π

Diogo Raphael Cravo

Universidade Federal do Rio Grande do Sul (UFRGS)

Junho de 2019

Sumário

Introdução

Introdução

- Inspirado em CCS (Calculus of Communicating Systems, criado por R. Milner no final da década de 70)
- Criado no final da década de 80, por R. Milner, J. Parrow e D. Walker, em "A Calculus of **Mobile Processes**"
- Serve de base para bígrafos (não confundir com bipartido)

Introdução

- Dois conceitos fundamentais:
  1. nomes/elos/canais
  2. processos/agentes
- Duas noções de mobilidade:
  1. passagem de nomes, trafega informação, ex: links na web
  2. passagem de processos, trafega instruções, ex: execução remota de código
- Nomes são canais, ao passar nomes, um processo pode causar **mudanças de escopo** e criar novas possibilidades de comunicação

Introdução

Cálculo λ (funções) e cálculo π (processos):

- $\lambda \rightarrow$ computação funcional/linear, funções como base
- $\pi \rightarrow$ computação concorrente, comunicação como base
- Diferem em **sequência** (de avaliação de termos) e **convergência** (Church-Rosser)

Church-Rosser: a ordem em que reduções são aplicadas não afeta o resultado, i.e. se P pode reduzir para Q e R, então sempre haverá S tal que Q e R reduzem para S, fechando o diagrama.

Introdução

Pontos fortes:

- expressa comunicação entre processos
- síncrono, assíncrono e outros paradigmas

Pontos fracos:

- baixo nível
- modelagem de dados (mas é possível)

Sintaxe

Sintaxe

Composto por **agentes** e **nomes**, onde nomes são letras minúsculas e agentes são:

$$
\begin{array}{rcll}
P & ::= & M & \text{summation} \\
 & | & P_1 \mid P_2 & \text{composition} \\
 & | & (x)P & \text{restriction} \\
 & | & !P & \text{replication} \\
M & ::= & \mathbf{0} & \text{inaction} \\
 & | & \bar{y}x.P & \text{negative prefix} \\
 & | & y(x).P & \text{positive prefix} \\
 & | & \tau.P & \text{silent prefix} \\
 & | & [x = y]P & \text{match prefix} \\
 & | & M_1 + M_2 & \text{summation}
\end{array}
$$

[Sangiorgi e Walker, 2003]

Sintaxe: precedência

$$\left.\begin{array}{c} \textit{Restriction} \\ (x)P \\ \\ \textit{Prefix} \\ \bar{y}x.P,\ y(x).P,\ \tau.P \\ \\ \textit{Match} \\ [x=y]P \end{array}\right\} > \begin{array}{c} \textit{Composition} \\ P_1 \mid P_2 \end{array} > \begin{array}{c} \textit{Summation} \\ P_1 + P_2 \end{array}$$

[Milner et al, 1992]

Sintaxe: precedência

$$(x)\bar{y}x.P|y(x).Q + [x = y]R$$

$$\begin{tikzcd}
+ \arrow[d] \arrow[dr] & & \\
\mid \arrow[d] \arrow[dr] & {[x = y]} \arrow[dr] & \\
(x) \arrow[d] & y(x) \arrow[d] & R \\
\bar{y}x \arrow[d] & Q & \\
P & &
\end{tikzcd}$$

Sintaxe: exemplos

1. $\bar{y}x.\mathbf{0} \mid y(z).\mathbf{0} \rightarrow \ \mathbf{0} \mid \mathbf{0} \rightarrow \ \mathbf{0}$
2. $\bar{y}x.P \mid y(z).Q \rightarrow \ P \mid Q\{x/z\}$
3. $!\bar{y}x.P \mid y(z).Q \equiv\, !\bar{y}x.P \mid \bar{y}x.P \mid y(z).Q \rightarrow\, !\bar{y}x.P \mid P \mid Q\{x/z\}$

Sintaxe: nomes livres e ligados

$fn(P)$, são os nomes livres, isto é, que não são captados nem por entradas (positive prefix), nem por ligações (restriction). Já $n(P)$ são todos os nomes de P. Exemplo:

$$
\begin{aligned}
P &\stackrel{def}{=} (x)y(z).\mathbf{0}, & fn(P) &= \{y\} \\
Q &\stackrel{def}{=} \bar{r}s.\mathbf{0}, & fn(Q) &= \{r, s\} \\
R &\stackrel{def}{=} [t = u].\mathbf{0}, & fn(R) &= \{t, u\} \\
S &\stackrel{def}{=} (r)P \mid Q \mid R, & fn(S) &= fn(P) \cup fn(Q) \cup fn(R) - \{r\} \\
& & &= \{y, s, t, u\}
\end{aligned}
$$

Nomes entre parênteses são ligados. Outros nomes são livres.

