# A RELATIONAL MODEL FOR NON-DETERMINISTIC PROGRAMS AND PREDICATE TRANSFORMERS*

Pedro Guerreiro[1]

Abstract :

A relational model for non-deterministic programs is presented. Several predicate transformers are introduced and it is shown that one of them satisfies all the healthiness criteria indicated by Dijkstra for a useful total correctness predicate transformer.

An axiomatic relational definition of the language of guarded commands is proposed. From it the predicate transformers associated to each command in the language are derived. The fact that Dijkstra's axioms are rediscovered proves their consistency in the model.

[1] IMAG, Grenoble (France) and the New University of Lisbon, (Portugal).
Address : IMAG, B.P. 53 X, 38041 GRENOBLE Cédex,
FRANCE

* Research reported herein was supported in part by the Calouste Gulbenkian Foundation, Lisbon, under grant number 14/78/B.

## 1. INTRODUCTION

The goal of this paper is to present a model for non-deterministic programs, well adapted to the formalization and the analysis of several usual concepts in the fields of the semantic definition and of the verification of properties, (e.g. predicate transformer, non-determinacy, invariant, termination). The proposed model is based on a special class of binary relations. It allows us to introduce, in a natural fashion some predicate transformers that capture different aspects of our intuition about the behaviour of non-deterministic programs. We show that one of these predicate transformers has all the properties indicated by Dijkstra [Dij76] for a useful total correctness predicate transformer. Then we axiomatically define the semantics of the language of guarded commands [Dij76] in terms of relations of the mentioned class. From these relations we derive the predicate transformers associated with each command in the language, and, as a side-effect, prove the consistency of Dijkstra's axioms in the model.

A different model where the consistency of Dijkstra's axioms was established is due to Hoare [Hoa78a] and uses traces. Other characterizations of Dijkstra's wp function have been given by de Roever [Roe76], and by Wand [Wan77]. Our approach to predicate transformers was suggested by recent work by Sifakis [Sif79]. It is therefore natural that certain points treated in this paper have counterparts in the works of those authors.

## 2. THE BASE MODEL

The behaviour of a "deterministic" program is usually defined by a function from a set of inputs to a set of outputs, or, more abstractly, from a suitable set of states into itself. By analogy, we expect that the behaviour of a "non-deterministic" program can be described by a binary relation over such a set of states.

Let Q be a set (of "states") and R a binary relation over Q, $R \subseteq Q \times Q$. Let $\mathfrak{R}$ denote the set of all these relations. R can also be seen as a function of $Q \to 2^Q$ : $R(x) = \{y \mid (x,y) \in R\}$. Thus, it is indifferent to write $(x,y) \in R$ or $y \in R(x)$. Let $\pi$ be a program whose state space is Q. If R is to represent $\pi$'s behaviour, it must be such that $(x,y) \in R$, if and only if it is possible that an execution of $\pi$ starting at state x terminates at state y.

Let X be the set of initial states for $\pi$. On the hypothesis just made the corresponding set of possible final states is

(1) $R(X) = \bigcup_{x \in X} R(x) = \{y \mid \exists x \; x \in X \wedge (x,y) \in R\}$

R(X) is called the "image of X by R".

Properties 1.[1]

1.1. $R(\emptyset) = \emptyset$

[1] R and S are binary relations, F is any family of indices.

1.2. $R(\bigcup_{i\in F} X_i) = \bigcup_{i\in F} R(X_i)$

1.3. $(R\circ S)(X) = S(R(X))$

Let $\mathcal{P}$ be the set of total predicates over Q, $\mathcal{P} = Q\to\{tt,ff\}$. $\mathcal{P}$ is a complete lattice for the order $\subseteq$ defined by $A\subseteq B$ if $\forall x\ A(x)\Rightarrow B(x)$, isomorphic to the lattice of the subsets of Q with normal set inclusion. As usual we define, for $A, B\in\mathcal{P}$:

1) $A\cup B = \lambda x.A(x)\vee B(x)$ ; 2) $A\cap B = \lambda x.A(x)\wedge B(x)$ ;[1]

3) $\neg A = \lambda x.\neg A(x)$. Two constants are used : $\top = \lambda x.tt$, $\bot = \lambda x.ff$. (We also abbreviate $A\cap\neg B$ by $A-B$ and $\neg A\cup B$ by $A\Rightarrow B$).

