A Compositional Protocol Verification Using Relativized Bisimulation

INFORMATION AND COMPUTATION 99, 80–108 (1992)

A Compositional Protocol Verification Using Relativized Bisimulation

KIM G. LARSEN

Aalborg University Centre, Denmark

AND

ROBIN MILNER

Edinburgh University, Scotland

The purpose of this paper is to illustrate a compositional proof method for communicating systems; that is, a method in which a property P of a complete system is demonstrated by first decomposing the system, then demonstrating properties of the subsystems which are strong enough to entail property P for the complete system. In any compositional proof method, it is essential that one can abstract away the behavioural aspects of the subsystem which are irrelevant in the context of the complete system. Our method is an extension of the well established notion of bisimulation; it is called *relative bisimulation*, and was developed specifically to allow for such abstractions. We illustrate the method in a proof of correctness for a version of the Alternating Bit Protocol.

Contents

INTRODUCTION

The purpose of this paper is to illustrate a compositional proof method for communicating systems; that is, a method in which a property P of a

0890-5401/92 $5.00

complete system is demonstrated by first decomposing the system, then demonstrating properties of the subsystems which are strong enough to entail property P for the complete system. In any compositional proof method, it is essential that one can express the behavioural constraint which is imposed upon each subsystem by the others, since it may be difficult to demonstrate a suitable property of the subsystem's behaviour in the absence of the constraint.

Our method is an extension of the well established notion of bisimulation; it is called *relative bisimulation*, and was developed specifically to express the behavioural constraints between subsystems. We illustrate the method in a proof of correctness for a version of the Alternating Bit Protocol (AB protocol) in which the communication-lines are assumed to have unbounded buffering capacity, but may either duplicate or lose messages. For comparison, we first give a non-compositional proof by simulation and then outline the compositional proof by relative bisimulation. The full proof appears in Appendix B, whereas Appendix A contains a short review of some of the concepts and results on which our proof methodology is based.

The AB Protocol (Tanenbaum, 1981) is a commonly used example, and we here mention three other treatments of it. The proof of Schoone and van Leeuwen (1985) is mathematically elegant, and indeed treats the AB Protocol as a simple member of the family of sliding window protocols. The present proof, unlike Schoone and van Leeuwen's, is done in a formal model of communicating systems. We hope to follow them to extend our approach to the more complex protocols in this family.

Among proofs in formal models, we cite that of Bergstra and Klop (1984) and the more recent Koymans and Mulder (1987). Both are done in process algebra; the former is unrelativised and uses only algebraic laws, while we use the mildly non-algebraic technique of bisimulation. The latter is a compositional proof, more in the spirit of the present paper; it introduces concepts for modularisation which are adequate to treat the example but have yet to be studied in general terms.

1. Preliminaries

Adopting the *reactive view* on processes (Pnueli, 1985) we model processes and their operational behaviour as a labelled transition system $\mathscr{P} = (\mathrm{Pr}, \mathrm{Act}, \rightarrow)$, where Pr is the *set of processes* (states of processes), Act is the set of *actions* performed by processes, and $\rightarrow \subseteq \mathrm{Pr} \times \mathrm{Act} \times \mathrm{Pr}$ is the *transition relation*: $(P, a, Q) \in \rightarrow$ may be interpreted as "the process P is able to perform the action a and evolve to process Q." Also, we use the notation $P \xrightarrow{a} Q$ for $(P, a, Q) \in \rightarrow$.

We assume that Act holds a distinguished *unobservable* (internal) action τ. The set of actions $\text{Act}\backslash\{\tau\}$ is referred to as the set of *observable* (external) actions and it is assumed that for each observable action there exist an *inverse*, observable action $\bar{a}$, such that $\bar{\bar{a}} = a$.

We assume that processes may be composed freely using the operators of CCS (Milner, 1980, 1989), including the following:

— the constant *prefixing* operator $a.$ for each $a \in \text{Act}$,

— a binary operator $+$ representing *nondeterministic choice*,

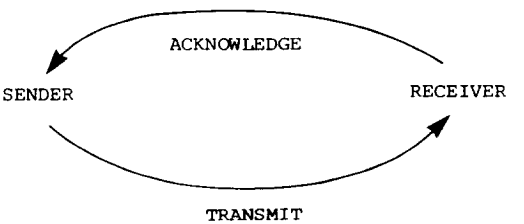
— a binary operator $|$ representing *parallel* composition, where $P|Q$ interleaves the behaviours of P and Q with possible communication (synchronization) on complementary (i.e., mutually inverse) actions,

— a unary *restriction* operator $\backslash L$ for $L \subset \text{Act}\backslash\{\tau\}$, where $P\backslash L$ behaves like P but with all actions of L and $\bar{L} = \{\bar{a} \mid a \in L\}$ disallowed.

Finally, we allow processes to be specified recursively either through (simultaneous) recursive equations or using an explicit recursion construct Fix. We urge the reader to consult Milner (1980, 1989) for more intuition and a formal presentation of CCS and its operational semantics.

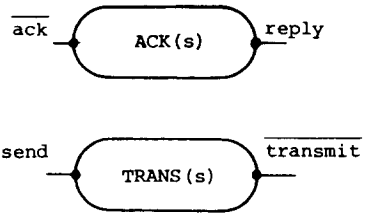
2. Description of the Alternating Bit Protocol

The protocol consists of two systems, which we temporarily call SENDER and RECEIVER, connected by two communication-lines—one for transmission of messages and one for acknowledgement of messages:



The communication-lines are assumed here to hold an unbounded number of messages (acknowledgements) each consisting of a *quantum* together with a boolean value. The lines are assumed to be faulty to the extent that they may either *lose* or *duplicate* any message or acknowledgement at any time. To present our technique as clearly as possible, we ignore the content of messages; i.e., we take a message or an acknowledgement to consist of a single boolean value. However, it is straightforward to redo the analysis in a scenario where the content of messages is included (using the "data-

independence" results of Wolper (1986) and Wolper and Lovinfasse (1989), it should suffice to carry out the analysis in a setting with only three different messages).



Thus, we model each communication line as a process parameterised with the sequence of messages (acknowledgements) that it is currently holding. Using the internal action τ to model loss and duplication of messages (acknowledgments) the transitions of these processes are as follows:

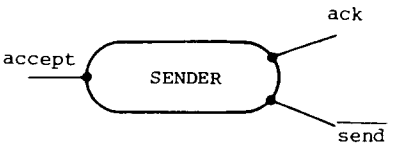
$$\mathrm{ACK}(bs) \xrightarrow{\overline{\mathrm{ack}}(b)} \mathrm{ACK}(s) \qquad \mathrm{ACK}(s) \xrightarrow{\mathrm{reply}(b)} \mathrm{ACK}(sb)$$

$$\mathrm{ACK}(sbt) \xrightarrow{\tau} \mathrm{ACK}(st) \qquad \mathrm{ACK}(sbt) \xrightarrow{\tau} \mathrm{ACK}(sbbt)$$

$$\mathrm{TRANS}(s) \xrightarrow{\mathrm{send}(b)} \mathrm{TRANS}(bs) \qquad \mathrm{TRANS}(sb) \xrightarrow{\overline{\mathrm{transmit}}(b)} \mathrm{TRANS}(s)$$

$$\mathrm{TRANS}(tbs) \xrightarrow{\tau} \mathrm{TRANS}(ts) \qquad \mathrm{TRANS}(tbs) \xrightarrow{\tau} \mathrm{TRANS}(tbbs)$$

Turning to the SENDER and RECEIVER, we model them in a way which exhibits the duality between them. The SENDER has two states (parameterised on a boolean value): ACKED(b), in which an acknowledgement "b" has just been received, and SENDING(b), in which a message "b" is being repeatedly transmitted. The SENDER may be diagrammed as



indicating the ports at which it accepts quanta from the communication source, sends messages on the TRANS line, and receives acknowledgements

on the ACK line. Its definition in CCS [Milner, 1980] is as follows, where we represent boolean complementation by ($\hat{}$):

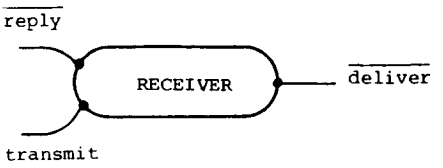
$$\begin{aligned}\mathrm{ACKED}(b) &= \mathrm{ack}(b).\mathrm{ACKED}(b) + \mathrm{ack}(\hat{b}).\mathrm{ACKED}(b) \\ &\quad + \mathrm{accept}.\mathrm{SENDING}(\hat{b}) \\ \mathrm{SENDING}(b) &= \overline{\mathrm{send}}(b).\,\mathrm{SENDING}(b) \\ &\quad + \mathrm{ack}(\hat{b}).\mathrm{SENDING}(b) \\ &\quad + \mathrm{ack}(b).\mathrm{ACKED}(b).\end{aligned}$$

For distinct actions $a_1, ..., a_n$ and p a process expression we write $(a_1 + ... + a_n).p$ for $a_1.p + ... + a_n.p$ and $(a_1 ... a_n).p$ for $a_1 .... a_n.p$ whenever $n \geqslant 1$. Then, for s a non-empty sequence of actions we write $s^*.p$ for the expression FIX x. $(s.x + p)$, where the variable x is chosen so that no free variable of p is captured. With the above abbreviations we may now write the definitions of ACKED and SENDING more conveniently as follows:

$$\begin{aligned}\mathrm{ACKED}(b) &= (\mathrm{ack}(b) + \mathrm{ack}(\hat{b}))^*.\mathrm{accept}.\mathrm{SENDING}(\hat{b}) \\ \mathrm{SENDING}(b) &= (\overline{\mathrm{send}}(b) + \mathrm{ack}(\hat{b}))^*.\mathrm{ack}(b).\mathrm{ACKED}(b).\end{aligned}$$

Note that ACKED(b) ignores any acknowledgement ack(a), whatever a. Note also that SENDING(b) may repeat $\overline{\mathrm{send}}(b)$ arbitrarily often; in a more refined model we would try to reflect that repeated transmissions are in response to a time-out signal, but we believe that this refinement only adds a minor complexity to our analysis–which we prefer to avoid for the sake of clarity.

