Action Systems, Unbounded Nondeterminism, and Infinite Traces

Michael Butler and Carroll Morgan
Oxford University Programming Research Group, Oxford, UK

February 1993

Abstract

Morgan [Mor90a] has described a correspondence between Back's action systems [BKS83] and the conventional *failures-divergences* model of Hoare's *communicating sequential processes* (CSP) formalism [Hoa85]. However, the CSP failures-divergences model does not treat unbounded nondeterminism, although unbounded nondeterminism arises quite naturally in action systems; to that extent, the correspondence between the two approaches is inadequate.

Fortunately there is an extended *infinite traces* model of CSP [RB89] which treats unbounded nondeterminism. We extend the CSP–action system correspondence, using that model instead, to take the unbounded nondeterminism of action systems properly into account.

In passing, we develop a definition of the weakest precondition under which an infinite heterogeneous trace of actions is enabled.

1 Introduction

Back's action system formalism [BKS83] is a state-based approach to distributed computing. Hoare's CSP [Hoa85] is an event-based approach to distributed computing. Morgan [Mor90a] has shown a correspondence between CSP and action systems by expressing the failures and divergences of CSP as weakest precondition formulae over action systems. With this CSP view of action systems, each action of an action system is visible to the environment of that action system and can be controlled by the environment. Often it is useful to make actions invisible to the environment and outside its control. Such an effect is achieved in CSP through the use of a hiding operator. An operator on action systems that has the effect of hiding a set of actions is introduced in [But92]. In most cases, this operator can be shown to correspond to the CSP hiding operator as defined in the CSP failures-divergences model. However, in the case of action systems with unbounded nondeterminism, the correspondence with CSP failures-divergences may be lost after hiding of actions.

We use an extended version of Dijkstra's guarded command language [Dij76] to specify the actions of an action system. One such extension is the *specification statement* which is found in the refinement calculus approach [Bac80, Mor88, Mrr87]. For program variable x, and predicates pre and $post$, the specification statement $x : [pre, post]$ has the effect of nondeterministically assigning some value satisfying $post$ to x provided pre is satisfied initially. This allows us to specify actions in an abstract manner. However, a specification statement can easily be unboundedly nondeterministic; consider, for example, $x : [true, x \in \mathbf{nat}]$. In order to maintain the correspondence between action systems and the CSP failures-divergences model, we could restrict actions to be boundedly nondeterministic. Unfortunately, when using specification statements, this restriction is somewhat arbitrary and showing that it is satisfied requires proof. We would prefer to allow actions to be unboundedly nondeterministic and still maintain the correspondence with CSP.

Roscoe & Barrett [RB89] have shown the inability of the CSP failures-divergences model to deal properly with unbounded nondeterminism, and so extended the model by including the possible infinite traces of a process.

In this paper, we introduce and define the formula $\overline{inf}(\Sigma)$ using infinite sequences of predicate transformers. This describes the conditions under which Σ, an infinite sequential composition of actions, may be executed. We then use $\overline{inf}$ to show that the correspondence between action systems and the infinite traces model for CSP holds. We also show that the definition of hiding in the infinite traces model corresponds to action system hiding even when an action system is unboundedly nondeterministic. Hence we regard the infinite traces model as a more appropriate link between CSP and action systems.

Before presenting the main results, we briefly present the failures-divergences model of CSP and the infinite traces model of CSP. We also present action systems and their correspondence with the failures-divergences model.

2 Communicating Sequential Processes

In the failures-divergences model for CSP [Hoa85], a process is denoted by three components (A, F, D), where A is an alphabet of events, F is a set of failures, and D is a set of divergences. A failure is a pair of the form (s, X), where s is a finite trace (element of A^*) and X is a refusal set (subset of A). If (s, X) is in F, then, after engaging in the event trace s, the process may refuse all events in X. A divergence is simply a finite trace, and s in D means that after engaging in event trace s the process may behave chaotically. To be a proper model of a CSP process, (A, F, D) must satisfy the seven well-formedness conditions

A CSP process denotation $\Theta = (A, F, D)$ must satisfy:

C0 $(\langle\rangle, \emptyset) \in F$
C1 $(s\hat{\ }t, X) \in F \;\Rightarrow\; (s, \emptyset) \in F$
C2 $(s, Y) \in F \wedge X \subseteq Y \;\Rightarrow\; (s, X) \in F$
C3 $(s, X) \in F \wedge Y \subseteq A \wedge (\forall a \in Y \cdot (s\hat{\ }\langle a\rangle, \emptyset) \notin F) \;\Rightarrow\; (s, X \cup Y) \in F$
C4 $D \subseteq \mathit{domain}(F)$
C5 $s \in D \wedge t \in A^* \;\Rightarrow\; s\hat{\ }t \in D$
C6 $s \in D \wedge X \subseteq A \;\Rightarrow\; (s, X) \in F$

Figure 1: Well-formedness conditions for failures-divergences model.

C0–**C6** listed in Figure 1 [Hoa85][1].

The behaviour of a process may be specified using the CSP process algebra. The term STOP represents the process that refuses to engage in any event. Sequencing of events is described by the prefix operator ($\rightarrow$): the term $a \rightarrow \mathcal{P}$ describes the process that engages in the event a and then behaves as process $\mathcal{P}$. External choice of behaviour is described by the choice operator ($[\!]$): $\mathcal{P}[\!]\mathcal{Q}$ represents the process that offers the choice to the environment between behaving as $\mathcal{P}$ or as $\mathcal{Q}$. CSP also has a separate nondeterministic-choice operator ($\sqcap$): $\mathcal{P} \sqcap \mathcal{Q}$ represents the process that internally chooses between behaving as $\mathcal{P}$ or as $\mathcal{Q}$, leaving the environment with no control over that choice. A recursive definition is written $(\mu X \cdot F(X))$, where $F(X)$ is some expression containing X. Each term $\mathcal{P}$ of the algebra is modelled by some triple (A, F, D), written $[\![\mathcal{P}]\!]$.

For example, the process that can continually perform a-events, or can nondeterministically stop at any time, is specified algebraically as:

$$\mathcal{M} \;\;\widehat{=}\;\; (\mu X \cdot (a \rightarrow X) \sqcap \mathit{STOP}).$$

The failures-divergences denotation of $\mathcal{M}$ is given by:

$$[\![\mathcal{M}]\!] \;\;=\;\; (\{a\},\ \{a\}^* \times \mathbb{P}\,\{a\},\ \emptyset).$$

Here, for example, the presence of failures $(\langle\rangle, \{a\})$ and $(\langle a\rangle, \emptyset)$ indicates that $\mathcal{M}$ can refuse to perform an a-event initially, but also can do an a-event.

3 Unbounded Nondeterminism in CSP

As Roscoe & Barrett [RB89] point out, the CSP failures-divergences model cannot deal adequately with nondeterminism that takes an unbounded amount of time to appear. Take the process $\mathcal{N}$, described algebraically as:

$$\mathcal{N} \;\widehat{=}\; \sqcap_{i \geq 0} \mathcal{N}_i \quad \text{where} \quad \mathcal{N}_0 \;\widehat{=}\; \mathit{STOP} \;\; \text{and} \;\; \mathcal{N}_{n+1} \;\widehat{=}\; a \rightarrow \mathcal{N}_n.$$

[1] The version of **C3** given here is stronger than that in [Hoa85]. It comes from [RB89] and is necessary for infinite A.

