Process Algebra with Partial Choice

J.C.M. Baeten
Department of Computer Science, Eindhoven University of Technology,
P.O.Box 513, 5600 MB Eindhoven, The Netherlands

J.A. Bergstra
Programming Research Group, University of Amsterdam,
Kruislaan 403, 1098 SJ Amsterdam, The Netherlands
and
Department of Philosophy, Utrecht University,
Heidelberglaan 8, 3584 CS Utrecht, The Netherlands

The objective of this paper is to bridge the gap between ACP and TCSP. To this end, ACP is extended with two non-deterministic choice operators in a setting of bisimulation semantics. With these operators, we can express safety properties of systems without the use of silent steps, and we can verify safety properties in a setting in which no assumption on fairness (or unfairness) has been made.

1980 Mathematics Subject Classification (1985 revision): 68Q55, 68Q10, 68Q45.
1987 CR Categories: F.1.2, D.3.1, F.3.1, D.1.3.
Key words & Phrases: non-determinism, choice, partial choice, safety property.
Note: This research was supported in part by ESPRIT basic research action 7166, CONCUR2. The second author is also partially supported by ESPRIT basic research action 6454, CONFER.

1. INTRODUCTION.

The objective of this paper is to bridge the gap between ACP [BEK84, BAW90] and TCSP [BRHR84, HOA85]. Earlier, in [BEKO88] and [BABK87], ready, failure and ready trace semantic models have been provided for ACP. However, these models do not take into account TCSP's rather sophisticated views on non-determinism versus choice. We agree with the observation underlying TCSP that externally influencible choice and non-determinism are to be distinguished and that more than one operator is needed.

After many experiments, we have concluded that it may be useful to extend ACP with two more operators:

- $\dashv\!\vdash$ *static* or *partial* alternative composition
- $\sqcup$ *collecting* alternative composition.

These operators exist together with the usual alternative composition operator

$+$ called *dynamic* alternative composition here.

A choice $p + q$ is made exactly when the first action of p or q is executed, so the choice is made at the root of the system. On the other hand, a choice $p \sqcup q$ is made a long time ago, before any action is executed in the system whatsoever. Finally, a choice $p \dashv\!\vdash q$ is somewhere in between the previous actions, but we don't know where; $p \dashv\!\vdash q$ behaves somewhere between $p \sqcup q$ and $p + q$.

The choice $p \pitchfork q$ is called static because when control arrives at its root state probably just one option is left; the choice $p \sqcup q$ is called collecting because it represents $\{p, q\}$ (with the notational convention in mind that $p = \{p\}$). The choice $p \pitchfork q$ is called partial because only partial information about the timing of the choice is available.

In [BAB94] we have provided a connection between the communication mechanisms of CCS (see [MIL80]) and ACP. This paper complements that one. It deviates from [BKO88] by being based on bisimulation entirely. In this way we hope to have it both ways: combine TCSP's sophistication concerning choice and non-determinism with the mathematical clarity of bisimulations, the cornerstone of CCS.

Further we find that the introduction of partial choice and collecting choice allows us in many cases to express safety properties of systems without the use of silent steps and to verify safety properties in a setting in which no assumption about fairness (or unfairness) has yet been made. Avoiding silent steps as well as fairness considerations are virtues of TCSP that we consider advantages indeed. We hope to have extended ACP in such a way that these virtues have become available in the ACP setting as well.

Moreover, our introduction of partial choice may shed new light on probabilistic choice (see [BABS92]) and the combination of probabilistic choice with other choice operators (cf. [JON93]).

Finally, our development is such that in the absence of silent steps, a model can also be found along the lines of the projective limit model of [BEK84], which is in fact an alternative presentation of the metric space model of [BAZ82].

2. PROCESS ALGEBRA WITH PARTIAL CHOICE.

We start by extending process algebra with the partial choice operator $\pitchfork$.

2.1 BASIC PROCESS ALGEBRA.

Assume that we have a (finite) set of constants $A = \{a,b,c,...\}$ and a special constant $\delta \notin A$. Further, we have the binary operators $+$ (dynamic choice), $\pitchfork$ (partial choice), $\cdot$ (sequential composition). The axioms for $+,\cdot$ are the standard axioms for BPA_δ, except that axiom A3 ($X + X = X$) is restricted to atomic actions. See table 1 ($a \in A$). A3 is restricted, because it does not hold anymore for processes involving the new choice operator. Note that the axioms in table 1 still provide a complete axiomatisation of the standard model of BPA_δ. This is a consequence of the following lemma.

$X + Y = Y + X$	A1
$(X+ Y) + Z = X + (Y + Z)$	A2
$a + a = a$	AA3
$(X + Y)\cdot Z = X\cdot Z + Y\cdot Z$	A4
$(X\cdot Y)\cdot Z = X\cdot(Y\cdot Z)$	A5
$X + \delta = X$	A6
$\delta\cdot X = \delta$	A7

TABLE 1. BPA_δ with restricted A3.

2.2 LEMMA. Let t be a closed term over BPA_δ. Then A1,2,4,6, AA3 $\vdash t + t = t$.
PROOF: We use induction on the structure of basic terms (see e.g. [BAW90]).
i. Constants: for atoms, this is AA3, for δ it follows from A6.
ii. Prefix multiplication: $a\cdot x + a\cdot x = (a + a)\cdot x = a\cdot x$.
iii. Dynamic sum: $(x + y) + (x + y) = x + (y + (x + y)) = x + ((x + y) + y) = x + (x + (y + y)) =$
$= (x + x) + (y + y) = x + y$.

2.3 PARTIAL SUM.
The partial alternative composition operator has the same laws as the dynamic alternative composition operator, except that we have the full analogue of axiom A3 of BPA ($X + X = X$), but no A6. As far as mutual distributivity of alternative composition operators is concerned, partial sum comes before dynamic sum. The partial sum operator obeys exactly the laws of the sum operator in [BABS92]. Thus it is the partial sum operator that can be extended to a probabilistic sum.

We call the basic theory with the two alternative composition operators $BPA_{\amalg}$.

$X \amalg Y = Y \amalg X$	PAC1
$(X \amalg Y) \amalg Z = X \amalg (Y \amalg Z)$	PAC2
$X \amalg X = X$	PAC3
$(X \amalg Y)\cdot Z = X\cdot Z \amalg Y\cdot Z$	PAC4
$(X \amalg Y) + Z = (X + Z) \amalg (Y + Z)$	PAC5

TABLE 2. Partial alternative composition.