Dually, the RECEIVER has two states: TRANSMITTED(b), in which a message "b" has just been received, and REPLYING(b), in which an acknowledgement "b" is repeatedly emitted. It may be diagrammed

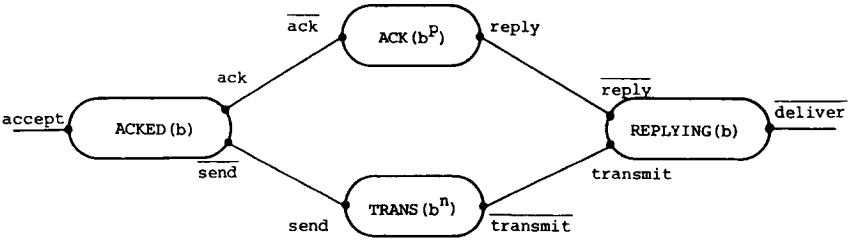


indicating its ports—in particular where it delivers quanta to the communication target. Its definition is as follows:

$$\text{TRANSMITTED}(b)$$
$$= (\text{transmit}(b) + \text{transmit}(\hat{b}))^*.\overline{\text{deliver}}.\text{REPLYING}(b)$$
$$\text{REPLYING}(b)$$
$$= (\overline{\text{reply}}(b) + \text{transmit}(b))^*.\text{transmit}(\hat{b}).\text{TRANSMITTED}(\hat{b}).$$

Note the duality with the SENDER; similar remarks apply about the ignoring of further transmissions and about repeated replies. Finally, note that the quanta accepted from source and delivered to target are completely omitted here; their incorporation present no problems, but their absence yields greater clarity.

In the initial state of the complete system, we may take the SENDER and RECEIVER to be in states ACKED(b) and REPLYING(b), which actually reflects that a quantum, transmitted with associated bit "b," has been both delivered and acknowledged and the system is ready to accept a new quantum. In a truly initial state, both lines would be empty, but we are concerned more generally with the state in which both lines contain a sequence b^n, of arbitrary length $n \geqslant 0$, of the same message "b," corresponding to residual copies of old messages and acknowledgements.



The corresponding expression in CCS, whose sort (i.e., set of external port names) is $\{\text{accept}, \overline{\text{deliver}}\}$, is

$$\text{SYSTEM}(b, n, p) = \text{ACKED}(b) \| \text{TRANS}(b^n) \| \text{ACK}(b^p) \| \text{REPLYING}(b).$$

Recall that in CCS a restricted composition of the form $(P \mid Q)\backslash A$, where A is the set of port names used for communication between P and Q, is often abbriviated as $P \| Q$."$\|$" is not associative in general, but in our restrained use it is so—which is a consequence of the property of our system that no two ports in the system are identically named.

3. Proof of Correctness

The content of messages being ignored, the specification of our system is that it should behave exactly like a quantum buffer of capacity one. That is,

$$\text{SPEC} = \text{accept}.\text{deliver}.\text{SPEC}$$

and we wish to prove that

$$\text{SYSTEM}(b, n, p) \approx \text{SPEC},$$

where "$\approx$" is observation equivalence. This equivalence relation is studied in Milner (1983) and Park (1981), where it is demonstrated that, to prove $P \approx Q$, it is sufficient to exhibit a bisimulation relation $\mathscr{B}$ between agents such that $(P, Q) \in \mathscr{B}$. Techniques for establishing such relations are described for example in (Prasad, 1989), and often the relations may be constructed mechanically (Larsen, 1986). However, we shall not give here the details which justify its existence in this case; we are mainly concerned to present it and to support it with some intuition.

What makes our example tractable is that the state-space, consisting of all possible combinations of states of our four agents, is for the most part inaccessible from the initial state SYSTEM(b, n, p). In particular, the TRANS and ACK lines can at most contain sequences of messages of the form $b^m\hat{b}^n$. In fact, the following bisimulation $\mathscr{B}$ is sufficient, and covers all accessible states of our system:

$$\mathscr{B} = \mathscr{B}_1 \cup \mathscr{B}_2 \cup \mathscr{B}_3 \cup \mathscr{B}_4$$

$$\mathscr{B}_1 = \{(\text{ACKED}(b) \| \text{TRANS}(b^n) \| \text{ACK}(b^n) \| \text{REPLYING}(b), \text{SPEC}); n, p \geqslant 0\}$$

$$\mathscr{B}_2 = \{(\text{SENDING}(\hat{b}) \| \text{TRANS}(\hat{b}^m b^n) \| \text{ACK}(b^n) \| \text{REPLYING}(b), \text{SPEC}'); m, n, p \geqslant 0\}$$

$$\mathscr{B}_3 = \{(\text{SENDING}(\hat{b}) \| \text{TRANS}(\hat{b}^m) \| \text{ACK}(b^p) \| \text{TRANSMITTED}(\hat{b}), \text{SPEC}'); m, p \geqslant 0\}$$

$$\mathscr{B}_4 = \{(\text{SENDING}(\hat{b}) \| \text{TRANS}(\hat{b}^m) \| \text{ACK}(b^p\hat{b}^q) \| \text{REPLYING}(\hat{b}), \text{SPEC}); m, p, q \geqslant 0\}.$$

Here $\text{SPEC}' = \overline{\text{deliver}}.\text{SPEC}$. To confirm that $\mathscr{B}$ is indeed a bisimulation it is, as required by the definition of bisimulation, enough to show that whenever $(P, Q) \in \mathscr{B}$ then

(i) Whenever $P \stackrel{u}{\Longrightarrow} P'$, then $Q \stackrel{u}{\Longrightarrow} Q'$ for some Q', and $(P', Q') \in \mathscr{B}$

(ii) Whenever $Q \stackrel{u}{\Longrightarrow} Q'$, then $P \stackrel{u}{\Longrightarrow} P'$ for some P', and $(P', Q') \in \mathscr{B}$

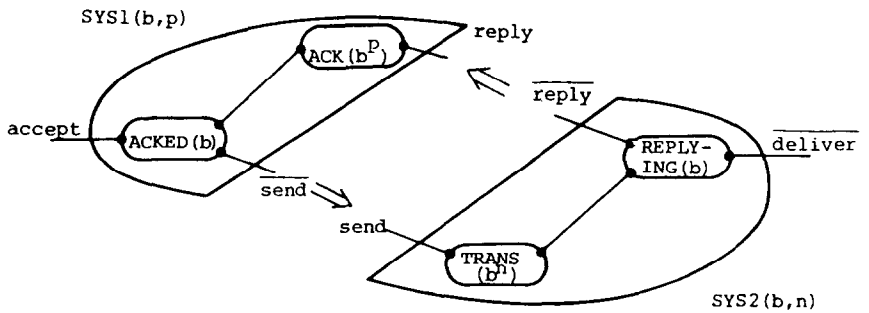
Here u stands for any sequence (possibly empty) of external actions and $\stackrel{u}{\Longrightarrow}$ allows the admixture of arbitrarily many internal communications. Formally, $\stackrel{\varepsilon}{\Longrightarrow}$ is the reflexive and transitive closure of $\stackrel{\tau}{\longrightarrow}$, and for

$u = a_1, ..., a_n \in (\mathrm{Act}\backslash\{\tau\})^+$ $\overset{u}{\Longrightarrow}$ is the relation $\overset{\varepsilon}{\Longrightarrow} \overset{a_1}{\longrightarrow} \overset{\varepsilon}{\Longrightarrow} \overset{a_2}{\longrightarrow} \cdots \overset{\varepsilon}{\Longrightarrow} \overset{a_n}{\longrightarrow} \overset{\varepsilon}{\Longrightarrow}$ where juxtaposition denotes composition of relations. The verification of this property of $\mathscr{B}$ is amenable to mechanical assistance, but in the present case it is not too tedious to be carried out with pencil and paper. However, the process is error-prone, and it is natural to demand that any such demonstration be done with machine assistance if a practical engineering design depends upon it.

4. Factoring the Problem

A defect of the foregoing proof is that it involves an analysis of the complete system directly in terms of its basic components, without mediating analysis of any subsystem. This defect is not serious in the present example, since the accessible states of the complete system are easily divided into four classes, those described in the four subrelations of the bisimulation relation $\mathscr{B}$, and thus the bisimulation is not tedious to exhibit. However, larger systems would often suffer a combinatorial explosion under similar treatment; we shall only be able to "scale up" our method of proof if we can tackle large systems by stages, grouping basic components into subsystems whose behaviour can be described with reasonable simplicity, then combining these subsystems—perhaps in larger subsystems—eventually into the complete system. We proceed to investigate this approach using the present example. The total effort in proving correctness will not be decreased— perhaps it will even be increased. However, we shall be able to illustrate a technique developed in Larsen (1986), which holds promise for larger systems, where we may indeed hope to reduce the effort of proof and also gain greater understanding of the systems.

Consider the following decomposition of SYSTEM(b, n, p) into two subsystems SYS1(b, p) and SYS2(b, n):



From the analysis in Section 3, it seems that the precise number of messages on the transmission lines is not important. Thus, we may hope to find simply defined agents SPEC1(b) and SPEC2(b) such that

(a) SYS1(b, p) $\approx$ SPEC1(b)

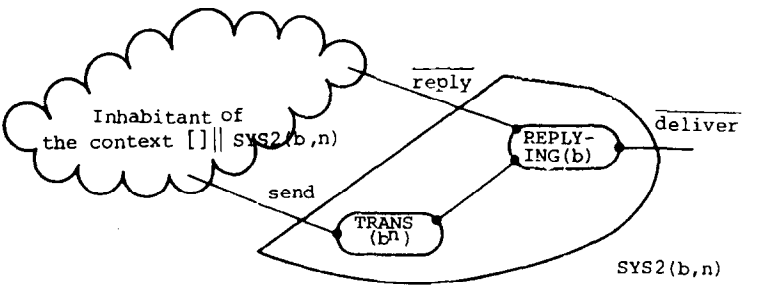
(b) SYS2(b, n) $\approx$ SPEC2(b).