Sintaxe: substituição

Se $y \notin n(P)$, então $P\{y/x\}$ é P, onde todas ocorrências livres de x são substituídas por y. Exemplo:

$$(Q \mid R)\{y/x\} = \bar{z}y.(Q'\{y/x\}) \mid (x)z(w).R'$$

$$Q \stackrel{def}{=} \bar{z}x.Q', \qquad fn(Q) = \{z, x\}$$
$$R \stackrel{def}{=} (x)z(w).R', \qquad fn(R) = \{z\}$$

Se $y \in n(P)$, então primeiro é necessário renomear as ocorrências ligadas de y por um nome novo, por exemplo, u. Exemplo:

$$T\{y/x\} = z(u).\bar{u}y.(T'\{u/y\})\{y/x\}$$

$$T \stackrel{def}{=} z(y).\bar{y}x.T', \quad fn(T) = \{z, x\}$$

Sintaxe: escopo

Intrusão de escopo é quando um nome livre invade o escopo de um nome ligado, exigindo substituições. Exemplo:

$$\bar{y}x.P \mid (x)(y(z).Q) \;\rightarrow\; P \mid (x')(Q\{x'/x\}\{x/z\})$$

Extrusão de escopo é quando um nome ligado estende seu escopo para um processo **que ainda não possui esse nome**. Exemplo:

$$(x)(\bar{y}x.P) \mid y(z).Q \;\rightarrow\; (x)(P \mid Q\{x/z\}),\; x \notin fn(Q)$$

Migração de escopo é uma extrusão em que o escopo diminui. Exemplo:

$$(x)(P \mid Q\{x/z\}) \;\rightarrow\; P \mid (x)(Q\{x/z\}),\; x \notin fn(P)$$

[Milner et al, 1992]

Sintaxe: conversão-α

- se $y \notin n(P)$, então $P\{y/x\}$ é P, onde ocorrências livres de x são substituídas por y
- se $y \notin n(P)$, então uma mudança de nomes ligados é
  - a troca de $z(x).P$ por $z(y).P\{y/x\}$
  - a troca de $(x)P$ por $(y)P\{y/x\}$
- Existe uma conversão-α entre P e Q, se houver uma sequência de mudanças de nomes ligados tal que $P = Q$

Exemplos:

$$z(x).P = z(y).P\{y/x\} = z(w).P\{w/x\} = ...$$

$$(x)z(w).P = (y)z(w).P\{y/x\} = (u)z(w).\{u/x\} = ...$$

$$z(y).\bar{y}x.P = z(u).\bar{u}x.P\{u/y\} = z(w).\bar{w}x.P\{w/y\} = ...$$

[Sangiorgi e Walker, 2003]

Sintaxe: convergência

$$Q_0 \stackrel{def}{=} (x)((x(y).x(z).\bar{y}z.\mathbf{0} \mid x(w).x(v).\bar{v}w.\mathbf{0}) \mid \bar{x}a.\bar{x}b.\mathbf{0})$$

$$Q_2 \stackrel{def}{=} (x)((\bar{x}b.\mathbf{0} \mid x(z).\bar{a}z.\mathbf{0}) \mid x(w).x(v).\bar{v}w.\mathbf{0})$$

$$Q_3 \stackrel{def}{=} (x)((\bar{x}b.\mathbf{0} \mid x(v).\bar{v}a.\mathbf{0}) \mid x(y).x(z).\bar{y}z.\mathbf{0})$$

$$Q_4 \stackrel{def}{=} (x)(x(z).\bar{a}z.\mathbf{0} \mid x(v).\bar{v}b.\mathbf{0}),\ Q_5 \stackrel{def}{=} (x)(\bar{a}b.\mathbf{0} \mid x(w).x(v).\bar{v}w.\mathbf{0})$$

