An elarged definition and complete axiomatization of observational congruence of finite processes

Philippe Darondeau

To cite this version:

Philippe Darondeau. An elarged definition and complete axiomatization of observational congruence of finite processes. [Research Report] RR-0140, INRIA. 1982. inria-00076420

INRIA

CENTRE DE RENNES
IRISA

Institut National
de Recherche
en Informatique
et en Automatique

Domaine de Voluceau
Rocquencourt
B.P.105
78153 Le Chesnay Cedex
France
Tél.:954 90 20

Rapports de Recherche

N° 140

AN ENLARGED DEFINITION AND COMPLETE AXIOMATIZATION OF OBSERVATIONAL CONGRUENCE OF FINITE PROCESSES

Philippe DARONDEAU

Juin 1982

AN ENLARGED DEFINITION AND COMPLETE AXIOMATIZATION OF OBSERVATIONAL CONGRUENCE OF FINITE PROCESSES

Philippe DARONDEAU*

Abstract : The paper is addressed to determine an adequate notion of observational equivalence of finite processes, and to give a complete axiomatization of the associated congruence. We begin with establishing the fact that recursive equivalence of processes as it has been defined in the work of Milner and his colleagues is not a fully observational equivalence, in that it is much more restrictive than it should be to agree in all cases with the judgement of an effective observer. Inspiring from CCS, an alternative syntax is proposed for processes, bringing forward n-ary guarding operators. Given p and q in that syntax, which allows invisible actions to be expressed, p and q are said equivalent iff after any common experiment, they both react by identical answers or absence of answer to any ambiguous communication offer that the observer may present. It is shown that this equivalence is also a congruence ; a finite set of equational axioms is given for the congruence, which we prove to be a complete proof system by argumenting over canonical forms of programs. In a second time, our language is enriched by adding it the necessary operators for expressing the parallel composition of processes and the renaming of their actions. The definition of the observational equivalence is extended accordingly, and it is shown that we still obtain a congruence, for which a complete proof system is finally given.

Résumé : UN DEFINITION ELARGIE ET UNE AXIOMATISATION COMPLETE DE LA CONGRUENCE OBSERVATIONNELLE DE PROCESSUS FINIS.

Le propos de ce rapport est de déterminer une notion adéquate de l'équivalence observationnelle de processus finis, et de construire une axiomatisation complète de la congruence associée. Nous commençons par établir le fait que l'équivalence récursive de processus telle qu'elle a été définie dans les travaux de Milner et de ses collègues n'est pas pleinement observationnelle, en ce sens qu'elle est trop restrictive pour être dans tous les cas exactement conforme au jugement d'un observateur effectif. Inspirée de CCS, une syntaxe alternative est proposée pour les processus, axée sur l'emploi d'opérateurs de garde n-aires et permettant l'expression d'actions invisibles.Etant donné p et q dans cette syntaxe, p et q sont dits équivalents si et seulement si après toute expéri mentation commune, tous deux réagissent par des réponses identiques (y compris l'absence de réponse) à toute offre de communication ambigüe que l'observateur peut leur soumettre. On montre que cette équivalence est une congruence ; on construit pour cette congruence une famille d'axiomes équationnels dont on prouve qu'il s'agit d'un système complet en raisonnant sur les formes canoniques des programmes. Dans une seconde étape, le langage est enrichi en lui intégrant les opérateurs nécessaires à exprimer la composition parallèle des processus et le renommage de leurs actions. La définition de l'équivalence observationnelle est étendue de pair, et on montre qu'il s'agit encore d'une congruence pour laquelle un système de preuve complet est finalement proposé.

* *IRISA, campus de Beaulieu - 35042 Rennes Cedex*

CENTRE NATIONAL DE LA RECHERCHE SCIENTIFIQUE
(L.A. 227)
UNIVERSITÉ DE RENNES 1 I.N.S.A. DE RENNES

INSTITUT NATIONAL DE RECHERCHE
EN INFORMATIQUE ET EN AUTOMATIQUE
(LABORATOIRE DE RENNES)

1. INTRODUCTION.

The basic notation introduced by R. Milner in [1] for expressing asynchronous behaviours has given rise to a series of programming languages, or behaviour algebras, which have been intensively studied regarding operational congruence of programs [2] [3] [4] [5]. All these languages incorporate the idea that communication is synchronized and takes place along lines. Differences between languages lay in the following points : behaviours may be only finite or they may be infinite, communication may engage the passing of values or it may be pure synchronization, elementary communication events may be restricted to occur one at a time or communications may be forced to be simultaneous. According to [2], two programs are operationally congruent if they may be exchanged with one another in any larger program "without affecting the behaviour of the latter", which bears evidence of the practical interest of proof systems for operational congruence. The present paper is motivated by the opinion that the precise notion of recursive equivalence which has been used in the above referenced strudies as a basis for defining the operational congruence of programs is more restrictive than needed if the only constraint to be respected is model realism. We argue that even in the simple case of finite processes without internal actions, the recursive equivalence of Milner discriminates between processes which cannot be distinguished from one another by any effective observer, for every potentialities of an ambiguous process cannot be experimented in a single run (such is the case for instance with processes p_1 and p_2 pictured in fig. 1). The above statement leads us to suggest an extended definition of a "fully observational" equivalence of processes as an alternative to Milner's equivalence which is not thoroughly practical.
As it has been done in [2], the paper restricts to finite processes with pure synchronization. The basic algebra of processes, excluding parallel composition and renaming operators from its signature, is introduced and discussed in section 2. Observational equivalence of processes is defined in section 3, and it is shown that such an equivalence justifies the relational representation of processes in the form of labelled trees. Section 4 gives a complete set of equational axioms for the congruence, which is found identical to the equivalence. Parallel composition and renamings are introduced in section 5 in the form of equationally defined operators upon basic processes, and it is finally shown that the equivalence studied so far is still a congruence for the extended signature.

2. BASIC PROCESSES.

Let $M = L \cup \{\tau\}$, where L is an arbitrary enumerable set, the elements λ of which will be called *observable action labels*, $\tau \notin L$ being the *unobservable action label*. For any finite integer $n \geq 0$, let $G_n = M^n$, the set of n-ary *guarding operators*, with $G_o = \{(\ )\} = \{NIL\}$. Our algebra of basic processes is W_Σ, the word-algebra over $\Sigma = \bigcup_{n \geq 0} G_n$. The intuitive meaning is as follows. NIL is the program which has no potential actions; $(\lambda_1, \ldots, \lambda_n).(p_1, \ldots, p_n)$ waits for some set of action demands $\{\lambda'_1, \ldots, \lambda'_k\}$ such that at least one λ'_j equals at least one λ_i, and subsequently signals acceptance of one such λ_i, taken arbitrarily, before entering the corresponding p_i; $(\mu_1, \ldots, \mu_n).(p_1, \ldots, p_n)$, if at least one of μ_i s equals τ, waits for an unforeseeable delay for some set of action demands $\{\lambda'_1, \ldots, \lambda'_k\}$ such that at least one λ'_j equals μ_i for some i, and then either behaves as above if such a demand occurs within that delay or else enters one arbitrary p_i such that $\mu_i = \tau$ for the corresponding i. It should be noticed that our algebra is somewhat different from what would be expected in the line of Milner's work, since no magic operator is provided for constructing an ambiguous process without explicitly guarding its alternatives. This slight distinction is in fact essential to our results.

3. OBSERVATIONAL EQUIVALENCE OF BASIC PROCESSES.

For any $\lambda \in L$, let $\xrightarrow{\lambda}$ be the following binary relation over $P_f(W_\Sigma)$, the set of finite parts of W_Σ.

$$\{NIL\} \xrightarrow{\lambda} \emptyset$$

$$\{(\mu_i, \ldots, \mu_n).(p_1, \ldots, p_n)\} \xrightarrow{\lambda}$$

$$\{p_i / \mu_i = \lambda\} \cup \biguplus_{\mu_j = \tau} F_j / \{p_j\} \xrightarrow{\lambda} F_j$$

$$F \cup G \xrightarrow{\lambda} F' \cup G' \ \underline{\text{if}}\ (F \neq \emptyset\ \underline{\text{or}}\ G \neq 0)\ \underline{\text{and}}$$

$$(F = \emptyset = F'\ \underline{\text{or}}\ F \xrightarrow{\lambda} F')\ \underline{\text{and}}\ (G = \emptyset = G'\ \underline{\text{or}}\ G \xrightarrow{\lambda} G')$$

For any non empty $\Lambda \in P_f(L)$, let $\downarrow\Lambda$ be the following property, defined on elements of $P_f(W_\Sigma)$.

$$\{NIL\} \downarrow\Lambda$$

$$\{(\mu_1, \ldots, \mu_n).(p_1, \ldots, p_n)\} \downarrow\Lambda\ \underline{\text{iff}}$$

$$(\forall i)\ (\mu_i \neq \tau \ \underline{\text{and}}\ \mu_i \notin \Lambda)\ \underline{\text{or}}\ (\exists i)\ (\mu_i = \tau \ \underline{\text{and}}\ \{p_i\} \downarrow \Lambda)$$

$$F \cup G \downarrow \Lambda \ \underline{\text{if}}\ (F \neq \emptyset \ \underline{\text{and}}\ F \downarrow \Lambda)\ \underline{\text{or}}\ (G \neq \emptyset \ \underline{\text{and}}\ G \downarrow \Lambda)$$

Let the equivalence $\sim$ over $P_f(W_\Sigma)$ be recursively defined as follows : $F \sim G$ iff

i) $\forall \Lambda \in P_f(L) \backslash \emptyset \qquad F \downarrow \Lambda \ \underline{\text{iff}}\ G \downarrow \Lambda$

ii) $\forall \lambda \in L \ (F \xrightarrow{\lambda} F' \text{ and } G \xrightarrow{\lambda} G')\ \underline{\text{imply}}\ F' \sim G'$.

Our observational equivalence $\sim$ over W_Σ is the derived equivalence $p \sim p'$ iff $\{p\} \sim \{p'\}$.

Namely, processes p and p' are equivalent iff the following conditions are fulfilled :

- for any sequence $S = (\Lambda_1 \to \lambda_1)(\Lambda_2 \to \lambda_2) \ldots (\Lambda_n \to \lambda_n)$ in which the observer has submitted demands Λ_i and received corresponding agreement answers $\lambda_i \in \Lambda_i$ in the order, then S is a possible experiment with p iff it is a possible experiment with p' ;
- for any such S, any possible answer that the observer may obtain from p, including the absence of answer, when he has submitted a new demand Λ after experiment S, might equally have been got from p' after identical experiment (and vice-versa).

Although our equivalence is still recursively defined as was Milner's one, recursion bears rather here on the language of experiments than on the internal structure of the programs which make them feasible.

The definition of $\sim$ makes it clear that for any context $\mathcal{C}[.]$, the following properties hold :

$$\mathcal{C}[(\mu_1, \mu_2, \mu_3, \ldots, \mu_n).(p_1, p_2, p_3, \ldots, p_n)]$$
$$\sim \mathcal{C}[(\mu_2, \mu_1, \mu_3, \ldots, \mu_n).(p_2, p_1, p_3, \ldots, p_n)]$$
$$\mathcal{C}[(\mu_1, \mu_2, \mu_2, \ldots, \mu_n).(p_1, p_2, p_2, \ldots, p_n)]$$
$$\sim \mathcal{C}[(\mu_1, \mu_2, \ldots, \mu_n).(p_1, p_2, \ldots, p_n)].$$

Those properties justify the relational representation of processes in the form of labelled trees. For instance, letting equivalent processes p_1, p_2, p_3 be defined as

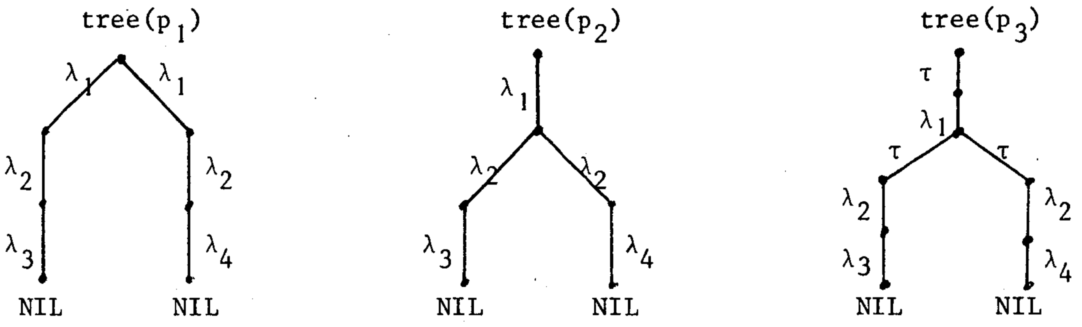
$p_1 = (\lambda_1, \lambda_1).((\lambda_2).(\lambda_4).(NIL), (\lambda_2).(\lambda_3).(NIL))$

$p_2 = (\lambda_1).(\lambda_2, \lambda_2, \lambda_2).((\lambda_3).(NIL), (\lambda_3).(NIL), (\lambda_4).(NIL))$

$p_3 = (\tau).(\lambda_1).(\tau, \tau).((\lambda_2).(\lambda_3).(NIL), (\lambda_2).(\lambda_4).(NIL))$,

their respective tree images are shown in the below figure 1. None of p_1, p_2, p_3 may be found equivalent to process $q = (\lambda_1).(\lambda_2).(\lambda_3, \lambda_4).(NIL, NIL)$.

Three equivalent processes



- Figure 1 -

4. AXIOMATIZING THE EQUIVALENCE AND THE CONGRUENCE.

To begin with, let us recall the definition of observational congruence $\simeq$ over W_Σ.

Definition. $p \simeq p'$ iff for any program context $\mathcal{C}[.]$, the following equivalence holds : $\mathcal{C}[p] \sim \mathcal{C}[p']$

The first property that we shall prove is the following

Proposition 1. The observational congruence $\simeq$ over W_Σ is just the equivalence $\sim$.