A function of $\mathcal{P}\to\mathcal{P}$ is called a "predicate transformer", (pt). One such function is obtained by rewriting (1) in terms of predicates. We call it image [R] :

(2) $\text{image}[R] = \lambda P\lambda y.\exists x\ P(x)\wedge(x,y)\in R$

Remark that the symbol image denotes a function of $\mathcal{R}\to(\mathcal{P}\to\mathcal{P})$.

Properties 2.

2.1. $\text{image}[R](\bot) = \bot$

2.2. $\text{image}[R](\bigcup_{i\in F} P_i) = \bigcup_{i\in F} \text{image}[R](P_i)$

2.3. $\text{image}[R\circ S](P) = \text{image}[S](\text{image}[R](P))$

Thus, image[R](P) is the predicate characterizing the set of possible final states for $\pi$ when activation is known to be in a state verifying P. We see that image[R] works "forward". Usually, "backwards" pt's are preferred. This leads us to consider the pt $\text{image}[R^{-1}]$, which we call pre[R].

(3) $\text{pre}[R] = \text{image}[R^{-1}] = \lambda P\lambda x.\exists y\ (x,y)\in R\wedge P(y)$

Properties 3.

3.1. $\text{pre}[R](\bot) = \bot$

3.2. $\text{pre}[R](\bigcup_{i\in F} P_i) = \bigcup_{i\in F} \text{pre}[R](P_i)$

3.3. $\text{pre}[R\circ S](P) = \text{pre}[R](\text{pre}[S](P))$

The interpretation of pre[R] is symmetrical to that of image[R] : if P represents a set of final states, pre[R](P) describes the set of those initial states from which it is possible that the computation reaches a final state in P. In other words, if the activation takes place outside pre[R](P), the execution

---

[1] The operations $\cup$ and $\cap$ are easily generalized to an infinite number of operands.

cannot terminate in P. This "double negation" suggests a third pt, dual[1] of pre[R] :

(4) $\tilde{pre}[R] = \lambda P.\neg pre[R](\neg P) = \lambda P \lambda x.\forall y\ (x,y) \in R \Rightarrow P(y)$

Properties 4.

4.1. $\tilde{pre}[R](\top) = \top$

4.2. $\tilde{pre}[R](\bigcap_{i \in F} P_i) = \bigcap_{i \in F} \tilde{pre}[R](P_i)$

4.3. $\tilde{pre}[R \circ S](P) = \tilde{pre}[R](\tilde{pre}[S](P))$

When the activation takes place in $\tilde{pre}[R](P)$, the termination cannot occur outside P, i.e., if the program does terminate the final state must be one of P. This means that $\tilde{pre}[R]$ acts like a partial correctness pt.

For a total correctness pt, consider the following :

(5) $\hat{pre}[R] = \lambda P.pre[R](P) \cap \tilde{pre}[R](P) = \lambda P \lambda x.\exists y\ (x,y) \in R \wedge \forall y\ (x,y) \in R \Rightarrow P(y)$

Supposing that whenever activation in state x may lead to a non-terminating computation R(x) is empty[2], we can in fact conclude from (5) that if the program starts in $\hat{pre}[R](P)$ it must terminate in P. (Note, however, that in general $\hat{pre}[R \circ S](P)$ is different from $\hat{pre}[R](\hat{pre}[S](P))$, a somewhat surprising fact, on account of properties 3.3. and 4.3. ; example : Q = {0,1}, R = {(0,0),(0,1)}, S = {(1,1)}).

## 3. THE EXPLICIT REPRESENTATION OF NON-TERMINATION

The hypothesis just made contradicts the one presented in the beginning of section 2. and implies a certain loss of information about the behaviour of the program. In fact, there are situations where, because of an arbitrary choice during execution, a program may terminate or not. Under the second hypothesis the possibility of termination is not recorded. Under the first one it is the possibility of non-termination that cannot be deduced from the relation. A wayout of this dilemma, taken also by de Roever [Roe76], is the introduction in Q of a special element, which we denote $\omega$, to represent the "final" state of the non-terminating computations. Thus, $(x,y) \in R$ with $y \neq \omega$, has the same meaning as before, while $(x,\omega) \in R$ indicates a non-terminating computation starting at x.

