TU/e EINDHOVEN UNIVERSITY OF TECHNOLOGY

Characterizations of delay-insensitive communication protocols

Citation for published version (APA):
Verhoeff, T. (1989). *Characterizations of delay-insensitive communication protocols*. (Computing science notes; Vol. 8906). Technische Universiteit Eindhoven.

Document status and date:
Published: 01/01/1989

Document Version:
Publisher's PDF, also known as Version of Record (includes final page, issue and volume numbers)

Please check the document version of this publication:

• A submitted manuscript is the version of the article upon submission and before peer-review. There can be important differences between the submitted version and the official published version of record. People interested in the research are advised to contact the author for the final version of the publication, or visit the DOI to the publisher's website.
• The final author version and the galley proof are versions of the publication after peer review.
• The final published version features the final layout of the paper including the volume, issue and page numbers.

Link to publication

General rights
Copyright and moral rights for the publications made accessible in the public portal are retained by the authors and/or other copyright owners and it is a condition of accessing publications that users recognise and abide by the legal requirements associated with these rights.

• Users may download and print one copy of any publication from the public portal for the purpose of private study or research.
• You may not further distribute the material or use it for any profit-making activity or commercial gain
• You may freely distribute the URL identifying the publication in the public portal.

If the publication is distributed under the terms of Article 25fa of the Dutch Copyright Act, indicated by the "Taverne" license above, please follow below link for the End User Agreement:

www.tue.nl/taverne

Take down policy
If you believe that this document breaches copyright please contact us at:

openaccess@tue.nl

providing details and we will investigate your claim.

Download date: 24. Aug. 2022

Characterizations of Delay-Insensitive Communication Protocols

by

Tom Verhoeff

89/6

May, 1989

COMPUTING SCIENCE NOTES

This is a series of notes of the Computing Science Section of the Department of Mathematics and Computing Science Eindhoven University of Technology.
Since many of these notes are preliminary versions or may be published elsewhere, they have a limited distribution only and are not for review.
Copies of these notes are available from the author or the editor.

Eindhoven University of Technology
Department of Mathematics and Computing Science
P.O. Box 513
5600 MB EINDHOVEN
The Netherlands
All rights reserved
Editors: prof.dr.M.Rem
prof.dr.K.M. van Hee

Characterizations of Delay-Insensitive Communication Protocols

Tom Verhoeff

Department of Mathematics and Computing Science
Eindhoven University of Technology
P.O. Box 513
NL–5600 MB EINDHOVEN
The Netherlands
wstomv@eutws1.uucp
wsintom@heitue5.bitnet

May 1989

Abstract

This paper studies protocols for asynchronous communication over an interface consisting of unidirectional channels. In asynchronous communication, the sender can initiate a transmission without cooperation of the receiver. Contrasting with traditional data-flow networks, the channels that we consider do not synchronize at the receiving end: messages, once on their way, are delivered regardless of the readiness of the receiver to accept them. The situation where a message is delivered to an unready receiver is called computation interference. A protocol is said to be delay-insensitive when it can be guaranteed—without making assumptions about propagation delays—that computation interference cannot occur. We give several characterizations of delay-insensitive protocols and a new proof for the Fundamental Characterization Theorem. The emphasis is on the mathematical treatment of the concepts involved.

0 Introduction and Overview

We begin by giving a physical motivation for our investigation. Consider a digital circuit connected to its environment by an interface consisting of conducting wires. In the digital mode of operation, circuit and environment communicate by exchanging discrete voltage transitions. A voltage transition, once initiated, propagates along a wire to the receiver. The receiving end, however, need not always be ready to process an incoming transition.

The reason for this is that the incoming transition may violate the assumptions underlying the digital mode of operation.

For example, an *OR*-gate in the stable state with one input low, the other input high, and—consequently—the output also high, cannot properly process transitions on both inputs "simultaneously". The best that can happen when both inputs change is that the output remains high or produces a pair of proper transitions (high to low and back to high). But it is also possible that a *non-digital* pulse ensues[0]. What actually happens when both inputs change depends intricately on the (relative) timing of the transitions and the physical structure of the circuitry.

We are interested in communication protocols that guarantee adherence to the digital mode assumptions even when no assumptions are made about the propagation delays incurred in the connecting wires. Such protocols are called delay-insensitive. In order to define and investigate such protocols we introduce a formal model for asynchronous two-party communication. The operational semantics is given in terms of a transition system. The model includes the possibility to specify under what circumstances a party is ready to accept which messages. Our model formalizes the Foam Rubber Wrapper Postulate put forward in [3] as an approach to define delay-insensitivity. We are aware of the gap between circuit physics on the one hand and transition-system semantics on the other. It is not the validity of this type of semantics for circuits that we wish to question here.

In Section 1 we present the formal communication model, define our notion of delay-insensitivity, and state the Fundamental Characterization Theorem. Sections 2, 3, and 4 introduce auxiliary concepts and prove characterizations based on these concepts. Together they constitute a new proof for the Fundamental Characterization Theorem. Finally, Section 5 summarizes the results and mentions some relationships with other work.

1 Delay-Insensitivity and the JTU-Rules

We start with the introduction of some terminology and notations. The two communicating parties are referred to as *Module* and *Environment*. Let I and O be disjoint sets of *symbols*, identifying the channels in the interface. The *direction* of symbols in I and O is said to be *input* and *output* respectively. The directions are to be interpreted with respect to Module. The sets I and O are fixed for the remainder of the paper. We denote their union by A. Variables a through d range over A.

A *trace* is a member of A^*, i.e. a finite-length sequence of symbols from A. It records a communication history at one side of the interface. The contents of the messages communicated is irrelevant for our problem. Hence, there is only the need to record the occurrence of a communication action, for which we employ the symbol identifying the channel involved in that communication action. Variables s through z range over A^*. The *empty trace* is denoted by ε and *concatenation* of traces is denoted by juxtaposition. Trace t is a *prefix* of trace s when $(\exists\, u :: tu = s)$. Subset T of A^* is called *prefix-closed* when

[0]In fact, the pair of proper transitions may also degrade into a non-digital pulse when propagated along the output wire, if no special precautions are taken.

$$(\forall\, s,t : s \in T \;\wedge\; t \text{ prefix of } s : t \in T).$$

The *length* of trace t is denoted by $\ell(t)$. The *symbol bag* of trace t is denoted by $t\#$, that is, $t\#$ is a mapping from A into the natural numbers such that $t\#a$ (the result of applying $t\#$ to a) is the *number of occurrences* of symbol a in trace t.

A *protocol specification*, or specification for short, is a non-empty prefix-closed subset of A^*. It gives the set of communication histories allowed at both ends of the interface. We define its operational semantics as a transition system. A transition system is a triple $\langle Q, q, \rightarrow \rangle$, where Q is some set of states, $q \in Q$ is the initial state, and $\rightarrow \subseteq Q \times Q$ is the transition relation. The transition system associated with specification T is

$$\langle A^* \times A^*, (\varepsilon,\varepsilon), \underset{T}{\rightarrow} \rangle, \tag{0}$$

where $\underset{T}{\rightarrow}$ is the smallest relation such that

$$\left.\begin{array}{lll} (t,u) \underset{T}{\rightarrow} (ta,u) & \text{if} & a \in O \;\wedge\; ta \in T \\ (t,u) \underset{T}{\rightarrow} (t,ua) & \text{if} & a \in I \;\wedge\; ua \in T \end{array}\right\} \textit{(transmissions)}$$
$$\left.\begin{array}{lll} (t,u) \underset{T}{\rightarrow} (ta,u) & \text{if} & a \in I \;\wedge\; t\#a < u\#a \\ (t,u) \underset{T}{\rightarrow} (t,ua) & \text{if} & a \in O \;\wedge\; u\#a < t\#a \end{array}\right\} \textit{(receptions)} \tag{1}$$

Thus, a state is a pair of traces over A, and in the initial state both traces of the pair are empty. The left component of the pair can be thought of as the local state of Module while the right component is associated with Environment. Outputs travel from Module to Environment, inputs from Environment to Module.

