Formal semantics

We now need to define how processes execute.

For example we would like

$$\mathsf{send}_c\langle M\rangle; P \mid \mathsf{recv}_c(x); Q \xrightarrow{\tau} P \mid Q[M/x]$$

where τ denotes a silent action (internal communication).

Let $fn(M)$ and $fn(P)$ be the set of free names in term M and process P respectively.

Let $fv(M)$ and $fv(P)$ be the set of free variables in term M and process P respectively.

Closed processes are processes without any free variables.

Let $P \triangleq \mathsf{new}\ c; \mathsf{new}\ K; \mathsf{recv}_d(x); \mathsf{case}\ x\ \mathsf{of}\ \{y\}_{K'} : \mathsf{send}_d\langle \{y\}_K, z, c\rangle; \mathsf{halt}$.

We have

$fn(\mathsf{send}_d\langle \{y\}_K, z, c\rangle; \mathsf{halt}) = \{c, d, K\}$

$fv(\mathsf{send}_d\langle \{y\}_K, z, c\rangle; \mathsf{halt}) = \{y, z\}$

$fn(\mathsf{case}\ x\ \mathsf{of}\ \{y\}_{K'} : \mathsf{send}_d\langle \{y\}_K, z, c\rangle; \mathsf{halt}) = \{c, d, K, K'\}$

$fv(\mathsf{case}\ x\ \mathsf{of}\ \{y\}_{K'} : \mathsf{send}_d\langle \{y\}_K, z, c\rangle; \mathsf{halt}) = \{x, z\}$

$fn(P) = \{d, K'\}$

$fv(P) = \{z\}$

$fn(\{y\}_K) = \{K\}$

$fv(\{y\}_K) = \{y\}$

First we define reduction relation $>$ on closed processes:

$$
\begin{array}{rl}
\mathsf{repeat}\ P & > P \mid \mathsf{repeat}\ P \\
\mathsf{check}\ (M == M); P & > P \\
\mathsf{let}\ (x, y) = (M, N); P & > P[M/x, N/y] \\
\mathsf{case}\ 0\ \mathsf{of}\ 0 : P,\ \mathsf{succ}\ (x) : Q & > P \\
\mathsf{case}\ \mathsf{succ}\ (M)\ \mathsf{of}\ 0 : P,\ \mathsf{succ}\ (x) : Q & > Q[M/x] \\
\mathsf{case}\ \{M\}_N\ \mathsf{of}\ \{x\}_N : P & > P[M/x]
\end{array}
$$

When these rules cannot be applied, it means that the process cannot be simplified.

The following processes cannot be simplified, hence cannot be executed further.

$\mathsf{check}\ (0 == \mathsf{succ}\ (0); P$ (comparison fails).

$\mathsf{let}\ (x, y) = 0; P$ (unpairing fails)

$\mathsf{case}\ (M, N)\ \mathsf{of}\ 0 : P,\ \mathsf{succ}\ (x) : Q$ (not an integer)

$\mathsf{case}\ (M, N)\ \mathsf{of}\ \{x, y\}_K : P$ (not an encrypted message)

$\mathsf{case}\ \{M, N\}_{K'}\ \mathsf{of}\ \{x, y\}_K : P$ where $K \neq K'$

When these rules cannot be applied, it means that the process cannot be simplified.

The following processes cannot be simplified, hence cannot be executed further.

$\mathsf{check}\ (0 == \mathsf{succ}\ (0); P$ (comparison fails).

$\mathsf{let}\ (x, y) = 0; P$ (unpairing fails)

$\mathsf{case}\ (M, N)\ \mathsf{of}\ 0 : P,\ \mathsf{succ}\ (x) : Q$ (not an integer)

$\mathsf{case}\ (M, N)\ \mathsf{of}\ \{x, y\}_K : P$ (not an encrypted message)

$\mathsf{case}\ \{M, N\}_{K'}\ \mathsf{of}\ \{x, y\}_K : P$ where $K \neq K'$

This is also based on the perfect cryptography assumption: distinct terms represent distinct messages.