For process model $\Theta = (A, F, D)$, $C \subseteq A$, let $B \mathrel{\widehat{=}} (A - C)$. Then:

$$\Theta \setminus C \quad \widehat{=} \quad (B,\ \Theta \setminus_{\mathcal{F}} C,\ \ \Theta \setminus_{\mathcal{D}} C),$$

where $\Theta \setminus_{\mathcal{F}} C$ are those $r \in B^*$, $X \subseteq B$ satisfying:

$$(\exists s \in A^* \cdot s \upharpoonright B = r \wedge (s, X \cup C) \in F) \ \vee \ r \in (\Theta \setminus_{\mathcal{D}} C),$$

and $\Theta \setminus_{\mathcal{D}} C$ are those $r \in B^*$ satisfying:

$$(\exists s \in A^* \cdot s \upharpoonright B \leq r \wedge (s \in D \ \vee \ (\forall n \geq 0 \cdot \exists u \in C^* \cdot (s\hat{\ }u, \emptyset) \in F)).$$

Figure 2: Definition of CSP hiding in the failures-divergences model.

$\mathcal{N}$ may perform an unbounded number of a-events though, unlike $\mathcal{M}$, we do not intend that it can perform an infinite number of a-events. Yet $\mathcal{N}$ has exactly the same failures-divergence sets as $\mathcal{M}$. We would prefer a model that differentiates $\mathcal{M}$ and $\mathcal{N}$ because we expect entirely different results when we hide a in $\mathcal{M}$ and in $\mathcal{N}$.

Divergence is introduced by the CSP hiding operator when possible infinite behaviour is hidden. In the failures-divergences model, infinite behaviour is interpolated from finite behaviour. By the definition of the hiding operator (see Figure 2) [Hoa85], we have $\mathcal{M} \setminus \{a\} = \mathit{CHAOS}$, the process that diverges immediately. This is because it is possible for $\mathcal{M}$ to perform an infinite number of a-events. Since $\mathcal{N}$ has the same failures-divergences model as $\mathcal{M}$, $\mathcal{N} \setminus \{a\} = \mathit{CHAOS}$ also. But, since $\mathcal{N}$ is not able to perform an infinite number of a-events, we expect $\mathcal{N} \setminus \{a\} = \mathit{STOP}$, the process that terminates immediately.

The infinite traces model was introduced by Roscoe & Barrett [RB89] in order to deal with the form of unbounded nondeterminism exhibited by $\mathcal{N}$. The infinite traces model extends the failures-divergences model by including all possible infinite traces of a process. A process model now has four components (A, F, D, I) where A, F and D are as before and I is some subset of A^ω, the set of infinite sequences of elements of A.

Now the model of $\mathcal{M}$ is given by:

$$[\![\mathcal{M}]\!] \quad = \quad (\{a\},\ \{a\}^* \times \mathbb{P}\,\{a\},\ \emptyset,\ \{a\}^\omega),$$

while the model of $\mathcal{N}$ is given by:

$$[\![\mathcal{N}]\!] \quad = \quad (\{a\},\ \{a\}^* \times \mathbb{P}\,\{a\},\ \emptyset,\ \emptyset).$$

In the infinite traces model, the hiding operator uses the I component to determine when divergence may be introduced, rather than interpolating it from F (see Figure 5). With this new hiding operator, $\mathcal{M} \setminus \{a\} = \mathit{CHAOS}$ as before, while $\mathcal{N} \setminus \{a\} = \mathit{STOP}$ as expected.

A process model $\Theta = (A, F, D, I)$ must satisfy **C0**–**C6** as well as:

C7 $s\hat{\ }u \in I \Rightarrow (s, \emptyset) \in F$

C8 $s \in D \wedge u \in A^{\omega} \Rightarrow s\hat{\ }u \in I$

C9 Let S be a set of process denotations each with alphabet A, and let:

$$\begin{array}{rcl} \sqcap S & \widehat{=} & (A, F, D, I) \\ \text{where } F & \widehat{=} & \cup\{ F' \mid (A, F', D', I') \in S \} \\ D & \widehat{=} & \cup\{ D' \mid (A, F', D', I') \in S \} \\ I & \widehat{=} & \cup\{ I' \mid (A, F', D', I') \in S \}. \end{array}$$

Then Θ should satisfy :

$\Theta = \sqcap \{ \Theta' \mid \Theta \sqsubseteq \Theta' \wedge \Theta' \text{ is pre-deterministic} \}$.

Figure 3: Well-formedness conditions for infinite traces model.

The components (F, D, I) must satisfy well-formedness conditions **C0**–**C6** as well as three extra conditions **C7**–**C9** listed in Figure 3 [RB89][2]. **C7** and **C8** are straightforward, w hile **C9** requires some explanation. The purpose of **C9** is to ensure that there are enough infinite traces to be consistent with what the failures prove the process Θ can be forced to do. For example, if F says that the process can never refuse an a-event, then I must contain an infinite sequence of a-events. The formulation of **C9** presented here says that Θ must be equal to the nondeterministic choice of all its pre-deterministic implementations. $\Theta' = (A, F', D', I')$ is an *implementation of* $\Theta = (A, F, D, I)$, written $\Theta \sqsubseteq \Theta'$, if:

$$F \supseteq F' \quad \wedge \quad D \supseteq D' \quad \wedge \quad I \supseteq I'.$$

A process is *pre-deterministic* if it a lways behaves deterministically unless it diverges. Pre-determinism will be defined formally in Section 9.

4 Actions

An action is any statement in an extended version of Dijkstra's guarded command language. Statements in the guarded command language are defined using *weakest precondition* formulae: for statement S and postcondition R, the formula $wp(S, R)$ (weakest precondition of S w.r.t. R) characterises those initial states from which action S is guaranteed to terminate in a state satisfying R. Dijkstra has given wp definitions of assignment ($x := E$), sequential composition ($S; T$), conditional statements (**if** ... **fi**), and iteration (**do** ... **od**). For example:

$$wp(S; T, R) \quad \widehat{=} \quad wp(S, wp(T, R)).$$

[2] **C7**–**C9** are numbered **6**–**8** in [RB89].

Along with these, we use *specification statements*, *naked guarded commands*, and *generalised choice*.

The specification statement $x : [pre, post]$ assigns some value to x satisfying $post$ provided pre is satisfied initially. Also, $post$ may refer to the initial value of x as x_0. The wp definition of $x : [pre, post]$ is as follows [Mor88] (let $Q[x \backslash E]$ represent Q with all free occurrences of x replaced by E):

Definition 1 $wp(x : [pre, post], R) \ \hat{=} \ pre \wedge (\forall x \cdot post \Rightarrow R)[x_0 \backslash x]$.