Proof. We have to establish that for any p, $p' \in W_\Sigma$, $p \sim p'$ implies $\mathcal{C}[p] \sim \mathcal{C}[p']$. We proceed by induction on tree $(\mathcal{C}[.])$, the tree image of the program context $\mathcal{C}[.]$.

Induction basis. If $\mathcal{C}[.]$ is the empty context, then $\mathcal{C}[p] = p \sim p' = \mathcal{C}[p']$.

Induction step. We have to prove that for any context $\mathcal{C}[.] = (\mu_1, \mu_2, \ldots, \mu_n).(\bullet, p_2, \ldots, p_n)$, $p_1 \sim p_1' \Rightarrow \mathcal{C}[p_1] \sim \mathcal{C}[p_1']$. From the definition of $\sim$, the proof is immediate for $n = 1$. Turning now to the other cases where $n > 1$, let process $q = (\mu_2, \ldots, \mu_n).(p_2, \ldots, p_n)$, and for any $\lambda \in L$, let $\{q\} \xrightarrow{\lambda} Q_\lambda$.

From the definition of $\downarrow\Lambda$, $\mathcal{C}[p_1] \downarrow\Lambda$ <u>iff</u>
$(\forall i\varepsilon[1,n])(\tau\neq\mu_i\notin\Lambda)$ or $(\mu_1=\tau$ and $\{p_1\}\downarrow\Lambda)$ or $(\exists i>1)(\mu_i=\tau$ and $\{p_i\}\downarrow\Lambda)$, which is equivalent to $\mathcal{C}[p_1'] \downarrow\Lambda$ since $p_1 \sim p_1'$ implies $(\forall\Lambda)$ $\{p_1\} \downarrow\Lambda$ <u>iff</u> $\{p_1'\} \downarrow\Lambda$.

Now, for $\lambda \varepsilon L$, let P_λ and P_λ' be defined as follows :

if $\mu_1 = \lambda$ then $P_\lambda = \{p_1\}$ else if $\mu_1 = \tau$ then $\{p_1\} \xrightarrow{\lambda} P_\lambda$ else $P_\lambda = \emptyset$;
if $\mu_1 = \lambda$ then $P_\lambda' = \{p_1'\}$ else if $\mu_1 = \tau$ then $\{p_1'\} \xrightarrow{\lambda} P_\lambda'$ else $P_\lambda' = \emptyset$.

Clearly, $p_1 \sim p_1'$ implies $P_\lambda \sim P_\lambda'$ in any case.

From the definition of $\xrightarrow{\lambda}$, it is easily shown that $\{\mathcal{C}[p_1]\} \xrightarrow{\lambda} P_\lambda \cup Q_\lambda$ and $\{\mathcal{C}[p_1']\} \xrightarrow{\lambda} P_\lambda' \cup Q_\lambda$.

In order to complete the proof, there remains to show $P_\lambda \cup Q_\lambda \sim P_\lambda' \cup Q_\lambda$ which is established in the following lemma. ⊠

<u>Lemma 1</u> : For any $P, P', Q \varepsilon P_f(W_\Sigma)$,
$P \sim P'$ implies $P \cup Q \sim P' \cup Q$.

From proposition 1, we know that any axiom system which is complete for $\sim$ is also a proof system for the congruence. In order to axiomatize $\simeq$, we shall therefore content ourselves with axiomatizing $\sim$. Some suitable notational conventions are now introduced before undertaking that job.

<u>Notations</u>. For any $\mu_i \varepsilon M$, $p_i \varepsilon W_\Sigma$, we let $(\mu_i\ p_i)$ stand for $(\mu_i).(p_i)$
$(\sum_{i=n}^{m} \mu_i\ p_i)$ stand for $(\mu_n, \mu_{n+1}, \ldots, \mu_m).(p_n, p_{n+1}, \ldots, p_m)$
– notice the use of brackets –

In the sequel, we shall also make free use of the following identities, where $m_o \leqslant m_1 \leqslant m_2 \ldots \leqslant m_n$:

$$\mu_i\ p_i \equiv \sum_{j=i}^{i} \mu_j\ p_j$$

$$\sum_{i=m_o}^{m_1-1} \mu_i\ p_i + \sum_{i=m_1}^{m_2-1} \mu_i\ p_i + \ldots + \sum_{i=m_{n-1}}^{m_n} \mu_i\ p_i \equiv \sum_{i=m_o}^{m_n} \mu_i\ p_i$$

– notice the absence of brackets in the above forms –

We are now ready to tackle the axiomatization of $\sim$. Our first step will be to establish the soundness of the following schemes of formulae A1-A7, where $f \overset{+}{\sim} f'$ stands for

$$(\sum_{i=1}^{n_1} \mu'_1 \; p'_i + f + \sum_{j=1}^{n_2} \mu''_j \; p''_j) \sim (\sum_{i=1}^{n_1} \mu'_i \; p'_i + f' + \sum_{j=1}^{n_2} \mu''_j \; p''_j)$$

A1 – $\mu_1 \; p_1 + \mu_2 \; p_2 \overset{+}{\sim} \mu_2 \; p_2 + \mu_1 \; p_1$

A2 – $\mu \; p + \mu \; p \overset{+}{\sim} \mu \; p$

A3 – $(\tau).(p) \sim (p)$

A4 – $\mu(\sum_{i=1}^{n} \mu' \; p_i) \overset{+}{\sim} \sum_{i=1}^{n} \mu(\mu' \; p_i)$

A5 – $\tau NIL + \sum_{i=1}^{n} \mu_i \; p_i \overset{+}{\sim} \tau NIL + \tau(\sum_{i=1}^{n} \mu_i \; p_i)$

A6 – $\sum_{i=1}^{n} \mu_i \; p_i + \tau(\sum_{j=1}^{m} \mu_j \; q_j) \overset{+}{\sim} \tau(\sum_{i=1}^{n} \mu_i \; p_i) + \tau(\sum_{j=1}^{m} \mu_j \; q_j)$ *if* $1 \leqslant m \leqslant n$

A7 – $\sum_{i=1}^{n} \mu_i \; p_i + \tau(\sum_{j=1}^{m} \mu_j \; q_j) \overset{+}{\sim} (\sum_{i=1}^{n} \mu_i(\tau p_i + \tau q_i) + \sum_{j=n+1}^{m} \mu_j \; q_j)$ *if* $1 \leqslant n \leqslant m$

Proposition 2. Any interpretation of one of the schemes A1-A7 is a sound formula.

Proof.

From the definition of $\sim$, the proof is immediate for A1-A3. Now, since A1 is sound, $n_2 = 0$ can be freely assumed for schemes A4-A7. Detailed verification follows.

A4 is sound.

Letting left $= (\sum_{i=1}^{k} \mu'_i \; p'_i + \mu(\sum_{i=1}^{n} \mu' \; p_i))$ and right $= (\sum_{i=1}^{k} \mu'_i \; p'_i + \sum_{i=1}^{n} \mu(\mu' \; p_i))$, we have to prove that left $\sim$ right.

From the definition of $\downarrow\Lambda$, proving $\{\text{left}\} \downarrow \Lambda$ *iff* $\{\text{right}\} \downarrow \Lambda$ amounts to prove $(\mu = \tau$ and $\{(\sum_{i=1}^{n} \mu' \; p_i)\} \downarrow\Lambda)$ *iff* $\mu = \tau$ and $(\exists i \in [1,n])(\{(\mu' \; p_i)\} \downarrow \Lambda)$, that is $\{(\sum_{i=1}^{n} \mu' p_i)\} \downarrow\Lambda$ *iff* $(\exists i \varepsilon [1,n])(\{(\mu' p_i)\} \downarrow\Lambda)$, which is easily verified.

For $\lambda \varepsilon L$, $\lambda \neq \mu$ implies $\{left\} \xrightarrow{\lambda} F$ *iff* $\{right\} \xrightarrow{\lambda} F$.

Supposing now $\lambda = \mu \neq \tau$, let $\{(\sum_{i=1}^{k} \mu'_i p_i)\} \xrightarrow{\lambda} G$. One has $\{left\} \xrightarrow{\lambda} G \cup \{(\sum_{i=1}^{n} \mu' p_i)\}$, $\{right\} \xrightarrow{\lambda} G \cup \bigcup_{i=1}^{n} \{(\mu' p_i)\}$.

Using the above lemma 1, soundness of A4 may be concluded from $\{(\sum_{i=1}^{n} \mu' p_i)\} \sim \bigcup_{i=1}^{n} \{(\mu' p_i)\}$, which is easily verified.

A5 is sound.

Let left = $(\sum_{i=1}^{k} \mu'_i p_i + \tau NIL + \sum_{i=1}^{n} \mu_i p_i)$ and let

right = $(\sum_{i=1}^{k} \mu'_i p_i + \tau NIL + \tau(\sum_{i=1}^{n} \mu_i p_i))$.

$(\{left\} \downarrow\Lambda$ *iff* $\{right\} \downarrow\Lambda)$ is implied by

$(\forall\Lambda)(\{left\} \downarrow\Lambda)$ and $(\forall\Lambda)(\{right\} \downarrow\Lambda)$.

Now, for any $\lambda \varepsilon L$, one has $(\{left\} \xrightarrow{\lambda} F$ *iff* $\{right\} \xrightarrow{\lambda} F)$

A6 is sound.

Let left = $(\sum_{i=1}^{k} \mu'_i p_i + \sum_{i=1}^{n} \mu_i p_i + \tau(\sum_{j=1}^{m} \mu_j q_j))$, and let

right = $(\sum_{i=1}^{k} \mu'_i p'_i + \tau(\sum_{i=1}^{n} \mu_i p_i) + \tau(\sum_{j=1}^{m} \mu_j q_j))$; $1 \leqslant m \leqslant n$.

From the definition of $\downarrow\Lambda$, one has $\{right\} \downarrow\Lambda$ *iff*

$(\exists i \varepsilon [1,k])(\mu'_i = \tau$ and $\{p'_i\} \downarrow\Lambda)$ or

$(\forall i \varepsilon [1,n])(\tau \neq \mu_i \notin \Lambda)$ or $(\exists i \varepsilon [1,n])(\mu_i = \tau$ and $\{p_i\} \downarrow\Lambda)$ or

$(\forall j \varepsilon [1,m])(\tau \neq \mu_j \notin \Lambda)$ or $(\exists j \varepsilon [1,m])(\mu_j = \tau$ and $\{q_j\} \downarrow\Lambda)$.

Since $m \leqslant n$, $(\forall i \varepsilon [1,n])(\tau \neq \mu_i \notin \Lambda) \Rightarrow (\forall j \varepsilon [1,m])(\tau \neq \mu_j \notin \Lambda)$.

One has therefore $\{right\} \downarrow\Lambda$ *iff*

$(\exists i \varepsilon [1,k])(\mu'_i = \tau$ and $\{p'_i\} \downarrow\Lambda)$ or

$(\exists i \varepsilon [1,n])(\mu_i = \tau$ and $\{p_i\} \downarrow\Lambda)$ or

$((\forall j \varepsilon [1,m])(\tau \neq \mu_j \notin \Lambda)$ or $(\exists j \varepsilon [1,m])(\mu_j = \tau$ and $\{q_j\} \downarrow\Lambda))$, that is

$\{right\} \downarrow\Lambda$ iff $\{left\} \downarrow\Lambda$.

Now, for any $\lambda \varepsilon L$, one has $(\{left\}) \xrightarrow{\lambda} F$ iff $\{right\} \xrightarrow{\lambda} F)$.

A7 is sound.

Let $left = (\sum_{i=1}^{k} \mu'_i\, p'_i + \sum_{i=1}^{n} \mu_i\, p_i + \tau(\sum_{j=1}^{m} \mu_j\, q_j))$, and let

$$right = (\sum_{i=1}^{k} \mu'_i\, p'_i + \tau(\sum_{i=1}^{n} \mu_i(\tau p_i + \tau q_i) + \sum_{j=n+1}^{m} \mu_j\, q_j)),$$

where $1 \leqslant n \leqslant m$. One has $\{right\} \downarrow\Lambda$ iff

$(\exists i \varepsilon [1,k])(\mu'_i = \tau$ and $\{p'_i\} \downarrow\Lambda)$ or $(\forall j \varepsilon [1,m])(\tau \neq \mu_j \notin \Lambda)$ or

$(\exists i \varepsilon [1,n])(\mu_i = \tau$ and $(\{p_i\} \downarrow\Lambda$ or $\{q_i\} \downarrow\Lambda))$ or

$(\exists j \varepsilon [n+1,m])(\mu_j = \tau$ and $\{q_j\} \downarrow\Lambda)$,

which is still equivalent to

$(\exists i \varepsilon [1,k])(\mu'_i = \tau$ and $\{p'_i\} \downarrow\Lambda)$ or $(\forall j \varepsilon [1,m])(\tau \neq \mu_j \notin \Lambda)$ or

$(\exists i \varepsilon [1,n])(\mu_i = \tau$ and $\{p_i\} \downarrow\Lambda)$ or

$(\exists j \varepsilon [1,m])(\mu_j = \tau$ and $\{q_j\} \downarrow\Lambda)$,

hence $\{left\} \downarrow\Lambda$ iff $\{right\} \downarrow\Lambda$.

For $\lambda \varepsilon L$, $(\forall i \varepsilon [1,n])(\lambda \neq \mu_i)$ implies

$(\{left\} \xrightarrow{\lambda} F$ iff $\{right\} \xrightarrow{\lambda} F)$.

