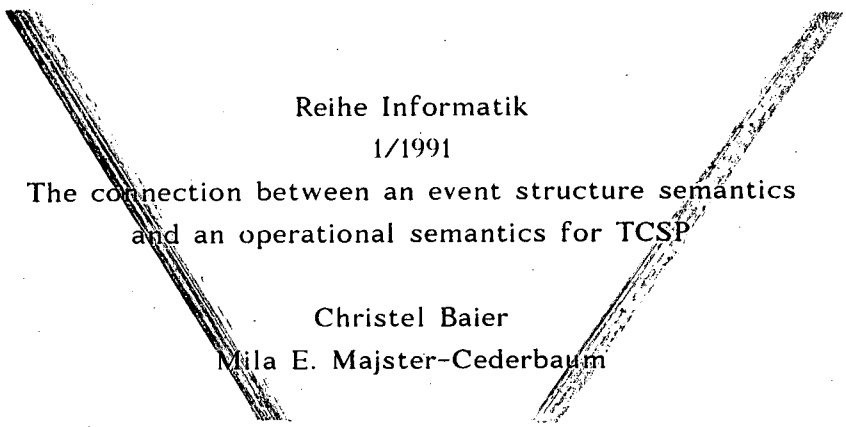


Reihe Informatik

1/1991

The connection between an event structure semantics and an operational semantics for TCSP

Christel Baier

Mila E. Majster-Cederbaum

Februar 1991

The connection between an event structure semantics and an operational semantics for TCSP

Christel Baier
Mila E. Majster-Cederbaum
Fakultät für Mathematik und Informatik
Universität Mannheim
Seminargebäude A5
6800 Mannheim 1, FRG

1991

1 Introduction

Various formalisms have been proposed in the past for the description of nondeterministic concurrent systems, the most well- known of which are CCS [14,15], ACP [2] and TCSP [6,13,17,19]. These languages or calculi have been given a variety of semantical descriptions [1,2,3,4,5,6,7,8,10,11,12,18,20,21,22]. A first classification of this semantics distinguishes between interleaving and noninterleaving models.

In noninterleaving models as [3,4,8,10,11,12,18,21,23] an attempt is made to capture ' true parallelism '. In such models a parallel construct as e.g. $P_1 = \alpha.stop \parallel \beta.stop$, that specifies the parallel execution of the process $\alpha.stop$ (that first performs the action α and then stops) with the process $\beta.stop$, gets a different meaning than the process $P_2 = \alpha.\beta.stop \,\Box\, \beta.\alpha.stop$ that specifies the choice between $\alpha.\beta.stop$ and $\beta.\alpha.stop$. In interleaving models as [1,2,5,6,7,19] concurrency is reduced to nondeterministic behaviour by arbitrary interleaving of atomic actions, hence P_1 and P_2 have the same meaning in these approaches.

The purpose of this paper is to study the relationship between two semantic specifications of full TCSP [17]. The first specification is an operational interleaving description using a transition system, while the second is a noninterleaving model based on labelled event structures [16,23,24,25].

In an earlier paper [11] it has been shown for *finite* TCSP processes without recursion and *div* that the interleaving transitions system based description and

the respective noninterleaving event structure semantics are consistent. It was an open problem, if this result also holds for full TCSP, i.e. including recursion, see [11]. As recursion is a very powerful and indispensable tool for the definition and modelling of processes, it is an interesting question if the consistency result carries over to full TCSP. We show here that this question has a positive answer. The result is particulary interesting as it not only relates an interleaving with a noninterleaving specification but also relates at the same time an operational transition system based specification with a compositional one, that describes the meaning of processes via structural induction using semantic operators.

2 The syntax of guarded TCSP

Let *Comm* be the set of possible communications. A special action τ, as in CCS, is introduced to describe internal actions which may not communicate. For notational convenience, τ is allowed to occur syntactially in expressions denoting processes.

So let the set *Act* of *actions* be defined as

$$Act := Comm \cup \{ \tau \}.$$

Let *Idf* be a set of identifiers which will serve as variables for programs. The set $TCSP$ of *TCSP terms* is defined by the following production system [6,17,19]:

$$P := stop \mid \alpha.P \mid div \mid P \; or \; Q \mid P \,\Box\, Q \mid$$
$$P \parallel_A Q \mid P \setminus \beta \mid x \mid fix\ x.P,$$

where $\alpha \in Act$, $\beta \in Comm$, $A \subseteq Comm$, $x \in Idf$.

2.1 Definition :
An occurence of an identifier x is called *free* in a term $P \in TCSP$ iff it does not occur within a subterm of the form $fix\ x.Q$. A TCSP term P is said to be *closed* iff it does not contain identifiers which occur free in P.

An identifier x is *guarded* in a term $P \in TCSP$ iff each free occurence of x in P is in the scope of a prefixing operation $Q \mapsto \alpha.Q$.
The guardedness condition will be needed for the event structure semantics. It is not necessary for the operational specification.
A term $P \in TCSP$ is called *guarded* iff in each subterm $fix\ x.Q$ of P the identifier x is guarded in Q.

Let $GTCSP$ be the set of all guarded TCSP terms. A *GTCSP process* is a closed, guarded TCSP term.

2.2 Definition :
Let $P, A_1, \ldots, A_n \in GTCSP$ and $x_1, \ldots, x_n \in Idf$ pairwise distinct identifiers. The GTCSP term

$$P[A_1/x_1, \ldots, A_n/x_n] \text{ or shortly } P[\tilde{A}/\tilde{x}]$$

arises from P by replacing each occurence of an identifier x in a subterm $fix\ x.Q$ of P by an identifier which does not appear in Q and in $A_1, \ldots, A_n$ and then by substituting each free occurrence of the identifiers $x_1, \ldots, x_n$ in P simultaneously by the GTCSP terms $A_1, \ldots, A_n$.

3 Transition systems

3.1 Definition :
$A = (S, L, \rightarrow, q_0)$ is called a *(labelled) transition system* iff

1. S is a set of *states*.
2. L is a set of *labels*.
3. $\rightarrow \subseteq S \times L \times S$, where we will write $p \xrightarrow{\alpha} q$ instead of $(p, \alpha, q) \in \rightarrow$.
4. $q_0 \in S$, q_0 is called the *initial state* of A.

3.2 Definition :
Two equally labelled transition systems $A_i = (S_i, L, \rightarrow_i, q_i)$, $i = 1, 2$, are *bisimular* $(A_1 \approx A_2)$ if there exists a *bisimulation* R between A_1 and A_2, i.e. a relation $R \subseteq S_1 \times S_2$ with $(q_1, q_2) \in R$ and, for all $(p, q) \in R$:

1. Whenever $p \xrightarrow{\alpha}_1 p'$ for some $p' \in S_1$ then there exists some $q' \in S_2$ with $(p', q') \in R$ and $q \xrightarrow{\alpha}_2 q'$

and symmetrically

2. whenever $q \xrightarrow{\alpha}_2 q'$ for some $q' \in S_2$ then there exists some $p' \in S_1$ with $(p', q') \in R$ and $p \xrightarrow{\alpha}_1 p'$.

4 An interleaving transition system based description for TCSP

Following Plotkin [20], Olderog [17] gives an operational semantics for a process P by structural induction on the syntactic structure of P as below.

4.1 Definition :
Let $\rightarrow$ be the ternary relation on TCSP that is defined as follows :

1. **Prefixing** $\alpha.P \xrightarrow{\alpha} P$

2. **Internal nondeterminism** $P \text{ or } Q \xrightarrow{\tau} P$, $P \text{ or } Q \xrightarrow{\tau} Q$

3. **External nondeterminism**
   External choice : $\dfrac{P \xrightarrow{\alpha} P'}{P \,\Box\, Q \xrightarrow{\alpha} P'}$, $\dfrac{Q \xrightarrow{\alpha} Q'}{P \,\Box\, Q \xrightarrow{\alpha} Q'}$, where $\alpha \neq \tau$.

   Internal choice : $\dfrac{P \xrightarrow{\tau} P'}{P \,\Box\, Q \xrightarrow{\tau} P' \,\Box\, Q}$, $\dfrac{Q \xrightarrow{\tau} Q'}{P \,\Box\, Q \xrightarrow{\tau} P \,\Box\, Q'}$.

4. **Parallel composition**
   Synchronisation case : $\dfrac{P \xrightarrow{\alpha} P' \;,\; Q \xrightarrow{\alpha} Q'}{P \parallel_A Q \xrightarrow{\alpha} P' \parallel_A Q'}$, where $\alpha \in A$.