A special case of the specification statement is *generalised assignment*, written $x : \odot E$, where $\odot$ is some binary relation such as $=, <, \in$, and E is some expression possibly containing x. The assignment $x : \odot E$ is short for $x : [true, x \odot E[x \backslash x_0]]$.

For predicate G and statement S, the naked guarded command $G \rightarrow S$ is defined by:

Definition 2 $wp(G \rightarrow S, R) \ \hat{=} \ G \Rightarrow wp(S, R)$.

The generalised choice of an I-indexed set of statements $S_i, i \in I$, is written $([] \, i \in I \cdot S_i)$, and defined as follows:

Definition 3 $wp(([] \, i \in I \cdot S_i), R) \ \hat{=} \ (\forall i \in I \cdot wp(S_i, R))$.

Weakest precondition formulae may be regarded as *predicate transformers* that map postconditions to preconditions. For statement S, we write the predicate transformer explicitly as $wp(S, \_)$. For any statement S in the extended guarded command language, $wp(S, \_)$ satisfies:

- $Q \Rightarrow R$ implies $wp(S, Q) \Rightarrow wp(S, P)$ (*monotonicity*)
- $wp(S, (\forall i \in I \cdot R_i)) = (\forall i \in I \cdot wp(S, R_i))$ (*conjunctivity*).

The following two properties are satisfied by statements of Dijkstra's basic guarded command language, but not by statements of the extended language:

- $wp(S, false) = false$ (*excluded miracle*)
- For any chain $\{R_i \mid i \in I\}$,
  $wp(S, (\exists i \in I \cdot R_i)) = (\exists i \in I \cdot wp(S, R_i))$ (*continuity*).

Note: $\{R_i \mid i \in I\}$ is a chain if, for each $i, j \in I$, $R_i \Rightarrow R_j$ or $R_j \Rightarrow R_i$.

For statement S, $wp(S, false)$ represents those initial states in which S is guaranteed to establish any postcondition, i.e., behave miraculously. For example, the statement $false \rightarrow S$ is miraculous in any initial state:

$$
\begin{array}{ll}
 & wp(false \rightarrow S, false) \\
= & false \Rightarrow wp(S, false) \qquad \textit{Definition 2} \\
= & true.
\end{array}
$$

We take the view that a statement is enabled only in those initial states in which it behaves non-miraculously. The condition in which a statement S is enabled is called its guard, written $gd(S)$, and is defined as follows:

$$
M \mathrel{\widehat{=}} \left(\begin{array}{l} \textbf{var } f : \textbf{bool} \\ \textbf{initially} \;\; f :\in \textbf{bool} \\ a : \quad f = \mathit{true} \rightarrow f :\in \textbf{bool} \end{array} \right)
$$

$$
N \mathrel{\widehat{=}} \left(\begin{array}{l} \textbf{var } n : \textbf{nat} \\ \textbf{initially} \;\; n :\in \textbf{nat} \\ a : \quad n > 0 \rightarrow n := n - 1 \end{array} \right)
$$

Figure 4: Example action systems

Definition 4 $gd(S) \mathrel{\widehat{=}} \neg wp(S, false)$.

For example, it can be shown that $gd(G \rightarrow x := E) = G$, provided E is well-defined.

An unboundedly nondeterministic statement need not satisfy the continuity property. For example, the predicate transformer $wp(x :\in \textbf{nat}, \_)$ does not distribute over the chain $(\exists i \in \textbf{nat} \cdot i > x)$.

5 Action Systems

An action system P [BKS83] consists of an alphabet of labels A, a set of labelled actions (statements) $P_A = \{P_a \mid a \in A\}$ and an initialisation $P_\imath$. An action system proceeds by firstly executing the initialisation. Then, repeatedly, an enabled action from the set P_A is selected and executed. An action system terminates if no action of P_A is enabled, and diverges whenever some action aborts.

When specifying an example action system, we write '**initially** S' for its initialisation, and '$a : S$' for the action labelled a. Figure 4 presents two example action systems. Action system M may continually perform a-events or may terminate at any stage. Action system N performs an unboundedly nondeterministic number of a-events and then terminates. Intuitively, we can see that the behaviour of action system M is similar to that of CSP process $\mathcal{M}$ of Section 2, while the behaviour of N is similar to that of CSP process $\mathcal{N}$ of Section 3.

6 The CSP–Action System Correspondence

[Mor90a] shows the correspondence between action systems and the CSP of [Hoa85] by defining the failures and divergences of an action system as *conjugate* weakest precondition formulae over the initialisation and the labelled actions. For statement S and postcondition R, the formula $\overline{wp}(S, R)$ (conjugate weakest precondition of S w.r.t. R) characterises those initial states from which S could *possibly* establish R (which includes the possibility that it may fail to terminate). It is defined as follows [Mor90a]:

Definition 5 $\overline{wp}(S, R) \;\widehat{=}\; \neg wp(S, \neg R)$.

For statement S, $\overline{wp}(S, \_)$ is monotonic, and is disjunctive rather than conjunctive. Also, $\overline{wp}(S, R) \Rightarrow gd(S)$ for any R.

For any set of labels $X \subseteq A$, we write $gd(P_X)$ for the *disjunction* of the guards of actions labelled from X. For any trace $t \in A^*$, we write P_t for the sequential composition of the actions of P labelled from t, with $P_{\langle\rangle} = \mathbf{skip}$. We require[3] that the initialisation $P_\imath$ is always enabled ($gd(P_\imath) = true$), and that the outcome of $P_\imath$ is independent of the initial value of any variable, that is, for any predicate R:

$$wp(P_\imath, R) = true \quad \text{or} \quad wp(P_\imath, R) = false.$$

The failures-divergences semantics of action system P, denoted $\{\!\!\{P\}\!\!\}$, are given by the following definition [Mor90a]:

Definition 6 *For action system* $P \;=\; (A,\; P_\imath,\; P_A)$,

$$\{\!\!\{P\}\!\!\} \;\;\widehat{=}\;\; (A,\; \mathcal{F}\{\!\!\{P\}\!\!\},\; \mathcal{D}\{\!\!\{P\}\!\!\}),$$

where $\mathcal{F}\{\!\!\{P\}\!\!\}$ *are those* $(s, X) \in (A^* \times \mathbb{P}\, A)$ *satisfying:*

$$\overline{wp}(P_{\langle\imath\rangle\frown s}, \neg gd(P_X)),$$

and $\mathcal{D}\{\!\!\{P\}\!\!\}$ *are those* $s \in A^*$ *satisfying:*

$$\overline{wp}(P_{\langle\imath\rangle\frown s}, false).$$

Thus (s, X) is a failure of P if, after initialisation, P could possibly engage in the action trace s and then refuse all actions X. Trace s is a divergence of P if $P_{\langle\imath\rangle\frown s}$ is not guaranteed to terminate.

