (7.22) is established.

Conversely, let $\hat{\tau}$ be any state such that $\mathbf{enb}(\alpha \oplus (i+I)\pi, \hat{\tau})$ and $\mathbf{ctrue}(B, \hat{\tau})$. Then, by Lemma 7.16, $(\hat{\tau} - i\pi)$ is a state and, by Lemma 5.15, $\mathbf{enb}(\alpha \oplus I\pi, \hat{\tau} - i\pi)$ and B is **true** in $(\hat{\tau} - i\pi)$. Moreover, if B is not critically **true** in $(\hat{\tau} - i\pi)$ then B is not critically **true** in $\hat{\tau}$ due to Lemma 5.16. Hence, $\mathbf{ctrue}(B, \hat{\tau} - i\pi)$ and (7.23) is verified. *Q.E.D.*

We are now ready to prove the correctness of Algorithm 2 in the general case where there are unstable disjuncts. The algorithm makes use of *unstab_disj_sub*() to recursively visit all states σ where a particular event $\mathtt{a0}$ is enabled. As described in Sub-section 7.5.1, *find_incoming*($\mathtt{s}$) returns $\mathbf{incoming}(\mathtt{s})$. Also, for any $\mathtt{b}$ in $\mathbf{incoming}(\mathtt{s})$, *prev_state*($\mathtt{s}$,$\mathtt{b}$) returns the state τ such that $\tau \xrightarrow{\mathtt{b}} \mathtt{s}$.

Lemma 7.18 *Let $\mathcal{X}'$ be the repetitive XER-system generated by Algorithm 2. Let $\mathcal{X}$ be the general XER-system induced by $\mathcal{X}'$. Then, there exists a constant I_{max} such that for all $i \geq I_{max}$, if the XER-system event $\langle u, i\rangle$ represents the event α, then the set of all cause sets of $\langle u, i\rangle$ in $\mathcal{X}$ represents a CSCS of α.*

Proof: Let α_j be an event in (7.13). If the guard of $\mathbf{tran}(\alpha_j)$ is known to have only stable disjuncts, then, by Lemma 7.12, the result holds for any XER-system event $\langle u(\alpha_j), i\rangle$ provided $I_{\max} \geq \theta_{\max}$.

So, it suffices to analyze the case when *unstab_disj*$(\alpha_{\mathtt{i}}, \mathtt{s})$ is invoked. The purpose of that procedure is to determine a CSCS for $\alpha = (\alpha_{\mathtt{i}} \oplus I\pi)$ according to Lemma 7.15 and Lemma 7.14. First, $\mathbf{last}(\alpha)$ is computed and assigned to s0 by using the identity $\mathbf{last}(\alpha_{\mathtt{i}} \oplus I\pi) = (\mathbf{last}(\alpha_{\mathtt{i}}) + I\pi)$. Then, after setting a0, U0, and V, *unstab_disj_sub*(V, a0, s0) is called.

Next, consider an instantiation of *unstab_disj_sub*(U, a0, s). From the topology of the procedures, the event a0 is enabled in state s. Also, from the definition of *find_incoming*(), t assumes every value τ such that $\tau \longrightarrow \mathtt{s}$. So, there is a call of *unstab_disj_sub*(V, a0, t) whenever there is a call of *unstab_disj_sub*(U, a0, s) provided $\mathtt{t} \longrightarrow \mathtt{s}$ and $\mathbf{enb}(\mathtt{a0}, \mathtt{t})$. Applying this observation recursively implies that there is a call of *unstab_disj_sub*$(\mathtt{U}, \mathtt{a0}, \sigma)$ whenever $\mathbf{enb}(\mathtt{a0}, \sigma)$ and $\sigma \overset{*}{\longrightarrow} \mathtt{s0}$.