Then, using the congruence property of $\approx$ w.r.t. $\|$, we may complete our proof of correctness (i.e., SYSTEM(b, n, p) $\approx$ SPEC) by proving

$$\text{SPEC1}(b) \| \text{SPEC2}(b) \approx \text{SPEC}.$$

Although this approach nicely factors our proof, it is thwarted by the fact that the subsystems SYS1(b, p) and SYS2(b, n) have very complicated behaviours. Thus the subspecifications SPEC1(b) and SPEC2(b) cannot be simple.

However, only a very minor part of the total behaviour of SYS1(b, p) (for example) will be permitted in the context[1] [ ] $\|$ SYS2(b, n). We therefore try to relax the condition of (a) by looking for an agent SPEC1(b) which need not be exactly equivalent to SYS1(b, p), but is indistinguishable from it under the limitations of the context [ ] $\|$ SYS2(b, n) in which this agent finds itself. For this purpose we shortly introduce the notions of *relative bisimulation* and *relative observation equivalence*.



The context [ ] $\|$ SYS2(b, n) imposes certain constraints on its inhabitant, with respect to $\overline{\text{send}}$ and reply actions (see above figure). Intuitively, the behaviour allowed by an inhabitant must satisfy the following:

Phase 1. All reply actions must be reply(b) until $\overline{\text{send}}(\hat{b})$ occurs

Phase 2. Trivially, all reply actions are reply(b) until reply($\hat{b}$) occurs

Phase 3. Thereafter, provided $\overline{\text{send}}(b)$ has *not* occurred during PHASE 2, the whole constraint applies again with b and $\hat{b}$ interchanged.

We now proceed to formalize these constraints.

[1] A *context*, C, is a CCS term with a "hole," [ ], in it. For P a CCS term, we use $C[P]$ to denote the term obtained by substituting P for [ ] in C.

5. Environments and Relativized Bisimulation

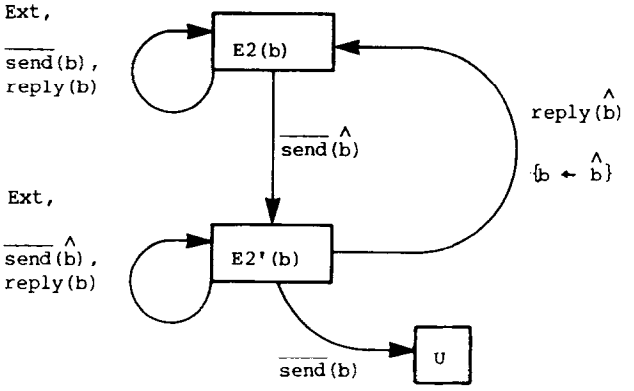
Formally we express the limitations imposed by contexts on their inhabitants in terms of *environments*. Operationally environments are objects which consume actions produced by their inhabitants. However, an environment's ability to consume actions might be limited, thereby limiting the behaviour of any inhabitant. We describe the consuming behaviour of environments in terms of a labelled transition system $\xi = (\text{Env}, \text{Act}, \rightarrow)$, where $\rightarrow \subseteq \text{Env} \times \text{Act} \times \text{Env}$ is the *consumption relation*. For $(E, a, F) \in \rightarrow$ we write $E \xrightarrow[a]{} F$ which is to be read: "the environment E is able to consume the action a, and change into the environment F." Note, that in order to avoid too much confusion with processes we have placed the action below the arrow for environments. This is mainly for clarity rather than for necessity; in the present application the environments used will actually be processes.

The limitations imposed by the context $[\,] \| \text{SYS2}(b, n)$ can now be described by the following environment, using U as the universal environment (i.e., $U \xrightarrow[a]{} U$ for all a in Act) and $\text{Ext} = \text{Act} - \{\text{send}, \overline{\text{send}}, \text{reply}, \overline{\text{reply}}\}$:

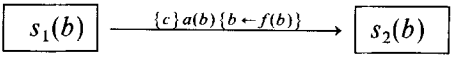
$$E2(b) = (\text{Ext} + \overline{\text{send}}(b) + \text{reply}(b))^*.\overline{\text{send}}(\hat{b}).E2'(b)$$

$$E2'(b) = (\text{Ext} + \overline{\text{send}}(\hat{b}) + \text{reply}(b))^*.(\overline{\text{send}}(b).U + \text{reply}(\hat{b}).E2(\hat{b})).$$

The following is a graphical representation of E2(b):



To read the above diagram some additional explanation is needed. For each possible value of b a box in the diagram represents a set of agents (environments). An arrow between two boxes is labelled



by a triple, $\{c\}\ a(b)\ \{b \leftarrow f(b)\}$, where c is a *precondition* on agents, $a(b)$ is a parameterized action (where actions in the above diagram are pairs), and $b \leftarrow f(b)$ is a *postassignment*, with f being a function on parameter values. For $c = \text{true}$ we omit the precondition $\{c\}$, and similarly for f the identity function we omit the postassignment $\{b \leftarrow f(b)\}$. Formally, the arrow between the boxes is to be interpreted as follows: for all b and all agents P in $S_1(b)$, whenever P satisfies condition c, there exists an agent Q in $S_2(f(b))$ such that $P \xrightarrow{a(b)} Q$.

Note that in the environment $E2'(b)$ a $\overline{\text{send}}(b)$ action is allowed after which there will be no restriction on the inhabitant's behaviour (expressed by the environment U). However, it turns out that the agent $\text{SYS1}(b, p)$, whenever it finds itself in the environment $E2'(b)$, will be unable to perform $\overline{\text{send}}(b)$.

To express the equivalence of two processes under environmental constraint, we introduce the notion of *relative bisimulation* developed in (Larsen, 1985, 1986). Given an environment system $\xi = (\text{Env}, \text{Act}, \rightarrow)$ a relative bisimulation $\mathcal{RB}$ consists of a family $\mathcal{RB}_E (E \in \text{Env})$ of relations such that whenever $(P, Q) \in \mathcal{RB}_E$ and $E \underset{u}{\Longrightarrow} E'$ then

(i) Whenever $P \overset{u}{\Longrightarrow} P'$, then $Q \overset{u}{\Longrightarrow} Q'$ for some Q', and $(P', Q') \in \mathcal{RB}_{E'}$

(ii) Whenever $Q \overset{u}{\Longrightarrow} Q'$, then $P \overset{u}{\Longrightarrow} P'$ for some P', and $(P', Q') \in \mathcal{RB}_{E'}$.

Again, u stands for some sequence (possibly empty) of external actions, and $\overset{u}{\Longrightarrow}$ as well as $\underset{u}{\Longrightarrow}$ allow arbitrarily many intermediate internal actions.

Intuitively, this is just like bisimulation except that only those "moves" (sequences of external actions) permitted by the environment E are considered; we do not care how the agents may perform for "moves" not permitted. Clearly, the simpler notion of bisimulation is just a relative bisimulation in which the environment system ξ consists of a single environment—the universal environment which allows any action at any time.

We say that P and Q are *observationally equivalent relative to* E, and write $P \approx_E Q$, if there is a relative bisimulation $\mathcal{RB}$ such that $(P, Q) \in \mathcal{RB}_E$.

6. Factoring the Proof

We can now see that our proof can be factored into parts which are described informally as follows:

(1) Find an agent $\text{SPEC1}(b)$ which is observationally equivalent to $\text{SYS1}(b, p)$ *relative* to the environment $E2(b)$.

(2) Show that the behaviour permitted by the context $[\,] \| \mathrm{SYS2}(b, n)$ is "contained" in the environment $E2(b)$.

(3) Find an environment $E1(b)$ which "contains" the behaviour permitted by the context $\mathrm{SPEC1}(b) \| [\,]$.

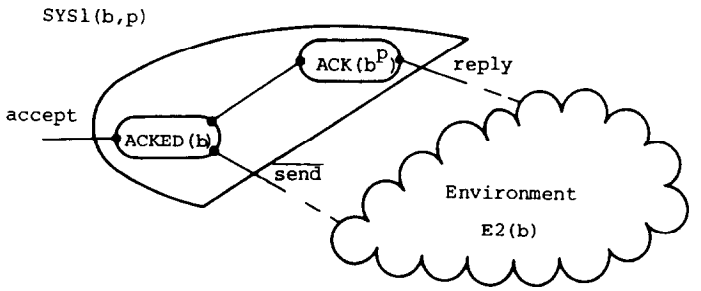
(4) Find an agent $\mathrm{SPEC2}(b)$ which is observationally equivalent to $\mathrm{SYS2}(b, n)$ *relative* to the environment $E1(b)$.

From (1) and (2) it follows that $\mathrm{SYSTEM}(b, n, p) = \mathrm{SYS1}(b, p) \| \mathrm{SYS2}(b, n) \approx \mathrm{SPEC1}(b) \| \mathrm{SYS2}(b, n)$ and from (3) and (4) it follows that $\mathrm{SPEC1}(b) \| \mathrm{SYS2}(b, n) \approx \mathrm{SPEC1}(b) \| \mathrm{SPEC2}(b)$. Thus to complete the proof of $\mathrm{SYSTEM}(b, n, p) \approx \mathrm{SPEC}$ it suffices to show that

(5) $\mathrm{SPEC1}(b) \| \mathrm{SPEC2}(b) \approx \mathrm{SPEC}$.

