

# Pi Calculus

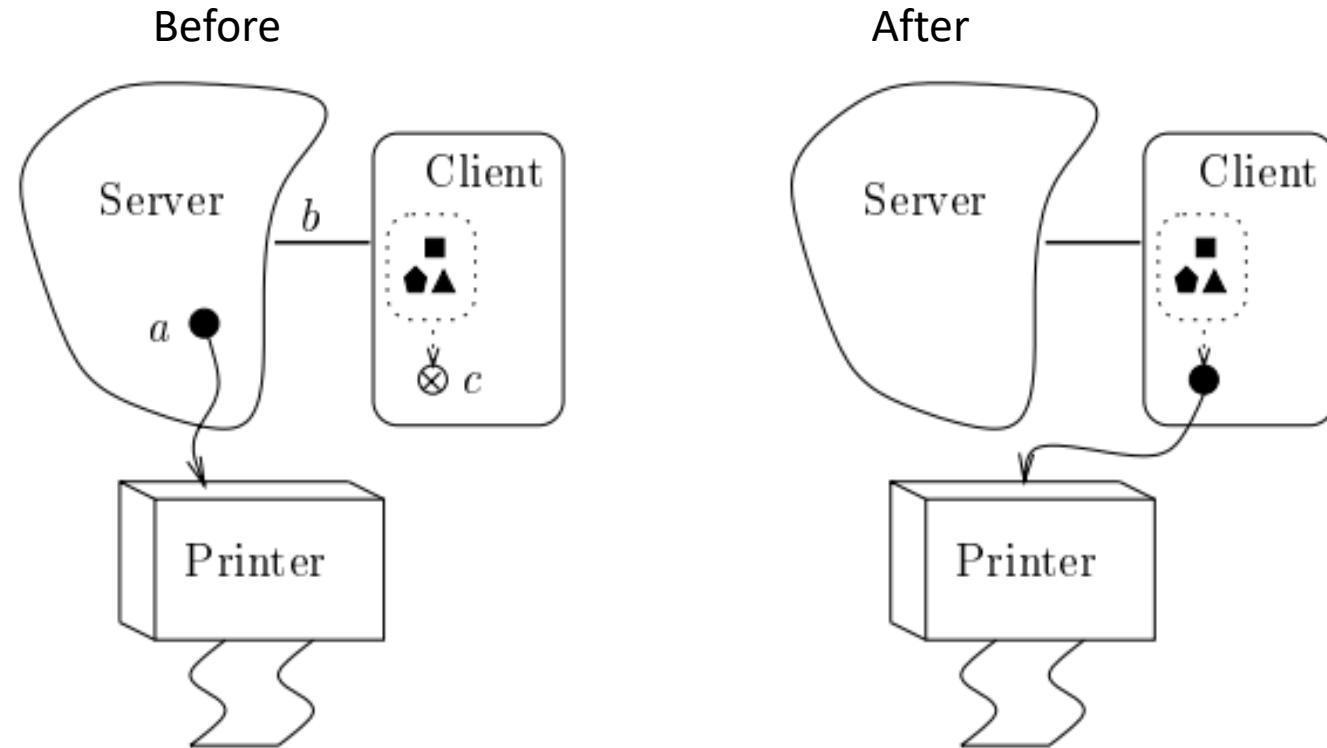
CS242

Lecture 8

# Motivation

- Assume we have three entities
  - A server
  - A client
  - A resource controlled by the server, such as a printer
- How do we model the client requesting access to the printer from the server?

# Picture



From "An Introduction to the Pi-Calculus". J. Parrow

# Process Calculi

- The essence of this problem is modeling the interaction of concurrent processes
- Models are known as *process calculi*
- There are many!
  - Much more diverse than the situation with, say, sequential programming
    - With functions, the only thing to observe is the output for a given input
  - Many more choices about what can be observed/modeled with concurrent systems

# Process Calculi

- All process calculi share communication over *channels*
- A *prefix* can
  - Send a message  $x$  on channel  $a$ :  $\bar{a}x$
  - Receive a message  $x$  on channel  $a$ :  $a(x)$
- An *agent* can be
  - Empty  $0$
  - A prefix  $p$  followed by an agent  $A$ :  $p.A$
- Example
  - Parallel composition of two agents:  $\bar{a}x.P \mid a(y).Q$

