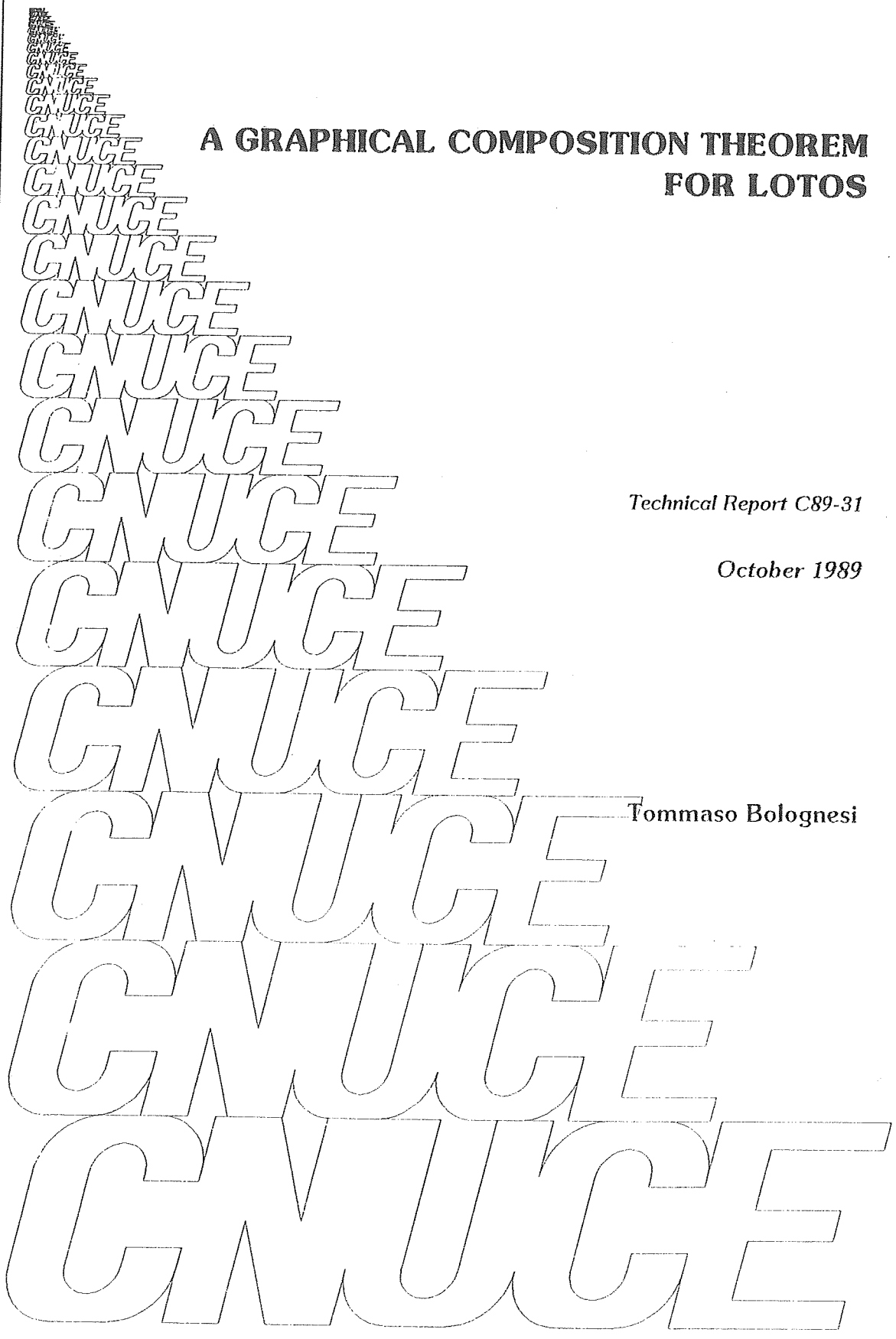


# A GRAPHICAL COMPOSITION THEOREM FOR LOTOS

*Technical Report C89-31*

*October 1989*

Tommaso Bolognesi

# A Graphical Composition Theorem for LOTOS[1]

Tommaso Bolognesi

C.N.R.-CNUCE, 36, Via S. Maria - 56100 Pisa - ITALY
phone: +39-50-577.201; e-mail : BOLOG @ ICNUCEVM.CNUCE.CNR.IT

**Abstract**

The graphical representation of the parallel composition of several LOTOS processes as a network of interconnected boxes is ambiguous, due to the nature of the LOTOS binary parallel operator. However, under suitable and sufficiently general conditions, such graphical representation is sound; a method for deriving from these graphs a family of strongly equivalent LOTOS expressions that describe the intended process composition is introduced and proved correct. The method can be used for transforming the structure of parallel LOTOS expressions, and is a generalization of previously known algebraic transformation laws.

## 0. Introduction

In the context of a stepwise refinement methodology for developing correct LOTOS specifications [ISO8807, BB87], the importance of being able to transform LOTOS behaviour expressions involving the parallel operator is easily recognized. The parallel operator plays a crucial role in any non trivial specification: in the early steps of the transformation chain, when a "constraint-oriented" specification style is often adopted [V*89], parallelism abstractly expresses the composition of logical constraints; in the subsequent, implementation oriented steps, when a "resource-oriented" style is more appropriate [V*89], it identifies the (boundaries between the) actual implementation components. Thus, any significative stepwise refinement methodology for LOTOS is likely to face the problem of transforming parallelism.