We focus on making (1) and (2) precise, since (3) and (4) are duals and (5) is already familiar. For (1) we search for an agent $\mathrm{SPEC1}(b)$ such that

$$\mathrm{SYS1}(b, p) = \mathrm{ACKED}(b) \| \mathrm{ACK}(b^p) \approx_{E2(b)} \mathrm{SPEC1}(b)$$



An intuition which guides the search is that, in the environment $E2(b)$, the index p in $\mathrm{ACK}(b^p)$ is immaterial—it has no effect on the behaviour of the subsystem. Furthermore, the environment $E2(b)$ ensures that no $\mathrm{reply}(\hat{b})$ can occur until $\mathrm{ACKED}(b)$ has done accept followed by $\overline{\mathrm{send}}(\hat{b})$. Thereafter, $\mathrm{SYS1}(b, p)$ ensures that no $\overline{\mathrm{send}}(b)$ can occur until the environment allows $\mathrm{reply}(\hat{b})$. Together, they ensure that the ACK line can only hold sequences with at most a single bit-change. We can prove

$$(*) \quad \mathrm{SYS1}(b, p) \approx_{E2(b)} \mathrm{SPEC1}(b),$$

where

$$\mathrm{SPEC1}(b) = \mathrm{reply}(b)^*.\mathrm{accept}.\mathrm{SPEC1}'(b)$$

$$\mathrm{SPEC1}'(b) = (\overline{\mathrm{send}}(\hat{b}) + \mathrm{reply}(b))^*.\mathrm{reply}(\hat{b}).\mathrm{SPEC1}''(b)$$

$$\mathrm{SPEC1}''(b) = (\overline{\mathrm{send}}(\hat{b}) + \mathrm{reply}(\hat{b}))^*.(\tau.\mathrm{SPEC1}'(b) + \tau.\mathrm{SPEC1}(\hat{b})).$$

We do not pretend that every detail of this definition is immediately obvious; however, a careful consideration of the subsystem leads one to propose a specification more or less of this form. In fact, because of the environment $E2(b)$, there seem to be several alternative definitions of SPEC1(b). Note in particular the τ-moves at the end of the last equation. The first τ-move represents a transition where the ACK line loses the first copy of a new acknowledgement; therefore a new copy of it has to be sent to the ACK line before it can reach the SENDER. The second τ-move represents the eventual acknowledgement ack(b), which occurs as a communication within the subsystem. Note also that SPEC1(b) is finite-state; the infinite state-space resulting from the unbounded indices p, q in ACK(b^p), ACK($b^p\hat{b}^q$) has been collapsed to a finite state-space. The relative bisimulation which now establishes the required equivalence (*) is not hard to exhibit and can be found in Appendix B.

Part (2) of our proof obligation, which is to show that the behaviour permitted by the context $[\ ] \| \text{SYS2}(b, n)$ is "contained" in the environment $E2(b)$, must now be tackled.

We prove that the behaviour permitted by a context C is contained in that permitted by an environment E as follows:

We first produce an environment E' (called $\mathcal{WIE}(C, U)$) by manipulating the transition rules for the context C. In many situations E' will actually be the weakest environment that will serve as E. We then show that the behaviours of E' are permitted by E by exhibiting a simulation relation from E' to E. The theory that justifies this method is developed at length in (Larsen, 1986) and summarized in Appendix A.

In the present example, we are interested in the context $\mathcal{C} \equiv [\ ] \| \text{SYS2}(b, n)$, and thus obliged to establish a simulation relation between $\mathcal{WIE}(\mathcal{C}, U)$ and $E2(b)$. It turns out that we can prove the above simulation ordering without explicit calculation of $\mathcal{WIE}(\mathcal{C}, U)$. The details of the proof may be found in Appendix B.

7. Related and Future Work

In the sequential case, Hoare Logic (Hoare, 1969) provides a well-established theory for compositional reasoning about programs as relations (between initial and final state).

The early work by Owicki and Gries (1976) extends Hoare Logic to parallel programs by observing that the Hoare triple should not be viewed just as an independent object but as the conclusion of a proof which carries the extra information required to obtain a compositional rule for parallel composition. The rule for parallel composition suggested by Owicki and Gries is essentially a *conjunction* (of both the pre- and the postcondition)

of the parallel component specifications together with a test for *interference freedom* of their proof trees: i.e., any precondition and final postcondition appearing in one should be "invariant" under the atomic actions of the other. As the test for interference freedom requires access to the implementation of the components (and not just their specifications), the parallel rule of Owicki and Gries is non-compositional.

The work by Stirling (1986) gives a compositional reformulation of Owicki and Gries's proof system using a relativized (with information about the pre- and postconditions and the "invariants") consequence relation. The information used by Stirling in the relativization is similar to the *rely and guarantee conditions* of Jones' rigorous development methods for parallel programs in (Jones, 1983). Here, in addition to pre- and postconditions, the *specification* of the components of a parallel composition contains a rely and guarantee pair. The proof rule for parallel composition then involves a test for *compatibility*: essentially it must be verified that the guarantee conditions of one component imply the rely condition of the other. The work by Stark (1985) provides alternative compositional proof rules for specifications containing rely and guarantee conditions. De Roever (1985) gives a useful survey of a number of compositional proof methods, including many of the above mentioned.

The work by Barringer, Kuiper, and Pnueli (1984), Zhou and Liu (1987), and more recent work by Lamport (1989) considers compositional proof rules using Temporal Logic as the basic specification formalism. In all three cases the specification of a component contains explicit information as to environments commitment: one only need specify how the component should behave in its *intended environment.*

Also the development methods by Lynch and Merrit (1986), Lynch and Tuttle (1987), and Chandy and Misra (1988) involve relativized specifications (in Lynch's school this is expressed in properties of "well-formed" schedules, in Chandy and Misra's book by "conditional properties").

Thus, it seems to be commonly agreed that the interference or interaction between the components of a parallel program should be *explicitly* reflected in their specifications. The proof methodology presented in this paper can be seen as carrying out this "paradigm" in the framework of Process Algebra (Milner, 1980, 1989; Hoare, 1978; Bergstra and Klop, 1985; Boudal, 1985) and in particular in the framework of CCS (Milner, 1980, 1989).

With the purpose of identifying the state-of-the-art in the area of compositional proof methods for parallel systems a REX workshop was recently held in Holland sponsored by the Dutch NFI Programme. We strongly urge the reader to consult the proceedings of this workshop (de Bakker, de Roever, and Rozenberg, 1989).

Obvious extensions of our method includes relativization of other

process equivalences (such as failure equivalence, testing equivalence, and acceptance equivalence) and, perhaps more interesting, preorders. Clearly, the simulation preorder can be relativized in a way similar to the relativization of bisimulation. However, as the simulation ordering is deadlock insensitive in the framework of CCS, we have not pursued this idea. A more promising (preorder) candidate for relativization is the divergence sensitive version of bisimulation (Milner, 1981; Walker, 1988). Another topic for future work is the extension of our method to contexts with several "holes." Work in that direction may be found in Larsen and Xinxin (1989).

APPENDIX A: REVIEW OF THEORY

In the present appendix we give a short review of some of the concepts and results from (Larsen, 1986) on which our proof methodology is based.

Recall that an environment F is a *sufficient inner environment* ($\mathcal{SIE}$) for a context $\mathcal{C}$ in an environment E provided $\mathcal{C}[P] \approx_E \mathcal{C}[Q]$ whenever $P \approx_F Q$. However, in order for this concept to be of any practical use in a compositional proof methodology, we need to provide an effective test for the $\mathcal{SIE}$ property. A guide to such a test is given by the following main theorem.

First, a *simulation* $\mathcal{S}$ is a relation between agents such that whenever $(P, Q) \in \mathcal{S}$ then condition (i) of bisimulation holds. Whenever (P, Q) is contained in some simulation $\mathcal{S}$ we write $P \lesssim Q$ and say that P is simulated by Q. It can easily be shown that $\lesssim$ is a preorder with NIL (the completely inactive agent) as minimal element and U as maximal element. Now, $\lesssim$ is related to relative observational equivalence through the following important theorem:

THEOREM 1. *$E \lesssim F$ implies*

$$\text{For all } P \text{ and } Q, \text{ if } P \approx_F Q \text{ then } P \approx_E Q.$$

Moreover, if ξ is image-finite in the sense that the set $\{F \mid E \stackrel{u}{\Longrightarrow} F\}$ is finite for all E and u, then the reverse implication holds as well.

From the theorem it follows that if $F \lesssim G$, where F is already known to be a $\mathcal{SIE}$ for $\mathcal{C}$ in E, then G is also a $\mathcal{SIE}$ for $\mathcal{C}$ in E. In the following we present a construction from (Larsen, 1986) of a $\lesssim$-weakest environment, $\mathcal{WIE}(\mathcal{C}, E)$, with a slightly stronger property than that of $\mathcal{SIE}$. This clearly makes the simulation $\mathcal{WIE}(\mathcal{C}, E) \lesssim F$ a valid, though not complete, test for the $\mathcal{SIE}$-property of F w.r.t. $\mathcal{C}$ in E. Moreover, this test is amenable to a proof technique similar to that of bisimulation.

The construction of $\mathscr{WIE}(\mathscr{C}, E)$ is facilitated by a *new operational semantics of contexts* in terms of action transducers; i.e., a context is viewed as an object which consumes actions from its inhabitants and produces actions for a surrounding environment. Formally, we describe the operational semantics of contexts in terms of a labelled transition system of the form $\mathscr{CC} = (\mathrm{Con}, (\mathrm{Act} \cup \{0\}) \times \mathrm{Act}, \rightarrow)$, where $0 \notin \mathrm{Act}$ is a special no-action symbol allowing a context to produce (outer) actions without consulting the inhabitant. We write $\mathscr{C} \xrightarrow{b}_{a} \mathscr{C}'$ for $(\mathscr{C}, (a, b), \mathscr{C}') \in \rightarrow$ and read $\mathscr{C} \xrightarrow{b}_{0} \mathscr{C}'$ and $\mathscr{C} \xrightarrow{b}_{a} \mathscr{C}'$ $(a \neq 0)$ in the obvious ways.