With this new interpretation it must be imposed that no state (other than $\omega$), may be reached "after" a non-terminating computation. Besides, it may be considered that every computation has a final state (maybe $\omega$). In this way we are res-

(1) The dual of a pt f is another pt, denoted $\tilde{f}$, s.t. $\tilde{f}(P) = \neg f(\neg P)$

(2) With this hypothesis $\hat{pre}[R]$ corresponds to the pt studied in [Wan77].

tricting the class of relations used to describe the behaviour of programs to those verifying the following conditions :

C1. $R(\omega) = \{\omega\}$

C2. $\forall x \quad R(x) \neq \emptyset$ (This is equivalent to $pre[R](\top)=\top$ and to $\tilde{pre}[R](\bot)=\bot$).

Such relations are called ω-relations.

## 4. A TOTAL CORRECTNESS PREDICATE TRANSFORMER

Let Q be a set of states (s.t. $\omega \in Q$), and $\mathfrak{R}\omega$ the set of ω-relations over Q. Let $\Omega = \lambda x.x=\omega$. Suppose the behaviour of a program π is described by a ω-relation R. Then, $\tilde{pre}[R](\neg\Omega)$ clearly represents the set of initial states from which the execution of π cannot reach $\neg\neg\Omega$, i.e., must terminate. Hence, $\tilde{pre}[R](P) \cap \tilde{pre}[R](\neg\Omega) = \tilde{pre}[R](P-\Omega)$ is the predicate characterizing the set of initial states from which the execution of π must terminate in P. We can therefore define a total correctness pt for ω-relations, which we denote wpr[R], by

(6) $wpr[R] = \lambda P.\tilde{pre}[R](P-\Omega) = \lambda P \lambda x.\forall y \ (x,y) \in R \Rightarrow P(y) \wedge y \neq \omega$ (1)

The following properties of wpr[R] are derived from those of $\tilde{pre}[R]$, knowing that R is a ω-relation.

Properties 5.

5.1. $wpr[R](\Omega) = \bot$

5.2. $wpr[R](\bigcap_{i \in F} P_i) = \bigcap_{i \in F} wpr[R](P_i)$ (for $F \neq \emptyset$)

5.3. $wpr[R \circ S](P) = wpr[R](wpr[S](P))$ (2)

In order to be sure that wpr[R] is a useful total correctness pt we must verify that it satisfies the five healthiness criteria [Dij76][Hoa78a]. For a pt f these criteria are written :

H1. $f(\Omega) = \bot$ (the law of the excluded miracle)

H2. $f(\bigcap_{i \in F} P_i) = \bigcap_{i \in F} f(P_i)$ (pour $F \neq \emptyset$)

H3. $A \subseteq B \Rightarrow f(A) \subseteq f(B)$

H4. $f(A) \cup f(B) \subseteq f(A \cup B)$

H5. f is continuous (from below), i.e., for every increasing sequence of predicates $\{P_i\}_{i \in \mathbb{N}}$, $f(\bigcup_{i \in \mathbb{N}} P_i) = \bigcup_{i \in \mathbb{N}} f(P_i)$.

It is easy to see that H3 and H4 are implied by H2. Therefore, by properties 5, wpr[R] satisfies H1, H2, H3 and H4. We shall now study the continuity of wpr[R]. Equation (6) suggests that we start by looking at $\tilde{pre}[R]$. First of all, three defi-

(1) This corresponds to the pt used in [Roe76].

(2) Note that if R and S are ω-relations, so is $R \circ S$.

nitions :

Definition 1. Determinacy. A relation R is deterministic if $\forall x\ |R(x)|\leq 1$.

Definition 2. Non-determinacy. A relation R is non-deterministic if $\exists x\ |R(x)|>1$.

Definition 3. Finite non-determinacy. A relation R is finitely non-deterministic if $\forall x\ \exists k\in\mathbb{N}\ |R(x)|\leq k$.

Theorem 1. $\widetilde{pre}[R]$ is continuous iff R is finitely non-deterministic.

Proof. If part : Suppose R finitely non-deterministic and let $\{P_i\}_i$ be an increasing sequence of predicates. We have to show that, for all x (omitting the [R]):

(7) $\widetilde{pre}(\cup P_i)(x) = \exists i\ \widetilde{pre}(P_i)(x)$.

