Acta Informatica 17, 21–29 (1982)

Acta Informatica

Synchronized Behaviours of Processes and Rational Relations

A. Arnold

Laboratoire d'Informatique, Université de Poitiers, 40 Avenue du Recteur Pineau, F-86022 Poitiers Cedex, France

Summary. Synchronized behaviours of processes, specified in three different ways, are compared with rational relations.

Introduction

A system of concurrent processes is a set of processes, the behaviour of each one depends on the behaviours of the others, either because some actions of some processes cannot be done simultaneously, – for example actions using a same unshareable resource –, or because some actions have to be done in some temporal order, – for example putting then removing an information in a buffer –.

Clearly the specification of a system of concurrent processes can be divided in two parts:

(i) specification of all the possible behaviours of each individual processes;

(ii) specification of the interference between the behaviours of these individual processes.

A way to realize this second point is to design a mechanism such that the processes are constrained to behave as requested. Since the main task of such a mechanism is to control the temporal order of execution of actions of each process, it is named a synchronization mechanism. The well-known semaphores are an example of such a synchronization mechanism [4].

But a synchronisation mechanism is rather the implementation of synchronization constraints. The specification of these constraints must be done on a more abstract – or less implementation-dependent – level.

Following Campbell and Habermann [3], Lauer, Torrigiani and Shields [6] have defined a system of specification based upon the notion of path. A formal semantics for this system of specification, i.e. for COSY programs, has been defined by Shields and Lauer [10] in terms of vector firing sequences.

Similarly, Nivat [8] has defined the set of S-synchronized behaviours of a set of processes in a way which looks very close to the definition of the

0001-5903/82/0017/0021/$01.80

semantics of COSY programs given by Shields and Lauer. Moreover Nivat introduces in this paper the notion of finite-state control of a set of processes which includes the notion of S-synchronization. Finite-state controls are a special case of controls of systems of processes defined by Arnold and Nivat [2] but seem to have nearly the same expressive power as COSY programs.

Thus we have three ways for specifying synchronization constraints which are presumably very close each to the others. The aim if this paper is to compare precisely their expressive power, when the last two ones are applied to rational processes, which is always the case for COSY programs.

We prove that the three classes *Synchro*, *Control*, *Cosy* of synchronized behaviours mentioned above are related together by $Synchro \subset Control = Sel_{-1}(Synchro)$, $Cosy \subset Synchro$, $Proj(Sel_{-1}(Cosy)) = Control$ and moreover, that they are also related to the subclass *PcRat* of rational relations recognized by sink-automata by $PcRat = Control$ where Sel_{-1} is the class of projections of n-any relations on their first $n-1$ components, and *Proj* is the class of products of strict alphabetic homomorphisms.

These results are not difficult to prove. They just show that the theoretical framework of rational relations is well suited to give a precise meaning to the intuitive notion of "similar" synchronization mechanisms. More precisely, the fact that synchronized behaviours are subclasses of the class of relational relations suggests that a study of these subclasses can be of some help to deal with synchronization problems.

In this paper we have not considered infinite behaviours of processes [8]. We believe that similar results can be obtained in the same way, using the notion of infinitary relation on infinite words introduced by Nivat [9].

The first part of this paper is just some recalls about rational languages, finite-state automata and rational relations. In the second one the specifications of synchronization we are studying are defined. These specifications are compared each other and are compared to rational relations.

This paper is an improved version of a lecture given at the Spring School on Petri Nets and Concurrency (Colleville, 1980) [1]. This new version has profited by stimulating discussions with M. Nivat.

1. Rational Languages and Rational Relations

Here we recall some well known facts about rational sets.

Let A be a finite alphabet.

An *automaton* $\mathscr{A}$ on A is a tuple $\langle A, Q, q_0, Q_F, \delta \rangle$

where Q is a finite set of *states*,
q_0 is an element of Q, the *initial* state,
Q_F is a subset of Q, the set of *final states*,
δ, the *transition function*, is a mapping from $Q \times A$ into Q.

The transition function δ is easily extended into a mapping, still noticed by δ, from $Q \times A^*$ into Q. The language $L \subset A^*$ recognized by is the set $L(\mathscr{A}) = \{u \in A^* \mid \delta(q_0, u) \in Q_F\}$.