# Pi Calculus

- The distinguishing feature of the Pi Calculus is that channels are values
  - Channels can themselves be sent as messages
- As we shall see, this gives the Pi Calculus great expressivity
- Pi Calculus is one of the most popular process calculi

# Syntax

- Prefixes  $p = \bar{a}x \mid a(x)$

- Agents  $P = 0$

$p.P$

$P + P$

$P \mid P$

$x \text{ op } y \Rightarrow P$

$\nu x P$

$!P$

empty

prefix

choice

parallel

match (op is = or  $\neq$ )

restriction

iteration

# Semantics

- Prefixes  $p = \bar{a}x \mid a(x)$

- Agents  $P = 0$  empty  
 $p.P$  prefix  
 $P + P$  choice  
 $P \mid P$  parallel  
 $x \text{ op } y \Rightarrow P$  match (op is = or  $\neq$ )  
 $\nu x P$  restriction  
 $!P$  iteration

$$\bar{a}x.P \mid a(y).Q \rightarrow P \mid Q\{y := x\}$$



# Semantics

- Prefixes  $p = \bar{a}x \mid a(x)$

- Agents  $P = 0$  empty  
     $p.P$  prefix  
     $P + P$  choice  
     $P \mid P$  parallel  
     $x \text{ op } y \Rightarrow P$  match (op is = or  $\neq$ )  
     $\nu x P$  restriction  
     $!P$  iteration

Idiom:  $(x = y \Rightarrow P) + (x \neq y \Rightarrow Q) \rightarrow P$  if  $x = y$

# Semantics

- Prefixes  $p = \bar{a}x \mid a(x)$

- Agents  $P = 0$

$p.P$

empty

prefix

$P + P$

choice

$P \mid P$

parallel

$x \text{ op } y \Rightarrow P$

match (op is = or  $\neq$ )

$\nu x P$

restriction

$!P$

iteration

$!P \rightarrow P \mid !P$

# Semantics

- Prefixes  $p = \bar{a}x \mid a(x)$

- Agents  $P = 0$ 

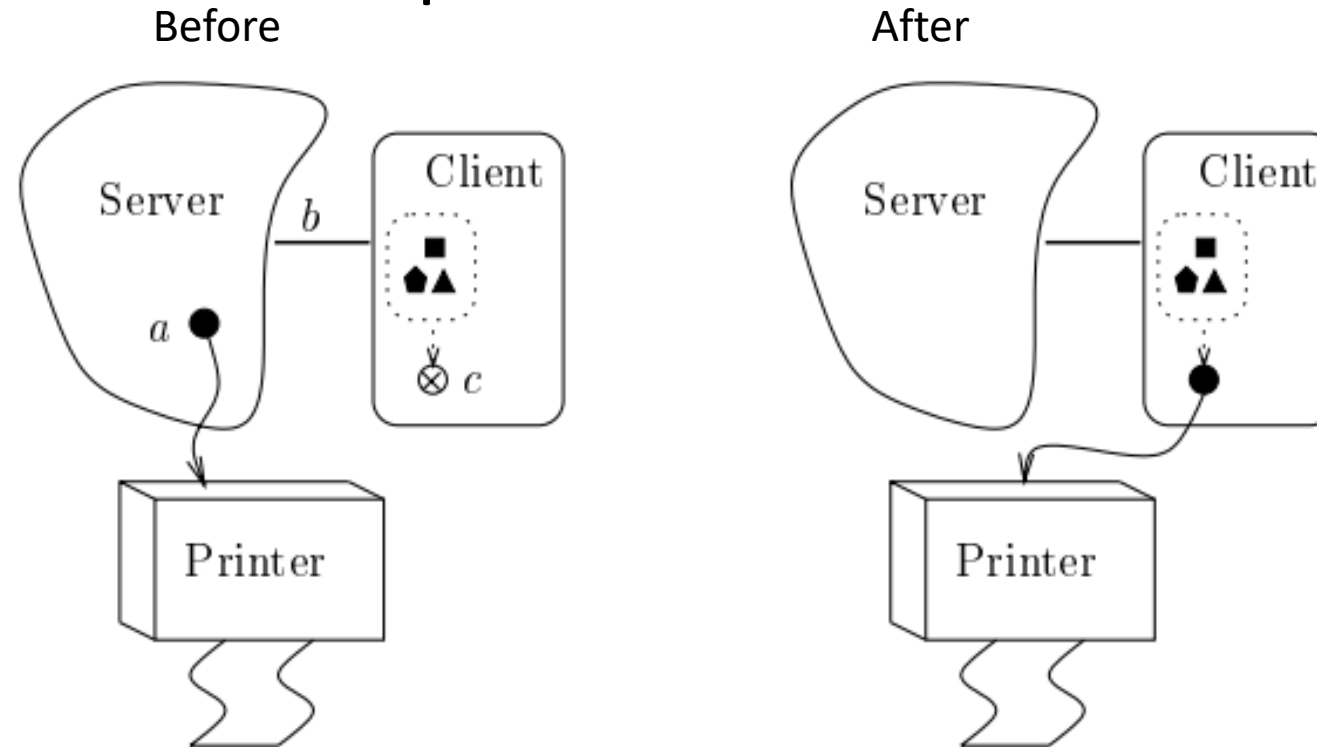
$p.P$	empty
$P + P$	prefix
$P \mid P$	choice
$x \text{ op } y \Rightarrow P$	parallel
$\nu x P$	match (op is = or $\neq$ )
$!P$	restriction
	iteration