EXAMPLE 2. The operational semantics of contexts of the form $([\,]\,|\,P)\backslash A$, where P is a process expression and $A \subseteq \mathrm{Act} - \{\tau\}$, is given by the following rules:

(i) $$([\,]\,|\,P)\backslash A \xrightarrow{a}_{a} ([\,]\,|\,P)\backslash A; \qquad a \notin A \cup \bar{A}$$

(ii) $$\frac{P \xrightarrow{a} P'}{([\,]\,|\,P)\backslash A \xrightarrow{a}_{0} ([\,]\,|\,P')\backslash A}; \qquad a \notin A \cup \bar{A}$$

(iii) $$\frac{P \xrightarrow{a} P'}{([\,])\,|\,P)\backslash A \xrightarrow{\tau}_{\bar{a}} ([\,]\,|\,P')\backslash A}.$$

(i) allows the inhabitant to perform any action not in $A \cup \bar{A}$ alone; (ii) makes it possible for the context to perform without consulting the inhabitant; and finally in (iii) the context may produce a τ-action as a result of an internal communication between the inhabitant (contributing $\bar{a}$) and P (contributing a). Actually, the three rules above are derived rules of a complete system in (Larsen, 1986) describing the operational semantics of all standard CCS-contexts.

The operational semantics of a context $\mathscr{C}$ must be related to the operational semantics of combined processes $\mathscr{C}[P]$ in the following way:

$$\mathscr{C}[P] \xrightarrow{b} R \quad \Leftrightarrow \quad \begin{array}{l} \textit{Either} \text{ there exists } \mathscr{D} \text{ s.t. } \mathscr{C} \xrightarrow{b}_{0} \mathscr{D} \\ \text{and } R = \mathscr{D}[P] \textit{ or} \text{ there exist } \mathscr{D}, \\ Q, \text{ and } a \neq 0 \text{ s.t. } \mathscr{C} \xrightarrow{b}_{a} \mathscr{D}, \\ P \xrightarrow{a} Q, \text{ and } R = \mathscr{D}[Q]. \end{array}$$

Fortunately, it is easily cheeked that the operational semantics for contexts of the form $([\,]\,|\,P)\backslash A$ given in Example 2 indeed has this property.

It is a well-known fact that $\approx$ is *not* preserved by all CCS-contexts, especially not sum-contexts (see Milner, 1980). From this fact it can be deduced that $\mathscr{WIE}(\mathscr{C}, E)$ cannot possibly exist for all contexts $\mathscr{C}$. However, based on the new operational semantics of contexts, a condition

may be formulated which ensures not only the preservation of $\approx$, but also the existence of $\mathcal{WIE}(\mathcal{C}, E)$ for all environments E. The condition is that of being *idle-preserving*, which is a property a context $\mathcal{C}$ enjoys provided for all actions a and contexts $\mathcal{D}$:

(i) $\mathcal{C} \xrightarrow{a}_{\tau} \mathcal{D}$ iff $a = \tau$ and $\mathcal{C} = \mathcal{D}$

(ii) All $\mathcal{C}$'s derivatives are idle-preserving.

Thus, an idle-preserving context can neither prevent nor detect a τ-action produced by its inhabitant, and is therefore unable to destinguish between observational equivalent processes. From Example 2 it is clear that all contexts of the form $([\ ] \mid P)\backslash A$ are idle-preserving.

Though $\mathcal{WIE}(\mathcal{C}, E)$ always exists when $\mathcal{C}$ is idle-preserving, its construction may be somewhat simplified by assuming the environment E to be *idle* (an environment may always be transformed into an $\lesssim$-equivalent, idle environment), which means

(i) $E \xrightarrow[\tau]{} F$ iff $E = F$

(ii) All derivatives of E are idle.

Note that the universal environment U is trivially idle. We can now define the operational semantics of $\mathcal{WIE}(\mathcal{C}, E)$ from the operational behaviours of $\mathcal{C}$ and E by the following two rules:

(i) $$\frac{E \xrightarrow[b]{} F \quad \mathcal{C} \xrightarrow[0]{b} \mathcal{D}}{\mathcal{WIE}(\mathcal{C}, E) \xrightarrow[\tau]{} \mathcal{WIE}(\mathcal{D}, F)}$$

(ii) $$\frac{E \xrightarrow[b]{} F \quad \mathcal{C} \xrightarrow[a]{b} \mathcal{D}}{\mathcal{WIE}(\mathcal{C}, E) \xrightarrow[a]{} \mathcal{WIE}(\mathcal{D}, F)}; \qquad a \neq 0,$$

With the above definitions, we can now state the following main theorem:

THEOREM 3. *If $\mathcal{C}$ is idle-preserving and E is idle, then $\mathcal{WIE}(\mathcal{C}, E)$ is a $\mathcal{SIE}$ for $\mathcal{C}$ in E.*

It can be shown that, under the above conditions, $\mathcal{WIE}(\mathcal{C}, E)$ is a $\lesssim$-weakest environment such that $\mathcal{C}[P] \cong_E C[Q]$ whenever $P \approx_{\mathcal{WIE}(\mathcal{C},E)} Q$, where $\mathcal{C}[P] \cong_E \mathcal{C}[Q]$ informally means that $\mathcal{C}[P] \approx_E \mathcal{C}[Q]$ holds *with* $\mathcal{C}$ interacting identically with P and Q. Thus, $\mathcal{WIE}(\mathcal{C}, E)$ is nearly a $\lesssim$-weakest $\mathcal{SIE}$ for $\mathcal{C}$ in E, and nearly enough so in practice as we shall demonstrate in the following Appendix B.

For idle environments the property of relative bisimulation may also be verified more easily as can be seen from the following easy result, where attention is only paid to sequences of length $\lesssim 1$.

PROPOSITION 4. *Let* $\xi = (\text{Env}, \text{Act}, \rightarrow)$ *be an environment system, where all environments are idle. Then an* Env*-indexed family* $\mathcal{RB}$ *is a relative bisimulation if whenever* $(P, Q) \in \mathcal{RB}_E$ *and* $E \xrightarrow{a} E'$ *then*

(i) *Whenever* $P \xrightarrow{a} P'$, *then* $Q \stackrel{\tilde{a}}{\Longrightarrow} Q'$ *for some* Q', *and* $(P', Q') \in \mathcal{RB}_{E'}$

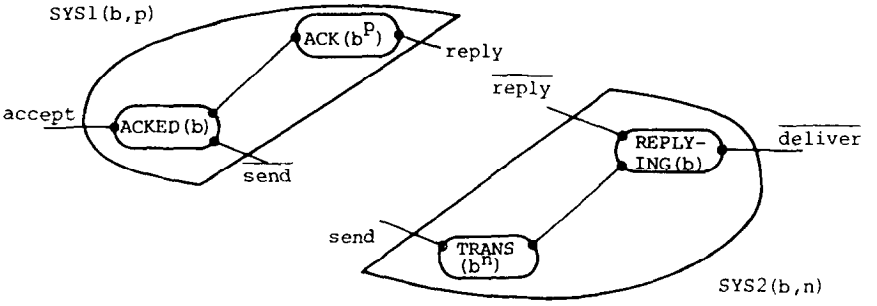
(ii) *Whenever* $Q \xrightarrow{a} Q'$, *then* $P \stackrel{\tilde{a}}{\Longrightarrow} P'$ *for some* P', *and* $(P', Q') \in \mathcal{RB}_{E'}$,

where $\tilde{\tau} = \varepsilon$ *and* $\tilde{a} = a$ *for* $a \neq \tau$.

APPENDIX B: FULL PROOF

In this appendix we complete the correctness proof of the Alternating Bit Protocol outlined in Section 5.

Recall that the proof is based on the following decomposition of SYSTEM(b, n, p):



Using the terminology introduced in Section 5 and Appendix A our proof obligations can now be reformulated as follows:

1. Find an environment $E2(b)$ which is $\mathcal{SIE}$ for $\mathcal{C}2(b, n) = [\ ] \| \text{SYS } 2(b, n)$ in U.

2. Find a specification SPEC1(b) such that SPEC1$(b) \approx_{E2(b)}$ SYS1(b, p).

3. Find an environment $E1(b)$ which is $\mathcal{SIE}$ for $\mathcal{C}1(b) =$ SPEC1$(b) \| [\ ]$ in U.

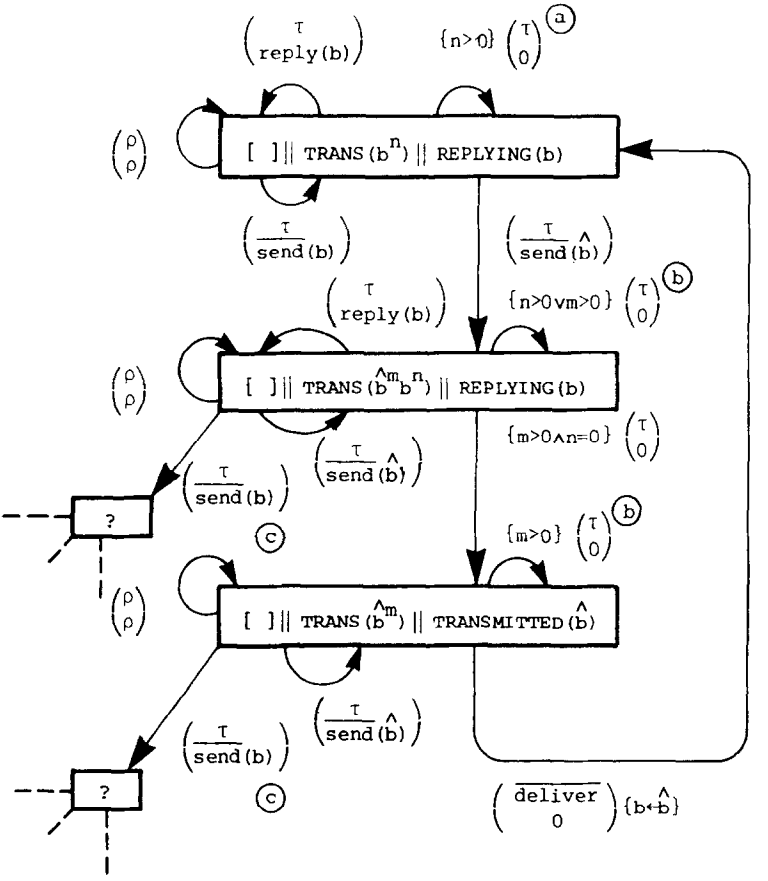
4. Find a specification SPEC2(b) such that SPEC2$(b) \approx_{E1(b)}$ SYS2(b, p).