A language L is *rational* (or regular) iff it is recognized by some automaton.

We say that a state q of an automaton $\mathscr{A}$ is a *sink-state* if $\forall a \in A$, $\delta(q, a) = q$.

An automaton $\mathscr{A}$ is a *sink automaton* if Q contains one sink-state q_s and $Q_F = Q - \{q_s\}$.

We say that a rational language is *prefix-closed* (*pc* for short) if

$$\mathrm{Init}(L) = L,$$

where $\mathrm{Init}(L) = \{u \in A^* \mid \exists v \in A^* : uv \in L\}$.

Let A and B be two alphabets. A homomorphism $\phi: A^* \to B^*$ is *alphabetic* if $\phi(A) \subset B \cup \{\Lambda\}$ and *strict alphabetic* (*sa* for short) if $\phi(A) \subset B$.

The following results are obvious

Proposition 1.1. *A language L is pc rational iff it is recognized by some sink automaton. The intersection of two pc rational languages is a pc rational language.*

Let $\phi': A^* \to B^*$ and $\phi: B^* \to A^*$ be alphabetic homomorphisms. If $L \subset A^*$ is pc rational then $\phi'(L)$ and $\phi^{-1}(L)$ are pc rational.

Let $A_1, \ldots, A_n$ be alphabets and let $\mathbf{A} = \bigtimes_{i=1}^{n} (A_i \cup \{\Lambda\}) - \{\langle \Lambda, \ldots, \Lambda \rangle\}$.

The class $\mathrm{Rat}(A_1, \ldots, A_n)$ of *rational relations* included in $\bigtimes_{i=1}^{n} A_i$ is the least class containing finite relations and closed under union, product and star.

A relation recognized by a finite automaton $\mathscr{A}$ on $\mathbf{A}$ is the set of images of words of $\mathbf{A}$ recognized by $\mathscr{A}$ under the canonical mapping $\pi: \mathbf{A}^* \to \bigtimes_{i=1}^{n} A_i^*$, where $\pi = \langle \pi_1, \ldots, \pi_n \rangle$ with $\pi_i: \mathbf{A}^* \to A_i^*$.

A *rational multimorphism* on $A_1, \ldots, A_n$ is a $(n+1)$-uple $\mu = \langle \phi_1, \ldots, \phi_n, L \rangle$ where L is a rational language on some alphabet A_{n+1} and $\phi_i: A_{n+1}^* \to A_i^*$ is an alphabetic homomorphism. We say that a rational multimorphism is *strict* if $\forall a \in A_{n+1}$, $\exists i \in \{1, \ldots, n\}: \phi_i(a) \neq \Lambda$. Hence we can associate with a (strict) rational multimorphism a canonical (strict) alphabetic homomorphism $\bar{\mu}: A_{n+1}^* \to \mathbf{A}^*$ defined by

$$\bar{\mu}(a) = \begin{cases} \langle \phi_1(a), \ldots, \phi_n(a) \rangle & \text{if } \exists i: \phi_i(a) \neq \Lambda \\ \Lambda & \text{otherwise.} \end{cases}$$

The relation $\hat{\mu}$ represented by a rational multimorphism μ is $\pi(\bar{\mu}(L))$, i.e. $\hat{\mu} = \{\langle \phi_1(u), \ldots, \phi_n(u) \rangle / u \in L\}$.

Let *Rat* be the class of all rational relations. The following characterizations of *Rat* are well known [5, 7].

Proposition 1.2. *A relation R is rational*

iff (i) it can be recognized by a finite automaton.
iff (ii) it can be represented by a rational multimorphism.
iff (iii) it can be represented by a strict rational multimorphism.

If $\psi = \langle \psi_1, \ldots, \psi_n \rangle$ is a vector of *sa* homomorphisms where $\psi_i: A_i^* \to B_i^*$ and R a relation in $A_1^* \times A_2^* \ldots \times A_n^*$ then $\psi(R)$ is the relation $\{\langle \psi_1(u_1), \ldots, \psi_n(u_n) \rangle / \langle u_1, \ldots, u_n \rangle \in R\}$ in $B_1^* \times \ldots \times B_n^*$.

