Concurrent Object-Oriented Programming

Prof. Dr. Bertrand Meyer

Lecture 11: An introduction to CSP
Origin

Communicating Sequential Processes: C.A.R. Hoare


Revised with help of S. D. Brooks and A.W. Roscoe

1985 book, revised 2004
CSP purpose

Concurrency formalism
- Expresses many concurrent situations elegantly
- Influenced design of several concurrent programming languages, in particular Occam (Transputer)

Calculus
- Formally specified: laws
- Makes it possible to prove properties of systems
Basic notions

Processes engage in events

Example:

BDVM = (coin $\rightarrow$ coffee $\rightarrow$ coin $\rightarrow$ coffee $\rightarrow$ STOP)

$\alpha$(BDVM) = \{coin, coffee\}
Basic CSP syntax

\[ P ::= \]
\[ \text{Stop} \mid \text{-- Does not engage in any events} \]
\[ a \rightarrow P \mid \text{-- Accepts } a \text{, then engages in } P \]
\[ P \sqcap P \mid \text{-- Internal choice} \]
\[ P \sqcup P \mid \text{-- External choice} \]
\[ P || P \mid \text{-- Concurrency} \]
\[ P ||| P \mid \text{-- Interleaving} \]
\[ P \setminus H \mid \text{-- Hiding (} H \text{: alphabet symbols)} \]
\[ \mu P \cdot f (P) \mid \text{-- Recursion} \]
Some examples

\[ \text{CLOCK} = (\text{tick} \rightarrow \text{CLOCK}) \]

This is an abbreviation for

\[ \text{CLOCK} = \mu P \bullet (\text{tick} \rightarrow P) \]

\[ \text{CVM} = (\text{in1f} \rightarrow (\text{coffee} \rightarrow \text{CVM})) \]
\[ = (\text{in1f} \rightarrow \text{coffee} \rightarrow \text{CVM}) \quad \text{-- Right-associativity} \]

\[ \text{CHM1} = (\text{in1f} \rightarrow \text{out50rp} \rightarrow \text{out20rp} \rightarrow \text{out20rp} \rightarrow \text{out10rp}) \]
\[ \text{CHM2} = (\text{in1f} \rightarrow \text{out50rp} \rightarrow \text{out50rp}) \]

\[ \text{CHM} = \text{CHM1} \Box \text{CHM2} \]
More examples

COPYBIT = (in.0 $\rightarrow$ out.0 $\rightarrow$ COPYBIT □

$\quad$ in.1 $\rightarrow$ out.1 $\rightarrow$ COPYBIT)
More examples

\[ VMC = \]
\[ (in2f \rightarrow \]
\[ (((large \rightarrow VMC) \square ) \]
\[ (small \rightarrow out1f \rightarrow VMC)) ) \]
\[ \square \]
\[ (in1f \rightarrow \]
\[ (((small \rightarrow VMC) \square ) \]
\[ (in1f \rightarrow large \rightarrow VMC)) ) \]

\[ FOOLCUST = (in2f \rightarrow large \rightarrow FOOLCUST \square ) \]
\[ in1f \rightarrow large \rightarrow FOOLCUST) \]

\[ FOOLCUST || VMC = \]
\[ \mu P \bullet (in2f \rightarrow large \rightarrow P \square in2f \rightarrow STOP) \]
Internal non-deterministic choice

\[ CH1F = (\text{in1f} \rightarrow \]
\[ ((\text{out20rp} \rightarrow \text{out20rp} \rightarrow \]
\[ \text{out20rp} \rightarrow \text{out20rp} \rightarrow \text{out20rp} \rightarrow CH1F) \]
\[ \Pi \]
\[ (\text{out50rp} \rightarrow \text{out50rp} \rightarrow CH1F)) \]
Laws of concurrency

\[ P \parallel Q = Q \parallel P \]
\[ P \parallel (Q \parallel R)) = ((P \parallel Q) \parallel R) \]
\[ P \parallel \text{STOP}_\alpha P = \text{STOP}_\alpha P \]
\[ (c \rightarrow P) \parallel (c \rightarrow Q) = (c \rightarrow (P \parallel Q)) \]
\[ (c \rightarrow P) \parallel (d \rightarrow Q) = \text{STOP} \quad -- \text{If } c \neq d \]
\[ (x: A \rightarrow P(x)) \parallel (y: B \rightarrow Q(y)) = (z: (A \cap B) \rightarrow (P(z) \parallel Q(z))) \]
Laws of non-deterministic internal choice

\[ P \sqcap Q = Q \sqcap P \]
\[ P \sqcap (Q \sqcap R) = (P \sqcap Q) \sqcap R \]
\[ x \rightarrow (P \sqcap Q) = (x \rightarrow P) \sqcap (x \rightarrow Q) \]

\[ P \parallel (Q \sqcap R) = (P \parallel Q) \sqcap (P \parallel R) \]
\[ (P \parallel Q) \sqcap R = (P \parallel R) \sqcap (Q \parallel R) \]

The recursion operator is not distributive; consider:

\[ P = \mu X \cdot ((a \rightarrow X) \sqcap (b \rightarrow X)) \]
\[ Q = (\mu X \cdot (a \rightarrow X)) \sqcap (\mu X \cdot (b \rightarrow X)) \]