Supposing now $(\exists i \varepsilon [1,n])(\lambda = \mu_i \neq \tau)$, let G, P_i, P, Q_j, Q be defined as follows :

if $k = 0$ then $G = \emptyset$ else $(\sum_{i=1}^{k} \mu'_i\, p'_i) \xrightarrow{\lambda} G$

$\{p_i\} \xrightarrow{\lambda} P_i$; $P = \cup \{P_i / i \varepsilon [1,n]$ and $\mu_i = \tau\}$

$\{q_j\} \xrightarrow{\lambda} Q_j$; $Q = \cup \{Q_j / j \varepsilon [1,m]$ and $\mu_j = \tau\}$

One has from the definition of $\xrightarrow{\lambda}$:

$\{left\} \xrightarrow{\lambda} G \cup P \cup Q \cup \{p_i / i \varepsilon [1,n]$ and $\mu_i = \lambda\}$
$\cup \{q_j / j \varepsilon [1,m]$ and $\mu_j = \lambda\}$

$\{right\} \xrightarrow{\lambda} G \cup P \cup Q \cup \{(\tau p_i + \tau q_i) / i \varepsilon [1,n]$ and $\mu_i = \lambda\}$
$\cup \{q_j / j \varepsilon [n+1,m]$ and $\mu_j = \lambda\}$

Using the above lemma 1, soundness of A7 may be concluded from
$\{(\tau p_i + \tau q_i)/i\varepsilon[1,n] \text{ and } \mu_i = \lambda\} \sim$
$\{p_i/i\varepsilon[1,n] \text{ and } \mu_i = \lambda\} \cup \{q_i/i\varepsilon[1,n] \text{ and } \mu_i = \lambda\}$,
which is easily verified. ■

Our next aim is to show that {A1-A7} is a complete axiom system for the observational equivalence $\sim$ (and thus for the observational congruence). Another more explicit formulation is given below.

Let $\backsim$ be the least equivalence over W_Σ for which properties i and ii are satisfied :

i) $p \backsim p'$ <u>if</u> $p \sim p'$ is a possible interpretation of one of schemes A1-A7

ii) $\mathcal{C}[p] \backsim \mathcal{C}[p']$ <u>if</u> $p \backsim p'$

Knowing from propositions 1 and 2 that $p \backsim p' \Rightarrow p \sim p'$, we shall try to establish the reverse implication $p \sim p' \Rightarrow p \backsim p'$.

From now on, let B1-B7 denote schemes obtained from A1-A7 when replacing symbol $\sim$ with symbol $\backsim$. The method that we shall use to establish the above implication is to prove that for any program p, there exists a canonical form $can(p) \equiv can(can(p)) \backsim p$ such that for any $p' \sim p''$ which verify $can(p') \equiv p'$ and $can(p'') \equiv p''$, p' and p'' must have identical tree-image (that is $p' \backsim p''$ can be proved using B1 and B2 only). Our objective will effectively be reached if such a canonical form is found, since
$[q \backsim can(q) \Rightarrow q \sim can(q)]$ entails $[p \sim p' \Rightarrow p \backsim can(p) \sim can(p') \backsim p'] \Rightarrow$
$p \backsim can(p) \backsim can(p') \backsim p' \Rightarrow p \backsim p'$.

The approach towards the construction of canonical forms will be cut into two successive steps. The first step is to show that for any $p \varepsilon W_\Sigma$, there exists $\hat{p} \backsim p$ such that for any sub-program q of $\hat{p}$ and for any $\lambda \varepsilon L$, $\{q\} \xrightarrow{\lambda} Q_\lambda$ implies Q_λ is a singleton set or $Q_\lambda = \emptyset$. The second step is to draw can(p) from $\hat{p}$. We now come to the first step.

<u>Definition</u>.

For $\mu \varepsilon M$, let $\xrightarrow{\mu}$ be the following relation over W_Σ :
$(\mu_1, \ldots, \mu_n).(p_1, \ldots, p_n) \xrightarrow{\mu} p_i$ for any i s.t. $\mu_i = \mu$. For $m \varepsilon M^+$, $m = \mu_1.\mu_2. \ldots .\mu_n$ with $n \geqslant 1$, let $\xrightarrow{m}$ be the following relation over W_Σ :

$p \xrightarrow{m} p'$ iff there exist $p_o, p_1, \dots, p_n$ in W_Σ which verify
$p = p_o \xrightarrow{\mu_1} p_1 \xrightarrow{\mu_2} p_2 \dots \xrightarrow{\mu_n} p_n = p'$.

Then p' is a sub-program of p iff $p = p'$ or there exists $m \, \varepsilon \, M^+$ s.t. $p \xrightarrow{m} p'$. ○

Definition.

For any $p, p' \, \varepsilon \, W_\Sigma$, p and p' are tree-equivalent ($p \overset{t}{\backsim} p'$) if $p \backsim p'$ may be proved from B1 and B2 only, that is if p and p' have identical tree-image. ○

In the sequel, tree-equivalent processes will not be distinguished any more from one another, and the ambiguous notation $(\sum_{i\varepsilon[1,n]} \mu_i \, p_i)$ will consequently be used to designate any one of processes which are tree-equivalent to $(\sum_{i=1}^{n} \mu_i \, p_i)$. Some lemmas are now needed for defining $\hat{p}$.

Lemma 2.

Let $p = (\sum_{i=1}^{n} \mu_i \, p_i)$ and let $\Lambda = \{\lambda \varepsilon L / \{p\} \xrightarrow{\lambda} F_\lambda \neq \emptyset\}$,

then $p \backsim (\sum_{\mu_i=\tau} \tau p_i + \sum_{\lambda\varepsilon\Lambda} \lambda \, (\sum_{p'\varepsilon F_\lambda} \tau p'))$

Lemma 3.

Let $\{q\} \xrightarrow{\lambda} \{q_1\} \cup Q$, then

$\tau q + \lambda(\sum_{i=1}^{n} \tau q_i) \overset{+}{\backsim} \tau q [\![(\sum_{i=1}^{n} \tau q_i)/_{\lambda} q_1]\!] + \lambda(\sum_{i=1}^{n} \tau q_i)$ comes from lemma 4.

where $f [\![g_1/_{\lambda} g_2]\!]$ is obtained by substituting g_2 for g_1 in f at every occurence g of g_1 such that
$f \xrightarrow{\tau^m\lambda} g$ for some m.

Lemma 4.

Taking $m \geqslant 1$ and $s'_o \equiv \sum_{i=1}^{n} \tau q_i$, let $v'_m \equiv (\tau(\dots(\tau(\lambda q_1 + s'_1)+s'_2)\dots+s'_m)$,

and $v''_m \equiv (\tau(\dots(\tau(\lambda(s'_o)+s'_1)+s'_2)\dots)+s'_m)$.

Then $\tau v''_m \overset{+}{\backsim} \tau(\tau v'_m + \tau v''_m)$.

Definition.

For $p \in W_\Sigma$, p is a uniform program (unif(p)) iff for any sub-program q of p and for any $\lambda \in L$, $\{q\} \xrightarrow{\lambda} Q_\lambda$ implies that Q_λ is a singleton set or $Q_\lambda = \emptyset$. ○

Proposition 3.

For any $p \in W_\Sigma$, there.exists a uniform program $\hat{p} \approx p$.

Proof.

We use induction on the maximal length l of experiments which are feasible with p.

Induction basis.

Let l = 0. Then the proposition is verified with taking $\hat{p} \equiv p$, since $(\forall \lambda \in L)(\{p\} \xrightarrow{\lambda} \emptyset)$.

Induction step.

Supposing that the proposition holds for $l \leqslant m-1$, let us consider the case $l = m \geqslant 1$ (whence $p \not\equiv NIL$).

Let $p \equiv (\mu_1, \ldots, \mu_n).(p_1, \ldots, p_n)$, let $\Lambda = \{\lambda_1, \ldots, \lambda_k\} = \{\lambda \in L / \{p\} \xrightarrow{\lambda} F_\lambda \neq \emptyset\}$ and let $\sigma_j \equiv (\sum_{p' \in F_{\lambda_j}} \tau p')$.

From lemma 2, one has

$$p \approx (\sum_{\mu_i = \tau} \tau p_i + \sum_{j=1}^{k} \lambda_j \sigma_j).$$

Now let $f[\![/_{\lambda_j}\ \sigma_j]\!]$ denote the result obtained from simultaneously replacing with σ_j every sub-program g of f s.t. $f \xrightarrow{\tau^* \lambda_j} g$, and that for any $j \in [1,k]$. Then $p \approx (\sum_{\mu_i = \tau} \tau p_i [\![/_{\lambda_j}\ \sigma_j]\!] + \sum_{j=1}^{k} \lambda_j \sigma_j)$ comes from repeated application of lemma 3.

For any $j \in [1,k]$, the maximal length of experiments feasible with σ_j is less than m, which implies by induction hypothesis that there exists a uniform program $\hat{\sigma}_j \approx \sigma_j$. One has finally

$$p \approx \hat{p} \equiv (\sum_{\mu_i = \tau} \tau p_i [\![/_{\lambda_j}\ \hat{\sigma}_j]\!] + \sum_{j=1}^{k} \lambda_j \hat{\sigma}_j) \text{ which is a uniform program.} \qquad \boxtimes$$

As it has been announced earlier, our next aim is to obtain a canonical form for uniform programs, which is the object of the following definition.

Definition.

Let p be a uniform program.

Let $\Lambda = \{\lambda_1, \ldots, \lambda_n\} = \{\lambda \varepsilon L / \{p\} \xrightarrow{\lambda} P_\lambda \neq \emptyset\}$, and for $i \ \varepsilon \ [1,n]$, let $\{p\} \xrightarrow{\lambda_i} \{p_i\}$.

Let $\overline{\Lambda}_1, \ldots, \overline{\Lambda}_k$ be the maximal subsets Λ' of Λ for which $p \downarrow \Lambda'$; for $j \ \varepsilon \ [1,k]$, let $\Lambda_j = \Lambda \setminus \overline{\Lambda}_j$, and let $\Lambda_o = \Lambda \setminus U \ \{\Lambda_j, \ j \varepsilon [1,k]\}$.

Then p is a canonical program iff $p \overset{t}{\approx} \tilde{p}$, given the following recursive definition of $\tilde{p}$:

- if $(p \downarrow \Lambda)$ then $\tilde{p} \equiv (\tau NIL + \sum_{i=1}^{n} \lambda_i \tilde{p}_i)$, else

$$\tilde{p} \equiv \Big(\sum_{\lambda_i \varepsilon \Lambda_o} \lambda_i \tilde{p}_i + \sum_{j=1}^{k} \tau \big(\sum_{\lambda_i \varepsilon \Lambda_j} \lambda_i \tilde{p}_i \big) \Big) \qquad \circ$$

Noticing that for any p, $\tilde{p}$ is a canonical program, we shall now try to show that $p \approx \tilde{p}$. Several lemmas are still necessary.

Lemma 5.

Let $q \equiv (\tau,\tau)(NIL,p)$ where p is a canonical program. Then $q \approx \tilde{q}$, and $\tilde{q}$ verifies

$$(\forall \lambda \varepsilon L)(\tilde{q} \xrightarrow{\tau^* \lambda} q_\lambda \ \underline{\text{iff}} \ p \xrightarrow{\tau^* \lambda} p_\lambda \overset{t}{\approx} q_\lambda).$$

Lemma 6.

$$s + \tau(s' + \tau y + \tau y') \overset{+}{\approx} s + s' + \tau y + \tau y'$$

where s,s' stand for any Σ-forms.

Lemma 7.

Let $q = (\tau,\tau)(p,p')$ be a uniform program, whose sub-programs p and p' are canonical programs, both of which different from NIL. Then $q \approx \tilde{q}$, and $\tilde{q}$ verifies :

$$(\forall \lambda \varepsilon L)(\tilde{q} \xrightarrow{\tau^* \lambda} q_\lambda \Rightarrow p \xrightarrow{\tau^* \lambda} p_\lambda \overset{t}{\approx} q_\lambda \text{ or } p' \xrightarrow{\tau^* \lambda} p'_\lambda \overset{t}{\approx} q_\lambda).$$

Lemma 8.

Let $q = (\tau,\tau, \ldots,\tau)(q_1,q_2, \ldots, q_k)$, with $k \geqslant 2$, be a uniform program whose sub-programs q_i are canonical programs. Then $q \approx \tilde{q}$ and $\tilde{q}$ verifies :

$$(\forall \lambda \varepsilon L)(\tilde{q} \xrightarrow{\tau^* \lambda} q_\lambda \Rightarrow (\exists i \varepsilon [1,k])(q_i \xrightarrow{\tau^* \lambda} p'_\lambda \overset{t}{\approx} q_\lambda)).$$

Lemma 9.

Let $q = (\lambda_{i_1}, \ldots, \lambda_{i_k}, \tau)(\tilde{p}_{i_1}, \ldots, \tilde{p}_{i_k}, p')$ be a uniform program whose sub-programs $\tilde{p}_{i_j}$ and p' are canonical programs. Then $q \backsim \tilde{q}$, and

$(\forall \lambda \varepsilon L)$

$(\tilde{q} \xrightarrow{\tau * \lambda} q_\lambda \Rightarrow q \xrightarrow{\tau * \lambda} p_\lambda \overset{t}{\backsim} q_\lambda)$.

Proposition 4.

For any uniform program p, $p \backsim \tilde{p}$ and $(\forall \lambda \varepsilon L)$

$(\tilde{p} \xrightarrow{\tau * \lambda} p_\lambda \Rightarrow p \xrightarrow{\tau * \lambda} p'_\lambda \backsim p_\lambda)$.

Proof.

For $l > k+1$, the following equivalence can be derived from B4 and B3 :

$(\lambda_1, \ldots, \lambda_k, \tau, \ldots, \tau)(p_1, \ldots, p_l) \backsim$

$(\lambda_1, \ldots, \lambda_k, \tau)(p_1, \ldots, p_k, (\tau, \ldots, \tau)(p_{k+1}, \ldots, p_l))$.