The vector ψ is a *projection* and *Proj* is the class of projections. For a class $\mathscr{C}$ of relations, we denote by $Proj(\mathscr{C})$ the class of all the images of relations in $\mathscr{C}$ under projections.

For a relation R in $A_1^* \times \ldots \times A_n^*$ let $\mathrm{sel}_{-1}(R)$ be the relation

$$\{\langle u_1, \ldots, u_{n-1}\rangle / \exists\, u_n : \langle u_1, \ldots, u_{n-1}, u_n\rangle \in R\}$$

and let $Sel_{-1}(\mathscr{C}) = \{\mathrm{sel}_{-1}(R) / R \in \mathscr{C}\}$.

A relation $R \subset \bigtimes_{i=1}^{n} A_i^*$ is said to be *recognizable* if it is a finite union of sets in the form $\bigtimes_{i=1}^{n} L_i$ where L_i is a rational language on A_i. It is well known (cf. [5]) that

Proposition 1.3. *The intersection of a rational relation with a recognizable relation is a rational relation.*

$$Proj(Rat) = Sel_{-1}(Rat) = Rat.$$

Let us define now the subclass *PcRat* of *Rat* as being the class of relations represented by a strict rational multimorphism $\langle \phi_1, \ldots \phi_n, L\rangle$ such that L is a *pc* rational language. It is clear that *PcRat* is also the class of relations recognized by sink-automata.

Similarly we say that a relation $R \subset \bigtimes_{i=1}^{n} A_i^*$ is *pc recognizable* if it is a finite union of sets in the form $\bigtimes_{i=1}^{n} L_i$ where L_i is a *pc* rational language on A_i.

From these definitions we get obviously

Proposition 1.4. *The intersection of a pc rational relation with a pc recognizable relation is a pc rational relation.*

$$Proj(PcRat) = Sel_{-1}(PcRat) = PcRat.$$

The motivation for introducing *pc* rational languages, *pc* rational relations and *pc* recognizable relations is that, as a language, a process is prefix-closed and thus, synchronized behaviours of processes become *pc* relations, as we shall see just now.

2. Synchronized Behaviours of Processes

Let A be a finite set of *actions*. A process P on A performs a sequence of actions in A. The set of behaviours $B(P)$ of A is the set of all these sequences, i.e. a language on A. Moreover, since every initial subsequence of a sequence in $B(P)$ is also performed by P, we get

$$\mathrm{Init}(B(P)) = B(P).$$

Hence we say that a process P on A is *rational* if $B(P)$ is a *pc* rational language.

Conversely we can assume that every *pc* rational language $L \subset A^*$ is equal to $B(P)$ for some rational process P on A.

Let us define now a *synchronized system* Q of processes as beeing a set $\{P_1, \ldots, P_n\}$ of processes, where P_i is a process on A_i, with some specification of synchronization constraints. Then the set of *synchronized behaviours* of this system (or the set of *vector firing sequences*) is a subset

$$B(Q) \subset \bigtimes_{i=1}^{n} B(P_i) \subset \bigtimes_{i=1}^{n} A_i^*.$$

From now on, let $\mathbf{P} = \langle P_1, \ldots, P_n \rangle$ be a vector of rational processes, where P_i is a process on A_i.

2.1. S-synchronization [8]. Let $S \subset \mathbf{A}$ be the set of vectors of actions which can be performed simultaneously in a system - if a component of an element s of S, say the i^{th}, is the empty word, it means that the i^{th} process must wait -.

The set of S-synchronized behaviours of this system is the set

$$B(\mathbf{P}, S) = \left(\bigtimes_{i=1}^{n} B(P_i) \right) \cap \pi(S^*).$$

Let *Synchro* be the class of all $B(\mathbf{P}, S)$.

Since $\pi(S^*)$ is obviously a *pc* rational relation we get, from Proposition 1.4 that $B(\mathbf{P}, S)$ is still a *pc* rational relation hence $Synchro \subset PcRat$. Moreover we can prove a kind of converse inclusion

Proposition 2.1. $Synchro \subset PcRat = Sel_{-1}(Synchro)$.

Proof. Since $Synchro \subset PcRat$, we have $Sel_{-1}(Synchro) \subset Sel_{-1}(PcRat) = PcRat$; let us now prove that $PcRat \subset Sel_{-1}(Synchro)$.