2.4 REALISATION.
Now we want to express a notion of realisation or implementation. We use a partial order $\leq$ in order to express this. Thus, if $X \leq Y$, then X *realises* Y, i.e. X has less static non-determinism than Y. The central law is PAC6 in table 3.

$X \leq X \amalg Y$	PAC6

TABLE 3. Realisation.

Thus, our axiom system contains equations and inequalities. We assume the standard properties of equality in reasoning about systems of equations. Similarly, we have the following properties for reasoning about inequalities:

$X \leq X$	reflexivity
$X \leq Y \wedge Y \leq X \Leftrightarrow X = Y$	antisymmetry
$X \leq Y \wedge Y \leq Z \Rightarrow X \leq Z$	transitivity
$X \leq Y \Rightarrow C[X] \leq C[Y]$	closed under contexts
$X \leq Y \Rightarrow \rho(X) \leq \rho(Y)$	substitutivity (ρ a mapping from variables

to terms), i.e. we have a *partial order algebra* in the terminology of [HEN88].

2.5 LEMMA. For all closed terms t over $BPA_{\barwedge}$ we have $t \leq t + t$.

PROOF: For terms over BPA_δ, this is a consequence of lemma 2.2. It remains to consider the partial sum operator. By induction, assume the lemma holds for x,y, then:

$$x \barwedge y \leq x \barwedge y \barwedge (x + y) \barwedge (y + x) \leq (x + x) \barwedge (y + x) \barwedge (x + y) \barwedge (y + y) =$$
$$= ((x \barwedge y) + x) \barwedge ((x \barwedge y) + y) = (x + (x \barwedge y)) \barwedge (y + (x \barwedge y)) = (x \barwedge y) + (x \barwedge y).$$

We can give an intuition for this inequality as follows: $t+t$ has more non-determinism than t, because different non-deterministic choices can have been made in the different copies of t.

2.6 SEMANTICS.

We define a process graph model, similar in spirit to the stratified model of [HAJ90, HAN91]. We distinguish between a partial sum branching and a dynamic sum branching.

We define a *p-graph* to be a quintuple $\langle S, D, \rightarrow, \dashrightarrow, r\rangle$ such that:

- S is the set of *static* or *partial states*
- D is the set of *dynamic states*; S and D are disjoint
- $\rightarrow \subseteq D \times A \times S$ is the *labeled transition relation*, denoting dynamic choice
- $\dashrightarrow \subseteq S \times D$ is the *unlabeled transition relation*, denoting static choice
- $r \in S$ is the *root node.*

Define $\mathbb{PG}$ to be the set of all p-graphs that satisfy the following restrictions:

i. S and D are countable

ii. the root node has at least one outgoing transition.

A static state without outgoing transitions is called an *end node*. A dynamic state without outgoing transitions is called a *deadlock node.* We write $s \dashrightarrow d$ iff $\langle s,d\rangle \in \dashrightarrow$, and $d \xrightarrow{a} s$ iff $\langle d,a,s\rangle \in \rightarrow$. The names of the states do not matter, so actually we are considering equivalence classes of p-graphs under graph isomorphism. This assumption enables us to assume that state sets are disjoint whenever this is convenient.

The interpretation function will assign a p-graph from $\mathbb{PG}$ to each process. We proceed to define the interpretation. First, constants.

$[\![a]\!] = \langle\{0,2\}, \{1\}, \{\langle 1,a,2\rangle\}, \{\langle 0,1\rangle\}, 0\rangle$

$[\![\delta]\!] = \langle\{0\}, \{1\}, \varnothing, \{\langle 0,1\rangle\}, 0\rangle$.

Fig. 1 illustrates these definitions. We use the following pictorial conventions: closed circles denote static states, open circles dynamic states, and a small incoming arrow denotes the root node.

FIGURE 1. Constants.

Partial sum. We define $g \barwedge h$, for two p-graphs g,h. Let g,h be given with disjoint state sets. Then $g \barwedge h$ is obtained by creating a new root node r, and adding, for each unlabeled transition $s \dashrightarrow d$ where s is the root of g or h, a new unlabeled transition $r \dashrightarrow d$.

Dynamic sum. Let two p-graphs g,h be given with disjoint state sets. We obtain $g+h$ by creating a new root node r, and creating a new dynamic node $\langle d,e\rangle$ for each pair consisting of a dynamic node d in g reachable from the root by an unlabeled arrow, e a similar node in h. We have unlabeled transitions $r \mapsto \langle d,e\rangle$ for each such new dynamic node, and a labeled transition $\langle d,e\rangle \xrightarrow{a} s$ whenever there is a transition $d \xrightarrow{a} s$ or $e \xrightarrow{a} s$. As an example, we show $(a \leftarrow\!\sqcup\ b) + c$ in fig.2 (omitting unreachable nodes).

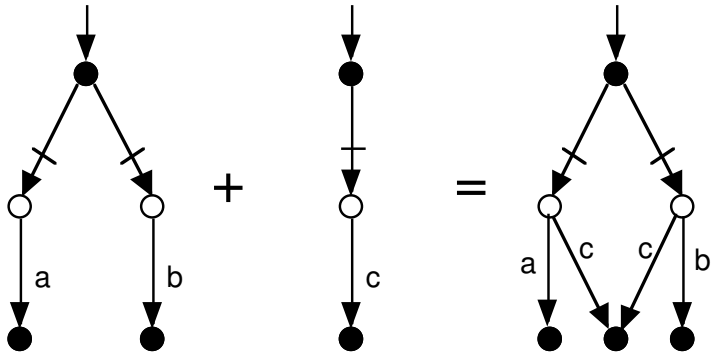

FIGURE 2. Dynamic sum.

Sequential composition. Let two p-graphs g,h be given with disjoint state sets. We obtain $g \cdot h$ by appending a copy of h to each endnode of g.