A first set of basic transformations for parallel behaviour expressions is given by the laws for weak bisimulation congruence found in Annex B of [ISO8807], which in fact preserve also *strong* bisimulation congruence; two more laws for strong bisimulation congruence are given in [V*89]. In this paper we take a further step in this direction, by providing a technique for deriving a family of strongly congruent behaviour expression which involve the parallel composition of several basic LOTOS processes. The expressions will differ in

- the order of appearance of processes,
- the nesting of parentheses, and
- the synchronization gate sets in the parallel operators,

and their congruence is proved in the central theorem of the paper. Our result originates from "graphic-oriented" intuition and reasoning, and it is hard to imagine how it could have been conceived without resorting to the potentially ambiguous yet expressive language of pictures; it generalizes the transformation laws of [V*89] for parallelism in the sense that it handles directly expressions with any number of parallel processes. We assume some familiarity with the process part of LOTOS, and with the parallel composition operators in particular.

---

[1] This work has been partly supported by the ESPRIT Project LOTOSPHERE.

## 1. Some useful facts about LOTOS parallel composition

In the sequel we will adopt the following notational conventions:

| | |
|---|---|
| B, $B_1$, ..., $B_n$, C, D, E | denote basic LOTOS behaviour expressions ('basic LOTOS' [BB87] is the subset of LOTOS where value expression and communication are not considered: processes synchronize with each other without exchanging data). |
| a, b, c, ... $g_1$, ... $g_m$ | denote gates, and the observable actions that occur at such gates; |
| S, $S_1$, $S_2$ | denote gate sets; |
| L(B), L($B_1$), ... | denote the *gate set* (also called *label set*) of expression B, $B_1$, ..., that is, the set of gates where behaviour B can perform some action; |
| '=' | denotes strong bisimulation congruence [P81, M84]. |

A fundamental law of LOTOS parallelism states that any particular instance of the parallel operator is associative:

**law 1** $\quad A \mid (B \mid C) \quad = \quad (A \mid B) \mid C$

where all occurrences of the symbol "|" denote the same instance of the parallel operator (e.g. "|[$g_1$, ... $g_n$]|"). Such law is listed in [ISO8807, Annex C] as a weak bisumulation equivalence law, but is clearly valid also for strong congruence. Two simple examples showing that associativity is no longer valid, in general, when different instances of the parallel operator appear in the expression are given below.

**Example 1.1**

(a; stop ||| a; stop) || a; stop = (a; a; stop [] a; a; stop) || a; stop = a; stop [] a; stop
a; stop ||| (a; stop || a; stop) = a; stop ||| (a; stop) = a; a; stop [] a; a; stop •

**Example 1.2**

(a; c; stop |[a]| b; stop) ||| a; stop= b; stop ||| a; stop = b; a; stop [] a; b; stop

a; c; stop |[a]| (b; stop ||| a; stop) =
= a; c; stop |[a]| (b; a; stop [] a; b; stop) = b; a; c; stop [] a; (b; c; stop [] c; b; stop) •

Sufficient conditions for the associativity of different instances of the parallel operator are studied in [V*89], where the following two laws are introduced and proved:

**law 2** $\quad (A \;|[S_1]|\; B) \;|[S_2]|\; C \quad = \quad A \;|[S_1]|\; (B \;|[S_2]|\; C)$
if $L(A) \cap S_2 \subseteq L(A) \cap S_1$, and $L(C) \cap S_1 \subseteq L(C) \cap S_2$

**law 3** $\quad (A \;|[S_1]|\; B) \;|||\; (C \;|[S_2]|\; D) \quad = \quad (A \;|||\; C) \;|[S_1 \cup S_2]|\; (B \;|||\; D)$
if $(L(A) \cup L(B)) \cap S_2 = \phi$, and $(L(C) \cup L(D)) \cap S_1 = \phi$.

Observe that in law 3 the parallel operators that appear in the l.h.s. are not the same that appear in the r.h.s. A further law for LOTOS parallelism, which also involves changes in the parallel operators, is proved in [S88]:

**law 4**

$$A \,|[S_{AB} \cup S_{ABC} \cup S_{AC}]|\, (B \,|[S_{ABC} \cup S_{BC}]|\, C)$$
$$= \quad B \,|[S_{BC} \cup S_{ABC} \cup S_{AB}]|\, (C \,|[S_{ABC} \cup S_{AC}]|\, A)$$
$$= \quad C \,|[S_{AC} \cup S_{ABC} \cup S_{BC}]|\, (A \,|[S_{ABC} \cup S_{AB}]|\, B)$$

if

$$S_{AB} = L(A) \cap L(B),$$
$$S_{AC} = L(A) \cap L(C),$$
$$S_{BC} = L(B) \cap L(C),$$
$$S_{ABC} = L(A) \cap L(B) \cap L(C).$$

The equations involved in laws 2, 3 and 4 are given a pictorial representation, respectively, in Figure 1.1 (a and b), Figure1.2 (a and b) and Figure 1.3 (a, b and c), where the following conventions, consistent with the current ISO-CCITT proposal for the graphical syntax for LOTOS [ISO3253] are adopted:

- a behaviour expression is represented by a box;
- the general parallel composition "$B_1$|S| $B_2$" is represented by two boxes $B_1$ and $B_2$, each one connected by a segment to a box with rounded corners including the list of synchronization gates S;
- independent parallel composition "$B_1$||| $B_2$" is represented by two unconnected boxes $B_1$and $B_2$.