Let $R \subset A_1^* \times A_2^* \ldots \times A_n^*$ be a *pc* rational relation. Hence from definitions, $R = \pi(L)$ where L is a *pc* rational language in $\mathbf{A}^*$. We construct the system of processes $Q = \langle P_1, \ldots, P_n, P_{n+1} \rangle$ where

$$B(P_i) = \begin{cases} A_i^* & \text{if } 1 \leqq i \leqq n \\ L & \text{if } i = n+1 \end{cases}$$

and we synchronize this system by $S \subset \mathbf{B} = (A_1 \cup A) \times \ldots \times (A_n \cup A) \times \mathbf{A} - A^{n+1}$, so that if P_{n+1} executes $\mathbf{a} \in \mathbf{A}$, each process P_i simultaneously executes $\pi_i(\mathbf{a})$ i.e. $S = \{\gamma(\mathbf{a})/\mathbf{a} \in A\}$ where $\gamma(\mathbf{a}) = \langle \pi_1(\mathbf{a}), \ldots, \pi_n(\mathbf{a}), \mathbf{a} \rangle$.

Let $\boldsymbol{\lambda} = \langle \lambda_1, \ldots, \lambda_n, \lambda_{n+1} \rangle$ with for $1 \leqq i \leqq n$,

$$\lambda_i : \quad \mathbf{B}^* \to A_i^*$$
$$\lambda_{n+1} : \mathbf{B}^* \to \mathbf{A}^*.$$

Obviously we have for $1 \leqq i \leqq n$, $\lambda_i = \pi_i \cdot \lambda_{n+1}$

$$\text{and for } u \in \mathbf{A}^* \quad \lambda_{n+1}(\gamma(u)) = u$$
$$\text{and for } u \in \mathbf{B}^* \quad \gamma(\lambda_{n+1}(u)) = u.$$

It follows that $\langle u_1, \ldots, u_n\rangle \in \mathrm{sel}_{-1}(B(\mathbf{P}, S))$ iff

$$\exists v \in S^*: u_i = \lambda_i(v) \text{ and } \lambda_{n+1}(v) \in L \text{ iff}$$
$$\exists u_{n+1} \in L: u_i = \lambda_i(\gamma(u_{n+1})) \text{ iff}$$
$$\exists u_{n+1} \in L: u_i = \pi_i(u_{n+1}) \text{ iff}$$
$$\langle u_1, \ldots, u_n\rangle \in R,$$

since $u_{n+1} = \lambda_{n+1}(v)$ iff $v = \gamma(u_{n+1})$ and $\pi_i(\gamma(u_{n+1})) = \pi_i(\lambda_{n+1}(\gamma(u_{n+1}))) = \pi_i(u_{n+1})$. □

2.2. Finite state controls [8, 2]. At every step of the evolution of a controlled system, the control allows a vector of actions to be fired depending on its own state, and changes its state depending on the vector just fired. Let us give a formal definition of this.

A finite state control M of the vector $\mathbf{P}$ of processes is a tuple $\langle Q, q^*, \delta, \psi\rangle$ where

Q is a finite set of *states*,
q^* is an element of Q, the *initial state*,
$\delta: Q \to \mathscr{P}(\mathbf{A})$,
$\psi: Q \times \mathbf{A} \to \mathscr{P}(Q)$.

The set of synchronized behaviours of this system is $B(\mathbf{P}, M)$ defined by $u \in B(\mathbf{P}, M)$ iff

i) $u \in \bigtimes_{i=1}^{n} B(P_i)$ and
ii) $\exists p \geqq 0, \exists \mathbf{a}_1, \ldots, \mathbf{a}_p \in \mathbf{A}, \exists q_0, \ldots, q_p \in Q$ such that
- $u = \mathbf{a}_1 \cdot \mathbf{a}_2 \cdot \ldots \cdot \mathbf{a}_p$,
- $q_0 = q^*$ and
- $\forall i \in \{1, \ldots, p\}$, $\mathbf{a}_i \in \delta(q_{i-1})$ and $q_i \in \psi(q_{i-1}, \mathbf{a}_i)$.