   Independent execution (modelled by interleaving):
   $\dfrac{P \xrightarrow{\alpha} P'}{P \parallel_A Q \xrightarrow{\alpha} P' \parallel_A Q}$, $\dfrac{Q \xrightarrow{\alpha} Q'}{P \parallel_A Q \xrightarrow{\alpha} P \parallel_A Q'}$, where $\alpha \notin A$.

5. **Hiding**
   $\dfrac{P \xrightarrow{\beta} P'}{P \setminus \beta \xrightarrow{\tau} P' \setminus \beta}$, $\dfrac{P \xrightarrow{\alpha} P'}{P \setminus \beta \xrightarrow{\alpha} P' \setminus \beta}$, where $\alpha \neq \beta$.

6. **Recursion** $\dfrac{P[fix\ x.P/x] \xrightarrow{\alpha} Q}{fix\ x.P \xrightarrow{\alpha} Q}$.

7. **Divergence** $div \xrightarrow{\tau} div$.

An interleaving model of a closed GTCSP term P is the transition system

$$A(\ P\ ) \ = \ (\ GTCSP,\ Act,\ \rightarrow,\ P\ ).$$

4.2 Definition :
For $P, Q \in GTCSP$ and $\omega \in Comm^*$, we define :
$P \stackrel{\omega}{\Rightarrow} Q$ iff there exists a sequence

$$P = P_1 \xrightarrow{\alpha_1} P_2 \xrightarrow{\alpha_2} \ldots \xrightarrow{\alpha_n} P_{n+1} = Q$$

where $n \geq 0$ and ω results from $\alpha_1 \ldots \alpha_n \in Act^*$ by skipping all occurrences of τ. We call Q a *derivative* of P.

Let P be a closed GTCSP term. Then, the transition system

$$O(\,P\,) \;=\; (\,GTCSP, Comm^{*}, \Rightarrow, P\,)$$

gives an operational semantics for P that specifies only the observable behaviour of the process P.

5 Labelled event structures

5.1 Definition :
$\varepsilon \;=\; (E, \leq, \#, l)$ is called a *(labelled) event structure* iff

1. E is a set (of *events*),
2. $\leq$ is a partial order on E,
3. $\#$ is an irreflexive, symmetric relation on E , called *conflict relation* , with : $\forall e_1, e_2, e_3 \in E :\; (e_1 \leq e_2 \text{ and } e_1 \# e_3\,) \implies e_2 \# e_3$,
4. $l : E \rightarrow Act$, where Act is the alphabet of *actions (labelling functions).*

5.2 Definition :
Let $\varepsilon = (E, \leq, \#, l)$ be an event structure, $E' \subset E$, $e \in E$.

1. $\#(e) \;:=\; \{\, e' \in E : e' \# e \,\}$.
2. $\#(E') \;:=\; \bigcup_{e \in E'} \#(e)$.
3. $\downarrow e \;:=\; \{\, e' \in E \;: e' \leq e \text{ and } e' \neq e \,\}$ is called the *preset of* e .

5.3 Definition :
Let $\varepsilon = (E, \leq, \#, l)$ be an event structure, $e \in E$.

$$depth(e) = \begin{cases} 1 & : \text{ if } \downarrow e = \emptyset \\ max\{depth(e') : e' \in \downarrow e\} \;+\; 1 & : \text{ if } \downarrow e \text{ is finite} \\ \infty & : \text{ otherwise} \end{cases}$$

5.4 Definition :
An event structure $\varepsilon = (E, \leq, \#, l)$ is called *(finitely) approximable* iff

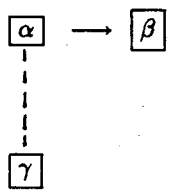
1. for each $e \in E$, $depth(e)$ is finite and
2. for each $n \in N$, $\{e \in E : depth(e) = n\}$ is finite.

Ev denotes the set of all finitely approximable event structures where we abstract from the names of the events, i.e. we will not distinguish isomorphic event structures. Two event structures $\varepsilon_i = (E_i, \leq_i, \#_i, l_i)$, $i = 1, 2$ are *isomorphic* if there exists a bijective mapping $f : E_1 \rightarrow E_2$ so that

1. $e_1 \leq_1 e_2 \iff f(e_1) \leq_2 f(e_2) \quad \forall e_1, e_2 \in E_1$,
2. $e_1 \#_1 e_2 \iff f(e_1) \#_2 f(e_2) \quad \forall e_1, e_2 \in E_1$ and
3. $l_2(f(e)) = l_1(e) \quad \forall e \in E_1$.

Event structures can be depicted graphically by representing events as boxes (inscribed with the event label) and connecting them with their direct predecessors and successors.
A conflict between two events is a *direct* conflict if no predecessors of the events are in conflict. Direct conflicts are depicted graphically by a broken line .

5.5 Example :
The event structure $\varepsilon = (E, \leq, \#, l)$ with $E = \{e_1, e_2, e_3\}$,
$e_1 \leq e_2$, $e_1 \# e_3$, $e_2 \# e_3$ and
$l(e_1) = \alpha$, $l(e_2) = \beta$, $l(e_3) = \gamma$ is shown as



The $\leq$ relation of an event structure models the causality of actions. Actions that are neither in a causal nor in a conflict relation may take place concurrently. On the other hand, one may derive from an event structure ε an interleaving behaviour by associating with ε a transition system as follows.

5.6 Definition :

1. Let $\mu \in Act$, ε, $\varepsilon' \in Ev$, $\varepsilon = (E, \leq, \#, l)$. The transition relation $\rightarrow \subseteq Ev \times Act \times Ev$ on event structures is defined by :

   $\varepsilon \xrightarrow{\mu} \varepsilon'$ iff there exists some event $e \in E$ with

   $$depth(e) = 1 \ , \ l(e) = \mu \ \text{ and } \ \varepsilon' = (E', \leq|_{E' \times E'}, \#|_{E' \times E'}, l|_{E'})$$
   $$\text{where } \ E' = E \setminus (\{e\} \cup \#(e)).$$

2. When we abstract from τ-events , we get the transition relation $\Rightarrow \subseteq Ev \times Comm^* \times Ev$:

   $\varepsilon \stackrel{\omega}{\Rightarrow} \varepsilon'$ iff there exists a sequence

   $$\varepsilon = \varepsilon_1 \stackrel{\mu_1}{\rightarrow} \varepsilon_2 \stackrel{\mu_2}{\rightarrow} \ldots \stackrel{\mu_n}{\rightarrow} \varepsilon_{n+1} = \varepsilon'$$

   where $n \geq 0, \mu_1, \ldots, \mu_n \in Act$ and $\omega \in Comm^*$ results from $\mu_1 \mu_2 \ldots \mu_n$ by removing all $\mu_i = \tau$.

3. The *(observable) interleaving semantics* of $\varepsilon \in Ev$ is defined as the transition system

   $$O(\varepsilon) = (Ev, Comm^*, \Rightarrow, \varepsilon).$$

5.7 Definition :
The event structures $\varepsilon_1, \varepsilon_2$ are called *τ-equivalent* , written $\varepsilon_1 \approx_\tau \varepsilon_2$, iff there exist event structures ε, Int_1, Int_2 , where all events in Int_1, Int_2 are labelled by τ , with $\varepsilon_i = \varepsilon \|_\emptyset Int_i$, $i = 1, 2$.

It is easy to see that τ-equivalence is an equivalence relation on Ev. [11] have shown that if $\varepsilon_1 \approx_\tau \varepsilon_2$ and $\varepsilon_1 \stackrel{\omega}{\Rightarrow} \varepsilon_1'$ then there exists $\varepsilon_2' \in Ev$ with $\varepsilon_1' \approx_\tau \varepsilon_2'$ and $\varepsilon_2 \stackrel{\omega}{\Rightarrow} \varepsilon_2'$.