Using the above equivalence, one may first show by structural induction over programs that for any uniform program p, one can construct a corresponding uniform program $p'' \backsim p$ such that $(\forall \lambda \varepsilon L)(p'' \xrightarrow{\tau * \lambda} p''_\lambda$ implies $p \xrightarrow{\tau * \lambda} p_\lambda \backsim p''_\lambda)$ and such that any sub-program of p'' is in one of forms : NIL, $(\lambda_1, \ldots, \lambda_k)(p_1, \ldots, p_k)$, $(\lambda_1, \ldots, \lambda_k, \tau)(p_1, \ldots, p_{k+1})$ or $(\tau, \ldots, \tau)(p_1, \ldots, p_k)$, where $k \geq 1$ and $\lambda_i = \lambda_j$ iff $i = j$.

As $p'' \backsim p \Rightarrow p'' \sim p = \tilde{p}'' \overset{t}{\backsim} \tilde{p}$ comes froms the definition of $\sim$, one can freely assume in the sequel that any sub-program of p is in one of the above forms. We shall now prove the proposition by induction on the structure of p.

Induction basis.

The proposition is immediately verified for $p = \text{NIL}$.

Induction step.

Assuming that the proposition holds for p_i s, we have to prove that it also holds for $p = (\mu_1, \ldots, \mu_n)(p_1, \ldots, p_n)$ if p is a uniform program of one of the above forms.
Several cases arise.

Case 1.

$$p = (\lambda_1, \ldots, \lambda_k)(p_1, \ldots, p_k).$$

As $\tilde{p}_i \backsim p_i$ holds from the induction hypothesis, and as $\tilde{p} = (\sum_i \lambda_i \tilde{p}_i)$, the proposition is immediately verified.

Case 2.

$$p = (\tau)(p_1).$$

Then $p \backsim p_1$ (B3) and $p_1 \backsim \tilde{p}_1$ (induction hypothesis) imply $p \backsim \tilde{p}_1$, and the proposition is immediately verified since $\tilde{p} = \tilde{p}_1$.

Case 3.

$$p = (\lambda_1, \ldots, \lambda_n, \tau)(p_1, \ldots, p_n, p').$$

For any $i \in [1,n]$, one has the following properties :

$\tilde{p}' \xrightarrow{\tau^*\lambda_i} q_i \Rightarrow p' \xrightarrow{\tau^*\lambda_i} q'_i \backsim q_i$ - induction hypothesis -

$q'_i \backsim p_i$ - since p is a uniform program -

$p_i \backsim \tilde{p}_i$ - induction hypothesis -

thus $q_i \backsim \tilde{p}_i$.

Now, letting $r \equiv (\lambda_1, \ldots, \lambda_n, \tau)(\tilde{p}_1, \ldots, \tilde{p}_n, \tilde{p}')$, $r \backsim p$ comes from the induction hypothesis, and thus $p \backsim r[\![\tilde{p}_i/_{\lambda_i}\ q_i]\!]$ which is still a uniform program with canonical sub-programs.

Since $(\forall \lambda \in L \backslash \{\lambda_1, \ldots, \lambda_n\})$, one has implications

$$r[\![\tilde{p}_i/_{\lambda_i}\ q_i]\!] \xrightarrow{\tau^*\lambda} r_\lambda \Rightarrow \tilde{p}' \xrightarrow{\tau^*\lambda} r'_\lambda \backsim r_\lambda \Rightarrow$$

$$p' \xrightarrow{\tau^*\lambda} r''_\lambda \backsim r'_\lambda \Rightarrow p \xrightarrow{\tau^*\lambda} r''_\lambda \backsim r_\lambda,$$

the remaining of the proof is a direct application of lemma 8.

Case 4.

$$p = (\tau, \ldots, \tau)(p_1, \ldots, p_n) \text{ with } n > 2.$$

From the induction hypothesis, $p \backsim (\tau, \ldots, \tau)(\tilde{p}_1, \ldots, \tilde{p}_n)$.
Again from the induction hypothesis, and since p is a uniform program, one may find canonical programs $r_i \backsim \tilde{p}_i$ such that $(\forall \lambda \in L)$, the following properties are satisfied :

$$r_i \xrightarrow{\tau^*\lambda} r_\lambda \Rightarrow p \xrightarrow{\tau^*\lambda} r'_\lambda \backsim r_\lambda$$

$$r_i \xrightarrow{\tau^*\lambda} r_\lambda \text{ and } r_j \xrightarrow{\tau^*\lambda} r'_\lambda \Rightarrow r_\lambda \equiv r'_\lambda \ .$$

Now, the remaining of the proof is a direct application of lemma 9, since $p \backsim (\tau, \ldots, \tau)(r_1, \ldots, r_n)$. ⊠

Leaning on propositions 1 to 4, we shall now establish the main result of the section, which is that {A1 ... A7} is a complete proof system for the observational equivalence.

Lemma 10.

For $p \in W_\Sigma$, let can(p) denote any canonical program p' such that $p \backsim p'$. Then can(p) exists for any p.

Lemma 11.

For p, p' $\in W_\Sigma$, $p \sim p' \Rightarrow \text{can}(p) \overset{t}{\backsim} \text{can}(p')$.

Corollary.

For $p \in W_\Sigma$, can(p) is defined up to the tree equivalence $\overset{t}{\backsim}$, and $\text{can}(\text{can}(p)) \overset{t}{\backsim} \text{can}(p)$.

Theorem 1.

{A1, ... A7} is a complete proof system for either the observational equivalence $\sim$ or for the observational congruence $\simeq$ over W_Σ.

Proof.

Let $p \sim p'$, then $p \backsim \text{can}(p) \backsim \text{can}(p') \backsim p'$ holds from lemmas 10 and 11, thus $p \backsim p'$.

As $p \backsim p' \Rightarrow p \sim p'$ also holds from proposition 2, one may conclude $\backsim \equiv \sim$ and {A1, ... A7} is a complete proof system for the observational equivalence, and therefore for the observational congruence (from proposition 1). ⊠

5. PARALLEL COMPOSITION AND RENAMINGS.

Following [2], we shall now add to our signature Σ a binary operator which represents the parallel composition of programs, together with a set of unary renaming operators whose purpose is to modify the observable action labels of programs. Our final objective is to establish an equivalent of theorem 1 for the new signature, let $\boldsymbol{\Sigma}$.

From now on, the set L of observable action labels is assumed to be the union of disjoint subsets Δ and $\overline{\Delta}$ which are connected to one another by reciprocal bijections "$\overline{}$" : $\alpha \in \Delta \xrightarrow{-} \overline{\alpha} \in \overline{\Delta} \longrightarrow \overline{\overline{\alpha}} = \alpha \in \Delta$. If p =(u/v), where "/" denotes parallel composition, then communication between components u and v of p will be possible <u>iff</u> there exists $\lambda \in L = \Delta \cup \overline{\Delta}$ such that λ and $\overline{\lambda}$ are observable action labels of u and v respectively. Recalling that $M = L \cup \{\tau\}$, let now M^M be the set of partial functions s from M to M which verify the following properties i to iii, where $\perp \notin M$ represents the undefined action label :

i) $s\mu = \tau$ <u>iff</u> $\mu = \tau$ ii) $s^{-1}(\lambda) \in P_f(L)$ for $\lambda \neq \tau$

iii) $(\forall \lambda \in L)\ (s(\overline{\lambda}) = \overline{s(\lambda)}$ or $s(\lambda) = \perp = s(\overline{\lambda}))$.

If p = (u[s]), where $s \in M^M$ and [s] is the corresponding unary operator, then p will have some action labelled λ' <u>iff</u> $s(\lambda) = \lambda'$ for some action label λ of u.

Given the above definitions, let $\mathcal{S} = \{[s]/s \in M^M\}$. Our new signature is $\boldsymbol{\Sigma} = \Sigma \cup \{/\} \cup \mathcal{S}$, and our algebra of extended programs is $W_{\boldsymbol{\Sigma}}$, the word algebra over $\boldsymbol{\Sigma}$. As it has been done for Σ in section 3, our first work with $W_{\boldsymbol{\Sigma}}$ is to define the observational equivalence of programs, let $\boldsymbol{\sim}$. As Σ is included in Σ, a possible short cut is to associate any extended process $p \in W_{\boldsymbol{\Sigma}}$ with an image πp in the set W_Σ of basic processes, and to take $p \boldsymbol{\sim} p'$ <u>iff</u> $\pi p \sim \pi p'$. Such an indirect way is used in the following definition of the observational equivalence $\boldsymbol{\sim}$ where we make abundant use of results from [2] without justifying them again.

<u>Definition.</u>

For $p \in W_{\boldsymbol{\Sigma}}$, the "$\Sigma$-image" of p is the basic process πp with $\pi : W_{\boldsymbol{\Sigma}} \rightarrow W_\Sigma$ defined up to the $\overset{t}{\sim}$ equivalence by the following recursive rules, where we let $(\sum_{i \in \emptyset} \mu_i\, p_i) = \emptyset$.

R1. $\pi(\sum_i \mu_i\ p_i) = (\sum_i \mu_i\ \pi p_i)$

R2. $\pi((\sum_i \mu_i\ p_i)[s]) = (\sum_{s\mu_i \neq \perp} s\mu_i\ \pi(p_i[s]))$

R'2. $\pi(p[s]) = \pi((\pi p)[s])$ if p is not a Σ form

R'3. $\pi(p/q) = \pi(\pi p/\pi q)$ if either p or q is not a Σ form

R3. If $p = (\sum_i \mu_i\ p_i)$ and $q\ (\sum_j \nu_j\ q_j)$ then $\pi(p/q) =$

$$(\sum_i \mu_i\ \pi(p_i/q) + \sum_j \nu_j\ \pi(p/q_j) + \sum_{\mu_i=\nu_j} \tau\pi(p_i/q_j)).$$ ○

Definition.

For $p, q \in W_\Sigma$, p and q are observationally equivalent ($p \approx q$) iff their Σ-images are equivalent ($\pi p \sim \pi q$). ○

From the above definitions and from theorem 1, it can be easily shown that each of the following schemes of formulae S1, ..., S8 is sound for every interpretation S.

S1. $\mu_1\ p_1 + \mu_2\ p_2 \overset{+}{\approx} \mu_2\ p_2 + \mu_1\ p_1$

S2. $\mu p + \mu p \overset{+}{\approx} \mu p$

S3. $(\tau)\cdot(p) \approx p$

S4. $\mu(\sum_{i\in I} \mu'\ p_i) \overset{+}{\approx} (\sum_{i\in I} \mu\mu'\ p_i)$ if $I \neq \emptyset$

S5. $\sum_{i\in I} \mu_i\ p_i + \tau(\sum_{j\in J} \mu_j\ q_j) \overset{+}{\approx} \tau(\sum_{i\in I} \mu_i\ p_i) + \tau(\sum_{j\in J} \mu_j\ q_j)$ if $J \subseteq I \neq \emptyset$

S6. $\sum_{i\in I} \mu_i\ p_i + \tau(\sum_{j\in J} \mu_j\ q_j) \overset{+}{\approx} \tau(\sum_{i\in I} \mu_i(\tau p_i + \tau q_i) + \sum_{j\in J\setminus I} \mu_j\ q_j)$

if $\emptyset \neq I \subseteq J$

S7. $(\sum_{i\in I} \mu_i\ p_i)[s] \approx (\sum_{s\mu_i\neq\perp} s\mu_i(p_i[s]))$

S8. If $p = (\sum_{i\in I} \mu_i\ p_i)$ and $q = (\sum_{j\in J} \nu_j\ q_j)$ then $p/q \approx$

$$(\sum_{i\in I} \mu_i(p_i/q) + \sum_{j\in J} \nu_j(p/q_j) + \sum_{\mu_i=\nu_j} \tau(p_i/q_j))$$

We shall now try to establish that equational schemes S1 to S8 are a complete proof system for the observational equivalence $\approx$ and for the associated congruence $\underline{\approx}$. The method that we shall use here again is to prove first that $\approx$ and $\underline{\approx}$ are identical to one another before establishing that S1-S8 are a proof system for $\approx$.

Lemma 12.

Let $\mathcal{C}[.] = (\mu_1, \mu_2, \ldots, \mu_n)(\bullet, p_2, \ldots, p_n)$, with $n \geq 1$, then $u \approx v \Rightarrow \mathcal{C}[u] \approx \mathcal{C}[v]$.

Lemma 13.

For u and $v \in W_\Sigma$ and $s \in \mathcal{S}$, $u \approx v \Rightarrow u[s] \approx v[s]$.

Lemma 14.

For $u, v, p \in W_\Sigma$,

$u \approx v \Rightarrow (p/u) \approx (p/v)$ and $(u/p) \approx (v/p)$.

Proposition 5.

For $p, q \in W_\Sigma$, let $p \underline{\approx} q$ iff

$(\forall \mathcal{C}[.] \in W_{\Sigma \cup \{.\}}) (\mathcal{C}[p] \approx \bullet\mathcal{C}[q])$.

Then the observational congruence $\underline{\approx}$ over W_Σ is just the observational equivalence $\approx$.

Proof.

We have to establish that for any $p, q \in W_\Sigma$, $p \approx q \Rightarrow \mathcal{C}[p] \approx \mathcal{C}[q]$. We proceed by induction on the term structure of $\mathcal{C}[.]$.

Induction basis.

If $\mathcal{C}$ is the empty context, then $\mathcal{C}[p] = p \approx q = \mathcal{C}[q]$.

Induction step.

We have to prove that for $\mathcal{C}[.]$ in any one of forms $(\mu_1, \ldots, \mu_n)(\bullet, p_2, \ldots, p_n)$ or $(\bullet)[s]$ or $(\bullet/r)$ or $(r/\bullet)$, $p \approx q \Rightarrow \mathcal{C}[p] \approx \mathcal{C}[q]$. We proceed by case to case verification.

Case 1.

$\mathcal{C}\,[.] = (\mu_1, \ldots, \mu_n)(\bullet, p_2, \ldots, p_n)$. Then $p \approx q \Rightarrow \mathcal{C}[p] \approx \mathcal{C}[q]$ by lemma 12.