RHS⇒LHS is a consequence of 4.2., (just like H4 is a consequence of H2).

Let $x\in Q$. If $R(x) = \emptyset$ then, for any predicate A, $\widetilde{pre}(A)(x) = tt$ and (7) is trivially established. If $R(x) \neq \emptyset$ consider that LHS is $\forall y\ (x,y)\in R \Rightarrow \exists i\ P_i(y)$. For each $y\in R(x)$ define $s(y) = \min\{i \mid P_i(y)\}$. The set $\{s(y) \mid y\in R(x)\}$ is finite and non-empty, and thus it has a maximum m. Since $\{P_i\}_i$ is increasing we have $\forall y\ (x,y)\in R \Rightarrow P_m(y)$, i.e., $\widetilde{pre}(P_m)(x)$. Thus, (7) holds in all cases.

Only if part : Suppose R is not finitely non-deterministic and choose $x_o$ s.t. $R(x_o)$ is infinite. Let D be a countable subset of $R(x_o)$, $D = \{y_0, y_1, \ldots, y_j, \ldots\}$. Let $\{P_i\}_i$ be the increasing sequence defined by

$$P_i(y) = \begin{cases} i>j & \text{if } y\in D \text{ where } y \text{ is } y_j \\ tt & \text{if } y\notin D \end{cases}$$

Easy calculations show that $\widetilde{pre}(\cup P_i)(x_o) = \widetilde{pre}(\tau)(x_o) = tt$, while $(\exists i\ \widetilde{pre}(P_i)(x_o)) = ff$. Hence, in this case $\widetilde{pre}[R]$ is not continuous. This ends the proof. □

It is clear by (6) that whenever R is finitely non-deterministic wpr[R] is continuous. However, there are programs that may produce an infinite number of results from a given initial state and, therefore, cannot be modelled by finitely non-deterministic relations. It is a general property of implementable programs that if such a situation occurs, non-termination is also possible, (see [Dij76, ch.9]). This property is called "bounded non-determinacy". We formalize it by :

Definition 4. Bounded non-determinacy. A ω-relation R is boundedly non-deterministic if $\forall x\ (\exists k\in\mathbb{N}\ |R(x)|\leq k \vee (x,\omega)\in R)$.

The following fundamental theorem shows that in all interesting cases wpr[R] is continuous.

Theorem 2. For $R\in\mathcal{R}_\omega$, wpr[R] is continuous iff R is boundedly non-deterministic.

Proof. If part : Suppose $R\in\mathcal{R}_\omega$ is boundedly non-deterministic and let $\{P_i\}_i$ be any increasing sequence of predicates. We want to show that, for all x,

(8) $wpr[R](\cup P_i)(x) = \exists i\ wpr[R](P_i)(x)$

Let $x_o\in Q$ s.t. $\exists k\ |R(x_o)|\leq k$, and set $R_o = \{(x_o,y) \mid (x_o,y)\in R\}$. Clearly, $R_o$

is finitely non-deterministic. Besides, $\tilde{pre}[R](A)(x_o) = \tilde{pre}[R_o](A)(x_o)$ for any predicate A. Hence, for $x_o$

$$\begin{aligned} LHS &= \tilde{pre}[R_o]((\cup P_i)-\Omega)(x_o) \\ &= \tilde{pre}[R_o](\cup(P_i-\Omega))(x_o) \\ &= \exists i\ \tilde{pre}[R_o](P_i-\Omega)(x_o) = RHS \quad \text{(by th.1)} \end{aligned}$$

Let now $x_o$ s.t. $(x,\omega)\in R$. Then, for all A, $wpr[R](A)(x_o) = ff$ and (8) is trivially established.

Only if part : Suppose $R\in\mathcal{R}_\omega$ is not boundedly non-deterministic, i.e., $\exists x\ (\forall k\ |R(x)|>k \wedge (x,\omega)\notin R)$. For such x, $wpr[R](A)(x) = \tilde{pre}[R](A)(x)$, for any A. We can therefore use the same example as in the proof of th.1 to show that (8) does not always hold. This ends the proof. □

## 5. A MODEL FOR THE LANGUAGE OF GUARDED COMMANDS

We have shown that wpr[R] is a healthy total correctness pt whenever R is a boundedly non-deterministic ω-relation, (a ωb-relation, for short). The question now is : does wpr correspond indeed to Dijkstra's wp function ? Remark that wpr[R] is formally defined in terms of a ω-relation R, while wp(S) is known through the axioms for the commands S of the language of guarded commands [Dij76]. To answer the question we must supply a means of mapping commands into ωb-relations and then use our definitions to derive the axioms. If we manage to do so, we shall have proven the consistency of Dijkstra's axioms in our model for non-deterministic programs.