The first transition rule expresses a state change where Module extends its local state with an output symbol if the resulting local state belongs to the specification. Similarly, the second transition rule expresses a state change where Environment's local state is extended with an input symbol (which acts as a transmission initiated by Environment). Hence, transmissions will only be initiated if they are in agreement with the specification. The third transition rule expresses a state change where Module's local state is extended with an input symbol if that symbol was sent more often (by Environment) than received so far (by Module), i.e. if a message was on its way over the channel identified by that symbol. Similarly, the fourth transition rule expresses a state change where Environment receives an output (sent earlier by Module). Hence, a reception takes place only if the corresponding transmission precedes it. Note, however, that at this stage receptions are not required to obey the specification.

We call state (t,u) *reachable* under specification T when it can be reached from the initial state via zero or more T-transitions, that is, when

$$(\varepsilon,\varepsilon) \overset{*}{\underset{T}{\rightarrow}} (t,u), \tag{2}$$

where $\overset{*}{\underset{T}{\rightarrow}}$ denotes the transitive and reflexive closure of $\underset{T}{\rightarrow}$. The set of states reachable under T is denoted by $\mathrm{r}T$. State (t,u) is called *safe* under T when $(t,u) \in T \times T$, that is, when both local states belong to the specification. Specification T is *delay-insensitive*, or *DI* for short, when

$$\mathbf{r}T \subseteq T \times T, \tag{3}$$

that is, when all reachable states are safe. The situation where a reachable state is not in agreement with the specification is called *computation interference*. It corresponds to a possible violation of the digital mode assumptions. The central problem of this paper is the characterization of delay-insensitivity.

We give three examples to illustrate these definitions.

Example 0 Consider the specification $T = \{\varepsilon\}$ for arbitrary I and O. Its transition relation is empty. The initial state is the only reachable state and, hence, $\mathbf{r}T = T \times T$. Consequently, specification T is delay-insensitive. □

Example 1 Assume $I = \{a, b\}$ with $a \neq b$ and $O = \emptyset$. Now consider specification $T = \{\varepsilon, a, ab\}$. For T's transition system we give two (of the three) maximal transition sequences starting in the initial state. The first sequence consists of two transmissions interleaved with the corresponding receptions:

$$(\varepsilon, \varepsilon) \underset{T}{\rightarrow} (\varepsilon, a) \underset{T}{\rightarrow} (a, a) \underset{T}{\rightarrow} (a, ab) \underset{T}{\rightarrow} (ab, ab).$$

Thus, $(ab, ab) \in \mathbf{r}T$. Notice that the final state (ab, ab) is safe. The second sequence consists of two transmissions followed by two receptions, where the order of the receptions differs from the order of the corresponding transmissions:

$$(\varepsilon, \varepsilon) \underset{T}{\rightarrow} (\varepsilon, a) \underset{T}{\rightarrow} (\varepsilon, ab) \underset{T}{\rightarrow} (b, ab) \underset{T}{\rightarrow} (ba, ab).$$

Thus, $(ba, ab) \in \mathbf{r}T$, but now the final state (ba, ab) is not safe, i.e. there is computation interference. In fact, the intermediate state (b, ab) was already not safe. Therefore, the specification T is not delay-insensitive.

This example also exhibits one of the complications inherent to asynchronous communication: Even if there are no messages on their way, then the local states of the communicating parties may differ. □

Example 2 Assuming $I = \{a\}$ and $O = \{b\}$ define specification T by

$$T = \{t \mid (\forall\, s : s \text{ prefix of } t : 0 \leq t\#a - t\#b \leq 1)\}.$$

Thus, T consists of all traces in which symbols a and b alternate and which do not start with b:

$$T = \{\varepsilon, a, ab, aba, abab, ababa, \ldots\}.$$

Specification T describes a two-phase protocol. In each state, exactly one transition is possible. Hence, there exists only one maximal transition sequence, which is infinite and starts out

$$(\varepsilon, \varepsilon) \underset{T}{\rightarrow} (\varepsilon, a) \underset{T}{\rightarrow} (a, a) \underset{T}{\rightarrow} (ab, a) \underset{T}{\rightarrow} (ab, ab) \underset{T}{\rightarrow} \cdots$$

The two-phase protocol T is delay-insensitive, because all reachable states are safe. □

In [4, 5] Udding gives a characterization of delay-insensitive specifications. He also deals with *transmission interference*—which can occur when a channel carries more than one message—but we will ignore that here: our channels have unbounded buffering capacity. Udding defines the following predicates on specifications. We adhere to the names given in [4].

Specification T satisfies Rule R_3 (called R_1 in [5]) when for all traces s and t, and symbols a and b of the same direction we have

$$sabt \in T \;\equiv\; sbat \in T. \tag{4}$$

Specification T satisfies Rule R_4'' (called R_2' in [5]) when for all traces s and t, and symbols a, b, and c such that the direction of a and c differs from the direction of b, we have

$$sabtc \in T \;\wedge\; sbat \in T \;\Rightarrow\; sbatc \in T. \tag{5}$$

Specification T satisfies Rule R_5''' (called R_3''' in [5]) when for all traces s and symbols a and b of different direction we have

$$sa \in T \;\wedge\; sb \in T \;\Rightarrow\; sab \in T. \tag{6}$$

Remark for the curious: Rules R_0 and R_1 of [4] were already incorporated in our notion of a specification. Rule R_2 deals with transmission interference, which we decided to ignore here.

We say that a specification satisfies the *JTU-Rules* when it satisfies Rules R_3, R_4'', and R_5'''. Notice that the specifications of Examples 0 and 2 trivially satisfy the JTU-Rules. The specification of Example 1, however, satisfies Rules R_4'' and R_5''' but not Rule R_3.

This paper is centered around the following theorem.

Theorem 0 *(Fundamental Characterization Theorem of Delay-Insensitivity)*
Specification T is DI if and only if it satisfies the JTU-Rules. □

The implication from right to left is "hard" and was—in a slightly more general form—first stated and proved in [4, Thm. 4.1][1]. Several attempts at simplifying the proof have failed. In this paper we present, what we belief to be, a simple proof. In a sense, the justification of the JTU-Rules as given in [4] constitutes an informal proof of the implication from left to right. This is the "easy" part. A formal proof of this part can also be found in [7].

[1] Note on terminology: In [4] Udding defines delay-insensitivity directly in terms of the JTU-Rules and he shows that it implies absence of computation interference, which he defines as (8) below.

2 Composability and Convexity

In this section we present three—fairly straightforward—characterizations of delay-insensitivity (Theorems 1, 2, and 3). They do not get very far in bridging the gap between delay-insensitivity and the JTU-Rules, but they are useful nonetheless, since they take us away from the definition of delay-insensitivity in terms of the operational transition system.