$$!x(y) \mid !\bar{x}z \rightarrow x(y) \mid !x(y) \mid \bar{x}z \mid !\bar{x}z \rightarrow !x(y) \mid !\bar{x}z$$

# Restriction

- $\nu x P$  introduces a channel local to  $P$
- Example:  $R \mid \nu x (P \mid Q)$ 
  - $P$  and  $Q$  share the name  $x$  and can communicate with each other over  $x$
  - $R$  does not have access to  $x$
  - $R \mid \nu x (\bar{x}a.P \mid x(b).Q) \rightarrow R \nu x (P \mid Q\{b := a\})$
- But  $x$  can be passed to  $R$  on a different channel that  $R$  shares with  $P$  or  $Q$
- Passing a restricted channel expands its scope to include the new agent
- $a(z).R \mid \nu x (\bar{a}x.P \mid Q) \rightarrow \nu x (R\{z := x\} \mid P \mid Q)$
- Note alpha renaming may be required to avoid capture of existing  $x$  in  $R$

# Back to the Example ...



$$(\nu a (\bar{b}a.S \mid a(e).P)) \mid b(c).\bar{c}d.C$$

From “An Introduction to the Pi-Calculus”. J. Parrow

# Unbounded Computation

We can create as many fresh channel names as we like:

$$!(vx\ P) \rightarrow vx\ P \mid !(vx\ P) \rightarrow vx\ P \mid vy\ P\{x := y\} \mid !(vx\ P)$$

# What Can We Do With That?

- A lot!
- Example: Linked Lists

# Lists

- Use two constants (unused channels) Nil and Cons
- The translation function  $L$  takes a list  $l$  and a channel  $c$  and returns an agent that provides access to a representation of  $l$  through  $c$
- Idea: A list element communicates its kind (Nil or Cons) and then its arguments over  $c$ .

$$L(\text{Nil}, x) = \bar{x} \text{Nil}$$

$$L(\text{Cons } H \ T, x) = \nu y \ \nu z \ \bar{x} \text{Cons} . \bar{x} y . \bar{x} z \mid L(H, y) \mid L(T, z)$$



# Example

$$L(\text{Nil}, x) = \bar{x} \text{ Nil}$$

$$L(\text{Cons } H \ T, x) = v y \ v z \ \bar{x} \text{Cons}.\bar{x}y.\bar{x}z \mid L(H,y) \mid L(T,z)$$

$$L(\text{Cons Nil Nil}, x) =$$

$$v y \ v z \ \bar{x} \text{Cons}.\bar{x}y.\bar{x}z \mid L(\text{Nil},y) \mid L(\text{Nil},z) =$$

$$v y \ v z \ \bar{x} \text{Cons}.\bar{x}y.\bar{x}z \mid \bar{y} \text{ Nil} \mid L(\text{Nil},z) =$$

$$v y \ v z \ \bar{x} \text{Cons}.\bar{x}y.\bar{x}z \mid \bar{y} \text{ Nil} \mid \bar{z} \text{ Nil}$$

