

# Covering Step Graph preserving Failure Semantics

François Vernadat, François Michel  
e-mail : {vernadat,fmichel}@laas.fr

LAAS-CNRS

7 avenue du Colonel Roche F-31077 Toulouse cedex - France

**Abstract:** Within the framework of concurrent systems, several verification approaches require as a preliminary step the complete derivation of the state space. Partial-order methods are efficient for reducing the state explosion due to the modeling of parallelism by interleaving.

In the case of persistent or sleep sets, only a subset of enable transitions is examined, the derived graph is then a subgraph of the whole graph. The resulting sub-graph may be used for verifying absence of deadlock or more specific properties.

The covering step graph (CSG) approach visits all the transitions, but some independent events are put together to build a single transition step, the firing of this transition step is then atomic.

In a CSG, steps of independent transitions are substituted as much as possible to the subgraph which would result from the firing of the independent transitions. The potential benefit of such a substitution may be exponential with respect to the number of “merged” independent transitions.

This paper investigates the on-the-fly derivation of covering step graphs preserving failure semantics. Testing Equivalence and CSP semantics are considered.

**Keywords:** concurrent systems, state space exploration, partial-order, failure semantics, verification methods.

## 1 Introduction

The state space derivation represents the preliminary step of several verification methods for concurrent systems. This approach is made attractive by the existence of efficient and automatic verification techniques, such as bisimulation and model-checking. The combinational explosion is the main limitation of this approach.

Many studies are currently in progress for reducing this problem: on-the-fly bisimulation [FM 90], symbolic marking graph derivation through symmetrical folding [Jen 87], so-called partial order techniques which attempt to avoid the combinational explosion resulting from the concurrency interpretation by means of interleaving: persistent sets [Val 89], sleep sets [GW 91]. An other direction, followed in [Esp 93, McMil 95], avoids the state explosion by using model checker directly on the system description (unfolding of Petri Nets).

The partial order techniques (see [WG 93] for a general survey) represent the framework of the approach developed in this paper. The basic principle of these approaches consists in considering a single specific path among all the sequences

which possess the same Mazurkiewicz's trace [Maz 87]. The aim is to obtain a sub-graph of the initial one where transition interleaving will be as reduced as possible. The set of firable transitions from each reached state is limited by means of "sleep set" or "persistent set" (both methods can be combined). Figure 1 depicts the results of these methods upon the graph describing the interleaving of  $ab$  and  $cd$ .

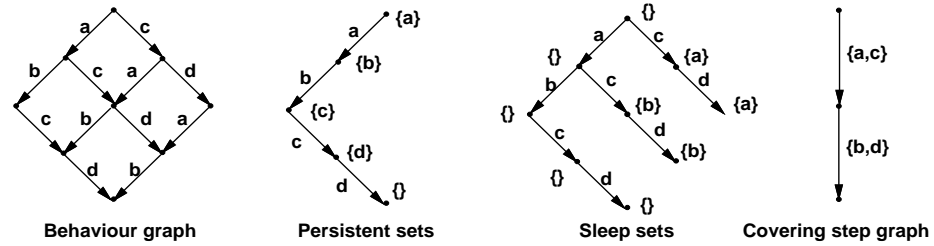


Fig. 1. Partial-order methods

The approach used here, visits all the transitions, but some independent events are put together to build a single transition step, the firing of this transition step is then atomic. The resulting graph is referred as covering step graph (CSG). The potential benefit of such a substitution may be exponential with respect to the number of "merged" independent transitions. In the case of Milner's scheduler [Mil 85], the gain is exponential: for  $n$  sites, the standard LTS consists of  $n \times 2^n$  states and  $(n^2 + n) \times 2^{n-1}$  transitions, while the CSG consists of  $n + 1$  states and  $n + 1$  transitions.

The CSG analysis allows the verification of global reachability properties, such as deadlock, but also more elaborated behavioral properties such as observational equivalence.

This paper follows the work initiated in [VAM 96]. The framework of failure semantics is now considered. First, a definition of such covering structures preserving failure semantics (testing equivalence [Bri 88] and CSP semantics [OH 86]) is proposed. In a second step, the problem of on-the-fly derivation is investigated. A general algorithm and associated sufficient conditions are presented, for such derivation. A specific instantiation of the general algorithm is then proposed and a first evaluation is made.

Section 2 presents the main concepts of failure semantics. General covering step graphs are presented in section 3. Section 4 adapts the general definition of CSG to failure semantics. Section 5 shows how to modify a standard enumeration algorithm in order to obtain a covering step graph preserving failure semantics. A preliminary assessment of the proposed approach is developed in section 6.

## 2 Failure Semantics

### Definition 1. Labelled Transition Systems (LTS)

A LTS is a quadruple  $\langle S, s_0, T, \rightarrow \rangle$  where :  $S$  is a set of states,  $s_0$  a distinguished state in  $S$ ,  $T$  is a set of transition labels,  $\rightarrow$  is a set of transitions ( $\rightarrow \subset S \times T \times S$ ).

Notations:

$$s \xrightarrow{t} \text{ iff } \exists s' \in S : (s, t, s') \in \rightarrow, s \not\xrightarrow{t} \text{ iff not } s \xrightarrow{t}, s \xrightarrow{t} s' \text{ iff } (s, t, s') \in \rightarrow$$

$$s_0 \xrightarrow{w} s_n \text{ iff } w = t_1.t_2 \dots t_n \text{ and } s_0 \xrightarrow{t_1} s_1, s_1 \xrightarrow{t_2} s_2, \dots, s_{n-1} \xrightarrow{t_n} s_n$$

In the sequel of the paper,  $\Sigma$  denotes a LTS,  $\Sigma = \langle S, s_0, T, \rightarrow \rangle$

### Definition 2. Observation and Experiments

Let  $T$  be the set of events and  $T_{Obs}$ , the subset of the **observed events** ( $T_{Obs} \subset T$ ).  $\epsilon$  denotes the empty sequence in  $T^*$  we define  $F_{Obs} : T^* \mapsto T_{Obs}^*$  the corresponding **hiding morphism**, as follows:

$$F_{Obs}(\epsilon) = \epsilon \text{ and for } o \in T, w \in T^* \quad F_{Obs}(o.w) = \begin{cases} o.F_{Obs}(w) & \text{if } o \in T_{Obs} \\ \epsilon.F_{Obs}(w) & \text{otherwise} \end{cases}$$

We use the following notation,  $w \in F_{Obs}^{-1}(\epsilon)$  to mean that  $w \in (T \setminus T_{Obs})^*$

- The notion of **experiment** is defined as usual,

For  $P, Q \in S$ ,  $w \in T_{Obs}^*$ , we note

$$P =_w Q \Leftrightarrow_{Def} \exists w' \in T^* : F_{Obs}(w') = w \text{ and } P \xrightarrow{w'} Q$$

As usual, we note  $P =_{\tau} Q$  instead of  $P =_{\epsilon} Q$

- The notion of **observable traces** is defined as usual, for  $P \in S$ , we note

$$Tr(\Sigma, P) =_{Def} \{w \in T_{Obs}^* : P =_w\} \text{ and } Tr(\Sigma) =_{Def} Tr(\Sigma, s_0)$$

### Definition 3. Stable State and Stable Failure

- A state  $P \in S$  is **stable** iff  $\forall t \in T : P \xrightarrow{t} \text{ implies } t \in T_{Obs}$

We note  $Stable(\Sigma)$ , the subset of  $S$  constituted of stable states.

- A **stable failure** of a state  $P$  is a pair  $(\rho, A)$ , where  $\rho \in T_{Obs}^*$  and  $A \subset T_{Obs}$ , satisfying:  $\exists Q \in Stable(\Sigma) : P =_{\rho} Q$  and  $\forall a \in A : Q \not\xrightarrow{a}$

We note  $S\_Ref(\Sigma, s)$ , the set of stable failures in LTS  $\Sigma$  of state  $s$

### Definition 4. Divergent state and Sequence of divergence

- A state  $P \in S$  is **divergent** iff  $[\forall k \geq 0, \exists w_k \in f_{Obs}^{-1}(\epsilon) : |w_k| = k \text{ and } P \xrightarrow{w_k}]$  where  $|w|$  denotes the length of sequence  $w$

We note  $Div(\Sigma)$ , the subset of  $S$  constituted of divergent states.