2.6 STRUCTURED OPERATIONAL SEMANTICS.

$$a \mapsto \boxed{a} \qquad \boxed{a} \xrightarrow{a} \surd \qquad \delta \mapsto \boxed{\delta}$$

$$\frac{x \mapsto \boxed{x'}}{x \cdot y \mapsto \boxed{x' \cdot y}} \qquad \frac{\boxed{x} \xrightarrow{a} x'}{\boxed{x \cdot y} \xrightarrow{a} x' \cdot y} \qquad \frac{\boxed{x} \xrightarrow{a} \surd}{\boxed{x \cdot y} \xrightarrow{a} y}$$

$$\frac{x \mapsto \boxed{x'},\ y \mapsto \boxed{y'}}{x+y \mapsto \boxed{x'+y'}}$$

$$\frac{\boxed{x} \xrightarrow{a} x'}{\boxed{x+y} \xrightarrow{a} x',\ \boxed{y+x} \xrightarrow{a} x'} \qquad \frac{\boxed{x} \xrightarrow{a} \surd}{\boxed{x+y} \xrightarrow{a} \surd,\ \boxed{y+x} \xrightarrow{a} \surd}$$

$$\frac{x \mapsto \boxed{x'}}{x \leftarrow\!\sqcup y \mapsto \boxed{x'},\ y \leftarrow\!\sqcup x \mapsto \boxed{x'}}$$

TABLE 4. Operational semantics of closed terms over $BPA_{\leftarrow\sqcup}$.

We can also obtain a p-graph for each closed term by means of a structured operational semantics. We show the rules in table 4. Static states are given by terms (over $BPA_{\barwedge}$), dynamic states by boxed terms (over BPA_{δ}). Modulo the definition of bisimulation that follows, the structured operational semantics yields the same graphs as the definitions above. We will omit the proof of this fact.

2.7 PARTIAL BISIMULATION.
Let two p-graphs g,h be given, and let R be a relation between nodes of g and nodes of h. R is a *partial bisimulation* if:

i. the roots are related
ii. if $R(s,t)$ and $s \rightarrow\!\!\!\!\bullet\ d$ then there is node e in h such that $t \rightarrow\!\!\!\!\bullet\ e$ and $R(d,e)$
iii. if $R(d,e)$ and $d \xrightarrow{a} s$ then there is node t in h such that $e \xrightarrow{a} t$ and $R(s,t)$
iv. if $R(d,e)$ and $e \xrightarrow{a} t$ then there is node s in g such that $d \xrightarrow{a} s$ and $R(s,t)$
v. if $R(s,t)$ then s is an end node iff t is an end node.

We say that g is *partially bisimilar* to h, denoted $g \underline{\leftrightarrow} h$, if there is a partial bisimulation from g to h.

We say that two p-graphs g,h are *bisimulation equivalent* if $g \underline{\leftrightarrow} h$ and $h \underline{\leftrightarrow} g$. It is easy to see that this is indeed an equivalence relation. Denote by $\mathbb{PG}/\underline{\leftrightarrow}$ the set of equivalence classes. Put $\mathbb{PG}/\underline{\leftrightarrow} \models s \leq t$ iff $[\![s]\!] \underline{\leftrightarrow} [\![t]\!]$ and $\mathbb{PG}/\underline{\leftrightarrow} \models s = t$ if $[\![s]\!]$ and $[\![t]\!]$ are bisimulation equivalent.

2.8 THEOREM (SOUNDNESS). Let s,t be closed terms. Then $BPA_{\barwedge} \vdash s \leq t \Rightarrow [\![s]\!] \underline{\leftrightarrow} [\![t]\!]$.
HINT OF PROOF: First we need to prove that $\mathbb{PG}/\underline{\leftrightarrow}$ is a partial order algebra. Then, it is sufficient to check each axiom separately.

We proceed to investigate the converse of theorem 2.8, i.e. the completeness of our axiomatisation. In order to do this, we first need the notion of a basic term.

2.9 DEFINITION. We define the set of basic terms $\mathcal{B}$ inductively, with the help of an intermediary set $\mathcal{B}_+$. In $\mathcal{B}$, the top operator is a partial sum and in $\mathcal{B}_+$ a dynamic sum.

i. $A \cup \{\delta\} \subseteq \mathcal{B}_+ \subseteq \mathcal{B}$
ii. $a \in A, t \in \mathcal{B} \Rightarrow a \cdot t \in \mathcal{B}_+$
iii. $t, s \in \mathcal{B}_+ \Rightarrow t + s \in \mathcal{B}_+$
iv. $t, s \in \mathcal{B} \Rightarrow t \barwedge s \in \mathcal{B}$.

An example of a basic term: $(a \cdot (b \barwedge (c + d)) + b) \barwedge \delta$.

Let p be a process in $BPA_{\barwedge}$. We say p has a *dynamic head normal form* ($p \in$ HNF+) if there are $n,m \in \mathbb{N}$ with $n + m > 0$, $a_1, ..., a_n \in A$, $b_1, ..., b_m \in A \cup \{\delta\}$ and processes $p_1, ..., p_n$ such that

$$BPA_{\barwedge} \vdash p = a_1 \cdot p_1 + ... + a_n \cdot p_n + b_1 + ... + b_m.$$

We say a process q has a *partial head normal form* ($q \in$ HNF$\barwedge$) if there is $k > 0$ and processes $q_1, ..., q_k$ in dynamic head normal form such that

$$BPA_{\barwedge} \vdash q = q_1 \barwedge ... \barwedge q_k.$$

Note that $\mathbb{PG}/\underline{\leftrightarrow} \models p = p + p$ holds for all $p \in$ HNF+. It is easy to see that this law does not hold for all processes, as the counterexample $p \equiv a \barwedge b$ shows.

2.10 THEOREM: Let s be a closed term. Then there is a basic term t such that $BPA_{\barwedge} \vdash s = t$.

HINT OF PROOF: By term rewrite analysis. It is straightforward to show that the term rewrite system shown in table 5 is confluent and normalising. Thus, each term has a unique normal form. Each normal form is a basic term. The reduction to normal form constitutes a proof in $BPA_{\between}$.

$(X + Y)\cdot Z \to X\cdot Z + Y\cdot Z$	RA4
$(X\cdot Y)\cdot Z \to X\cdot(Y\cdot Z)$	RA5
$\delta\cdot X \to \delta$	RA7
$(X \between Y)\cdot Z \to X\cdot Z \between Y\cdot Z$	RPAC4
$(X \between Y) + Z \to (X + Z) \between (Y + Z)$	RPAC5
$X + (Y \between Z) \to (X + Y) \between (X + Z)$	RPAC5'

TABLE 5. Rewrite system.