Theorem 1 *In the failures-divergences model,* $\{\!\!\{P\}\!\!\}$ *is well-formed.*

Proof outline: As is claimed in [Mor90a], it can be shown that $\{\!\!\{P\}\!\!\}$ satisfies well-formedness conditions **C0**–**C6** of Figure 1, using properties of $\overline{wp}$. For example, **C1** says that the traces of P must be prefix-closed:

$$\begin{array}{rcll}
(s\frown t, X) \in \mathcal{F}\{\!\!\{P\}\!\!\} & = & \overline{wp}(P_{\langle\imath\rangle\frown s\frown t}, \neg gd(P_X)) & \textit{Definition 6} \\
& = & \overline{wp}(P_{\langle\imath\rangle\frown s}, \overline{wp}(P_t, \neg gd(P_X))) & \textit{sequential comp.} \\
& \Rightarrow & \overline{wp}(P_{\langle\imath\rangle\frown s}, true) & \textit{monotonicity} \\
& = & (s, \emptyset) \in \mathcal{F}\{\!\!\{P\}\!\!\} & \textit{Definition 6.}
\end{array}$$

[3]These conditions on the initialisation could be relaxed, but Definition 6 would then be complicated slightly.

The other conditions are similarly proven.
□

Using Definition 6, we can show that both M and N have the same failures-divergence sets as the CSP processes $\mathcal{M}$ and $\mathcal{N}$.

7 Hiding in Action Systems

In this section, we present a hiding operator for action systems [But92]. First we present a program construct for nondeterministic iteration. The iterate construct **it** S **ti** [Mor79] causes the action S to be executed zero or more times. It should satisfy the following equation:

$$Y \quad = \quad \mathbf{skip} \,[\!]\, (S\,; Y).$$

We define it to be the least solution of this equation, where least is w.r.t. the refinement ordering on statements ($S \preceq S'$):

Definition 7 *$S \preceq S'$ iff for each postcondition R, $wp(S, R) \;\Rightarrow\; wp(S', R)$.*

Now $wp(\mathbf{it}\ S\ \mathbf{ti}, R)$ is given by the following lemma (let $(\mu X \cdot f(X))$ be the least fixed-point of f):

Lemma 1 *For predicate R,* $\quad wp(\mathbf{it}\ S\ \mathbf{ti}, R) \;=\; (\mu X \cdot R \wedge wp(S, X)).$

Assume we have an action system $P = (A, P_\imath, P_A)$. For $C \subseteq A$, the action system $P \setminus C$ is the result of hiding the set of actions C in P and is constructed as follows:

Definition 8

$$\begin{array}{rcl} (P \setminus C)_\imath & \widehat{=} & P_\imath \,; IT_C \\ (P \setminus C)_a & \widehat{=} & IT_C \,; P_a \,; IT_C, \quad \text{each } a \in (A - C) \\ & & \text{where } IT_C \;\widehat{=}\; \mathbf{it}\ ([\!]\, c \in C \cdot P_c)\ \mathbf{ti}. \end{array}$$

Intuitively, IT_C represents the execution of an arbitrary number of hidden actions, and so such executions may occur in between execution of the visible actions of $P \setminus C$. (Note that if IT_C can execute infinitely, it is equivalent to **abort**.)

In the CSP process algebra, the hiding operator is defined in terms of the semantic model, i.e., for term $\mathcal{P}$:

$$[\![\mathcal{P} \setminus C]\!] \quad \widehat{=} \quad [\![\mathcal{P}]\!] \setminus C,$$

where the function '$\setminus C$' is as defined in Figure 2. For action system P, provided $P_\imath$ and each P_a are boundedly nondeterministic and C is finite, then it can be shown that Definition 8 corresponds to CSP hiding in the failures-divergences model. That is, using properties of wp, we can show that:

$$\{\![P \setminus C]\!\} \quad = \quad \{\![P]\!\} \setminus C.$$

Note that hiding may introduce unbounded nondeterminism into a boundedly nondeterministic action system: if C is infinite then IT_C may be unboundedly nondeterministic, so the composite actions of $P \setminus C$ will also be unboundedly nondeterministic.

We now hide the a action in both examples of Figure 4. In each case, we end up with just an initialisation and no labelled actions. The initialisations are calculated as follows:

$$\begin{array}{lll}
& (M \setminus \{a\})_\imath & \\
= & f :\in \mathbf{bool} \;\; ; \;\; \mathbf{it}\ f = true \to f :\in \mathbf{bool}\ \mathbf{ti} & \textit{Definition 8} \\
= & f :\in \mathbf{bool} \;\; ; \;\; f : [f = false, f = false] & \text{(i)} \\
= & \mathbf{abort} & \textit{sequential composition.}
\end{array}$$

$$\begin{array}{lll}
& (N \setminus \{a\})_\imath & \\
= & n :\in \mathbf{nat}\,;\mathbf{it}\ n > 0 \to n := n - 1\ \mathbf{ti} & \textit{Definition 8} \\
= & n :\in \mathbf{nat}\,; n :\leq n & \text{(ii)} \\
= & n :\in \mathbf{nat} & \textit{sequential composition.}
\end{array}$$

Note: steps (i) and (ii) may be proven using Theorem 3, which is presented later.

Now, using Definition 6, it can be shown that $M \setminus \{a\}$ has the same failures-divergence sets as the CSP process *CHAOS*, and that $N \setminus \{a\}$ has the same failures-divergence sets as the CSP process *STOP*, i.e.:

$$\begin{array}{lll}
\{\![M \setminus \{a\}]\!\} & = & (\emptyset,\ \{(\langle\rangle, \emptyset)\},\ \{\langle\rangle\}) \\
\{\![N \setminus \{a\}]\!\} & = & (\emptyset,\ \{(\langle\rangle, \emptyset)\},\ \emptyset).
\end{array}$$

So, in the failures-divergences model, although $\mathcal{N}$ and N have the same behaviour model, $\mathcal{N} \setminus \{a\}$ and $N \setminus \{a\}$ have different behaviour models.

Comparison of the examples demonstrates that, in the case where an action system is unboundedly nondeterministic, the CSP–action system correspondence of [Mor90a] is lost after application of the hiding operator. The examples also suggest that the infinite traces model may be a more appropriate way of linking CSP with unboundedly nondeterministic action systems.

8 Infinite Sequential Composition

To formalise infinite sequential composition, we employ infinite sequences of predicates of the form $\Phi = \langle \Phi_i \mid i \geq 0 \rangle$. Implication on such sequences is defined pointwise:

$$\Phi \Rightarrow \Phi' \quad \text{iff} \quad (\forall i \geq 0 \cdot \Phi_i \Rightarrow \Phi_i').$$

Infinite sequences of predicate transformers are of the form $F = \langle\, F_i \mid i \geq 0 \,\rangle$. Application of F to Φ is defined by shifting Φ one place and applying F pointwise:

$$F(\Phi) \;\widehat{=}\; \langle\, F_i(\Phi_{i+1}) \mid i \geq 0 \,\rangle.$$

Such application is monotonic provided each individual F_i is monotonic.

