Formal Specification of Distributed Systems: Communicating Sequential Processes (CSP)

Nicola Dragoni
Embedded Systems Engineering
DTU Informatics

3 Nondeterminism
Nondeterminism

• Consider the following machine which offers a choice of two combinations of change for 5p (the choice is exercised by the customer):

\[
\text{CH5C} = \text{in}5\text{p} \rightarrow (\text{out}1\text{p} \rightarrow \text{out}1\text{p} \rightarrow \text{out}1\text{p} \rightarrow \text{out}2\text{p} \rightarrow \text{CH5C} \\
| \text{out}2\text{p} \rightarrow \text{out}1\text{p} \rightarrow \text{out}2\text{p} \rightarrow \text{CH5C})
\]

• Such process is called deterministic, because whenever there is more than one event possible, the choice between them is determined externally by the environment of the process.
Nondeterminism

• Consider the following machine which offers a choice of two combinations of change for 5p (the choice is exercised by the customer):

\[
\text{CH5C} = \text{in}5p \rightarrow (\text{out1p} \rightarrow \text{out1p} \rightarrow \text{out1p} \rightarrow \text{out2p} \rightarrow \text{CH5C} \\
| \text{out2p} \rightarrow \text{out1p} \rightarrow \text{out2p} \rightarrow \text{CH5C})
\]

• Such process is called deterministic, because whenever there is more than one event possible, the choice between them is determined externally by the environment of the process.

• Sometimes a process has a range of possible behaviours, but the environment of the process does not have any ability to influence or even observe the selection between the alternatives.
Nondeterminism

• Consider the following machine which offers a choice of two combinations of change for 5p (the choice is exercised by the customer):

\[
\text{CH5C} = \text{in5p} \rightarrow (\text{out1p} \rightarrow \text{out1p} \rightarrow \text{out1p} \rightarrow \text{out2p} \rightarrow \text{CH5C})
\]

| \text{out2p} \rightarrow \text{out1p} \rightarrow \text{out2p} \rightarrow \text{CH5C} |

• Such process is called deterministic, because whenever there is more than one event possible, the choice between them is determined externally by the environment of the process.

• Sometimes a process has a range of possible behaviours, but the environment of the process does not have any ability to influence or even observe the selection between the alternatives.

• There is nothing mysterious about this kind of nondeterminism: it arises from a deliberate decision to ignore the factor which influence the selection.
Nondeterministic or (⨅)

• If P and Q are processes, then we introduce the notation

\[ P \sqcap Q \quad (P \text{ or } Q) \]

to denote a process which behaves either like P or like Q, where the selection between them is made arbitrarily, *without the knowledge of control of the external environment*.

• The alphabets of the operands are assumed to be the same:

\[ \alpha(P \sqcap Q) = \alpha P = \alpha Q \]
Example: Nondeterministic Change-Giving Machine

- A change-giving machine which always gives the right change in one of two combinations:

\[
\text{CH5D} = (\text{in5p} \rightarrow ( (\text{out1p} \rightarrow \text{out1p} \rightarrow \text{out1p} \rightarrow \text{out2p} \rightarrow \text{CH5D}) \\
\quad \land (\text{out2p} \rightarrow \text{out1p} \rightarrow \text{out2p} \rightarrow \text{CH5D})))
\]
Why Do We Need Nondeterminism?

• $\land$ is **not** intended as a useful operator for *implementing* a process...

  ‣ It would be very foolish to build both P and Q, put them in a black bag, make an arbitrary choice between them, and then throw the other one away!

• The **main advantage of nondeterminism** is in *specifying* a process.

  ‣ A process specified as $(P \land Q)$ can be implemented either by building P or by building Q.

  ‣ The choice can be made in advance by the implementor on grounds not relevant (and deliberately ignored) in the specification, such as low cost, fast response times, or early delivery.
[Nondeterminism] Some Laws

L1  \( P \cap P = P \)  (idempotence)

L2  \( P \cap Q = Q \cap P \)  (symmetry)

L3  \( P \cap (Q \cap R) = (P \cap Q) \cap P \)  (associativity)

L4  \( x \rightarrow (P \cap Q) = (x \rightarrow P) \cap (x \rightarrow Q) \)  (distribution of \( \rightarrow \))

L5  \( P \parallel (Q \cap R) = (P \parallel Q) \cap (P \parallel R) \)  (distribution of \( \parallel \))

L6  \( (P \cap Q) \parallel R = (P \parallel Q) \cap (P \parallel R) \)  (distribution of \( \parallel \))
General Choice

• The environment of \((P \sqcap Q)\) has no control or even knowledge of the choice that is made between \(P\) and \(Q\), or even the time at which the choice is made.

• So \((P \sqcap Q)\) is not a helpful way of combining processes, because the environment must be prepared to deal with either \(P\) or \(Q\).