A barb β is either

- a name n (representing input on channel n), or
- a co-name $\overline{n}$ (representing output on channel n)

An action is either

- a barb (representing input or output to the outside world), or
- τ (representing a silent action i.e. internal communication)

We write $P \xrightarrow{\alpha} Q$ to mean that P makes action α after which Q is the remaining process that is left to be executed.

Commitment relation Consider again $\mathsf{send}_c\langle M\rangle; P \mid \mathsf{recv}_c(x); Q$

Commitment relation Consider again $\mathsf{send}_c\langle M\rangle; P \mid \mathsf{recv}_c(x); Q$

The first subprocess makes an output action on channel c.

We will represent it as $\mathsf{send}_c\langle M\rangle; P \xrightarrow{\overline{c}} \langle M\rangle P$.

$\langle M\rangle P$ is called a concretion: it represents a commitment to output message M after which P will be executed.

Commitment relation Consider again $\mathsf{send}_c\langle M\rangle; P \mid \mathsf{recv}_c(x); Q$

The first subprocess makes an output action on channel c.

We will represent it as $\mathsf{send}_c\langle M\rangle; P \xrightarrow{\overline{c}} \langle M\rangle P$.

$\langle M\rangle P$ is called a concretion: it represents a commitment to output message M after which P will be executed.

The second subprocess makes an input action on channel c.

We will represent it as $\mathsf{recv}_c(x); Q \xrightarrow{c} (x)Q$.

$(x)Q$ is called an abstraction:it represents a commitment to input some x after which Q will be executed.

Commitment relation Consider again $\mathsf{send}_c\langle M\rangle; P \mid \mathsf{recv}_c(x); Q$

The first subprocess makes an output action on channel c.

We will represent it as $\mathsf{send}_c\langle M\rangle; P \xrightarrow{\overline{c}} \langle M\rangle P$.

$\langle M\rangle P$ is called a concretion: it represents a commitment to output message M after which P will be executed.

The second subprocess makes an input action on channel c.

We will represent it as $\mathsf{recv}_c(x); Q \xrightarrow{c} (x)Q$.

$(x)Q$ is called an abstraction:it represents a commitment to input some x after which Q will be executed.

Abstractions and concretions can be combined:

$\langle M\rangle P \mathbin{@} (x)Q = P \mid Q[M/x]$

Formally an abstraction F is of the form

$$(x_1, \ldots, x_k)P$$

where $k \geq 0$ and P is a process.

A concretion C is of the form

$$(\mathsf{new}\ n_1, \ldots, n_l)\langle M_1, \ldots, M_k \rangle P$$

where $n_1, \ldots, n_l$ are names, $l, k \geq 0$ and P is a process.

Formally an abstraction F is of the form

$$(x_1, \ldots, x_k)P$$

where $k \geq 0$ and P is a process.

A concretion C is of the form

$$(\mathsf{new}\ n_1, \ldots, n_l)\langle M_1, \ldots, M_k \rangle P$$

where $n_1, \ldots, n_l$ are names, $l, k \geq 0$ and P is a process.

For $F \triangleq (x_1, \ldots, x_k)P$ and $C \triangleq (\mathsf{new}\ n_1, \ldots, n_l)\langle M_1, \ldots, M_k \rangle Q$

with $\{n_1, \ldots, n_l\} \cap fn(P) = \emptyset$ we define interaction of F and C as

$$\boxed{\begin{aligned} F \ @\ C &\triangleq \mathsf{new}\ n_1; \ldots \mathsf{new}\ n_l; (P[M_1/x_1, \ldots, M_k/x_k] \mid Q) \\ C \ @\ F &\triangleq \mathsf{new}\ n_1; \ldots \mathsf{new}\ n_l; (Q \mid P[M_1/x_1, \ldots, M_k/x_k]) \end{aligned}}$$

An agent A is an abstraction, concretion or a process.

We write the commitment relation as $P \xrightarrow{\alpha} A$ where P is a closed process, A is a closed agent ($fv(A) = \emptyset$) and α is an action.

An agent A is an abstraction, concretion or a process.