5.8 Example :

If ε is $\boxed{\alpha} \longrightarrow \boxed{\tau}$ and Int_1 is $\boxed{\tau} \longrightarrow \boxed{\tau}$, $Int_2 = (\emptyset, \emptyset, \emptyset, \emptyset)$, then

$$\varepsilon \|_\emptyset Int_1 \approx_\tau \varepsilon \|_\emptyset Int_2 = \varepsilon \ .$$

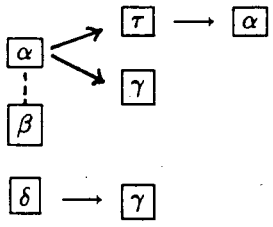
$\varepsilon \|_\emptyset Int_1$ is given by

$\boxed{\alpha} \longrightarrow \boxed{\tau}$

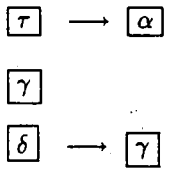
$\boxed{\tau} \longrightarrow \boxed{\tau}$

5.9 Example :

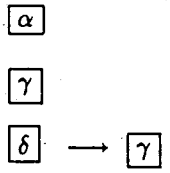
If ε is



then $\varepsilon \xrightarrow{\alpha} \varepsilon'$, where ε' is given by



and we have $\varepsilon \stackrel{\alpha}{\Rightarrow} \varepsilon''$, where ε'' is given by



6 Composition operations for event structures

The event structure semantics for GTCSP to be defined is compositional, which means that composition operators corresponding to the syntactical operators prefix, *or* , $\Box$, $\|_A$, $\backslash\beta$ and fix have to be defined. This section gives the operations for finite approximable event structures modelling the operations of GTCSP as defined in [11].

6.1 Definition :
Let $stop \in Ev$ to be defined as

$$stop := (\emptyset, \emptyset, \emptyset, \emptyset) \ .$$

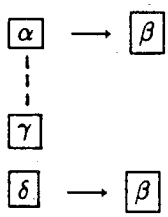
6.2 Definition :
Let $\varepsilon = (E, \leq, \#, l) \in Ev, \alpha \in Act, e_0 \notin E$. Then, the event structure $\alpha.\varepsilon$ will describe a process which first performs α and then behaves like ε.

$$\alpha.\varepsilon \;=\; (E', \leq', \#', l')$$

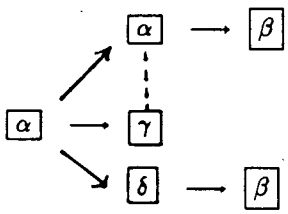
where

1. $E' \;=\; E \cup \{e_0\}$,
2. $e_1 \leq' e_2 \iff e_1 = e_0$ or $(e_1, e_2 \in E \;\&\; e_1 \leq e_2)$
3. $e_1 \#' e_2 \iff e_1, e_2 \in E \;\&\; e_1 \# e_2$
4. $l' : E' \to Act$ is defined by $l'(e) = l(e)$, if $e \in E$, and $l'(e_0) = \alpha$.

6.3 Example : Prefixing
$\alpha.\varepsilon$ describes a process that first performs α and then behaves like ε. If ε is



then $\alpha.\varepsilon$ is



6.4 Definition :
For $\varepsilon = (E, \leq, \#, l) \in Ev$, we define the set of *initial internal events* by

$$In(\varepsilon) \;:=\; \{e \in E : \forall e' \in E, e' \leq e : l(e') = \tau\} \;.$$

6.5 Definition :
Let $\varepsilon_i = (E_i, \leq_i, \#_i, l_i) \in Ev, i = 1,2$, w.l.o.g. $E_1 \cap E_2 = \emptyset$. The *conditional composition* of ε_1 and ε_2 is defined by

$$\varepsilon_1 \,\square\, \varepsilon_2 \;:=\; (E, \leq, \#, l)$$

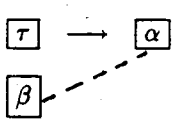
where

1. $E = E_1 \cup E_2$
2. $\leq \;=\; \leq_1 \cup \leq_2$
3. $e_1 \# e_2 \iff (e_1, e_2 \in E_1 \;\&\; e_1 \#_1 e_2)$ or $(e_1, e_2 \in E_2 \;\&\; e_1 \#_2 e_2)$ or
   $(e_1 \in E_1 \setminus In(\varepsilon_1) \;\&\; e_2 \in E_2 \setminus In(\varepsilon_2))$ or
   $(e_1 \in E_2 \setminus In(\varepsilon_2) \;\&\; e_2 \in E_1 \setminus In(\varepsilon_1))$
4. $l : E \to Act, l(e) = l_i(e)$ if $e \in E_i$, $i = 1,2$.

$\varepsilon_1 \,\square\, \varepsilon_2$ describes the process which behaves like one of the event structures ε_1 or ε_2 where the decision which alternative is left open as long as only internal actions are being performed.

6.6 Example : $\square$ - choice

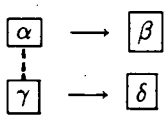
Let ε_1 be $\boxed{\tau} \longrightarrow \boxed{\alpha}$ and ε_2 be $\boxed{\beta}$. Then $\varepsilon_1 \,\square\, \varepsilon_2$ is given by



which describes that ε_1 may perform its τ-actions independently and that a decision has to take place as soon as communications are involved.

6.7 Example : $\square$ - choice

Let ε_1 be $\boxed{\alpha} \longrightarrow \boxed{\beta}$ and ε_2 be $\boxed{\gamma} \longrightarrow \boxed{\delta}$, then $\varepsilon_1 \,\square\, \varepsilon_2$ is



describing external choice.

6.8 Definition :
Let $\varepsilon_i = (E_i, \leq_i, \#_i, l_i) \in Ev$, $i = 1,2$, w.l.o.g. $E_1 \cap E_2 = \emptyset$.
The *nondeterministic combination* of ε_1 and ε_2 is defined by

$$\varepsilon_1 \; or \; \varepsilon_2 \; := \; (E, \leq, \#, l)$$

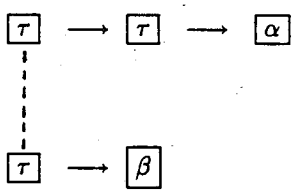
where

1. $E = E_1 \cup E_2 \cup \{f_1, f_2\}$, $f_1, f_2 \notin E_1 \cup E_2$
2. $e_1 \leq e_2 \iff (e_1, e_2 \in E_i \;\&\; e_1 \leq_i e_2,\; i = 1 \text{ or } i = 2)$ or
   $(e_i = f_i \;\&\; e_2 \in E_i,\; i = 1 \; or \; i = 2)$ or $e_1 = e_2$
3. $\#$ is the symmetric closure of $\#_1 \cup \#_2 \cup ((E_1 \cup \{f_1\}) \times (E_2 \cup \{f_2\}))$
4. $l : E \to Act$, $l(e) = l_i(e)$ if $e \in E_i$ and $l(f_i) = \tau, i = 1,2$.

The nondeterministic combination ε_1 *or* ε_2 behaves like ε_1 or like ε_2 where an internal decision chooses the alternative.

6.9 Example : *or*-choice

Let ε_1 be $\boxed{\tau} \longrightarrow \boxed{\alpha}$ and ε_2 be $\boxed{\beta}$.
Then ε_1 *or* ε_2 is given by



The internal character of the *or* -choice is modelled by prefixing the respective event structures with internal actions and by imposing a conflict between these internal actions .

6.10 Definition :
Let $\varepsilon_i = (E_i, \leq_i, \#_i, l_i) \in Ev$, $i = 1,2$, w.l.o.g. $E_1 \cap E_2 = \emptyset$ and $A \subseteq Comm$.

1. The *syntactical communication* of ε_1 and ε_2 on A is defined by

$$\begin{array}{rcl} Comm_A(\varepsilon_1,\varepsilon_2) & = & \{\,(e,\star) \,:\, e \in E_1,\ l_1(e) \notin A\,\} \\ & \cup & \{\,(\star,e) \,:\, e \in E_2,\ l_2(e) \notin A\,\} \\ & \cup & \{\,(e_1,e_2) \in E_1 \times E_2 \,:\, l_1(e_1) = l_2(e_2) \in A\,\} \end{array}$$

There $\star$ is an auxiliary symbol, $\star \notin E_1 \cup E_2$. We extend the relation $\leq_i$ and $\#_i$ on the argument $\star$ by defining

$$(\star \leq_i e) \ \vee \ (e \leq_i \star)) \iff e = \star$$

and

$$\neg(\ \star \#_i e\ )\ \ \forall\, e \in E_i \cup \{\star\}$$

2. Two communications $(e_1,e_2),(e_1',e_2') \in Comm_A(\varepsilon_1,\varepsilon_2)$ are *in conflict* iff they contain conflicting events, i.e. $e_1 \#_1 e_1'$ or $e_2 \#_2 e_2'$, or one event communicates with two distinct events, i.e. $(e_1 = e_1' \neq \star \wedge e_2 \neq e_2')$ or $(e_2 = e_2' \neq \star \wedge e_1 \neq e_1')$.