# Length of a List

Len computes the length of a list  $x$  and leaves the result on channel  $a$ .

$$\text{Len}(x,a) = \nu p \, \nu k \, \nu s \, \text{Fold}(k,s) \mid \text{Sum}(s,p,a) \mid \bar{p}0 \mid \bar{k}x$$
$$\text{Sum}(s,p,a) = !(s(i).p(n). \, i = 0 \Rightarrow \bar{a}n + i \neq 0 \Rightarrow \bar{p} \, n+i)$$
$$\text{Fold}(k,s) = !(k(z).z(y). \, y = \text{Nil} \Rightarrow \bar{s}0 + y = \text{Cons} \Rightarrow \bar{s}1.z(h).z(t).\bar{k}t)$$

# Example

$(vy \ vz \ \bar{x}\text{Cons}.\bar{x}y.\bar{x}z \mid \bar{y} \text{Nil} \mid \bar{z} \text{Nil})$

|

Len(x,a)

# Example

$(vy \ vz \ \bar{x}\text{Cons}.\bar{x}y.\bar{x}z \mid \bar{y} \text{Nil} \mid \bar{z} \text{Nil})$

|

$\text{Fold}(k,s)$

|

$\text{Sum}(s,p,a)$

|

$\bar{p}0$

|

$\bar{k}x$

(dropping top-level binders for p, k and s for brevity)

# Example

$(vy \ vz \ \bar{x}\text{Cons}.\bar{x}y.\bar{x}z \mid \bar{y} \text{Nil} \mid \bar{z} \text{Nil})$

|

$!(k(z).z(y). \ y = \text{Nil} \Rightarrow \bar{s}0 + y = \text{Cons} \Rightarrow \bar{s}1. z(h).z(t).\bar{k}t)$

|

$!(s(i).p(n). \ i = 0 \Rightarrow \bar{a}n + i \neq 0 \Rightarrow \bar{p} \ n+i)$

|

$\bar{p}0$

|

$\bar{k}x$

# Example

$(vy \ vz \ \bar{x}\text{Cons}.\bar{x}y.\bar{x}z \mid \bar{y} \text{Nil} \mid \bar{z} \text{Nil})$

|

$(k(z).z(y). \ y = \text{Nil} \Rightarrow \bar{s}0 + y = \text{Cons} \Rightarrow \bar{s}1. z(h).z(t).\bar{k}t) \mid \text{Fold}(k,s)$

|

$!(s(i).p(n). \ i = 0 \Rightarrow \bar{a}n + i \neq 0 \Rightarrow \bar{p} \ n+i)$

|

$\bar{p}0$

|

$\bar{k}x$

# Example

$(vy \ vz \ \bar{x}\text{Cons}.\bar{x}y.\bar{x}z \mid \bar{y} \text{Nil} \mid \bar{z} \text{Nil})$

|

$x(y). y = \text{Nil} \Rightarrow \bar{s}0 + y = \text{Cons} \Rightarrow \bar{s}1. x(h).x(t).\bar{k}t \mid \text{Fold}(k,s)$

|

$!(s(i).p(n). i = 0 \Rightarrow \bar{a}n + i \neq 0 \Rightarrow \bar{p} \ n+i)$

|

$\bar{p}0$

|

0

# Example

$(vy \ vz \ \bar{x}y. \bar{x}z \mid \bar{y} \text{ Nil} \mid \bar{z} \text{ Nil})$

|

$\text{Cons} = \text{Nil} \Rightarrow \bar{s}0 + \text{Cons} = \text{Cons} \Rightarrow \bar{s}1. x(h).x(t).\bar{k}t \mid \text{Fold}(k,s)$

|

$!(s(i).p(n). i = 0 \Rightarrow \bar{a}n + i \neq 0 \Rightarrow \bar{p} \ n+i)$

|

$\bar{p}0$

|

0



# Example

$(vy \ vz \ \bar{x}y.\bar{x}z \mid \bar{y} \text{ Nil} \mid \bar{z} \text{ Nil})$

