Independent Composition

Dependent Composition $P.Q$ (sequential execution)

P and Q must have exactly the same state variables

Independent Composition $P||Q$ (parallel execution)

P and Q must have completely different state variables

and the state variables of the composition are those of both P and Q

Ignoring time and space variables

$$P||Q \;=\; P \wedge Q$$

Independent Composition

example in integer variables x , y , and z

$x := x+1 \parallel y := y+2$ partition the variables:

put x in left part, put y and z in right part

$= \quad x' = x+1 \parallel y' = y+2 \wedge z' = z$

$= \quad x' = x+1 \wedge y' = y+2 \wedge z' = z$

reasonable partition rule

If either x' or $x :=$ appears in a process specification, then x belongs to that process

(then neither x' nor $x :=$ can appear in the other process specification).

If neither x' nor $x :=$ appears at all, then x can be placed on either side of the partition.

Independent Composition

example in variables x , y , and z

$x := y \parallel y := x$ partition: put x in left, y in right, z in either

$=$ $x' = y \wedge y' = x \wedge z' = z$

implementation of a process makes a private copy of the initial value of a variable belonging

to the other process if the other process contains an assignment to that variable

Independent Composition

example in binary variable b and integer variable x

$b := x=x \parallel x := x+1$ replace $x=x$ by $\top$

$= \quad b := \top \parallel x := x+1$

example in integer variables x and y

$(x := x+1 \,.\; x := x-1) \parallel y := x$

$= \quad ok \parallel y := x$

$= \quad y := x$

Independent Composition

$$(x := x+y.\ \ x := x \times y) \ \ \| \ \ (y := x-y.\ \ y := x/y)$$

You should have written

$$(x := x+y \ \| \ y := x-y) ; (x := x \times y \ \| \ y := x/y)$$

Independent Composition

$$P \| Q \;=\; \exists tP, tQ \cdot \quad (\text{substitute } tP \text{ for } t' \text{ in } P)$$
$$\wedge\ (\text{substitute } tQ \text{ for } t' \text{ in } Q)$$
$$\wedge\ t' = max\ tP\ tQ$$

laws

$(x := e \,\|\, y := f).\ P \;=\; (\text{for } x \text{ substitute } e \text{ and independently for } y \text{ substitute } f \text{ in } P)$

$P \| Q = Q \| P$ symmetry

$P \| (Q \| R) = (P \| Q) \| R$ associativity

$P \| Q \vee R = (P \| Q) \vee (P \| R)$ distributivity

$P \| \textbf{if } b \textbf{ then } Q \textbf{ else } R \textbf{ fi} = \textbf{if } b \textbf{ then } P \| Q \textbf{ else } P \| R \textbf{ fi}$ distributivity

$\textbf{if } b \textbf{ then } P \| Q \textbf{ else } R \| S \textbf{ fi} = \textbf{if } b \textbf{ then } P \textbf{ else } R \textbf{ fi} \,\|\, \textbf{if } b \textbf{ then } Q \textbf{ else } S \textbf{ fi}$ distributivity

List Concurrency

$$Li := e \quad = \quad L'i = e \ \wedge \ (\forall j : 0,..\#L \cdot j \neq i \Rightarrow L'j = Lj) \ \wedge \ x' = x \ \wedge \ y' = y \ \wedge \ ...$$

$$Li := e \quad = \quad L'i = e \ \wedge \ (\forall j : (\text{this part}) \cdot j \neq i \Rightarrow L'j = Lj) \ \wedge \ x' = x \ \wedge \ ...$$

example find the maximum item in a nonempty list

findmax 0 (#*L*) where

$$findmax \ = \ \langle i, j \rightarrow i<j \Rightarrow L' \ i = MAX \ L \ [i;..j] \rangle$$

findmax i j $\Leftarrow$ **if** $j-i = 1$ **then** *ok*

else (*findmax i* (*div* (*i*+*j*) 2) $\|$ *findmax* (*div* (*i*+*j*) 2) *j*).