• We therefore introduce another operation

\[
(P \Box Q) \quad (P\text{ choice } Q)
\]

• for which the environment can control which of \(P\) and \(Q\) will be selected, provided that this control is exercised on the very first action.
P □ Q

α(P □ Q) = αP = αQ

• If the first action is not a possible first action of P, then Q will be selected.
• But if Q cannot engage initially in the action, P will be selected.
• If, however, the first action is possible for both P and Q, then the choice between them is nondeterministic.
• If the event is impossible for both P and Q, then it just cannot happen.

L5  (x : A → P(x)) □ (y : B → Q(y)) =
    (z : (A ∪ B) →
     (if z ∈ (A - B) then
      P(z)
     else if z ∈ (B - A) then
      Q(z)
     else if z ∈ (A ∩ B) then
      (P(z) ∩ Q(z)))))
Let $P = c \rightarrow P$

Let $Q = d \rightarrow Q$

If $c = d$ then

If $c \neq d$ then

Convention: $\rightarrow$ binds more than $\Box$. 
Let $P = c \rightarrow P$

Let $Q = d \rightarrow Q$

If $c = d$ then

$(c \rightarrow P \square d \rightarrow Q) = (c \rightarrow P \cap d \rightarrow Q)$

If $c \neq d$ then

Convention: $\rightarrow$ binds more than $\square$. 
\[ P \sqcap Q \]

- Let \( P = c \rightarrow P \)
- Let \( Q = d \rightarrow Q \)

- If \( c = d \) then
  \[
  (c \rightarrow P \sqcap d \rightarrow Q) = (c \rightarrow P \sqcap d \rightarrow Q)
  \]

- If \( c \neq d \) then
  \[
  (c \rightarrow P \sqcap d \rightarrow Q) = (c \rightarrow P | d \rightarrow Q)
  \]

- Convention: \( \rightarrow \) binds more than \( \sqcap \).
Question: $P \square Q$ vs $P \mid Q$?
Question: $P \square Q$ vs $P \mid Q$?

• The question makes no sense!!

• From Chapter 1:

  • Note that the choice symbol $\mid$ is **not** an operator on processes.

  • It would be **syntactically incorrect** to write $P \mid Q$, for processes $P$ and $Q$.

  • $\square$ is an operator on processes, while $\mid$ is an operator on prefixes (i.e., $x \rightarrow P$).

  • This is just “syntactic sugar”, but you must remember it to avoid ambiguity in the specification.
Laws for □

• The algebraic laws for □ are similar to those for △, and for the same reasons.

L1-L3 □ is idempotent, symmetric and associative

L4 P □ STOP = P

L6 P □ (Q △ R) = (P □ Q) △ (P □ R) □ distributes through △

L7 P △ (Q □ R) = (P △ Q) □ (P △ R) △ distributes through □
Question: $P \Box Q$ vs $P \sqcap Q$?

- They cannot be distinguished by their traces, because each trace of one of them is also a possible trace of the other.

- However, it is possible to put them in an environment in which $(P \sqcap Q)$ can deadlock at its first step, but $(P \Box Q)$ cannot.

Let $x \neq y$ and $P = (x \rightarrow P)$, $Q = (y \rightarrow Q)$, $\alpha P = \alpha Q = \{x,y\}$

Then

$$(P \Box Q) || P = (x \rightarrow P) = P$$

but

$$(P \sqcap Q) || P = (P || P) \sqcap (Q || P) = P \sqcap \text{STOP}$$

In environment $P$, process $(P \sqcap Q)$ may reach deadlock but process $(P \Box Q)$ cannot.
Hiding (\)

• In describing the behaviour of a process, we often need to consider events representing internal transitions of that process.

• Such events may denote the interactions and communications between concurrently acting components from which the process has been constructed.

• After construction, we might conceal the structure of its components; and we also wish to conceal all occurrences of actions internal to the process.

• If C is a finite set of events to be concealed, then \( P \setminus C \) is a process which behaves like P, except that each occurrence of any event in C is concealed.

• \( \alpha(P \setminus C) = \alpha(P) - C \)
Example

\[
\text{NOISYVM} = (\text{coin} \rightarrow \text{clink} \rightarrow \text{choc} \rightarrow \text{clunk} \rightarrow \text{NOISYVM})
\]

• The noisy vending machine can be placed in a soundproof box, so that both\text{clink} and\text{clunk} cannot be “seen” from the outside.

\[
\text{NOISYVM} \setminus \{\text{clink, clunk}\}
\]

• We can re-formulate NOISYVM as:

\[
\text{NOISYVM} = (\text{coin} \rightarrow \tau \rightarrow \text{choc} \rightarrow \tau \rightarrow \text{NOISYVM})
\]

where\tau indicates an \text{internal event} which cannot be seen from the outside.