- $\rho \in T_{Obs}^*$  is a **sequence of divergence** of state  $P$  iff  $\exists Q \in Div(\Sigma) : P =_{\rho} Q$

We note  $Div\_Seq(\Sigma, P)$ , the subset of  $T_{Obs}^*$  constituted of divergent sequences in LTS  $\Sigma$  of state  $P$ .

**Definition 5.** The CSP semantics of a state  $P$  in LTS  $\Sigma$ ,  $CSP\_sem(\Sigma, P)$ , is the pair  $\langle CSPdiv(\Sigma, P), CSPref(\Sigma, P) \rangle$  where  
 $CSPdiv(\Sigma, P) =_{Def} \{\rho_1, \rho_2 \in T_{Obs}^* : \rho_1 \in Div\_Seq(\Sigma, P), \rho_2 \in T_{Obs}^*\}$   
 $CSPref(\Sigma, P) =_{Def} \{(\rho, A) \in T_{Obs}^* \times \mathcal{P}(T_{Obs}) : \rho \in CSPDiv(\Sigma, P) \text{ or } (\rho, A) \in S\_Ref(\Sigma, P)\}$

**Definition 6.** Conformance Relation and Testing Equivalence

Given two LTS,  $\Sigma$  and  $\Sigma'$  and their respective initial states  $s_0, s'_0$

- $\Sigma \underline{\mathbf{Conf}} \Sigma'$  iff  $\forall \sigma \in Tr(\Sigma'), \forall A \subset T_{Obs} :$   
 $[\exists q \in S, \forall a \in A : s_0 =\sigma \Rightarrow q \wedge q \not\Rightarrow a] \Rightarrow$   
 $[\exists q' \in S', \forall a \in A : s'_0 =\sigma \Rightarrow q' \wedge q' \not\Rightarrow a]$
- $\Sigma \underline{\mathbf{Te}} \Sigma'$  iff  $Tr(\Sigma') = Tr(\Sigma)$  and  $[\Sigma \underline{\mathbf{Conf}} \Sigma' \wedge \Sigma \underline{\mathbf{Conf}} \Sigma']$

### 3 Covering Step Graphs

#### 3.1 Mazurckiewicz's Traces and Independence Relation

**Mazurckiewicz's Traces [Maz 87]**

• A **concurrent alphabet** is a couple  $\mathcal{E} = (\alpha, \parallel)$  where  $\alpha$  is an alphabet and  $\parallel$  the *dependency* in  $\alpha$  ( $\parallel$  is a reflexive and symmetric relation). The complementary<sup>1</sup> relation  $\parallel^C$  of relation  $\parallel$  in  $\alpha$  is called *independency* relation in  $\alpha$ .

• **Mazurckiewicz's Traces:**

Let  $\mathcal{E}$  be concurrent alphabet  $(\alpha, \parallel)$ , the equivalence relation in  $\alpha^*$ ,  $\equiv_{\mathcal{E}}$  is defined by:  $W \equiv_{\mathcal{E}} W'$  iff there exists a finite sequence  $(W_0, W_1, \dots, W_n)$  such that

$W_0 = W$  and  $W_n = W'$  and

$\forall i, 1 \leq i \leq n : \exists u, v \in \alpha^*, \exists (a, b) \in \parallel^C$  such that  $W_{i-1} = uabv$  et  $W_i = ubav$

Relation  $\equiv_{\mathcal{E}}$  defines the trace equivalence over  $\mathcal{E}$ . Equivalence classes of  $\equiv_{\mathcal{E}}$  are called traces over  $\mathcal{E}$ . A trace generated by a string  $w$  is denoted by  $[w]_{\mathcal{E}}$ .

• **Trace properties**

$\equiv_{\mathcal{E}}$  is the least congruence in the monoid  $(\alpha^*, \cdot, \epsilon)$  satisfying:

$$(a, b) \in \parallel^C \Rightarrow a.b \equiv_{\mathcal{E}} b.a$$

**Independency relation [WG 93]**

$\wr$  a binary relation over the set of transitions ( $\wr \subset T \times T$ ) is an independent relation over a LTS  $\Sigma$  iff  $\forall s, s_1, s_2 \in S, \forall t_1, t_2 \in T :$

$$[t_1 \neq t_2, s \xrightarrow{t_1} s_1, s \xrightarrow{t_2} s_2 \text{ and } t_1 \wr t_2] \Rightarrow s_1 \xrightarrow{t_2} s' \text{ and } s_2 \xrightarrow{t_1} s'$$

$\wr^C$ , the complement relation of  $\wr$  is called *conflict*, or *dependency*, relation. In the sequel of the paper, the conflict relation  $\wr^C$  is denoted  $\#$ .

**Proposition** Let  $\Sigma, \wr, \mathcal{E}$  be respectively a LTS  $\langle S, s_0, T, \rightarrow \rangle$ , an independency relation over  $\Sigma$  and an associated concurrent alphabet  $(T, \wr^C)$ , then:

$$\forall s \in S, w_1, w_2 \in T^* \text{ such that } s \xrightarrow{w_1} s_1 \ \& \ s \xrightarrow{w_2} s_2 : [w_1]_{\mathcal{E}} = [w_2]_{\mathcal{E}} \Rightarrow s_1 = s_2$$

<sup>1</sup> For a binary relation  $R \subset U \times U$ ,  $R^C$  denotes the binary relation  $(U \times U) \setminus R$

### 3.2 Transition Steps

**Definition** The transition set  $E$  defines a **transition step** wrt  $\wr$  iff

$$\forall t_1, t_2 \in E : t_1 \wr t_2$$

In the sequel,  $Step(T, \wr)$  denotes the set of all transition steps derived from  $T$  wrt  $\wr$ . When no confusion is possible, we note  $Step(T)$  instead of  $Step(T, \wr)$ .

#### Sequences and Traces of Steps

Let  $Seq$  be the mapping  $\mathcal{P}(T) \mapsto \mathcal{P}(T^*)$  defined by:

$$Seq(\emptyset) = \{\epsilon\} \text{ and } Seq(E) = \bigcup_{e \in E} Seq(E \setminus \{e\}) \otimes e$$

where for  $\Omega \in \mathcal{P}(T^*)$  and  $w \in T^* : \Omega \otimes w = \{\omega.w : \omega \in \Omega\}$

**Property:**  $\forall E \in Step(T, \wr), \omega_1$  and  $\omega_2 \in Seq(E) \Rightarrow [\omega_1]_{(T, \wr)} = [\omega_2]_{(T, \wr)}$

**Trace of a transition step:** the former property allows to define the transition step trace as the trace of a string associated with the step by means of function  $Seq$ .

For  $E \in Step(T, \wr)$ ,  $[E]_{(T, \wr)} =_{def} [\mathcal{E}]_{(T, \wr)}$  where  $\mathcal{E} \in Seq(E)$

**Trace of a sequence of transition steps** is defined as the trace of the string obtained by concatenation of the strings associated with each step of the sequence. For  $E_1.E_2 \dots E_n \in Step(T, \wr)^*$

$$[E_1.E_2 \dots E_n]_{(T, \wr)} =_{def} [\mathcal{E}_1.\mathcal{E}_2 \dots \mathcal{E}_n]_{(T, \wr)} \text{ where } \mathcal{E}_i \in Seq(E_i)$$

#### Extension of the reachability relation and $\wr$ to transition steps

##### Support of a string:

Let  $\|\cdot\|$  be the mapping  $T^* \mapsto \mathcal{P}(T)$  defined by:

$$\|\epsilon\| = \emptyset \text{ and for } u \in T \text{ and } \omega \in T^* : \|u.\omega\| = \{u\} \cup \|\omega\|$$

##### Extensions of $\wr$

Let  $\wr \subset Step(T) \times Step(T)$  defined by  $T_1 \wr T_2$  iff  $T_1 \times T_2 \subset \wr$

Let  $\wr \subset Step(T) \times T^*$  defined by  $T_1 \wr w$  iff  $T_1 \times \|w\| \subset \wr$

Let  $\wr \subset T^* \times T^*$  defined by  $w_1 \wr w_2$  iff  $\|w_1\| \times \|w_2\| \subset \wr$