|

$\text{Cons} = \text{Cons} \Rightarrow \bar{s}1.x(h).x(t).\bar{k}t \mid \text{Fold}(k,s)$

|

$!(s(i).p(n). i = 0 \Rightarrow \bar{a}n + i \neq 0 \Rightarrow \bar{p} \ n+i)$

|

$\bar{p}0$

|

0

# Example

$(vy \ vz \ \bar{x}y.\bar{x}z \mid \bar{y} \text{ Nil} \mid \bar{z} \text{ Nil})$   
|  
 $\bar{s}1.x(h).x(t).\bar{k}t \mid \text{Fold}(k,s)$   
|  
 $!(s(i).p(n). i = 0 \Rightarrow \bar{a}n + i \neq 0 \Rightarrow \bar{p} \ n+i)$   
|  
 $\bar{p}0$   
|  
 $0$

# Example

$(vy \ vz \ \bar{x}y.\bar{x}z \mid \bar{y} \text{ Nil} \mid \bar{z} \text{ Nil})$

$\mid$

$\bar{s}1.x(h).x(t).\bar{k}t \mid \text{Fold}(k,s)$

$\mid$

$(s(i).p(n). i = 0 \Rightarrow \bar{a}n + i \neq 0 \Rightarrow \bar{p} \ n+i) \mid \text{Sum}(s,p,a)$

$\mid$

$\bar{p}0$

$\mid$

$0$

# Example

$(vy \ vz \ \bar{x}y.\bar{x}z \mid \bar{y} \text{ Nil} \mid \bar{z} \text{ Nil})$

|

$x(h).x(t).\bar{k}t \mid \text{Fold}(k,s)$

|

$p(n). 1 = 0 \Rightarrow \bar{a}n + 1 \neq 0 \Rightarrow \bar{p} \ n+1 \mid \text{Sum}(s,p,a)$

|

$\bar{p}0$

|

0

# Example

$(vy \ vz \ \bar{x}y.\bar{x}z \mid \bar{y} \text{ Nil} \mid \bar{z} \text{ Nil})$

|

$x(h).x(t).\bar{k}t \mid \text{Fold}(k,s)$

|

$1 = 0 \Rightarrow \bar{a}n + 1 \neq 0 \Rightarrow \bar{p} \ 0+1 \mid \text{Sum}(s,p,a)$

|

0

|

0

# Example

$(vy \ vz \ \bar{x}y.\bar{x}z \mid \bar{y} \text{ Nil} \mid \bar{z} \text{ Nil})$

|

$x(h).x(t).\bar{k}t \mid \text{Fold}(k,s)$

|

$1 \neq 0 \Rightarrow \bar{p} \ 0+1 \mid \text{Sum}(s,p,a)$

|

0

|

0

# Example

$(vy \ vz \ \bar{x}y.\bar{x}z \mid \bar{y} \text{ Nil} \mid \bar{z} \text{ Nil})$

|

$x(h).x(t).\bar{k}t \mid \text{Fold}(k,s)$

|

$\bar{p} \ 0+1 \mid \text{Sum}(s,p,a)$

|

0

|

0

# Example

$(vy \ vz \ \bar{x}y.\bar{x}z \mid \bar{y} \text{ Nil} \mid \bar{z} \text{ Nil})$