2.11 THEOREM (COMPLETENESS). Let s,t be closed terms. Then $[\![s]\!] \leftrightarrows [\![t]\!] \Rightarrow BPA_{\between} \vdash s \le t$.
HINT OF PROOF: By 2.10, it is enough to consider basic terms. Using the structured operational semantics of 2.6, we show that for each basic term t we have:

- $t \overset{a}{\to} t' \Leftrightarrow t' \in \mathcal{B} \wedge \exists t'' \in \mathcal{B} \exists s,s' \in \mathcal{B}_+ \ ((PAC1,2 \vdash t = t'' \between s \vee PAC1,2 \vdash t = s) \wedge (A1,2 \vdash s = s' + a\cdot t' \vee A1,2 \vdash s = a\cdot t'))$
- $t \overset{a}{\to} \surd \Leftrightarrow \exists t'' \in \mathcal{B} \exists s,s' \in \mathcal{B}_+ \ ((PAC1,2 \vdash t = t'' \between s \vee PAC1,2 \vdash t = s) \wedge (A1,2 \vdash s = s' + a \vee A1,2 \vdash s = a))$

On the basis of these equivalences, the theorem follows easily.

2.12 ADDITIONAL OPERATORS.
We sketch the extension of $BPA_{\between}$ with additional operators. First, free merge. In table 6, $a \in A \cup \{\delta\}$.

$X \between Y = X \between Y + Y \between X$
$a \between X = a\cdot X$
$a\cdot X \between Y = a\cdot(X \between Y)$
$(X \Box Y) \between Z = X \between Z \Box Y \between Z$ for $\Box \in \{+, \between\}$

TABLE 6. Additional axioms for $PA_{\between}$.

Next, merge with communication, and encapsulation. In table 7, $a \in A \cup \{\delta\}$.

$a \mid b = b \mid a$	
$(a \mid b) \mid c = a \mid (b \mid c)$	
$a \mid \delta = \delta$	
$X \between Y = X \between Y + Y \between X + X \mid Y$	
$a \between X = a \cdot X$	
$a \cdot X \between Y = a \cdot (X \between Y)$	
$(X \Box Y) \between Z = X \between Z \Box Y \between Z$	for $\Box \in \{+, \leftthreetimes\}$
$a \cdot X \mid b = (a \mid b) \cdot X$	
$a \mid b \cdot X = (a \mid b) \cdot X$	
$a \cdot X \mid b \cdot Y = (a \mid b) \cdot (X \between Y)$	
$(X \Box Y) \mid Z = X \mid Z \Box Y \mid Z$	for $\Box \in \{+, \leftthreetimes\}$
$X \mid (Y \Box Z) = X \mid Y \Box X \mid Z$	for $\Box \in \{+, \leftthreetimes\}$
$\partial_H(a) = a$	if $a \notin H$
$\partial_H(a) = \delta$	if $a \in H$
$\partial_H(X \Box Y) = \partial_H(X) \Box \partial_H(Y)$	for $\Box \in \{+, \cdot, \leftthreetimes\}$

TABLE 7. Additional axioms for $ACP_{\leftthreetimes}$.

2.13 RECURSION.

Finally, we will consider systems of recursive equations. Treatment of these is standard, see e.g. [BAW90]. We will only consider completely guarded equations in the sequel. As an open question, we leave the formulation of axioms or proof rules in order to deal with such systems. Thus, we will only use semantic reasoning, where infinite processes are concerned.

2.14 GRAPH MODEL.

Definition of parallel composition on the graph model is not so straightforward. We give the definition of merge with communication only for a restricted subset, for graphs in which each dynamic state has exactly one incoming transition. We can do this because each p-graph is bisimulation equivalent with such a graph (proof omitted). Then the definition goes as follows:

let two p-graphs $g = \langle S_1, D_1, \rightarrow_1, \dashrightarrow_1, r_1 \rangle$, $h = \langle S_2, D_2, \rightarrow_2, \dashrightarrow_2, r_2 \rangle$ be given with disjoint state sets and with exactly one incoming transition for each dynamic node. The set of static nodes of $g \between h$ is $S_1 \times S_2$, the set of dynamic nodes $D_1 \times D_2$, the root is $\langle r_1, r_2 \rangle$, we have $\langle s,t \rangle \dashrightarrow \langle d,e \rangle$ iff $s \dashrightarrow_1 d$ and $t \dashrightarrow_2 e$ and $\langle d,e \rangle \xrightarrow{a} \langle s,t \rangle$ if

- $d \xrightarrow{a}_1 s$ and $t \dashrightarrow_2 e$, or
- $e \xrightarrow{a}_2 t$ and $s \dashrightarrow_1 d$, or
- $d \xrightarrow{b}_1 s$, $e \xrightarrow{c}_2 t$ and $b \mid c = a \neq \delta$.

Next, the definition of encapsulation is straightforward. Let p-graph g be given. $\partial_H(g)$ is obtained from g by erasing all transitions $d \xrightarrow{a} s$ with $a \in H$.

We give a structured operational semantics for the additional operators in table 8 ($a,b,c \in A$). Since after a static step in a parallel composition, the component not chosen has to "take back" the static step, we have to remember the static nodes. This requires an operator with four arguments: $_\,_\between_\,_$, the

inner arguments denoting the dynamic states, the outer ones the static states they came from. The last line gives an operational semantics for recursion: here $X = t_X$ is an equation in a recursive specification E.