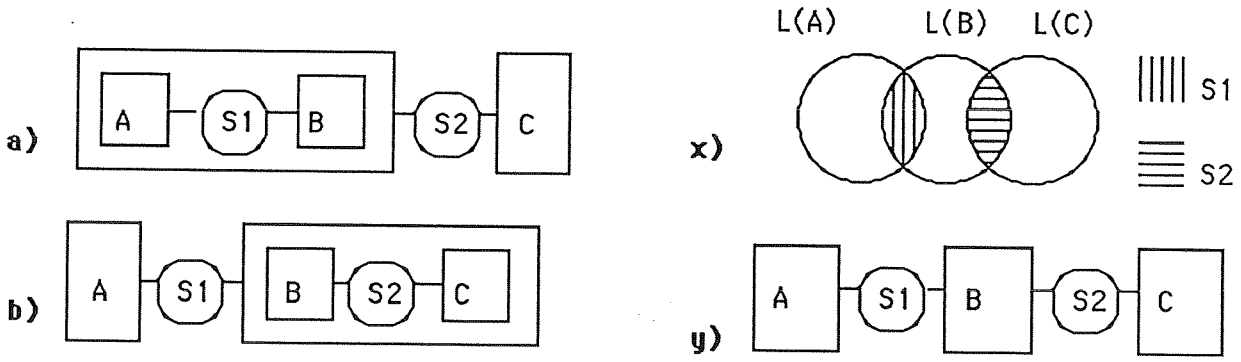


Figure 1.1 - An illustration of law 2

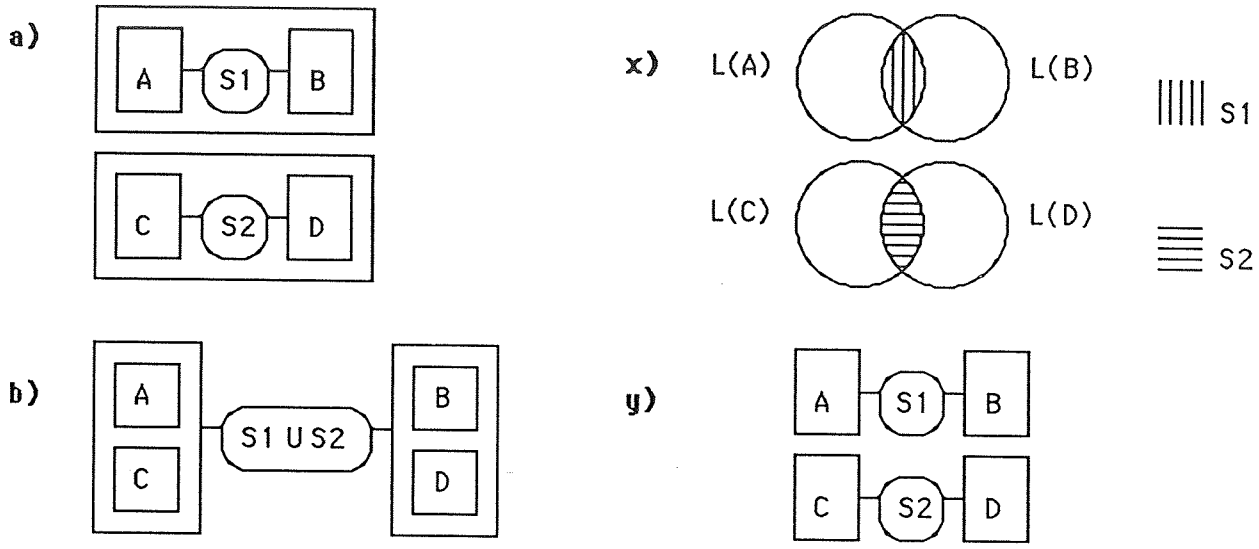


Figure 1.2 - An illustration of law 3

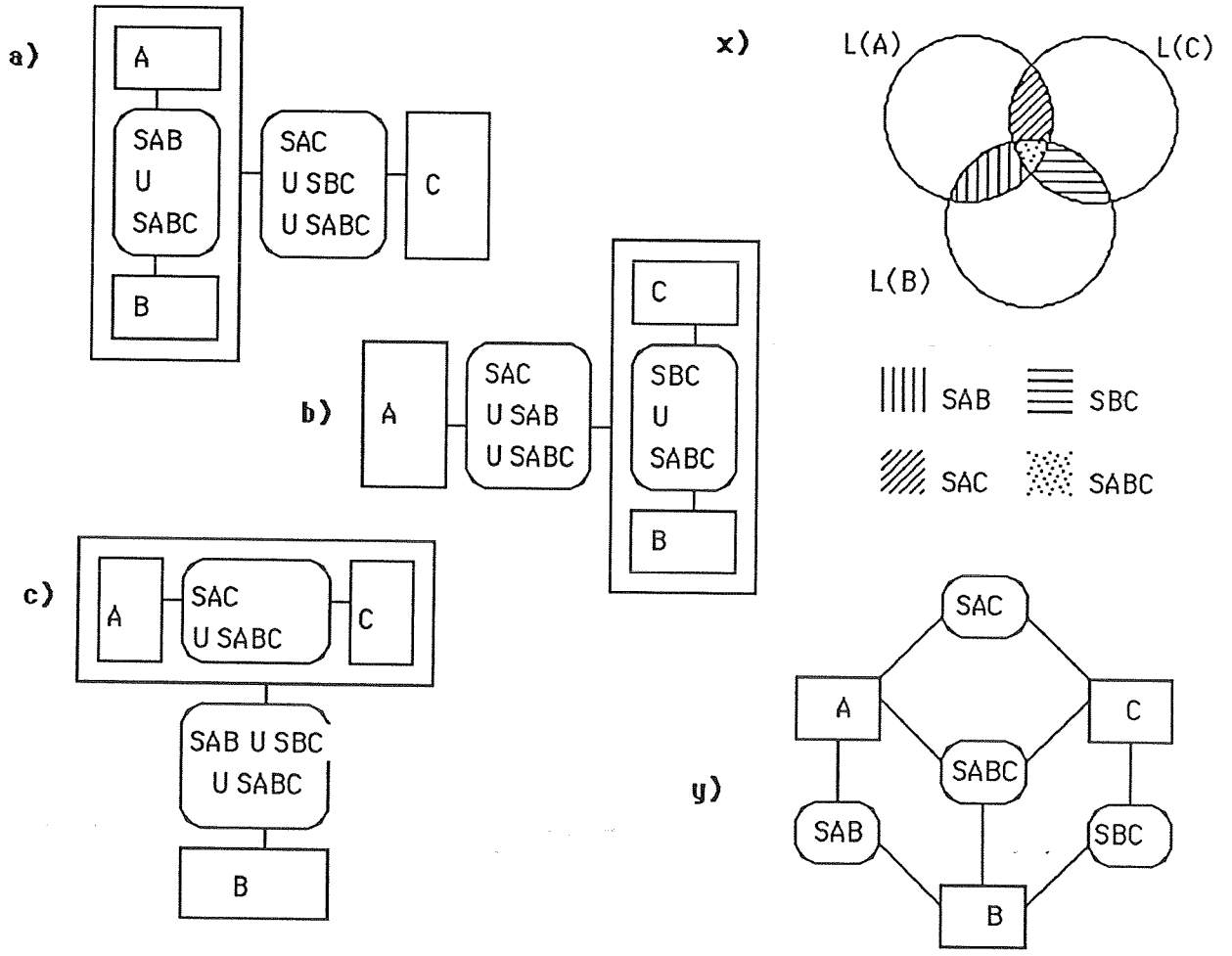


Figure 1.3 - An illustration of law 4

In Figures N.x (N = 1.1, 1.2, and 1.3) we illustrate special patterns for the relations between label sets and synchronization gates, which satisfy the conditions for the validity of, resp., laws 2, 3 and 4. For example, if L(A), L(B) and L(C) are as depicted in Figure 1.1.x, where in particular L(A) and L(B) are disjoint, and we define $S_1 = L(A) \cap L(B)$ and $S_2 = L(B) \cap L(C)$, then the conditions for the applicability of law 1 hold, as the reader may easily check.

In Figure N.y (N = 1.1, 1.2, or 1.3), we propose an alternative graphical representation for the corresponding Figure N.x. Clearly Figures N.x and N.y are isomorphic, as it becomes apparent by associating box nodes and rounded nodes in the latter with, respectively, label sets and their intersections in the former. Please observe that, in spite of the similarity with Figures N.a, b and c, Figures N.y cannot be considered as 'legal' graphical LOTOS representations, due to their potential ambiguity: in general, the many alternative textual interpretations of one such diagram may describe different behaviours.

In proposing the special patterns of Figures N.x (N.y) for the relations between label sets and synchronization gates, we have lost some generality with respect to the conditions for laws 2 and 3; however, such patterns prove useful for triggering a