$\mid$

$x(h).x(t).\bar{k}t \mid \text{Fold}(k,s)$

$\mid$

$\bar{p} \ 1$

$\mid$

$\text{Sum}(s,p,a)$

*Remove 0 agents as they will not contribute further.*



# Example

$(vy \ vz \ \bar{x}z \mid \bar{y} \text{ Nil} \mid \bar{z} \text{ Nil})$

$\mid$

$x(t).\bar{k}t \mid \text{Fold}(k,s)$

$\mid$

$\bar{p} \ 1$

$\mid$

$\text{Sum}(s,p,a)$

# Example

$(vy \ vz \ 0 \mid \bar{y} \text{ Nil} \mid \bar{z} \text{ Nil})$

$\mid$

$\bar{k}z \mid \text{Fold}(k,s)$

$\mid$

$\bar{p} \ 1$

$\mid$

$\text{Sum}(s,p,a)$

# Example

$(vy \ vz \ 0 \mid \bar{y} \text{ Nil} \mid \bar{z} \text{ Nil})$

$\mid$

$\bar{k}z$

$\mid$

$\text{Fold}(k,s)$

$\mid$

$\bar{p} \ 1$

$\mid$

$\text{Sum}(s,p,a)$

# Example

$(vy \ vz \ 0 \mid \bar{y} \text{ Nil} \mid \bar{z} \text{ Nil})$

$\mid$

$\bar{k}z$

$\mid$

$!(k(z).z(y). \ y = \text{Nil} \Rightarrow \bar{s}0 + y = \text{Cons} \Rightarrow \bar{s}1. z(h).z(t).\bar{k}t)$

$\mid$

$\bar{p} \ 1$

$\mid$

$!(s(i).p(n). \ i = 0 \Rightarrow \bar{a}n + i \neq 0 \Rightarrow \bar{p} \ n+i)$

# Example

$(vy \ vz \ 0 \mid \bar{y} \text{ Nil} \mid \bar{z} \text{ Nil})$

|

$\bar{k}z$

|

$(k(z).z(y). \ y = \text{Nil} \Rightarrow \bar{s}0 + y = \text{Cons} \Rightarrow \bar{s}1.z(h).z(t).\bar{k}t) \mid \text{Fold}(k,s)$

|

$\bar{p} \ 1$

|

$(s(i).p(n). \ i = 0 \Rightarrow \bar{a}n + i \neq 0 \Rightarrow \bar{p} \ n+i) \mid \text{Sum}(s,p,a)$

# Example

$(vy \ vz \ 0 \mid \bar{y} \text{ Nil} \mid \bar{z} \text{ Nil})$

$\mid$

$0$

$\mid$

$(z(y). \ y = \text{Nil} \Rightarrow \bar{s}0 + y = \text{Cons} \Rightarrow \bar{s}1. z(h).z(t).\bar{k}t) \mid \text{Fold}(k,s)$

$\mid$

$\bar{p} \ 1$

$\mid$

$(s(i).p(n). \ i = 0 \Rightarrow \bar{a}n + i \neq 0 \Rightarrow \bar{p} \ n+i) \mid \text{Sum}(s,p,a)$

# Example

$(vy \ vz \ 0 \mid \bar{y} \text{ Nil} \mid 0)$

|

0

|

$(\text{Nil} = \text{Nil} \Rightarrow \bar{s}0 + \text{Nil} = \text{Cons} \Rightarrow \bar{s}1. z(h).z(t).\bar{k}t) \mid \text{Fold}(k,s)$

|

$\bar{p} \ 1$

|

$(s(i).p(n). i = 0 \Rightarrow \bar{a}n + i \neq 0 \Rightarrow \bar{p} \ n+i) \mid \text{Sum}(s,p,a)$

# Example

$(vy \ vz \ 0 \mid \bar{y} \ \text{Nil} \mid 0)$

$\mid$

$0$

$\mid$

$\text{Nil} = \text{Nil} \Rightarrow \bar{s}0 \mid \text{Fold}(k,s)$

$\mid$

$\bar{p} \ 1$

$\mid$

$(s(i).p(n).i = 0 \Rightarrow \bar{a}n + i \neq 0 \Rightarrow \bar{p} \ n+i) \mid \text{Sum}(s,p,a)$



# Example

$(vy \ vz \ 0 \mid \bar{y} \ \text{Nil} \mid 0)$

$\mid$

$0$

$\mid$

$\bar{s}0 \mid \text{Fold}(k,s)$

$\mid$

$\bar{p} \ 1$

$\mid$

$(s(i).p(n).i = 0 \Rightarrow \bar{a}n + i \neq 0 \Rightarrow \bar{p} \ n+i) \mid \text{Sum}(s,p,a)$

# Example

$(vy \ vz \ 0 \mid \bar{y} \ \text{Nil} \mid 0)$

$\mid$

$0$

$\mid$

$0 \mid \text{Fold}(k,s)$

$\mid$

$\bar{p} \ 1$

$\mid$

$(p(n). \ 0 = 0 \Rightarrow \bar{a}n + 0 \neq 0 \Rightarrow \bar{p} \ n+0) \mid \text{Sum}(s,p,a)$