Let  $\Rightarrow$ , the extension of  $\rightarrow$  to transition steps be defined by:

$$\forall s, s' \in S, \forall E \in Step(T) : s \xrightarrow{E} s' \text{ iff } \forall e \in E : s \xrightarrow{e} \text{ and } \exists \omega \in Seq(E) : s \xrightarrow{\omega} s'$$

#### Diamond properties

1. For  $w_1, w_2 \in T^* : \text{If } s \xrightarrow{w_1}, s \xrightarrow{w_2} \text{ and } w_1 \wr w_2 \text{ then } s \xrightarrow{w_1.w_2} s' \text{ and } s \xrightarrow{w_2.w_1} s'$
2. For  $w \in T^*, E \in Step(T) : \text{If } s \xrightarrow{w}, s \xrightarrow{E} \text{ and } E \wr w \text{ then } s \xrightarrow{E.w} s' \text{ and } s \xrightarrow{w.E} s'$   
 $\forall \mathcal{E} \in Seq(E)$
3. For  $E_1, E_2 \in Step(T) : \text{If } s \xrightarrow{E_1}, s \xrightarrow{E_2} \text{ and } E_1 \wr E_2 \text{ then } s \xrightarrow{E_1 \cup E_2} s'$   
 $\text{And } s \xrightarrow{\mathcal{E}_1.\mathcal{E}_2} s' \text{ and } s \xrightarrow{\mathcal{E}_2.\mathcal{E}_1} s' \forall \mathcal{E}_i \in Seq(E_i)$

### 3.3 Covering Step Graph

**Definition 7.** Covering Step Graph (CSG)

Let  $\Sigma = \langle S, s_o, T, \rightarrow \rangle$ , be a LTS,  $\iota$  an independency relation over  $\Sigma$  (we let  $\# = \iota^C$ )

$\mathcal{E} = \langle S', s_o, \mathbf{T}, \rightsquigarrow \rangle$  is a Covering Step Graph for  $\Sigma$ , wrt  $\iota$  iff

- (1)  $S' \subset S$                       (2)  $\mathbf{T} \subset \text{Step}(T, \iota)$
- (3)  $\forall s, s' \in S' : s \rightsquigarrow^E s' \text{ implies } s \xrightarrow{E} s'$
- (4)  $\forall s \in S', \forall s' \in S' :$

$$s \xRightarrow{\omega} s' \text{ implies } \begin{cases} \exists s'' \in S', \exists \omega' \in T^*, \exists P_{\omega, \omega'} \in \mathbf{T}^* : \\ s' \xRightarrow{\omega'} s'', s \rightsquigarrow^{P_{\omega, \omega'}} s'' \text{ and } [w.\omega']_{(T, \#)} = [P_{\omega, \omega'}]_{(T, \#)} \end{cases}$$

note: any LTS may be viewed as a CSG by considering  $\iota = \emptyset$

**Milner's Scheduler** This scheduler is described in [Mil 85].  $n$  sites cyclically perform action  $A_i$  then action  $B_i$ . A scheduler constraints the execution of the whole system in such a way that the  $n$  sites alternatively perform actions  $A_1, A_2, \dots, A_n$ . Figure 2 depicts the net of the whole system in the case of  $n$  sites (scheduler +  $n$  sites), then the corresponding LTS and a CSG are derived in the case of 2 sites.

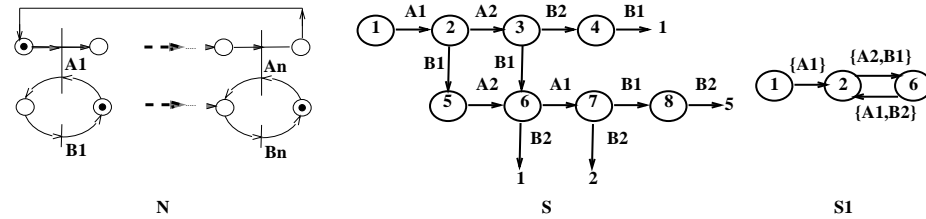


Fig. 2. Milner's Scheduler

Condition (1), each state of the step graph is a state of the standard LTS. For the Milner's scheduler,  $t_1 \not\mid t_2$  iff  $t_1 \neq t_2$ , consequently each subset of  $T$  constitutes a step of transitions and (2) holds. Condition (3) means that each transition step in the CSG ( $S1$ ) corresponds to a firing sequence in the standard LTS ( $S$ ): for instance  $:2 \xrightarrow{\{A_2, B_1\}} 6$  corresponds to  $2 \xrightarrow{A_2} 3 \xrightarrow{B_1} 6$  (or  $2 \xrightarrow{B_1} 5 \xrightarrow{A_2} 6$ ).

Finally, condition (4) expresses a "covering condition" between the firing sequences of the standard LTS and the step sequences of the CSG. In  $S$ ,  $1 \xrightarrow{A_1} 2 \xrightarrow{A_2} 3 \xrightarrow{B_2} 4 \xrightarrow{B_1} 1$ , condition 4 of definition 7 ensures that a "covering" step sequence exists in  $S1$ . The initial sequence,  $A_1.A_2.B_2.B_1$ , is extended by  $A_1$ , in the CSG the following step sequence  $\Omega = \{A_1\}.\{A_2, B_1\}.\{A_1, B_2\}$  covers it:  $1 \xrightarrow{\Omega} 2$  and  $1 \xrightarrow{A_1.A_2.B_2.B_1.A_1} 2$  with  $[A_1.A_2.B_2.B_1.A_1] = [\{A_1\}.\{A_2, B_1\}.\{A_1, B_2\}]$

## 4 Covering Step Graphs preserving Failure Semantics

In this section, we propose a finer definition for covering structures in order to provide reduced labelled transition systems preserving failure semantics. In the sequel, we use FCSG as shortland to denote such structures.

**Definition 8.** Hiding morphism on Steps

Let  $T$  be an alphabet,  $T_{Obs}$ , the subset of the observed events ( $T_{Obs} \subset T$ ) and  $\mathbb{T} \in \mathcal{P}(\mathcal{P}(T))$

$T_{Obs}$  and  $\mathbb{T}$  are **compatible** iff  $\forall E \in \mathbb{T} : Card(E \cap T_{Obs}) \leq 1$   
( $Card(F)$  denotes the number of elements of set  $F$ )

For  $T_{Obs}$  and  $\mathbb{T}$  compatible, we define  $\mathbb{F}_{Obs}$ , a corresponding **hiding morphism** as follows:

$$\mathbb{F}_{Obs} : \mathbb{T}^* \mapsto T_{Obs}^* \text{ by: } \mathbb{F}_{Obs}(E.F) = \begin{cases} e.\mathbb{F}_{Obs}(F) & \text{if } E \cap T_{Obs} = \{e\} \\ \epsilon.\mathbb{F}_{Obs}(F) & \text{otherwise} \end{cases}$$

We use the following notation,  $w \in \mathbb{F}_{Obs}^{-1}(\epsilon)$  to mean that  $w \in \mathbb{T}^*$  and  $\mathbb{F}_{Obs}(w) = \epsilon$

**Property:** For  $T_{Obs}$  and  $\mathbb{T}$  compatible:  $\forall E \in \mathbb{T}^*, \forall \mathcal{E} \in Seq(E) : \mathbb{F}_{Obs}(E) = f_{Obs}(\mathcal{E})$

**Definition 9.** Covering Step Graph preserving Failure Semantics (FCSG)

A Covering Step Graph  $\mathbb{S} = \langle \mathcal{G}, s_o, \mathbb{T}, \sim \rangle$  for LTS  $\Sigma$ , wrt  $\wr$  is a FCSG iff

- (i)  $\forall E \in \mathbb{T} : [E \cap T_{Obs} \neq \emptyset \Rightarrow E = \{t\}$  for some  $t \in T_{Obs}]$
- (ii)  $\forall s \in S \cap \mathcal{G}, \forall s' \in \mathcal{G} : s \xrightarrow{\omega} s'$  implies  $\left\{ \begin{array}{l} \exists s'' \in \mathcal{G}, \exists \omega' \in f_{Obs}^{-1}(\epsilon), \exists P_{\omega.\omega'} \in \mathbb{T}^* : \\ - s' \xrightarrow{\omega'} s'', s \xrightarrow{P_{\omega.\omega'}} s'' \\ - [\omega.\omega']_{(T, \#)} = [P_{\omega.\omega'}]_{(T, \#)} \\ - f_{Obs}(\omega.\omega') = \mathbb{F}_{Obs}(P_{\omega.\omega'}) \end{array} \right.$

Condition (i) ensures that a step of transition is either reduced to a single observable transition or only composed of unobservable transitions. In this case  $T_{Obs}$  and  $\mathbb{T}$  are compatible. Condition (ii) is similar to condition (4) of definition 7. Two differences appear: first the sequence  $\omega'$  is, now, only composed of unobservable transitions and, second, equality of traces (in the sense of Mazurkiewicz) is augmented by a condition ensuring that the associated sequences (respectively in the initial LTS and in the CSG) corresponds to the same observation.