Next, let the guard of $\mathbf{tran}(\alpha)$ be $B_0 \vee B_1 \vee \ldots \vee B_M$ and consider the operations executed by *unstab_disj_sub*(U,a0,s). Before the first loop in the procedure, $\mathtt{U} = \{B_j : B_j \text{ is } \mathbf{true} \text{ in } \mathtt{s} : \mathbf{wit}(B_j, \mathtt{s})\}$. Now, $\mathbf{wit}(B_j, \mathtt{s})$ is removed from U if and only if there exists an assignment to t and a disjunct $B_{\hat{j}}$ such that

$$\mathbf{enb}(\mathtt{a0}, \mathtt{t}) \wedge \mathtt{t} \longrightarrow \mathtt{s} \wedge \mathbf{wit}(B_{\hat{j}}, \mathtt{s}) = \mathbf{wit}(B_j, \mathtt{t}). \tag{7.24}$$

Since the guard is in DNF, the last conjunct above implies $\hat{j} = j$. Hence, at the end of the first loop in *unstab_disj_sub*(), $\mathtt{U} = \{B_j : \mathbf{ctrue}(B_j, \mathtt{s}) : \mathbf{wit}(B_j, \mathtt{s})\}$.

The second loop in *unstab_disj_sub*() computes $\mathbf{cause}(\mathtt{a0}, B_j, \mathtt{s})$ according to Lemma 7.14. For each B_j such that $\mathbf{ctrue}(B_j, \mathtt{s})$, *gen_template*(Z, a0) is called with Z set equal to $\mathbf{cause}(\mathtt{a0}, B_j, \mathtt{s})$. Let $\mathcal{W}$ be the set of $\mathcal{C}$'s such that there is a call of *gen_template*(Z, a0) with $\mathtt{Z} = \mathcal{C}$ during the execution of the algorithm. Since all states σ such that $\mathbf{enb}(\mathtt{a0}, \sigma)$ and all disjuncts B_j such that $\mathbf{ctrue}(B_j, \sigma)$ are included, by Lemma 7.15, $\mathcal{W}$ is a CSCS of a0. Consequently, using arguments similar to those used in the proof of Lemma 7.12 and the periodic behavior implied by Lemma 7.17, it can then be shown that for all $i + I \geq \theta_{\max}$, the set of all cause sets of $\langle u, i + I\rangle$ represents a CSCS of $(\alpha_j \oplus (i + I)\pi)$ whenever $u = u(\alpha_j)$. *Q.E.D.*

7.5.5 Implementation Issues

In Algorithm 2, there are several implementation issues that need to be addressed. The most important one is that the identity of I does not need to be determined beforehand. Instead, assume an implicit offset of $I\pi$ has been added to all states and events. Thus, the cycle (7.13), when used by the algorithm, actually represents

$$(\sigma_0 + I\pi) \xrightarrow{\alpha_0 \oplus I\pi} (\sigma_1 + I\pi) \xrightarrow{\alpha_1 \oplus I\pi} \cdots \xrightarrow{\alpha_{n-1} \oplus I\pi} (\sigma_n + I\pi). \tag{7.25}$$

Also, the addition and extension by $I\pi$ in *unstab_disj*() and the check for $(\mathtt{s}[k] \neq \sigma_{\text{init}}[k])$ in *find_incoming*() are removed. With these modifications, a state may now have negative components in the program. However, through the use of the implicit offset, the value of I can be assumed to be large enough so that all of the states are reachable from σ_{init}. Similar arguments apply to events with negative occurrence numbers in the algorithm. Since the same XER-system is generated as before, the explicit references to I can be removed from Algorithm 2 without affecting its correctness.

Next, observe that in *unstab_disj_sub*(), we are only interested in finding β's satisfying

$$(\exists \tau :: \tau \xrightarrow{\beta} \mathtt{s}) \wedge (\tau \xrightarrow{\beta} \mathtt{s} \Rightarrow \mathbf{enb}(\mathsf{a0}, \tau)). \tag{7.26}$$

The first condition is guaranteed by *find_incoming*() returning $\mathbf{incoming}(\mathtt{s})$ and the second condition is checked explicitly in *unstab_disj_sub*(). Note, however, that the second condition can be checked statically. Hence, it should be checked first so as to reduce the number of candidates for which the first condition needs to be verified. This optimization can be realized by modifying *find_incoming*() so that it accepts an additional parameter a0 and removes from I every event β that does not satisfy the the second condition in (7.26). The removal should be done at the beginning of the foreach-loop for maximum efficiency.

7.6 Complexity

We have presented Algorithm 2 for converting a non-separable PR set with a given minimal cycle into a repetitive XER-system $\mathcal{X}'$. For each event α, if its guard is conjunctive or mutex, then its CSCS can be computed immediately. Else, if the PR for **tran**(α) contains only stable disjuncts, then one needs to trace a path to find **last**(α). Since α is enabled in every state along that path, there appears to be a limit on how many states there are in that path. In fact, from experience, we believe that the actual length of the path cannot be greater than the number of events in the cycle though currently no proof exists. Assuming the bound is correct, then, if the PR set does not contain unstable disjuncts, the number of states visited by Algorithm 2 is at most quadratic in the number of events in the cycle.