We write the commitment relation as $P \xrightarrow{\alpha} A$ where P is a closed process, A is a closed agent $(fv(A) = \emptyset)$ and α is an action.

$$\mathsf{send}_m\langle M_1, \ldots, M_k\rangle; P \xrightarrow{\overline{m}} (\mathsf{new}\ )\langle M_1, \ldots, M_k\rangle P$$

An agent A is an abstraction, concretion or a process.

We write the commitment relation as $P \xrightarrow{\alpha} A$ where P is a closed process, A is a closed agent ($fv(A) = \emptyset$) and α is an action.

$$\mathsf{send}_m\langle M_1, \ldots, M_k\rangle; P \xrightarrow{\overline{m}} (\mathsf{new}\ )\langle M_1, \ldots, M_k\rangle P$$

$$\mathsf{recv}_m(x_1, \ldots, x_k); P \xrightarrow{m} (x_1, \ldots, x_k)P$$

An agent A is an abstraction, concretion or a process.

We write the commitment relation as $P \xrightarrow{\alpha} A$ where P is a closed process, A is a closed agent ($fv(A) = \emptyset$) and α is an action.

$$\mathsf{send}_m\langle M_1, \ldots, M_k\rangle; P \xrightarrow{\overline{m}} (\mathsf{new}\ )\langle M_1, \ldots, M_k\rangle P$$

$$\mathsf{recv}_m(x_1, \ldots, x_k); P \xrightarrow{m} (x_1, \ldots, x_k)P$$

$$\frac{P \xrightarrow{m} F \quad Q \xrightarrow{\overline{m}} C}{P \mid Q \xrightarrow{\tau} F @ C}$$

An agent A is an abstraction, concretion or a process.

We write the commitment relation as $P \xrightarrow{\alpha} A$ where P is a closed process, A is a closed agent $(fv(A) = \emptyset)$ and α is an action.

$$\mathsf{send}_m\langle M_1, \ldots, M_k\rangle; P \xrightarrow{\overline{m}} (\mathsf{new}\ )\langle M_1, \ldots, M_k\rangle P$$

$$\mathsf{recv}_m(x_1, \ldots, x_k); P \xrightarrow{m} (x_1, \ldots, x_k)P$$

$$\frac{P \xrightarrow{m} F \quad Q \xrightarrow{\overline{m}} C}{P \mid Q \xrightarrow{\tau} F \mathbin{@} C}$$

$$\frac{P \xrightarrow{\overline{m}} C \quad Q \xrightarrow{m} F}{P \mid Q \xrightarrow{\tau} C \mathbin{@} F}$$

Example

Define

$P \triangleq \mathsf{send}_c\langle \mathsf{succ}\ (0)\rangle; \mathsf{halt}$

$Q \triangleq \mathsf{recv}_c(x); \mathsf{case}\ x\ \mathsf{of}\ 0 : \mathsf{halt},\ \mathsf{succ}\ (y) : (\mathsf{send}_d\langle y\rangle; \mathsf{halt})$

From our rules we have

Example

Define

$P \triangleq \mathsf{send}_c\langle \mathsf{succ}\ (0)\rangle ; \mathsf{halt}$

$Q \triangleq \mathsf{recv}_c(x) ; \mathsf{case}\ x\ \mathsf{of}\ 0 : \mathsf{halt},\ \mathsf{succ}\ (y) : (\mathsf{send}_d\langle y\rangle ; \mathsf{halt})$

From our rules we have

$$P \xrightarrow{\bar{c}} \langle \mathsf{succ}\ (0)\rangle \mathsf{halt}$$

$(\langle M_1, \ldots, M_k\rangle P'$ denotes $(\mathsf{new}\ )\langle M_1, \ldots, M_k\rangle P')$

Example

Define

$P \triangleq \mathsf{send}_c\langle \mathsf{succ}\ (0)\rangle; \mathsf{halt}$

$Q \triangleq \mathsf{recv}_c(x); \mathsf{case}\ x\ \mathsf{of}\ 0 : \mathsf{halt},\ \mathsf{succ}\ (y) : (\mathsf{send}_d\langle y\rangle; \mathsf{halt})$