We start by noting the following symmetry in the transition system $\mathcal{S}$ associated with specification T. The transition system obtained from S by exchanging left and right components of states equals the transition system associated with T when the roles of I and O are exchanged. This symmetry will be referred to as I/O-symmetry. Also notice that the JTU-Rules are I/O-symmetric, since they involve (in)equality of direction only.

The first characterization (Theorem 1 below) is based on the observation that the initial state is safe and that *transmission* transitions do not disturb safety by definition. Therefore, all reachable states are safe if and only if each *reception* transition from a safe reachable state leads to a safe state. We have also incorporated some knowledge about reachable states viz. that the number of receptions of a symbol cannot exceed the number of its transmissions. This is formally expressed in

Property 0 For specification T and state $(t, u) \in \mathbf{r}T$ we have

$$(\forall\, a : a \in I : t\#a \leq u\#a) \;\wedge\; (\forall\, a : a \in O : t\#a \geq u\#a). \tag{7}$$

□

Therefore, if $(t, u) \in \mathbf{r}T$ and $t\#a < u\#a$, then $a \notin O$ and, hence, $a \in I$.

Theorem 1 Specification T is DI if and only if

$$\begin{array}{lll} (\forall\, t, u, a & : & (t, u) \in \mathbf{r}T \cap (T \times T) \\ & : & (t\#a < u\#a \;\Rightarrow\; ta \in T) \;\wedge\; (u\#a < t\#a \;\Rightarrow\; ua \in T)). \end{array} \tag{8}$$

Proof

Only if: Assuming T is DI we show (8). Let $(t, u) \in \mathbf{r}T$ be such that $t \in T$ and $u \in T$. We derive

$$\begin{array}{ll} & t\#a < u\#a \\ \Rightarrow & \{\ \text{Property 0}\ \} \\ & a \in I \;\wedge\; t\#a < u\#a \\ \Rightarrow & \{\ \text{definition of } \underset{T}{\rightarrow}\ \} \\ & (t, u) \underset{T}{\rightarrow} (ta, u) \\ \Rightarrow & \{\ \text{definition of } \mathbf{r}T, \text{ using } (t, u) \in \mathbf{r}T\ \} \\ & (ta, u) \in \mathbf{r}T \end{array}$$

$\Rightarrow$ { T is assumed DI }
$(ta, u) \in (T \times T)$
$\Rightarrow$ { set calculus }
$ta \in T$

The conjunct $u\#a < t\#a \Rightarrow ua \in T$ follows from I/O-symmetry.

If: Assume T satisfies (8). We prove by induction on $\ell(t) + \ell(u)$ that all states $(t, u) \in \mathbf{r}T$ are safe.

Base: $\ell(t) + \ell(u) = 0$, hence, $t = \varepsilon = u$. The state $(\varepsilon, \varepsilon)$ is safe because T is non-empty and prefix-closed.

Step: $\ell(t) + \ell(u) > 0$, hence, we can find state $(t', u') \in \mathbf{r}T$ such that

$$(t', u') \underset{T}{\rightarrow} (t, u). \tag{9}$$

On account of the induction hypothesis, using that $\ell(t') + \ell(u') < \ell(t) + \ell(u)$, we know that (t', u') is safe and, hence,

$$t' \in T \ \wedge \ u' \in T. \tag{10}$$

We distinguish two cases: $t' = t$ and $u' = u$. Because of I/O-symmetry we need only investigate the case $t' = t$. In that case, u can be written as $u'a$ for some symbol a. Furthermore, $t \in T$ follows from (10). All that we need to show now is $u'a \in T$. We derive for the cases $a \in I$ and $a \in O$, respectively:

$a \in I$
$\Rightarrow$ { (9), $u = u'a$, and definition of $\underset{T}{\rightarrow}$ }
$u'a \in T$

and

$a \in O$
$\Rightarrow$ { (9), $u = u'a$, and definition of $\underset{T}{\rightarrow}$, using $t = t'$ }
$u'\#a < t'\#a$
$\Rightarrow$ { (8), using $(t', u') \in \mathbf{r}T$ }
$u'a \in T$

□

The preceding characterization can be simplified a little by introducing the *composability*[2] relation $\mathbf{C}$ on A^* defined as $\mathbf{r}(A^*)$. That is, $t\,\mathbf{C}\,u$ holds when (t, u) is reachable under the specification A^*. Specification A^* does not restrict transmission transitions and, thus, relation $\mathbf{C}$ captures only the restriction imposed by the condition that symbols arrive no earlier than they were sent.

[2]The name 'composability' is taken from [4].

Example 3 Assuming $a \in I$ and $b \in O$, we have $\varepsilon \,\mathbf{C}\, a$ but $\neg(a \,\mathbf{C}\, \varepsilon)$, and also $ba \,\mathbf{C}\, ab$ but $\neg(ab \,\mathbf{C}\, ba)$. □

Relation $\mathbf{C}$ enjoys a number of nice properties.

Property 1 For specification T we have

$$\mathbf{r}T \cap (T \times T) = \mathbf{C} \cap (T \times T). \tag{11}$$

Furthermore, we have

$$a \in I \wedge ta \,\mathbf{C}\, u \;\equiv\; t\#a < u\#a \wedge t \,\mathbf{C}\, u \tag{12}$$

$$a \in I \wedge t \,\mathbf{C}\, u \;\Rightarrow\; t \,\mathbf{C}\, ua \tag{13}$$

$$t \,\mathbf{C}\, ua \;\Rightarrow\; (\exists t_0, t_1 : t = t_0 t_1 : t_0 \,\mathbf{C}\, u) \tag{14}$$

and $\mathbf{C}$ is reflexive, i.e. for all t we have $t \,\mathbf{C}\, t$. □

Of course, on account of I/O-symmetry we also have dual forms of (12) through (14) obtained by interchanging I and O, and left- and right-hand arguments of $\mathbf{C}$. For example, the dual of (13) is: $a \in O \wedge u \,\mathbf{C}\, t \;\Rightarrow\; ua \,\mathbf{C}\, t$.

We now give a characterization of delay-insensitivity in which reachability under the specification in question has been traded for $\mathbf{C}$.

Theorem 2 Specification T is DI if and only if

$$\begin{array}{lll} (\forall t, u, a & : & t \in T \wedge u \in T \\ & : & (a \in I \wedge ta \,\mathbf{C}\, u \;\Rightarrow\; ta \in T) \wedge \\ & & (a \in O \wedge t \,\mathbf{C}\, ua \;\Rightarrow\; ua \in T)). \end{array} \tag{15}$$