$$\frac{x \dashrightarrow \boxed{x'},\ y \dashrightarrow \boxed{y'}}{x \| y \dashrightarrow x \boxed{x' \| y'} y} \qquad \frac{\boxed{x'} \xrightarrow{a} x''}{x \boxed{x' \| y'} y \xrightarrow{a} x'' \| y,\ y \boxed{y' \| x'} x \xrightarrow{a} y \| x''}$$

$$\frac{\boxed{x'} \xrightarrow{a} \surd}{x \boxed{x' \| y'} y \xrightarrow{a} y,\ y \boxed{y' \| x'} x \xrightarrow{a} y} \qquad \frac{\boxed{x'} \xrightarrow{b} x'',\ \boxed{y'} \xrightarrow{c} y'',\ b \mid c = a}{x \boxed{x' \| y'} y \xrightarrow{a} x'' \| y''}$$

$$\frac{\boxed{x'} \xrightarrow{b} \surd,\ \boxed{y'} \xrightarrow{c} \surd,\ b \mid c = a}{x \boxed{x' \| y'} y \xrightarrow{a} \surd} \qquad \frac{\boxed{x'} \xrightarrow{b} x'',\ \boxed{y'} \xrightarrow{c} \surd,\ b \mid c = a}{x \boxed{x' \| y'} y \xrightarrow{a} x'',\ y \boxed{y' \| x'} x \xrightarrow{a} x''}$$

$$\frac{x \dashrightarrow \boxed{x'}}{\partial_H(x) \dashrightarrow \boxed{\partial_H(x')}} \qquad \frac{\boxed{x} \xrightarrow{a} x',\ a \notin H}{\boxed{\partial_H(x)} \xrightarrow{a} \partial_H(x')} \qquad \frac{\boxed{x} \xrightarrow{a} \surd,\ a \notin H}{\boxed{\partial_H(x)} \xrightarrow{a} \surd}$$

$$\frac{t_X \dashrightarrow \boxed{y}}{X \dashrightarrow \boxed{y}} \qquad \frac{\boxed{t_X} \xrightarrow{a} y}{\boxed{X} \xrightarrow{a} y} \qquad \frac{\boxed{t_X} \xrightarrow{a} \surd}{\boxed{X} \xrightarrow{a} \surd}$$

TABLE 8. Operational semantics of additional operators.

2.14 EXAMPLE. We illustrate the definition of parallel composition. We assume there is no communication.

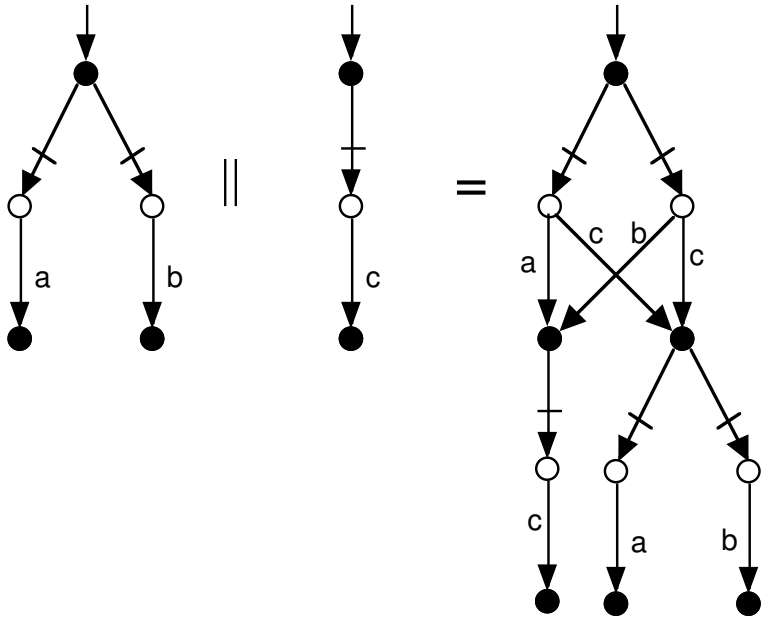

FIGURE 3.

3. IMPLEMENTATION.

If a process P can be written without the $\between$ operator, we call P *total*. If $P \leq Q$, we say that P is a *realisation* of Q. If moreover P is total and P is computable, then we say that P is an *implementation* of Q.

3.1 STANDARD REALISATION.

We define the standard realisation function SR inductively ($a \in A \cup \{\delta\}$):

i. $SR(a) = a$
ii. $t \in HNF+ \Rightarrow SR(a \cdot t) = a \cdot SR(t)$
ii. $SR(a \cdot (t \between s)) = SR(a \cdot t) + SR(a \cdot s)$
iii. $SR(t + s) = SR(t) + SR(s)$
iv. $SR(t \between s) = SR(t) \between SR(s)$.

Note that since $a \cdot (t \between s) = (a + a) \cdot (t \between s) = a \cdot (t \between s) + a \cdot (t \between s) \geq a \cdot t + a \cdot s$, we always have $SR(p) \geq p$. Moreover, if t is in dynamic head normal form, then $SR(t)$ is total.

EXAMPLE: an unreliable channel, that may corrupt messages is given by the recursive equation

$$K = \sum_{d \in D} r_1(d) \cdot i \cdot (s_2(d) \between s_2(\bot)) \cdot K$$

The standard realisation satisfies $SR(K) = \sum_{d \in D} r_1(d) \cdot (i \cdot s_2(d) + i \cdot s_2(\bot)) \cdot SR(K)$.

3.2 DEFINITION.

We call a process P *deadlockfree* iff P has a p-graph without a deadlock node.

3.3 PROPOSITION.

Let $Y \leq X$. If X is deadlockfree, then Y is deadlockfree.

HINT OF PROOF: Suppose not, then Y has a path to a deadlock node s. Since $[\![Y]\!] \leftrightarrows [\![X]\!]$, there is also a dynamic state in X bisimulating with s. By definition of partial bisimulation, this state can have no outgoing transitions. Thus, X has a deadlock node, contradiction.

3.4 NOTE.

A method for verification may now proceed as follows. First, a system is described in the form $P = \partial_H(P_1 \between \ldots \between P_n)$ for certain processes P_i that might contain partial choice operators. Then, a proof is given that $\partial_H(SR(P_1) \between \ldots \between SR(P_n))$ is a solution for X, in a specification that is usually in terms of $\tau_I(X)$, say $\tau_I(X) = E(\tau_I(X))$ for a certain recursive equation E. Using standard realisation, each of the components is given an implementation, so that we get an automaton model. Often, the first step is skipped in this procedure. This introduces philosophical complications about the states of the specification.

If $\partial_H(SR(P_1) \between \ldots \between SR(P_n))$ fails the specification, we know that a very plausible implementation fails, so we are warned about the validity of $\partial_H(P_1 \between \ldots \between P_n)$. Often, that $\partial_H(SR(P_1) \between \ldots \between SR(P_n))$ satisfies the specification is proven under a general overall fairness assumption (which is quite reasonable in case of finite state systems). Then, if implementations $Q_i \leq P_i$ are considered which are sometimes not fair, we must be careful: process $\partial_H(Q_1 \between \ldots \between Q_n)$ may contain an I-livelock

that $\partial_H(SR(P_1) \parallel ... \parallel SR(P_n))$ does not show. An example of this is given by the unreliable channel in 3.1, used in an alternating bit protocol (as in 3.5), taking the implementation

$$Q = \sum_{d\in D} r_1(d)\cdot i\cdot s_2(\bot)\cdot Q$$

of the channel. Even in other cases, the proof of $\tau_I(X) = E(\tau_I(X))$ may use a kind of combined fairness which the Q_i cannot be supposed to have.