$$Q_6 \stackrel{def}{=} (x)(\bar{b}a.\mathbf{0} \mid x(y).x(z).\bar{y}z.\mathbf{0}),\ Q_7 \stackrel{def}{=} (x)(x(v).\bar{v}a.\mathbf{0} \mid x(z).\bar{b}z.\mathbf{0})$$

$$Q_0 \xrightarrow{a/y} Q_2 \xrightarrow{b/w} Q_4,\ Q_0 \xrightarrow{a/y} Q_2 \xrightarrow{b/z} Q_5$$

$$Q_0 \xrightarrow{a/w} Q_3 \xrightarrow{b/v} Q_6,\ Q_0 \xrightarrow{a/w} Q_3 \xrightarrow{b/y} Q_7$$

$$\text{Onde } Q_4 \stackrel{\alpha}{\neq} Q_5 \stackrel{\alpha}{\neq} Q_6 \stackrel{\alpha}{\neq} Q_7$$

Sintaxe: convergência

$$Q_0 \overset{def}{=} (x)((x(y).x(z).\bar{y}z.\mathbf{0} \mid x(w).x(v).\bar{v}w.\mathbf{0}) \mid \bar{x}a.\bar{x}b.\mathbf{0})$$

$$Q_2 \overset{def}{=} (x)((\bar{x}b.\mathbf{0} \mid x(z).\bar{a}z.\mathbf{0}) \mid x(w).x(v).\bar{v}w.\mathbf{0})$$

$$Q_3 \overset{def}{=} (x)((\bar{x}b.\mathbf{0} \mid x(v).\bar{v}a.\mathbf{0}) \mid x(y).x(z).\bar{y}z.\mathbf{0})$$

$$Q_4 \overset{def}{=} (x)(x(z).\bar{a}z.\mathbf{0} \mid x(v).\bar{v}b.\mathbf{0}),\ Q_5 \overset{def}{=} (x)(\bar{a}b.\mathbf{0} \mid x(w).x(v).\bar{v}w.\mathbf{0})$$

$$Q_6 \overset{def}{=} (x)(\bar{b}a.\mathbf{0} \mid x(y).x(z).\bar{y}z.\mathbf{0}),\ Q_7 \overset{def}{=} (x)(x(v).\bar{v}a.\mathbf{0} \mid x(z).\bar{b}z.\mathbf{0})$$

$$Q_0 \overset{a/y}{\rightarrow} Q_2 \overset{b/w}{\rightarrow} Q_4,\ Q_0 \overset{a/y}{\rightarrow} Q_2 \overset{b/z}{\rightarrow} Q_5$$

$$Q_0 \overset{a/w}{\rightarrow} Q_3 \overset{b/v}{\rightarrow} Q_6,\ Q_0 \overset{a/w}{\rightarrow} Q_3 \overset{b/y}{\rightarrow} Q_7$$

$$\text{Onde } Q_4 \overset{\alpha}{\neq} Q_5 \overset{\alpha}{\neq} Q_6 \overset{\alpha}{\neq} Q_7$$

Sintaxe: convergência

Para garantir o envio ao mesmo processo de todos nomes, é necessário estabelecer um canal de comunicação:

$$(x)((x(u).u(y).u(z).\bar{y}z.\mathbf{0} \mid x(t).t(w).t(v).\bar{v}w.\mathbf{0}) \mid (s)\bar{x}s.\bar{s}a.\bar{s}b.\mathbf{0})$$

No exemplo há extrusão do escopo de s.

[Sangiorgi e Walker, 2003]

Semântica I: Redução

Redução

A redução é a evolução de P para P' através de uma **ação interna** a P.

$$P \rightarrow P'$$

A redução acontece por meio da aplicação de **regras inferência**. Para poder aplicar as regras, é necessário manipular os processos através de uma **relação de congruência estrutural**.

[Sangiorgi e Walker, 2003]

Redução: regras de inferência

$$\frac{}{(\bar{x}y.P_1 + M_1) \mid (x(z).P_2 + M_2) \to P_1 \mid P_2\{y/z\}}\ \text{R-INTER}$$

$$\frac{}{\tau.P + M \to P}\ \text{R-TAU}$$

$$\frac{P_1 \to P_1'}{P_1 \mid P_2 \to P_1' \mid P_2}\ \text{R-PAR} \qquad \frac{P \to P'}{(z)P \to (z)P'}\ \text{R-RES}$$

$$\frac{P_1 \equiv P_2 \to P_2' \equiv P_1'}{P_1 \to P_1'}\ \text{R-STRUCT}$$

Só quatro regras, é simples, exceto pela **relação de congruência** $\equiv$ [Sangiorgi e Walker, 2003]

Redução: relação de congruência estrutural

Congruentes são os processos que, inseridos no mesmo contexto, qualquer que seja este contexto, continuam congruentes.