Proof On account of Theorem 1 it is sufficient to prove the equivalence of (8) and (15). We derive

$$\begin{array}{lll} & (\forall t, u, a & : (t, u) \in \mathbf{r}T \cap (T \times T) \\ & & : (t\#a < u\#a \;\Rightarrow\; ta \in T) \;\wedge\; (u\#a > t\#a \;\Rightarrow\; ua \in T)) \\ = & \{ \text{ Property 1(11) } \} & \\ & (\forall t, u, a & : (t, u) \in \mathbf{C} \cap (T \times T) \\ & & : (t\#a < u\#a \;\Rightarrow\; ta \in T) \;\wedge\; (u\#a > t\#a \;\Rightarrow\; ua \in T)) \\ = & \{ \text{ predicate and set calculus } \} & \\ & (\forall t, u, a & : (t, u) \in (T \times T) \\ & & : (t\#a < u\#a \wedge t \,\mathbf{C}\, u \;\Rightarrow\; ta \in T) \wedge \\ & & \quad (u\#a > t\#a \wedge t \,\mathbf{C}\, u \;\Rightarrow\; ua \in T)) \\ = & \{ \text{ Property 1(12) } \} & \\ & (\forall t, u, a & : (t, u) \in (T \times T) \\ & & : (a \in I \wedge ta \,\mathbf{C}\, u \;\Rightarrow\; ta \in T) \wedge \\ & & \quad (a \in O \wedge t \,\mathbf{C}\, ua \;\Rightarrow\; ua \in T)) \end{array}$$

$$
\begin{array}{ll}
= & \{ \text{ set calculus } \} \\
 & (\forall t,u,a \ : \ t \in T \wedge u \in T \\
 & \qquad : \ (a \in I \wedge ta \,\mathbf{C}\, u \ \Rightarrow \ ta \in T) \wedge \\
 & \qquad \quad (a \in O \wedge t \,\mathbf{C}\, ua \ \Rightarrow \ ua \in T))
\end{array}
$$

□

This characterization can be further simplified:

Theorem 3 Specification T is DI if and only if

$$(\forall t,u,z : t \in T \wedge u \in T : t \,\mathbf{C}\, z \wedge z \,\mathbf{C}\, u \ \Rightarrow \ z \in T). \tag{16}$$

Specification T is called *convex* when it satisfies (16).

Proof On account of Theorem 2 it is sufficient to prove the equivalence of (15) and (16).

If: Assuming T is convex we derive (15). Let $t \in T$ and $u \in T$. We derive

$$
\begin{array}{ll}
 & a \in I \wedge ta \,\mathbf{C}\, u \\
= & \{ \text{ reflexivity of } \mathbf{C} \text{ on account of Property 1 } \} \\
 & a \in I \wedge t \,\mathbf{C}\, t \wedge ta \,\mathbf{C}\, u \\
\Rightarrow & \{ \text{ Property 1(13) } \} \\
 & t \,\mathbf{C}\, ta \wedge ta \,\mathbf{C}\, u \\
\Rightarrow & \{ \text{ (16) assumed, using } t \in T \text{ and } u \in T \ \} \\
 & ta \in T
\end{array}
$$

The other conjunct follows I/O-symmetrically.

Only if: Assuming (15) we prove (16) by induction on the length of z.

Base: $z = \varepsilon$. Since T is non-empty and prefix-closed we have $z = \varepsilon \in T$.

Step: $z = z'a$. Assuming $t \in T$ and $u \in T$ such that $t \,\mathbf{C}\, z$ and $z \,\mathbf{C}\, u$, we show $z'a \in T$. We distinguish the cases $a \in I$ and $a \in O$. Because of I/O-symmetry we consider only the first case. Therefore, assume $a \in I$. On account of Property 1(14), using $a \in I$ and $t \,\mathbf{C}\, z'a$, we can let t' be a prefix of t such that $t' \,\mathbf{C}\, z'$. We derive

$$
\begin{array}{ll}
 & \text{true} \\
= & \{ \text{ context so far } \} \\
 & t \in T \wedge u \in T \wedge t' \,\mathbf{C}\, z' \wedge a \in I \wedge z'a \,\mathbf{C}\, u \\
\Rightarrow & \{ \ t' \text{ is a prefix of } t, \ T \text{ is prefix-closed, and Property 1(12) } \} \\
 & t' \in T \wedge u \in T \wedge t' \,\mathbf{C}\, z' \wedge z' \,\mathbf{C}\, u \wedge a \in I \wedge z'a \,\mathbf{C}\, u \\
\Rightarrow & \{ \text{ induction hypothesis, using } \ell(z') < \ell(z) \ \} \\
 & z' \in T \wedge u \in T \wedge a \in I \wedge z'a \,\mathbf{C}\, u \\
\Rightarrow & \{ \text{ (15) assumed } \} \\
 & z'a \in T
\end{array}
$$

□

3 Inversions

In [8] a proof is given for 'the JTU-Rules imply convexity'. That proof is based on a construction in terms of graphs. It is easy to understand if one is willing to accept some intuitions about graphs. A complete formalization is still quite lengthy. We will not take that road here. We postpone this implication and instead concentrate on the (easier) converse.

Before we tackle the converse it is useful to get to know the composability relation C a little better. We call $(t'a, u'b)$ an *inversion* in (t, u) when

$t'a$ is a prefix of t,
$u'b$ is a prefix of u,
$t'a\#a > u'b\#a$, and
$t'a\#b < u'b\#b$.

The first condition expresses that $t'a$ locates an occurrence of symbol a in trace t and, similarly, $u'b$ locates an occurrence of b in u on account of the second condition. The third condition expresses that the occurrence of a in u, that corresponds to $t'a$, occurs to the 'right' of $u'b$—if it exists at all. Similarly, the fourth condition expresses that the occurrence of b in t which corresponds to $u'b$ occurs to the 'right' of $t'a$. Hence, the order of these occurrences of a and b in t differs from the order of the corresponding occurrences in u. The *set of inversions* in state (t, u) will be denoted by $inv(t, u)$. Inversion $(t'a, u'b)$ in (t, u) is called a *t-neighbor inversion* when $t'ab$ is a prefix of t and $t'a\# b = u'\#b$, that is, when these occurrences of a and b are adjacent in t.

Notice that the concept of inversion does not involve the directions of symbols, i.e., it is independent of how A is partitioned into I and O. We will only be interested in inversions in states (t, u) for which $t\# = u\#$. Let us look at an example.

Example 4 Assuming that symbols a, b, and c are distinct, there are three inversions

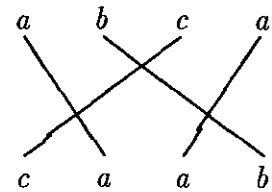

Figure 0: Inversion diagram for $(abca, caab)$

in $(abca, caab)$. The set $inv(abca, caab)$ consists of (a, c), (ab, c), and (ab, caa). Only one of these, viz. (ab, c), is an $abca$-neighbor inversion. This is illustrated in Figure 0, where line segments connect corresponding symbol occurrences. Each pair of intersecting line segments corresponds to an inversion. □

We draw the attention to some well-known properties of inversions to be used later on:

Property 2 For state (t, u) such that $t\# = u\#$ we have

$inv(t, u)$ is finite,
$inv(t, u) = \emptyset \equiv t = u$,
$inv(t, u) \neq \emptyset \equiv inv(t, u)$ contains a t-neighbor inversion.

If $(ta, u'b)$ is a $tabv$-neighbor inversion in $(tabv, u)$, then

$$inv(tbav, u) = inv(tabv, u) - \{(ta, u'b)\}.$$

□

Composability can be characterized in terms of inversions:

Property 3 For state (t, u) we have

$$t \mathbf{C} u \Rightarrow (\forall t', u', a, b : (t'a, u'b) \in inv(t, u) : a \in O \vee b \in I) \quad (17)$$

and this is an equivalence if $t\# = u\#$. □

Example 5 Assuming $a \in I$ and $b \in O$, we have seen in Example 3 that $\neg(ab \mathbf{C} ba)$ holds. State (ab, ba) has one inversion, viz. (a, b), for which $\neg(a \in O \vee b \in I)$. □

By the way, from Property 3 it follows (non-trivially) that $\mathbf{C}$ is transitive (hence, a pre-order), but we will not need that here.