conjecture on the existence of a wide class of strong bisimulation equivalent LOTOS expressions, that is, of a general law for LOTOS parallelism, which will be formally stated and proved in the next section. The conjecture arises by deliberately misusing the diagrams in Figures 1.1, 1.2, and 1.3, that is, by forgetting the way in which they were produced and attempting a new interpretation for them. We do it in three steps:

1) Take Figure N.y as a starting point, and consider it no more than what it is: a graph with labelled rectangular and rounded nodes. Discover now the algorithm for deriving from such graph the graphical LOTOS expressions of Figures N.a, N.b, possibly N.c (and, indeed, more). So far, nothing can be said about the equivalence of the derived expressions. (The derivation algorithm is given only in the next section, and will directly provide *textual* LOTOS expressions from graphs like the ones in Figures N.y; however, the reader may want to discover it now, by inferring it from the pictures.)

2) Interpret now graph N.y as follows:
   - assume that the label of *each* node in the graph be the name of a gate set, and that all these gate sets be pairwise disjoint (in fact, it would suffice to require this for the gate sets of rounded nodes only);
   - assume that the label of each rectangular node, say P, be also the name of a process, and that the label set of this process be the union of the gate sets associated with all the nodes of the star-shaped subgraph centered in P (that is, the subgraph including P and all the rounded nodes adjacent to it).