3. A subset C of $Comm_A(\varepsilon_1,\varepsilon_2)$ is *conflict-free* iff no two communications in C are in conflict.

4. Let $C \subseteq Comm_A(\varepsilon_1,\varepsilon_2)$ be conflict- free, $(e_1,e_2),(f_1,f_2) \in C$.

   (a) The relation $\prec$ is defined by

   $$(e_1,e_2) \prec (f_1,f_2) \iff ((e_1 \leq_1 f_1) \wedge \neg(e_2 >_2 f_2)) \vee ((e_2 \leq_2 f_2) \wedge \neg(e_1 >_1 f_1))$$

   We say (e_1,e_2) *precedes* (f_1,f_2) if $(e_1,e_2) \prec (f_1,f_2)$.

   (b) C is called *complete* iff
   $\forall (e_1,e_2) \in C, \forall f_1 \in E_1$ with $f_1 \leq_1 e_1$ there exists $(f_1,f_2) \in C$ with

   $$(f_1,f_2) \prec (e_1,e_2)$$

   and symmetrically
   $\forall (e_1,e_2) \in C, \forall f_2 \in E_2$ with $f_2 \leq_2 e_2$ there exists $(f_1,f_2) \in C$ with

   $$(f_1,f_2) \prec (e_1,e_2).$$

   (c) C is called *cycle-free* iff the transitive closure of $\prec$ is antisymmetric.

5. The *parallel composition* of ε_1 and ε_2 with communication on $A \subseteq Comm$ is given by

$$\varepsilon_1 \parallel_A \varepsilon_2 := (E, \leq, \#, l)$$

where

(a) $E = \{C_{(e_1,e_2)} : C_{(e_1,e_2)} \subseteq Comm_A(\varepsilon_1, \varepsilon_2)$ is conflict-free, cycle-free, complete and $(e_1, e_2) \in C_{(e_1,e_2)}$ is the only maximal element (with respect to $\prec$) $\}$.

(b) $\leq \; = \; \subseteq$

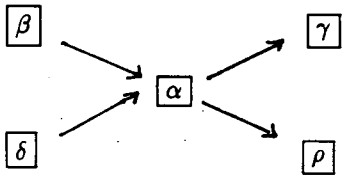
(c) $\# \; = \; \{ (C_1, C_2) \in E \times E : \exists (e_1, e_2) \in C_1, (f_1, f_2) \in C_2$ with $(e_1, e_2), (f_1, f_2)$ in conflict $\}$

(d) $l : E \to Act$, $l(C_{(e_1,e_2)}) = label(e_1, e_2)$
where $label(e_1, e_2) = l_1(e_1)$ if $e_1 \in E_1$ and
$label(e_1, e_2) = l_2(e_2)$ if $e_2 \in E_2$.

The parallel composition $\varepsilon_1 \parallel_A \varepsilon_2$ describes the independent execution of ε_1 and ε_2 where the actions of A may only be executed as joint actions by both processes together. In particular, $\parallel_\emptyset$ stand for fully independent execution (without synchronisation), and on the other extrem, $\parallel_{Comm}$ only allows actions which are performed in common.

6.11 Example : Parallel composition $\parallel_A$

Let ε_1 be $\boxed{\beta} \longrightarrow \boxed{\alpha} \longrightarrow \boxed{\gamma}$ and ε_2 be $\boxed{\delta} \longrightarrow \boxed{\alpha} \longrightarrow \boxed{\rho}$,
then $\varepsilon_1 \parallel_{\{\alpha\}} \varepsilon_2$ is given by



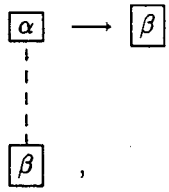
6.12 Definition :
Let $\varepsilon = (E, \leq, \#, l) \in Ev, \beta \in Comm$.

$$\varepsilon \setminus \beta := (E, \leq, \#, l')$$

where $l' : E \to Act$, $l'(e) = l(e)$ if $l(e) \neq \beta$ and $l'(e) = \tau$ otherwise.
The hiding operator transforms the actions labelled by β into internal actions, i.e. τ-events.

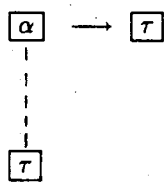
6.13 Example : Hiding

Let ε be



,

then $\varepsilon \setminus \beta$ is



7 The metric space of finite approximable event structures

In this section we will define a metric d on finite approximable event stuctures. [11] have shown that (Ev, d) is a complete ultametric space. Thus, every Banach-contractive mapping $\Phi : Ev \to Ev$ has a unique fixpoint in Ev.

7.1 Definition :
Let $\varepsilon, \varepsilon' \in Ev, n \in \mathbb{N}, \varepsilon = (E, \leq, \#, l)$.

1. The *truncation* of ε (of the depth n) is defined as follows :
$$\varepsilon^n := (E^n, \leq |_{E^n \times E^n}, \# |_{E^n \times E^n}, l|_{E^n})$$
where $E^n := \{e \in E : depth(e) \leq n\}$.

2. The distance between the event structures $\varepsilon, \varepsilon'$ is defined by
$$d(\varepsilon, \varepsilon') = 0 \quad :\Longleftrightarrow \quad \varepsilon = \varepsilon'$$
$$d(\varepsilon, \varepsilon') = \tfrac{1}{2^n} \quad :\Longleftrightarrow \quad \varepsilon \neq \varepsilon' \text{ and } n = max\{i : \varepsilon^i = \varepsilon'^i\}.$$

We recall that we deal with isomorphism class of event structures, i.e. we abstract of the names of the events $e \in E$. It is clear that the distance $d(\varepsilon, \varepsilon')$ is independent of chosen representatives.

7.2 Definition :
Let $Env := \{\sigma : \sigma : Idf \to Ev\}$ the set of *environments*. These are mappings which assign a meaning to free identifiers of a term.

For $\varepsilon_1, \ldots, \varepsilon_n \in Ev$, we define $\sigma[\varepsilon_1/x_1, \ldots, \varepsilon_n/x_n] : Idf \to Ev$ by

$$\begin{aligned} x_i &\mapsto \varepsilon_i , \quad i = 1, \ldots, n , \\ y &\mapsto \sigma(y) \quad \text{if } y \notin \{x_1, \ldots, x_n\}. \end{aligned}$$

Let $\Phi : GTCSP \times Env \times Idf \to (Ev \to Ev)$ be given by

$$\Phi(P, \sigma, x)(\varepsilon) := M[P]\sigma[\varepsilon/x] ,$$

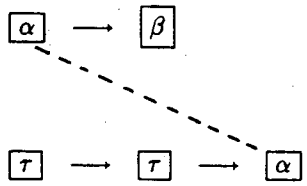
where M is the meaning function

$$M : GTCSP \times Env \to Ev$$

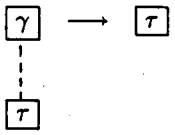
given by :
Let $\sigma \in Env$, $\alpha \in Act$, $\beta \in Comm$, $A \subseteq Comm$, $P, P_1, P_2 \in GTCSP$.

1. $M[stop]\sigma := (\emptyset, \emptyset, \emptyset, \emptyset) = stop$.
2. $M[div]\sigma := (I\!N, \leq, \emptyset, \tau)$.
3. $M[x]\sigma := \sigma(x)$ where $x \in Idf$.
4. $M[\alpha.P]\sigma := \alpha.M[P]\sigma$.
5. $M[P \setminus \beta]\sigma := M[P]\sigma \setminus \beta$.
6. $M[P_1 \Box P_2]\sigma := M[P_1]\sigma \Box M[P_2]\sigma$.
7. $M[P_1 or P_2]\sigma := M[P_1]\sigma \; or \; M[P_2]\sigma$.
8. $M[P_1 \|_A P_2]\sigma := M[P_1]\sigma \|_A M[P_2]\sigma$.
9. $M[fix\ x.P]\sigma := fix\ \Phi(P, \sigma, x)$
   where $fix\ \Phi(P, \sigma, x)$ denotes the unique fixpoint of the Banach - contractive mapping $\Phi(P, \sigma, x)$. See [11],where it has been shown that $\Phi(P, \sigma, x)$ is Banach - contractive .