The rest of this section shows that the former definition provides a definition of covering structures preserving CSP semantics and Testing Equivalence. First, the concepts associated with failure semantics (stability, divergence, failure) are (re)-defined in the case of covering structures and then we show that these properties may be decided directly on the FCSG.

#### 4.1 Preservation of failure semantics concepts

**Conservation of Experiments** We first define the notion of experiments in Covering structures and show after that the two notions coincide.

The notion of experiment is defined as usual, for  $P, Q \in \mathcal{G}$  and  $\rho \in T_{Obs}^*$ , we note

$$P \approx \rho \approx \approx Q \Leftrightarrow_{Def} \exists w_\rho \in \mathbb{T}^* : \mathbb{F}_{Obs}(w_\rho) = \rho \text{ and } P \xrightarrow{w_\rho} Q$$

As usual, we note  $P \approx \tau \approx \approx Q$  instead of  $P \approx \epsilon \approx \approx Q$

**Proposition 4.1**  $\forall s \in \mathcal{G}$  :

a)  $[s \approx \rho \approx \approx q \text{ implies } s = \rho \Rightarrow q]$  And b)  $[s = \rho \Rightarrow q \text{ implies } \exists q' \in \mathcal{G} : s \approx \rho \approx \approx q']$

a) by definition of  $\approx \rho \approx \approx$ ,  $\exists P_{w_\rho} \in \mathbb{T}^*$  such that  $\mathbb{F}_{Obs}(P_{w_\rho}) = \rho$  and  $s \xrightarrow{P_{w_\rho}} q$ . Since  $\mathbb{E}$  is a CSG of  $\Sigma$ , condition (3) (def 7) holds and  $\exists w_\rho \in T^*$  : such that  $f_{Obs}(w_\rho) = \rho$  and  $s \xrightarrow{w_\rho} q$  and, finally  $s = \rho \Rightarrow q$ .

b) by definition of  $= \rho \Rightarrow$ , there exists a sequence  $w_\rho$  such that  $f_{Obs}(w_\rho) = \rho$  and  $s \xrightarrow{w_\rho} q$ .

By application of condition (i) of FCSG def. (9),  $\exists w' \in f_{Obs}^{-1}(\epsilon)$ ,  $\exists q' \in \mathcal{G}$ ,  $\exists P_{w_\rho} \in \mathbb{T}^*$  such that  $q \xrightarrow{w'} q'$ ,  $\mathbb{F}_{Obs}(P_{w_\rho}) = \rho$  and  $s \xrightarrow{P_{w_\rho}} q'$  and, finally  $s \approx \rho \approx \approx q'$ .

**Corollary 4.1.a**  $\forall s \in \mathcal{G} : Tr(\Sigma, s) = Tr(\mathbb{E}, s)$

**Corollary 4.1.b**  $\forall s \in \mathcal{G} : s = \rho \Rightarrow q$  and  $q \in Stable(\Sigma) \Rightarrow s \approx \rho \approx \approx q$

In the proof of b), since  $q$  is stable in  $\Sigma$ , necessarily  $w'$  equals  $\epsilon$  and  $q' = q$ .

**Corollary 4.1.c**  $Stable(\Sigma) \subset \mathcal{G}$  direct consequence of Cor. 4.1.b.

#### Preservation of stability and divergence

For technical reasons, we introduce the stability level of a state  $s$ , that is the maximal length of unobservable sequences from state  $s$ . This notion is introduced for LTS and also for covering structures.

**Stability level in a LTS:**  $\tau\_max(\Sigma, s) = k$  iff

$$\exists w \in f_{Obs}^{-1}(\epsilon) : |w| = k \text{ and } s \xrightarrow{w} \text{ and } \forall w' \in f_{Obs}^{-1}(\epsilon) : [|w'| > k \Rightarrow s \not\xrightarrow{w'}]$$

**Stability level in a CSG:**  $\tau\_max(\mathbb{E}, s) = k$  iff

$$\exists P_\tau \in \mathbb{F}_{Obs}^{-1}(\epsilon) : \|P_\tau\| = k, s \xrightarrow{P_\tau} \text{ and } \forall P' \in \mathbb{F}_{Obs}^{-1}(\epsilon) : [|P'| > k \Rightarrow s \not\xrightarrow{P'}]$$

**Proposition 4.1:**  $\forall s \in \mathcal{G} : \tau\_max(\Sigma, s) = \tau\_max(\mathbb{E}, s)$

•  $\tau\_max(\Sigma, s) \leq \tau\_max(\mathbb{E}, s)$

Let  $k = \tau\_max(\Sigma, s)$ , there exists a sequence  $w \in f_{Obs}^{-1}(\epsilon)$ ,  $\exists q \in S : |w| = k, s \xrightarrow{w} q$ . Since  $s \in \mathcal{G}$ , condition (ii) of FCSG def. (9) holds and  $\exists w' \in f_{Obs}^{-1}(\epsilon)$ ,  $\exists q' \in \mathcal{G}$ ,  $\exists P_{w.w'} \in \mathbb{T}^*$  verifying:  $q \xrightarrow{w'} q'$ ,  $s \xrightarrow{P_{w.w'}} q'$ ,  $[P_{w.w'}] = [w.w']$  and  $\mathbb{F}_{Obs}(P_{w.w'}) = f_{Obs}(w.w')$

$[P_{w.w'}] = [w.w']$  implies that  $\|P_{w.w'}\| > k$ . Since  $\mathbb{F}_{Obs}(P_{w.w'}) = f_{Obs}(w.w')$ , we have  $\mathbb{F}_{Obs}(P_{w.w'}) = \epsilon$ . Finally,  $s \xrightarrow{P_{w.w'}} q'$  then  $\tau\_max(\mathbb{E}, s) \geq k$

- $\tau\_max(\mathbb{E}, s) \leq \tau\_max(\Sigma, s)$

Let  $k = \tau\_max(\mathbb{E}, s)$ , there exists  $P_\tau \in \mathbb{F}_{Obs}^{-1}(\epsilon)$ ,  $\exists q \in \mathcal{G} : \|P_\tau\| = k, s \xrightarrow{P_\tau} q$ . Condition (3) of CSG def. (7) holds and  $\exists w \in f_{Obs}^{-1}(\epsilon)$  verifying  $|w| = k, [P_\tau] = [w]$  and  $s \xrightarrow{w} q$  then  $\tau\_max(\Sigma, s) \geq k$

**Corollary 4.1\_a:**  $Stable(\Sigma) = Stable(\mathbb{E})$

First note that a state  $s$  is stable in  $\Sigma$  iff  $\tau\_max(\Sigma, s) = 0$

- $Stable(\Sigma) \subset Stable(\mathbb{E})$ : Direct application of Corollary 4.1.c, and prop. 4.1
- $Stable(\mathbb{E}) \subset Stable(\Sigma)$ :  $Stable(\mathbb{E}) \subset \mathcal{G} \subset S$  and prop. 4.1 gives the result.

