TECHNION - Israel Institute of Technology
Computer Science Department

THE BEHAVIORAL DESCRIPTION OF ARBITERS

by

M. Yoeli, R. Ladin and Z. Barzilai

Technical Report #125

March 1978
THE BEHAVIORAL DESCRIPTION OF ARBITERS

by

M. Yoeli, R. Ladin and Z. Barzilai

ABSTRACT

This paper introduces formal models for the behavioral description of m-user n-server arbiters, based either on an arbitrary-choice or a priority assignment rule.

For this purpose suitable extensions of the concept of Petri net are defined.
1. INTRODUCTION

In most parallel processing systems various types of arbiters play an important role. Although a variety of arbitration network designs have been proposed in the literature [MIS, OH-KO-NE-SU, PLU], these papers do not prove their design to be correct. Indeed, most of these papers do not even provide precise specifications of the network behavior they intend to realize.

As first step towards a design verification methodology for arbitration networks, we discuss in this paper various mathematical models, by means of which arbiters may be precisely and conveniently specified. In preparation of this goal, we describe arbiters in the next section in a somewhat informal way.

2. INFORMAL DESCRIPTION OF ARBITERS

Consider a situation where \( n > 1 \) processes share \( m \) \( (1 \leq m < n) \) identical resources [PAT]. The resources form a pool from which they may be borrowed as needed. When a process needs one of the resources, it issues a request for one. If one of the resources is free, it is assigned to the process. When the resource is no longer required by the process, it is returned to the pool.

In the above example we speak of an \( m \)-server \( n \)-user arbiter [PAT], the processes being the users, and the resources the servers.

The \( i \)-th user \( (1 \leq i \leq n) \) of an \( n \)-user arbiter may send an UP-signal (notation: \( i+ \)) to the arbiter to issue a request. This request may be terminated by the user by sending a DOWN-signal (notation \( i+ \)). Usually
the DOWN-signal will be issued by the \( i \)-th user some time after the resource has been assigned to it, to indicate that the resource is no longer needed. However, the DOWN-signal may also be issued during the waiting period, to indicate cancelation of the request.

The UP and DOWN signals may consist of the corresponding transitions (0 → 1 and 1 → 0, respectively) of a binary level signal. Alternatively, UP and DOWN may be separate pulse signals. In any case, we assume that any user transmits UP and DOWN signals alternatively.

The behavior of a 1-server, 2-user arbiter may be specified as shown in Figure 1. The state diagram of Figure 1(b) is based on the simplifying assumption that input signals may appear only one at a time. However, such a simplifying assumption is not satisfactory.

To completely describe the behavior of the arbiter, we must face the possibility that two signals appear simultaneously. We single out two approaches to the treatment of simultaneous signals.

* (ACP) - Arbitrary-Choice Principle (see [KEL], "Arbitration Condition").

If two input signals appear simultaneously, the action taken is as if one signal occurred first and then the other. If the action depends on the order of occurrence, the order may be chosen arbitrarily. In this case the behavior of the system is *non-deterministic*. For example, if the system of Figure 1 is in state \( q_0 \) and input signals \( 1^+ \) and \( 2^+ \) appear simultaneously, then the system is "free to choose" between the actions:

\[
q_0 \xrightarrow{1^+} q_1 \xrightarrow{2^+} q_3 \quad \text{and} \quad q_0 \xrightarrow{2^+} q_2 \xrightarrow{1^+} q_4.
\]
Figure 1: A 1-server 2-user arbiter
(a) external connections
(b) simplified state diagram
(c) output table.
(PP) - Priority Principle

Every input signal is assigned some non-negative priority. If input signals \( \sigma_1, \sigma_2 \) appear simultaneously, and \( \sigma_1 \) has higher priority than \( \sigma_2 \), then the action taken is as if \( \sigma_1 \) occurred first and then \( \sigma_2 \). If they have equal priorities, then the Arbitrary-Choice Principle applies.

The above two principles influence the actions taken upon the arrival of two or more simultaneous input signals. Similar principles are applicable to the case where a server becomes available, while two or more users are waiting for service. Specifically, we formulate the following two assignment rules.