The situation can worsen considerably if there are PR's with unstable disjuncts. For an event, α, whose transition involves such a PR, every state where α is enabled needs to be visited. In the worst case, this number can be exponential in the number of variables though, typically, it is much less. Also, in practice, PR's with unstable disjuncts are rare. Thus, Algorithm 2 provides a simple way to convert PR sets into repetitive XER-systems.

Chapter 8

Conclusion

8.1 Summary

In this thesis, we have addressed the problems involved in the timing analysis of asynchronous VLSI circuits. We have presented examples of practical circuits (quick-decision zero-checkers) that are inherently disjunctive and use them as our motivation for generalizing Burns' ER-systems. We have verified that an extended ER-system (XER-system) retains many properties of the original and shown how its period can be computed. The main result on XER-systems is Theorem 4.2 which states that this period is a good indication of the steady-state performance of a repetitive XER-system.

We have also considered the issues involved in determining the periodic behavior of a data-dependent circuit. Using cumulative state graphs and indexed events, we have developed an abstraction for studying the states in the execution of an asynchronous circuit. With this abstraction, we are able to establish some important properties (Theorem 5.1 and Theorem 5.2) concerning the periodic behavior of such a circuit. Subsequently, we have presented a simple algorithm, index-priority simulation, for finding all minimal periods in the state graph of an asynchronous circuit. We have also shown that the computation performed by the algorithm can be further reduced if the graph is known to be uniform; therefore, sufficient criteria for uniform graphs have been given.

The representation of an asynchronous circuit, once its minimal periods have been determined, as an XER-system is the topic of Chapter 7. There, we have argued that our definition of the cause sets of an event corresponds to the assumption on delay-insensitivity. Then, we have presented an algorithm to systematically extract the causality relationships from the circuit and model them in an XER-system. A distinction is made between circuits that have only stable disjuncts and those that do not since the computation required for the former is much less than that for the latter.

To summarize, index-priority simulation is used to model a circuit as an XER-system, whose period can be analytically computed and accurately indicates the speed of the circuit. Thus, we have developed a systematic approach to evaluating and optimizing the performance of asynchronous VLSI circuits, even those that are data-dependent and inherently disjunctive. The approach is efficient for many practical circuits and, we believe, serves as a good framework for future work in this area. Furthermore, many of the results can be applied to other concurrent systems. In particular, index-priority simulation is a simple and efficient way for finding minimal cycles in the state graphs of these systems.

8.2 Future Work

In this section, we list some possible areas of further research. First, we are currently investigating how the results of this thesis can be applied to the analysis of other metrics of a circuit, such as energy, latency, power-delay product, etc. Moreover, besides transistor sizing, we plan to incorporate other methods of speeding up a circuit — for instance, adding inverters to the circuit or reordering the transistors within an element — into the performance optimization procedure. Also, a systematic way to choose "typical" environmental scenarios and to combine results from them would be very useful.

In regard to XER-systems, finding more efficient ways to compute their periods would be very beneficial when there are many transitions that are disjunctively caused. The proof of Theorem 4.2 is very long and a more direct proof may be possible. Furthermore, in order for (4.75) in that theorem to hold, we believe requiring every transition vertex to be reachable from a critical path in a critical scenario is both necessary and sufficient; however, this conjecture remains to be verified.

The conditions derived in Section 6.3 are sufficient but not necessary for a graph to be uniform; for the sake of completeness, a complete characterization would be desirable. Also, the bound given on the number of states visited in Algorithm 2 when there are only stable disjuncts is empirical; a rigorous proof would be more satisfactory. Furthermore, it would be beneficial to have a definite answer to the complexity involved in finding the CSCS for a transition whose guard has unstable disjuncts.

Finally, though the definitions of causes given in Section 7.2 are valid if the delays between transitions are arbitrary, it is possible to come up with alternative definitions that result in smaller XER-systems if one makes some delay assumptions on the timing model being used. Some preliminary work on using only triggers, as defined in Sub-section 6.3.1, as causes has already been done and has shown promising results.