Given an infinite sequence of actions $\langle\, S_i \mid i \geq 0 \,\rangle$ we write the infinite sequential composition over this sequence as:

$$(\;\; ; i \mid i \geq 0 \cdot S_i).$$

It is intended that $(\;\; ; i \mid i \geq 0 \cdot S_i) = (S_0 \,; S_1 \,; S_2 \,; \ldots)$. Given an infinitary action $\Sigma = (\;\; ; i \mid i \geq 0 \cdot S_i)$, an infinite sequence of predicate transformers can be constructed:

Definition 9 $\quad \overline{wp}_\infty(\Sigma, \_) \;\widehat{=}\; \langle\, \overline{wp}(S_i, \_) \mid i \geq 0 \,\rangle.$

For an infinitary action $\Sigma = (\;\; ; i \mid i \geq 0 \cdot S_i)$, we introduce the formula $\overline{inf}(\Sigma)$: we require that $\overline{inf}(\Sigma)$ characterise those initial states in which execution of all of Σ is *possible*. Operationally, for this to be possible in some state it must be the case that S_0 is enabled and that execution of S_0 could result in a state in which S_1 is enabled and so on for each S_i. So for infinite predicate Φ satisfying:

$$\Phi_i \;\Rightarrow\; \overline{wp}(S_i, \Phi_{i+1}), \quad \textit{each } i \in \mathbf{nat}, \tag{iii}$$

we have that Φ_0 represents a set of initial states in which execution of all of Σ is possible. This includes the possibility that one of the individual S_i's will abort. Now (iii) can be rewritten as:

$$\Phi \;\Rightarrow\; \overline{wp}_\infty(\Sigma, \Phi). \tag{iv}$$

Since we are looking for the weakest condition under which execution of all Σ is possible, we define $\overline{inf}(\Sigma)$ to be the first element of the infinite sequence Ψ, where Ψ is the weakest (i.e. greatest) solution of (iv). By the Knaster-Tarski Theorem [Tar55], the greatest solution of the equation $X \Rightarrow f(X)$, for monotonic f, is also the greatest fixed-point of f, written $(\bar{\mu}\, X \cdot f(X))$. So $\overline{inf}(\Sigma)$ is defined as follows:

Definition 10 *For* $\Sigma = (\; ; i \mid i \geq 0 \cdot S_i)$,

$$\overline{inf}(\Sigma) \;\widehat{=}\; \textit{head}(\Psi)$$

where $\Psi \;\widehat{=}\; (\bar{\mu}\, X \cdot \overline{wp}_\infty(\Sigma, X))$.

The set of infinite sequences of predicates forms a complete lattice: the greatest lower bound of a chain is defined pointwise, and the top element is the infinite sequence $\langle true \mid i \geq 0\rangle$. Also, the function $\overline{wp}_{\infty}(\Sigma, \_)$ is monotonic. Hence, by Knaster-Tarski [Tar55], the greatest fixed-point of $\overline{wp}_{\infty}(\Sigma, \_)$ does indeed exist.

9 The New Correspondence

In this section, we present the correspondence between action systems and the infinite traces model for CSP. For action system $P = (A, P_\imath, P_A)$ and infinite trace $u \in A^\omega$, let:

$$P_u \quad \widehat{=} \quad (\ ;\ i \mid i \geq 0 \cdot P_{u_i}).$$

$\{\!\![P]\!\!\}$ now has an extra component: the failures and divergences of P are as given in Definition 6, and the infinites of P are defined as follows:

Definition 11 *$\mathcal{I}\{\!\![P]\!\!\}$ are those $u \in A^\omega$ satisfying:*

$$\overline{inf}(P_{\langle\imath\rangle\hat{}u}).$$

Theorem 2 *In the infinite traces model, $\{\!\![P]\!\!\}$ is well-formed.*

Proof outline: We already know that well-formedness conditions **C0**–**C6** are satisfied by Definition 6. Using some properties of $\overline{inf}$ developed from Definition 10, it can be shown that **C7**–**C9** are satisfied by Definitions 6 and 11, as follows. **C7** and **C8** can easily be checked using the following lemma which follows from Definition 10:

Lemma 2 $\overline{inf}(S\ ;\ \Sigma) \;=\; \overline{wp}(S, \overline{inf}(\Sigma))$.

Proof that **C9** is satisfied is a little more involved, and is outlined here.

Firstly we look at pre-determinism. A pre-deterministic process satisfies the following two conditions [RB89]:

$$(s, X) \in F \;\Rightarrow\; s \in D \vee (\forall a \in X \cdot (s\hat{}\langle a\rangle, \emptyset) \notin F) \tag{v}$$
$$u \in A^\omega \wedge (\forall s < u \cdot (s, \emptyset) \in F) \;\Rightarrow\; u \in I. \tag{vi}$$

Formula (v) says that if a pre-deterministic process can refuse X after having performed s, then either it has diverged (in which case it can do anything anyway) or it cannot perform any event in X (deterministic). Formula (vi) says that if a pre-deterministic process can perform all (finite) prefixes of an infinite trace, then it can perform that infinite trace too.

We say that a statement S is pre-deterministic if, for any R, it satisfies:

$$\overline{wp}(S, R) \wedge wp(S, true) \quad\Rightarrow\quad wp(S, R). \tag{vii}$$

An action system is pre-deterministic if its initialisation and each of its actions is pre-deterministic. Using (vii) and Definitions 6 and 11, it can be shown that for pre-deterministic action system P, the semantic model of P is pre-deterministic, i.e. $\{\!\!\{P\}\!\!\}$ satisfies (v) and (vi).

Now, Roscoe & Barrett [RB89] have shown that **C9** may be reformulated as follows: $\Theta = (A, F, D, I)$ satisfies **C9** iff:

$$\begin{aligned} F \quad = \quad & \cup \{ \ F' \mid (\exists D', I' \cdot \Theta \sqsubseteq (A, F', D', I') \\ & \qquad\qquad \wedge (A, F', D', I') \text{ is pre-deterministic}) \ \}, \end{aligned}$$

which is equivalent to:

$$\begin{aligned} (s, X) \in F \quad \Rightarrow \quad & (\exists F', D', I' \cdot (s, X) \in F' \ \wedge \ \Theta \sqsubseteq (A, F', D', I') \\ & \qquad \wedge (A, F', D', I') \text{ is pre-deterministic}). \end{aligned} \qquad \text{(viii)}$$

To show that this is satisfied by $\{\!\!\{P\}\!\!\}$, we construct, for each $(s, X) \in \mathcal{F}\{\!\!\{P\}\!\!\}$, action system $P'_{(s,X)}$, which is a pre-deterministic implementation of P with (s, X) as a failure. $P'_{(s,X)}$ is constructed by constructing pre-deterministic implementations of each of the actions of P in such a way that $(s, X) \in \mathcal{F}\{\!\!\{P'_{(s,X)}\}\!\!\}$. As explained above, $\{\!\!\{P'_{(s,X)}\}\!\!\}$ is then pre-deterministic by construction. So for each $(s, X) \in F$, there is a model, $\{\!\!\{P'_{(s,X)}\}\!\!\}$, satisfying the right-hand side of (viii). Details may be found in [But92].
□

10 Action System Hiding Again

Using properties of wp and $\overline{inf}$, we can prove that the construction of action system $P \setminus C$ as described in Definition 8 corresponds to CSP hiding in the infinite traces model, even if P is unboundedly nondeterministic. The proof is outlined here. A more detailed proof may be found in [But92].