Processos congruentes têm o mesmo comportamento potencial.

$$\begin{aligned} \text{reflexividade} &: P = P \\ \text{simetria} &: P = Q \text{ implica } Q = P \\ \text{transitividade} &: P = Q \text{ e } Q = R \text{ implica } P = R \\ \text{congruência} &: P = Q \text{ implica } C[P] = C[Q] \end{aligned}$$

Como determinar se $P = Q$?

[Sangiorgi e Walker, 2003]

Redução: relação de congruência estrutural

Axiomas de congruência estrutural:

$$
\begin{aligned}
[x = x]\pi.P &\equiv \pi.P && \text{(SC-MAT)} \\
M_1 + (M_2 + M_3) &\equiv (M_1 + M_2) + M_3 && \text{(SC-SUM-ASSOC)} \\
M_1 + M_2 &\equiv M_2 + M_1 && \text{(SC-SUM-COMM)} \\
M + \mathbf{0} &\equiv M && \text{(SC-SUM-INACT)} \\
P_1 \mid (P_2 \mid P_3) &\equiv (P_1 \mid P_2) \mid P_3 && \text{(SC-COMP-ASSOC)} \\
P_1 \mid P_2 &\equiv P_2 \mid P_1 && \text{(SC-COMP-COMM)} \\
P \mid \mathbf{0} &\equiv P && \text{(SC-COMP-INAC)} \\
(z)(w)P &\equiv (w)(z)P && \text{(SC-RES)} \\
(z)\mathbf{0} &\equiv \mathbf{0} && \text{(SC-RES-INACT)} \\
(z)(P_1 \mid P_2) &\equiv P_1 \mid (z)P_2,\ z \notin fn(P_1) && \text{(SC-RES-COMP)} \\
!P &\equiv P \mid !P && \text{(SC-REP)}
\end{aligned}
$$

[Sangiorgi e Walker, 2003]

Redução: exemplo

Dado que

$$P \stackrel{def}{=} !(y)Q \quad Q \stackrel{def}{=} \bar{x}y.\bar{y}y.\mathbf{0} \ + \ x(z).z(w).\mathbf{0}$$

$$R \stackrel{def}{=} (y)(\bar{y}y.\mathbf{0} \mid (v)y(w).\mathbf{0}) \mid P$$

Provaremos

$$P \rightarrow R$$

$$\frac{P \equiv (v)(y)(Q \mid Q\{v/y\} \mid P) \rightarrow (v)(y)(\bar{y}y.\mathbf{0} \mid y(w).\mathbf{0} \mid P) \equiv R}{P \rightarrow R} \ \textsc{r-struct}$$

Redução: exemplo

Lembrando que

$$Q \stackrel{def}{=} \bar{x}y.\bar{y}y.\mathbf{0} \ + \ x(z).z(w).\mathbf{0}$$

Provamos

$$(v)(y)(Q \mid Q\{v/y\} \mid P) \to (v)(y)(\bar{y}y.\mathbf{0} \mid y(w).\mathbf{0} \mid P)$$

$$\dfrac{\dfrac{\dfrac{\overline{Q \mid Q\{v/y\} \to \bar{y}y.\mathbf{0} \mid z(w).\mathbf{0}\{y/z\}}\ \text{R-INTER}}{Q \mid Q\{v/y\} \mid P \to \bar{y}y.\mathbf{0} \mid y(w).\mathbf{0} \mid P}\ \text{R-PAR}}{(y)(Q \mid Q\{v/y\} \mid P) \to (y)(\bar{y}y.\mathbf{0} \mid y(w).\mathbf{0} \mid P)}\ \text{R-RES}}{(v)(y)(Q \mid Q\{v/y\} \mid P) \to (v)(y)(\bar{y}y.\mathbf{0} \mid y(w).\mathbf{0} \mid P)}\ \text{R-RES}$$

