Aula 4

CCS:Calculus of Communicating Systems

- o processo mais simples é o que não executa nenhuma ação(deadlock): 0
- Se P é um processo e a uma ação $a.P$ é um processo: que executa a e depois comporta-se como P
- Um fósforo: $strike.light.extinguish.0$
- Uma máquina de vender café: $coin.coffee.0$
- Ações internas: τ
- $strike.\tau.0$
- Se houver uma escolha $strike.(light.0 + \tau.0)$
- Dois fósforos: $strike.(light.0|\tau.0)$

Exemplo de $1 - Buffer$

$$Buffer \quad := \quad put?.get?.Buffer$$

Calcula $[\![Buffer]\!]_\Gamma$.

$$BufferM \quad := \quad put?.get?.BufferM + get?.put?.BufferM$$

Calcula $[\![put?.BufferM]\!]_\Gamma$.

$$\begin{aligned} Buffer0 &:= put?.Buffer1 \\ Buffer1 &:= get?.Buffer2 + get?.Buffer0 \\ Buffer2 &:= get?.Buffer1 \end{aligned}$$

Calcula $[\![Buffer0]\!]_\Gamma$.

Operadores do CCS

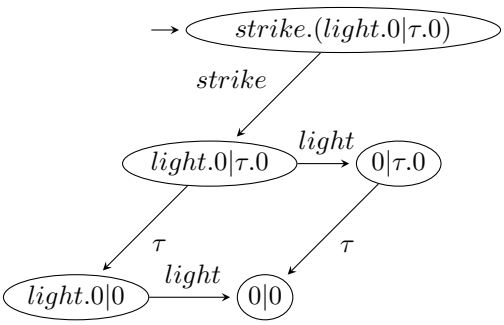
"." **"Prefixo** a execução de $\alpha.P$ começa com a execução da ação $\alpha \in Act$ e depois comporta-se como P

"+ "**Escolha** O processo $P+Q$ comporta-se como o processo P ou o processo Q. É a escolha não determinística

"| "**Composição Paralela** O processo $P|Q$ representa a execução concorrente de P e Q (que progridem independentemente no tempo).

Paralelismo

Dois fósforos: $strike.(light.0|\tau.0)$



As ações são instantaneas

Regras de inferência Par

$$\text{ParE}\ \frac{P \xrightarrow{\alpha} P'}{P|Q \xrightarrow{\alpha} P'|Q}$$

$$\text{ParD}\ \frac{Q \xrightarrow{\alpha} Q'}{P|Q \xrightarrow{\alpha} P|Q'}$$

Sincronia em CCS

- O conjunto de ações observáveis Com é divido em dois
- $Com = A^! \cup A^?$
- $A^!$ conjunto de ações de saída (*envio*)
- $A^?$ conjunto de ações de entrada (*recebidas*)
- Ações com o mesmo nome formam um par e são complementares:
- um processo envia $a^!$ e e outro recebe $a^?$
- O complemento de $a \in A^! \cup A^?$ designa-se por $\overline{a}$: se $a \in A^!$ então $\overline{a} \in A^?$ e vice-versa.
- Para a ação interna τ, temos $\tau = \overline{\tau}$.
- $\forall \alpha \in Act, \overline{\overline{\alpha}} = \alpha$.

Regra de inferência de Sincronia

$$\frac{P \xrightarrow{a} P' \qquad Q \xrightarrow{\overline{a}} Q'}{P|Q \xrightarrow{\cdots} P'|Q'}$$

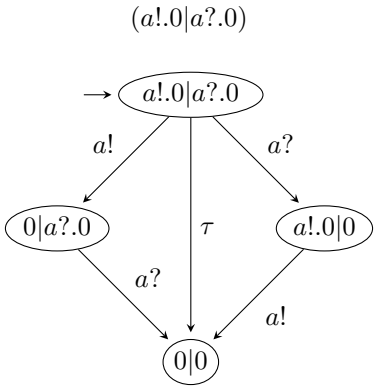
A sincronização não é observável por processos externos A ação interna τ representa a sincronização para o exterior

$$\text{Sync}\ \frac{P \xrightarrow{a} P' \qquad Q \xrightarrow{\overline{a}} Q'}{P|Q \xrightarrow{\tau} P'|Q}$$

$$\text{ParE}\ \frac{P \xrightarrow{\alpha} P'}{P|Q \xrightarrow{\alpha} P'|Q}$$

$$\text{ParD}\ \frac{Q \xrightarrow{\alpha} Q'}{P|Q \xrightarrow{\alpha} P|Q'}$$

Exemplo



A regra Sync não exclui a utilização de ParD e ParE

Operador de Restrição

- Proibe que pares de ações observáveis (a e $\overline{a}$) sejam usadas (individualmente)
- Força a sincronia