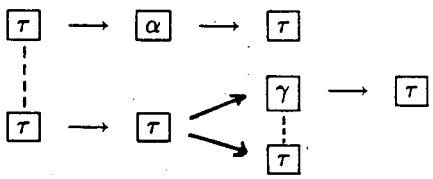
7.3 Example :
Let $P = \alpha.\beta.stop \Box \tau.\tau.\alpha.stop$, then $M[P]\sigma$ is independent of the environment σ :



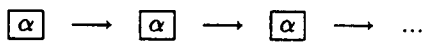
7.4 Example :
Let $P = \alpha.\tau.stop \; or \; \tau.x$ and $\sigma \in Env$ where $\sigma(x)$ is given by :



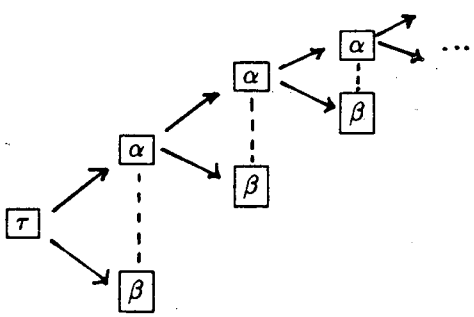
Then , $M[P]\sigma$ is given by :



7.5 Example :
Let $P = fix\ x.\ \alpha.x$, then $M[P]\sigma$ is



7.6 Example :
Let $P = \tau.fix\ x.\ (\alpha.x \Box \beta.stop)$, then $M[P]\sigma$ is given by :



8 The consistency of $O(P)$ and $M[P]$

In this section, we establish the result that the operational semantics $O(P)$ is consistent with the compositional event structure semantics $M[P]$ for every GTCSP process P. In particular, we show that $O(P)$ and the observable interleaving semantics $O(\ M[P]\ )$ are bisimular.

Lemma 1 :

Let $x \in Idf$ be guarded in $P \in GTCSP$.

1. $\sigma_1, \sigma_2 \in Env, \sigma_1(y) = \sigma_2(y) \forall y \in Idf \setminus \{x\}$
   $\Longrightarrow fix\ \Phi(P, \sigma_1, x) = fix\ \Phi(P, \sigma_2, x)$.
2. $M[P]\sigma$ depends only on $\sigma(x_1), \ldots, \sigma(x_n)$ where $x_1, \ldots, x_n$ are the identifiers which occur free in P. In particular, if P is closed, then $M[P]\sigma$ and $fix\ \Phi(P, \sigma, x)$ are independent of the environment σ.
3. Let $x_1, \ldots, x_n$ be pairwise distinct identifiers, $A_1, \ldots, A_n \in GTCSP$, then :

$$M[\ P[A_1/x_1, \ldots, A_n/x_n]\ ]\sigma \ = \ M[P]\sigma[\ M[A_1]\sigma/x_1, \ldots, M[A_n]\sigma/x_n\ ]\ .$$

Proof :

1. follows immediately from the definition of Φ.
2. is clear .
3. We define for $P \in GTCSP$: $P(\tilde{A}) := P[\ A_1/x_1, \ldots, A_n/x_n\ ]$ and
   for $\sigma \in Env$: $\sigma(\tilde{A}) := \sigma[\ M[A_1]\sigma/x_1, \ldots, M[A_n]\sigma/x_n\ ]$.

   By structural induction on the syntax of P we show that

$$M[P]\sigma(\tilde{A}) \ = \ M[P(\tilde{A})]\sigma \ .$$

   Basis of induction :
   1. $P = stop$ or $P = div$: $P(\tilde{A}) = P$ and $M[P]\sigma_1 = M[P]\sigma_2 \ \forall\ \sigma_1, \sigma_2 \in Env$.
   2. $P = x \in Idf$:

$$M[x]\sigma(\tilde{A}) = \begin{cases} M[A_i]\sigma = M[P(\tilde{A})]\sigma & : \text{if } x = x_i \\ \sigma(x) = M[x]\sigma = M[P(\tilde{A})]\sigma & : \text{if } x \in Idf \setminus \{x_1, \ldots, x_n\}. \end{cases}$$

   Induction step :
   1. $P = P_1\ op\ P_2$ where $op \in \{\Box, or, \|_A\}$:

$$\begin{aligned} M[P]\sigma(\tilde{A}) \ &= \ (M[P_1]\sigma(\tilde{A}))\ op\ (M[P_2]\sigma(\tilde{A})) \\ &= \ M[P_1(\tilde{A})]\sigma\ \ op\ \ M[P_2(\tilde{A})]\sigma \\ &= \ M[\ P_1(\tilde{A})\ op\ P_2(\tilde{A})\ ]\sigma \ = \ M[P(\tilde{A})]\sigma. \end{aligned}$$

   2. $P = op(P')$ where $op(P') = \alpha.P'$ or $op(P') = P'/\beta$:

$$\begin{aligned} M[P]\sigma(\tilde{A}) \ &= \ op(\ M[P']\sigma(\tilde{A})\ ) \ = \ op(\ M[P'(\tilde{A})]\sigma\ ) \\ &= M[\ op(P'(\tilde{A}))\ ]\sigma \ = \ M[P(\tilde{A})]\sigma. \end{aligned}$$

3. $P = fix\ x.P'$:

If the identifier x occurs free in $A_1, \ldots, A_n$, then we can deal with $R = fix\ z.P'[z/x]$ where $z \in Idf \setminus \{x_1, \ldots, x_n\}$ is an identifier which does not occur free in $A_1, \ldots, A_n$ and P'. Then,

$$M[P(\tilde{A})]\sigma = M[R(\tilde{A})]\sigma$$

and for each $\sigma \in Ev$:

$$M[P]\sigma = M[R]\sigma.$$

Therefore, we can assume w.o.l.g. that $R = P$, i.e. that x does not occur free in $A_1, \ldots, A_n$.

Case 1 : $x \notin \{x_1, \ldots, x_n\}$

$$\Longrightarrow \quad P(\tilde{A}) = fix\ x.P'(\tilde{A})$$

Then, for all $\sigma \in Env$:

$M[P'(\tilde{A})]\sigma = M[P']\sigma(\tilde{A})$ (by induction hypothesis)
$\Longrightarrow \Phi(P'(\tilde{A}), \sigma, x) = \Phi(P', \sigma(\tilde{A}), x)$
$\Longrightarrow M[P(\tilde{A})]\sigma = fix\ \Phi(P'(\tilde{A}), \sigma, x) = fix\ \Phi(P', \sigma(\tilde{A}), x) = M[P]\sigma(\tilde{A})$.

Case 2 : $x = x_i$ for an index $i \in \{1, \ldots, n\}$. Then, x does not occur free in P. W.o.l.g. $x = x_1$.
Then, we have : $P(\tilde{A}) = P[A_2/x_2, \ldots, A_n/x_n]$.
Since $x = x_1$ does not occur free in $P(\tilde{A})$, we get by case 1 :

$$\begin{aligned} M[P(\tilde{A})]\sigma &= M[\ P[A_2/x_2, \ldots, A_n/x_n]\ ]\sigma \\ &= M[\ P[A_2/x_2, \ldots, A_n/x_n]\ ]\sigma[M[A_1]\sigma/x_1] \\ &= M[P]\sigma[M[A_1]\sigma/x_1][M[A_2]\sigma/x_2,\ldots,M[A_n]\sigma/x_n] \qquad \text{(by case 1)} \\ &= M[P]\sigma(\tilde{A}). \end{aligned}$$

Lemma 2 :

Let $P, B, A_1, \ldots, A_n \in GTCSP$ and let $x_1, \ldots, x_n, y \in Idf$ be pairwise distinct identifiers, so that y does not occur free in $A_1, \ldots, A_n$. Then,

$$P[A_1/x_1, \ldots, A_n/x_n, B[\tilde{A}/\tilde{x}]/y] = P[B/y][\tilde{A}/\tilde{x}].$$

Proof :
We proof the equality by induction on the syntax of P .
We use the following notation :

$$\overline{Q} := Q[\, A_1/x_1, \ldots, A_n/x_n, B[\tilde{A}/\tilde{x}]/y \,]$$

and

$$\widetilde{Q} := Q[B/y][\tilde{A}/\tilde{x}]$$

where $Q \in GTCSP$.

Basis of induction :
1. $P = stop$ or $P = div$: Then, $\overline{P} = \widetilde{P} = P$, since P is closed.