Let π represent an arbitrary program whose state space is a set Q (with $\omega\in Q$), and let x denote π's generalized program variable. Let ρ represent a command, μ a guard, γ a guarded command, σ a set of guarded commands, ε a total computable function of Q→Q s.t. $\varepsilon(\omega) = \omega$. The abstract syntax of the language of guarded commands is :

π ::= ρ

ρ ::= skip | abort | x:=ε(x) | if σ fi | do σ od | ρ1 ; ρ2

σ ::= γ1 □ γ2 □ ... □ γn (n≥0)

γ ::= μ→ρ

We shall now define a function rho taking a command as argument and delivering the ωb-relation that axiomatically characterizes that command.

i. rho[[ skip ]] = {(x,y) | y=x} = SKIP

pre[SKIP](P) = λx.∃y (x,y)∈SKIP ∧ P(y) = λx.∃y y=x∧P(y) = λx.P(x) = P

$\tilde{pre}[SKIP](P) = \neg pre[SKIP](\neg P) = \neg\neg P = P$

wpr[SKIP](P) = $\tilde{pre}[SKIP](P-\Omega) = P-\Omega$

ii. rho[[ abort ]] = {(x,y) | y=ω} = ABORT

$$pre[ABORT](P) = \lambda x.\exists y\ (x,y)\in ABORT \wedge P(y) = \lambda x.\exists y\ y=\omega \wedge P(y) = \lambda x.P(\omega) = \begin{cases} \top & \text{if } P(\omega)=tt \\ \bot & \text{if } P(\omega)=ff \end{cases}$$

$$\tilde{pre}[ABORT](P) = \neg\lambda x.\neg P(\omega) = \lambda x.P(\omega)$$

$$wpr[ABORT](P) = \lambda x.(P\text{-}\Omega)(\omega) = \lambda x.ff = \bot$$

iii. $rho[\![\ x:=\varepsilon(x)\ ]\!] = \varepsilon$

$$pre[\varepsilon](P) = \lambda x.\exists y\ y=\varepsilon(x) \wedge P(y) = \lambda x.P(\varepsilon(x)) = P\circ\varepsilon$$

$$\tilde{pre}[\varepsilon](P) = \neg(\neg P\circ\varepsilon) = P\circ\varepsilon$$

$$wpr[\varepsilon](P) = (P\text{-}\Omega)\circ\varepsilon = P\circ\varepsilon\text{-}\Omega\circ\varepsilon$$

We now introduce three auxiliary functions mu, gamma and sigma, giving formal meaning to guards, guarded commands and sets of guarded commands, respectively.

iv. A guard $\mu$ is a total computable predicate over Q such that $\mu(\omega)=tt$.

$$mu[\![\ \mu\ ]\!] = \{(x,y) \mid \mu(x) \wedge y=x\}.$$

v. $gamma[\![\ \mu \to \rho\ ]\!] = mu[\![\ \mu\ ]\!] \circ rho[\![\ \rho\ ]\!]$

vi. $sigma[\![\ \gamma 1 \square \gamma 2 \square \ldots \gamma n\ ]\!] = \bigcup_{i=1}^{n} gamma[\![\gamma i]\!]$

When $n=0$ we define $sigma[\![\ ]\!] = \{(\omega,\omega)\}$

The following extension operators on relations are convenient in the sequel :

$$R\uparrow = R \cup \{(x,x) \mid R(x)=\emptyset\}, \quad R\downarrow = R \cup \{(x,\omega) \mid R(x)=\emptyset\}.$$

vii. $rho[\![\underline{if}\ \sigma\ \underline{fi}\ ]\!] = sigma[\![\ \sigma\ ]\!]\downarrow$ = IF. If all the $rho[\![\rho i]\!]$ are $\omega$b-relations, so is IF. Let else = $\lambda x.sigma[\![\ \sigma\ ]\!](x)=\emptyset$. Then $else(\omega)=ff$ and for $n\geq 1$ $else=\neg\bigcup_{i=1}^{n} \mu i$.