Firstly, we have the following theorem about **it** S **ti**:

Theorem 3 $\overline{wp}(\textbf{it}\ S\ \textbf{ti}, R) \ = \ (\exists i \geq 0 \cdot \overline{wp}(S^i, R)) \vee \overline{inf}(S^\infty)$.

Here, S^i represents i copies of S composed sequentially ($S^0 = \textbf{skip}$), and S^∞ represents $(\ ; i \mid i \geq 0 \cdot S)$, the infinite homogeneous sequence of S-actions. The theorem says that **it** S **ti** may establish R if there is some finite sequence of S-actions that may establish R, or if an infinite sequence of S-actions is possible (i.e. **it** S **ti** loops forever). The theorem is proven in the appendix. The proof uses a generalised fixed-point theorem which gives the greatest fixed-point of a non-continuous monotonic function. This generalised limit theorem uses ordinal induction and is based on one used in [HP72] and later in [Nel89].

The following lemmas, which follow from Theorem 3, are useful:

Lemma 3 $\quad \mathbf{it}\ S\ \mathbf{ti}\ ;\ \mathbf{it}\ S\ \mathbf{ti}\ =\ \mathbf{it}\ S\ \mathbf{ti}$.

Lemma 4 *For any predicate R:*
$\overline{wp}(\mathbf{it}\ ([\!]\, c \in C \cdot P_c)\ \mathbf{ti}, R)\ =\ (\exists t \in C^* \cdot \overline{wp}(P_t, R)) \vee (\exists u \in C^\omega \cdot \overline{inf}(\mathcal{P}_u))$.

Lemma 4 allows us to express certain constructs of action system $P \setminus C$ in terms of action system P. Assume that $P = (A, P_\imath, P_A)$, $Q = P \setminus C$ and $B = A - C$. Q is given by Definition 8. From Lemma 4 and induction over finite traces, the following lemma, which describes the effect of visible trace r in Q in terms of traces of P, can be proven:

Lemma 5 *For trace $r \in B^*$, predicate R:*

$$\begin{aligned} \overline{wp}(Q_{\langle\imath\rangle^\frown r}, R) \quad = \quad & (\exists s \in A^* \cdot s \upharpoonright B = r \wedge \overline{wp}(P_{\langle\imath\rangle^\frown s}, R)) \\ & \vee(\exists u \in A^\omega \cdot u \upharpoonright B \leq r \wedge \overline{inf}(P_{\langle\imath\rangle^\frown u})). \end{aligned}$$

Here, $s \upharpoonright B$ represents the trace s restricted to elements of B only. In the second disjunct of the right-hand side of this lemma, $u \upharpoonright B \leq r$ means that $u \upharpoonright B$ is a subsequence of the finite sequence r, and thus that u ends in an infinite sequence of C-events.

There is a similar lemma for infinite traces which may be proven using Definition 10:

Lemma 6 *For trace $v \in B^\omega$:*

$$\overline{inf}(Q_{\langle\imath\rangle^\frown v}) \quad = \quad (\exists u \in A^\omega \cdot u \upharpoonright B \leq v \wedge \overline{inf}(P_{\langle\imath\rangle^\frown u})).$$

The following lemma, which relates guardedness conditions in Q to guardedness conditions in P, may be proven using Lemma 1:

Lemma 7 *For label set $X \subseteq B$:*

$$\overline{wp}(IT_C, \neg gd(Q_X)) \ = \ \overline{wp}(IT_C, \neg gd(P_{X \cup C})).$$

CSP hiding is defined by the semantic function presented in Figure 5, taken from [RB89]. This describes the failures, divergences, and infinites of $\Theta \setminus C$ in terms of those of Θ. Using the lemmas just presented, we can show that the failures, divergences, and infinites of action system Q may be described in terms of those of P using the exact same function:

Theorem 4 *For action system $P = (A, P_\imath, P_A)$, $C \subseteq A$:*

$$\{\!\!\{P \setminus C\}\!\!\} = \{\!\!\{P\}\!\!\} \setminus C$$

in the infinite traces model.

For process model $\Theta = (A, F, D, I)$, $C \subseteq A$, let $B \mathrel{\widehat{=}} (A - C)$. Then:

$$\Theta \setminus C \quad \widehat{=} \quad (B,\ \Theta \setminus_{\mathcal{F}} C,\ \ \Theta \setminus_{\mathcal{D}} C,\ \ \Theta \setminus_{\mathcal{I}} C),$$

where $\Theta \setminus_{\mathcal{F}} C$ are those $r \in B^*$, $X \subseteq B$ satisfying:

$$(\exists s \in A^* \cdot s \restriction B = r \wedge (s, X \cup C) \in F) \ \vee\ (\exists u \in A^\omega \cdot u \restriction B \leq r \wedge u \in I),$$

$\Theta \setminus_{\mathcal{D}} C$ are those $r \in B^*$ satisfying:

$$(\exists s \in A^* \cdot s \restriction B = r \wedge s \in D) \ \vee\ (\exists u \in A^\omega \cdot u \restriction B \leq r \wedge u \in I),$$

and $\Theta \setminus_{\mathcal{I}} C$ are those $v \in B^\omega$ satisfying:

$$(\exists u \in A^\omega \cdot u \restriction B \leq v \wedge u \in I).$$

Figure 5: Definition of CSP hiding in the infinite traces model.

Proof: To show failures equality we have, for $r \in B^*$, $X \subseteq B$,

$$\begin{array}{lll}
& (r, X) \in \mathcal{F}\{\!\![P \setminus C]\!\!\} & \\
= & \overline{wp}(Q_{\langle\iota\rangle^\frown r}, \neg gd(Q_X)) & \textit{Definition 11} \\
= & \overline{wp}(Q_{\langle\iota\rangle^\frown r}, \overline{wp}(IT_C, \neg gd(Q_X)\ )) & \textit{Lemma 3} \\
= & \overline{wp}(Q_{\langle\iota\rangle^\frown r}, \overline{wp}(IT_C, \neg gd(P_{X\cup C})\ )) & \textit{Lemma 7} \\
= & \overline{wp}(Q_{\langle\iota\rangle^\frown r}, \neg gd(P_{X\cup C})\ ) & \textit{Lemma 3} \\
= & (\exists s \in A^* \cdot s \restriction B = r \wedge \overline{wp}(P_{\langle\iota\rangle^\frown s}, \neg gd(P_{X\cup C})\ )) & \\
& \vee(\exists u \in A^\omega \cdot u \restriction B \leq r \wedge \overline{inf}(P_{\langle\iota\rangle^\frown u})) & \textit{Lemma 5} \\
= & (\exists s \in A^* \cdot s \restriction B = r \wedge (s, X \cup C) \in \mathcal{F}\{\!\![P]\!\!\}) & \\
& \vee(\exists u \in A^\omega \cdot u \restriction B \leq r \wedge u \in \mathcal{I}\{\!\![P]\!\!\}) & \textit{Definitions 6,11} \\
= & (r, X) \in \{\!\![P]\!\!\} \setminus_{\mathcal{F}} C & \textit{Figure 5.}
\end{array}$$