2. $P = z \in Idf$:

Case 1 : $z = x_i$ for an index $i \in \{1, \ldots, n\}$.
$\Rightarrow \overline{P} = A_i$ and $\widetilde{P} = P[B/y][\tilde{A}/\tilde{x}] = P[\tilde{A}/\tilde{x}] = A_i$.

Case 2 : $z = y$, then , $\overline{P} = B[\tilde{A}/\tilde{x}]$ and $\widetilde{P} = P[B/y][\tilde{A}/\tilde{x}] = B[\tilde{A}/\tilde{x}]$.

Case 3 : $z \notin \{y, x_1, \ldots, x_n\}$, then $\overline{P} = \widetilde{P} = P = z$.

Induction step :
1. $P = P_1\ op\ P_2$ where $op \in \{\ \Box, or, \|_A\ \}$.
$\Longrightarrow \overline{P} = \overline{P_1}\ op\ \overline{P_2} = \widetilde{P_1}\ op\ \widetilde{P_2} = \widetilde{P}$.

2. $P = op(P_1)$ where $op(P') = \alpha.P'$ or $op(P') = P' \setminus \beta$.
$\Longrightarrow \overline{P} = op(\ \overline{P_1}\ ) = op(\ \widetilde{P_1}\ ) = \widetilde{P_1}$.
3. $P = fix\ z.P'$:
Case 1 : z does not occur free in $A_1, \ldots, A_n, B$:

$$\Longrightarrow \overline{P} = fix\ z.\overline{P'} = fix\ z.\widetilde{P'} = \widetilde{P}\ .$$

Case 2 : z occurs free in one of the terms $A_1, \ldots, A_n$ or B:
Let $w \in Idf \setminus \{y, x_1, \ldots, x_n\}$ be an identifier which does not occur free in $A_1, \ldots, A_n$ and B. We define

$$R := fix\ w.P'[w/z].$$

By case 1, it follows:

$$\overline{R} = \widetilde{R}.$$

P arises from R by substituting each free occurence of w in $P'[w/z]$ by z. Therefore, $\overline{P}$ arise from $\overline{R}$ by substituting each free occurence of w in $P'[\hat{w}/z]$ by z . Analogous, $\widetilde{P}$ arises from $\widetilde{R}$ by substituting each free occurence of w in $P'[\widetilde{w}/z]$ by z .

$$\Longrightarrow \quad \overline{P} = \widetilde{P}.$$

Lemma 3 :

Let $P \in GTCSP$. Then, for all $x_1, \ldots, x_n \in Idf$ pairwise distinct identifiers, which are guarded in P , and for all $A_1, \ldots, A_n \in GTCSP$:
If $P[A_1/x_1, \ldots, A_n/x_n] \xrightarrow{\alpha} Q$, then there exists $P' \in GTCSP$ with

1. $P \xrightarrow{\alpha} P'$ and
2. $P'[\, A_1/x_1, \ldots, A_n/x_n \,] \;=\; Q$.

Proof :
Basis of induction :
1.$P = stop$: $P[\tilde{A}/\tilde{x}] = stop$ has no derivative.

2.$P = div$: $P[\tilde{A}/\tilde{x}] = div \xrightarrow{\alpha} Q$, then $\alpha = \tau$, $Q = div$.
With $P' := div$ follows :

$$P \xrightarrow{\alpha} P' \;,\; P'[\tilde{A}/\tilde{x}] \;=\; Q \;.$$

3. $P = z \in Idf$:
Since x_i is guarded in P , $i = 1, \ldots, n$, we have : $z \notin \{x_1, \ldots, x_n\}$.

$$\Rightarrow \quad P[\tilde{A}/\tilde{x}] = z \text{ has no derivative.}$$

Induction step :
Let $P[\tilde{A}/\tilde{x}] \xrightarrow{\alpha} Q$. We consider only the following three cases :

1. $P = \beta.P_1$, then $P[\tilde{A}/\tilde{x}] = \beta.P_1[\tilde{A}/\tilde{x}]$

$$\Rightarrow \; P_1[\tilde{A}/\tilde{x}] = Q \text{ and } \alpha = \beta \;,\; P \xrightarrow{\alpha} P_1.$$

2. $P = P_1$ *or* P_2, then $P[\tilde{A}/\tilde{x}] = P_1[\tilde{A}/\tilde{x}]$ *or* $P_2[\tilde{A}/\tilde{x}]$

$$\Rightarrow \; \alpha = \tau \text{ and } (\; Q = P_1[\tilde{A}/\tilde{x}] \text{ or } Q = P_2[\tilde{A}/\tilde{x}] \;),$$

w.o.l.g. $Q = P_1[\tilde{A}/\tilde{x}]$. Then $P \xrightarrow{\alpha} P_1$.

3. $P = fix\ x.P_1$.

Case 1 : $x \notin \{x_1, \ldots, x_n\}$ and x does not occur free in $A_1, \ldots, A_n$. Then,

$$P[\tilde{A}/\tilde{x}] = fix\ x.P_1[\tilde{A}/\tilde{x}].$$

We define : $x_{n+1} := x$ and $A_{n+1} := fix\ x.P_1[\tilde{A}/\tilde{x}]$.

Then, $x_1, \ldots, x_{n+1}$ are pairwise distinct identifiers which are guarded in P_1. Since x does not occur free in $A_1, \ldots, A_n$:

$$\begin{aligned} & P_1[\ A_1/x_1, \ldots, A_n/x_n, A_{n+1}/x_{n+1}\ ] \\ &= P_1[\ A_1/x_1, \ldots, A_n/x_n\ ][\ fix\ x.P_1[\tilde{A}/\tilde{x}]/x\ ] \\ &= P_1[\tilde{A}/\tilde{x}][\ fix\ x.P_1[\tilde{A}/\tilde{x}]/x\ ]\ . \end{aligned}$$

Since $P[\tilde{A}/\tilde{x}] \xrightarrow{\alpha} Q$:

$$P_1[\tilde{A}/\tilde{x}][\ fix\ x.P_1[\tilde{A}/\tilde{x}]/x\ ] \xrightarrow{\alpha} Q,$$

and since x does not occur free in $A_1, \ldots, A_n$:

$$P_1[\ A_1/x_1, \ldots, A_n/x_n, A_{n+1}/x_{n+1}\ ] \xrightarrow{\alpha} Q.$$

By induction hypothesis , there exists $P_1' \in GTCSP$ with

1. $P_1 \xrightarrow{\alpha} P_1'$ and
2. $P_1'[\ A_1/x_1, \ldots, A_n/x_n, A_{n+1}/x_{n+1}\ ] = Q$.

By Lemma 2 :

$$\begin{aligned} & P_1'[\ fix\ x.P_1/x\ ][\tilde{A}/\tilde{x}] \\ &= P_1'[\ A_1/x_1, \ldots, A_n/x_n,\ (fix\ x.P_1)[\tilde{A}/\tilde{x}]/x\ ] \\ &= P_1'[\ A_1/x_1, \ldots, A_n/x_n,\ fix\ x.P_1[\tilde{A}/\tilde{x}]\ /x\ ] \\ &= P_1'[\ A_1/x_1, \ldots, A_n/x_n, A_{n+1}/x_{n+1}\ ] \\ &= Q\ . \end{aligned}$$

Since $P_1 \xrightarrow{\alpha} P_1'$, we have :

$$P_1[\ fix\ x.P_1/x\ ] \xrightarrow{\alpha} P_1'[\ fix\ x.P_1/x\ ],$$

and so

$$P = fix\ x.P_1 \xrightarrow{\alpha} P_1'[\ fix\ x.P_1/x\ ].$$

With $P' := P_1'[\ fix\ x.P_1/x\ ]$, we have :

$$P \xrightarrow{\alpha} P' \quad \text{and} \quad P'[\tilde{A}/\tilde{x}] = Q.$$

Case 2 : $x \notin \{x_1, \ldots, x_n\}$, but there is a free occurence of x in one of the terms A_i.