5. Finally, show SPEC1$(b) \|$ SPEC2$(b) \approx$ SPEC.

Proof Obligation 1. Since $\mathscr{C}2(b, n)$ is idle-preserving and U is idle it suffices to find an environment $E2(b)$ such that

$$\mathscr{WIE}(\mathscr{C}2(b, n), U) \lesssim E2(b).$$

In order to establish this simulation we first (partly) determine the behaviour of the context $\mathscr{C}2(b, n) = [\ ] \| \mathrm{SYS2}(b, n)$, using the rules of Example 2 in Appendix A. The resulting behaviour is illustrated by the compressed state-transition diagram



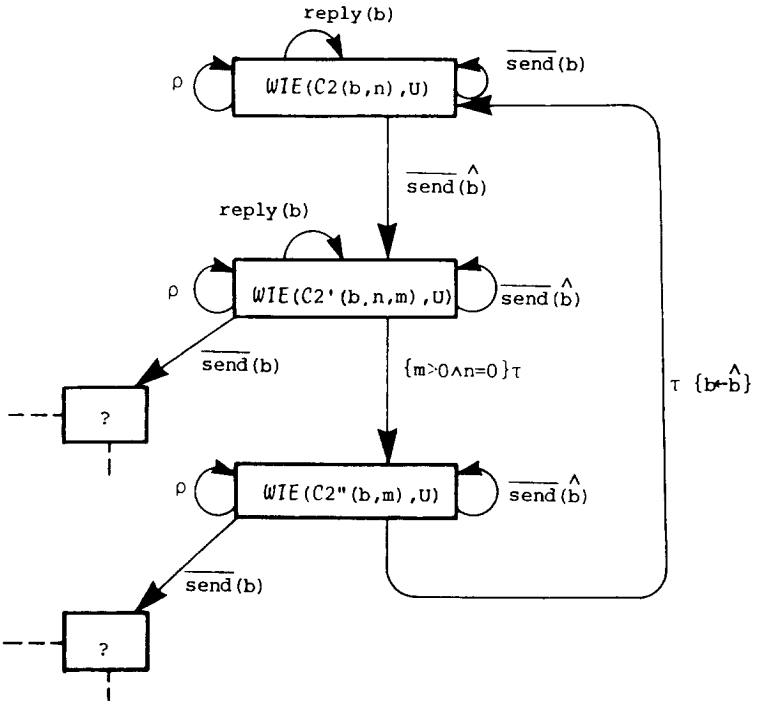
where $\rho \in \mathrm{Ext} = \mathrm{Act} - \{\mathrm{send}, \overline{\mathrm{send}}, \mathrm{reply}, \overline{\mathrm{reply}}\}$.

The diagram for $\mathscr{C}2(b, n)$ can be determined more or less automatically from the rules of Example 2 in Appendix A and the operational semantics of SYS2(b, n). Even so, let us motivate some of the arrows in the diagram. The arrow marked ⓐ in fact represents three different transductions, all with no participation of the inhabitant, and all requiring the index n to be stricty greater than 0. One transduction occurs when the TRANS line

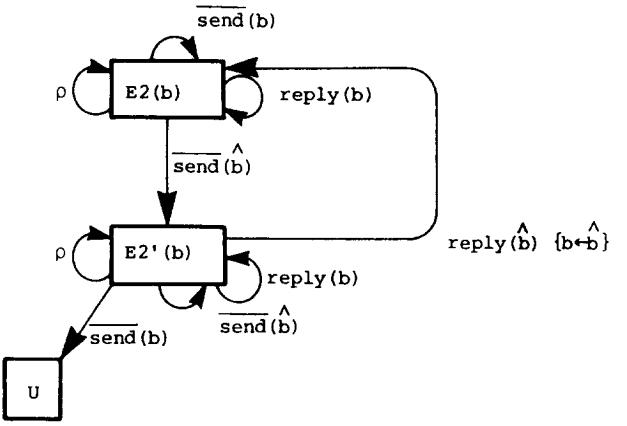
passes an already passed messages to the RECEIVER; the two other transductions correspond to the TRANS line duplicating and losing messages. The arrows marked ⓑ both represent three similar transductions. Finally, the arrows marked ⓒ represent transductions, where the inhabitant sends new messages to the TRANS line, thus leaving the line in a state where it holds sequences with possibly two bitchanges. However, with SYS1(b, p) as inhabitant these transductions will never be utilized, since SYS1(b, p) never sends a new message before having recieved acknowledgement for the previous one. Thus, the transductions ⓒ are in this application uninteresting and therefore left open.

Using the rules from Appendix A, we can now determine the behaviour of the environment $\mathcal{WIE}(\mathcal{C}2(b, n), U)$. Since $\mathcal{C}2(b, n)$ is idle-preserving and U is idle, this environment will indeed be a $\mathcal{SIE}$ for $\mathcal{C}2(b, n)$ in U. Let $\mathcal{C}2'(b, n, m)$ and $\mathcal{C}2''(b, m)$ be the two (parameterized sets of) derivatives of $\mathcal{C}2(b, n)$ shown in the previous diagram.

Using the rules for the $\mathcal{WIE}$-construction from Appendix A, the diagram for the environment $\mathcal{WIE}(C2(b, n), U)$ is easily derived (essentially the diagram is derived from that of $C2(b, n)$, simply replacing labels $\binom{b}{a}$, where $a \neq 0$, with a, and labels $\binom{b}{0}$ with τ).



However, we want to use the following finite-state environment, $E2(b)$, as $\mathscr{SIE}$ for $\mathscr{C}2(b, n)$ in U:



To justify this we must prove $\mathscr{WIE}(\mathscr{C}2(b, n), U) \lesssim E2(b)$. However, it is easily shown that the following is a simulation, from which this desired inequality directly follows:

$$\mathscr{S} = \mathscr{S}_1 \cup \mathscr{S}_2 \cup \mathscr{S}_3 \cup \mathscr{S}_4 \cup \mathscr{S}_5$$

where

$$\mathscr{S}_1 = \{(\mathscr{WIE}(\mathscr{C}2(b, n), U), E2(b)) \mid n \geq 0\}$$

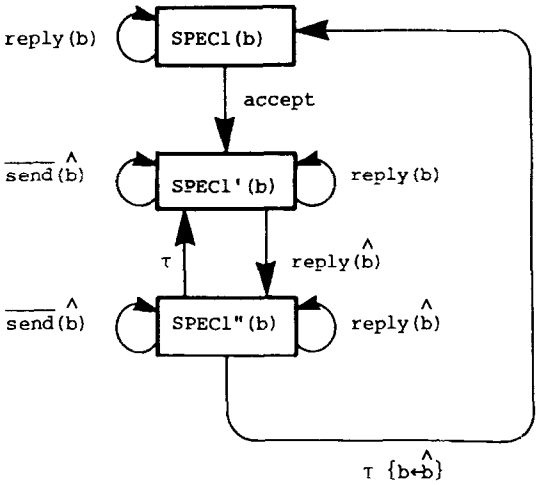
$$\mathscr{S}_2 = \{(\mathscr{WIE}(\mathscr{C}2(b, n), U), E2'(\hat{b})) \mid n \geq 0\}$$

$$\mathscr{S}_3 = \{(\mathscr{WIE}(\mathscr{C}2'(b, n, m), U), E2'(b)) \mid n, m \geq 0\}$$

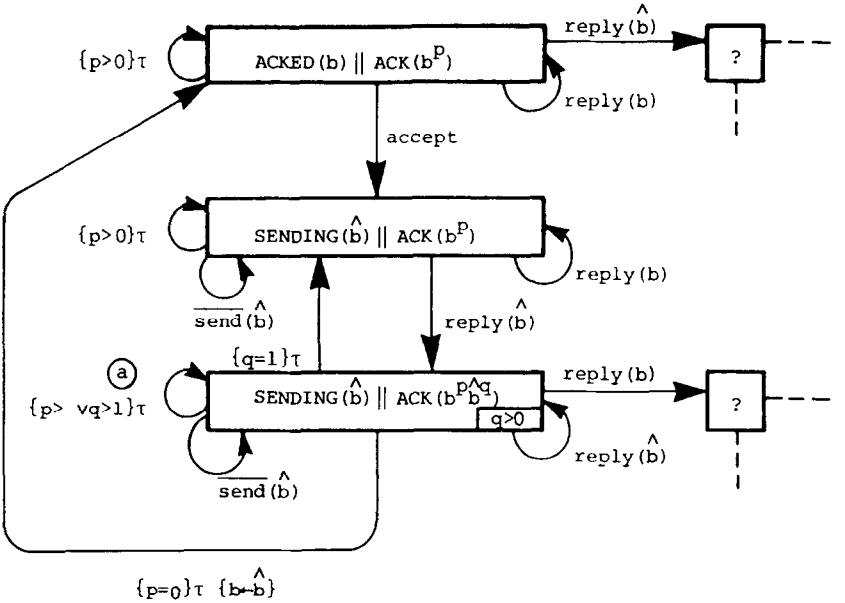
$$\mathscr{S}_4 = \{(\mathscr{WIE}(\mathscr{C}2''(b, m), U), E2'(b)) \mid m \geq 0\}$$

$$\mathscr{S}_5 = \{(\mathscr{WIE}(\mathscr{C}, U), U) \mid \mathscr{C} \text{ any context}\}.$$

Proof Obligation 2. As subspecification for SYS1(b, p) we want the finite-state agent SPEC1(b) from Section 3, which may be graphically represented as



Let us show the part of $\mathrm{SYS1}(b, p) = \mathrm{ACKED}(b) \,\|\, \mathrm{ACK}(b^p)$ relevant in the environment $E2(b)$:



The τ-move marked ⓐ in the above diagram actually represents three different types of transitions: one transition occurs when the ACK line passes an already passed acknowledgment on to the SENDER; the remaining two transitions are caused by acknowledgments being lost or duplicated. Thus, for any agent of the form $\mathrm{SENDING}(\hat{b}) \,\|\, \mathrm{ACK}(b^p \hat{b}^q)$ with $q > 0$ we have

$$\mathrm{SENDING}(\hat{b}) \,\|\, \mathrm{ACK}(b^p \hat{b}^q) \stackrel{\varepsilon}{\Longrightarrow} \mathrm{SENDING}(\hat{b}) \,\|\, \mathrm{ACK}(b^0 \hat{b}^q)$$

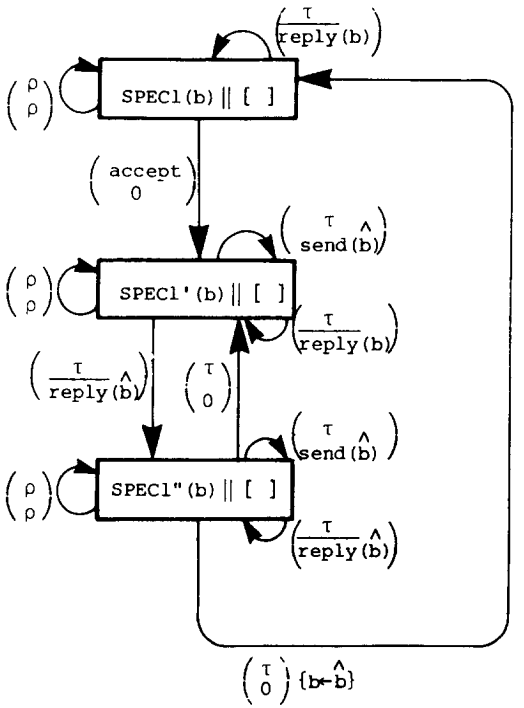
and

$$\text{SENDING}(\hat{b}) \| \text{ACK}(b^p \hat{b}^q) \stackrel{\varepsilon}{\Longrightarrow} \text{SENDING}(\hat{b}) \| \text{ACK}(b^p \hat{b}^1).$$

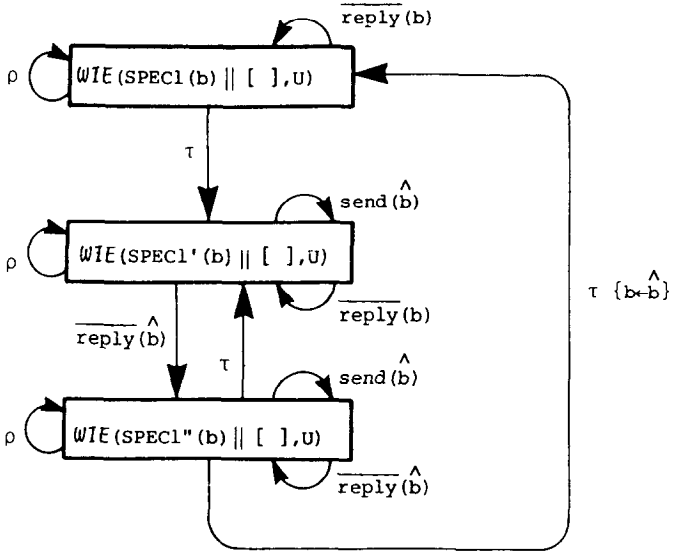
We want to prove that $\text{SPEC}(b) \approx_{E2(b)} \text{SYS1}(b, p)$. However, it can now easily be verified that the following family $\mathscr{RB}$ is a relative bisimulation, from which the above relative equivalence follows directly:

$$\begin{aligned}
\mathscr{RB}_{E2(b)} = \{&(\text{ACKED}(b) \| \text{ACK}(b^p), \text{SPEC1}(b)),\\
&(\text{SENDING}(\hat{b}) \| \text{ACK}(b^p), \text{SPEC1}'(b)),\\
&(\text{SENDING}(b) \| \text{ACK}(\hat{b}^p b^q), \text{SPEC1}''(\hat{b})),\\
&(\text{SENDING}(b) \| \text{ACK}(\hat{b}^p), \text{SPEC1}'(\hat{b})) \mid p \geqslant 0, q > 0\}\\
\mathscr{RB}_{E2'(b)} = \{&(\text{SENDING}(\hat{b}) \| \text{ACK}(b^p), \text{SPEC1}'(b)) \mid p \geqslant 0\}\\
\mathscr{RB}_U = \varnothing.&
\end{aligned}$$

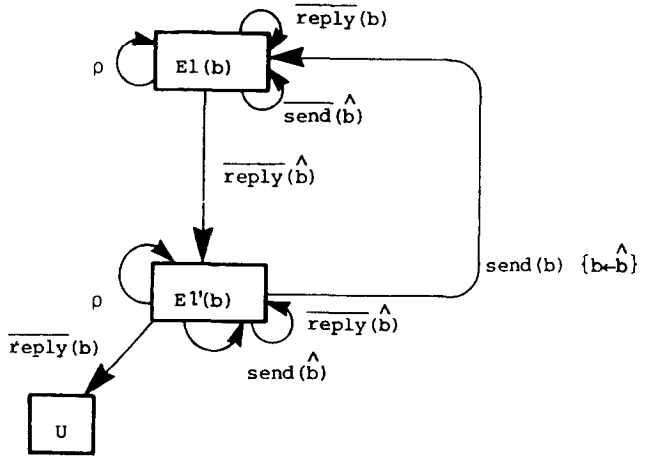
Proof Obligation 3. Let us first determine the behaviour of the context $\text{SPEC1}(b) \| [\ ]$ using the rules of Example 2 in Appendix A:



Using the rules for the $\mathcal{WIE}$ construction from Appendix A, the diagram for the environment $\mathcal{WIE}(\text{SPEC1}(b) \| [\ ], U)$ is then found to be the following:



Since $\text{SPEC}(b) \| [\ ]$ is idle-preserving and U is idle it follows from the results stated in Appendix A that $\mathcal{WIE}(\text{SPEC1}(b) \| [\ ], U)$ is a $\mathcal{SIE}$ for $\text{SPEC1}(b) \| [\ ]$ in U. However, we can do with the following simpler environment $E1(b)$ dual to the environment $E2(b)$:



To justify using $E1(b)$ as $\mathscr{S}\mathscr{I}\mathscr{E}$ for SPEC1$(b)\|[\ ]$ in U it remains to prove that $\mathscr{W}\mathscr{I}\mathscr{E}(\text{SPEC1}(b)\|[\ ], U) \lesssim E1(b)$. However, this follows directly from the easily established fact that the relation below is a simulation:

$$\mathscr{S} = \{(\mathscr{W}(b), E1(b)), (\mathscr{W}'(b), E1(b)),$$

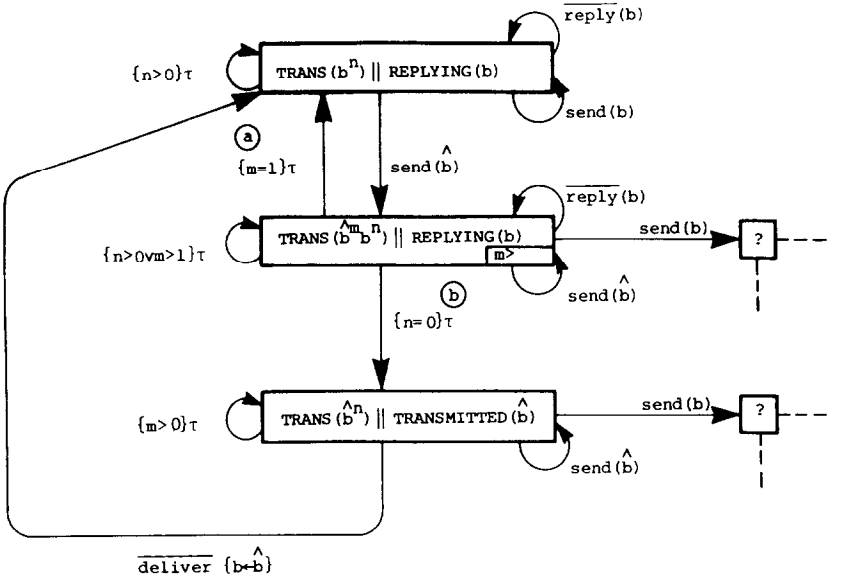
$$(\mathscr{W}''(b), E1'(b)), (\mathscr{W}(\hat{b}), E1'(b)),$$

$$(\mathscr{W}'(\hat{b}), E1'(b)), (\mathscr{W}'(b), E1'(b)),$$

$$(\mathscr{W}\mathscr{I}\mathscr{E}(\mathscr{C}, U), U) \mid \mathscr{C} \text{ any context}\},$$

where $\mathscr{W}(b) = \mathscr{W}\mathscr{I}\mathscr{E}(\text{SPEC1}(b)\|[\ ], U)$, $\mathscr{W}'(b) = \mathscr{W}\mathscr{I}\mathscr{E}(\text{SPEC1}'(b)\|[\ ], U)$ and similarly $\mathscr{W}''(b) = \mathscr{W}\mathscr{I}\mathscr{E}(\text{SPEC1}''(b)\|[\ ], U)$.