Appendix A

Algorithms

The following algorithms are written in a language that is a slight variation to Pidgin ALGOL of [1]. For clarity, we have adopted the following conventions: names for storage variables and types are in `typewriter` font, words reserved by the language are in sans serif font, procedure names are in *slanted* font, and theoretical expressions and descriptive proses are written in their standard formats. We have used the construct "foreach variable s.t. condition do statement," instead of the standard for loop, when the order in which the loop variable assumes its range of possible values is not important. Also, the instantiation operator ("such that") is denoted by $\ni$; e.g., "$i \ni i + 5 = 6$" returns the number 1. Finally, procedure calls are by reference and all program variables are assumed to be global except when masked by the formal parameters of a procedure or by its local variables.

A.1 Algorithm 1

Input: A stable PR set $\mathcal{P}$ with variables $x_0, x_1, \ldots, x_{K-1}$ and an initial state σ_{init}.

Output: A non-transitory state S[n] and an array of cycles C[ ] such that if $\mathtt{C}[i] = \sigma_i \twoheadrightarrow (\sigma_i + \pi_i)$ for $0 \le i < \mathtt{p}$, then (6.18) and (6.19) are satisfied.

```
Algorithm index-priority_simulation()
  begin
   Cycle c;
   Array_of_Cycles C[ ];
   Array_of_States S[ ];
   Array_of_Events A[ ];
   Set_of_Variables U;
   Set_of_Variables V;
   Integer n, p;

   p ← 0;
   V ← ∅;
   S[0] ← σ_init;
   n ← 0;
   repeat
      begin
         c ← find_cycle(U);
         if (c ≠ empty_cycle) then
            begin
               C[p] ← c;
               V ← V ∪ U;
               p ← p + 1
            end
      end
   until (c = empty_cycle);
   return S[n] and C
  end
```

```
procedure find_cycle(Set_of_Variables U)
  begin
    Set_of_Events E;
    State s;
    Integer k, i, m;

    E ← {α : enb(α, S[n]) ∧ var(α) ∉ V : α};
    while (E ≠ ∅) do
      begin
        k ← max{κ, λ : ⟨x_κ, λ⟩ ∈ E : κ};
        A[n] ← ⟨x_κ, λ⟩ ∋ ⟨x_κ, λ⟩ ∈ E ∧ κ = k;
        s ← next_state(S[n], A[n]);
        if (∃ĩ : 0 ≤ ĩ ≤ n : bool(S[ĩ]) = bool(s)) then
          begin
            i ← ĩ ∋ bool(S[ĩ]) = bool(s);
            U ← {ĩ : i ≤ ĩ ≤ n : var(A[ĩ])};
            m ← n;
            n ← i;
            return S[i] --A[i]--> S[i + 1] --A[i + 1]--> ... --A[m]--> s
          end
        else
          begin
            n ← n + 1;
            S[n] ← s;
            E ← {α : enb(α, s) ∧ var(α) ∉ V : α}
          end
      end;
    return empty_cycle
  end
```

A.2 Algorithm 2

Input: A non-separable PR set $\mathcal{P}$ with set of events $\mathcal{E}(\mathcal{P})$; a minimal cycle $\sigma_0 \xrightarrow{\alpha_0} \sigma_1 \xrightarrow{\alpha_1} \cdots \xrightarrow{\alpha_{n-1}} \sigma_n$ in the state graph of $\mathcal{P}$ with σ_0 a non-transitory state, π the period of the cycle, and $\mathcal{A} = \{i : 0 \leq i < n : \alpha_i\}$; the constant I as prescribed by Lemma 7.16; and the transition set $E' = \{i : 0 \leq i < n : u(\alpha_i)\}$ for the repetitive XER-system $\mathcal{X}' = \langle E', R', \delta, \theta \rangle$ which is to model $\mathcal{P}$.