   Due to this interpretation of the graph, which we may call "process-gate net", we have put together a set of processes whose label sets satisfy the (restricted) conditions for the applicability of laws 2, 3, or 4, illustrated in Figures N.x. Thus, for such processes we may safely consider the graphical LOTOS expressions N.a, N.b and, possibly, N.c, as strongly equivalent.

3) Conjecture: all the graphical LOTOS expression derived from a given process-gate net via the derivation algorithm mentioned above are strongly equivalent, provided that the label set of each process P involved be the one obtained by considering the star-shaped subgraph of the network centered in node P.

The correctness of our conjecture is proved in the next section.

## 2. Graphical composition theorem

We illustrate a technique for deriving a family of strongly equivalent behaviour expression which involve the parallel composition of several basic LOTOS processes. The expressions will differ in the order of appearance of processes, the nesting of parentheses, and the synchronization gate sets in the parallel operators.

Let 'Processes' and 'Gates' be two predefined finite sets of, respectively, process names and gate names. The starting point for deriving the family of strongly equivalent expressions is the 'general process-gate net' defined below.

**Definition 2.1 (general process-gate net)**
A general process-gate net is a 5-tuple (P, G, E, p, gs), where:

| | |
|---|---|
| (P, G, E) | is an undirected, bipartite graph with nodes P ∪ G, where |
| P | are box nodes, called process-nodes, |
| G | are rounded nodes, called gate-nodes, |
| E | ⊆ (P∪G) × (P∪G) are the undirected edges, which can only connect process-nodes with gate-nodes, |
| p: | P --> Processes is a labelling function that associates a process name to every process-node, |
| gs: | P∪G --> $2^{Gates}$ is a labelling function that associates a gate set to every gate-node. • |

Although the theorem to be proved in this section could be formulated for general process-gate nets as defined below, we find it convenient, mainly for notational reasons, to slightly restrict such definition by imposing that exactly one gate name be associated to every gate-node, and that no gate set be associated to any process-node. We will thus prove our theorem for the simplified 'process-gate nets' defined below, where no explicit node-labelling is needed any more; the extension to the case of 'general process-gate nets' is straightforward.

**Definition 2.2 (process-gate net, PGN)**
A process-gate net (PGN) is an undirected, bipartite graph (P, G, E), where:

| | |
|---|---|
| P | = ($P_1$, ..., $P_n$) is an ordered n-tuple of box nodes, called process-nodes, or simply processes; |
| G | = {$g_1$, ..., $g_m$} is a set of rounded nodes, called gate-nodes, or simply gates; |
| E | ⊆ (P∪G) × (P∪G) are the undirected edges, which can only connect process-nodes with gate-nodes. • |

For example, the PGN of Figure 1 contains 8 processes {$P_1$, ..., $P_8$} and 8 gates {a, b, c, d, e, f, g, h}.

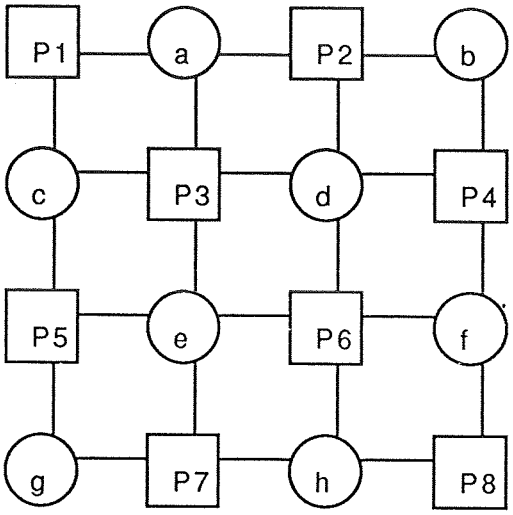


Figure 2.1 - A process-gate net

A few notational conventions are now introduced:

| | |
|---|---|
| $\pi$ | $= (\pi(1), ..., \pi(n))$ is a permutation of the array (1, ..., n) of process indices; |
| $\pi(P)$ | $= (P_{\pi(1)}, ..., P_{\pi(n)})$ is a permutation of the tuple of processes; |
| gates($P_i$) | is the set of gates directly connected to process $P_i$ in the PGN. (Example from Figure 2.1: $g(P_1) = \{a, c\}$.) |
| processes($g_j$) | is the set of processes directly connected to gate $g_j$ in the PGN. (Example from Figure 2.1: $g(a) = \{P_1, P_2, P_3\}$.) |
| $P_i$--$g_j$--> | process $P_i$ is ready to perform action $g_j$. |

Starting from a PGN we derive a family of labelled binary trees which essentially represent the parse trees of many, strongly equivalent behaviour expressions, as follows.

Let T be a binary tree with n leaves ($l_1$, ..., $l_n$), ordered from left to right, and let $\pi$ be a permutation of the process indices (1, ..., n). We label the leaves of T with the two fields "process" and "gates", and its internal nodes with the two fields "syn-gates" and "gates" again. We proceed bottom-up as follows.