Let $z \in Idf \setminus \{x, x_1, \ldots, x_n\}$ an identifier which does not occur free in $P, A_1, \ldots, A_n$.
We define $B_i := A_i[z/x]$, $i = 1, \ldots, n$. Then,

$$P[\tilde{A}/\tilde{x}] \;=\; (fix\ x.P_1[\tilde{B}/\tilde{x}])[x/z]\ .$$

Since $P[\tilde{A}/\tilde{x}] \xrightarrow{\alpha} Q$:

$$fix\ x.P_1[\tilde{B}/\tilde{x}] = P[\tilde{A}/\tilde{x}][z/x] \xrightarrow{\alpha} Q[z/x].$$

Since x does not occur free in $B_1, \ldots, B_n$, we have :

$$fix\ x.P_1[\tilde{B}/\tilde{x}] \;=\; (fix\ x.P_1)[\tilde{B}/\tilde{x}] \;=\; P[\tilde{B}/\tilde{x}]\ .$$

It follows from case 1 that there exists $P' \in GTCSP$ with
1. $P = fix\ x.P_1 \xrightarrow{\alpha} P'$ and
2. $P'[\tilde{B}/\tilde{x}] \;=\; Q[z/x]$.

Since P and $A_1, \ldots, A_n$ not contain a free occurence of z, the terms P' and Q also not contain any free occurence of z . It follows :

$$P'[\tilde{A}/\tilde{x}] \;=\; P'[\tilde{B}/\tilde{x}][x/z] \;=\; Q[z/x][x/z] \;=\; Q\ .$$

Case 3 : $x = x_i$ for an index $i \in \{1, \ldots, n\}$, w.l.o.g. $x = x_1$. Then,

$$P[\tilde{A}/\tilde{x}] \;=\; P[\ A_2/x_2, \ldots, A_n/x_n\ ]$$

and $x \notin \{x_2, \ldots, x_n\}$.
By cases 1 and 2, it follows that there exists a term $P' \in GTCSP$ with
1. $P \xrightarrow{\alpha} P'$ and
2. $P'[\ A_2/x_2, \ldots, A_n/x_n\ ] \;=\; Q$.

Since the identifier $x = x_1$ does not have free occurences in P , the term P' does also not contain free occurences of $x = x_1$.
It follows :

$$P'[\tilde{A}/\tilde{x}] \;=\; P'[\ A_2/x_2, \ldots, A_n/x_n\ ] \;=\; Q\ .$$

8.1 Remark :
Let $P, Q, A_1, \ldots, A_n \in GTCSP$ and $x_1, \ldots, x_n \in Idf$ be pairwise distinct identifiers which are guarded in P so that $P[\tilde{A}/\tilde{x}] \xrightarrow{\alpha} Q$.

Then, by lemma 3, there exists $P' \in GTCSP$ with
1. $P \xrightarrow{\alpha} P'$ and
2. $P'[\tilde{A}/\tilde{x}] = Q$.

It is easy to see that for all terms $B_1, \ldots, B_n \in GTCSP$:

$$P[\tilde{B}/\tilde{x}] \xrightarrow{\alpha} P'[\tilde{B}/\tilde{x}].$$

8.2 Remark :
If $A \in GTCSP$ is closed then

$$M[A]\sigma_1 = M[A]\sigma_2 \quad \forall\, \sigma_1, \sigma_2 \in Env.$$

So, we can define

$$M[A] := M[A]\sigma \quad \text{where } \sigma \in Env.$$

Lemma 4 :

Let $P \in GTCSP$, $\alpha \in Act$, $\sigma \in Env$.

1. If $P \xrightarrow{\alpha} P'$ then $M[P]\sigma \stackrel{\hat{\alpha}}{\Rightarrow} M[P']\sigma$, where $\hat{\alpha} = \alpha$ if $\alpha \neq \tau$ and $\hat{\alpha}$ denotes the empty word in $Comm^*$ if $\alpha = \tau$.

2. If $x_1, \ldots, x_n$ be the pairwise distinct identifiers that are guarded in P and occur free in P and if $\sigma(x_i) = M[A_i]$ where A_i is a closed GTCSP term, $i = 1, \ldots, n$, then, for all event structures $\varepsilon' \in Ev$ with $M[P]\sigma \xrightarrow{\alpha} \varepsilon'$, there exists a closed term $P' \in GTCSP$ with
   1. $P[\, A_1/x_1, \ldots, A_n/x_n \,] \stackrel{\hat{\alpha}}{\Rightarrow} P'$ and
   2. $M[P'] = M[P']\sigma \approx_\tau \varepsilon'$.

Proof :
We will prove the statements by induction on the structure of P .

1. We assume that $P \xrightarrow{\alpha} P'$.
   Basis of induction :
   1. $P = stop$ has no derivatives.
   2. $P = div$, then $\alpha = \tau$ and $P' = div$, $M[P]\sigma = M[P']\sigma$.
   3. $P = z \in Idf$, then P has no derivatives.

Induction step :
The most interesting operator is the fix- operator.
$P = fix\ x.Q$, then $Q[fix\ x.Q/x] \xrightarrow{\alpha} P'$.

By Lemma 3, there exists $Q' \in GTCSP$ with $Q \xrightarrow{\alpha} Q'$ and $Q'[fix\ x.Q/x] = P'$.
By induction hypothesis :

$$M[Q]\sigma[\ M[P]\sigma/x\ ] \stackrel{\hat{\alpha}}{\Rightarrow} M[Q']\sigma[\ M[P]\sigma/x\ ]$$

On the other side, we have :

$$M[P]\sigma = fix\Phi(Q,\sigma,x) = M[Q]\sigma[\ M[P]\sigma/x\ ]$$

and

$$M[P']\sigma = M[Q'[fix\ x.Q/x]]\sigma = M[\ Q'[P/x]\ ]\sigma = M[Q']\sigma[\ M[P]\sigma/x\ ]$$

(Lemma 1,3.).

Then, $M[P]\sigma \stackrel{\hat{\alpha}}{\Rightarrow} M[P']\sigma$.

2. Induction step :
Again, we only consider the fix - operator : $P = fix\ x.Q$.

The identifiers occuring free in Q are $x_1, \ldots, x_n$ and x . We get :

$$M[P]\sigma = fix(Q,\sigma,x) = \Phi(Q,\sigma,x)(\ M[P]\sigma\ ) = M[Q]\sigma[\ M[P]\sigma/x\ ]$$

and

$$M[P]\sigma = M[\ P[A_1/x_1, \ldots, A_n/x_n]\ ]$$

(by Lemma 1,3.).

Hence

$\sigma[\ M[P]\sigma/x\ ](x) = M[P]\sigma = M[\ P[\tilde{A}/\tilde{x}]\ ]$ and
$\sigma[\ M[P]\sigma/x\ ](x_i) = \sigma(x_i) = M[A_i]$, $i = 1, \ldots, n$.

Since $M[P]\sigma \xrightarrow{\alpha} \varepsilon'$,

$$M[Q]\sigma[\ M[P]\sigma/x\ ] \xrightarrow{\alpha} \varepsilon'.$$

By induction hypothesis, there exists $P' \in GTCSP$ with

$$Q[\ A_1/x_1, \ldots, A_n/x_n, P[\tilde{A}/\tilde{x}]/x\ ] \stackrel{\hat{q}}{\Rightarrow} P' \quad \text{and}$$
$$M[P']\sigma[\ M[P]\sigma/x\ ] \approx_\tau \varepsilon'.$$

Since P' is closed, we have : $M[P'] = M[P']\sigma[\ M[P]\sigma/x\ ] = M[P']\sigma$.

Since the terms $A_1, \ldots, A_n$ are closed, we get :

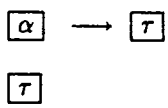
$$Q[\tilde{A}/\tilde{x}][\ P[\tilde{A}/\tilde{x}]/x\ ] = Q[\ A_1/x_1, \ldots, A_n/x_n, P[\tilde{A}/\tilde{x}]/x\ ].$$

Then : $Q[\tilde{A}/\tilde{x}][\ fix\ x.Q[\tilde{A}/\tilde{x}]/x\ ] \stackrel{\hat{q}}{\Rightarrow} P'$

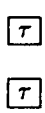
We get : $P[\tilde{A}/\tilde{x}] = fix\ x.Q[\tilde{A}/\tilde{x}] \stackrel{\hat{q}}{\Rightarrow} P'$.

8.3 Remark :
In Lemma 4,2., we have to deal with τ-equivalence :
When P is a closed GTCSP - term and $M[P] \stackrel{\alpha}{\rightarrow} \varepsilon$, so we cannot conclude that there exists $P' \in GTCSP$ with $M[P'] = \varepsilon$ and $P \stackrel{\alpha}{\rightarrow} P'$.