# Example

$(vy \ vz \ 0 \mid \bar{y} \text{ Nil} \mid 0)$

|

0

|

0 | Fold(k,s)

|

0

|

$(0 = 0 \Rightarrow \bar{a}1 + 0 \neq 0 \Rightarrow \bar{p} \ 1+0) \mid \text{Sum}(s,p,a)$

# Example

Fold(k,s)

|

$(0 = 0 \Rightarrow \bar{a}1 + 0 \neq 0 \Rightarrow \bar{p} 1+0)$

|

Sum(s,p,a)

*Remove 0 and useless agents ...*

# Example

Fold(k,s)

|

$(0 = 0 \Rightarrow \bar{a}1)$

|

Sum(s,p,a)

# Example

Fold(k,s)

|

$\bar{a}1$

|

Sum(s,p,a)

No further computation can be done at this point until an external agent decides to read the length from a.

# SKI

- We can also encode the SKI calculus or the lambda calculus in pi calculus.
- SKI is a little easier to explain ...

# Idea

- We represent combinator expressions as trees, as usual
- Each combinator takes an extra (last argument) for its parent
- There is a separate combinator for each possible number of children
  - E.g.,  $S_0(p)$  is  $S$  applied to no children and just the parent
  - $S_1(x, p)$  is  $S$  applied to 1 child  $x$  and the parent
  - $S_2(x, y, p)$  is  $S$  applied to 2 children  $x$  and  $y$  and the parent



# One New Construct

- Add recursive definitions

$$A(x_1, \dots, x_n) = P$$

- Now  $A(\dots)$  can also appear inside  $P$
- Not necessary, can be simulated with  $!P$ 
  - For definitions with single recursive calls:
  - Each invocation of  $P$  writes its result to a unique global channel  $p$
  - Where the result of recursive call is needed  $P$  reads the value from  $p$
  - See length of a list example ...

# Collecting Children

$$S_0(p) = \nu w \bar{p}w.(\bar{w}s_0 \mid S_0(p))$$

$$S_1(p,x) = \nu w \bar{p}w.(\bar{w}s_1. \bar{w}x \mid S_1(p,x))$$

$$S_2(p,x,y) = \nu w \bar{p}w.(\bar{w}s_2. \bar{w}x. \bar{w}y \mid S_1(p,x))$$

$$K_0(p) = \nu w \bar{p}w.(\bar{w}k_0 \mid K_0(p))$$

$$K_1(p,x) = \nu w \bar{p}w.(\bar{w}k_1. \bar{w}x \mid K_1(p,x))$$

$$I_0(p) = \nu w \bar{p}w.(\bar{w}i_0 \mid I_0(p))$$

$$I_1(p,x) = \nu w \bar{p}w.(\bar{w}i_0. \bar{w}x \mid I_1(p,x))$$

Each combinator sends a series of messages to its parent: first the private channel, then the combinator's identity, then its children. Afterwards, it resets itself for another application.

# Application

The application agent  $@(x,y,p)$  applies a left child  $x$  to a right child  $y$  with parent  $p$  using the rules of SKI.

$@(x,y,p) = x(w).w(a).$

$$\begin{aligned} & ( \quad a = s_0 \Rightarrow S_1(y,p) + \\ & \quad a = s_1 \Rightarrow w(u).S_2(u,y,p) + \\ & \quad a = s_2 \Rightarrow w(u).w(v). \nu p_1 \nu p_2 \quad @(u,y,p_1) \mid \\ & \quad \quad \quad @(v,y,p_2) \mid \\ & \quad \quad \quad @(p_1,p_2,p) + \end{aligned}$$
$$\begin{aligned} & a = k_0 \Rightarrow K_1(y,p) + \\ & a = k_1 \Rightarrow w(u).I_1(u,p) + \\ & a = i_0 \Rightarrow I_1(y,p) + \\ & a = i_1 \Rightarrow w(u).@(u,y,p) ) \end{aligned}$$