We are now ready for

Theorem 4 If specification T is convex (cf. (16)), then it satisfies the JTU-Rules.

Proof Assume specification T is convex. Each JTU-Rule can be viewed as a special case of convexity. We prove them one by one.

Rule R_3: For traces s and t, and symbols a and b of the same direction we have, on account of Property 3,

$$sabt \mathbf{C} sbat \mathbf{C} sabt \mathbf{C} sbat.$$

Using convexity we now infer $sabt \in T \equiv sbat \in T$ and, hence, T satisfies R_3.

Rule R_4'': For traces s and t, and symbols a, b, and c such that the direction of a and c differs from that of b, we have, using Properties 3 and 1(13),

$$a \in O \Rightarrow sabtc \mathbf{C} sbatc \mathbf{C} sbat$$
$$a \in I \Rightarrow sbat \mathbf{C} sbatc \mathbf{C} sabtc$$

On account of convexity we thus have that $sabtc \in T \wedge sbat \in T$ implies $sbatc \in T$ and, hence, T satisfies R_4''.

Rule R_5''': For trace s and symbols a and b of different direction we have, using reflexivity of $\mathbf{C}$ and Properties 1(12) and 1(13),

$$a \in O \Rightarrow sa \mathbf{C} sab \mathbf{C} sb$$
$$a \in I \Rightarrow sb \mathbf{C} sab \mathbf{C} sa$$

On account of convexity we then have that $sa \in T \wedge sb \in T$ implies $sab \in T$ and, hence, T satisfies R_5'''. □

4 New Representation for Specifications

The major innovation in our proof of the Fundamental Characterization Theorem of Delay-Insensitivity is a new representation for specifications. This representation is based on *enhanced characteristic functions*, or ECFs for short. An ECF is a mapping from A^* to $\{0,1,2\}$. The enhancement consists of the additional value 2 in the co-domain, which enables us to distinguish two ways in which a trace does not belong to a specification. We use $\cdot$ to denote functional application for ECFs, which has a weaker binding power than concatenation, i.e. $f{\cdot}st$ stands for $f{\cdot}(st)$.

Let T be a specification. Observe that each trace $t \notin T$ can be uniquely written as t_0at_1 such that $t_0 \in T$ and $t_0a \notin T$, since T is non-empty and prefix-closed. We now define ECF $\mathbf{f}T$ by

$$\mathbf{f}T{\cdot}t = \begin{cases} 0 & \text{if } (\exists\, t_0, a, t_1 : t = t_0at_1 \wedge t_0 \in T \wedge t_0a \notin T : a \in O) \\ 1 & \text{if } t \in T \\ 2 & \text{if } (\exists\, t_0, a, t_1 : t = t_0at_1 \wedge t_0 \in T \wedge t_0a \notin T : a \in I) \end{cases} \tag{18}$$

Let $f = \mathbf{f}T$, then f enjoys the following properties:

$$\begin{array}{ll} (F_0) & f{\cdot}\varepsilon = 1 \\ (F_1) & f{\cdot}ta = f{\cdot}t \text{ if } f{\cdot}t \neq 1 \\ (F_2) & f{\cdot}ta \leq f{\cdot}t \text{ if } a \in O \\ (F_3) & f{\cdot}ta \geq f{\cdot}t \text{ if } a \in I \end{array}$$

These follow immediately from the definition of $\mathbf{f}T$ and the fact that T is non-empty and prefix-closed. Properties F_1 through F_3 are readily generalized to

$$\begin{array}{ll} (F_1') & f{\cdot}tu = f{\cdot}t \text{ if } f{\cdot}t \neq 1 \\ (F_2') & f{\cdot}tu \leq f{\cdot}t \text{ if } u \in O^* \\ (F_3') & f{\cdot}tu \geq f{\cdot}t \text{ if } u \in I^* \end{array}$$

by induction on the length of u. Furthermore, $F_1 \wedge F_2 \wedge F_3$ is equivalent to the conjunction of

$$\begin{array}{ll} (F_2'') & f{\cdot}ta < f{\cdot}t \;\Rightarrow\; f{\cdot}t = 1 \wedge f{\cdot}ta = 0 \wedge a \in O \\ (F_3'') & f{\cdot}ta > f{\cdot}t \;\Rightarrow\; f{\cdot}t = 1 \wedge f{\cdot}ta = 2 \wedge a \in I \end{array}$$

For ECF f we define its trace set $\mathbf{t}f$ by

$$\mathbf{t}f = \{t \mid f{\cdot}t = 1\}. \tag{19}$$

We now trivially have for $T \subseteq A^*$

$$\mathbf{t}(\mathbf{f}T) = T. \tag{20}$$

We also claim that for ECF f satisfying F_0 through F_3 we have that $\mathbf{t}f$ is non-empty and prefix-closed (viz. on account of F_0 and F_1) and that

$$\mathbf{f}(\mathbf{t}f) = f. \tag{21}$$

We have thus established a one-one correspondence between specifications and ECFs satisfying F_0 through F_3. Notice that exchanging the role of I and O corresponds to exchanging the role of 0 and 2 in ECFs.

Using the ECF of a specification, the JTU-Rules can be condensed into a single rule. We prove only an implication here; the converse will follow from Theorems 6, 7, and 4.

Theorem 5 If specification T satisfies the JTU-Rules then

$$(\forall\, s, a, b, t : a \in O \;\vee\; b \in I : \mathbf{f}T{\cdot}sabt \leq \mathbf{f}T{\cdot}sbat). \tag{22}$$

Predicate (22) will be called the *Neighbor-Swap Rule.*

Proof Assuming T satisfies the JTU-Rules we show that T satisfies the Neighbor-Swap Rule. Because the co-domain of $\mathbf{f}T$ is $\{0, 1, 2\}$ and

	0	1	2
0	$\leq$	$\leq$	$\leq$
1	$>$	$\leq$	$\leq$
2	$>$	$>$	$\leq$

it is sufficient to prove for all traces s and t, and symbols a and b such that $a \in O \vee b \in I$:

$$\mathbf{f}T{\cdot}sabt = 2 \;\Rightarrow\; \mathbf{f}T{\cdot}sbat = 2 \text{ and}$$
$$\mathbf{f}T{\cdot}sbat = 0 \;\Rightarrow\; \mathbf{f}T{\cdot}sabt = 0.$$

On account of I/O-symmetry we confine ourselves to the first of these. Therefore, also assume that $\mathbf{f}T{\cdot}sabt = 2$. From the definition of $\mathbf{f}T$ now follows that we can find u and c such that

$$uc \text{ prefix of } sabt \;\wedge\; u \in T \;\wedge\; uc \notin T \;\wedge\; c \in I. \tag{23}$$

Our goal is to show that $\mathbf{f}T{\cdot}sbat = 2$ as well. We distinguish four cases: uc prefix of s, $uc = sa$, $uc = sab$, and sab prefix of u.

Case uc prefix of s: Then $\mathbf{f}T{\cdot}sbat = 2$ by (23) and the definition of $\mathbf{f}T$.