L i := *max* (*L i*) (*L* (*div* (*i*+*j*) 2)) **fi**

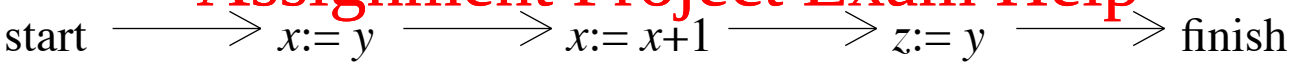
recursive time = *ceil* (*log* (*j*–*i*))

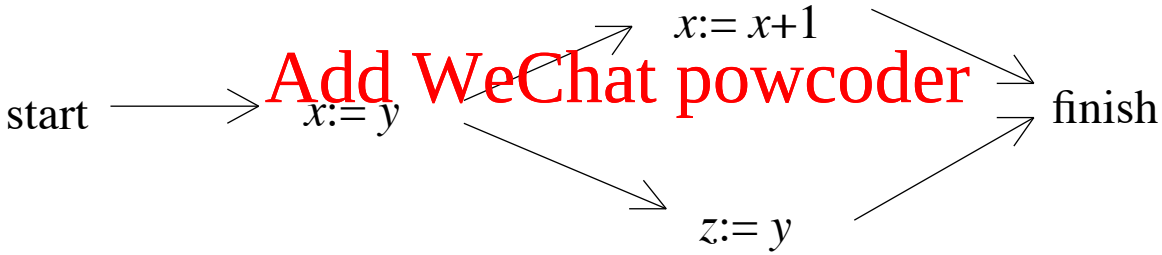
Sequential to Parallel Transformation

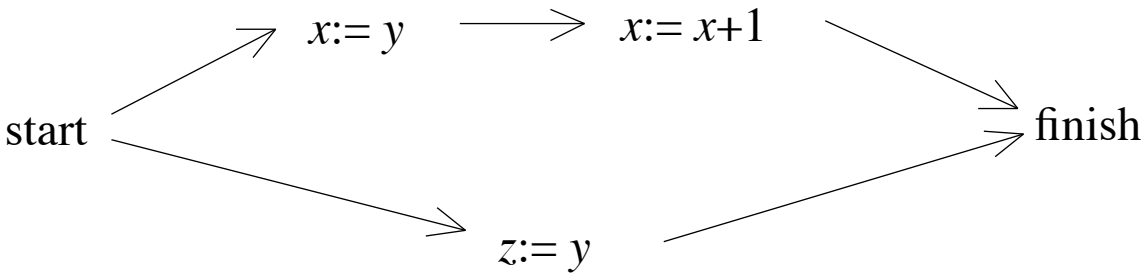
$x := y.\ x := x+1.\ z := y$

$=$ $x := y.\ (x := x+1 \parallel z := y)$

$=$ $(x := y.\ x := x+1) \parallel z := y$







Sequential to Parallel Transformation

rules

Whenever two programs occur in sequence, and neither assigns to a variable appearing in the other, they can be placed in parallel.

example $x := z.\ \ y := z$ becomes $x := z \parallel y := z$

Whenever two programs occur in sequence, and neither assigns to a variable assigned in the other, and no variable assigned in the first appears in the second, they can be placed in parallel; a copy must be made of the initial value of any variable appearing in the first and assigned in the second.