In other words $B(\mathbf{P}, M) = (B(P_1) \times \ldots B(P_n)) \cap \pi(L)$ where L is the rational language in $\mathbf{A}^*$ recognized by M, and it is clear from the definition (cf. ii) above) of M that L is prefix-closed.

Let us denote by *Control* the class of all $B(\mathbf{P}, M)$.

From Proposition 1.4 we get $Control \subset PcRat$. Conversely let $R = \pi(L)$ in PcRat. Then $R = (A_1^* \times \ldots \times A_n^*) \cap \pi(L)$, which is in *Control*. Therefore

Proposition 2.2. *PcRat = Control*

2.3. COSY programs [6, 10]. A COSY program is a vector of *cyclic* processes $\mathbf{P} = \langle P_1, \ldots, P_n\rangle$ and another vector of cyclic processes $\mathbf{Q} = \langle Q_1, \ldots, Q_n\rangle$ named *paths*.

A cyclic process is a rational process, which have as a set of behaviours, $\mathrm{Init}(L^*)$ where L is a rational language.

The definition of the set of synchronized behaviours of a COSY program is given in [10]. Let us recall this definition here.

Let $\langle \mathbf{P}, \mathbf{Q} \rangle$ be a COSY program with $\mathbf{P} = \langle P_1, \ldots, P_n \rangle$, $\mathbf{Q} = \langle P_{n+1}, \ldots, P_{n+m} \rangle$ and P_i is a cyclic rational process on A_i.

For any $i \in \{1, \ldots, n+m\}$, let $\sigma_i = A_i^* \to (\{i\} \times A_i)$ and $\tau_i : (\{1, \ldots, n\} \times A_i)^* \to A_i^*$ be the *sa* homomorphisms defined by $\sigma_i(a) = \langle i, a \rangle$ and $\tau_i(j, a) = a$.

Let us set $L_i = \sigma_i(B(P_i))$ for $i \in \{1, \ldots, n\}$, and $L_i = \tau_i^{-1}(B(P_i))$ for $i \in \{n+1, \ldots, m\}$.

At last let S be the subset of $\mathbf{B}$ defined by

$$\langle s_1, \ldots, s_{n+m} \rangle \in S \text{ iff}$$

$a \in \bigcup_{i=1}^{n+m} A_i$, $j \in \{1, \ldots, n\}$ such that

$$s_k = \begin{cases} \langle j, a \rangle & \text{if } \langle j, a \rangle \in B_k \\ \Lambda & \text{otherwise.} \end{cases}$$

The set of behaviours $B(\mathbf{P}, \mathbf{Q})$ of this program is $\tau\left(\left(\bigtimes_{i=1}^{n+m} L_i\right) \cap \pi(S^*)\right)$ where τ is $\langle \tau_1, \ldots, \tau_{n+m} \rangle$.

The previous definition of S amounts to say that if an action can be performed by a set I of processes and a set J of paths, then it must be performed simultaneously by only one process in I and by all paths in J. Thus we can define a subset T of $\mathbf{A}$ by $\langle b_1, \ldots, b_{n+m} \rangle \in T$ iff $a \in \bigcup_{i=1}^{n+m} A_i$, $\exists j \in \{1, \ldots, n\}$ such that

$$b_k = \begin{cases} a & \text{if } a \in A_k \text{ and } (k = j \text{ or } n+1 \leqq k \leqq n+m) \\ \Lambda & \text{otherwise.} \end{cases}$$

And then $B(\mathbf{P}, \mathbf{Q}) = \left(\bigtimes_{i=1}^{n+m} B(P_i)\right) \cap \pi(T^*)$. Whence it follows that $Cosy \subset Synchro$.

The difference between the synchronization mechanisms for *Cosy* and *Synchro* is twofold. For *Synchro* simultaneous actions can have different names (i.e. a vector in S can have different components) while for *Cosy* these actions must be the same (a vector in T has the same non-empty components). Thus this difference is only a matter of names of actions and then *Cosy* and *Synchro* can be easily related by *Proj*. The second difference is that in *COSY* processes are always cyclic. This difference is not important because of the following "trick".

Let L any *pc* rational language and let c a new letter not occuring in L. Then $L' = \mathrm{Init}((Lc)^*)$ is a cyclic process, but the set of words in L' in which c does not occur is exactly L.