Similarly, using Lemma 5 with $R = false$ we get, for $r \in B^*$:

$$r \in \mathcal{D}\{\!\![P \setminus C]\!\!\} \ = \ r \in \{\!\![P]\!\!\} \setminus_{\mathcal{D}} C,$$

and using Lemma 6 we get, for $v \in B^\omega$:

$$v \in \mathcal{I}\{\!\![P \setminus C]\!\!\} \ = \ v \in \{\!\![P]\!\!\} \setminus_{\mathcal{I}} C.$$

$\square$

11 Related Work

Woodcock & Morgan [WM90] have shown how CSP refinement between action systems, $P \sqsubseteq P'$, may be checked using *simulation*. This involves comparison of similarly labelled actions of P and P', rather than having to reason about traces of actions. In [But92], as well as the hiding operator described here, a parallel operator for action systems is defined, in which interaction between systems is based on synchronisation over shared actions. This operator is shown to correspond to CSP parallel composition in the manner of Theorem 4. The notion of interaction is also extended so that action systems may pass values on synchronisation in the manner of CSP channels.

Josephs [Jos88] and He Jifeng [He89] define the CSP failures of action systems in which actions are relations between states rather than predicate transformers. He Jifeng defines divergences as well. Both define hiding in the relational approach, though neither treats hiding of unboundedly nondeterministic actions adequately. In [He89], the divergent state needs to be modelled explicitly, which complicates the definition of the hiding operator. This is avoided, in our case, by the use of predicate transformers rather than relations.

In Back's approach to action systems [BKS83, Bac90], more emphasis is placed on the state of an action system rather than its action labels. An action system is modelled by a set of state-traces describing the possible evolution of the state. Back shows that simulation between action systems implies state-trace refinement. He describes a special form of simulation in which the refined action system has extra actions over the abstract one [Bac90]. In terms of our CSP approach, this is akin to showing $P \sqsubseteq (P' \setminus C)$, where C represents the extra actions of the refined action system. Back also defines a parallel operator for action systems in which interaction is based on shared variables rather than shared actions [Bac90].

Back has developed a characterisation of the DO-loop that is similar to our characterisation of the IT-loop in Theorem 3 [Bac89, Bac92]. However, his characterisation is valid only for boundedly nondeterministic statements. Back's characterisation may be written as follows:

$$wp(\mathbf{do}\ S\ \mathbf{od}, R) \quad = \quad (\forall i \geq 0 \cdot wp(S^i, gd(S) \vee R)) \wedge (\exists i \geq 0 \cdot wp(S^i, false)).$$

This may be generalised to deal with unbounded nondeterminism using our $\overline{inf}$ as follows:

$$wp(\mathbf{do}\ S\ \mathbf{od}, R) \quad = \quad (\forall i \geq 0 \cdot wp(S^i, gd(S) \vee R)) \wedge \neg\overline{inf}(S^\infty).$$

The *occam* programming language is based on CSP [JG88]. Back & Sere [BS90] have shown how a certain class of action system may be implemented as occam programs. However, their correspondence between action systems and occam is informal, in contrast to our correspondence between action systems and CSP.

12 Conclusions

In the case where an action system is unboundedly nondeterministic, the correspondence between the CSP failures-divergences model and action systems of [Mor90a] is lost after action hiding is performed. We introduced the formula $\overline{inf}(\Sigma)$ for infinitary actions Σ which characterises those initial states in which all of Σ may be executed. This allowed us to show the correspondence between action systems and the CSP infinite traces model.

The new correspondence then allowed us to show that action system hiding corresponds to CSP hiding even for unboundedly nondeterministic action systems. When developing this proof, $\overline{inf}$ provided a very natural way of describing when divergence may be introduced by action system hiding (see Lemma 5).

Specification statements provide a convenient way of describing actions. Restricting these to bounded nondeterminism places a proof burden on the specifier. Also, unbounded nondeterminism can be introduced into an action system if an infinite number of actions are hidden. If we wish to describe CSP processes using the action system formalism, we are no longer restricted to bounded nondeterminism or to hiding of only finitely many actions.

In addition, dealing properly with infinite behaviour through the introduction of $\overline{inf}$ may lead to improvements in the way we can treat issues such as fairness and probabilistic computations.

Appendix

Lemma 8 $\overline{inf}(S^\infty) \;=\; (\bar{\mu}\, X \cdot \overline{wp}(S, X))$.

Proof by mutual entailment:

(a). By Lemma 2, $\overline{inf}(S^\infty)$ is a solution of:

$$X \;\Rightarrow\; \overline{wp}(S, X).$$

Hence, by Knaster-Tarski, $\overline{inf}(S^\infty) \;\Rightarrow\; (\bar{\mu}\, X \cdot \overline{wp}(S, X))$.

(b). Let $\Phi \;\widehat{=}\; \langle\; (\bar{\mu}\, X \cdot \overline{wp}(S, X)) \mid i \geq 0 \;\rangle$. Now Φ is a solution of:

$$X \;\Rightarrow\; \overline{wp}_\infty(S^\infty, X).$$

Hence, by Knaster-Tarski, $\Phi \;\Rightarrow\; (\bar{\mu}\, X \cdot \overline{wp}_\infty(S^\infty, X))$, and, by Definition 10, $(\bar{\mu}\, X \cdot \overline{wp}(S, X)) \;\Rightarrow\; \overline{inf}(S^\infty)$.

□

As a corollary to Lemma 8, we have:

Lemma 9 $\neg\overline{inf}(S^\infty) \;=\; (\mu X \cdot wp(S, X))$.

For continuous f, it can be shown that:

$$(\mu X \cdot f(X)) \;=\; (\exists i \geq 0 \cdot f^i(false)),$$

where f^i is application of f i times. Hitchcock & Park [HP72] have improved on this in their *Generalised Limit Theorem*, by extending the range of i to the ordinals so that continuity is no longer required. We use the version presented by Nelson [Nel89]:

Theorem 5 (Generalised Limit Theorem) *Let f be a monotonic function, and for ordinal α, let f^α be defined inductively by:*

$$f^\alpha \quad \widehat{=} \quad (\exists \beta \cdot \beta < \alpha \wedge f(f^\beta)).$$

Then f has a least *fixed-point given by f^α, for some ordinal α.*

Now, let $f(X) \mathrel{\widehat{=}} wp(S, X)$, $g(X) \mathrel{\widehat{=}} R \wedge f(X)$, and $C \mathrel{\widehat{=}} (\forall i \geq 0 \cdot wp(S^i, R))$. Assume f^α and g^α are as defined in Theorem 5. Then we have lemmas as follows:

Lemma 10 *For any ordinal α,* $\qquad g(C \wedge f^\alpha) \ = \ C \wedge f^{\alpha+1}.$