**Corollary 4.1\_b:**  $Div(\Sigma) \cap \mathcal{G} = Div(\mathbb{E})$ : By absurd and application of prop 4.1.

**Preservation of stable failures** We first recall the notion of output sets for LTS and give the corresponding in CSG.

**Output sets in a LTS** Let  $Out$ , the map from:  $S \rightsquigarrow \mathcal{P}(T)$  defined as follows:

$$Out(\Sigma, s) =_{def} \{t \in T : s \xrightarrow{t}\}$$

**Output sets in a CSG** Let  $Out$ , the map from:  $\mathcal{G} \rightsquigarrow \mathcal{P}(T)$  defined as follows:

$$Out(\mathcal{G}, s) =_{def} \bigcup_{E \in \mathcal{T}} \{E : s \xrightarrow{E}\}$$

**Proposition 4.1:** For  $s \in Stable(\mathbb{E}) : Out(\Sigma, s) = Out(\mathbb{E}, s)$

- $Out(\Sigma, s) \subset Out(\mathbb{E}, s)$  Let  $t \in T_{Obs} \cap Out(\Sigma, s)$  that is  $s \xrightarrow{t} s'$ . Condition ii) of FCSG def (9) implies that there exists an unobservable sequence  $w'$ , a state  $s'$  and a sequence of transition steps  $P_{w.w'}$  such that  $s' \xrightarrow{w'} s''$  and  $s \xrightarrow{P_{w.w'}} s'$  and  $\mathbb{F}_{Obs}(P_{w.w'}) = f_{Obs}(t.w') = t$ . Since  $s$  is stable for  $\mathbb{E}$ ,  $P_{t.w'}$  is necessarily of the following form  $P_{t.w'} = \{t\}.P_\tau$  where  $P_\tau$  is a sequence of unobservable transition steps and consequently,  $t \in Out(\mathbb{E}, s)$

- $Out(\mathbb{E}, s) \subset Out(\Sigma, s)$  Let  $t \in T_{Obs} \cap Out(\mathbb{E}, s)$  that is  $s \xrightarrow{\{t\}} s'$ . Condition (3) of CSG def (7) implies that  $s \xrightarrow{t} s'$  and  $t \in Out(\Sigma, s)$

**Corollary 4.1:**  $S\_Ref(\Sigma, s_0) = S\_Ref(\mathbb{E}, s_0)$

- $S\_Ref(\Sigma, s_0) \subset S\_Ref(\mathbb{E}, s_0)$ : Let  $(\rho, A) \in S\_Ref(\Sigma, s_0)$  that is  $s_0 = \rho \Rightarrow q$ ,  $q \in Stable(\Sigma)$  and  $Out(\Sigma, q) = A$ . Corollary 4.1.b holds and we have  $s_0 \approx \rho \approx q$ . Moreover  $q \in Stable(\mathbb{E})$  (prop 4.1.c). Finally prop 4.1 holds and  $Out(\mathbb{E}, q) = Out(\Sigma, q)$  then  $(\rho, A) \in S\_Ref(\mathbb{E}, s_0)$ .

- $S\_Ref(\mathbb{E}, s_0) \subset S\_Ref(\Sigma, s_0)$ : Let  $(\rho, A) \in S\_Ref(\mathbb{E}, s_0)$ . Prop 4.1 holds and we have  $s_0 = \rho \Rightarrow q$ . Moreover  $q \in Stable(\Sigma)$  (prop 4.1). Finally prop 4.1 holds and  $Out(\Sigma, q) = Out(\mathbb{E}, q)$  then  $(\rho, A) \in S\_Ref(\Sigma, s_0)$ .

### Preservation of divergent sequences

$\forall s \in \mathcal{G} : Div\_Seq(\Sigma, s) = Div\_Seq(\mathbb{E}, s)$

**Lemma:**  $\forall s \in \mathcal{G} : Div\_Seq(\Sigma, s) \subset Div\_Seq(\mathbb{E}, s)$

$\rho \in Div\_Seq(\Sigma, s)$  and  $s \in \mathcal{G}$  imply that  $s \approx \rho \approx q'$  (prop 4.1). Now suppose by absurd that  $\rho \notin Div\_Seq(\mathbb{E}, s)$ .

For technical reasons, we introduce the stability level of a non divergent sequence, that is, the maximum level of stability of each intermediate state associated with the sequence.

#### Stability level of a non divergent sequence

Let  $Pref(\rho)$  be the set of all prefixes of  $\rho$ .

$$Pref(\rho) =_{def} \{\rho_1 \in T_{Obs}^* : \exists \rho_2 \in T_{Obs}^* \text{ such that } \rho = \rho_1 \cdot \rho_2\}$$

Let  $Pref(\rho, s)$  be the subset of  $\mathcal{G}$  defined as follows:

$$Pref(\rho, s) =_{def} \{q \in \mathcal{G} : \exists \rho_1 \in Pref(\rho) \text{ such that } s \approx \rho_1 \approx q\}$$

For  $\rho$  a non divergent sequence, let  $\tau\_max(\rho, s) =_{def} Max(\{\tau\_max(\mathbb{E}, q) : q \in Pref(\rho, s)\})$

Let  $\rho \notin Div\_Seq(\mathbb{E}, s)$ , and  $k = \tau\_max(\rho, s)$

Since  $\rho \in Div\_Seq(\Sigma, s)$ ,  $\exists w_\rho \in T^*$ ,  $\exists q \in Div(\Sigma)$  such that  $f_{Obs}(w_\rho) = \rho$  and  $s \xrightarrow{w_\rho} q$

Since  $q \in Div(\Sigma)$ ,  $\exists w_\tau \in (T \setminus T_{Obs})^{((k+1) \times |w_\rho|)}$ ,  $\exists q' \in S$  such that  $q \xrightarrow{w_\tau} q'$

Now, consider the sequence  $w_\rho \cdot w_\tau$ , we have  $s \xrightarrow{w_\rho \cdot w_\tau} q'$ . Since  $s \in \mathcal{G}$ , condition (ii) of FCSG def. (9) holds and it follows that  $\exists w' \in f_{Obs}^{-1}(\epsilon)$ ,  $\exists q' \in \mathcal{G}$ ,  $\exists P_{w_\rho \cdot w_\tau \cdot w'}$   $\in$

$T^*$  such that  $s \xrightarrow{P_{w_\rho \cdot w_\tau \cdot w'}} q'$  and  $[P_{w_\rho \cdot w_\tau \cdot w'}] = [w_\rho \cdot w_\tau \cdot w']$

By construction,  $|w_\rho \cdot w_\tau \cdot w'| \geq |w_\rho| + (k+1) \times |w_\rho|$  (i.e.  $(k+2) \times |w_\rho|$ )

$$\text{Since } [P_{w_\rho \cdot w_\tau \cdot w'}] = [w_\rho \cdot w_\tau \cdot w'], \|P_{w_\rho \cdot w_\tau \cdot w'}\| \geq (k+2) \times |w_\rho|$$

On the other hand, since  $k = \tau\_max(\rho, s)$ ,  $\|P_{w_\rho \cdot w_\tau \cdot w'}\| \leq (k+1) \times |f_{Obs}(w_\rho)|$

Finally, since  $|f_{Obs}(w_\rho)| \leq |w_\rho|$  the following inequation holds:

$$(k+2) \times |w_\rho| \leq \|P_{w_\rho \cdot w_\tau \cdot w'}\| \leq (k+1) \times |w_\rho| \text{ leading to the contradiction.}$$

**A contradiction occurs in the two cases, then  $\rho \in Div\_Seq(\mathbb{E}, s)$**

**Lemma**  $Div\_Seq(\mathbb{E}, s) \subset Div\_Seq(\Sigma, s)$

Let  $\rho \in Div\_Seq(\mathbb{E}, s)$ , Conservation of experiments (cor. 4.1) and divergent states (cor. 4.1.b) imply that  $\rho \in Div\_Seq(\Sigma, s)$ .

### Preservation of CSP semantics

If  $\mathbb{E}$  is a FCSG of LTS  $\Sigma$  then  $CSP\_Sem(\Sigma) = CSP\_Sem(\mathbb{E})$

By construction  $s_0$  is a common initial state for  $S$  and  $\mathcal{G}$ , the result follows from corollaries 4.1 and 4.1.

### Preservation of Testing Equivalence

If  $\mathbb{E}$  is a FCSG of LTS  $\Sigma$  then  $\Sigma \underline{\text{Te}} \mathbb{E}$

From Corollary 4.1.a, it follows that  $Tr(\Sigma) = Tr(\mathbb{E})$ . We have to prove that  $\Sigma \underline{\text{Conf}} \mathbb{E}$  and  $\mathbb{E} \underline{\text{Conf}} \Sigma$ .