For leaf $l_i$ (i = 1, ..., n) we define:

$$\text{process}(l_i) = P_{\pi(i)}$$
$$\text{gates}(l_i) = \text{gates}(P_{\pi(i)})$$

For internal node x, with sons x.1 and x.2 we define:

$$\text{syn-gates}(x) = \text{gates}(x.1) \cap \text{gates}(x.2)$$
$$\text{gates}(x) = \text{gates}(x.1) \cup \text{gates}(x.2).$$

For example, given the PGN of Figure 2.1, one of the binary trees that we may derive is shown in Figure 2.2.

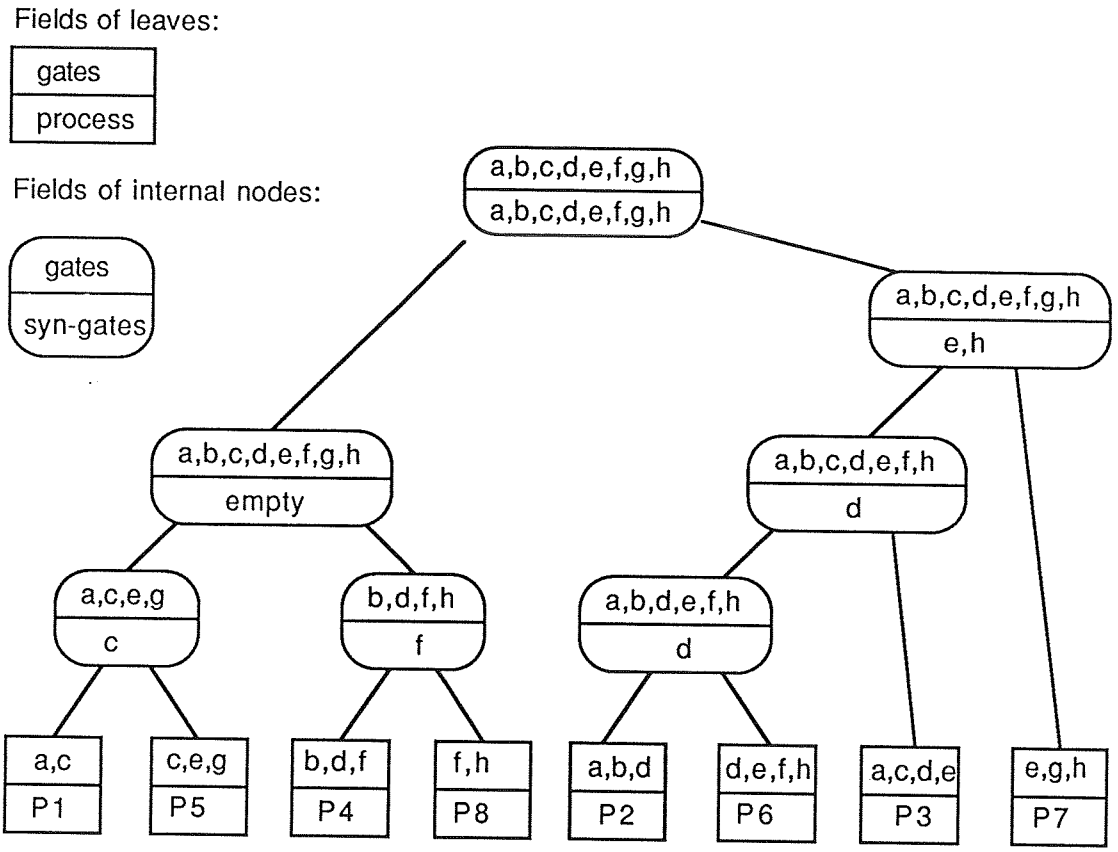


Figure 2.2 - A labelled binary tree derived from the PGN of Figure 2.1

By looking at T as a parse tree and visiting it in in-order, we immediately derive a basic LOTOS expression, as follows:

- we interpret every visited leaf l as a process instantiation of process process(l) with gates gates(l);
- we interpret every visited internal node x as a parallel operator "|[syn-gates(x)]|";
- we parenthesize the expression according to the structure of the tree.

For example, the expression associated to the tree of Figure 2.2 is:

((P1[a,c] |[c]| P5[c,e,g]) ||| (P4[b,d,f] |[f]| P8[f,h]))
|[a,b,c,d,e,f,g,h]|
(((P2[a,b,d] |[d]| P6[d,e,f,h]) |[d]| P3[a,c,d,e]) |[e,h]| P7[e,g,h])

(Recall that the LOTOS parallel operator '|||' is equivalent to '|ϕ|', thus it denotes 'independent parallel composition' between two processes, with an empty set of synchronization gates.)

The expression derived above depends on the n-tuple $P = (P_1, ..., P_n)$ of processes, on the permutation $\pi$ chosen, and on the binary tree T, and we shall denote it by $E_T(\pi(P))$, or $E_T(\pi(P_1, ..., P_n))$, or $E_T(P_{\pi(1)}, ..., P_{\pi(n)})$. Another expression obtained by chosing a different permutation and a different binary tree is:

(P5[c,e,g] |[c]| (P1[a,c] ||| P4[b,d,f]))
|[a,b,c,d,e,f,g]|
(((P3[a,c,d,e] |[e]| P7[e,g,h]) |[a,d]| P2[a,b,d]) |[d,e,h]| (P6[d,e,f,h] |[f,h]| P8[f,h]))

We shall prove our graphical composition theorem based on the following lemmas, where we assume that $P = (P_1, ..., P_n)$, $\pi$, and $E_T(\pi(P))$ are as defined above.