Case $uc = sa$: We derive

$$\begin{array}{ll} & uc = sa \\ = & \quad \{ \text{ trace calculus } \} \\ & u = s \;\wedge\; s = a \\ \Rightarrow & \quad \{ \text{ (23) } \} \\ & s \in T \;\wedge\; sa \notin T \;\wedge\; a \in I \\ \Rightarrow & \quad \{ T \text{ is prefix-closed and } a \in O \vee b \in I \text{ assumed } \} \\ & s \in T \;\wedge\; sab \notin T \;\wedge\; a \in I \;\wedge\; b \in I \end{array}$$

$=$ { Rule R_3 assumed }
$s \in T \wedge sba \notin T \wedge a \in I \wedge b \in I$
$\Rightarrow$ { predicate calculus, doing case analysis on $sb \in T$ }
$(s \in T \wedge sb \notin T \wedge b \in I) \vee (sb \in T \wedge sba \notin T \wedge a \in I)$
$\Rightarrow$ { definition of $\mathbf{f}T$ }
$\mathbf{f}T{\cdot}sbat = 2$

Case $uc = sab$: We derive

$uc = sab$
$=$ { trace calculus }
$u = sa \wedge c = b$
$\Rightarrow$ { (23) }
$sa \in T \wedge sab \notin T \wedge b \in I$
$\Rightarrow$ { predicate calculus }
$(sa \in T \wedge sab \notin T \wedge a \in O \wedge b \in I) \vee (sa \in T \wedge sab \notin T \wedge a \in I \wedge b \in I)$
$\Rightarrow$ { Rules R_5''' and R_3 assumed and T is prefix-closed }
$(s \in T \wedge sb \notin T \wedge b \in I) \vee (s \in T \wedge sba \notin T \wedge a \in I \wedge b \in I)$
$\Rightarrow$ { predicate calculus, doing case analysis on $sb \in T$ }
$(s \in T \wedge sb \notin T \wedge b \in I) \vee (sb \in T \wedge sba \notin T \wedge a \in I)$
$\Rightarrow$ { definition of $\mathbf{f}T$ }
$\mathbf{f}T{\cdot}sbat = 2$

Case sab prefix of u: Hence, we can write $t = t_0 c t_1$ such that $u = sabt_0$. We distinguish two subcases depending on the equality of the directions of a and b. For a and b having the same direction we derive

$u = sabt_0$
$\Rightarrow$ { (23) }
$sabt_0 \in T \wedge sabt_0 c \notin T \wedge c \in I$
$=$ { Rule R_3 assumed, using that a and b have same direction }
$sbat_0 \in T \wedge sbat_0 c \notin T \wedge c \in I$
$\Rightarrow$ { definition of $\mathbf{f}T$ }
$\mathbf{f}T{\cdot}sbat = 2$

For symbols a and b with different directions we proceed as follows. From the assumption $a \in O \vee b \in I$ we now infer $a \in O \wedge b \in I$ and we derive

$u = sabt_0$
$\Rightarrow$ { (23) }

$$
\begin{array}{ll}
 & sabt_0 \in T \ \wedge \ sabt_0c \notin T \ \wedge \ c \in I \\
\Rightarrow & \quad \{ \text{ Rule } R_4'' \text{ assumed, using that } a \in O \ \wedge \ b \in I \ \} \\
 & sabt_0 \in T \ \wedge \ sbat_0c \notin T
\end{array}
$$

Hence, we can find prefix vd of $sbat_0c$ such that $v \in T$ but $vd \notin T$. On account of the definition of $\mathbf{f}T$ it is sufficient to show $d \in I$ in order to have $\mathbf{f}T{\cdot}sbat = 2$. Finally, we distinguish the five ways in which vd can be a prefix of $sbat_0c$, viz. vd prefix of s, $vd = sb$, $vd = sba$, vd prefix of $sbat_0$ with sba prefix of v, and $vd = sbat_0c$. The first case is excluded by $s \in T$. In the second case we have $d = b \in I$. The third case is excluded by $sa \in T$ and Rule R_5'''. For the fourth case note that $d \in O$ is excluded by Rule R_4'' and the fact that $sabt_2d$ is a prefix of $sabt_0$ with $sabt_0 \in T$. In the last case we have $d = c \in I$. This completes the proof. □

The Neighbor-Swap Rule can be generalized as follows:

Theorem 6 Specification T satisfies the Neighbor-Swap Rule (22) if and only if

$$(\forall t, u : t \mathbf{C} u : \mathbf{f}T{\cdot}t \leq \mathbf{f}T{\cdot}u). \tag{24}$$

We say that specification T is *monotonic* if it satisfies (24).

Proof

If: That (24) implies (22) follows immediately from

$$a \in O \ \vee \ b \in I \ \equiv \ sabt \mathbf{C} sbat,$$

which is a consequence of Property 3.

Only if: Assuming T satisfies the Neighbor-Swap Rule we prove that T is monotonic. Let t and u be such that $t \mathbf{C} u$. We first deal with the case where $t\# = u\#$. We prove $\mathbf{f}T{\cdot}t \leq \mathbf{f}T{\cdot}u$ by induction on the number of inversions in (t, u).

Base: $inv(t, u) = \emptyset$. Hence, on account of Property 2 using $t\# = u\#$, we have $t = u$ and, thus, $\mathbf{f}T{\cdot}t \leq \mathbf{f}T{\cdot}u$.

Step: $inv(t, u) \neq \emptyset$. Hence, on account of Property 2 using $t\# = u\#$, there exists a t-neighbor inversion in (t, u), say, (t_0a, u_0b). Therefore, we can write $t = t_0abt_1$. From Property 3 and assumption $t \mathbf{C} u$ follows $a \in O \ \vee \ b \in I$. We now derive

$$
\begin{array}{ll}
 & \mathbf{f}T{\cdot}t \\
= & \quad \{ \ t = t_0abt_1 \ \} \\
 & \mathbf{f}T{\cdot}t_0abt_1 \\
\leq & \quad \{ \ T \text{ satisfies the Neighbor-Swap Rule by assumption, using } a \in O \ \vee \ b \in I \ \} \\
 & \mathbf{f}T{\cdot}t_0bat_1 \\
\leq & \quad \{ \text{ induction hypothesis, using } inv(t_0bat_1, u) \subset inv(t_0abt_1, u) \text{ by Property 2 } \} \\
 & \mathbf{f}T{\cdot}u
\end{array}
$$

Finally, we consider the other case where $t\# \neq u\#$. In that case we can find $v \in I^*$ and $w \in O^*$ such that $tv\# = uw\#$ and $tv \,\mathbf{C}\, uw$ on account of Property 1(12). We now derive

$$
\begin{array}{ll}
 & \mathbf{f}T{\cdot}t \\
\leq & \quad \{\ (F_3'),\ \text{using } v \in I^*\ \} \\
 & \mathbf{f}T{\cdot}tv \\
\leq & \quad \{\ \text{first case, using } tv\# = uw\# \text{ and } tv \,\mathbf{C}\, uw\ \} \\
 & \mathbf{f}T{\cdot}uw \\
\leq & \quad \{\ (F_2'),\ \text{using } w \in O^*\ \} \\
 & \mathbf{f}T{\cdot}u
\end{array}
$$

□

At last, we can close the gap between delay-insensitivity and the JTU-Rules:

Theorem 7 If specification T is monotonic (cf. (24)) then it is convex (cf. (16)).