3.5 ALTERNATING BIT PROTOCOL.

Let D be a finite set of data. We use the standard read/send communication function given by $r_k(x) \mid s_k(x) = c_k(x)$ for communication port k and message x (and δ otherwise). We specify sender S, unreliable channels K,L and receiver R as in [BAW90].

$S = S0\cdot S1\cdot S$

$Sb = \sum_{d\in D} r_1(d)\cdot Sb_d$ for $b = 0,1$.

$Sb_d = s_2(db)\cdot((r_6(1\text{-}b)+r_6(\bot))\cdot Sb_d + r_6(b))$ for $b = 0,1$, $d \in D$.

$$K = \sum_{d\in D, b\in B} r_2(db)\cdot i\cdot(s_3(db) \leftthreetimes s_3(\bot))\cdot K$$

$$L = \sum_{b\in B} r_5(b)\cdot i\cdot (s_6(b) \leftthreetimes i\cdot s_6(\bot))\cdot L$$

$R = R1\cdot R0\cdot R$

$$Rb = \Big(\sum_{d\in D} r_3(db) + r_3(\bot)\Big)\cdot s5(b)\cdot Rb + \sum_{d\in D} r_3(d(1\text{-}b))\cdot s_4(d)\cdot s_5(1\text{-}b) \quad \text{for } b=0,1$$

$ABP = t_I\circ\partial_H(S \parallel K \parallel L \parallel R)$,

where $H = \{r_k(x), s_k(x) : k \in \{2,3,5,6\}, x \in D \cup (D \times B) \cup B \cup \{\bot\}\}$ is the set of communication actions, $I = \{c_k(x) : k \in \{2,3,5,6\}, x \in D \cup (D \times B) \cup B \cup \{\bot\}\} \cup \{i\}$ is the set of internal actions, and t_I is the *pre-abstraction* operator (from [BAB88]), that renames all internal actions into t.

ABP is a partial process specifying the alternating bit protocol. Constructing the transition system for this process, we can derive the following recursive specification for ABP:

$$X = \sum_{d\in D} r_1(d)\cdot Y_d$$

$Y_d = t\cdot t\cdot(t\cdot t\cdot t\cdot t\cdot t\cdot Y_d \leftthreetimes t\cdot s_4(d)\cdot Z)$ for $d \in D$.

$Z = t\cdot t\cdot(t\cdot t\cdot t\cdot t\cdot t\cdot Z \leftthreetimes t\cdot X)$.

Adding more non-determinism, we can simplify this specification considerably. Consider the following recursive specification, and let process $BUF1$ be a solution for this specification:

$$X' = t\cdot X' \leftthreetimes \sum_{d\in D} r_1(d)\cdot Y'_d$$

$Y'_d = t\cdot Y'_d \leftthreetimes s_4(d)\cdot X'$ for $d \in D$.

By considering the p-graphs of these processes, we can derive $ABP \leq BUF1$ holds in our model. It is an open question, how to derive this inequality on the basis of algebraic calculations.

3.6 FIFO QUEUES.

BUF1 is a general specification of a one-place buffer. It has many different realisations. We think that every one-place buffer described will be a realisation of this process. We can generalise even further, and present (without further comment) a specification (with input port 1 and output port 2) such that every unbounded or bounded buffer or queue will be a realisation of it. The variables are parametrised by sequences of data elements from D.

$$Q_\varepsilon = Q^t_\varepsilon \between Q^r_\varepsilon \between Q^{rt}_\varepsilon \qquad (\varepsilon \text{ empty sequence})$$

$$Q_{\sigma d} = Q^t_{\sigma d} \between Q^r_{\sigma d} \between Q^{rt}_{\sigma d} \between Q^s_{\sigma d} \between Q^{sr}_{\sigma d} \between Q^{st}_{\sigma d} \between Q^{srt}_{\sigma d} \qquad (\sigma \in D^*, d \in D)$$

$$Q^t_\sigma = t \cdot Q_\sigma \qquad (\sigma \in D^*)$$

$$Q^r_\sigma = \sum_{d \in D} r_1(d) \cdot Q_{d\sigma} \qquad (\sigma \in D^*)$$

$$Q^{rt}_\sigma = t \cdot Q_\sigma + \sum_{d \in D} r_1(d) \cdot Q_{d\sigma} \qquad (\sigma \in D^*)$$

$$Q^s_{\sigma d} = s_2(d) \cdot Q_\sigma \qquad (\sigma \in D^*, d \in D)$$

$$Q^{sr}_{\sigma d} = s_2(d) \cdot Q_\sigma + \sum_{e \in D} r_1(e) \cdot Q_{e\sigma d} \qquad (\sigma \in D^*, d \in D)$$

$$Q^{st}_{\sigma d} = s_2(d) \cdot Q_\sigma + t \cdot Q_{\sigma d} \qquad (\sigma \in D^*, d \in D)$$

$$Q^{srt}_{\sigma d} = s_2(d) \cdot Q_\sigma + t \cdot Q_{\sigma d} + \sum_{e \in D} r_1(e) \cdot Q_{e\sigma d} \qquad (\sigma \in D^*, d \in D).$$

4. PROCESS ALGEBRA WITH COLLECTING CHOICE.

In a setting where we also have collecting sum $\sqcup$, some things change. The main difference is, that all axioms that involve a duplication of a variable turn into inequalities.

4.1 BASIC PROCESS ALGEBRA.

We start with BPA, so we have a (finite) set of constants $A = \{a,b,c,...\}$, a special constant $\delta \notin A$, and binary operators $+$, $\sqcup$, $\between$, $\cdot$. Axioms A4, PAC3,4,5 are not valid anymore. The axioms in table 9 replace the ones of $BPA_{\between}$ in tables 1 and 2. Notice that the inequality version of PAC3, $X \leq X \between X$, is a direct consequence of PAC6.