Redução: exemplo

Lembrando que

$$Q \overset{def}{=} \bar{x}y.\bar{y}y.\mathbf{0} \ + \ x(z).z(w).\mathbf{0}$$

$$\begin{aligned}
P &= !(y)Q \\
&\equiv (y)Q \mid !(y)Q \\
&\equiv (y)Q \mid (y)Q \mid !(y)Q \\
&\overset{\alpha}{\equiv} (y)Q \mid (v)Q\{v/y\} \mid !(y)Q, \ v \notin n(Q) \\
&\equiv (y)Q \mid (v)Q\{v/y\} \mid P \\
&\equiv (v)((y)Q \mid Q\{v/y\} \mid P), \ v \notin fn((y)Q \mid P) \\
&\qquad (y)Q \mid Q\{v/y\} \mid P \equiv (y)(Q \mid Q\{v/y\} \mid P), \\
&\qquad y \notin fn(Q\{v/y\} \mid P) \\
&\qquad C = (v)[.] \\
&\equiv (v)(y)(Q \mid Q\{v/y\} \mid P)
\end{aligned}$$

Redução: exemplo

$$
\begin{aligned}
(v)(y)(\bar{y}y.\mathbf{0} \mid y(w).\mathbf{0} \mid P) \equiv & \\
& \equiv (y)(v)(\bar{y}y.\mathbf{0} \mid y(w).\mathbf{0} \mid P) \\
& \equiv (y)(\bar{y}y.\mathbf{0} \mid (v)y(w).\mathbf{0} \mid P), v \notin fn(\bar{y}y.\mathbf{0}, P) \\
& \equiv (y)(\bar{y}y.\mathbf{0} \mid (v)y(w).\mathbf{0}) \mid P, y \notin fn(P) \\
& \equiv R
\end{aligned}
$$

Redução: derivação normalizada

Teorema

Se $P \rightarrow Q$, então existe uma derivação que inicia em R-INTER ou R-TAU, seguida de R-PAR, seguida de n aplicações de R-RES e finalizando com uma aplicação de R-STRUCT.

Semântica II: Sistemas de transições rotuladas (STR)

STR: Sistema de transições rotuladas

A **redução** descreve uma ação interna ao processo, mas de que forma o sistema interage com o ambiente em que está inserido?

O **sistema de transições rotuladas** (STR) quebra os passos da redução em mais regras, permitindo análise do processo de forma fragmentada.

STR: ações/rótulos

α	tipo
$\bar{x}y$	saída livre
xy	entrada
$\bar{x}(z)$	saída ligada
τ	ação interna

STR: transições

$$\frac{}{\bar{x}y.P \xrightarrow{\bar{x}y} P}\ \mathrm{OUT} \qquad \frac{}{x(z).P \xrightarrow{xy} P\{y/z\}}\ \mathrm{INP}$$

$$\frac{}{\tau.P \xrightarrow{\tau} P}\ \mathrm{TAU} \qquad \frac{\pi.P \xrightarrow{\alpha} P'}{[x = x]\pi.P \xrightarrow{\alpha} P'}\ \mathrm{MAT}$$

$$\frac{P \xrightarrow{\bar{x}y} P' \quad Q \xrightarrow{xy} Q'}{P \mid Q \xrightarrow{\tau} P' \mid Q'}\ \mathrm{COMM\text{-}L}$$

[Sangiorgi e Walker, 2003]

STR: transições (cont. I)

$$\frac{P \xrightarrow{\alpha} P'}{P \mid Q \xrightarrow{\alpha} P' \mid Q} \ \mathrm{PAR\text{-}L}, \ bn(\alpha) \cap fn(Q) = \emptyset$$

$$\frac{P \xrightarrow{\alpha} P'}{(z)P \xrightarrow{\alpha} (z)P'} \ \mathrm{RES}, \ z \notin n(\alpha) \qquad \frac{P \xrightarrow{\alpha} P'}{P + Q \xrightarrow{\alpha} P'} \ \mathrm{SUM\text{-}L}$$

$$\frac{P \xrightarrow{\bar{x}z} P'}{(z)P \xrightarrow{\bar{x}(z)} P'} \ \mathrm{OPEN}, \ z \neq x$$

$$\frac{P \xrightarrow{\bar{x}(z)} P' \quad Q \xrightarrow{xz} Q'}{P \mid Q \xrightarrow{\tau} (z)(P' \mid Q')} \ \mathrm{CLOSE\text{-}L}, \ z \notin fn(Q)$$