- $P\backslash H$ onde P é um processo e H um conjunto de ações de comunicação que serão proibidas
- As ações internas τ não podem estar em H

$$\text{Res}\ \frac{P \xrightarrow{\alpha} P' \qquad \alpha \notin H}{P\backslash H \xrightarrow{\alpha} P'\backslash H}$$

Exemplo de Restrição

$$((a!.0|a!.0)|a?.0)\backslash\{a!, a?\} \xrightarrow{\tau} ((a!.0|0)|0)\backslash\{a!, a?\}$$

$$((a!.0|a!.0)|a?.0)\backslash\{a!, a?\} \xrightarrow{\tau} ((0|a!.0)|0)\backslash\{a!, a?\}$$

$$\text{ParE}\ \frac{P \xrightarrow{\alpha} P'}{P|Q \xrightarrow{\alpha} P'|Q}$$

$$\text{Res}\ \frac{P \xrightarrow{\alpha} P' \qquad \alpha \notin H}{P\backslash H \xrightarrow{\alpha} P'\backslash H}$$

$$\text{ParD}\ \frac{Q \xrightarrow{\alpha} Q'}{P|Q \xrightarrow{\alpha} P|Q'}$$

In PseuCo

$$\begin{aligned} Match &:= strike.MatchOnFire \\ MatchOnFire &:= light!.MatchOnFire + extinguish!.0 \\ TwoFireCracker &:= light?.(bang!.0|bang!.0) \end{aligned}$$

$$(Match|TwoFireCracker)\backslash\{light\}$$

CCS (quase) completo

Seja $Com = A^! \cup A^?$ conjunto de ações de comunicação, $Act = Com \cup \{\tau\}$ um conjunto de ações, e Var um conjunto de nomes (variáveis). As expressões do CCS são

$$P \ ::= \ 0 \mid X \mid P + P \mid \alpha.P \mid P|P \mid P\backslash H$$

onde $\alpha \in Act$, $X \in Var$ e $H \subseteq Com$. Supomos um conjunto Γ de equações $X := P$.

Semântica do CCS

A semântica das expressões do CCS é então

$$[\![\_]\!] : (Var \to CCS) \to CCS \to LTS_{CCS}$$

tal que

$$[\![P]\!]_\Gamma = (CCS, \longrightarrow_\Gamma, P)$$

com

$$LTS_{CCS} = \{(CCS, T, s) \mid T \subseteq CCS \times Act \times CCS, \wedge s \in CCS\}$$

onde $\longrightarrow_\Gamma$ a mais pequena relação que satisfaz as regras de inferência.

$$\text{Pref}\ \frac{}{\alpha.P \xrightarrow{\alpha} P}$$

$$\text{ParE}\ \frac{P \xrightarrow{\alpha} P'}{P|Q \xrightarrow{\alpha} P'|Q}$$

$$\text{EscD}\ \frac{Q \xrightarrow{\alpha} Q'}{P + Q \xrightarrow{\alpha} Q'}$$

$$\text{EscE}\ \frac{P \xrightarrow{\alpha} P'}{P + Q \xrightarrow{\alpha} P'}$$

$$\text{Sync}\ \frac{P \xrightarrow{a} P' \qquad Q \xrightarrow{\overline{a}} Q'}{P|Q \xrightarrow{\tau} P'|Q'}$$

$$\text{ParD}\ \frac{Q \xrightarrow{\alpha} Q'}{P|Q \xrightarrow{\alpha} P|Q'}$$

$$\text{Res}\ \frac{P \xrightarrow{\alpha} P' \qquad \alpha \notin H}{P\backslash H \xrightarrow{\alpha} P'\backslash H}$$

$$\text{Rec}\ \frac{P \xrightarrow{\alpha} P' \qquad \Gamma(X) = P}{X \xrightarrow{\alpha} P'}$$

Mais regras de prioridade

- $P|Q|R$ é $(P|Q)|R$
- $\alpha.P|Q$ é $(\alpha.P)|Q$
- $P + R|Q$ é $(P + R)|Q$

Expressividade do CCS

- CCS_0 só pode produzir sistemas de transição finitos acíclicos
- CCS_0^ω com Γ finito só pode produzir sistemas de transição estado-finitos (a menos de isomorfismo)
- CCS pode produzir sistemas e transição infinitos e com ramificação infinita
- Ex: $X := a.0|X$ tem ramificação infinita
- com mais uma operação (a que falta)- renomeação $P[f]$ tem a potência duma Máquina de Turing.