- $\mathcal{E}$  **Conf**  $\Sigma$  Let  $q \in \mathcal{G}$ ,  $A \subset T_{Obs}$  and  $\rho \in T_{Obs}^* : s_0 \approx \rho \approx q$  and  $q \approx \mu \approx$   
 Since  $s_0 \approx \rho \approx q$ , Prop 4.1 implies that  $s_0 = \rho \Rightarrow q$ . Since  $q \in \mathcal{G}$ ,  $Tr(\Sigma, q) = Tr(\mathcal{E}, q)$  then  $q \neq \mu \Rightarrow \forall a \in A$
- $\Sigma$  **Conf**  $\mathcal{E}$  Let  $q \in S$ ,  $A \subset T_{Obs}$  and  $\rho \in T_{Obs}^* : s_0 = \rho \Rightarrow q$  and  $q \neq \mu \Rightarrow$   
 Since  $\Sigma$  is a FCSG of  $\Sigma$ , condition (ii) of FCSG def. holds and  $\exists w \in f_{Obs}^{-1}(\epsilon)$ ,  $\exists q' \in \mathcal{G}$ ,  $\exists P_{\rho, w} \in \mathbb{T}^* : F_{Obs}(P_{\rho, w}) = \rho$  then  $s_0 \approx \rho \approx q'$  and  $q = \tau \Rightarrow q'$ . Since  $q' \in \mathcal{G} : Tr(\Sigma, q') = Tr(\mathcal{E}, q')$ . On the other hand, since  $q = \tau \Rightarrow q'$ ,  $Tr(\Sigma, q') \subset Tr(\Sigma, q)$ .  
 Finally,  $Tr(\mathcal{E}, q') \subset Tr(\Sigma, q)$  and  $\forall a \in A \quad q' \approx \mu \approx$

## 5 Generation on the fly of FCSG

We first, recall the main steps for the on the fly generation of CSG and we show how to adapt the previous construction to provide FCSG.

### 5.1 On the Fly Covering Step Graph Derivation

Left part of Table 1 describes the general structure of the basic algorithm. This algorithm is similar to a standard algorithm for computing a reachable marking graph, that is  $Enabled(q)$  computes the set of transitions enabled by state  $q$ , and  $fire(q, t)$  computes the state obtained from state  $q$  by firing transition  $t$ .

The changes with respect to a standard algorithm are the following.

- The independence relation  $\iota$  is supplied,
- The enable transitions are split in two subsets by means of functions  $T_U$  and  $T_M$  :
  - $T_u$ , defines transitions to be explored in a standard way,
  - $T_m$  defines transitions whose exploration will be conducted within a step. In the sequel, such transitions will be referred as “mergeable” transitions.
- The set of transition steps  $\Pi_{T_m}$  built by means of function  $\Pi$ , whose domain is the set of mergeable transitions  $T_m$

Right part of Table 1 describes some sufficient conditions, on the sets  $T_u, T_m$ , and  $\Pi_{T_m}$ , ensuring that the algorithm of Table 1 produces indeed a CSG. These conditions are parameterized by a transitive conflict relation weaker than the initial one. This transitive relation, denoted  $\#\#$ , is such that  $\# \subset \#\#$ , consequently  $\#\#^C \subset \iota$ .

Condition  $CA_1$  defines a partition of the enabled transition set. Condition  $CA_2$  implies that no (initial) conflict may occur between a mergeable transition and a disabled transition. Condition  $CB_1$  implies that no (direct or indirect) conflict exists between transitions within a step. Condition  $CB_2$  implies that the considered computation steps *subsume* any subset of strongly independent transitions.

<pre> 1. Initialise: Stack is empty ; push <math>s_0</math> onto Stack H is empty ; enter <math>s_0</math> in H A is empty ;  2. Loop : while Stack <math>\neq \emptyset</math> loop {  pop(<math>q</math>) from stack    <math>T \leftarrow Enabled(q)</math>;     <math>T_u \leftarrow T_U(T, \iota)</math>    <math>T_m \leftarrow T_M(T, \iota)</math>     <math>\Pi_{T_m} \leftarrow \Pi(T_m, \iota)</math>    <math>\forall \pi \in \Pi_{T_m}</math> do      { <math>q'' \leftarrow q</math>;        <math>\forall p \in \pi</math> do { <math>q' = fire(q'', p)</math>;                      <math>q'' \leftarrow q'</math>;                      enter <math>\langle q, s, q'' \rangle</math> in A;                      if <math>q'' \in H</math> then                        {enter <math>q''</math> in H;                         put <math>q'</math> onto Stack}                      }      }     <math>\forall t \in T_u</math> do      { <math>q' = fire(q, t)</math>;        enter <math>\langle q, \{t\}, q' \rangle</math> in A        if <math>q' \in H</math> then          {enter <math>q'</math> in H;           put <math>q'</math> onto Stack}      } } </pre>	<p>Conditions for General CSG derivation:  <math>(CB_1), (CB_2), (CA_1), (CA_2)</math></p> <p><math>(CB_1)</math> Steps of Independent transitions  <math>\forall \pi \in \Pi_{T_m} : \pi \in Step(T, \#\#^C)</math></p> <p><math>(CB_2)</math> Covering Condition  <math>\forall P \in Step(T_m, \#\#^C), \exists \pi \in \Pi_{T_m} : P \subseteq \pi</math></p> <p><math>(CA_1)</math> Partition of <math>Enabled(q)</math>  <math>T_m \cup T_u = Enabled(q)</math> and <math>T_m \cap T_u = \emptyset</math></p> <p><math>(CA_2)</math> In order to avoid confusion  If <math>t \in T_m</math> then <math>t' \# t \Rightarrow t' \in Enabled(q)</math></p> <p style="text-align: center;">*****</p> <p>Conditions for Failure CSG derivation:  <math>(CB_1), (CB_2), (CA_1), (FCA_2)</math></p> <p><math>(FCA_2)</math> Take into account Confusion  and Observation</p> <p>If <math>t \in T_m</math> then <math>t \notin T_{Obs}</math> and  <math>[t' \# t \Rightarrow [t' \in Enabled(q) \text{ and } t' \notin T_{Obs}]]</math></p>
--	--

**Table 1.** General algorithm and Sufficient Conditions for CSG derivation

## 5.2 Specific Condition for FCSG derivation

The initial conditions are maintained, however the condition  $(CA_2)$  is refined in order to take into account the set of observed events,  $T_{Obs}$ . The new formulation, called  $(FCA_2)$ , is as follows:

If  $t \in T_m$  then  $[t \notin T_{Obs} \text{ and } t' \# t \Rightarrow [t' \in Enabled(q) \text{ and } t' \notin T_{Obs}]]$

This new condition implies that  $T_m$  the set of mergeable transitions does not contain any observed event or any event in conflict with an observed event.

$(FCA_2)$  is obviously stronger than the initial formulation  $(CA_2)$ . When  $T_{Obs}$  is reduced to the empty set, then the two conditions are equivalent.

**Motivation of  $(FCA_2)$**  Consider the LTS product associated with the two LTS  $A$  and  $B$  of the left part of fig. 3. With respect to LTS  $A$ ,  $E1$  and  $O1$  are

in conflict. CSG, of the left part of fig. 3, violates condition  $FCA_2$  (consider steps of transitions involving transition E1) and does not satisfy condition (ii) of FCSG definition: the following sequence of the standard LTS: E2.O2.O1.E1 is not covered in the CSG. For instance, the following sequence of transition steps  $\{E1, E2\}. \{O2\}. \{O1\}$  corresponds to the same observation (O2.O1) but does not admit the same trace (cf permutation of E1 and O1).

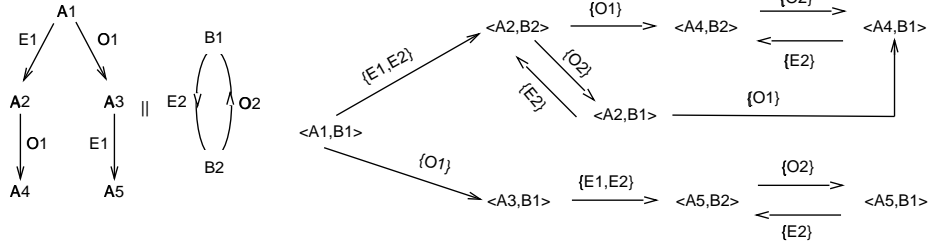


Fig. 3. Motivation of ( $FCA_2$ )

### 5.3 Sufficient Conditions for FCSG derivation

Since ( $FCA_2$ ) is stronger than  $CA_2$ , the basic algorithm, depicted by Table 1 leads to a correct CSG computation, as far as the other conditions hold.