[Sangiorgi e Walker, 2003]

STR: transições (cont. II)

$$\frac{!P \overset{\alpha}{\rightarrow} P' \mid !P}{P \overset{\alpha}{\rightarrow} P'} \text{ REP-ACT}$$

$$\frac{P \overset{\bar{x}y}{\rightarrow} P' \quad P \overset{xy}{\rightarrow} P''}{!P \overset{\tau}{\rightarrow} (P' \mid P'') \mid !P} \text{ REP-COMM}$$

$$\frac{P \overset{\bar{x}(z)}{\rightarrow} P' \quad P \overset{xz}{\rightarrow} P''}{!P \overset{\tau}{\rightarrow} ((z)(P' \mid P'')) \mid !P} \text{ REP-CLOSE}, \; z \notin fn(P)$$

[Sangiorgi e Walker, 2003]

STR: exemplo

Dado que

$$P \stackrel{def}{=} !(y)Q \qquad Q \stackrel{def}{=} \bar{x}y.\bar{y}y.\mathbf{0} \;+\; x(z).z(w).\mathbf{0}$$

$$R \stackrel{def}{=} (y)(\bar{y}y.\mathbf{0} \mid (v)y(w).\mathbf{0}) \mid P$$

Provamos

$$P \xrightarrow{\tau} R$$

$$\dfrac{\dfrac{\dfrac{\dfrac{}{\bar{x}y.\bar{y}y.\mathbf{0} \xrightarrow{\bar{x}y} \bar{y}y.\mathbf{0}}\;\textsc{out}}{Q \xrightarrow{\bar{x}y} \bar{y}y.\mathbf{0}}\;\textsc{sum}}{(y)Q \xrightarrow{\bar{x}(y)} \bar{y}y.\mathbf{0}}\;\textsc{open},\; y \neq x \qquad \dfrac{\dfrac{\dfrac{\dfrac{}{x(z).z(w).\mathbf{0} \xrightarrow{xy} z(w).\mathbf{0}\{y/z\}}\;\textsc{inp}}{Q \xrightarrow{xy} y(w).\mathbf{0}}\;\textsc{sum}}{(v)Q \xrightarrow{xy} (v)y(w).\mathbf{0}}\;\textsc{res},\; v \notin n(xy)}{(y)Q\{y/v\} \xrightarrow{xy} (v)y(w).\mathbf{0}}\;(y)Q = (v)Q}{!(y)Q \xrightarrow{\tau} (y)(\bar{y}y.\mathbf{0} \mid (v)y(w).\mathbf{0}) \mid !(y)Q}\; y \notin fn(P)$$

STR

Lema (Lema da Harmonia)

$$P \equiv\xrightarrow{\alpha} P' \ implica \ P \xrightarrow{\alpha}\equiv P' \tag{1}$$

$$P \rightarrow P' \ sse \ P \xrightarrow{\tau}\equiv P' \tag{2}$$

Extensões

Cálculo poliádico

No cálculo poliádico, uma comunicação pode enviar mais de um nome:

$$\frac{}{(\bar{x}(\widetilde{y}).P_1 + M_1) \mid (x(\widetilde{z}).P_2 + M_2) \rightarrow P_1 \mid P_2\{\widetilde{y}/\widetilde{z}\}} \text{ R-INTER}, |\widetilde{y}| = |\widetilde{z}|$$

Cálculo assíncrono

O cálculo assíncrono é um subcálculo do cálculo π, onde envio de nomes não pode ter guardas, i.e. $\bar{x}y$ só pode ocorrer em $\bar{x}y.\mathbf{0}$, nunca em $\bar{x}y.P$ ou $\bar{x}y.\mathbf{0} \ + \ Q$:

$$\bar{x}y.\mathbf{0} \mid (x(z).P \ + \ M) \ \rightarrow \ P\{y/z\}$$

Cálculo tipado

Por que tipos? Encontrar erros por análise estática.