Proof Assuming T is monotonic, we show that it is convex. Let $t \in T$ and $u \in T$ such that $t \,\mathbf{C}\, z$ and $z \,\mathbf{C}\, u$. We derive

$$
\begin{array}{ll}
 & 1 \\
= & \quad \{\ \text{definition of } \mathbf{f}T,\ \text{using } t \in T\ \} \\
 & \mathbf{f}T{\cdot}t \\
\leq & \quad \{\ \text{monotonicity assumed, using } t \,\mathbf{C}\, z\ \} \\
 & \mathbf{f}T{\cdot}z \\
\leq & \quad \{\ \text{monotonicity assumed, using } z \,\mathbf{C}\, u\ \} \\
 & \mathbf{f}T{\cdot}u \\
= & \quad \{\ \text{definition of } \mathbf{f}T,\ \text{using } u \in T\ \} \\
 & 1
\end{array}
$$

Hence, $\mathbf{f}T{\cdot}z = 1$ and from the definition of $\mathbf{f}T$ now follows $z \in T$. □

We conclude this section with the proof for Theorem 0:

Proof On account of Theorem 3 it is sufficient to show that convexity is equivalent to the JTU-Rules. We show the two implications in one derivation:

$$
\begin{array}{ll}
 & T \text{ is convex} \\
\Rightarrow & \quad \{\ \text{Theorem 4}\ \} \\
 & T \text{ satisfies the JTU-Rules} \\
\Rightarrow & \quad \{\ \text{Theorem 5}\ \} \\
 & T \text{ satisfies the Neighbor-Swap Rule} \\
= & \quad \{\ \text{Theorem 6}\ \} \\
 & T \text{ is monotonic} \\
\Rightarrow & \quad \{\ \text{Theorem 7}\ \} \\
 & T \text{ is convex}
\end{array}
$$

□

5 Concluding Remarks

We have studied protocols for asynchronous communication between two parties over an interface of directed channels. Non-empty prefix-closed trace sets have been used to specify communication protocols. Such a specification embodies restrictions on the initiation of transmissions and the readiness for receptions, for *both* parties. An operational semantics for the communication activity has been given in terms of a transition system. We have defined the notion of a delay-insensitive protocol specification based on absence of computation interference as a correctness concern. This correctness concern derives from an interpretation of the model as an abstraction of digital circuit physics.

The central problem of this paper has been the characterization of delay-insensitive protocol specifications. In summary, we have shown that for all protocol specifications T the following statements are equivalent:

- T is delay-insensitive (DI)
- T satisfies (8)
- T satisfies (15)
- T is convex (cf. (16))
- T is monotonic (cf. (24))
- T satisfies the Neighbor-Swap Rule (cf. (22))
- T satisfies the JTU-Rules

The characterization with JTU-Rules is due to [4] and that with convexity first appears in [7]. The characterizations in terms of monotonicity and the Neighbor-Swap Rule are new. Both are based on a new representation of protocol specifications by means of enhanced characteristic functions. The Neighbor-Swap Rule and monotonicity have turned out to be convenient stepping stones for a new proof of the Fundamental Characterization Theorem of Delay-Insensitivity.

Because of its simplicity, the Neighbor-Swap Rule is preferable to the JTU-Rules, for example, when checking a specification for delay-insensitivity. We should point out, however, that Udding [4] used variations on the JTU-Rules to classify delay-insensitive specifications. This classification is not obvious in terms of the Neighbor-Swap Rule and the variations are also easier to check in the minimal-deterministic-state-graph representation of specifications.

Dill's canonical process descriptions in [1] can be related to our's as follows. Protocol specification T has canonical process description

$$\langle I, O, T, \{t \mid \mathbf{f}T{\cdot}t = 2\}\rangle$$

and canonical process description $\langle I, O, S, F\rangle$ corresponds to protocol specification S (recall that the sets I and O are fixed in our context; the F-component of a canonical process description is superfluous). Our new representation in terms of the enhanced characteristic function is so nice because it maintains the I/O-symmetry and, thus, allows a uniform treatment of the three sets S, F, and, $(I \cup O)^* - (S \cup F)$.

The partial order $\sqsubseteq$ on specifications defined in [6] corresponds to the point-wise order on enhanced characteristic functions. For specifications S and T we have

$$S \sqsubseteq T \;\equiv\; (\forall\, s :: \mathbf{f}S{\cdot}s \leq \mathbf{f}T{\cdot}s).$$

This property greatly simplifies the analysis of the $\sqsubseteq$-lattice of protocol specifications. The alternative representation $\mathcal{T}'$ of specifications suggested in [6] consists of pairs

$$\langle \{t \mid \mathbf{f}T{\cdot}t \leq 1\}, \{t \mid \mathbf{f}T{\cdot}t = 2\}\rangle.$$

Both these sets are $\mathbf{C}$-upward closed for DI specifications. The relation **nai** of [6] enjoys the property

$$S \;\mathbf{nai}\; T \equiv (\forall\, s, t : s\,\mathbf{C}\,t : \mathbf{f}S{\cdot}s \leq \mathbf{f}T{\cdot}t)$$

and, therefore, the ECF $\hat{f}$ of T's DI-equivalent, i.e. of $lub.[T]$, satisfies

$$\hat{f}{\cdot}t = (\mathbf{MAX}\, s : s\,\mathbf{C}\,t : \mathbf{f}T{\cdot}s).$$

In this paper we have dealt with the case of two parties communicating according to a single protocol specification. In [0] general networks of asynchronously communicating processes are studied. There, it is also shown that the special case of a closed network consisting of two processes with the same trace set plays an important role in defining a denotational semantics.

The relationship with [2] by Josephs et al. is also prominent. Their relation $\sqsubseteq$ on traces can be expressed as follows:

$$u \sqsubseteq t \;\equiv\; t\,\mathbf{C}\,u \;\wedge\; t\# = u\#.$$

They denote an asynchronous process by a pair $\langle F, D\rangle$ of trace sets satisfying certain closure properties. Because of these closure properties, the trace sets F and D can be reconstructed from $F - D$. The prefix-closures of these difference sets, i.e. $\hat{F} - D$, precisely span our space of DI specifications.

In this paper we have investigated safety aspects only. Liveness aspects can be incorporated, but this requires a more refined notion of protocol specification and, in general, a more subtle way of defining the operational semantics. This will be reported on in a separate paper. It results in a specification space isomorphic to the one presented in [2].

References

[0] W. Chen, J. T. Udding, and T. Verhoeff. Networks of communicating processes and their (de)-composition. In R. Backhouse and J. van de Snepscheut, editors, *The Mathematics of Program Construction*, number 375 in Lecture Notes in Computer Science, pages ??–?? Springer-Verlag, 1989.

[1] D. L. Dill. *Trace Theory for Automatic Hierarchical Verification of Speed-Independent Circuits.* PhD thesis, C.S. Dept., Carnegie Mellon Univ., Pittsburgh, PA, Feb. 1988.

[2] M. B. Josephs, C. A. R. Hoare, and H. Jifeng. A theory of asynchronous processes. *J. ACM*, (submitted), 1989.

[3] C. E. Molnar, T.-P. Fang, and F. U. Rosenberger. Synthesis of delay-insensitive modules. In H. Fuchs, editor, *1985 Chapel Hill Conference on Very Large Scale Integration*, pages 67–86. Computer Science Press, 1985.