Operador de renomeação

- Já vimos uma máquina de café em CCS
- $CM := coin?.coffee!.CM$
- Mas agora se quisermos um chocolate?
- Seria igual apenas mudando a accão de envio.
- $ChM := coin?.choco!.ChM$
- e o mesmo para outros produtos.
- Então podemos ter
- $VM := coin?.item!.VM$
- e definir

$$\begin{array}{rcl} CM & := & VM[coffee/item] \\ ChM & := & VM[choc/item] \\ \ldots & & \end{array}$$

Operador de renomeação

Seja $f : Act \to Act$ uma função de renomeação tal que

$$\begin{array}{rcl} f(\tau) & := & \tau, \\ f(\overline{a}) & := & \overline{f(a)} \ \ \forall a \in Com. \end{array}$$

A função f pode representar-se da forma $[b_1/a_1, \ldots b_n/a_n]$ se $f(a_i) = b_i$.

Então, sendo P um processo $P[f]$ é também um processo em que cada ação a é substituida por $f(a)$ (assim como os complementos).

$$(a.B + b.B)[a/b, b/a]$$
$$(a.0 + \overline{a}.A)[a/b]$$

Regra de inferência para $P[f]$

$$\text{Rel}\ \frac{P \xrightarrow{\alpha} P'}{P[f] \xrightarrow{f(\alpha)} P'[f]}$$

Exemplo 4.1. *Seja $A := a.b.B$, calcular*

$$[\![(A|b?.a.B) + (b?.A)[a/b]]\!]_\Gamma$$

Operadores Estáticos e Dinâmicos

- *Operadores dinâmicos*: $\cdot$ e $+$
- Desaparecem após se efectuar uma transição

$$\text{Pref}\ \frac{}{\alpha.P \xrightarrow{\alpha} P}$$

$$\text{EscD}\ \frac{Q \xrightarrow{\alpha} Q'}{P + Q \xrightarrow{\alpha} Q'}$$

- *Operadores estáticos*: $|$ e $\setminus$ (também $[f]$)
- Não desaparecem após ser efectuada uma transição

$$\text{Sync}\ \frac{P \xrightarrow{a} P' \qquad Q \xrightarrow{\overline{a}} Q'}{P|Q \xrightarrow{\tau} P'|Q'}$$

$$\text{ParD}\ \frac{Q \xrightarrow{\alpha} Q'}{P|Q \xrightarrow{\alpha} P|Q'}$$

$$\text{Res}\ \frac{P \xrightarrow{\alpha} P' \qquad \alpha \notin H}{P\setminus H \xrightarrow{\alpha} P'\setminus H}$$

CCS Regular

Não é permitida recursão sobre operadores estáticos.

$$\begin{aligned} P &::= 0 \mid X \mid P+P \mid \alpha.P \mid R \\ R &::= 0 \mid R+R \mid \alpha.R \mid R|R \mid R\backslash H \end{aligned}$$

Proposição 4.1. *Se Γ é uma função parcial cujo contradomínio só tem expressões regulares então o Reach($[\![P]\!]_\Gamma$) é estados finito para todo $P \in CCS$ (i.e. o LTS atingível é estados-finito).*

Este é o cálculo que se usa em geral na prática.

Buffer Paralelos

- Calcular $[\![Buffer|Buffer]\!]_\Gamma$. para

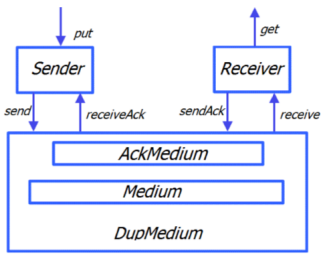
$$Buffer := put?.get?.Buffer$$

-

$$\begin{aligned} BufferL &:= put?.pass!.BufferL \\ BufferR &:= pass?.get?.BufferR \end{aligned}$$