**Lemma 2.1**

If $P_i \in$ processes(g), for some process $P_i$ and some gate g, then g is found in the 'gates' field of leaf $l_i$ of tree T, and in the 'gates' field of all its ancestors in T, up to the root.

**Proof.**

This is a trivial consequence the way the 'gates' fields in our binary trees are computed. •

**Lemma 2.2**

If $E_T(\pi(P))$--g-->, then g belongs to the 'gates' field of T's root.

**Proof**

It sufficies to realize that if $E_T(\pi(P))$--g-->, then $P_i$--g-->, for some process $P_i$ in P. This implies that $P_i \in$ processes(g), thus, by Lemma 2.1, g belongs to the 'gates' field of T's root. •

**Lemma 2.3 (maximal cooperation)**

Let $E_T(\pi(P_1, ..., P_n))$ be a basic LOTOS parallel composition expression as described above. Then:

$E_T(\pi(P_1, ..., P_n))$ --g-->X *if and only if*

1) $\forall P_i \in$ processes(g). $\exists P_i'$. $P_i$--g-->$P_i'$, and
2) $X = E_T(\pi(PP_1, ..., PP_n))$, where
   $PP_i = P_i$ if $P_i \notin$ processes(g);
   $PP_i = P_i'$ if $P_i \in$ processes(g)
   (i = 1, ..., n).

In different words, a transition $E_T(\pi(P_1, ..., P_n))$ --g-->X always occurs with the participation of *all* processes attached to gate g in the PGN. Observe that the lemma accounts not only for the case of gates shared by more processes, but also for gates owned by a single process.

**Proof**

We first consider the *only if* part, and prove it by induction on the number n of processes. The case of n=1 is trivial, keeping in mind that $E_T(\pi(P_1)) == P_1$ (where '==' denotes syntactic equality). Consider now expression $E_T(P_{\pi(1)}, ..., P_{\pi(n)})$., with n > 1. By the construction of such expression, we have:

$E_T(P_{\pi(1)}, ..., P_{\pi(n)}) =$ $E_{T.1}(P_{\pi(1)}, ..., P_{\pi(k)})$
|syn-gates(T)|
$E_{T.2}(P_{\pi(k+1)}, ..., P_{\pi(n)})$,

for some k such that $1 \leq k < n$, where:

- T, T.1 and T.2 ambiguously identify binary trees and their roots,
- T.1 and T.2 are, respectively, the left and right sons of T.

We abbreviate the equation above as 'E = E1 |S| E2'. Suppose now that E--g-->. This may happen only if one of the two inference rules for the basic LOTOS parallel composition operator apply [BB87]; such rules are recalled below (the LOTOS successful termination gate $\partial$ has been ignored):

| | | | |
|---|---|---|---|
| r.1) | P--g-->P', g $\notin$ S | implies | P \|S\| Q --g--> P' \|S\| Q |
| r.2) | Q--g-->Q', g $\notin$ S | implies | P \|S\| Q --g--> P \|S\| Q' |
| r.3) | P--g-->P', Q--g-->Q', g $\in$ S | implies | P \|S\| Q --g--> P' \|S\| Q'. |

We distinguish two cases.

1) g $\in$ S. This means that rule r.3 was applied for deriving E--g--> : thus $E_1$--g--> and $E_2$--g-->. By the inductive hypothesis we conclude that all processes in $(P_{\pi(1)}, ..., P_{\pi(k)}) \cap$ processes(g) and in $(P_{\pi(k+1)}, ..., P_{\pi(n)}) \cap$ processes(g) are indeed involved in the execution of, respectively, $E_1$--g--> and $E_2$--g-->. Thus, all processes in processes(g) are engaged in performing E--g-->.

2) g $\notin$ S. This means that rule r.1 is applicable for deriving E--g--> (the case of rule r.2 is symmetric). Then we know that $E_1$--g-->, and also, by Lemma 2.2, that g $\in$ gates(T.1). This is sufficient to conclude that g $\notin$ gates(T.2), because, according to the construction rules of our binary trees, it is S = syn-gates(T) = gates(T.1) $\cap$ gates(T.2). Since g $\notin$ gates(T.2), it must be $(P_{\pi(k+1)}, ..., P_{\pi(n)}) \cap$ processes(g) = $\phi$, by Lemma 2.1. Furthermore, since $E_1$--g-->, all processes in $(P_{\pi(1)}, ..., P_{\pi(k)}) \cap$ processes(g) must be involved in the execution of $E_1$--g-->, by the inductive hypothesis. Thus, all processes in processes(g) are engaged in performing E--g-->.

We have thus proved the *only if* part. The *if* part can be proved similarly, by induction on the number of composed processes, and is omitted. •

The fact that any choice of the permutation $\pi$ of the process list and of the binary tree T gives an expression in the same strong equivalence class is captured by the following theorem.

**Theorem 2.1 (graphical composition theorem)**
Let PGN(P, G, E) be a process-gate net, and let $P = (P_1, ..., P_n)$ be the tuple of process-nodes. Let $\pi$ and $\pi'$ be two permutations of the tuple (1, ..., n), and let T and T' be two binary trees with n leaves each. Then the two basic LOTOS expressions $E_T(\pi(P))$ and $E_{T'}(\pi'(P))$ are strongly equivalent.