Case 2.

$\mathcal{C}\,[.] = (\bullet)\ [s]$

$\pi(\mathcal{C}[p]) = \pi(p[s]) = \pi((\pi p)[s])$ for any p, since $p = \pi p$ if $p \in W_\Sigma$. The same way, one has $\pi(\mathcal{C}[q]) = \pi((\pi q)[s])$ for any q.

Now, $p \approx q \Rightarrow \pi p \sim \pi q$, where $\pi p, \pi q \in W_\Sigma$, and therefore $\pi((\pi p)[s]) \sim \pi((\pi q)[s])$ by lemma 13, that is still $\mathcal{C}\,[p] \approx \mathcal{C}\,[q]$.

Case 3.

$\mathcal{C}\,[.] = (\bullet/r)$

$\pi(\mathcal{C}[p]) = \pi(p/r) = \pi(\pi p/\pi r)$

$\pi(\mathcal{C}[q]) = \pi(\pi q/\pi r)$

Now, $p \approx q \Rightarrow \pi p \sim \pi q$, $\pi p = \pi(\pi p)$, and $\pi q = \pi(\pi q)$, thus $\pi(\pi p) \sim \pi(\pi q)$ which entails $\pi p \approx \pi q$.

By lemma 14 : $\pi p \approx \pi q \Rightarrow (\pi p/\pi r) \approx (\pi q/\pi r)$

$\Rightarrow \pi(\pi p/\pi r) \sim \pi(\pi q/\pi r)$

$\Rightarrow \mathcal{C}[p] \approx \mathcal{C}[q]$.

Case 4.

$\mathcal{C}\,[.] = (r/\bullet)$

similar to case 3. $\boxtimes$

The final result of the paper may now be stated before some conclusions are drawn.

Theorem 2.

Equational schemes {S1, ..., S8} are a complete proof system for either the observational equivalence $\approx$ or for the observational congruence $\underline{\approx}$ over W_Σ.

Proof.

Let p and $q \in W_\Sigma$ such that $p \approx q$ (or equivalently $p \underline{\approx} q$).

As $(u \approx v \Rightarrow \mathcal{C}[u] \approx \mathcal{C}[v])$ holds from proposition 5, one can easily verify, using structural induction over terms of W_Σ, that for any $r \in W_\Sigma$, $r \approx \pi r$ may be proved by a finite number of applications of axioms S7, S8.

Therefore, $(p \boldsymbol{\sim} \pi p)$ and $(q \boldsymbol{\sim} \pi q)$ can be given proofs in the axiomatic system {S1 ... S8}.

Let u and v ε W_{Σ} such that $u \boldsymbol{\sim} v$.
$\pi u = u$ and $\pi v = v \Rightarrow u \sim v$ (from the definition of $\boldsymbol{\sim}$).
By theorem 1, there exists a proof of $u \sim v$ in the axiomatic system {A1 ... A7}. Clearly, for any such proof $\mathscr{P}_{\sim}$, there exists a corresponding proof $\mathscr{P}_{\boldsymbol{\sim}}$ of $(u \boldsymbol{\sim} v)$ in the axiomatic system {S1, ..., S6}.

Therefore, $(\pi p \boldsymbol{\sim} \pi q)$ can be proved from {S1, ..., S8} since πp and πq ε W_{Σ}.

⊠

6. CONCLUSIONS.

In the paper, we have expressed the opinion that processes p and q are equivalent iff identical answers or absence of answer may be obtained from p and q for any ambiguous communication offer that the observer may present to either p or q after any identical sequence of interactions with the observed processes. We have established that the observational equivalence so defined is also a congruence, and that for two different signatures : Σ (n-ary guarding operators), and $\mathfrak{S}$ (guarding operators, renaming operators, and parallel composition). Complete proof systems have been exhibited for the corresponding equivalences $\sim$ and $\approx$, given in the form of finite sets of equational axiom schemas. Technical developments which appear in the paper moreover show a possible strategy for efficient mechanized proofs : in order to prove $p \approx q$, a possible way is to prove $can(\pi p) \overset{t}{\sim} can(\pi q)$ through the following steps 1 to 4 :

1. From p and q, derive πp and πq using S7 and S8 ;
2. From πp and πq, derive uniform programs $\widetilde{\pi p}$ and $\widetilde{\pi q}$, using constructive versions of prop. 3 and lemmas 2-4 ;
3. From $\widetilde{\pi p}$ and $\widetilde{\pi q}$, derive $can(\pi p)$ and $can(\pi q)$, using constructive versions of prop. 4 and lemmas 5-9 ;
4. Verify that $can(\pi p)$ and $can(\pi q)$ have identical tree-image, using S1 and S2.

Although our signature $\mathfrak{S}$ slightly differs from signature Σ_3 which has been considered in [2], our congruence $\approx$ may appear as a proper extension of Hennessy-Milner's congruence $\underline{\sim}_3$ over W_{Σ_3} : if p and q ε $W_{\mathfrak{S}}$ can be translated into programs of W_{Σ_3} by applying them the syntactical transformation trans : $(W_{\mathfrak{S}} \longrightarrow W_{\Sigma_3}) : (\mu_1, \ldots, \mu_n)(p_1, \ldots, p_n) \xrightarrow{trans} (\mu_1 p_1 + (\mu_2 p_2 + (\ldots \mu_n p_n) \ldots))$, then $trans(p) \underline{\sim}_3 trans(q) \Rightarrow p \approx q$. In fact, every axiom which has been given for $\underline{\sim}_3$ can be shown to derive from S1-S8 up to syntactical translation.

As was remarked in [3], "the initial algebra for laws {S1 ... S8} gives a possible denotational semantics for $W_{\mathfrak{S}}$, which is fully abstract with respect to the operational semantics". To the opposite, it seems not so easy to construct a direct denotational semantics of programs in the domain of labelled trees, although tree $(can(\pi p))$ is univoquely determined for $p \ \varepsilon \ W_{\mathfrak{S}}$. Another difficulty is to extend our results to infinite processes, which is our next objective, without neglecting the issue of fairness.

Aknowledgements.

Thanks are due to P. Le Guernic for helpful discussions and advices.

References.

[1] Milner R. (1978). Synthesis of Communicating Behaviour.
Proc. 7th MFCS Conference. Zakopane Poland.
Springer-Verlag LNCS Vol. 64, pp. 61-83.

[2] Hennessy M. & Milner R. (1980). On observing nondeterminism and concurrency.
ICALP'80. Noordwijkerhout.
Springer-Verlag LNCS Vol. 74.

[3] Hennessy M. & Plotkin G. (1980). A term model for CCS.
Proc. 9th MFCS Conference. Rydzyna Poland.
Springer-Verlag LNCS Vol. 88, pp. 261-274.

[4] Milner R. (1980). A Calculus of Communicating Systems.
Springer-Verlag LNCS Vol. 92, (170 pp.).

[5] Milner R. (1980). On relating synchrony and asynchrony.
University of Edinburg.
Report CSR 75-80, (December 1980).

lemma 1 For any $P, P', Q \in P_f(W_\Sigma)$,

$P \sim P'$ implies $P \cup Q \sim P' \cup Q$.

proof. We proceed by induction on the maximal length l of experiments which are feasible on members q of Q, $P \neq \phi \neq P'$ being implicitly assumed.

induction basis. $l = 0$ is assumed.

$(P \cup Q) \downarrow \Lambda$ *iff* $P \downarrow \Lambda$ or $Q \downarrow \Lambda$, which is equivalent to $(P' \cup Q) \downarrow \Lambda$ since $P \sim P'$ implies $P \downarrow \Lambda$ *iff* $P' \downarrow \Lambda$.

Now, for any $\lambda \in L$:

$(P \cup Q) \xrightarrow{\lambda} R_\lambda$ implies $P \xrightarrow{\lambda} R_\lambda$.

$(P' \cup Q) \xrightarrow{\lambda} R'_\lambda$ implies $P' \xrightarrow{\lambda} R'_\lambda$

whence $R_\lambda \sim R'_\lambda$ since $P \sim P'$.

induction step. $l > 0$ and the property is assumed to hold for any $l' < l$.

$(P \cup Q) \downarrow \Lambda$ *iff* $(P' \cup Q) \downarrow \Lambda$ is shown as above.

Now, for any $\lambda \in L$:

$P \xrightarrow{\lambda} P_\lambda$ and $Q \xrightarrow{\lambda} Q_\lambda$ imply $(P \cup Q) \xrightarrow{\lambda} (P_\lambda \cup Q_\lambda)$

$P' \xrightarrow{\lambda} P'_\lambda$ and $Q \xrightarrow{\lambda} Q_\lambda$ imply $(P' \cup Q) \xrightarrow{\lambda} (P'_\lambda \cup Q_\lambda)$

and $(P_\lambda \cup Q_\lambda) \sim (P'_\lambda \cup Q_\lambda)$ holds from the induction hypothesis since $P \sim P'$ implies $P_\lambda \sim P'_\lambda$ ⊠

lemma 2

Let $p = (\sum_{i=1}^{n} \mu_i p_i)$ and let $\Lambda = \{\lambda \in L \,/\, \{p\} \xrightarrow{\lambda} F_\lambda \neq \phi\}$,

then $p \approx (\sum_{\mu_i = \tau} \tau p_i + \sum_{\lambda \in \Lambda} \lambda (\sum_{p' \in F_\lambda} \tau p'))$.

proof

Let $s \equiv \sum_{i=1}^{n} \mu_i p_i$. In a first part, $p \xrightarrow{\tau^m \lambda} p' \Longrightarrow$ $(s \overset{+}{\approx} s + \lambda p')$ is proved using induction over m.

For $m = 0$, the proof is a direct application of B2.

Supposing that the property holds for $m \leqslant l-1$, let us show that it is also the case with $m = l \geqslant 1$.

$p \xrightarrow{\tau^m \lambda} p' \Longrightarrow p \overset{t}{\approx} (\tau p'_0 + \sum_{j=1}^{n-1} \mu'_j p'_j) \overset{t}{\approx} (s)$, where $p \xrightarrow{\tau}$ $p'_0 \xrightarrow{\tau^{m-1} \lambda} p'$. Let s_0 such that $p'_0 \equiv (s_0)$.

From the induction hypothesis, one may assume

$s'_0 \overset{+}{\approx} s'_0 + \lambda p'$, and $s \overset{+}{\approx} \tau(s'_0 + \lambda p') + \sum_{j=1}^{n-1} \mu'_j p'_j$.

Now, from B3, B2 and B7 :-

$\tau(\lambda p' + s'_0) \overset{+}{\approx} \tau(\lambda(\tau p') + s'_0) \overset{+}{\approx} \tau(\lambda(\tau p' + \tau p') + s'_0)$

$\overset{+}{\approx} \lambda p' + \tau(\lambda p' + s'_0)$, and therefore

$s \overset{+}{\approx} \lambda p' + \tau(s'_0 + \lambda p') + \sum_{j=1}^{n-1} \mu'_j p'_j \overset{+}{\approx} s + \lambda p'$,

which proves that the above property is valid for any m.

Applying that property in the second part of our proof, we get

$p \approx (s + \sum_{\lambda \in \Lambda} \sum_{p' \in F_\lambda} \lambda p')$

$\approx (\sum_{\mu_i = \tau} \tau p_i + \sum_{\lambda \in \Lambda} \sum_{p' \in F_\lambda} \lambda(\tau p'))$ - from B2, B3 -

$\approx (\sum_{\mu_i = \tau} \tau p_i + \sum_{\lambda \in \Lambda} \lambda (\sum_{p' \in F_\lambda} \tau p'))$ - from B4 □

lemma 3

Let $\{q\} \xrightarrow{\lambda} \{q_1\} \cup Q$, then

$$\tau q + \lambda(\sum_{i=1}^{n} \tau q_i) \overset{+}{\sim} \tau q [\![(\sum_{i=1}^{n} \tau q_i)/_{\lambda}\, q_1]\!] + \lambda(\sum_{i=1}^{n} \tau q_i)$$

where $f[\![g_1 /_{\lambda} g_2]\!]$ is obtained by substituting g_2 for g_1 in f at every occurence g of g_1 such that $f \xrightarrow{\tau^m \lambda} g$ for some m.

proof

$\{q\} \xrightarrow{\lambda} \{q_1\} \cup Q$ implies that for some m, there exist sums $s_1, \dots, s_m$ which verify

$$\tau q \overset{t}{\sim} \tau(\tau(\dots(\tau(\lambda q_1 + s_m) + s_{m-1}) + \dots s_1).$$

In order to prove the lemma, one may satisfy to establish property P:

$$\tau(\tau(\dots(\tau(\lambda q_1 + s_m) + s_{m-1}) + \dots s_1) + \lambda(\sum_{i=1}^{n} \tau q_i) \overset{+}{\sim}$$
$$\tau(\tau(\dots(\tau(\lambda(\sum_{i=1}^{n} \tau q_i) + s_m) + \dots s_1) + \lambda(\sum_{i=1}^{n} \tau q_i),$$

since lemma 3 then follows by repeated application of B1 and of the above property.

Induction over m will be used for proving P.

In lines below, t' and t'' respectively stand for the left and right members of the equivalence to be proven, and $s_0 \equiv \sum_{i=1}^{n} \tau q_i$.

Induction basis. Let $m = 1$.

$$t' \equiv \tau(\lambda q_1 + s_1) + \lambda(s_0)$$

$$\overset{+}{\sim}\ \lambda(\sum_{i=1}^{n} \tau q_i) + \tau(\lambda q_1 + s_1) + \lambda(s_0) \qquad \text{– B1, B2 –}$$

$$\overset{+}{\sim}\ \tau(\lambda(\tau(s_0) + \tau q_1) + s_1) + \lambda(s_0) \qquad \text{– B7 –}$$

$$\overset{+}{\sim}\ \tau(\lambda(\sum_{i=1}^{n} \tau(\tau q_i) + \tau q_1) + s_1) + \lambda(s_0) \qquad \text{– B4 –}$$

$$\overset{+}{\sim}\ \tau(\lambda(s_0) + s_1) + \lambda(s_0) \equiv t'' \qquad \text{– B2, B3 –}$$

Induction step

Supposing that P holds for $m \leq \ell - 1$, let us consider $m = \ell \geq 2$.