• Note that the resulting process is equal to the simple vending machine:

\[
\text{NOISYVM} \setminus \{\text{clink, clunk}\} = \text{VMS} = (\text{coin} \rightarrow (\text{choc} \rightarrow \text{VMS}))
\]
Hiding

• When two processes have been combined to run concurrently, their mutual interactions are usually regarded as internal workings of the resulting systems.

• They are intended to occur autonomously and as quickly as possible, without the knowledge or intervention of the system’s outer environment.

• Thus it is the symbols in the intersection of the alphabets of the two components that need to be concealed.
Example

Let $\alpha P = \{a, c\}$ and $P = (a \rightarrow c \rightarrow P)$

$\alpha Q = \{b, c\}$ and $Q = (c \rightarrow b \rightarrow Q)$

$(P \parallel Q) = (a \rightarrow c \rightarrow Z)$

$Z = (a \rightarrow b \rightarrow c \rightarrow Z$

$\mid b \rightarrow a \rightarrow c \rightarrow Z)$

$(P \parallel Q) \setminus \{c\}$?

HOMEWORK
Example

Let $\alpha P = \{a, c\}$ and $P = (a \rightarrow c \rightarrow P)$
$\alpha Q = \{b, c\}$ and $Q = (c \rightarrow b \rightarrow Q)$

$(P || Q) = (a \rightarrow c \rightarrow Z)$
$Z = (a \rightarrow b \rightarrow c \rightarrow Z \mid b \rightarrow a \rightarrow c \rightarrow Z)$

$(P \ || \ Q) \ \setminus \ \{c\} \ ?$

 HOMEWORK

- The action $c$ in the alphabet of both $P$ and $Q$ is now regarded as an internal action, to be concealed:

$(P \ || \ Q) \ \setminus \ \{c\} = (a \rightarrow Z)$
$Z = (a \rightarrow b \rightarrow Z \mid b \rightarrow a \rightarrow Z)$
[Hiding] Some Laws

L1  \( P \setminus \{\} = P \)

L2  \((P \setminus B) \setminus C = P \setminus (B \cup C)\)

L3  \((P \cap Q) \setminus C = (P \setminus C) \cap (Q \setminus C)\)  \(\text{(distribution of } \setminus)\)

L4  \((x \rightarrow P) \setminus C = x \rightarrow (P \setminus C)\)  \(\text{if } x \not\in C\)
     \(= P \setminus C\)  \(\text{if } x \in C\)

L5  if \(\alpha P \cap \alpha Q \cap C = \{\}\) then
     \((P \parallel Q) \setminus C = (P \setminus C) \parallel (Q \setminus C)\)  \(\text{(distribution of } \setminus)\)

If \(C\) contains only events in which \(P\) and \(Q\) participate independently, concealment of \(C\) distributes through their concurrent composition.
Pictures

- Nondeterministic choice can be represented in a picture by a node from which emerge two or more *unlabelled* arrows.

- On reaching the node, a process passes imperceptibly along one of the emergent arrows, the choice being nondeterministic.

- Example: $P \sqcap Q$
Another Example

\[ (((c \rightarrow P) \cap (d \rightarrow Q)) \setminus \{c, d\} \]

by L3
Interleaving

• The \( || \) operator was defined in such a way that actions in the alphabet of both operands require *simultaneous participation* of them both.

• Using \( || \), it is possible to combine interacting processes with differing alphabets into systems exhibiting concurrent activity, but without introducing nondeterminism.

• However, it is sometimes useful to join processes with the same alphabet to operate concurrently without directly interacting or synchronising with each other.
### Operator

| P ||| Q | P interleave Q |
|---|---|
| \[ \alpha(P ||| Q) = \alpha P = \alpha Q \] |

- Each action of the system is an action of **exactly one** of the processes.

- If one of the processes cannot engage in the action, then it must have been the other one.

- If both processes could have engaged in the same action, the choice between them is **nondeterministic**.
Example

\[
\text{VMS} = (\text{coin} \rightarrow (\text{choc} \rightarrow \text{VMS}))
\]

\[
(\text{VMS} ||| \text{VMS}) ?
\]
Example

\[ \text{VMS} = (\text{coin} \rightarrow (\text{choc} \rightarrow \text{VMS})) \]

\[ (\text{VMS} ||| \text{VMS}) \]

*A vending machine that will accept up to two coins before dispensing up to two chocolates.*

\[ (\text{VMS} ||| \text{VMS}) = \text{VMS2} \text{ where:} \]

\[ \text{VMS2} = (\text{coin} \rightarrow \text{VMCRED}) \]

\[ \text{VMCRED} = (\text{coin} \rightarrow \text{choc} \rightarrow \text{VMCRED} \mid \text{choc} \rightarrow \text{coin} \rightarrow \text{VMCRED}) \]

*HOMEWORK*