(ACAR) - Arbitrary-Choice Assignment Rule

If a server becomes available, while two or more users are waiting for service, the system may assign the available server arbitrarily to any of the users waiting for service. Here, again, the behavior of the system becomes non-deterministic.

(PAR) - Priority Assignment Rule

Every user is assigned some non-negative priority. If a server becomes available, preference is given to the user on the waiting list having highest priority.

Some arbiter-like systems operate on the "first-come first-served" principle. However, in this paper we shall not be concerned with such systems. Arbiters in a telephone-system environment are also referred to as service request controllers. For the implementation of service request controllers using various assignment strategies see [BR-YO].
3. MIXED INHIBITOR NETS

An extensive literature is presently available demonstrating the suitability of Petri nets or related concepts to the modelling of discrete-event systems involving parallel processing [PA-DE, YOE, WEN, HE-YO, YO-BR, YO-BA].

In this section we introduce a modified version of Petri nets, namely Mixed Inhibitor Nets. In Sections 4 and 5 we demonstrate the suitability of Mixed Inhibitor Nets to the precise and convenient representation of arbitration networks.

A Mixed Inhibitor Graph (MIG) is a 5-tuple $G = \langle P, T, R, B, H \rangle$, where

- $P$ - is a finite set of places
- $T$ - is a finite set of transitions, disjoint from $P$
- $R$ - is a binary relation (Petri-net relation) on $P \cup T$ satisfying the condition $R \subseteq (P \times T) \cup (T \times P)$
- $B$ - is a (possibly empty) subset of $P$ (elements in $B$ are Boolean-type places; elements in $P - B$ are Integer-type places), and
- $H$ - is a binary relation (inhibiting relation) between $P$ and $T$ ($H \subseteq P \times T$).

The place $p \in P$ is an enabling (input) place of the transition $t \in T$ iff $p R t$, an inhibiting (input) place of $t$, iff $p H t$, and an output place of $t$, iff $t R p$.

Let $N$ denote the set of non-negative integers. A marking of the MIG $G = \langle P, T, R, B, H \rangle$ is a function $m: P \rightarrow N$, satisfying the condition

\[ \forall p \in B: m(p) \in \{0, 1\} \]
A Mixed Inhibitor Net (MIN) is an ordered pair <G, m> where G is an MIG and m is a marking of G. A place p of G is marked, iff \( m(p) > 0 \), and empty, iff \( m(p) = 0 \).

MIN's are conveniently represented by means of diagrams, an example of which is shown in Figure 2. The value of \( m(p) \) is shown by placing \( m(p) \) dots ("tokens") inside the circle representing p, or alternatively by writing the corresponding number inside the circle.

**Symbols**

- Transition: \( p \rightarrow t \)
- Boolean-type place: \( p \)
- Integer-type place: \( p \)

![Figure 2](image-url):

(a) Example of MIN;
(b) MIN obtained after firing \( t_1 \);
(c) MIN obtained after firing \( t_2 \).
System Interpretation of MIN

Petri nets are special cases of MIN's. The type of Petri net mostly discussed in the literature is an MIN with \( B=\emptyset \) and \( H=\emptyset \). Following [YOE] and [YO-BA] we refer to this type of net as "Integer-type Petri Net" (IPN). Similarly a "Boolean-type Petri Net" (BPN) is an MIN, where \( P=B \) and \( H=\emptyset \), and a "Mixed-type Petri Net" (MPN) is an MIN with \( H=\emptyset \) (cf. [WEN] and [YO-BA]).

Petri nets were intended to model discrete systems, with emphasis on such aspects as concurrency, resource conflict, deadlock, etc. However, Petri nets are not powerful enough to model arbiters which apply priority rules, whereas Inhibitor nets are (cf. [HACK]). This point will be discussed further later on.

In general, a Boolean-type place of an MIN represents a condition. The marking of the place indicates that the corresponding condition is satisfied. Integer-type places represent various kinds of counters. A transition \( t \) represents an event. This event may happen, if all enabling input places of \( t \) are marked, whereas all inhibiting input places of \( t \) are empty (i.e. their marking is zero). Mathematically precise rules on the "dynamics" of an MIN are given next.