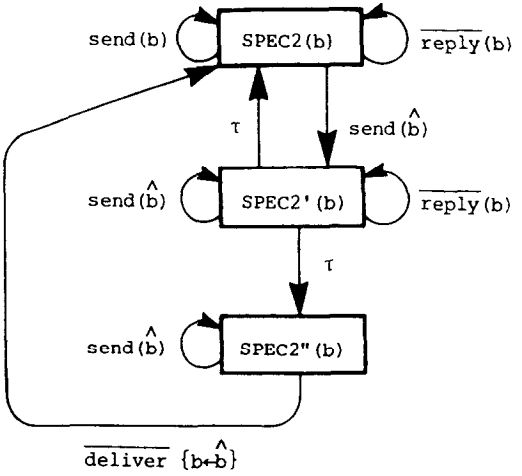
Proof Obligation 4. Let us first show the part of SYS2(b, n) = TRANS-$(b^n)\|$REPLYING(b) relevant in the environment $E1(b)$:



The τ-move marked ⓐ in the above diagram represents a transition, where the TRANS line loses a new message that has not yet reached the RECEIVER. Moreover, the TRANS line only holds a single copy of the

new message, hence a new copy of it must be send to the TRANS line before it can be passed on to the RECEIVER. The τ-move marked ⓑ represents the situation where a new message eventually is passed from the TRANS line to the RECEIVER.

As subspecification for SYS2(b, n) we want to use the following agent SPEC2(b), which is a dual of SPEC1(b):



We want to prove SPEC2(b) $\approx_{E1(b)}$ SYS2(b, n). This follows directly from the fact that the following is a relative bisimulation:

$$\mathcal{RB}_{E1(b)} = \{(\text{TRANS}(b^n) \,\|\, \text{REPLYING}(b), \text{SPEC2}(b)),$$

$$(\text{TRANS}(\hat{b}^m b^n) \,\|\, \text{REPLYING}(b), \text{SPEC2}'(b)),$$

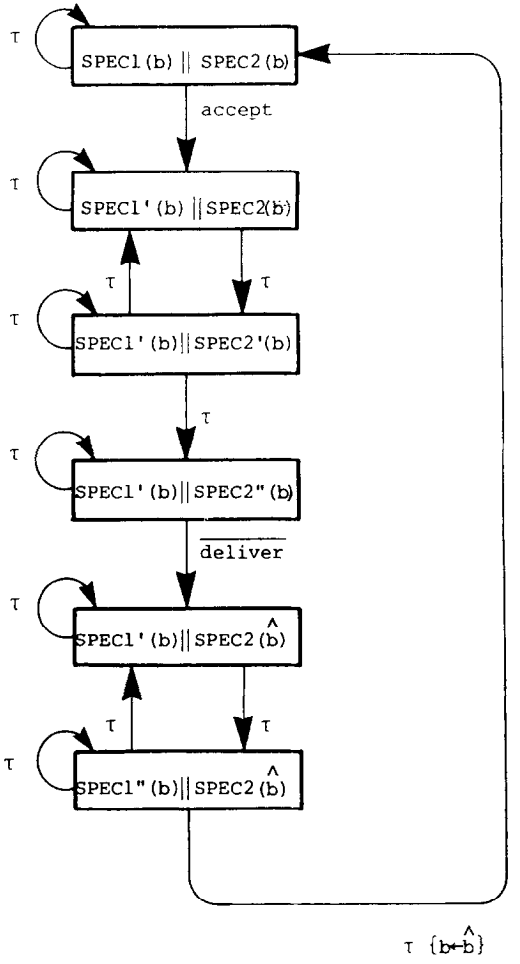
$$(\text{TRANS}(\hat{b}^n) \,\|\, \text{TRANSMITTED}(\hat{b}), \text{SPEC2}''(b)),$$

$$(\text{TRANS}(\hat{b}^n) \,\|\, \text{REPLYING}(\hat{b}), \text{SPEC2}(\hat{b})) \mid n \geq 0, m > 0\}$$

$$\mathcal{RB}_{E1'(b)} = \{(\text{TRANS}(\hat{b}^n) \,\|\, \text{REPLYING}(\hat{b}), \text{SPEC2}(\hat{b}) \mid n \geq 0\}$$

$$\mathcal{RB}_U = \varnothing.$$

Proof Obligation 5. We must prove SPEC1(b) $\|$ SPEC2(b) $\approx$ SPEC, where SPEC = accept.$\overline{\text{deliver}}$.SPEC. However, the behaviour of SPEC1(b) $\|$ SPEC2(b) is easily seen to be given by the diagram



from which the equivalence to SPEC follows directly.

RECEIVED August 17, 1987; FINAL MANUSCRIPT RECEIVED September 19, 1990.

REFERENCES

BARRINGER, KULPER, AND PNUELI (1984), Now you may compose temporal logic specifications, *in* "ACM Symposium on Theory of Computing," pp. 51–63.

BERGSTRA, J. A., AND KLOP, J. W. (1984), "Verification of an Alternating Bit Protocol by Means of Process Algebra, Technical Report CS-R8404, Centrum voor Wiskunde en Informatica, Amsterdam.

BERGSTRA, J. A., AND KLOP, J. W. (1985). Algebra of communicating processes with abstraction, *Theoret. Comput. Sci.* **37**, 77.

BOUDOL, G. (1985), "Calcul de processus et verification," Technical Report 424, INRIA.

CHANDY, K. M., AND MISRA, J. (1988), "Parallel Program Design: A Foundation," Addison–Wesley, Reading, MA.

DE BAKKER, J. W., DE ROEVER, W. P., AND ROZENBERG, G. (Eds.) (1989), "REX Workshop on Stepwise Refinement of Distributed Systems: Models, Formalism, Correctness," Lecture Notes in Computer Science, Vol. 443, Springer-Verlag, Berlin/New York.

DE ROEVER, W. P. (1985), "The quest for compositionality," Technical Report RUU–CS–85–2, University of Utrecht.

HOARE, C. A. R. (1969), An axiomatic basis for computer programming, *Comm. ACM* **12**, No. 10, 576.

HOARE, C. A. R. (1978), Communicating sequential processes, *Comm. ACM*, **21**, No. 8, 666.

JONES, C. (1983), Tentative steps toward a development method for interfering programs, *ACM Trans. Programming Languages Systems* **5**, No. 4, 596.

KOYMANS, C. P. J., AND MULDER, J. C. (1987), "A Modular Approach to Protocol Verification Using Process Algebra," Technical Report, University of Utrecht.

LAMPORT, L. (1989), Refinement and composition of specifications, *in* "Proceedings of REX Workshop on Stepwise Refinement of Distributed Systems," Lecture Notes in Computer Science, Vol. 430, Springer-Verlag, Berlin/New York.

LARSEN, K. G. (1987), A context dependent equivalence between processes, *in* "Lecture Notes in Computer Science," Vol. 194, Springer-Verlag, Berlin/New York; full version (1987) in *Theoret. Comput. Sci.* **49**, 185.

LARSEN, K. G. (1986), "Context-Dependent Bisimulation Between Processes," Ph.D. thesis, University of Edinburgh, Scotland.

LARSEN, K. G., AND XINXIN, L. (1990), Compositionality through an operational semantics of contexts, *in* "17th International Colloquium on Automata, Languages and Programming," Lecture Notes in Computer Science, Vol. 443, Springer-Verlag, Berlin/New York.

LYNCH, N., AND MERRIT, M. "Introduction to the Theory of Nested Transactions," Technical Report MIT/LCS/TR-367, MIT Laboratory for Computer Science, Cambridge, MA.

LYNCH, N., AND TUTTLE, M. (1987), Hierarchical correctness proofs for distributed algorithms, *in* "Proceedings of the 6th ACM Symposium on Principles of Distributed Computation," pp. 137–151.

MILNER, R. (1980), "Calculus of Communicating Systems," Lecture Notes in Computer Science, Vol. 92, Springer-Verlag, Berlin/New York.

MILNER, R. (1981), A modal characterization of observable machine-behaviour, *in* "Lecture Notes in Computer Science," Vol. 112, Springer-Verlag, Berlin/New York.

MILNER, R. (1983), Calculi for synchrony and asynchrony, *Theoret. Comput. Sci.* **25**, 267.

MILNER, R. (1989), "Communication and Concurrency," Prentice–Hall, Englewood Cliffs, NJ.

OWICKI, S., AND GRIES, D. (1976), An axiomatic proof technique for parallel programs, I, *Acta Informat.* **6**, No. 4, 319.

PARK, D. (1981), Concurrency and automata on infinite sequences, *in* "Proceedings of 5th GI Conference," Lecture Notes in Computer Science, Vol. 104, Springer-Verlag, Berlin/New York.

PNUELI, A. (1985), Linear and branching structures in the semantics and logics of reactive systems, *in* "Proceedings of 12th International Colloquium on Automata, Languages and Programming," Lecture Notes in Computer Science, Vol. 194, Springer-Verlag, Berlin/New York.

PRASAD, K. V. S. (1984), "Specification and Proof of a Simple Fault Tolerant System in CCS," Technical Report, Edinburgh University.

SCHOONE, A. A., AND VAN LEEUWEN, J. (1985), "Verification of Balanced Link-Level Protocol," Technical Report RUU-CS-85-12, University of Utrecht.

STARK, E. D. (1985), A proof technique for rely/guarantee properties, *in* "Lecture Notes in Computer Science," Vol. 206, p. 369, Springer-Verlag, Berlin/New York.

STIRLING, C. (1986), A compositional reformulation of Owicki-Gries's partial correctness logic for a concurrent while language, *in* "Lecture Notes in Computer Science," Vol. 226, Springer-Verlag, Berlin/New York.

TANENBAUM, A. S. (1981), "Computer Networks," Prentice–Hall, Englewood Cliffs, NJ.

WALKER, D. (1988), Bisimulation and divergence, "Proceedings of Logic in Computer Science."

WOLPER, P. (1986), Expressing interesting properties of programs in propositional temporal logic, *in* "Proceedings of 13th Symposium on Principles of Programming," pp. 184–192.

WOLPER, P., AND LOVINFOSSE, V. (1989), Verifying properties of large sets of processes with network invariants, *in* "Proceedings of Workshop on Automatic Verification Methods for Finite State Systems," Lecture Notes in Computer Science, Vol. 407, Springer-Verlag, Berlin/New York.

ZHOU, CHAOCHEN, AND LIU, JUNBO (1987), "System and Its Environment," Technical Report, Software Institute, Academia Sinica, Beijing, China.