Also : $rho[\![\underline{if}\ \underline{fi}\ ]\!] = \{(\omega,\omega)\}\downarrow = \{(x,y) \mid y=\omega\}$ = ABORT, as expected !

$$pre[IF](P) = \lambda x.\exists y\ ((x,y)\in sigma[\![\ \sigma\ ]\!] \vee y=\omega \wedge else(x)) \wedge P(y)$$

$$= \lambda x.\exists i\ \mu i(x) \wedge (\exists y(x,y)\in rho[\![\ \rho i\ ]\!] \wedge P(y)) \cup \lambda x.else(x) \wedge P(\omega)$$

$$= \bigcup_{i=1}^{n} \mu i \cap pre[rho[\![\ \rho i\ ]\!]](P) \cup \lambda x.P(\omega) \cap else$$

$$\tilde{pre}[IF](P) = \bigcap_{i=1}^{n} (\mu i \Rightarrow \tilde{pre}[rho[\![\ \rho i\ ]\!]](P)) \cap (\lambda x.P(\omega) \cup \neg else)$$

$$wpr[IF](P) = \neg else \cap \bigcap_{i=1}^{n} (\mu i \Rightarrow wpr[rho[\![\ \rho i\ ]\!]](P))$$

Two unary operators $*$ and $\times$ on pt's are used in the next subsection :

$$f^{*}(P) = \bigcup_{i\in\mathbb{N}} f^{i}(P), \quad f^{\times}(P) = \bigcap_{i\in\mathbb{N}} f^{i}(P)$$

viii. We want to define $rho[\![\ \underline{do}\ \sigma\ \underline{od}\ ]\!]$ = DO.

Let $exit = \lambda x.sigma[\![\ \sigma\ ]\!](x)=\emptyset$. (exit is like else in vii.)

Let $GG = sigma[\![\ \sigma\ ]\!]\uparrow$. GG is a $\omega$b-relation describing one iteration. Thus, the terminating computations of $\underline{do}\ \sigma\ \underline{od}$ are captured by the relation DOT = $\{(x,y) \mid (x,y)\in GG^{*} \wedge exit(y)\}$, where $GG^{*} = \bigcup_{i\in\mathbb{N}} GG^{i}$ is the reflexive transitive closure of GG. The non-terminating ones are given by a relation

DONT = $\{(x,\omega) \mid W(x)\}$, where $W$ is the predicate characterizing the set of states that are starting points of non-terminating computations. Thus DO = DOT ∪ DONT.
We now proceed to define W in terms of GG. Note that, obviously

(9) $W \subseteq \neg exit$

Besides, it must be possible for an iteration starting at a state in W to terminate in W or not to terminate :

(10) $\forall x \;\; W(x) \Rightarrow \exists y \;\; (x,y) \in GG \wedge (W(y) \vee y=\omega)$

By (6), (10) can be rewritten $W \subseteq \neg wpr[GG](\neg W)$.
Defining $\tilde{wpr}[R] = \lambda P.\neg wpr[R](\neg P)$, (the dual of $wpr[R]$),
(10) becomes

(11) $W \subseteq \tilde{wpr}[GG](W)$ or

(12) $W = W \cap \tilde{wpr}[GG](W)$ or still[1]

(13) $W = (I \wedge \tilde{wpr}[GG])(W)$

Thus, W is a fixed point of the pt $I \wedge \tilde{wpr}[GG]$.
It is a fact equivalent to th.2 that $\tilde{wpr}[R]$ is continuous from above iff R is a ωb-relation. (A pt f is continuous from above if for every decreasing sequence $\{P_i\}_i$, $f(\cap P_i) = \cap f(P_i)$). As a consequence, $I \wedge \tilde{wpr}[GG]$ is continuous from above. Furthermore, for every $P_o$, $(I \wedge \tilde{wpr}[GG])(P_o) \subseteq P_o$. This implies that the greatest fixed point of $I \wedge \tilde{wpr}[GG]$ that is less than $P_o$ exists and is given by $(I \wedge \tilde{wpr}[GG])^{\times}(P_o)$. With (9), it is clear that we must define
$W = (I \wedge \tilde{wpr}[GG])^{\times}(\neg exit)$.