Output: The template set R' and occurrence-index offset function θ for $\mathcal{X}'$.

```
Algorithm generate_template_set()
  begin
   Integer i;
   Set_of_Templates R;
   Set_of_(Transition,Transition,Template,Integer) T;

   R ← ∅;
   T ← ∅;
   i ← 0;
   while (i < n) do
       begin
           if (the guard of tran(αi) is conjunctive) then
               gen_template(wit(the guard of tran(αi), σi), αi)
           else if (the guard of tran(αi) has only stable disjuncts) then
               stab_disj(αi, σi)
           else
               unstab_disj(αi, σi);
           i ← i+1
       end;
   return R and T
  end
```

```
procedure stab_disj(Event a, State s)
  begin
   State t;
   Disjunct B;

   if (the disjuncts in the guard of tran(a) are mutex) then
       begin
           B ← B ∋ B is a disjunct in the guard of tran(a) ∧ B is true in s;
           gen_template(wit(B, s), a)
       end
   else
       begin
           t ← fire_only(ℰ(𝒫) \ {a}, s);
           foreach B s.t. B is a disjunct in the guard of tran(a) ∧
                                                        B is true in t do
               gen_template(wit(B, t), a)
       end
  end

procedure fire_only(Set_of_Events D, State s)
  begin
   Set_of_Events E;
   Event b;
   State t;

   t ← s;
   E ← {β : enb(β, t) ∧ β ∈ D : β};
   while (E ≠ ∅) do
       begin
           b ← β ∋ β ∈ E;
           t ← next_state(t, b);
           E ← {β : enb(β, t) ∧ β ∈ D : β}
       end;
   return t
  end
```

```
procedure gen_template(Set_of_Events Z, Event a)
  begin
    Integer l, e;
    Event b;
    Variable x;
    Transition u;
    Set_of_Transitions C;
    Set_of_(Transition,Integer) F;

    C ← ∅;
    F ← ∅;
    foreach ⟨x,l⟩ s.t. ⟨x,l⟩ ∈ Z do
        begin
            e ← ε ∋ (⟨x,l⟩ ⊕ επ) ∈ 𝒜;
            b ← ⟨x,l⟩ ⊕ eπ;
            C ← C ∪ {u(b)};
            F ← F ∪ {(u(b),e)}
        end;
    R ← R ∪ {C↦u(a)};
    foreach (u,e) s.t. (u,e) ∈ F do
        T ← T ∪ {(u,u(a),C↦u(a),e)}
  end

procedure unstab_disj(Event a, State s)
  begin
    Set_of_Sets_of_Events U0,V;
    Event a0;
    State s0;

    s0 ← fire_only(ℰ(𝒫) \ {a}, s) + Iπ;
    a0 ← a ⊕ Iπ;
    U0 ← {B : B is a disjunct in the guard of tran(a) ∧
                                    B is true in s0 : wit(B, s0)};
    V ← U0;
    unstab_disj_sub(V, a0, s0)
  end
```

```
procedure unstab_disj_sub( Set_of_Sets_of_Events U, Event a0, State s)
  begin
   Set_of_Sets_of_Events V;
   Set_of_Events I, W, Y;
   State t;
   Event b;

   I ← find_incoming(s);
   foreach b s.t. b ∈ I do
      begin
         t ← prev_state(s, b);
         if (enb(a0, t)) then
            begin
               V ← {B : B is a disjunct in the guard of tran(a0) ∧
                                          B_j is true in t : wit(B_j, t)};
               U ← U \ V;
               unstab_disj_sub(V, a0, t)
            end
      end;

   foreach W s.t. W ∈ U do
      if (W ∈ U0) then
         gen_template(W, a0)
      else
         begin
            Y ← W;
            foreach b s.t. b ∈ (wit(guard of tran(a0), s) \ W) do
               begin
                  t ← fire_only(E(P) \ {b}, s);
                  if (W has occurred in t) then
                     Y ← Y ∪ {b}
               end;
            gen_template(Y, a0)
         end
 end
```

```
procedure find_incoming(State s)
  begin
   Set_of_Events I,E,D;
   State p, t;
   Event b;

   E ← {β : guard of tran(β) is true in s : β};
   I ← {x_k, l : ⟨x_k, l⟩ ∈ E ∧ s[k] = l ∧ s[k] ≠ σ_init[k] : ⟨x_k, l⟩};
   foreach b s.t. b ∈ I do
       begin
           D ← 𝒟 ∋ ∃φ :: (φ -𝒟->* s) ∧ (b ∈ 𝒟);
           p ← φ ∋ φ -D->* s;
           t ← fire_only(D \ {b}, p);
           if (¬(t -b-> s)) then
               I ← I \ {b}
       end;
   return I
  end
```

Bibliography

[1] A.V. Aho, J.E. Hopcroft, and J.D. Ullman. *The Design and Analysis of Computer Algorithms*. Addison-Wesley Publishing Company, Reading, MA, 1974.