From our rules we have

$$P \xrightarrow{\bar{c}} \langle \mathsf{succ}\ (0)\rangle \mathsf{halt}$$

$$(\langle M_1, \ldots, M_k\rangle P' \text{ denotes } (\mathsf{new}\ )\langle M_1, \ldots, M_k\rangle P')$$

$$Q \xrightarrow{c} (x)\mathsf{case}\ x\ \mathsf{of}\ 0 : \mathsf{halt},\ \mathsf{succ}\ (y) : (\mathsf{send}_d\langle y\rangle; \mathsf{halt})$$

Example

Define

$P \triangleq \mathsf{send}_c\langle \mathsf{succ}\ (0)\rangle; \mathsf{halt}$

$Q \triangleq \mathsf{recv}_c(x); \mathsf{case}\ x\ \mathsf{of}\ 0 : \mathsf{halt},\ \mathsf{succ}\ (y) : (\mathsf{send}_d\langle y\rangle; \mathsf{halt})$

From our rules we have

$$P \xrightarrow{\bar{c}} \langle \mathsf{succ}\ (0)\rangle \mathsf{halt}$$

$$(\langle M_1, \ldots, M_k\rangle P' \text{ denotes } (\mathsf{new}\ )\langle M_1, \ldots, M_k\rangle P')$$

$$Q \xrightarrow{c} (x)\mathsf{case}\ x\ \mathsf{of}\ 0 : \mathsf{halt},\ \mathsf{succ}\ (y) : (\mathsf{send}_d\langle y\rangle; \mathsf{halt})$$

$$P \mid Q \xrightarrow{\tau} \mathsf{halt} \mid \mathsf{case}\ \mathsf{succ}\ (0)\ \mathsf{of}\ 0 : \mathsf{halt},\ \mathsf{succ}\ (y) : (\mathsf{send}_d\langle y\rangle; \mathsf{halt})$$

Example

Define

$P \triangleq \mathsf{send}_c\langle \mathsf{succ}\ (0)\rangle; \mathsf{halt}$

$Q \triangleq \mathsf{recv}_c(x); \mathsf{case}\ x\ \mathsf{of}\ 0 : \mathsf{halt},\ \mathsf{succ}\ (y) : (\mathsf{send}_d\langle y\rangle; \mathsf{halt})$

From our rules we have

$$P \xrightarrow{\bar{c}} \langle \mathsf{succ}\ (0)\rangle \mathsf{halt}$$

$$(\langle M_1, \ldots, M_k\rangle P' \text{ denotes } (\mathsf{new}\ )\langle M_1, \ldots, M_k\rangle P')$$

$$Q \xrightarrow{c} (x)\mathsf{case}\ x\ \mathsf{of}\ 0 : \mathsf{halt},\ \mathsf{succ}\ (y) : (\mathsf{send}_d\langle y\rangle; \mathsf{halt})$$

$$P \mid Q \xrightarrow{\tau} \mathsf{halt} \mid \mathsf{case}\ \mathsf{succ}\ (0)\ \mathsf{of}\ 0 : \mathsf{halt},\ \mathsf{succ}\ (y) : (\mathsf{send}_d\langle y\rangle; \mathsf{halt})$$

$$\xrightarrow{\bar{d}} \langle 0\rangle(\mathsf{halt} \mid \mathsf{halt}) \quad \text{using the following rules\ldots}$$

$$\frac{P > Q \quad Q \xrightarrow{\alpha} A}{P \xrightarrow{\alpha} A}$$

$$\frac{P \xrightarrow{\alpha} A}{P \mid Q \xrightarrow{\alpha} A \mid Q} \qquad \frac{Q \xrightarrow{\alpha} A}{P \mid Q \xrightarrow{\alpha} P \mid A}$$

where

$$P_1 \mid (x_1, \ldots, x_k)P_2 \triangleq (x_1, \ldots, x_k)(P_1 \mid P_2)$$

$$P_1 \mid (\mathsf{new}\ n_1, \ldots, n_k)\langle M_1, \ldots, M_l\rangle P_2 \triangleq (\mathsf{new}\ n_1, \ldots, n_k)\langle M_1, \ldots, M_l\rangle(P_1 \mid P_2)$$

provided that $x_1, \ldots, x_k \notin fv(P_1)$ and $n_1, \ldots, n_k \notin fn(P_1)$