It would now be possible to prove that DO is a ωb-relation.

Example : show that rho[[ do od ]] = SKIP.

The calculations for the pt's associated with DO are longer than the ones presented. They yield[2] :

$pre[DO](P) = (pre[GG])^{*}(exit \cap P) \cup \lambda x.P(\omega) \cap (I \wedge \tilde{wpr}[GG])^{\times}(\neg exit)$

$\tilde{pre}[DO](P) = (\tilde{pre}[GG])^{\times}(exit \Rightarrow P) \cap (\lambda x.P(\omega) \cup (I \vee wpr[GG])^{*}(exit))$

$wpr[DO](P) = (wpr[GG])^{\times}(exit \Rightarrow P) \cap (I \vee wpr[GG])^{*}(exit)$

(The pt's for GG are obtained like in vii.

In particular, $wpr[GG](P) = (else \Rightarrow P) \cap \bigcap_{i=1}^{n} (\mu i \Rightarrow wpr[rho[\![ \rho i ]\!](P))$ )

The expression we have for $wpr[DO](P)$ does not look like the one for wp(DO,P) in [Dij76]. However, further calculations would give
$wpr[DO](P) = (I \vee wpr[GG])^{*}(exit \Rightarrow P)$. Since $I \vee wpr[GG] = I \vee wpr[IF]$ we finally get :

$wpr[DO](P) = (I \vee wpr[IF])^{*}(exit \Rightarrow P)$,

which in fact corresponds to Dijkstra's axiom.

ix. $rho[\![ \rho 1;\rho 2 ]\!] = rho[\![ \rho 1 ]\!] \circ rho[\![ \rho 2 ]\!]$

---

(1) $I \wedge \tilde{wpr}[R]$ is the pt $\lambda P.P \cap \tilde{wpr}[R](P)$

(2) $I \vee wpr[R]$ is the pt $\lambda P.P \cup wpr[R](P)$

The associated pt's are immediately derived from properties 3.3, 4.3., 5.3. (Remark that the composition of two ωb-relations yields a ωb-relation).

x. Theorem 3. For every command $c$ in the language of guarded commands, $rho[\![c]\!]$ is a ωb-relation.

xi. Consistency. By inspecting the expressions obtained we conclude that wpr corresponds in our model to Dijkstra's wp function.

## 6. CONCLUSION

The proposed relational model has served here to study several predicate transformers, to give a semantics for the language of guarded commands, and to prove the consistency of Dijkstra's axioms for that language. Elsewhere [Gue79] we used it to formalize the concept of invariant and study its properties. We show that the current intuition is captured by defining an invariant of a ωb-relation as a solution of the inequation $J \subseteq wpr[R](J)$. We derive that the invariants are fixed points of the predicate transformer $I \wedge wpr[R]$.

We believe that the relational model provides helpful tools for reasoning about (non-deterministic) programs. The experiments we have made [Gue80] with a formalism similar to CSP [Hoa78b] let us hope that the model can be enlarged in order to cope also with certain classes of parallel programs.

## REFERENCES

[Dij76] E.W. Dijkstra, "A Discipline of Programming"
Prentice Hall, 1976.

[Gue79] P. Guerreiro,"Un modèle relationnel pour les programmes non-déterministes".
Rapport de D.E.A., Univ. Grenoble I, 1979.

[Gue80] P. Guerreiro, "Relational semantics of strongly communicating communicating sequential processes".
IMAG Report, Grenoble (to appear).

[Hoa78a] C.A.R. Hoare, "Some properties of predicate transformers".
JACM, 25, 3,July 1978, pp. 461/480.

[Hoa78b] C.A.R. Hoare, "Communicating sequential processes".
CACM 21, 8, August 1978, pp. 666/677.

[Roe76] W.P. de Roever, "Dijkstra's predicate transformer, non-determinism, recursion and termination".
Math. Found. Comp. Sci, LNCS 45, Springer, 1976, pp. 472/481.

[Sif79] J. Sifakis, "Le Contrôle des Systèmes Asynchrones : Concepts, Propriétés, Analyse Statique". Thèse d'Etat, Univ. Grenoble I, 1979.

[Wan77] M. Wand, "A characterization of weakest preconditions".
JCSS 15, 1977, pp. 209/212.