[2] J.M. Berger. A note on error detection codes for asymmetric channels. *Information and Control*, 4:68-73, 1961.

[3] G. Birkhoff. *Lattice Theory*. American Mathematical Society, Providence, RI, 1940.

[4] S.S. Bizzan, G.A. Jullien and W.C. Miller. Analytical approach to sizing nFET chains. *Electronics Letters*, 28(14):1334-1335, 1992.

[5] S.M. Burns. *Automated Compilation of Concurrent Programs into Self-timed Circuits*. M.S. thesis, CS-TR-88-2, California Institute of Technology, 1988.

[6] S.M. Burns. *Performance Analysis and Optimization of Asynchronous Circuits*. Ph.D. thesis, California Institute of Technology, 1991.

[7] J.-P. Caisso, E. Cerny, and N.C. Rumin. A recursive technique for computing delays in series-parallel MOS transistor circuits. *IEEE Transactions on Computer-Aided Design*, 10(5):589-595, 1991.

[8] P.K. Chan and K. Karplus. Computing signal delay in general RC networks by tree/link partitioning. *IEEE Transactions on Computer-Aided Design*, 9(8):898-902, 1990.

[9] C.-Y. Chu and M.A. Horowitz. Charge-sharing models for switch-level simulation. *IEEE Transactions on Computer-Aided Design*, CAD-6(6):1053-1061, 1987.

[10] T.A. Chu. *Synthesis of Self-Timed VLSI Circuits from Graph-Theoretic Specifications*. Ph.D. Thesis, Massachusetts Institute of Technology, 1987.

[11] E.W. Dijkstra. *A Discipline of Programming*. Prentice-Hall, Englewood Cliffs, NJ, 1976.

[12] J.C. Ebergen. *Translating Programs into Delay-Insensitive Circuits*, Ph.D. thesis, Technische Universiteit Eindhoven, 1987.

[13] W.C. Elmore. The transient response of damped linear networks with particular regard to wideband amplifiers. *Journal of Applied Physics*, 19(1):55-63, January 1948.

[14] J. Franklin. *Methods of Mathematical Economics*. Springer-Verlag, New York, NY, 1980.

[15] J. Gunawardena. Timing analysis of digital circuits and the theory of min-max functions. In *TAU'93, ACM International Workshop on Timing Issues in the Specification and Synthesis of Digital Systems*, 1993.

[16] P.J. Hazewindus. *Testing Delay-Insensitive Circuits*. Ph.D. thesis, California Institute of Technology, 1992.

[17] C.A.R. Hoare. Communicating sequential processes. *Communications of the ACM*, 21(8):666-677, 1978.

[18] W. Keister, A.E. Ritchie, and S.H. Washburn. *The Design of Switching Circuits*, D. Van Nostrand, Princeton, NJ, 1951.

[19] J.C. Lagarias. The $3x+1$ problem and its generalizations. *The American Mathematical Monthly* 92:3-23, 1985.

[20] C.E. Leiserson, F.M. Rose, and J.B. Saxe. Optimizing synchronous circuitry by retiming. In R. Bryant, editor, *Third Caltech Conference on Very Large Scale Integration*, pp. 87-116, Computer Science Press, Rockville, MD 1983.

[21] T.-M. Lin and C.A. Mead. Signal delay in general RC networks. *IEEE Transactions on Computer-Aided Design*, CAD-3(4):331-349, 1984.

[22] P.C. McGeer and R.K. Brayton. Provably correct critical paths. In C.L. Seitz, editor, *Advanced Research in VLSI: Proceedings of the Decennial Caltech Conference on VLSI*, pp. 119-142, MIT Press, Cambridge, MA, 1989.

[23] D.P. Marple and A. El Gamal. Optimal selection of transistor sizes in digital VLSI circuits. In P. Losleben, editor, *Advanced Research in VLSI, Proceedings of the 1987 Stanford Conference*, pp. 151-172. MIT Press, Cambridge, MA, 1987.

[24] A.J. Martin. Asynchronous datapaths and the design of an asynchronous adder. In *Formal Methods in System Design*, Kluwer, 1992.

[25] A.J. Martin. An axiomatic definition of synchronization primitives. *Acta Informatica* 16:219-235, 1981.