For the previous example we have:

$$\mathsf{case}\ \mathsf{succ}\ (0)\ \mathsf{of}\ 0 : \mathsf{halt},\ \mathsf{succ}\ (y) : (\mathsf{send}_d\langle y\rangle; \mathsf{halt}) > \mathsf{send}_d\langle 0\rangle; \mathsf{halt}$$

and

$$\mathsf{send}_d\langle 0\rangle; \mathsf{halt} \xrightarrow{\bar{d}} \langle 0\rangle \mathsf{halt}$$

hence

$$\mathsf{case}\ \mathsf{succ}\ (0)\ \mathsf{of}\ 0 : \mathsf{halt},\ \mathsf{succ}\ (y) : (\mathsf{send}_d\langle y\rangle; \mathsf{halt}) \xrightarrow{\bar{d}} \langle 0\rangle \mathsf{halt}$$

hence

$$\begin{aligned}\mathsf{halt} \mid \mathsf{case}\ \mathsf{succ}\ (0)\ \mathsf{of}\ 0 : \mathsf{halt},\ \mathsf{succ}\ (y) : (\mathsf{send}_d\langle y\rangle; \mathsf{halt}) &\xrightarrow{\bar{d}} \mathsf{halt} \mid \langle 0\rangle \mathsf{halt} \\ &= \langle 0\rangle(\mathsf{halt} \mid \mathsf{halt})\end{aligned}$$

Consider $P \triangleq (\mathsf{recv}_c(x); P_1) \mid \mathsf{new}\ c; (\mathsf{send}_c\langle 0\rangle; P_2 \mid \mathsf{recv}_c(x); P_3)$

We would like $P \xrightarrow{\tau} (\mathsf{recv}_c(x); P_1) \mid \mathsf{new}\ c; (P_2 \mid P_3[0/x])$

but not $P \xrightarrow{\tau} P_1[0/x] \mid \mathsf{new}\ n; (P_2 \mid \mathsf{recv}_c(x); P_3)$

Consider $P \triangleq (\mathsf{recv}_c(x); P_1) \mid \mathsf{new}\ c; (\mathsf{send}_c\langle 0\rangle; P_2 \mid \mathsf{recv}_c(x); P_3)$

We would like $P \xrightarrow{\tau} (\mathsf{recv}_c(x); P_1) \mid \mathsf{new}\ c; (P_2 \mid P_3[0/x])$

but not $P \xrightarrow{\tau} P_1[0/x] \mid \mathsf{new}\ n; (P_2 \mid \mathsf{recv}_c(x); P_3)$

Hence we have the rule

$$\boxed{\frac{P \xrightarrow{\alpha} A \quad \alpha \notin \{n, \overline{n}\}}{\mathsf{new}\ n; P \xrightarrow{\alpha} \mathsf{new}\ n; A}}$$

where

$$(\mathsf{new}\ m)(x_1, \ldots, x_k)P \triangleq (x_1, \ldots, x_k)\mathsf{new}\ m; P$$

$$(\mathsf{new}\ m)(\mathsf{new}\ m_1, \ldots, m_k)\langle M_1, \ldots, M_l\rangle P \triangleq (\mathsf{new}\ m, m_1, \ldots, m_k)\langle M_1, \ldots, M_l\rangle P$$

provided that $m \notin \{m_1, \ldots, m_k\}$

We have $\mathsf{send}_c\langle 0\rangle; P_2 \xrightarrow{\overline{c}} \langle 0\rangle P_2$

and $\mathsf{recv}_c(x); P_3 \xrightarrow{c} (x)P_3$

hence $\mathsf{send}_c\langle 0\rangle; P_2 \mid \mathsf{recv}_c(x); P_3 \xrightarrow{\tau} \langle 0\rangle P_2 \mathbin{@} (x)P_3 = P_2 \mid P_3[0/x]$