Let $u' \equiv \tau(\ldots(\tau(\lambda q_1 + s_\ell) + \ldots s_2)$, and

let $u'' \equiv \tau(\ldots(\tau(\lambda(s_0) + s_\ell) + \ldots s_2)$.

$$t' \equiv \tau(\tau(u') + s_1) + \lambda(s_0)$$

$$\overset{+}{\sim}\ \tau(\tau(\tau(u') + \tau(u')) + s_1) + \lambda(s_0) \qquad \text{– B2, B3 –}$$

$$\overset{+}{\sim}\ \tau(u') + \tau(\tau(u') + s_1) + \lambda(s_0) \qquad \text{– B7 –}$$

$$\overset{+}{\sim}\ \tau(u') + \lambda(s_0) + \tau(\tau(u') + s_1) \qquad \text{– B1 –}$$

$$\overset{+}{\sim}\ \tau(u'') + \lambda(s_0) + \tau(\tau(u') + s_1) \qquad \text{– induction hypothesis –}$$

$$\overset{+}{\sim}\ \tau(u'') + \tau(\tau(u') + s_1) + \lambda(s_0) \qquad \text{– B1 –}$$

$$\overset{+}{\sim}\ \tau(\tau(\tau(u') + \tau(u'')) + s_1) + \lambda(s_0) \qquad \text{– B7 –}$$

Since $t'' \equiv \tau(\tau(u'') + s_1) + \lambda(s_0)$, $t' \overset{+}{\sim} t''$ then comes by direct application of the below lemma 4 which shows $\tau(\tau(u') + \tau(u'')) \overset{+}{\sim} \tau(u'')$ □

<u>lemma 4</u> Taking $m \geq 1$ and $s'_0 \equiv \sum_{i=1}^{n} \tau q_i$, let

$v'_m \equiv (\tau(\ldots(\tau(\lambda q_1 + s'_1) + s'_2)\ldots) + s'_m)$, and

$v''_m \equiv (\tau(\ldots(\tau(\lambda(s'_0) + s'_1) + s'_2)\ldots) + s'_m)$.

Then $\tau v''_m \overset{+}{\approx} \tau(\tau v'_m + \tau v''_m)$

<u>proof</u> (by induction over m).

<u>Induction basis</u>. Let $m = 1$.

$v'_1 \equiv (\lambda q_1 + s'_1)$, $v''_1 \equiv (\lambda(s'_0) + s'_1)$

$\tau(\tau v'_1 + \tau v''_1) \equiv \tau(\tau(\lambda q_1 + s'_1) + \tau(\lambda(s'_0) + s'_1))$

$\overset{+}{\approx} \tau(\lambda q_1 + s'_1 + \tau(\lambda(s'_0) + s'_1))$ — B6 —

$\overset{+}{\approx} \tau(\lambda q_1 + \tau(\lambda(s'_0) + s'_1) + s'_1)$ — B2 —

$\overset{+}{\approx} \tau(\tau(\lambda(\tau q_1 + \tau(s'_0)) + s'_1) + s'_1)$ — B7 —

$\overset{+}{\approx} \tau(\tau(\lambda(\tau q_1 + \sum_{i=1}^{n} \tau q_i) + s'_1) + s'_1)$ — B4, B3 —

$\overset{+}{\approx} \tau(\tau(\lambda(s'_0) + s'_1) + s'_1) \overset{+}{\approx} \tau(s'_1 + \tau(s'_1 + \lambda(s'_0)))$

$\overset{+}{\approx} \tau(s'_1 + \lambda(s'_0))$ — B7, B2, B3 —

$\overset{+}{\approx} \tau(\lambda(s'_0) + s'_1) \equiv v''_1$

<u>Induction step</u>

Supposing that the lemma is verified for $m \leq k = l-1$, let us consider $m = l \geq 2$.

$\tau(\tau v'_m + \tau v''_m) \equiv \tau(\tau(\tau v'_k + s'_m) + \tau(\tau v''_k + s'_m))$

$\overset{+}{\approx} \tau(\tau v'_k + s'_m + \tau(\tau v''_k + s'_m))$ — B6 —

$\overset{+}{\approx} \tau(\tau(\tau v''_k + \tau v'_k) + s'_m)$ — B7, B2, B3 —

$\overset{+}{\approx} \tau(\tau v''_k + s'_m)$ — induction hypothesis —

$\equiv \tau v''_m$ □

<u>lemma 5</u> Let $q \equiv (\tau,\tau)(NIL,p)$, where p is a canonical program. Then $q \approx \tilde{q}$, and $\tilde{q}$ verifies $(\forall \lambda \in L)(\tilde{q} \xrightarrow{\tau^* \lambda} q_\lambda$ <u>iff</u> $p \xrightarrow{\tau^* \lambda} p_\lambda \overset{t}{\approx} q_\lambda$).

<u>proof</u> Immediate for $p = NIL$. Two other cases have to be considered, according to the above definition whose notations are re_used.

<u>case 1</u> $p \overset{t}{\approx} (\tau NIL + \sum_{i=1}^{n} \lambda_i \tilde{p}_i)$.

$$q \overset{t}{\approx} (\tau NIL + \tau(\tau NIL + \sum_{i=1}^{n} \lambda_i \tilde{p}_i))$$

$$\approx (\tau(\tau(\tau NIL + \tau NIL) + \sum_{i=1}^{n} \lambda_i \tilde{p}_i)) \qquad \text{– B7 –}$$

$$\approx (\tau NIL + \sum_{i=1}^{n} \lambda_i \tilde{p}_i) \equiv \tilde{q} \qquad \text{– B2, B3 –}$$

<u>case 2</u> $p \overset{t}{\approx} (\sum_{\lambda_i \in \Lambda_0} \lambda_i \tilde{p}_i + \sum_{j=1}^{k} \tau(\sum_{\lambda_i \in \Lambda_j} \lambda_i \tilde{p}_i))$

$$q \overset{t}{\approx} (\tau NIL + \tau(\underline{\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad}))$$

$$\approx (\tau NIL + \underline{\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad}) \qquad \text{– B5 –}$$

$$\approx (\tau NIL + \sum_{\lambda_i \in \Lambda_0} \lambda_i \tilde{p}_i + \sum_{j=1}^{k} \sum_{\lambda_i \in \Lambda_j} \lambda_i \tilde{p}_i) \qquad \text{– B5 –}$$

$$\approx (\tau NIL + \sum_{\lambda_i \in \Lambda} \lambda_i \tilde{p}_i) \overset{t}{\approx} \tilde{q} \qquad \text{– B2 –}$$

□

<u>lemma 6</u>

$$s + \tau(s' + \tau y + \tau y') \overset{+}{\approx} s + s' + \tau y + \tau y'$$

<u>proof</u>

$$s + \tau(s' + \tau y + \tau y')$$

$$\overset{+}{\approx} s + s' + \tau y + \tau y' + \tau(s' + \tau y + \tau y') \qquad \text{– B7, B2, B3 –}$$

$$\overset{+}{\approx} s + s' + \tau y + \tau(\tau y') + \tau(s' + \tau y + \tau y') \qquad \text{– B3 –}$$

$$\overset{+}{\approx} s + s' + \tau y + \tau(\tau y') + s' + \tau y + \tau y' \qquad \text{– B6 –}$$

$$\overset{+}{\approx} s + s' + \tau y + \tau y' \qquad \text{– B3, B2 –}$$

□

<u>Lemma 7</u> Let $q = (\tau,\tau)(p,p')$ be a uniform program, whose subprograms p and p' are canonical programs, both of which different from NIL. Then $q \backsim \tilde{q}$, and $\tilde{q}$ verifies:

$$(\forall \lambda \in L)(\tilde{q} \xrightarrow{\tau * \lambda} q_\lambda \Rightarrow p \xrightarrow{\tau * \lambda} p_\lambda \overset{t}{\backsim} q_\lambda \text{ or } p' \xrightarrow{\tau * \lambda} p'_\lambda \overset{t}{\backsim} q_\lambda).$$

<u>proof</u>. Five different cases will be considered.

<u>case 1.</u>

$p \overset{t}{\backsim} (\tau NIL + \sum_{i \in I_1} \lambda_i \tilde{p}_i)$, $p' \overset{t}{\backsim} (\tau NIL + \sum_{i \in I_2} \lambda_i \tilde{p}_i)$.

Then $I_1 \neq \phi \neq I_2$ comes from the hypothesis.

$$q \overset{t}{\backsim} (\tau p + \tau p')$$

$$\backsim (\tau(\tau NIL + \tau(\sum_{i \in I_1} \lambda_i \tilde{p}_i)) + \tau(\tau NIL + \tau(\sum_{i \in I_2} \lambda_i \tilde{p}_i))) \quad \text{-B5-}$$

$$\backsim (\tau NIL + \tau(\rule{2cm}{0.4pt}) + \tau NIL + \tau(\rule{2cm}{0.4pt})) \quad \text{-B4,B3-}$$

$$\backsim (\tau NIL + \sum_{i \in I_1} \lambda_i \tilde{p}_i + \tau NIL + \sum_{i \in I_2} \lambda_i \tilde{p}_i) \quad \text{-B5-}$$

$$\backsim (\tau NIL + \sum_{i \in I_1 \cup I_2} \lambda_i \tilde{p}_i) \overset{t}{\backsim} \tilde{q}.$$

<u>case 2</u>

$p \overset{t}{\backsim} (\tau NIL + \sum_{i \in I_1} \lambda_i \tilde{p}_i)$

$p' \overset{t}{\backsim} (\sum_{i \in I_2} \lambda_i \tilde{p}_i + \sum_{j=3}^{k} \tau(\sum_{i \in I_j} \lambda_i \tilde{p}_i))$

Then $I_1 \neq \phi \neq I_2 \cup \bigcup \{I_j / 3 \leqslant j \leqslant k\}$.

$$q \overset{t}{\backsim} (\tau(\tau NIL + \sum_{i \in I_1} \lambda_i \tilde{p}_i) + \tau(\sum_{i \in I_2} \lambda_i \tilde{p}_i + \sum_{j=3}^{k} \tau(\sum_{i \in I_j} \lambda_i \tilde{p}_i)))$$

$$\backsim (\tau NIL + \tau(\rule{2cm}{0.4pt}) + \tau(\rule{6cm}{0.4pt})) \quad \text{-B5,B4,B3}$$

$$\backsim (\tau NIL + \rule{2cm}{0.4pt} + \rule{6cm}{0.4pt}) \quad \text{-B5-}$$

$$\backsim (\tau NIL + \sum_{i \in I_1} \lambda_i \tilde{p}_i + \sum_{i \in I_2} \lambda_i \tilde{p}_i + \sum_{j=3}^{k} \sum_{i \in I_j} \lambda_i \tilde{p}_i) \quad \text{-B5-}$$

$$\backsim (\tau NIL + \sum_{i \in \bigcup\{I_j / 1 \leqslant j \leqslant k\}} \lambda_i \tilde{p}_i) \overset{t}{\backsim} \tilde{q}.$$

<u>case 3</u> $p \overset{t}{\backsim} (\sum_{i \in I_1} \lambda_i \tilde{p}_i)$, $p' \overset{t}{\backsim} (\sum_{i \in I_2} \lambda_i \tilde{p}_i)$,
where $I_1 \neq \phi \neq I_2$ and $\lambda_i = \lambda_j$ <u>iff</u> $i = j$.

If I_1 and I_2 are incomparable sets, then $(\tau p + \tau p') \overset{t}{\backsim} \tilde{q}$ comes directly from the definition of canonical programs. Else, letting for instance $I_2 \subseteq I_1$, one has

$(\tau p + \tau p') \backsim (\sum_{i \in I_1} \lambda_i \tilde{p}_i + \tau (\sum_{i \in I_2} \lambda_i \tilde{p}_i))$ — B6 —

$\backsim (\sum_{i \in I_1 - I_2} \lambda_i \tilde{p}_i + \sum_{i \in I_2} \lambda_i \tilde{p}_i + \tau (\sum_{i \in I_2} \lambda_i \tilde{p}_i))$

$\backsim (\underline{\qquad\qquad\qquad\qquad} + \tau (\sum_{i \in I_2} \lambda_i (\tau \tilde{p}_i + \tau \tilde{p}_i)))$ — B7 —

$\backsim (\sum_{i \in I_1 - I_2} \lambda_i \tilde{p}_i + \tau (\sum_{i \in I_2} \lambda_i \tilde{p}_i)) = r$ — B2, B3 —

Now, if $I_1 \neq I_2$, then $r \overset{t}{\backsim} \tilde{q}$ comes from the definition of canonical forms, else $q \backsim r = (\tau)(p') \backsim p'$ which is a canonical program, thus $\tilde{q} \overset{t}{\backsim} p'$ and $q \backsim \tilde{q}$.

<u>case 4</u> $p \overset{t}{\backsim} (s) \equiv (\sum_{i \in I_1} \lambda_i \tilde{p}_i)$, $p' \overset{t}{\backsim} (s' + y')$,
$s' \equiv \sum_{i \in I_0} \lambda_i \tilde{p}_i$, $y' \equiv \sum_{j=2}^{k} \tau (\sum_{i \in I_j} \lambda_i \tilde{p}_i)$,
where $I_1 \neq \phi \neq I_2 \cup I_3 \ldots \cup I_k$ and $\lambda_i = \lambda_j$ <u>iff</u> $i = j$.