$X + Y = Y + X$	A1
$(X+ Y) + Z = X + (Y + Z)$	A2
$a + a = a$	AA3
$(X + Y)\cdot Z \leq X\cdot Z + Y\cdot Z$	A4≤
$(X\cdot Y)\cdot Z = X\cdot(Y\cdot Z)$	A5
$X + \delta = X$	A6
$\delta\cdot X = \delta$	A7
$X \between Y = Y \between X$	PAC1
$(X\between Y) \between Z = X \between (Y \between Z)$	PAC2
$(X \between Y)\cdot Z \leq X\cdot Z \between Y\cdot Z$	PAC4≤
$(X\between Y) + Z \leq (X + Z) \between (Y + Z)$	PAC5≤
$X \leq X \between Y$	PAC6

TABLE 9. Axioms replacing $BPA_{\between}$ in the presence of collecting sum.

Next, we show the axioms for the collecting alternative composition in table 10. This operator distributes both ways over sequential composition. This is an important difference with dynamic and static choice. Replacing all + operators by ⊔ operators gives us the traces of a process. Thus, we do not view trace theory as a homomorphic image of a bisimulation based process algebra, but rather obtain a set of traces by using static choice only. Note that dynamic and partial choice distribute over ⊔ but not the other way around, thus the static choice has precedence (comes before) other forms of choice. In TCSP there is distribution both ways. We think that this two-way distribution is incompatible with the bisimulation model, and forces the use of failure semantics. Therefore, we do not include the other distributive law here.

As for partial choice, $X \sqcup \delta = X$, does not hold. This is because the choice has already been made, and so deadlock cannot be avoided if the wrong choice has been made. CAC8 expresses the idea that realisation reduces collecting non-determinism.

The axioms in tables 9 and 10 together give the theory $BPA_{\sqcup}$.

$X \sqcup Y = Y \sqcup X$	CAC1
$(X\sqcup Y) \sqcup Z = X \sqcup (Y \sqcup Z)$	CAC2
$X = X \sqcup X$	CAC3
$(X \sqcup Y)\cdot Z = X\cdot Z \sqcup Y\cdot Z$	CAC4
$X\cdot(Y \sqcup Z) = X\cdot Y \sqcup X\cdot Z$	CAC5
$(X\sqcup Y) + Z = (X + Z) \sqcup (Y + Z)$	CAC6
$(X\sqcup Y) \between Z = (X \between Z) \sqcup (Y \between Z)$	CAC7
$X \leq X \sqcup Y$	CAC8

TABLE 10. Collecting alternative composition.

If a process P can be written without the $\sqcup$ operator, we call P *connected*. Thus, in section 2 we treated the theory of the connected processes. Consistent with the terminology in section 3, if P can be written without $\between$ and $\sqcup$, we call P *total*.

4.2 SEMANTICS.
In a setting with collecting sum, our semantic domain now consists of nonempty sets of p-graphs. Let $\mathbb{PPG}$ be the set of nonempty sets of p-graphs from $\mathbb{PG}$. The interpretation of the constants now becomes the singleton set consisting of the earlier interpretation.

For collecting sum, we simply put $[\![P \sqcup Q]\!] = [\![P]\!] \cup [\![Q]\!]$. For all other binary operators $\Box$ (so $\Box \in \{+, \cdot, \between\}$) we put $[\![P \Box Q]\!] = \{g \Box h : g \in [\![P]\!], h \in [\![Q]\!]\}$.

We define partial bisimulation on $\mathbb{PPG}$ as follows: $G \underline{\leftrightarrow} H$ iff $\forall g \in G\ \exists h \in H\ g \underline{\leftrightarrow} h$.

We can still obtain soundness as before. For basic terms, we need an extra layer for collecting sum at the top.

4.3 DEFINITION. We define an extended set of basic terms $\mathcal{B}_\sqcup$ by adding the following clauses to definition 2.9.
i. $\mathcal{B} \subseteq \mathcal{B}_\sqcup$
ii. $t, s \in \mathcal{B}_\sqcup \Rightarrow t \sqcup s \in \mathcal{B}_\sqcup$.
An example of an extended basic term: $((a\cdot(b \between (c + d)) + b) \between \delta) \sqcup (a\cdot b \between a\cdot(b \between c))$.

4.4 THEOREM: Let s be a closed term. Then there is a basic term t such that $BPA_\sqcup \vdash s = t$.
SKETCH OF PROOF: By term rewrite analysis. We do this in two stages. First, we rewrite using the rules in table 11 below. This gives us a $\sqcup$-normal form, i.e. a term where all $\sqcup$-operators appear at the top level only (i.e. there is no operator other than $\sqcup$ above $\sqcup$ in the parse tree). Then, on the subterms below the $\sqcup$ operators, we use the rewrite rules of table 5, in 2.10.

$(X \sqcup Y)\cdot Z \to X\cdot Z \sqcup Y\cdot Z$	RCAC4
$X\cdot(Y \sqcup Z) \to X\cdot Y \sqcup X\cdot Z$	RCAC5
$(X\sqcup Y) + Z \to (X + Z) \sqcup (Y + Z)$	RCAC6
$X + (Y\sqcup Z) \to (X + Y) \sqcup (X + Z)$	RCAC6'
$(X\sqcup Y) \between Z \to (X \between Z) \sqcup (Y \between Z)$	RPAC6
$X \between (Y\sqcup Z) \to (X \between Y) \sqcup (X \between Z)$	RPAC6'

TABLE 11. Rewrite system.

4.5 LEMMA: For all closed $BPA_\sqcup$-terms t we have $BPA_\sqcup \vdash t \leq t + t$.
PROOF: By 4.4, we may suppose $t \in \mathcal{B}_\sqcup$. We use induction on the structure of basic terms. All cases except the following are handled in lemma 2.5. Suppose the lemma holds for x,y, then:
$x \sqcup y \leq x \sqcup y \sqcup (x + y) \sqcup (y + x) \leq (x + x) \sqcup (y + x) \sqcup (x + y) \sqcup (y + y) =$
$= ((x \sqcup y) + x) \sqcup ((x \sqcup y) + y) = (x + (x \sqcup y)) \sqcup (y + (x \sqcup y)) = (x \sqcup y) + (x \sqcup y)$.