Proof:

$$
\begin{array}{lll}
 & g(C \wedge f^\alpha) & \\
= & R \wedge wp(S, (\forall i \geq 0 \cdot wp(S^i, R)) \wedge f^\alpha) & \textit{definition of } g \textit{ and } C \\
= & R \ \wedge \ wp(S, (\forall i \geq 0 \cdot wp(S^i, R))) \ \wedge \ wp(S, f^\alpha) & \textit{conjunction} \\
= & R \wedge (\forall i \geq 0 \cdot wp(S^{i+1}, R)) \wedge f^{\alpha+1} & \textit{conjunction} \\
= & wp(S^0, R) \wedge (\forall i \geq 0 \cdot wp(S^{i+1}, R)) \wedge f^{\alpha+1} & S^0 = \mathbf{skip} \\
= & C \wedge f^{\alpha+1} & \textit{join conjuncts}
\end{array}
$$

□

Lemma 11 *For each ordinal α,* $\qquad g^\alpha \ = \ C \wedge f^\alpha.$

Proof by ordinal induction:

$$
\begin{array}{rcll}
g^\alpha & = & (\exists \beta \cdot \beta < \alpha \wedge g(g^\beta)) & \textit{definition of } g^\alpha \\
 & = & (\exists \beta \cdot \beta < \alpha \wedge g(C \wedge f^\beta)) & \textit{induction hypothesis} \\
 & = & (\exists \beta \cdot \beta < \alpha \wedge C \wedge f^{\beta+1}) & \textit{Lemma 10} \\
 & = & C \wedge (\exists \beta \cdot \beta < \alpha \wedge f(f^\beta)) & C \textit{ is independent of } \beta \\
 & = & C \wedge f^\alpha & \textit{definition of } f^\alpha
\end{array}
$$

□

Lemma 12 *For ordinal α, if f^α is a fixed-point of f, then g^α is a fixed-point of g.*

Proof: Follows from Lemma 11.

Theorem 6 $wp(\textbf{it}\ S\ \textbf{ti}, R) \ = \ (\forall i \geq 0 \cdot wp(S^i, R)) \ \wedge \ \neg\overline{inf}(S^\infty)$.

Proof:

$$
\begin{array}{lll}
 & C \wedge \neg\overline{inf}(S^\infty) & \\
= & C \wedge f^\alpha & \textit{by Lemma 9 and Theorem 5, where } f^\alpha \ = \ f^{\alpha+1} \\
= & g^\alpha & \textit{Lemma 11} \\
= & wp(\textbf{it}\ S\ \textbf{ti}, R) & \textit{by Lemma 12, Theorem 5, Lemma 1}
\end{array}
$$

□

Theorem 3 $\overline{wp}(\textbf{it}\ S\ \textbf{ti}, R) \ = \ (\exists i \geq 0 \cdot \overline{wp}(S^i, R)) \ \vee \ \overline{inf}(S^\infty)$.

Proof: Conjugate of Theorem 6.

Acknowledgements

Thanks to the anonymous referees for their constructive comments.

References

[Bac80] R.J.R. Back. *Correctness Preserving Program Refinements: Proof Theory and Applications.* Tract 131, Mathematisch Centrum, Amsterdam, 1980.

[Bac89] R.J.R. Back. Refining atomicity in parallel algorithms. In *PARLE'89*, volume LNCS 366. Springer–Verlag, 1989.

[Bac90] R.J.R. Back. Refinement calculus II: Parallel and reactive systems. In J.W. de Bakker, W.P. de Roever, and G. Rozenberg, editors, *Stepwise Refinement of Distributed Systems*, volume LNCS 430. Springer–Verlag, 1990.

[Bac92] R.J.R. Back. Refinement calculus, lattices and higher order logic. In *Marktoberdorf International Summer School — Program Design Calculi*, July 1992.

[BKS83] R.J.R. Back and R. Kurki-Suonio. Decentralisation of process nets with centralised control. In *2nd ACM SIGACT-SIGOPS Symp. on Principles of Distributed Computing*, pages 131–142, 1983.

[BS90] R.J.R. Back and K. Sere. Deriving an occam implementation of action systems. In C.C. Morgan and J.C.P. Woodcock, editors, *3rd BCS-FACS Refinement Workshop*. Springer–Verlag, 1990.

[But92] M.J. Butler. *A CSP Approach To Action Systems.* D.Phil. Thesis, Programming Research Group, Oxford University, 1992.

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

[He89] J. He. Process refinement. In J. McDermid, editor, *The Theory and Practice of Refinement.* Butterworths, 1989.

[Hoa85] C.A.R. Hoare. *Communicating Sequential Processes.* Prentice–Hall, 1985.

[HP72] P. Hitchcock and D. Park. Induction rules and termination proofs. In *IRIA Conference on Automata Languages and Programming Theory*, France, 1972.

[JG88] G. Jones and M. Goldsmith. *Programming in occam 2.* Prentice–Hall, 1988.

[Jos88] M.B. Josephs. A state-based approach to communicating sequential processes. *Distributed Computing*, 3:9–18, 1988.

[Mor79] C.C. Morgan. *Parallel Programming Without Synchronisation.* Ph.D. Thesis, University of Sydney, 1979.

[Mor88] C.C. Morgan. The specification statement. *ACM Trans. Prog. Lang. and Sys.*, 10(3), 1988. Reprinted in [MRG88].

[Mor90a] C.C. Morgan. Of wp and CSP. In W.H.J. Feijen, A.J.M. van Gasteren, D. Gries, and J. Misra, editors, *Beauty is our business: a birthday salute to Edsger W. Dijkstra.* Springer–Verlag, 1990.

[Mor90b] C.C. Morgan. *Programming from Specifications.* Prentice–Hall, 1990.

[MRG88] C.C. Morgan, K.A. Robinson, and P.H.B. Gardiner. *On the Refinement Calculus.* Technical Monograph PRG–70, Programming Research Group, Oxford University, October 1988.

[Mrr87] J.M. Morris. A theoretical basis for stepwise refinement and the programming calculus. *Sci. Comp. Prog.*, 9(3):298–306, 1987.

[Nel89] G. Nelson. A generalization of Dijkstra's calculus. *ACM Trans. Prog. Lang. Sys.*, 11(4):517–561, 1989.

[RB89] A.W. Roscoe and G. Barrett. *Unbounded Nondeterminism in CSP.* In *MFPS '89*, volume LNCS 298. Springer–Verlag, 1989. To appear in *Journal of Logic and Computation.*

[Ros88] A.W. Roscoe. *Unbounded Nondeterminism in CSP.* Technical Monograph PRG-67, Programming Research Group, Oxford University, July, 1988.

[Tar55] A. Tarski. A lattice-theoretic fixpoint theorem and its applications. *Pacific Journal Math.*, 5, 1955.

[WM90] J.C.P. Woodcock and C.C. Morgan. Refinement of state-based concurrent systems. In D. Bjørner, C.A.R. Hoare, and H. Langmaack, editors, *VDM '90*, volume LNCS 428. Springer–Verlag, 1990.