With respect to the FCSG definition 9 we have to prove

(i)  $\forall E \in \mathbb{T} : [E \cap T_{Obs} \neq \emptyset \Rightarrow E = \{t\} \text{ for } t \in T_{Obs}]$

(ii)  $\forall s \in S \cap \mathcal{G}, \forall s' \in \mathcal{G} : s \xrightarrow{\omega} s' \text{ implies } \begin{cases} \exists s'' \in \mathcal{G}, \exists \omega' \in f_{Obs}^{-1}(\epsilon), \exists P_{\omega.\omega'} \in \mathbb{T}^* : \\ s' \xrightarrow{\omega'} s'', s \xrightarrow{P_{\omega.\omega'}} s'' \\ \text{and } [\omega.\omega']_{(T,\#)} = [P_{\omega.\omega'}]_{(T,\#)} \\ \text{and } f_{Obs}(\omega.\omega') = \mathbb{F}_{Obs}(P_{\omega.\omega'}) \end{cases}$

Since  $FCA_2$  implies that  $T_M \cap T_{Obs} = \emptyset$ , condition (i) trivially holds. The next section is devoted to the proof of condition (ii).

The proof is similar to the one given in [VAM 96] for general CSG derivation.

**Proof of condition (ii)** First, we introduce a factorization lemma which permits the proof of condition (ii) to be established by a simple recurrence on  $|\omega|$  (the proof omitted here is similar to the one given in [VAM 96]). Due to the lack of space, we only give the factorization lemma and its proof. A specific factorization operator is defined in order to take into account new constraints due to the set of observed events.

**Factorization Lemma**

$\forall s \in \mathcal{S}, \forall w \in T^* : s \xrightarrow{w} s' \Rightarrow$  (a) or (b) where

(a)  $\exists t \in T_u, s \xrightarrow{t.w_1} s'$  with  $[w] = [t.w_1]$

(b)  $\exists F \in \Pi_{T_m}, \exists E \subset F : s \xrightarrow{E.w_1} s'$  with  $[w] = [E.w_1]$  and  $(F \setminus E) \perp ||w_1||$

(b) means that  $w$  may be rewritten as  $[E.w_1]$ , where  $E$  is a computation step included in step  $F$ , (or  $E = F$ ), such that any  $F$  transition which does not occur within  $E$  is independent of sequence  $w_1$ .

**Factorization operator  $\pi$ :** For technical reasons, *factorization* operator  $\pi$  is first introduced. This operator extracts from any transition sequence the maximal computation step, or the maximal step prefix. Operator  $\pi$  is defined according to the selected conflict relation and the set of observed events  $T_{Obs}$ .

$\pi$  is the mapping from  $S \times Step(T) \times T^* \times T^* \mapsto Step(T) \times T^*$  defined by:

$$\begin{aligned} \pi(s, E, w, \epsilon) &= (E, w) \\ \pi(s, E, w_1, w_2) &= (E, w_1.w_2) \text{ if } E \in \Pi_{T_m} \\ \pi(s, E, w_1, t.w') &= \begin{cases} \pi(s, E \cup \{t\}, w_1, w') & \text{if } t \in T_m, \{t\} \perp ||w_1|| \\ & \text{and } E \cup \{t\} \in Step(T, \#^C) \\ (\{t\}, \mathcal{E}.w_1.w') & \text{if } t \in T_u, \{t\} \perp (||w_1|| \cup E), \\ & \mathcal{E} \in Seq(E) \text{ and} \\ & [t \in T_{Obs} \Rightarrow ||w_1|| \cap T_{Obs} = \emptyset] \\ \pi(s, E, w_1.t, w') & \text{otherwise} \end{cases} \end{aligned}$$

**$\pi$  Properties**

Let  $s \in \mathcal{S}, w, w_1 \in T^*$  be such that  $s \xrightarrow{w} s'$  and  $\pi(s, \emptyset, \epsilon, w) = (E, w_1)$  Then

- (1)  $s \xrightarrow{E} s_1, s_1 \xrightarrow{w_1} s', [w] = [E.w_1]$  and  $f_{Obs}(w) = f_{Obs}(E.w_1)$
- (2)  $\forall t \in ||w_1|| : t \xrightarrow{s} \Rightarrow$  (a) or (b) where
  - (a)  $t \notin T_{Obs}$  and  $\exists t' \in E : t \# t'$
  - (b)  $t \notin T_{Obs}$  and  $w_1 = u.t.v$  for  $u, v \in T^*$  and  $f_{Obs}(u) \neq \epsilon$

*proof of factorization lemma:*

Let  $s \in S, w, w_1 \in T^*$  be such that  $s \xrightarrow{w} s'$  and  $\pi(s, \emptyset, \epsilon, w) = (E, w_1)$

As a result of the  $\pi$  construction, three cases occur:  $E = \{t\}$  and  $t \in T_u, E \in \Pi_{T_m}, E \notin \Pi_{T_m}$ . The first two cases are similar, and the result directly follows from  $\pi$  property (1). The third case  $E \notin \Pi_{T_m}$  needs to be developed.

By construction,  $E \in Step(T, \#^C)$ , condition  $CB_2$  implies that transition subset  $F \in \Pi_{T_m}$  exists, verifying  $E \subset F$ ;  $\pi$  property (1) implies trace equality and the correspondence of observation, the relation  $F \setminus E \perp w_1$  is the last to be proved.

By absurd, let  $R$  be  $F \setminus E$  and assume that there exists transition  $t' \in ||w_1||$  in conflict with  $t \in R$ . Because of  $t \in R$  implies that  $s \xrightarrow{t}$ , condition  $FCA_2$  implies that  $t \notin T_{Obs}$  and  $s \xrightarrow{t'}$ .  $\pi$  property (2) then holds:  $\exists t'' \in E$  and  $t' \equiv_{\#} t''$ . Finally,  $\exists t, t'' \in F$  such that  $t \not\equiv_{\#} t''$  ( $\equiv_{\#}$  is transitive) and  $F \in Step(T, \equiv_{\#}^C)$ , which furnishes the contradiction.

#### 5.4 Step Graph of Crossed Conflicts

As in [VAM 96], Step Graph of Crossed Conflicts may be used to obtain a specific algorithm to build Failure Covering Step Graph by sample instantiation of the general algorithm depicted in Table 1. The specificity due to the failure semantics is taken into account during the choice of  $T_m$ : the set of "mergeable transitions".

This section presents a particular conflict relation and definitions of functions  $T_M, T_U$ , and  $\Pi$ . Thereby, a specific implementation of the former basic algorithm is provided to build FCSG.

**Weak Conflict and Strong Independence** Let  $\equiv_{\#}$  be the reflexive and transitive closure of relation  $\#$ . This relation is called *weak conflict relation*. Relation  $\equiv_{\#}$  is an equivalence relation and  $\# \subset \equiv_{\#}$ . Relation  $\equiv_{\#}^C$ , i.e. the complement relation, is called strong independence relation and  $\equiv_{\#}^C \subset \imath$

An easy way to obey to rule  $CB_2$  is to consider only maximal computation steps. Since conflict relation is an equivalence relation ( $\equiv_{\#}$ ), the associated conflict sets supply a partition of the transition set. The set of maximal computation steps results from the *orthoproduct* [PF 90] of the equivalence classes of relation  $\equiv_{\#}$

**Orthoproduct** : Let  $\mathcal{IE}$  be a set of sets ( $\mathcal{IE} \in \mathcal{P}(\mathcal{P}(U))$ ),  $\mathcal{IE} = \{E_1, E_2, \dots, E_n\}$   
 $\Pi_C(\mathcal{IE}) =_{def} \{\{e_1, e_2, \dots, e_n\} : (e_1, e_2, \dots, e_n) \in E_1 \times E_2 \dots \times E_n\}$

The set  $\Pi_C(T_m / \equiv_{\#})$  is called **Crossed Conflict Step**.