Since $\tau \notin \{\overline{c}, c\}$

hence $\mathsf{new}\ c; (\mathsf{send}_c\langle 0\rangle; P_2 \mid \mathsf{recv}_c(x); P_3) \xrightarrow{\tau} \mathsf{new}\ c; (P_2 \mid P_3[0/x])$

Hence $(\mathsf{recv}_c(x); P_1) \mid \mathsf{new}\ c; (\mathsf{send}_c\langle 0\rangle; P_2 \mid \mathsf{recv}_c(x); P_3)$

$$\xrightarrow{\tau} (\mathsf{recv}_c(x); P_1) \mid \mathsf{new}\ c; (P_2 \mid P_3[0/x])$$

Consider $P \triangleq (\mathsf{new}\ K; \mathsf{send}_c\langle K\rangle; \mathsf{halt}) \mid (\mathsf{recv}_c(x); \mathsf{send}_d\langle x\rangle; \mathsf{halt})$

We have $\mathsf{send}_c\langle K\rangle; \mathsf{halt} \xrightarrow{\overline{c}} (\mathsf{new}\ )\langle K\rangle\mathsf{halt}$

hence $\mathsf{new}\ K; \mathsf{send}_c\langle K\rangle; \mathsf{halt} \xrightarrow{\overline{c}} \mathsf{new}\ K; (\mathsf{new}\ )\langle K\rangle\mathsf{halt} = (\mathsf{new}\ K)\langle K\rangle\mathsf{halt}$

Also $\mathsf{recv}_c(x); \mathsf{send}_d\langle x\rangle; \mathsf{halt} \xrightarrow{c} (x)\mathsf{send}_d\langle x\rangle; \mathsf{halt}$

Hence
$P \xrightarrow{\tau} (\mathsf{new}\ K)\langle K\rangle\mathsf{halt}\ @\ (x)\mathsf{send}_d\langle x\rangle; \mathsf{halt} = (\mathsf{new}\ K)(\mathsf{halt} \mid \mathsf{send}_d\langle K\rangle; \mathsf{halt})$

Equivalence on processes

A test is of the form (Q, β) where Q is a closed process and β is a barb.

A process P passes the test (Q, β) iff

$$(P \mid Q) \xrightarrow{\tau} Q_1 \ldots \xrightarrow{\tau} Q_n \xrightarrow{\beta} A$$

for some $n \geq 0$, some processes $Q_1, \ldots, Q_n$ and some agent A.

Q is the "environment" and we test whether the process together with the environment inputs or outputs on a particular channel.

Testing preorder $P_1 \sqsubseteq P_2$ iff for every test (Q, β), if P_1 passes (Q, β) then P_2 passes (Q, β).

Testing equivalence $P_1 \simeq P_2$ iff $P_1 \sqsubseteq P_2$ and $P_2 \sqsubseteq P_1$.

Secrecy

Consider process P with only free variable x.

We will consider x as secret if for all terms M, M' we have $P[M/x] \simeq P[M'/x]$.

I.e. an observer cannot detect any changes in the value of x.

Example Consider $P \triangleq \mathsf{send}_c\langle x\rangle; \mathsf{halt}$.

x is being sent out on a public channel. Consider test $(Q, \overline{d})$ where

environment $Q \triangleq \mathsf{recv}_c(x); \mathsf{check}\ (x == 0); \mathsf{send}_d\langle 0\rangle; \mathsf{halt}$.

We have $P[0/x] \mid Q \xrightarrow{\tau} \mathsf{halt} \mid \mathsf{send}_d\langle 0\rangle; \mathsf{halt} \xrightarrow{\overline{d}} \langle 0\rangle(\mathsf{halt} \mid \mathsf{halt})$.

Hence $P[0/x]$ passes the test. However $P[\mathsf{succ}\ (0)/x]$ fails the test.

Hence P does not preserve secrecy of x.