[26] A.J. Martin. The design of a delay-insensitive microprocessor: An example of circuit synthesis by program transformation. In M. Leeser and G. Brown, editors, *Hardware Specification, Verification and Synthesis: Mathematical Aspects*, vol. 408 of *Lecture Notes in Computer Science*, pp. 244-259, Springer-Verlag, 1989.

[27] A.J. Martin. The design of a self-timed circuit for distributed mutual exclusion. In H. Fuchs, editor, *1985 Chapel Hill Conference on VLSI*, pp. 247-260, Computer Science Press, Rockville, MD, 1985.

[28] A.J. Martin. The limitation to delay-insensitivity in asynchronous circuits. In W.J. Dally, editor, *Advanced Research in VLSI: Proceedings of the Sixth MIT Conference*, pp. 263-278, MIT Press, Cambridge, MA, 1990.

[29] A.J. Martin. The probe — an addition to communication primitives. *Information Processing Letters*, 20(3):125-130, 1985.

[30] A.J. Martin. Programming in VLSI: From communicating processes to delay-insensitive circuits. In C.A.R. Hoare, editor, *UT Year of Programming Institute on Concurrent Programming*, Addison-Wesley, Reading, MA 1990.

[31] A.J. Martin. Tomorrow's digital hardware will be asynchronous and verified. In J. van Leeuwen, editor, *Algorithms, Software, Architecture, Information Processing 92, Vol. I*, Elsevier Science Publishers B.V., North-Holland, 1992.

[32] A.J. Martin, S.M. Burns, T.K. Lee, D. Borković, and P.J. Hazewindus. The design of an asynchronous microprocessor. In C.L. Seitz, editor, *Advanced Research in VLSI: Proceedings of the Decennial Caltech Conference on VLSI*, pp. 351-373, MIT Press, Cambridge, MA, 1989.

[33] A.J. Martin, S.M. Burns, T.K. Lee, D. Borković, and P.J. Hazewindus. The first asynchronous microprocessor: The test results. Technical Report, CS-TR-86-6, California Institute of Technology, 1989.

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

[35] T.H.-Y. Meng, R.W. Brodersen, and D.G. Messerschmitt. Automatic synthesis of asynchronous circuits from high-level specifications. *IEEE Transactions on Computer-Aided Design of Integrated Circuits*, 8(11):1185-1205, November 1989.

[36] R.E. Miller. *Switching Theory, Vol. 2*. Wiley, New York, NY, 1965.

[37] D.E. Muller and W.S. Bartky. A theory of asynchronous circuits. In *The Annals of the Computation Laboratory of Harvard University. Volume XXIX: Proceedings of an International Symposium on the Theory of Switching, Part I.*, pp. 204-243, 1959.

[38] C.J. Myers and A.J. Martin. The design of an asynchronous memory management unit. Technical Report, CS-TR-93-30, California Institute of Technology, 1994.

[39] C.J. Myers and T. H.-Y. Meng. Synthesis of timed asynchronous circuits. *IEEE International Conference on Computer Design, ICCD-1992*, 1992.

[40] C.D. Nielsen and A.J. Martin. Design of a delay-insensitive multiply-accumulate unit. *Integration — The VLSI Journal*, 15(3):291-311, 1993.

[41] C.L. Seitz. System Timing. Chapter 7 in Carver Mead and Lynn Conway, *Introduction to VLSI Systems*, Addison-Wesley, Reading MA, 1980.

[42] J.A. Tierno. *An Energy Complexity Model for VLSI Computations.* Ph.D. thesis, California Institute of Technology, 1995.

[43] J.A. Tierno, A.J. Martin, D. Borković, and T.K. Lee. A 100-MIPS GaAs asynchronous microprocessor. *IEEE Design & Test of Computers*, 11(2):43-49, 1994.

[44] S.H. Unger. *Asynchronous Sequential Switching Circuits*. Wiley-Interscience, New York, NY, 1969.

[45] K. van Berkel. *Handshake Circuits: An Asynchronous Architecture for VLSI Programming.* International Series on Parallel Computation 5, Cambridge University Press, Cambridge, England, 1993.

[46] T. Verhoeff. Delay-insensitive codes: An overview. Technical Report 87/04, Department of Mathematics and Computing Science, Eindhoven Institute of Technology, 1987.