So we can prove

Proposition 2.3. *$Cosy \subset Synchro$;*

$$Proj(Sel_{\_1}(Cosy)) = PcRat.$$

Proof. To obtain this result we have just to prove

$$PcRat \subset Proj(Sel_{\_1}(Cosy)).$$

Let $R=\pi(L)$ a *pc* rational relation where $L\subset \mathbf{A}^*$. For every $i\leqq n$, let $\mathbf{A}_i=\{\mathbf{a}\in\mathbf{A}/\pi_i(\mathbf{a})\neq\Lambda\}\cup\{c\}$ where c is a new letter and $\mathbf{A}_{k+1}=\mathbf{A}\cup\{c\}$, and let ϕ_i the canonical alphabetic homomorphism from $\mathbf{A}^*\to\mathbf{A}_i^*$. Let $\mathbf{B}=\bigtimes_{i=1}^{n+1}(\mathbf{A}_i\cup\Lambda)-\Lambda^{n+1}$ and for every $\mathbf{a}\in\mathbf{A}$ let $\gamma(\mathbf{a})\in\mathbf{B}$ be defined by $\gamma(\mathbf{a})_i=\phi_i(\mathbf{a})$. Let us consider the cyclic paths P_i such that for $1\leqq i\leqq n$, $B(P_i)=(\mathbf{A}_i-\{c\})^*$ and $B(P_{n+1})=\mathrm{Init}((Lc)^*)$. The synchronized behaviour of this COSY-program is $R'=\bigtimes_{i=1}^{n+1}B(P_i)\cap\lambda(T^*))$ where $\lambda:\mathbf{B}^*\to\mathbf{A}_1^*\times\ldots\times\mathbf{A}_{n+1}^*$ and

$$T=\{\delta(a)/a\in\bigcup_{i=1}^{n+1}\mathbf{A}_i\}\quad\text{with}\quad\delta(a)_i=\begin{cases}a & \text{if } a\in\mathbf{A}_i\\ \wedge & \text{if } a\notin\mathbf{A}_i\end{cases}.$$

Hence $T=\{\gamma(\mathbf{a})/\mathbf{a}\in\mathbf{A}\}\cup\{\langle c,\ldots,c\rangle\}$.

But since $B(P_i)$ does not contain c for $i=1,\ldots,n$, we have $R'=\bigtimes_{i=1}^{n+1}B(P_i)\cap\lambda(\gamma(\mathbf{A}))$.

It follows that $\langle u_1,\ldots,u_{n+1}\rangle\in R'$ iff $\exists v\in(\gamma(\mathbf{A}))^*:\lambda_i(v)\in(\mathbf{A}_i-\{c\})^*$ for $1\leqq i\leqq n$ and $\lambda_{n+1}(v)\in L$. But $\lambda_i(v)=\phi_i(\lambda_{n+1}(v))$ and $\lambda_{n+1}(v)=u_{n+1}$ iff $v=\gamma(u_{n+1})$, therefore $\langle u_1,\ldots,u_{n+1}\rangle\in R'$ iff $u_{n+1}\in L$ and $u_i=\phi_i(u_{n+1})$. Then $\mathrm{sel}_{-1}(R')=\{\langle u_1,\ldots,u_n\rangle/\exists u_{n+1}\in L: u_i=\phi_i(u_{n+1})\}$ and since the restriction of π_i to $\mathbf{A}_i-\{c\}$ is a strict homomorphism which satisfies $\pi_i\phi_i(u_{n+1})=\pi_i(u_{n+1})$ we get $\pi(\mathrm{sel}_{-1}(R'))=\{\langle u_1',\ldots,u_n'\rangle/\exists u_{n+1}\in L;\ u_i'\in\pi_i/u_{n+1})\}=R$. Hence $R\in Proj(Sel_{-1}(Cosy))$. □

2.4. Conclusions. Now we can collect together the previous results:

$$Cosy\subset Synchro\subset Control=PcRat$$

$$PcRat=Sel_{-1}(Synchro)=Proj(Sel_{-1}(Cosy))$$

which makes appear some open questions.

– Are the inclusions in the first line strict? We think they are.