$(\tau p + \tau p') \backsim (\tau(s) + \tau(\tau(y') + s'))$ — B4, B3, B1 —

$\backsim (\tau(\tau(\tau(y') + \tau(s)) + s'))$ — B7 —

$\backsim (s' + \tau(s) + \tau(y'))$ — B3, B4, B1 —

$\backsim (s' + \tau(s) + y')$ — B4, B3 —

$\backsim (\sum_{i \in I_0} \lambda_i \tilde{p}_i + \sum_{j=1}^{k} \tau (\sum_{i \in I_j} \lambda_i \tilde{p}_i))$.

Now let $J = \{0\} \cup \{ j \in [1,k] \,/\, (\exists j' \in [1,k])(j' \neq j \text{ and } I_j \supset I_{j'}) \}$, and $J' = [1,k] - J$. By repeated application of B6, one obtains :

$$(\tau p+\tau p') \backsim \Big(\sum_{j\in J} \sum_{i\in I_j} \lambda_i \tilde{p}_i + \sum_{j\in J'} \tau\big(\sum_{i\in I_j} \lambda_i \tilde{p}_i \big)\Big)$$

Let $I' = (\cup\{I_j / j\in J\}) \setminus (\cup\{I_j / j \in J'\})$. By repeated application of property $\lambda p + \tau(\lambda p + s) \overset{+}{\backsim} \tau(\lambda p + s)$ which derives from B7, B2, B3, one finally obtains :

$$(\tau p+\tau p') \backsim \Big(\sum_{i\in I'} \lambda_i \tilde{p}_i + \sum_{j\in J'} \tau\big(\sum_{i\in I_j} \lambda_i \tilde{p}_i \big)\Big) \equiv r .$$

By construction of r, one has $r \overset{t}{\backsim} \tilde{r}$. As $q \backsim r \Longrightarrow q \sim r \Longrightarrow \tilde{q} \overset{t}{\backsim} \tilde{r}$ comes from proposition 2 and from the definition of $\sim$, $\tilde{q} \backsim r \backsim q$ is still verified.

<u>case 5</u>. $p \overset{t}{\backsim} (s+y)$, $p' \overset{t}{\backsim} (s'+y')$,

$s \equiv \sum_{i\in I'_0} \lambda_i \tilde{p}_i$, $y \equiv \sum_{j=1}^{\ell} \tau\big(\sum_{i\in I_j} \lambda_i \tilde{p}_i \big)$,

$s' \equiv \sum_{i\in I''_0} \lambda_i \tilde{p}_i$, $y' \equiv \sum_{j=\ell+1}^{k} \tau\big(\sum_{i\in I_j} \lambda_i \tilde{p}_i \big)$,

$I_1 \cup \dots \cup I_\ell \neq \emptyset \neq I_{\ell+1} \cup \dots \cup I_k$, $\lambda_i = \lambda_j$ <u>iff</u> $i=j$.

$$\begin{aligned}
q &\backsim (\tau p + \tau p') \backsim (\tau(s+\tau(y)) + \tau(\tau(y')+s')) && \text{– B4, B3, B2 –}\\
&\backsim (\tau(\tau(\tau(y') + \tau(s+\tau(y))) + s')) && \text{– B7 –}\\
&\backsim (s' + \tau(y') + \tau(\tau(y)+s)) && \text{– B3, B2, B4 –}\\
&\backsim (s' + \tau(\tau(\tau(y)+\tau(y')) + s)) && \text{– B7 –}\\
&\backsim (s' + \tau(s+\tau(y) + \tau(y'))) && \text{– B2, B4, B3 –}\\
&\backsim (s+s'+\tau(y) + \tau(y')) && \text{– lemma 6 –}\\
&\backsim (s+s'+y+y') && \text{– B4, B3 –}\\
&\backsim \Big(\sum_{i\in I'_0 \cup I''_0} \lambda_i \tilde{p}_i + \sum_{j=1}^{k} \tau\big(\sum_{i\in I_j} \lambda_i \tilde{p}_i \big)\Big).
\end{aligned}$$

Now, letting $I_0 = I'_0 \cup I''_0$, $q \backsim \tilde{q}$ may be shown the same way that in case 4, using B6 and B7 □

Lemma 8 Let $q = (\tau, \tau, \dots, \tau)(q_1, q_2, \dots, q_k)$, with $k \geq 2$, be a uniform program whose subprograms q_i are canonical programs. Then $q \backsim \tilde{q}$ and $\tilde{q}$ verifies:

$(\forall \lambda \in L)(\tilde{q} \xrightarrow{\tau * \lambda} q_\lambda \Longrightarrow (\exists i \in [1, k])(q_i \xrightarrow{\tau * \lambda} p'_\lambda \overset{t}{\backsim} q_\lambda))$.

proof Immediate from lemmas 5 and 7 since for $k > 2$, $q \backsim (\tau, \tau)(q_1, (\tau, \dots, \tau)(q_2, \dots, q_k)$ derives along B4 and B3, and $q_\lambda \overset{t}{\backsim} p_\lambda$ implies $\mathcal{C}[q_\lambda] \overset{t}{\backsim} \mathcal{C}[p_\lambda]$. □

Lemma 9 Let $q = (\lambda_{i_1}, \dots, \lambda_{i_k}, \tau)(\tilde{p}_{i_1}, \dots, \tilde{p}_{i_k}, p')$ be a uniform program whose subprograms $\tilde{p}_{i_j}$ and p' are canonical programs. Then $q \backsim \tilde{q}$, and $(\forall \lambda \in L)$ $(\tilde{q} \xrightarrow{\tau * \lambda} q_\lambda \Longrightarrow q \xrightarrow{\tau * \lambda} p_\lambda \overset{t}{\backsim} q_\lambda)$.

proof Let $I_0 = \{ i_j \mid j \in [1, k] \}$.

Then $q \backsim (\sum_{i \in I_0} \lambda_i \tilde{p}_i + \tau p')$.

Four different cases will be considered.

case 1 $p' = NIL$.

Then $q \overset{t}{\backsim} \tilde{q}$.

case 2 $p' \overset{t}{\backsim} (\tau NIL + \sum_{i \in I_1} \lambda_i \tilde{p}_i)$.

$q \backsim (\sum_{i \in I_0} \lambda_i \tilde{p}_i + \tau(\tau NIL + \tau(\sum_{i \in I_1} \lambda_i \tilde{p}_i)))$ — B5 —

$\backsim (\tau NIL + \sum_{i \in I_0} \lambda_i \tilde{p}_i + \tau(\sum_{i \in I_1} \lambda_i \tilde{p}_i))$ — B4, B3, B1 —

$\backsim (\tau NIL + \sum_{i \in I_0 \cup I_1} \lambda_i \tilde{p}_i) \overset{t}{\backsim} \tilde{q}$ — B5, B2 —

case 3 $p' \overset{t}{\backsim} (\sum_{i \in I_1} \lambda_i \tilde{p}_i)$, with $I_1 \neq \phi$.

$q \backsim (\sum_{i \in I_0} \lambda_i \tilde{p}_i + \tau (\sum_{i \in I_1} \lambda_i \tilde{p}_i)) \equiv r$, and $r \backsim \tilde{q}$ may be proved by repeated application of B6 and B7, the same way as it has been done in case 4 of lemma 7.

case 4 $p' \overset{t}{\backsim} (s' + y)$, $s' \equiv \sum_{i \in I'_0} \lambda_i \tilde{p}_i$,

$y \equiv \sum_{j=1}^{k} \tau (\sum_{i \in I_j} \lambda_i \tilde{p}_i)$, $I_1 \cup \ldots \cup I_k \neq \phi$,

$\lambda_i = \lambda_j$ iff $i = j$.

Let $s \equiv \sum_{i \in I_0} \lambda_i \tilde{p}_i$, then

$q \backsim (s + \tau (s' + y)) \backsim (s + \tau (s' + \tau (y)))$ — B4, B3 —

$\backsim (s + \tau (s' + \tau (y) + \tau (y)))$ — B2 —

$\backsim (s + s' + \tau (y) + \tau (y))$ — lemma 6 —

$\backsim (s + s' + \tau (y))$ — B2, B4, B3 —

$\backsim (\sum_{i \in I_0 \cup I'_0} \lambda_i \tilde{p}_i + \sum_{j=1}^{k} \tau (\sum_{i \in I_j} \lambda_i \tilde{p}_i)) \equiv r$,

and $r \backsim \tilde{q}$ may again be proved the same way as in case 4 of lemma 7 □

<u>lemma 10</u>. For $p \in W_\Sigma$, let $can(p)$ denote any canonical program p' such that $p \backsim p'$. Then $can(p)$ exists for any p.

<u>proof</u> From proposition 3, there always exists a uniform program $\hat{p} \backsim p$, and from proposition 4, $\hat{p} \equiv q \backsim \tilde{q}$ where $\tilde{q}$ is a canonical program, thus $p \backsim \tilde{q}$ □

<u>lemma 11</u> For $p, p' \in W_\Sigma$, $p \sim p' \Rightarrow can(p) \overset{t}{\backsim} can(p')$.

<u>proof</u>. From proposition 2, one has implications

$p \backsim can(p) \Rightarrow p \sim can(p)$

$p' \backsim can(p') \Rightarrow p' \sim can(p')$.

Therefore, $p \sim p' \Rightarrow can(p) \sim can(p')$.

Now, $can(p) \sim can(p') \Rightarrow can(p) \overset{t}{\backsim} can(p')$ comes from the definition of canonical forms □

<u>lemma 12</u> Let $\mathcal{C}[.] = (\mu_1, \mu_2, \dots, \mu_n)(\cdot, p_2, \dots, p_n)$, with $n \geqslant 1$, then $u \sim v \Rightarrow \mathcal{C}[u] \sim \mathcal{C}[v]$.

<u>proof</u>

$$\pi(\mathcal{C}[u]) = (\mu_1 \pi u + \sum_{i=2}^{n} \mu_i \pi p_i)$$

$$\pi(\mathcal{C}[v]) = (\mu_1 \pi v + \sum_{i=2}^{n} \mu_i \pi p_i)$$

$$u \sim v \Rightarrow \pi u \sim \pi v \Rightarrow \pi(\mathcal{C}[u]) \sim \pi(\mathcal{C}[v]) \Rightarrow \mathcal{C}[u] \sim \mathcal{C}[v]. \ \square$$

<u>lemma 13</u> For u and $v \in W_{\Sigma}$ and $s \in \mathcal{S}$, $u \approx v \Rightarrow u[s] \approx v[s]$.

<u>proof</u>

$u \approx v \Rightarrow u \sim v$ since $\pi p = p$ for any $p \in W_{\Sigma}$;

one has therefore to prove that $u \sim v \Rightarrow \pi(u[s]) \sim \pi(v[s])$.

For $P = \{p_1, \dots, p_n\} \in P_f(W_{\Sigma})$, let $P[s] = \{\pi(p_i[s])\}$.

The above implication is a particular case of the more general implication

$(\forall U, V \in P_f(W_{\Sigma}))\ (U \sim V \Rightarrow U[s] \sim V[s])$

which we shall now establish, using induction on the maximal length ℓ of experiments which are feasible on members of $(U) \cup (V)$.

<u>Induction basis</u>.

Let $\ell = 0$. Then $U[s] = U$ and $V[s] = V$, thus $U[s] \sim V[s]$.

Induction step

Let us suppose that the property holds for $\ell < k$ and consider now the case where $\ell = k \geq 1$

One has to verify the following points :

i) for $\Lambda \in P_f(L) \setminus \phi$, $U[s] \downarrow \Lambda \Rightarrow V[s] \downarrow \Lambda$ (and vice-versa)

ii) for $\lambda \in L$, $U[s] \xrightarrow{\lambda} U'$ and $V[s] \xrightarrow{\lambda} V' \Rightarrow U' \sim V'$.

first point

Let $\Lambda \in P_f(L) \setminus \phi$, and let $\Lambda' = \{ \lambda' \in \Lambda \,/\, (\exists \lambda \in L)(\lambda' = s\lambda) \}$.

We shall prove that $U[s] \downarrow \Lambda \Rightarrow V[s] \downarrow \Lambda$.

If $\Lambda' = \phi$, then $V[s] \downarrow \Lambda$ comes immediately , since $V \neq \phi$.

Let us suppose in the sequel that $\Lambda' \neq \phi$.

From the definition of $\downarrow$, $U[s] \downarrow \Lambda \Rightarrow U[s] \downarrow \Lambda'$.

Now, as $s\mu = \tau$ $\underline{\text{iff}}$ $\mu = \tau$, $U[s] \downarrow \Lambda'$ implies that there exists $u \in U$ and $u' = (\sum_{i \in I} \lambda_i \, u'_i)$ such that

$u \xrightarrow{\tau^*} u'$ $\underline{\text{and}}$ $\pi(u[s]) \xrightarrow{\tau^*} \pi(u'[s])$ $\underline{\text{and}}$

$\{ i \in I \,/\, s\lambda_i = \tau \} = \phi$ $\underline{\text{and}}$ $\{ s\lambda_i \,/\, i \in I \} \cap \Lambda' = \phi$.

Let $s^{-1}(\Lambda')$ be the maximal subset of L such that $s(s^{-1}(\Lambda')) = \Lambda'$; then $s^{-1}(\Lambda') \in P_f(L)$ comes from the definition of the set $\mathcal{S}$ of renamings , and one has thus from the above :

$u \xrightarrow{\tau^*} u'$ $\underline{\text{and}}$ $\{ i \in I \,/\, \lambda_i = \tau \} = \phi$ $\underline{\text{and}}$ $\{ \lambda_i \,/\, i \in I \} \cap s^{-1}(\Lambda') = \phi$

$\Rightarrow U \downarrow s^{-1}(\Lambda')$,

which implies $V \downarrow s^{-1}(\Lambda')$ from the hypothesis $U \sim V$.