- Adiciona wrong aos elementos sintáticos
- Define valores como valores básicos (ints, bools, etc.) e nomes
- Define erros como termos em que valores básicos tomam o lugar de um canal, e.g. $\overline{1}x$ ou $1(x)$
- Cria um conjunto de regras de inferência de tipos
- Modifica a semântica do cálculo π de modo que todas regras respeitem a tipagem

Cálculo de alta ordem

No cálculo de alta ordem, processos podem ser enviados através de canais.

- Adiciona abstrações aos valores, e.g. $(x).P$
- Adiciona aplicações aos processos, e.g. $x\lfloor y \rfloor$
- Permite passagem de abstrações através de canais, e.g. $\bar{x}((w)P).Q \mid x(y).y\lfloor z \rfloor \xrightarrow{\tau} Q \mid P\{z/w\}$
- Cria regra de inferência para aplicação:

$$\frac{}{((x).P)\lfloor v \rfloor \xrightarrow{\tau} P\{v/x\}}\ \mathrm{APP}$$

Demo

Demo

GitHub: Diogo

Aplicações

Aplicações: SPI

Motivação: cálculo para verificar protocolos criptográficos (simétricos e assimétricos)

Cálculo pi já oferece canais de comunicação privados (x).

Cálculo spi adiciona operações criptográficas:

- Encriptar mensagem M com chave N: $\{M\}_N$
- Decriptar mensagem L com chave N, seguindo com $P\{M/x\}$: $\mathrm{case}\ L\ \mathrm{of}\ \{x\}_N\ \mathrm{in}\ P$

[Abadi e Gordon, 1999]

Aplicações: SPI (cont.)

Propriedades de protocolos como **sigilo** (M não é lida em trânsito) e **integridade** (adversário não consegue substituir M por outra mensagem) verificadas através de relação de **equivalência**.

[Abadi e Gordon, 1999]

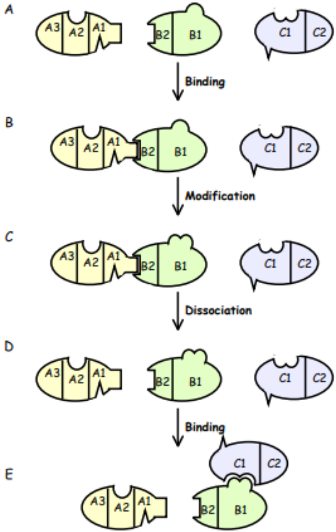
Aplicações: processos moleculares

Via metabólica: sequência de reações químicas que acontecem nas células. Exemplo: glicólise, a sequência de reações que transforma glicose em ATP.

Cálculo π define verificação de especificações quanto ao comportamento, o que permite comparação de processos moleculares quanto à homologia.

Homologia: estudo de estruturas presentes em diferentes espécies originadas de um ancestral comum.

Aplicações: processos moleculares (cont.)



[Regev e Shapiro, 2004]

Aplicações: processos moleculares (cont.)

PiFCP implementa o cálculo π em FCP (Flat Concurrent Prolog), e permite uma análise qualitativa dos processos.

[Regev et al, 2001]

PsiFCP utiliza uma extensão de cálculo π, o cálculo estocástico, em que probabilidades são atribuídas a prefixos, o que permite análises quantitativas (tempo, performance, probabilidade).

[Priami et al, 2001]

Referências

Robin Milner, Joachim Parrow e David Walker (1992)
A Calculus of Mobile Processes, Part I

Robin Milner, Joachim Parrow e David Walker (1992)
A Calculus of Mobile Processes, Part II

Abadi e Gordon (1999)
A Calculus for Cryptographic Protocols: The Spi Calculus

Corrado Priami, Aviv Regev, Ehud Shapiro e William Silverman (2001)
Application of stochastic name-passing calculus to representation and simulation of molecular processes

Aviv Regev e Ehud Shapiro (2004)
The π calculus as an Abstraction for Biomolecular Systems

Aviv Regev, William Silverman e Ehud Shapiro (2001)
Representation and simulation of biochemical processes using the π-calculus process algebra

Referências (cont.)

Davide Sangiorgi e David Walker (2003)
The π-calculus, A Theory of Mobile Processes

Applied Category Theory (2019)
UCR Applied Category Theory Seminar: The Pi Calculus. Link: https://www.youtube.com/watch?v=NTJBMbTIJis

Obrigado!

π

Diogo Raphael Cravo (diogo.raphael.cravo@gmail.com)

Junho de 2019