4.6 THEOREM (COMPLETENESS). Let s,t be closed terms. Then $[\![s]\!] \leftrightarrows [\![t]\!] \Rightarrow BPA_{\sqcup} \vdash s \leq t$.
PROOF: Omitted.

4.7 ADDITIONAL OPERATORS.
We show the axioms for $ACP_{\sqcup}$. The ones for $PA_{\sqcup}$ can be obtained easily from these. We see that the full expansion theorem is no longer valid. The way to expand the merge is to first get all collecting sums to the top level, factor them out, and then use the full expansion theorem on the connected subterms.

$a \mid b = b \mid a$	
$(a \mid b) \mid c = a \mid (b \mid c)$	
$a \mid \delta = \delta$	
$X \parallel Y \leq X \mathbin{\mathbb{L}} Y + Y \mathbin{\mathbb{L}} X + X \mid Y$	
$a \mathbin{\mathbb{L}} X = a \cdot X$	
$a \cdot X \mathbin{\mathbb{L}} Y = a \cdot (X \parallel Y)$	
$(X \square Y) \mathbin{\mathbb{L}} Z \leq X \mathbin{\mathbb{L}} Z \square Y \mathbin{\mathbb{L}} Z$	for $\square \in \{+, \dot{+}\}$
$(X \sqcup Y) \mathbin{\mathbb{L}} Z = X \mathbin{\mathbb{L}} Z \sqcup Y \mathbin{\mathbb{L}} Z$	
$X \mathbin{\mathbb{L}} (Y \sqcup Z) = X \mathbin{\mathbb{L}} Y \sqcup X \mathbin{\mathbb{L}} Z$	
$(X \sqcup Y) \parallel Z = X \parallel Z \sqcup Y \parallel Z$	
$X \parallel (Y \sqcup Z) = X \parallel Y \sqcup X \parallel Z$	
$a \cdot X \mid b = (a \mid b) \cdot X$	
$a \mid b \cdot X = (a \mid b) \cdot X$	
$a \cdot X \mid b \cdot Y = (a \mid b) \cdot (X \parallel Y)$	
$(X \square Y) \mid Z \leq X \mid Z \square Y \mid Z$	for $\square \in \{+, \dot{+}\}$
$X \mid (Y \square Z) \leq X \mid Y \square X \mid Z$	for $\square \in \{+, \dot{+}\}$
$(X \sqcup Y) \mid Z = X \mid Z \sqcup Y \mid Z$	
$X \mid (Y \sqcup Z) = X \mid Y \sqcup X \mid Z$	
$\partial_H(a) = a$	if $a \notin H$
$\partial_H(a) = \delta$	if $a \in H$
$\partial_H(X \square Y) = \partial_H(X) \square \partial_H(Y)$	for $\square \in \{+, \cdot, \sqcup, \dot{+}\}$

TABLE 12. Axioms for $ACP_{\sqcup}$.

5. CONCLUSION.

We have extended ACP, in a setting of bisimulation semantics, with two additional choice operators: partial sum and collecting sum. This serves to bridge the gap between ACP and TCSP: we can combine TCSP's distinction between choice and non-determinism with the operational structure of bisimulations.

Further we found that the introduction of partial choice and collecting choice allows us, in some cases, to express safety properties of systems without the use of silent steps and to verify safety

properties in a setting in which no assumption about fairness (or unfairness) has been made. These verifications take place in the model; an algebraic style of verification has not been proposed here. Avoiding silent steps as well as fairness considerations are virtues of CSP that we consider advantages indeed. We think we have extended ACP in such a way that these virtues have become available in the ACP setting as well.

We found that it is the partial sum operator, that extends nicely to a probabilistic sum operator. Thus, the theory presented here forms a possible basis for a theory involving probabilistic choice together with other forms of choice. This allows one to describe systems where we know the probability of some choices, but not of all choices.

REFERENCES.

[BAB88] J.C.M. BAETEN & J.A. BERGSTRA, *Global renaming operators in concrete process algebra,* I&C 78, 1988, pp. 205-245.

[BAB94] J.C.M. BAETEN & J.A. BERGSTRA, *On sequential composition, action prefixes and process prefix,* to appear in FAC 6 (3), 1994.

[BABK87] J.C.M. BAETEN, J.A. BERGSTRA & J.W. KLOP, *Ready trace semantics for concrete process algebra with priority operator,* The Computer Journal 30 (6), 1987, pp. 498-506.

[BABS92] J.C.M. BAETEN, J.A. BERGSTRA & S.A. SMOLKA, *Axiomatizing probabilistic processes: ACP with generative probabilities,* in Proc. CONCUR'92, Stony Brook (W.R. Cleaveland, ed.), LNCS 630, Springer Verlag 1992, pp. 472-485.

[BAW90] J.C.M. BAETEN & W.P. WEIJLAND, *Process algebra,* Cambridge Tracts in Theoretical Computer Science 18, Cambridge University Press 1990.

[BAZ82] J.W. DE BAKKER & J.I. ZUCKER, *Processes and the denotational semantics of concurrency,* I&C 54, pp. 70-120.

[BEK84] J.A. BERGSTRA & J.W. KLOP, *Process algebra for synchronous communication,* Inf. & Control 60, pp. 109-137.

[BEKO88] J.A. BERGSTRA, J.W. KLOP & E.-R. OLDEROG, *Readies and failures in the algebra of communicating processes,* SIAM J. of Comp. 17 (6), 1988, pp. 1134-1177.

[BRHR84] S.D. BROOKES, C.A.R. HOARE & W. ROSCOE, *A theory of communicating sequential processes,* JACM 31, 1984, pp. 560-599.

[HAN91] H. HANSSON, *Time and probability in formal design of distributed systems,* Ph.D. thesis, DoCS 91/27, Univ. of Uppsala 1991.

[HAJ90] H. HANSSON & B. JONSSON, *A calculus for communicating systems with time and probabilities,* in Proc. RTSS90, Orlando, IEEE Computer Society Press 1990.

[HEN88] M. HENNESSY, *Algebraic theory of processes,* MIT Press 1988.

[HOA85] C.A.R. HOARE, *Communicating sequential processes,* Prentice Hall 1985.

[JON93] B. JONSSON, *Probabilistic processes,* notes of a tutorial at CONCUR'93, Hildesheim 1993.

[MIL80] R. MILNER, *A calculus of communicating systems,* LNCS92, Springer 1980.