Firing Rules for MIN's

We now proceed to specify transformation rules for marked MIN's. These rules correspond to the possible occurrences of events within the system represented.

Let \( <G,m> \) be a MIN, where \( G=\langle P,T,R,B,H \rangle \). The transition \( t \in T \) is firable (in \( <G,m> \)) iff all its enabling places are marked, whereas
all its inhibiting places are empty.

If \( t \) is firable, the firing of \( t \) consists in the following:

1. decrease by 1 the marking of its enabling places
2. increase by 1 the marking of all its integer-type output places
3. mark all its empty Boolean-type output places (i.e. change their marking from 0 to 1).

In the example of Figure 2(a), both transitions \( t_1 \) and \( t_2 \) are firable. If \( t_1 \) fires, the new marking is shown in Figure 2(b). If \( t_2 \) in Figure 2(a) fires, we obtain the marking shown in Figure 2(c).

We shall write \( m \xrightarrow{t} m' \) to indicate that the marking \( m' \) is obtained from the marking \( m \) by firing of the transition \( t \).

By a firing sequence for a M\( \text{MIN} <G,m> \), where \( G = <P,T,R,B,H> \), we mean a sequence (string) \( s \) of transitions

\[ s = t_1, \ldots, t_k \quad (k \geq 1) \]

such that there exist markings \( m_1, \ldots, m_k \) satisfying the conditions

\[ m \xrightarrow{t_1} m_1, \quad m_1 \xrightarrow{t_2} m_2, \quad \ldots, \quad m_{k-1} \xrightarrow{t_k} m_k. \]

Extending our notation accordingly, we shall write \( m \xrightarrow{s} m_k \).

We denote by \( T^+ \) the set of all finite strings (excluding the empty string) over \( T \).

For our purpose of modelling arbitration networks, we need the concept of labelled M\( \text{MIN} \)'s, which will be introduced next.
4. Labeled Mixed Inhibitor Nets

MIN's lack the facility of modelling the occurrence of input and output signals. In this section we extend the concept of MIN's to Labeled MINs, in order to incorporate this facility.

A Labeled Mixed Inhibitor Net (LMIN) consists of the following:

1) A MIN \( < G, m > \), where \( G = < P, T, R, B, H > \).
2) A finite input alphabet \( \Sigma \), together with a function \( f : T \rightarrow \Sigma \cup \{ \lambda \} \).
3) A finite output alphabet \( \Theta \), together with a partial function \( g \) from \( B \) to \( \Theta \).

With any LMIN we associate a behavior relation \( \beta \subseteq (2^\Sigma)^+ \times (2^\Theta)^* \), to be specified below.

We admit the possibility of the simultaneous appearance of several input signals. Hence we consider an input string \( x \) to be a string of subsets of \( \Sigma \).

However, we restrict ourselves to systems which satisfy the ACP. Consequently, we associate with any input string \( x \in (2^\Sigma)^+ \) the set \( \hat{x} \subseteq \Sigma^+ \) of all "corresponding" \( \Sigma \)-strings. More formally, let \( x = (x_1, \ldots, x_k) \), where \( x_i \subseteq \Sigma \), (\( 1 \leq i \leq k \)). Then \( w \in \hat{x} \) iff there exist repetition-free \( \Sigma \)-strings \( w_1, \ldots, w_k \) such that

\[
w = \prod_{i=1}^{k} w_i
\]

where \( \prod \) denotes concatenation, and \( \{ \sigma \mid \sigma \in w_i \} = x_i \), for \( 1 \leq i \leq k \).

In accordance with the ACP, the system will convert an input string
x ∈ (Σ^*)^+ into one of the corresponding Σ-strings w ∈ x.

Intuitively, if x β y, then the occurrence of an input string x may cause the appearance of the output string y = (y_1, ..., y_h), where h ≥ k, and y_i ∈ Θ, (1 ≤ i ≤ h).

Let M be the set of all possible markings of the MIG G of the given LMIN.

For m', m'' ∈ M, and σ ∈ Σ, we write

m' \xrightarrow{σ} m''

if the following conditions are satisfied:

1) There exists s ∈ T^+, such that

m' \xrightarrow{s} m''.

