

Fundamental operators

- Prefixing $a \rightarrow P$
 $in5 \rightarrow out2 \rightarrow out2 \rightarrow out1 \rightarrow \dots$
- Recursion $P = F(P)$ oder $\mu p.F(p)$
 $P = in5 \rightarrow out2 \rightarrow out2 \rightarrow out1 \rightarrow P$
 $\mu p.in5 \rightarrow out2 \rightarrow out2 \rightarrow out1 \rightarrow P$
- Guarded alternative $P \mid Q$
 $a_1 \rightarrow P_1 \mid a_2 \rightarrow P_2$
 $?x : A \rightarrow P(x)$

• Choice operators

- External choice $P \square Q$

$$a \rightarrow ((a \rightarrow P) \square (b \rightarrow Q))$$

$$(a \rightarrow a \rightarrow P) \square (a \rightarrow b \rightarrow Q)$$

- Internal choice $P \sqcap Q$

$$a \rightarrow ((a \rightarrow P) \sqcap (b \rightarrow Q))$$

- Conditional choice $P \leftarrow b \rightarrow Q$ (if b then P else Q)

$$P = \text{left?}x \rightarrow \text{right!}((-x) \leftarrow x < 0 \rightarrow x) \rightarrow P$$

$$P = \text{left?}x \rightarrow ((\text{right!}(-x) \rightarrow P) \leftarrow x < 0 \rightarrow (\text{right!}x) \rightarrow P))$$

Definition 1 (Determinism and Nondeterminism)

A *deterministic* process is one where the range of events offered to the environment depends only on things it has seen, i.e., the sequence of communications so far.

A process is *nondeterministic* when some internal decision can lead to uncertainty about what will happen.

Parallel operators

- Synchronous parallel $P \parallel Q$

$$(a \rightarrow \text{REPEAT}) \parallel \text{REPEAT}$$

$$\text{REPEAT} = ?x : \Sigma \rightarrow x \rightarrow \text{REPEAT} \quad (a \in \Sigma)$$

- Alphabetized parallel $P \parallel_B Q$

$$\text{COPY}(in, out) = in?x : T \rightarrow out.x \rightarrow \text{COPY}(in, out)$$

$$\text{COPYS} =$$

$$\text{COPY}(c_0, c_1) \parallel_{\{c_0, c_1\}} \{c_1, \dots, c_n\} (\text{COPY}(c_1, c_2) \parallel_{\{c_1, c_2\}} \{c_2, \dots, c_n\})$$

$$(\dots (\text{COPY}(c_{n-2}, c_{n-1}) \parallel_{\{c_{n-2}, c_{n-1}\}} \{c_{n-1}, c_n\}) \\ \text{COPY}(c_{n-1}, c_n) \dots))$$

- Interleaving $P \parallel Q$

$\text{FORK}(i) = (\text{picksup}.i.i \rightarrow \text{putsdwn}.i.i \rightarrow \text{FORK}(i))$

□ $(\text{picksup}.i\ominus 1.i \rightarrow \text{putsdwn}.i\ominus 1.i \rightarrow \text{FORK}(i))$

$\text{PHIL}(i) = \text{thinks}.i \rightarrow \text{sits}.i \rightarrow \text{picksup}.i.i \rightarrow \text{picksup}.i.i\oplus 1$
 $\rightarrow \text{eats}.i \rightarrow \text{putsdwn}.i.i\oplus 1 \rightarrow \text{putsdwn}.i.i$
 $\rightarrow \text{getsup}.i \rightarrow \text{PHIL}(i)$

$\text{FORK} = \text{FORK}(0) \parallel \text{FORK}(1) \parallel \dots \parallel \text{FORK}(4)$

$\text{PHIL} = \text{PHIL}(0) \parallel \text{PHIL}(1) \parallel \dots \parallel \text{PHIL}(4)$

- Generalized parallel $P \underset{A}{\parallel} Q$

PHIL $\underset{\{picksup, putsdwn\}}{\parallel}$ FORK

$$P \parallel\parallel Q = P \underset{\{\}}{\parallel} Q$$

$$P \underset{A}{\parallel} \underset{B}{\parallel} Q = P \underset{A \cap B}{\parallel} Q$$