*Properties of Crossed Conflict Step:*

- $E \in \Pi_C(T_m / \equiv_{\#}) \Rightarrow E \in Step(T_m, \equiv_{\#}^C)$  a fortiori  $E \in Step(T_m, \imath)$
- $E \in \Pi_C(T_m / \equiv_{\#}), t \notin E \Rightarrow E \cup \{t\} \notin Step(T_m, \equiv_{\#}^C)$   
*these properties follow from  $\Pi_C$  definition, and from  $\equiv_{\#}^C \subset \imath$*

**Crossed Conflict Step Construction** The final algorithm is derived from the basic one (cf table 1) by considering the following functions:

$$\begin{aligned} T_M(T, \imath) &= \{t \in T \setminus T_{Obs} : t' \# t \Rightarrow t' \in T \setminus T_{Obs}\} \\ T_U(T, \imath) &= T \setminus T_M(T, \imath) \\ \Pi(T_M(T, \imath)) &= \Pi_C((T, \imath) / \equiv_{\#}) \end{aligned}$$

This algorithm produces a FCSG. Conditions  $CA_1, FCA_2$  et  $CB_1$  hold (trivial). Condition  $CB_2$  results from properties of crossed conflict steps.

## 6 Evaluation and Conclusion

A small prototype written in Prolog has been developed for this purpose. All the measurements were made on a Sparc Station with 32 Mbyte of Memory.

The considered example is the Milner's scheduler presented in [Mil 85]. Two kinds of observation are considered, in the first case synchronization actions (or task invitation) are observed. In the second case, the local service associated with each site is considered.

**Observation of Synchronization Actions** ( $O_1 = \{A_i : i \in Sites\}$ ) Fig 4 depicts the obtained FCSG in the case of two sites (all the steps corresponds to the unobservable action  $\tau$ ).

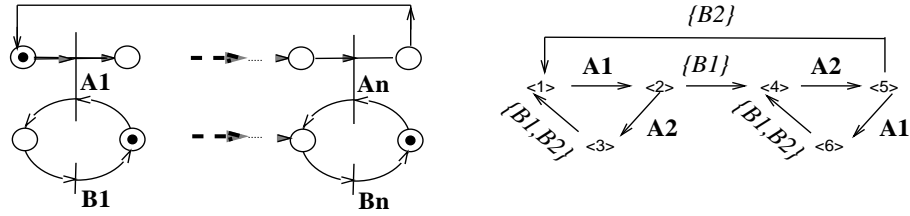


Fig. 4. FCSG associated with the observation of synchronization actions

In this case, the underlying LTS associated with the FCSG is composed of 6 states and 8 arcs while the standard LTS is composed of 8 states and 12 arcs. The obtained FCSG is clearly not minimal (the minimal LTS is a cycle with two states and two arcs).

$n$	2	4	6	12	20
No reduction	8/12	64/160	384/1344	49152/319488	$\approx 2 \cdot 10^7 / 2 \cdot 10^8$
<b>FCSG wrt <math>O_1</math></b>	6/8 1.0s	20/32 1.6s	42/72 3.3s	156/288 21.3s	420/800 116s
Minimal LTS equivalent wrt $O_1$	2/2	4/4	6/6	12/12	20/20
<b>FCSG wrt <math>O_2</math></b>	6/8 1.0s	13/19 1.2s	17/25 1.4s	29/43 2.8s	45/67 7.1s
Minimal LTS equivalent wrt $O_2$	2/2	2/2	2/2	2/2	2/2

Table 2. Experimental Results

Table 2 allows to appreciate the gain realized with the proposed approach when the number of sites increases. Recall that, in the case of  $n$  sites, the stan-

standard LTS is composed of  $n \times 2^n$  states and  $(n^2 + n) \times 2^{n-1}$ . The size of the obtained FCSG seems quadratic with respect the number of sites.

**Observation of the local service ( $O_2 = \{A_i, B_i\}$ )**

Fig 5 presents the obtained FCSG in the case of three sites ( $\{0, 1, 2\}$ , site 1 is observed ( $O_2 = \{A_1, B_1\}$ ). In the case of two sites, the FCSG obtained is identical to the standard LTS. Contrary to the previous case, the minimal LTS, equivalent wrt  $O_2$ , remains constant in size. Table 2 gives the experimental results. In this case, the size of the obtained FCSG seems linear with respect the number of sites.

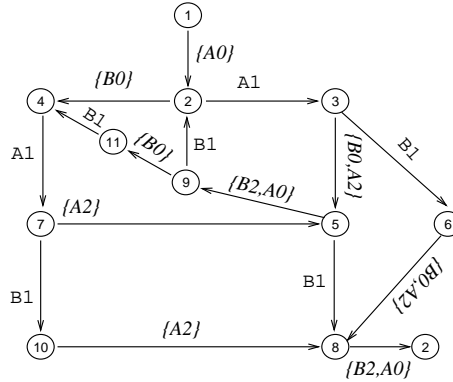


Fig. 5. FCSG associated with the Local Service

**Conclusion and Future works:**

This paper has presented a novel partial-order method to analyze systems with respect to failure semantics. Our contribution is twofold. First, we have identified a family of covering step graphs, the FCSG, preserving the CSP semantics as well as Testing equivalence of studied systems. Then, in an operational point of view, an on-the-fly derivation of such structures has been designed. A first evaluation has shown that the FCSG approach may drastically save time and memory requirements during the enumeration process.

On the other hand, this approach could be extended to failure dedicated structures such as failure [Bri 88] or acceptance trees [Hen 85]. Thereby, this work might also be considered as a first step in the field of conformance testing. Indeed, if the independence relation is shared by a specification and its implementation, it seems obvious to claim that eliminating interleaving in a test suite would not weaken the testing process. Hence, specifying a test case generation based on FCSG could be very interesting in order to reduce the size of complete test suites.

## References

- [Bri 88] E. BRINKSMA *A theory for the derivation of tests*  
In S. Aggrawal and K. Sabani Eds, PSTV, Vol. VIII.  
Elsevier Science Publishers B.V., North Holland, 1988
- [Esp 93] J. ESPARZA *Model checking using net unfoldings*  
In TAPSOFT'93, 1993, LNCS 668
- [FM 90] J. FERNANDEZ, L. MOUNIER  
*Verifying Bisimulation on the Fly*  
3rd Int. Conf on Formal Description Techniques, Madrid, 1990
- [GW 91] P. GODEFROID, P. WOLPER  
*Using partial orders for efficient verification of deadlock freedom  
and safety properties*  
3rd Int. Conf on Computer Aided Verification, 1991, LNCS 575
- [GP 93] P. GODEFROID, D. PIROTIN  
*Refining Dependencies Improves Partial-Order Verification Methods*  
5th Int. Conf on Computer Aided Verification, 1993, LNCS 697
- [Hen 85] M. HENNESSY *Acceptance trees* Journal of the A.C.M Volume 32 1985
- [Jen 87] K. JENSEN *Coloured Petri Nets.*  
In Brauer, W., Reisig, W. & Rozenberg, G. (Ed.): Petri Nets: Central Models  
and their Properties. Advances in Petri Nets LNCS 254
- [McMil 95] K. L. McMILLAN  
*Trace theoretic verification of asynchronous circuits using unfoldings*  
In Computer Aided Verification, 1995, LNCS 939
- [Maz 87] A. MAZURKIEWICZ *Trace Theory*  
In "Petri Nets: Applications and Relationship to other models of concurrency"  
LNCS 255
- [Mil 85] R. MILNER *Communication and Concurrency* Prentice Hall.
- [OH 86] E.R. OLDEROG, C.A. HOARE  
*Specification-Oriented Semantics for Communicating Processes*  
Acta Informatica 23, 1986, pp 9-66
- [PF 90] D. H. PITT, D. FREESTONE  
*The derivation of conformance tests from LOTOS specifications*  
IEEE Transactions on Software Engineering, 16(12), 1990
- [Val 89] A. VALMARI *Stubborn sets for reduced state space generation*  
10 th Int. Conf on Application and Theory of Petri Nets, Bonn, 1989, LNCS 483
- [VAM 96] F. VERNADAT, P. AZÉMA, F. MICHEL *Covering Step Graphs*  
17 th Int. Conf on Application and Theory of Petri Nets 96, June 24-28 1996,  
Osaka - Japan, LNCS 1091
- [WG 93] P. WOLPER, P. GODEFROID *Partial Order Methods for Temporal Verification*  
Proceedings of CONCUR'93, LNCS 715