Therefore, there must exist $v \in V$ and $v' = (\sum_{j \in J} \nu_j v'_j)$ s.t.
$v \xrightarrow{\tau^*} v'$ and $\{ j \in J / \nu_j = \tau \} = \phi$ and $\{ \nu_j / j \in J \} \cap s^{-1}(\Lambda') = \phi$,
which implies
$\pi(v[s]) \xrightarrow{\tau^*} \pi(v'[s])$ and
$\{ j \in J / s\nu_j = \tau \} = \phi$ and $\{ s\nu_j / j \in J \} \cap \Lambda = \phi$,
that is still $V[s] \downarrow \Lambda$.

second point

Let $U[s] \xrightarrow{\lambda} U'$ and $V[s] \xrightarrow{\lambda} V'$.
If $s^{-1}\{\lambda\} = \phi$, then necessarily $U' = \phi = V' \Rightarrow U' \sim V'$.
Let now $s^{-1}\{\lambda\} = \{\lambda_1, \dots, \lambda_n\}$, with $n \geq 1$.
Since $\ell \geq 1$ implies $U \neq \phi \neq V$, there must exist U_i, V_i $\in P_f(W_\Sigma)$ such that $U \xrightarrow{\lambda_i} U_i$ and $V \xrightarrow{\lambda_i} V_i$ for $1 \leq i \leq n$.
From property $s\mu = \tau$ iff $\mu = \tau$ and from the definition of $P[s]$, one has necessarily :

$$U' = (\bigcup_{i=1}^{n} (U_i))[s] = \bigcup_{i=1}^{n} (U_i [s])$$

$$V' = (\bigcup_{i=1}^{n} (V_i))[s] = \bigcup_{i=1}^{n} (V_i [s])$$

Now, $U \sim V \Rightarrow (\forall i)(U_i \sim V_i)$ – from the def. of $\sim$ –
$\Rightarrow (\forall i)(U_i[s] \sim V_i[s])$ – from the induction hypothesis –
$\Rightarrow \bigcup_{i=1}^{n} (U_i[s]) \sim \bigcup_{i=1}^{n} (V_i[s])$ – from lemma 1 –
$\Rightarrow U' \sim V'$

□

lemma 14. For $u, v, p \in W_\Sigma$,
$u \sim v \Rightarrow (p/u) \sim (p/v)$ and $(u/p) \sim (v/p)$.

proof For symetry considerations, one may content to establish the implication $u \approx v \Rightarrow (u/p) \approx (v/p)$.

$u \approx v \Rightarrow u \sim v$ since $\pi p = p$ for any $p \in W_\Sigma$;

one has therefore to prove that $u \sim v \Rightarrow \pi(u/p) \sim \pi(v/p)$.

For $P = \{p_1, \dots, p_n\} \in P_f(W_\Sigma)$, $Q = \{q_1, \dots, q_m\} \in P_f(W_\Sigma)$, let $P/Q = \{\pi(p_i / q_j)\}$.

The above implication is a particular case of the more general implication :

$(\forall U, V, P \in P_f(W_\Sigma))\ (U \sim V \Rightarrow (U/P) \sim (V/P))$,

which we shall now establish, using induction on the sum of maximal lengths of experiments which are feasible on members of U and P respectively, let ℓ that sum.

induction basis. Let $\ell = 0$.

If $P = \emptyset$, then $(U/P) = \emptyset = (V/P)$.

If $U = \emptyset$, then $U \sim V \Rightarrow V = \emptyset$, thus $(U/P) = \emptyset = (V/P)$.

If $V = \emptyset$, then $U \sim V \Rightarrow U = \emptyset$, thus $(U/P) = \emptyset = (V/P)$.

If any one of U, V, P differs from $\emptyset$, one has from the definition of P/Q :

$(\forall \lambda \in L)\ (\forall W \in P_f(W_\Sigma))\ (((U/P) \xrightarrow{\lambda} W$ <u>or</u> $(V/P) \xrightarrow{\lambda} W) \Rightarrow W = \emptyset)$

$(\forall \Lambda \in P_f(L) \setminus \emptyset)\ ((U/P) \downarrow \Lambda$ <u>and</u> $(V/P) \downarrow \Lambda)$,

from which we can conclude $(U/P) \sim (V/P)$.

induction step. Let us suppose that the property holds for $\ell < k$, and consider the case where $\ell = k \geqslant 1$.

If any one of U, V, P equals $\emptyset$, then $(U/P) \sim (V/P)$ may be concluded the same way as above. Let us now assume that any one of U, V, P differs from $\emptyset$. One has to verify the following points:

i) for $\Lambda \in P_f(L) \setminus \emptyset$, $(U/P) \downarrow \Lambda \Rightarrow (V/P) \downarrow \Lambda$ (and vice-versa),

ii) for $\lambda \in L$, $(U/P) \xrightarrow{\lambda} U'$ and $(V/P) \xrightarrow{\lambda} V' \Rightarrow U' \sim V'$.

first point

From the definition of $\downarrow$, $(U/P) \downarrow \Lambda$ implies that there exist $u \in U$, $p \in P$, and $\omega \in L^*$ such that, letting $n = 0$ if ω is empty and $\omega = \lambda_1 \lambda_2 \dots \lambda_n$ in other cases, one has

$$u = u_0 \xrightarrow{\tau^* \lambda_1} u_1 \dots \xrightarrow{\tau^* \lambda_n} u_n \xrightarrow{\tau^*} u' = (\sum_{i \in I} \mu_i u'_i)$$

$$p = p_0 \xrightarrow{\tau^* \overline{\lambda_1}} p_1 \dots \xrightarrow{\tau^* \overline{\lambda_n}} p_n \xrightarrow{\tau^*} p' = (\sum_{j \in J} \nu_j p'_j)$$

where the following properties are verified :

$(\forall i \in I)(\mu_i \neq \tau)$ and $(\forall j \in J)(\nu_j \neq \tau)$

$\{\mu_i / i \in I\} \cap \{\overline{\nu_j} / j \in J\} = \emptyset$

$\{\mu_i / i \in I\} \cap \Lambda = \emptyset = \Lambda \cap \{\nu_j / j \in J\}$.

Let $U = U_0 \xrightarrow{\lambda_1} U_1 \xrightarrow{\lambda_2} U_2 \dots \xrightarrow{\lambda_n} U_n$, then $u_n \in U_n$.

Let $N = \{\overline{\nu_j} / j \in J\}$, then $U_n \downarrow (\Lambda \cup N)$.

$U \sim V \Rightarrow \exists V_1, \dots, V_n \in P_f(W_\Sigma) \setminus \emptyset$ such that

$V = V_0 \xrightarrow{\lambda_1} V_1 \xrightarrow{\lambda_2} V_2 \dots \xrightarrow{\lambda_n} V_n$ and $U_n \sim V_n$,

whence $V_n \downarrow (\Lambda \cup N)$.

One can therefore find $v_1, \dots, v_n, v' \in W_\Sigma$ s.t.

$$v = v_0 \xrightarrow{\tau^* \lambda_1} v_1 \dots \xrightarrow{\tau^* \lambda_n} v_n \xrightarrow{\tau^*} v' = (\sum_{h \in H} \sigma_h v'_h) ,$$

where the following properties are verified :

$(\forall h \in H)(\sigma_h \neq \tau)$ and $(\forall j \in J)(\nu_j \neq \tau)$

$\{\sigma_h / h \in H\} \cap \{\overline{\nu_j} / j \in J\} = \phi$

$\{\sigma_h / h \in H\} \cap \Lambda = \phi = \Lambda \cap \{\nu_j / j \in J\}$.

Now, $\pi(v/p) \in (V/P)$, $\pi(v/p) \xrightarrow{\tau^*} \pi(v'/p')$, and

$\pi(v'/p') = (\sum_{h \in H} \sigma_h \, \pi(v'_h / p') + \sum_{j \in J} \nu_j \, \pi(v'/p'_j))$, which finally imply that $(V/P) \downarrow \Lambda$.

second point

We shall first introduce some convenient notations.

Define $L^* = L^+ \cup \{1\}$, where 1 represents the empty word.

For $\omega \in L^*$, notation $\overline{\omega}$ will stand for 1 if $\omega = 1$, or else $\overline{\omega} = \overline{\lambda_1} \dots \overline{\lambda_n}$ if $\omega = \lambda_1 \dots \lambda_n \in L^+$.

For $Q, Q' \in P_f(W_\Sigma)$ and $\omega = \lambda_1 \dots \lambda_n \in L^+$, let notation $Q \overset{\omega}{\Longrightarrow} Q'$ stand for $\exists Q_0, Q_1, \dots, Q_n \in P_f(W_\Sigma)$ such that $Q = Q_0 \xrightarrow{\lambda_1} Q_1 \dots \xrightarrow{\lambda_n} Q_n = Q'$, thus $Q_i \neq \phi$ for $i < n$.

For $Q, Q' \in P_f(W_\Sigma)$, let notation $Q \overset{1}{\Longrightarrow} Q'$ stand for $Q' = \{ q' \in W_\Sigma / (\exists q \in Q)(q \xrightarrow{\tau^*} q') \}$.

From the definition of $\sim$, it is easily shown that

$(\forall Q, Q' \in P_f(W_\Sigma))((Q \overset{1}{\Longrightarrow} Q') \Rightarrow (Q \sim Q'))$,

since $(\forall \Lambda \in P_f(L) \setminus \phi).(Q \downarrow \Lambda$ iff $Q' \downarrow \Lambda)$ and

$(\forall \lambda \in L)(\forall Q_\lambda \in P_f(W_\Sigma))(Q \xrightarrow{\lambda} Q_\lambda$ iff $Q' \xrightarrow{\lambda} Q_\lambda)$.

Using the above notations, we shall now prove that for $U \sim V$, $(U/P) \xrightarrow{\lambda} U'$ and $(V/P) \xrightarrow{\lambda} V' \Rightarrow U' \sim V'$.

Define $\mathcal{W} = \{ \omega \in L^* / \ U \xRightarrow{\omega} U_\omega$ for some $U_\omega \}$, and $\mathcal{W}' = \{ \omega \in L^* / \ V \xRightarrow{\omega} V_\omega$ for some $V_\omega \}$; then $\mathcal{W}$ and $\mathcal{W}'$ are finite subsets of L^* and $U \sim V \Rightarrow \mathcal{W} = \mathcal{W}'$.
Define $\mathcal{W}_\lambda = \{ \omega \in L^* / \ U \xRightarrow{\omega . \lambda} U_\omega$ for some $U_\omega \}$, and $\mathcal{W}'_\lambda = \{ \omega \in L^* / \ V \xRightarrow{\omega . \lambda} V_\omega$ for some $V_\omega \}$; then $\mathcal{W}_\lambda$ and $\mathcal{W}'_\lambda$ are finite subsets of L^* and $U \sim V \Rightarrow \mathcal{W}_\lambda = \mathcal{W}'_\lambda$.

Let $U'_1 = \bigcup_{\omega \in \mathcal{W}} \{ (U''/P'') / \ U \xRightarrow{\omega} U_\omega \xRightarrow{1} U''$ and $P \xRightarrow{\overline{\omega} . \lambda} P'' \}$,
and $U'_2 = \bigcup_{\omega \in \mathcal{W}_\lambda} \{ (U''/P'') / \ U \xRightarrow{\omega . \lambda} U''$ and $P \xRightarrow{\overline{\omega}} P_\omega \xRightarrow{1} P'' \}$,
then $U' = (U'_1) \cup (U'_2)$ comes from the definition of (P/Q).
Let $V'_1 = \bigcup_{\omega \in \mathcal{W}} \{ (V''/P'') / \ V \xRightarrow{\omega} V_\omega \xRightarrow{1} V''$ and $P \xRightarrow{\overline{\omega} . \lambda} P'' \}$,
and $V'_2 = \bigcup_{\omega \in \mathcal{W}_\lambda} \{ (V''/P'') / \ V \xRightarrow{\omega . \lambda} V''$ and $P \xRightarrow{\overline{\omega}} P_\omega \xRightarrow{1} P'' \}$,
then $V' = (V'_1) \cup (V'_2)$.

For any $\omega \in \mathcal{W}$, U'' and $P'' \in P_f(W_\Sigma)$ such that $U \xRightarrow{\omega} U_\omega \xRightarrow{1} U''$ and $P \xRightarrow{\overline{\omega} . \lambda} P''$, let ℓ' be the sum of maximal lengths of experiments which are feasible on respective members of U'' and P'' ; then $\ell' < \ell = k$, and from the induction hypothesis $U'' \sim U_\omega$ implies that $(U''/P'') \sim (U_\omega / P'')$. One has therefore from lemma 1 :
$U'_1 = \bigcup_{\omega \in \mathcal{W}} \{ (U''/P'') / \ U \xRightarrow{\omega} U''$ and $P \xRightarrow{\overline{\omega} . \lambda} P'' \}$,
and similarly,
$V'_1 = \bigcup_{\omega \in \mathcal{W}} \{ (V''/P'') / \ V \xRightarrow{\omega} V''$ and $P \xRightarrow{\overline{\omega} . \lambda} P'' \}$.

$U \sim V \Rightarrow (\forall \omega \in \mathcal{W}) (U \xRightarrow{\omega} U''$ and $V \xRightarrow{\omega} V''$ imply $U'' \sim V'')$,
thus $U'_1 \sim V'_1$ comes from the induction hypothesis and

from lemma 1.

$U \sim V \Rightarrow (\forall \omega \in \mathcal{W}_{\lambda})(U \xRightarrow{\omega . \lambda} U''$ and $V \xRightarrow{\omega . \lambda} V''$ imply $U'' \sim V'')$, thus $U'_2 \sim V'_2$ follows from the induction hypothesis and from lemma 1.

As $U' = (U'_1) \cup (U'_2)$ and $V' = (V'_1) \cup (V'_2)$, $U' \sim V'$ finally comes by one more application of lemma 1. □

Imprimé en France
par
l'Institut National de Recherche en Informatique et en Automatique