**Proof.**
We must exhibit a strong bisimulation [] that contains the pair of expressions $(E_T(\pi(P)), E_{T'}(\pi'(P))$. Let gates($P_i$) be the set of gates directly connected to process $P_i$ in the net. The bisimulation relation R is simply:

$R = \{(E_T(\pi(E_1, ..., E_n), E_{T'}(\pi'((E_1, ..., E_n))) \mid E_i$ is a basic LOTOS expressions whose label set is included in gates($P_i$), i = 1, ..., n$\}$.

The definition of $E_T(\pi(E_1, ..., E_n)$ is completely analogous to the definition of $E_T(\pi(P_1, ..., P_n))$: one has simply to label the box nodes of the given PGN with the $E_i$'s rather than the $P_i$'s, and then build the parallel expressions in the usual way; in particular, the field 'gates' of T's leaves will still be assigned the value of gates($P_i$). The fact that R is indeed a bisimulation is proved as follows. Suppose that $E_T(\pi(E_1, ..., E_n))$--g-->X, where g is some gate in the PGN. By the 'maximal cooperation' Lemma 2.3 ('only if' part), we may write this transition more precisely as $E_T(\pi(E_1, ..., E_n)$ --g--> $E_T(\pi(EE_1, ..., EE_n)$, where $E_i$--g-->$EE_i$ for all $E_i$'s connected to gate-node g in the PGN, and $E_i = EE_i$ for the remaining $E_i$'s. We use the latter fact and apply now the 'if' part of the 'maximal cooperation' lemma, for finding that $E_{T'}(\pi'((E_1, ..., E_n))$ --g--> $E_{T'}(\pi'((EE_1, ..., EE_n))$. Since pair $(E_T(\pi(EE_1, ..., EE_n), E_{T'}(\pi'((EE_1, ..., EE_n))$ is still in R, the relation is a bisimulation. Observe that the case of transitions involving the LOTOS unobservable action 'i' can be treated in exactly the same way as the case of observable actions occurring at a gate owned by a single process, which is already covered above. •

## 3. Conclusions

Anyone who has written a LOTOS specification where several processes are composed in parallel (this invariably happens when the so called 'constraint-oriented' specification style [V*89] is adopted), has faced the somewhat tricky question of how much freedom is available in formulating the (linear) parallel behaviour expression that expresses such process composition, that is, whether or not the particular permutation of the composed processes, and the choice of the specific parallel operators and synchronization gates, actually reflects the intended behaviour. The specifier is usually guided by graphic-oriented reasoning, and represents the desired process composition as a graph that we have called process-gate network. However, such representation is potentially ambiguous, and leads to 'classical' errors.
Essentially, the graphical composition theorem states that a process-gate network is a non ambiguous representation of a LOTOS behaviour if we assume that all gates of each process are explicitly shown in the net, and that all these gates are different. We have described a technique for deriving (anyone of) the textual LOTOS expressions of the behaviour expressed by a given process-gate network. On such basis one could adopt the process-gate network as a legal graphical LOTOS n-ary operator, which is largely preferable, in terms of readability, to any one of its textual counterparts.

An application of the results presented here is to be found in the early phases of a LOTOS-based system design, when a single algebraic expression is to be formulated for describing the interaction/interconnection of several processes (or 'constraints'): based on the introduced theorem the designer is aware of the available freedom in correctly formulating such LOTOS expression. A further possible application of our theorem can be envisaged for subsequent phases of system development, when transformations of the parallel structure of the LOTOS specification that preserve correctness (strong equivalence in this case) are seeked: in this respect, the theorem represents a generalization of the transformation rules (congruence laws) found in [ISO8807, Annex B]. Finally, the process-gate network can be used as convenient alternative LOTOS notation for describing harware systems such as multiprocessors, where several, possibly identical elementary units are interconnected in regular patterns.

## References


[BB87] T. Bolognesi, E. Brinksma, "Introduction to the ISO Specification Language LOTOS", Computer Networks and ISDN Systems, Vol. 14, No 1, 1987.

[ISO8807] ISO- Information Processing Systems- Open Systems Interconnection- "LOTOS- A Formal Description Technique Based on the Temporal Ordering of Observational Behaviour", IS 8807, 1989.

[ISO3253] ISO/IEC JTC 1 / SC 21 N. 3253, "Proposal for a New Work Item to develop an addendum to ISO 8807 on graphical LOTOS", 1988.

[M84] R. Milner, "A Complete Inference System for a Class of Regular Behaviours", Journal of Computer and System Sciences, Vol. 28, pp. 439-466, 1984.

[P81] D. M. R. Park, "Concurrency and Automata on Finite Sequences", Computer Science Dept., Univ. of Warwick, 1981.

[S88] C. Salvatori, "Specifiche formali in LOTOS: stili di specifica e tecniche di trasformazione", Tesi di Laurea, Univ. di Pisa, Dipart. di Informatica, Feb. 1989.

[V*89] C. A. Vissers, G. Scollo, M. van Sinderen, E. Brinksma, "On the Use of Specification Styles in the Design of Distributed Systems", to appear in Proc. of TAPSOFT 1989.