[4] J. T. Udding. *Classification and Composition of Delay-Insensitive Circuits.* PhD thesis, Dept. of Math. and C.S., Eindhoven Univ. of Technology, 1984.

[5] J. T. Udding. A formal model for defining and classifying delay-insensitive circuits. *Distributed Computing*, 1(4):197–204, 1986.

[6] J. T. Udding and T. Verhoeff. The mathematics of directed specifications. Technical Report WUCS-88-20, Dept. of C.S., Washington Univ., St. Louis, MO, June 1988.

[7] T. Verhoeff. Notes on delay-insensitivity. Master's thesis, Dept. of Math. and C.S., Eindhoven Univ. of Technology, 1985.

[8] T. Verhoeff. DSG08. Notes of the Directed Specifications Group, Jan. 1989.

In this series appeared :

No.	Author(s)	Title
85/01	R.H. Mak	The formal specification and derivation of CMOS-circuits
85/02	W.M.C.J. van Overveld	On arithmetic operations with M-out-of-N-codes
85/03	W.J.M. Lemmens	Use of a computer for evaluation of flow films
85/04	T. Verhoeff H.M.J.L. Schols	Delay insensitive directed trace structures satisfy the foam rubber wrapper postulate
86/01	R. Koymans	Specifying message passing and real-time systems
86/02	G.A. Bussing K.M. van Hee M. Voorhoeve	ELISA, A language for formal specifications of information systems
86/03	Rob Hoogerwoord	Some reflections on the implementation of trace structures
86/04	G.J. Houben J. Paredaens K.M. van Hee	The partition of an information system in several parallel systems
86/05	Jan L.G. Dietz Kees M. van Hee	A framework for the conceptual modeling of discrete dynamic systems
86/06	Tom Verhoeff	Nondeterminism and divergence created by concealment in CSP
86/07	R. Gerth L. Shira	On proving communication closedness of distributed layers
86/08	R. Koymans R.K. Shyamasundar W.P. de Roever R. Gerth S. Arun Kumar	Compositional semantics for real-time distributed computing (Inf.&Control 1987)
86/09	C. Huizing R. Gerth W.P. de Roever	Full abstraction of a real-time denotational semantics for an OCCAM-like language
86/10	J. Hooman	A compositional proof theory for real-time distributed message passing
86/11	W.P. de Roever	Questions to Robin Milner - A responder's commentary (IFIP86)
86/12	A. Boucher R. Gerth	A timed failures model for extended communicating processes

86/13	R. Gerth W.P. de Roever	Proving monitors revisited: a first step towards verifying object oriented systems (Fund. Informatica IX-4)
86/14	R. Koymans	Specifying passing systems requires extending temporal logic
87/01	R. Gerth	On the existence of sound and complete axiomatizations of the monitor concept
87/02	Simon J. Klaver Chris F.M. Verberne	Federatieve Databases
87/03	G.J. Houben J.Paredaens	A formal approach to distributed information systems
87/04	T.Verhoeff	Delay-insensitive codes - An overview
87/05	R.Kuiper	Enforcing non-determinism via linear time temporal logic specification.
87/06	R.Koymans	Temporele logica specificatie van message passing en real-time systemen (in Dutch).
87/07	R.Koymans	Specifying message passing and real-time systems with real-time temporal logic.
87/08	H.M.J.L. Schols	The maximum number of states after projection.
87/09	J. Kalisvaart L.R.A. Kessener W.J.M. Lemmens M.L.P. van Lierop F.J. Peters H.M.M. van de Wetering	Language extensions to study structures for raster graphics.
87/10	T.Verhoeff	Three families of maximally nondeterministic automata.
87/11	P.Lemmens	Eldorado ins and outs. Specifications of a data base management toolkit according to the functional model.
87/12	K.M. van Hee and A.Lapinski	OR and AI approaches to decision support systems.
87/13	J.C.S.P. van der Woude	Playing with patterns, searching for strings.
87/14	J. Hooman	A compositional proof system for an occam-like real-time language

87/15	C. Huizing R. Gerth W.P. de Roever	A compositional semantics for statecharts
87/16	H.M.M. ten Eikelder J.C.F. Wilmont	Normal forms for a class of formulas
87/17	K.M. van Hee G.-J.Houben J.L.G. Dietz	Modelling of discrete dynamic systems framework and examples
87/18	C.W.A.M. van Overveld	An integer algorithm for rendering curved surfaces
87/19	A.J.Seebregts	Optimalisering van file allocatie in gedistribueerde database systemen
87/20	G.J. Houben J. Paredaens	The R^2 -Algebra: An extension of an algebra for nested relations
87/21	R. Gerth M. Codish Y. Lichtenstein E. Shapiro	Fully abstract denotational semantics for concurrent PROLOG
88/01	T. Verhoeff	A Parallel Program That Generates the Möbius Sequence
88/02	K.M. van Hee G.J. Houben L.J. Somers M. Voorhoeve	Executable Specification for Information Systems
88/03	T. Verhoeff	Settling a Question about Pythagorean Triples
88/04	G.J. Houben J.Paredaens D.Tahon	The Nested Relational Algebra: A Tool to handle Structured Information
88/05	K.M. van Hee G.J. Houben L.J. Somers M. Voorhoeve	Executable Specifications for Information Systems
88/06	H.M.J.L. Schols	Notes on Delay-Insensitive Communication
88/07	C. Huizing R. Gerth W.P. de Roever	Modelling Statecharts behaviour in a fully abstract way
88/08	K.M. van Hee G.J. Houben L.J. Somers M. Voorhoeve	A Formal model for System Specification
88/09	A.T.M. Aerts K.M. van Hee	A Tutorial for Data Modelling

88/10	J.C. Ebergen	A Formal Approach to Designing Delay Insensitive Circuits
88/11	G.J. Houben J.Paredaens	A graphical interface formalism: specifying nested relational databases
88/12	A.E. Eiben	Abstract theory of planning
88/13	A. Bijlsma	A unified approach to sequences, bags, and trees
88/14	H.M.M. ten Eikelder R.H. Mak	Language theory of a lambda-calculus with recursive types
88/15	R. Bos C. Hemerik	An introduction to the category theoretic solution of recursive domain equations
88/16	C.Hemerik J.P.Katoen	Bottom-up tree acceptors
88/17	K.M. van Hee G.J. Houben L.J. Somers M. Voorhoeve	Executable specifications for discrete event systems
88/18	K.M. van Hee P.M.P. Rambags	Discrete event systems: concepts and basic results.
88/19	D.K. Hammer K.M. van Hee	Fasering en documentatie in software engineering.
88/20	K.M. van Hee L. Somers M.Voorhoeve	EXSPECT, the functional part.
89/1	E.Zs.Lepoeter-Molnar	Reconstruction of a 3-D surface from its normal vectors.
89/2	R.H. Mak P.Struik	A systolic design for dynamic programming.
89/3	H.M.M. Ten Eikelder C. Hemerik	Some category theoretical properties related to a model for a polymorphic lambda-calculus.
89/4	J.Zwiers W.P. de Roever	Compositionality and modularity in process specification and design: A trace-state based approach.
89/5	Wei Chen T.Verhoeff J.T.Udding	Networks of Communicating Processes and their (De-)Composition.
89/6	T.Verhoeff	Characterizations of Delay-Insensitive Communication Protocols.
89/7	P.Struik	A systematic design of a paralell program for Dirichlet convolution.