2) Let s = t_1, ..., t_k, k ≥ 1.

Then f(t_1) = σ, and f(t_i) = λ for 1 ≤ i ≤ k.

3) No transition t ∈ T-D(f) is firable in <G, m''>.

We also define a function \( \hat{g} : M \rightarrow 2^\Theta \) by

\[ \hat{g}(m) = \{ g(p) \mid p ∈ D(g) \land m(p) = 1 \} \].

Let now x ∈ (Σ^*)^+ and y ∈ (Θ^*)^+. Then x β y iff the following conditions are satisfied:
1) There exists a finite sequence of markings \( m = m_0, m_1, \ldots, m_h \) such that

\[
m_0 \xrightarrow{\sigma_1} m_1 \xrightarrow{\sigma_2} \ldots \xrightarrow{\sigma_h} m_h
\]

where \( (\sigma_1, \sigma_2, \ldots, \sigma_h) \in \hat{x} \).

2) \( y = (\hat{g}(m_1), \ldots, \hat{g}(m_h)) \).

5. FORMAL DESCRIPTIONS OF ARBITERS

In this section we apply the concept of LMIN to the formal description of arbiters.

The 1-server 2-user arbiter shown in Figure 1(a) may be represented by the LMIN of Figure 3. This representation, as well as Figure 1(b), is based on both the Arbitrary-Choice Principle (ACP) and the Arbitrary-Choice Assignment Rule (ACAR), discussed in Section 1.

Consider for example the input string \( x = ((1+, 2+), 1+, 2+) \). Then \( x \not\sim y^1 \), as well \( x \not\sim y^2 \), where

\[
y^1 = (\{z_1\}, \{z_1\}, \{z_2\}, \emptyset)
\]

and

\[
y^2 = (\{z_2\}, \{z_2\}, \{z_2\}, \emptyset).
\]
$\Sigma = \{1+, 2+, 1+, 2+\}$

$\theta = \{z_1, z_2\}$

$t_1^1/1+$ indicates that
$f(t_1^1) = 1+$

$p_3^1/z_1$ indicates that
$g(p_3^1) = \{z_1\}$

Figure 3: LMIN - Representation of 1-server, 2-user arbiter, based on ACP and ACAR.
The LMIN of Figure 4 represents a 1-server 3-user arbiter, based on ACP and the Priority Assignment Rule (PAR). We assume that user #1 has the highest priority, and user #3 has the lowest.

\[ E = \{It, 2t, 3t, 1+, 2+, 3+\} \]

\[ \Sigma = \{1+, 2+, 3+, 1+, 2+, 3+\} \]

\[ \Theta = \{z_1, z_2, z_3\} \]

**Figure 4:** LMIN – Representation of 1-server, 3-user arbiter, based on ACP and PAR.
To simplify the LMIN-representation of the general case, we introduce the notations of Figure 5.

Figure 5: Simplified notations for LMIN
(a) is replaceable by (b)
(c) is replaceable by (d).
We now consider \( m \)-server \( n \)-user arbiters, with the input and output leads shown in Figure 6. An LMIN-representation based on ACP and ACAR is shown in Figure 7. Each user has access to each server. If two or more servers are available to serve a request, the allocation is at random. Similarly, if two or more users compete for a single available server, the allocation is again at random.

An LMIN-representation of an \( m \)-server \( n \)-user arbiter, based on ACP and PAR, is shown in Figure 8. We assume that user \#1 has highest priority, and user \#n lowest priority.

Figure 6: External connections of general arbiter.
Figure 7: LMIN-representation of m-server, n-user arbiter, based on ACP and ACAR.
Figure 8: LMIN-representation of m-server, n-user arbiter, based on ACP and PAR.
CONCLUSIONS.

We have demonstrated the applicability of Labelled Mixed Inhibitor Nets to the behavioral descriptions of general arbiters. However, in order to describe arbiters based on the Priority Principle (PP), a more powerful model is required. Such a model is proposed in [BA-ST-YO].
REFERENCES


REFERENCES (cont'd)