8.4 Example :
Let $P = (\alpha.\beta.stop \,\square\, \tau.stop) \setminus \beta$, then $M[P]$ is



and we have : $M[P] \stackrel{\alpha}{\rightarrow} \varepsilon$, where ε is



On the other side, P has only one α-derivative :

$P \stackrel{\alpha}{\rightarrow} (\beta.stop) \setminus \beta$ and $M[\ (\beta.stop) \setminus \beta\ ]$ is given by .

Corollary :
Let $R := \{ (P,\varepsilon) : P \in GTCSP, P \text{ closed}, \varepsilon \in Ev,\ \varepsilon \approx_\tau M[P] \}$.
Then , R is a bisimulation between the transition systems $O(P)$ and $O(\ M[P]\ )$.

Proof :
Let $(P,\varepsilon) \in R$.
It is easy to see that it is enough to show that :
1. If $P \xrightarrow{\alpha} P'$, then there exists $\varepsilon' \in Ev$ with
$M[P] \stackrel{\hat{\alpha}}{\Rightarrow} \varepsilon'$ and $(P',\varepsilon') \in R$ and
2. If $M[P] \xrightarrow{\alpha} \varepsilon'$, then there exists $P' \in GTCSP$ with
$P \stackrel{\alpha}{\Rightarrow} P'$ and $(P',\varepsilon') \in R$.

1. When $P \xrightarrow{\alpha} P'$, so we have by Lemma 4,1. : $M[P] \stackrel{\hat{\alpha}}{\Rightarrow} M[P']$.

Since $\varepsilon \approx_\tau M[P]$, there exists $\varepsilon' \in Ev$ with $\varepsilon \stackrel{\alpha}{\Rightarrow} \varepsilon'$ and $\varepsilon' \approx_\tau M[P']$.
Then $(P',\varepsilon') \in R$.

2. When $\varepsilon \xrightarrow{\alpha} \varepsilon'$, then there exists $\varepsilon'' \in Ev$ with

$$M[P] \stackrel{\hat{\alpha}}{\Rightarrow} \varepsilon'' \text{ and } \varepsilon' \approx_\tau \varepsilon''.$$

By Lemma 4,2., it is easy to show that there exists $P' \in GTCSP$, P' closed, with

$$P \stackrel{\hat{\alpha}}{\Rightarrow} P' \ ,\ M[P'] \approx_\tau \varepsilon'' \ .$$

Then, $M[P'] \approx_\tau \varepsilon'$ and $(P',\varepsilon') \in R$.

Theorem :

For every closed $P \in GTCSP$(i.e. every guarded process), the transition systems $O(P)$ and $O(M[P])$ are bisimular.

9 Conclusion

We have shown that an interleaving specification of a GTCSP process P and a noninterleaving meaning of P are 'bisimular'. One difficulty in establishing such a result, in particular when including recursion via the fix-operator, is, that a compositional semantics that provides semantic operators for the syntactical constructs, is compared with an operational semantics using a transition system. Hence, in order to establish a relation between the two meanings of a process P

we may not simply perform an induction on the structure of P . In particular, in the case of recursion, we have no operator that determines the 'meaning' of $fix\ x.Q$ from the 'meaning' of Q in the transition system case.
Our proof works by obtaining information on the behaviour of a process P from the knowledge of the behaviour of $P[\tilde{A}/\tilde{x}]$, see lemma 3 and lemma 4 .
The obtained theorem may be interpreted as a consistency result.
Consistency problems concerning noninterleaving and interleaving models are also discussed in [9,18,21]. These investigations differ from the present work in particular in the noninterleaving model (petri nets, prime event structures) and/or in the language studied and in the proof method.

10 References

1. J.W. de Bakker, J.I.Zucker :
   Processes and the Denotational Semantics of Concurrency,
   Information and Control, Vol.54, No 1/2, pp 70-120, 1982 .

2. J.A. Bergstra, J.W.Klop :
   Process Algebra for Synchronous Communication,
   Information and Control, Vol 60 , No 1-3, pp 109 - 137, 1984 .

3. G. Boudol, I.Castellani :
   On the Semantics of Concurrency : Partial Orders and Transition Systems,
   Proc. TAPSOFT 87, Vol 1, Lecture Notes in Computer Science 249, Springer - Verlag, pp 123 - 137, 1987 .

4. G. Boudol, I.Castellani :
   Permutation of transitions : An event structure semantics for CCS and SCCS,
   Proc. School/Workshop on Linear Time, Branching Time and Partial Order in Logics and Models for Concurrency ,
   Lecture Notes in Computer Science 354 , Springer - Verlag, pp 411-427, 1989 .

5. S.D. Brookes :
   A Model for Communicating Sequential Processes,
   report CMU-CS 83-149, Carnegie-Mellon University, January 1983 .

6. S.D. Brookes, C.A.R. Hoare, A.W. Roscoe :
   A Theory of Communicating Sequential Processes,
   Journal ACM, Vol. 31, No. 3, July 1984 .

7. S.D. Brookes, A.W. Roscoe :
   An improved Failure Model for Communicating Processes,
   Seminar on Concurrency, Lecture Notes in Computer Science 197, Springer - Verlag, 1985 .

8. P.Degano, R.De Nicola, U. Montanari :
A Distributed Operational Semantics for CCS Based on Condition/Event Systems,
Acta Informatica 26 , pp 59 - 91 , 1988 .

9. P.Degano, R.De Nicola, U. Montanari :
On the Consistency of 'Truly Concurrent' Operational and Denotational Semantics,
Proc. Symposium on Logic in Computer Science, Edinburgh, pp 133 - 141, 1988 .

10. U. Goltz :
On Representing CCS Programs as Finite Petri Nets,
Proc.MFCS 88, Lecture Notes in Computer Science 324, Springer-Verlag, pp 339 - 350, 1988.

11. U. Goltz, R. Loogan :
Modelling Nondeterministic Concurrent Processes with Event Structures,
Fundamentae Informaticae, Vol.14, No.1, pp 39 - 73 , 1991.

12. U. Goltz, A. Mycroft :
On the Relationship of CCS and Petri Nets,
Proc. ICALP 84, Lecture Notes in Computer Science 172, Springer - Verlag , 1984 .

13. C.A.R. Hoare :
Communication Sequential Processes,
Prentice Hall, 1985 .

14. R. Milner :
A Calculus of Communication Systems,
Lecture Notes in Computer Science 92, Springer - Verlag, 1980 .

15. R. Milner :
Lectures on a Calculus of Communicating Systems,
Seminar on Concurrency, Lecture Notes in Computer Science 197, Springer - Verlag, 1985 .

16. M. Nielsen, G. Plotkin , G. Winskel :
Petri - Nets, Event Structures and Domains ,
Theoretical Computer Science, Vol. 13, No. 1, pp 85 - 108, 1981 .

17. E.R. Olderog :
TCSP : Theory of Communicating Sequential Processes ,
Advances in Petri - Nets 1986 ,
Lecture Notes in Computer Science 255 , Springer - Verlag , pp 441 - 465, 1987 .

18. E.R. Olderog :
Operational Petri - Net Semantics for CCSP,
Advances in Petri - Nets 1987 ,
Lecture Notes in Computer Science 266, Springer - Verlag, pp 196 - 223 , 1987 .

19. E.R. Olderog, C.A.R. Hoare :
Specification - oriented semantics for communicating processes ,
Acta Informatica 23 , pp 9 - 66 , 1986 .

20. G.D. Plotkin :
An Operational Semantics for CSP ,
Formal Description of Programming Concepts II, North Holland, pp 199 - 225, 1983 .

21. W. Reisig :
Partial Order Semantics versus Interleaving Semantics for CSP - like languages and its Impact on Fairness,
Proc. ICALP 84, Lecture Notes in Computer Science 172, Springer - Verlag, pp 403 - 413, 1984 .

22. D. Taubner, W. Vogler :
The Step Failure Semantics,
Proc. STACS 87, Lecture Notes in Computer Science 247, Springer - Verlag, pp 348 - 359, 1987 .

23. G. Winskel :
Events in Computation ,
Ph.D.Thesis, University of Edinburgh, report CST-10-80, December 1980 .

24. G. Winskel :
Event Structure Semantics for CCS and Related Languages,
Proc. ICALP 82, Lecture Notes in Computer Science 140, Springer - Verlag, pp 561 - 576, 1982 . Theoretical Computer Science, May 1985 .

25. G. Winskel :
Event Structures,
Petri - Nets : Applications and Relationships to Other Models of Concurrency,
Lecture Notes in Computer Science 255, Springer - Verlag, pp 325 - 392, 1987 .