example $x := y.\ \ y := z$ becomes $c := y.\ (x := c \parallel y := z)$

Buffer

$produce = \cdots\cdots b := e \cdots\cdots$

$consume = \cdots\cdots x := b \cdots\cdots$

$control = produce.\ consume.\ control$

Buffer

$produce = \cdots\cdots b := e \cdots\cdots$

$consume = \cdots\cdots x := b \cdots\cdots$

$control = produce.\ newcontrol$

$newcontrol = consume.\ produce.\ newcontrol$

$produce = \cdots\cdots b := e \cdots\cdots$

$consume = \cdots\cdots x := b \cdots\cdots$

$control = produce.\ newcontrol$

$newcontrol = (consume \,||\, produce).\ newcontrol$

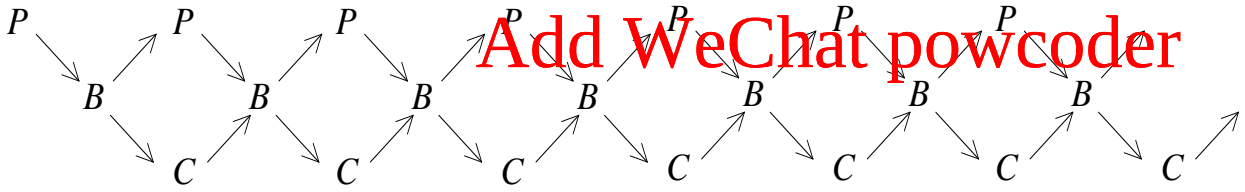
Buffer

$produce = \cdots\cdots b := e \cdots\cdots$

$consume = \cdots\cdots x := c \cdots\cdots$

$control = produce.\ newcontrol$

$newcontrol = c := b.\ (consume \parallel produce).\ newcontrol$



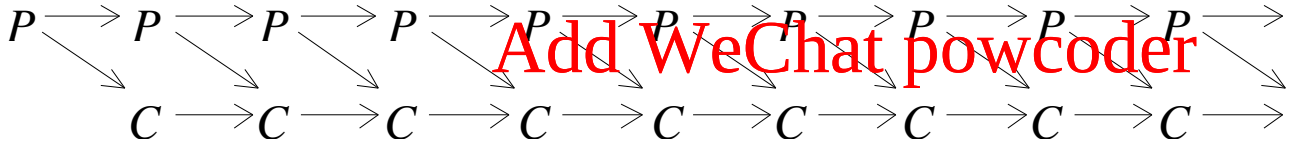
Buffer

$produce = \cdots\cdots b\ w := e.\ w := w+1\cdots\cdots$

$consume = \cdots\cdots x := b\ r.\ r := r+1\cdots\cdots$

$control = w := 0.\ r := 0.\ newcontrol$

$newcontrol = produce.\ consume.\ newcontrol$



Buffer

$produce = \cdots\cdots b\ w := e.\ w := mod\ (w+1)\ n\cdots\cdots$

$consume = \cdots\cdots x := b\ r.\ r := mod\ (r+1)\ n\cdots\cdots$

$control = w := 0.\ r := 0.\ newcontrol$

$newcontrol = produce.\ consume.\ newcontrol$

Insertion Sort

define

$$sort \;=\; \langle n \to \forall i, j{:}\ 0,..n{\cdot}\ i{\le}j \Rightarrow L\,i \le L\,j \rangle$$

$$swap \;=\; \langle i, j{:}\ 0,..\#L \to L\,i := L\,j \parallel L\,j := L\,i \rangle$$

$$sort'\ (\#L) \;\Leftarrow\; sort\ 0 \Rightarrow sort'\ (\#L)$$

$$sort\ 0 \Rightarrow sort'\ (\#L) \;\Leftarrow\; \textbf{for}\ n := 0;..\#L\ \textbf{do}\ sort\ n \Rightarrow sort'\ (n{+}1)\ \textbf{od}$$

$$sort\ n \Rightarrow sort'\ (n{+}1) \;\Leftarrow$$

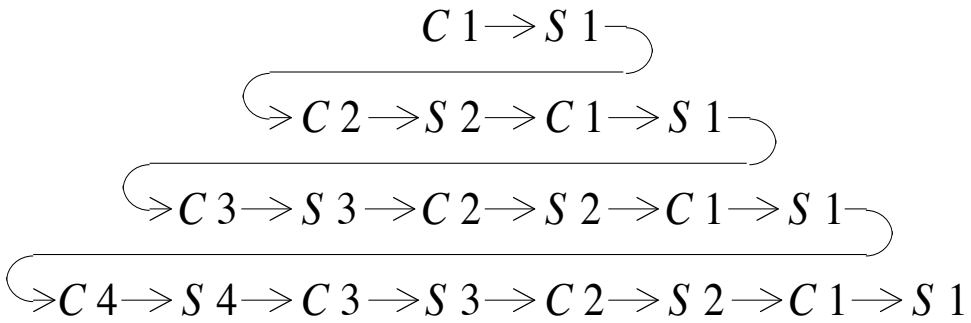
if $n{=}0$ **then** *ok*

else if $L\ (n{-}1) \le L\ n$ **then** *ok*

else $swap\ (n{-}1)\ n.\quad sort\ (n{-}1) \Rightarrow sort'\ n$ **fi fi**

$$[\ L\,0\ ;L\,1\ ;L\,2\ ;L\,3\ ;L\,4\ ]$$
$$0 \qquad 1 \qquad 2 \qquad 3 \qquad 4 \qquad 5$$