# Comments

Note two new tree nodes are created for the two new applications in  $S \cup v \ y = (u \ y) \ (v \ y)$ . Here  $p_1 = u \ y$  and  $p_2 = v \ y$ .

$$@(\mathbf{x}, \mathbf{y}, \mathbf{p}) = \mathbf{x}(\mathbf{w}) \cdot \mathbf{w}(\mathbf{a}).$$

$$\begin{aligned} & ( \quad a = s_0 \Rightarrow S_1(y, p) + \\ & \quad a = s_1 \Rightarrow w(u).S_2(u, y, p) + \\ & \quad a = s_2 \Rightarrow w(u).w(v). \nu p_1 \nu p_2 \quad @ (u, y, p_1) \mid \\ & \quad @ (v, y, p_2) \mid \\ & \quad @ (p_1, p_2, p) \quad + \end{aligned}$$

$$\begin{aligned} a = k_0 &\Rightarrow K_1(y, p) + \\ a = k_1 &\Rightarrow w(u) \cdot l_1(u, p) + \\ a = i_0 &\Rightarrow l_1(y, p) + \\ a = i_1 &\Rightarrow w(u) \cdot @ (u, y, p) \end{aligned}$$

# Comments

$\lambda x$  is not immediately rewritten to  $x$ . Instead we wait until it is applied to something  $(\lambda x) y = x y$ , which allows us to use a uniform three argument application rule.

$@(x,y,p) = x(w).w(a).$

$$\begin{aligned} & ( \quad a = s_0 \Rightarrow S_1(y,p) + \\ & \quad a = s_1 \Rightarrow w(u).S_2(u,y,p) + \\ & \quad a = s_2 \Rightarrow w(u).w(v). \nu p_1 \nu p_2 \quad @(u,y,p_1) \mid \\ & \quad \quad \quad @(v,y,p_2) \mid \\ & \quad \quad \quad @(p_1,p_2,p) + \end{aligned}$$
$$\begin{aligned} & a = k_0 \Rightarrow K_1(y,p) + \\ & a = k_1 \Rightarrow w(u). I_1(u,p) + \\ & a = i_0 \Rightarrow I_1(y,p) + \\ & a = i_1 \Rightarrow w(u).@(u,y,p) ) \end{aligned}$$

# Comments

$$S_0(p) = vw \bar{p}w.(\bar{w}s_0 \mid S_0(p))$$

$$S_1(p,x) = vw \bar{p}w.(\bar{w}s_1. \bar{w}x \mid S_1(p,x))$$

$$S_2(p,x,y) = vw \bar{p}w.(\bar{w}s_2. \bar{w}x. \bar{w}y \mid S_1(p,x))$$

$$K_0(p) = vw \bar{p}w.(\bar{w}k_0 \mid K_0(p))$$

$$K_1(p,x) = vw \bar{p}w.(\bar{w}k_1. \bar{w}x \mid K_1(p,x))$$

$$I_0(p) = vw \bar{p}w.(\bar{w}i_0 \mid I_0(p))$$

$$I_1(p,x) = vw \bar{p}w.(\bar{w}i_0. \bar{w}x \mid I_1(p,x))$$

The sharing of combinators means that the structure we build is actually a DAG not a tree – the same subterm may be shared by multiple parents. That is a much more efficient representation, as when a subterm is reduced it is shared by any other expressions that use that same subterm.

# Applications of Pi Calculus

- Languages based on pi calculus are usually *modeling languages*, not programming languages
- A modeling language is used for building models
  - And proving things about them
  - But not for running the models

# Applications of Pi Calculus

- Analysis of concurrent protocols
  - e.g., client/server
- Analysis of cryptographic protocols
- Systems biology
  - Building models of cell pathways
- Modeling business processes
- And more!



# What is Modeled?

- Proving properties of protocols
  - That a protocol doesn't deadlock
  - That a cryptographic protocol is secure against an attacker (another agent)
- In general
  - That a system is guaranteed to reach some state
  - That a system is guaranteed *not* to reach some state

# Summary

- Process calculi
  - Model concurrent agents
  - Fundamental operation is sending a message on a channel
  - Pi calculus allows the messages to also be channels
- A very active area
  - Lots of work
  - In many different fields where concurrent agents with a fixed set of interactions is a good description of how things work