– *Proj(Synchro)* and *Proj(Cosy)* and $Sel_{-1}(Cosy)$ are obviously included in *PcRat;* are these inclusions strict? We think they are.

– Does there exist some relationship between *Proj(Cosy)* and *Synchro*?

In the definition of *Cosy* we met the condition that processes have to be cyclic. If we give up this condition we will get the class *Excosy* which obviously satisfies $Cosy\subset Excosy\subset Synchro$.

– Then does there exist some relationship between *Proj(Excosy)* and *Synchro*? between $Sel_{-1}(Excosy)$ and others classes?

Moreover it is possible to consider the class *Exproj* of projections where alphabetic homomorphisms are not necessarily strict. Then we have

Proposition 2.4. $PcRat=Exproj(Synchro)=Exproj(Excosy)$.

Proof. Since *Exproj(PcRat)* included in *PcRat* it is sufficient to prove that $PcRat\subset Exproj(Excosy)$. Let $R=\pi(L)$. Let us define $\mathbf{B}=\{\langle\mathbf{a},\ldots,\mathbf{a}\rangle/\mathbf{a}\in\mathbf{A}\}$. Ob-

viously $R' = (L \times L \ldots \times L) \cap \lambda(\mathbf{B}^*)$ is the behaviour of an Excosy program and $\langle u_1, \ldots, u_n \rangle \in R'$ iff $u_1 = u_2 = \ldots = u_n \in L$, hence

$$\pi(R') = \{\langle \pi_1(u_1), \ldots, \pi_n(u_n) \rangle / \langle u_1, \ldots, u_n \rangle \in R'\}$$
$$= \{\langle \pi, (u), \ldots, \pi_n(u) \rangle / u \in L\} = R. \quad \square$$

At last it is also possible to give up the condition that processes are prefix-closed languages and we will get other classes of synchronized behaviours which will have to be compared to the class *Rat* of rational relations.

Thus there exist many subclasses of the class *PcRat* (and of the class *Rat*) which correspond to behaviours of synchronized systems of processes and relationship between these subsclasses should be of interest for studying synchronization problems.

References

1. Arnold, A.: A comparison of three specifications of synchronization constraints. Rapport de Recherche n° 5, Laboratoire d'Informatique, Université de Poitiers (1980)
2. Arnold, A., Nivat, M.: Controlling behaviours of systems: Some basic concepts and some applications. In: 9th Symposium on Mathematical Foundations of Computer Science, Rydzyna 1980 (P. Dembinski, ed.). Lecture Notes in Computer Science, Vol. 88, pp. 113–122. Springer-Verlag (1980)
3. Campbell, R.M., Habermann, A.N.: The specification of process synchronization by path expressions. In: Operating Systems, Int. Symp. Rocquencourt, 1974 (E. Gelenbe, C. Kaiser, eds.). Lecture Notes in Computer Science, Vol. 16, pp. 89–102. Springer-Verlag (1974)
4. Dijkstra, E.W.: Co-operating sequential processes. In: Programming Languages (F. Genuys, ed.), Academic Press, New York 1967
5. Eilenberg, S.: Automata, Languages and Machines, Vol. A. Academic Press, New York 1974
6. Lauer, P.E., Torrigiani, P.R., Shields, M.W.: COSY. A system specification language based on paths and processes. Acta Informat. **12**, 109–158 (1979)
7. Nivat, M.: Transductions des languages de Chomsky. Ann. Inst. Fourier, Grenoble **18**, pp. 339–456 (1968)
8. Nivat, M.: Sur la synchronisation des processus. Revue Technique Thomson-CSF **11**, 889–919 (1979)
9. Nivat, M.: Infinitary relations. In: CAAP 81, Genoa, 1981 (E. Astesiano, C. Böhm, eds.). Lecture Notes in Computer Science, Vol. 112, pp. 46–75. Springer-Verlag (1981)
10. Shields, M.W., Lauer, P.E.: A formal semantics for concurrent systems. In: Automata, languages and programming, 6th Colloquium, Graz (H.A. Maureer, ed.). Lecture Notes in Computer Science, Vol. 71, pp. 571–584. Springer-Verlag (1979)

Received February 17, 1981 / January 21, 1982