Hiding and renaming

- Hiding $P \setminus A$

COPYS $\setminus \{c_1, \dots, c_{n-1}\}$

$$(a \rightarrow P \sqcap b \rightarrow Q) \setminus \{b\} =$$

$$(a \rightarrow (P \setminus \{b\}) \sqcap Stop) \sqcap (Q \setminus \{b\})$$

$$P \triangleright Q = (P \sqcap Stop) \sqcap Q \quad (\textit{time-out})$$

- Renaming and alphabet transformations $P \parallel f \parallel$

- Injective functions $f : \Sigma \rightarrow \Sigma$

$$g(\textit{in}.x) = \textit{left}.x \quad g(\textit{out}.x) = \textit{right}.x$$

$$\text{COPY}(\textit{in}, \textit{out}) \parallel g \parallel = \text{COPY}(\textit{left}, \textit{right})$$

- Non-injective functions

$$\text{SPLIT} = in?x : T \rightarrow$$

$$((out_1.x \rightarrow \text{SPLIT}) \leftarrow x \in S \rightarrow (out_2.x \rightarrow \text{SPLIT}))$$

$$\text{SPLIT} \parallel \text{forget} \parallel = in \rightarrow (out_1 \rightarrow \text{SPLIT} \sqcap out_2 \rightarrow \text{SPLIT})$$

$$\text{forget}(in.x) = in$$

$$\text{forget}(out_1.x) = out_1$$

$$\text{forget}(out_2.x) = out_2$$

Termination and sequential composition

- What is termination? *Stop* vs. *Skip*

$$\textit{Skip} = \checkmark \rightarrow \textit{Stop}$$

- Sequential operator $P; Q$

$$P^* = P; P^*$$

$$\text{COPY}(in, out) = (in?x : T \rightarrow out.x \rightarrow \textit{Skip})^*$$

$$(\text{FOR } n = a, b \text{ DO } P) =$$

$$\textit{Skip} \text{ † } a > b \text{ ‡ } (P[a/n]; (\text{FOR } n = a + 1, b \text{ DO } P))$$

Examples

Beispiel 1 (Piping)

$$P \gg Q = (P \parallel f \parallel \parallel Q \parallel g \parallel) \setminus \{mid\}$$

$\{mid\}$

$$f(right) = mid \quad g(left) = mid$$

$$ITER = left?(data, x) \rightarrow right.(data, F(data, x)) \rightarrow ITER$$

$$ITER \gg ITER \gg \dots \gg ITER$$

Exercise: Develop a process which produces the fibonacci numbers.

Beispiel 2 (Buffers)

$$\begin{aligned}
 \mathit{BUFF}(\langle \rangle) &= \mathit{left}?x \rightarrow \mathit{BUFF}(\langle x \rangle) \\
 \mathit{BUFF}(s \hat{ } \langle a \rangle) &= ((\mathit{left}?x \rightarrow \mathit{BUFF}(\langle x \rangle \hat{ } s \hat{ } \langle a \rangle)) \sqcap \mathit{Stop}) \\
 &\quad \sqcap \mathit{right}.a \rightarrow \mathit{BUFF}(s)
 \end{aligned}$$

$$\mathit{PROD} = \mathit{produce}?x \rightarrow \mathit{left}.x \rightarrow \mathit{PROD}$$

$$\mathit{CONS} = \mathit{right}?x \rightarrow \mathit{consume}.x \rightarrow \mathit{CONS}$$

$$\begin{aligned}
 \mathit{PC_SYS} &= ((\mathit{PROD} \parallel \mathit{CONS}) \\
 &\quad \parallel \mathit{BUFF}(\langle \rangle)) \setminus \{\mathit{left}, \mathit{right}\} \\
 &\quad \{\mathit{left}, \mathit{right}\}
 \end{aligned}$$

Beispiel 3 (Shared variables)

$$SVAR(x) = (read.x \rightarrow SVAR(x))$$

$$\square (write?y \rightarrow SVAR(y))$$

$$USER(i) = \dots read?x \dots write.v \dots USER(i)$$

$$SVAR_SYS = (SVAR(init) \quad \parallel \quad \{read, write\} \\ (USER(1) \quad ||| \quad \dots \quad ||| \quad USER(n))) \setminus \{read, write\}$$

Summary: CSP operators vs. FDR operators

	CSP operator	FDR operator
prefix	\rightarrow	\dashrightarrow
external choice	\square	$[]$
internal choice	\sqcap	$ \sim $
parallel operator	\parallel	
	\parallel A	$[A]$
	$A \parallel B$	$[A \parallel B]$
		$[c \dashrightarrow c']$

	CSP operator	FDR operator
interleaving	$ $	$ $
hiding	\backslash	\backslash
sequential comp.	$;$	$;$
condition	$\langle b \rangle$	if-then-else
interrupt	Δ_e	\wedge
time-out	\triangleright	$[>$
stop-Process	Stop	STOP
skip-Process	Skip	SKIP