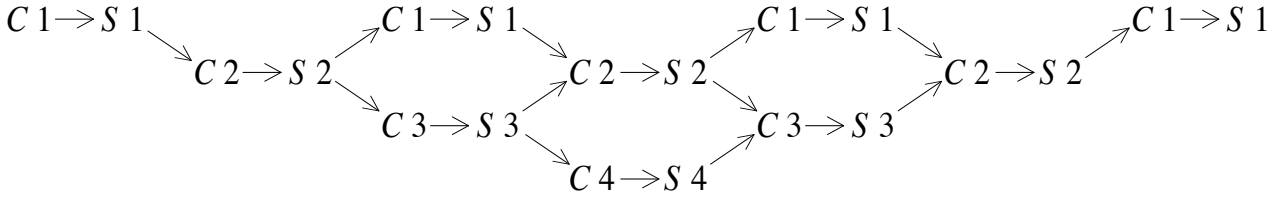
Insertion Sort



If $abs\ (i–j) > 1$ then $S\ i$ and $S\ j$ in parallel

If $abs\ (i–j) > 1$ then $S\ i$ and $C\ j$ in parallel

$C\ i$ and $C\ j$ in parallel



Dining Philosophers

Dining Philosophers

$$life = (P\,0 \vee P\,1 \vee P\,2 \vee P\,3 \vee P\,4).\ life$$

$$P\,i = up\,i.\ up(i{+}1).\ eat\,i.\ down\,i.\ down(i{+}1)$$

$$up\,i = chopstick\,i := \top$$

$$down\,i = chopstick\,i := \bot$$

$$eat\,i = \cdots\cdots chopstick\,i \cdots\cdots chopstick(i{+}1) \cdots\cdots$$

If $i{\neq}j$, $(up\,i.\,up\,j)$ becomes $(up\,i \parallel up\,j)$.

If $i{\neq}j$, $(up\,i.\,down\,j)$ becomes $(up\,i \parallel down\,j)$.

If $i{\neq}j$, $(down\,i.\,up\,j)$ becomes $(down\,i \parallel up\,j)$.

If $i{\neq}j$, $(down\,i.\,down\,j)$ becomes $(down\,i \parallel down\,j)$.

If $i{\neq}j \wedge i{+}1{\neq}j$, $(eat\,i.\,up\,j)$ becomes $(eat\,i \parallel up\,j)$.

If $i{\neq}j \wedge i{\neq}j{+}1$, $(up\,i.\,eat\,j)$ becomes $(up\,i \parallel eat\,j)$.

If $i{\neq}j \wedge i{+}1{\neq}j$, $(eat\,i.\,down\,j)$ becomes $(eat\,i \parallel down\,j)$.

If $i{\neq}j \wedge i{\neq}j{+}1$, $(down\,i.\,eat\,j)$ becomes $(down\,i \parallel eat\,j)$.

If $i{\neq}j \wedge i{+}1{\neq}j \wedge i{\neq}j{+}1$, $(eat\,i.\,eat\,j)$ becomes $(eat\,i \parallel eat\,j)$.

Dining Philosophers

$$
\begin{array}{lcl}
life & = & (P\,0 \vee P\,1 \vee P\,2 \vee P\,3 \vee P\,4).\ life \\
P\,i & = & up\ i.\ up(i{+}1).\ eat\ i.\ down\ i.\ down(i{+}1) \\
up\ i & = & chopstick\ i := \top \\
down\ i & = & chopstick\ i := \bot \\
eat\ i & = & \cdots\cdots chopstick\ i \cdots\cdots chopstick(i{+}1) \cdots\cdots
\end{array}
$$

$$
\begin{array}{lcl}
life & = & P\,0 \parallel P\,1 \parallel P\,2 \parallel P\,3 \parallel P\,4 \\
P\,i & = & (up\ i \parallel up(i{+}1)).\ eat\ i.\ (down\ i \parallel down(i{+}1)).\ P\,i
\end{array}
$$