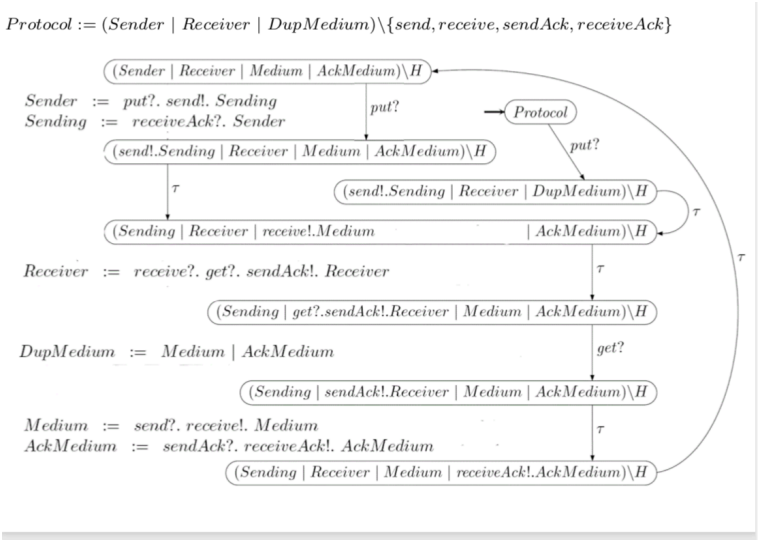
Calcular $[\![(BufferL|BufferR)\backslash\{pass!, pass?\}]\!]$

Exemplo de um Protocolo com Ack (Pseuco)



$$\begin{aligned} Sender &:= put?.send!.Sending \\ Sending &:= receiveAck?.Sender \\ Receiver &:= receive?.get?.sendAck!.Receiver \\ Medium &:= send?.receive!.Medium \\ AckMedium &:= sendAck?.receiveAck!.AckMedium \\ DupMedium &:= Medium|AckMedium \\ ProtocolG &:= (Sender \mid Receiver \mid DupMedium)\backslash \\ &\quad \{send, receive, sendAck, receiveAck\} \end{aligned}$$

Exemplo de um Protocolo com Ack (Pseuco, Holger H.)

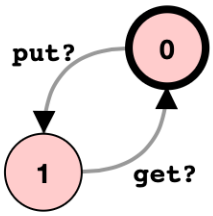


Exemplo de um Protocolo com erro no meio –Pseuco

$$
\begin{aligned}
Sender &:= put?.send!.Sending \\
Sending &:= receiveAck?.Sender + receiveNAck?.send!.Sending \\
Receiver &:= receive?.get?.sendAck!.Receiver + \\
&\quad gargled?.sendNAck!.Receiver \\
Medium &:= send?.(receive!.Medium + i.garbled!.Medium) \\
AckMedium &:= sendAck?.receiveAck!.AckMedium + \\
&\quad sendNAck?.receivedNAck!.AckMedium \\
DupMedium &:= Medium|AckMedium \\
Protocol &:= (Sender \mid Receiver \mid DupMedium)\backslash \\
&\quad \{send, receive, sendAck, receiveAck, \\
&\quad receiveNAck, sendNAck, garbled\}
\end{aligned}
$$

Comportamento Externo

Para um obesrvador externo os processos $Buffer$, $Protocol$ e $ProtocolG$ têm o mesmo comportamento (mas LTSs diferentes, claro!).



CCS em PSeuco -ver exemplos

If `P` and `Q` are valid CCS processes, then the following are valid CCS processes as well:

Process	Semantics
`0`	cannot perform any action
`1`	terminated process – emits δ, behaves like `0` afterwards
`a.P`	performs action `a`, behaves like `P` afterwards
`P + Q`	non-deterministic choice between `P` and `Q`
`P | Q`	concurrent execution of `P` and `Q`
`P \ {a,b,c}`	behaves like `P` except that actions `a`, `b`, `c` are **not** allowed
`P \ {*,a,b,c}`	behaves like `P` except that **only** actions `a`, `b`, `c` are allowed
`P; Q`	behaves like `P` until it terminates, then performs a τ-transition to `Q`
`X`	behaves like `P` if `X` was defined before as `X := P`

A CCS action is any combination of letters starting with a lower-case letter. Some of them have a special meaning:

Input Sequence	Action	Meaning
`i`	τ (tau)	represents in internal, non-observable action
`e`	δ (delta)	signals that a process has terminated

CCS actions may contain a `!` or `?`. If both counterparts can be executed at the same time, they can synchronize, resulting in an internal τ transition. After `!`, a sending expression can be given, the value of which will be bound to the input variable given after `?`. Input variables may be bound to a range, defined as `range R := 0..0`, with `:R`. Ranges of string length can be specified, too: `$..$$$` allows strings of length 1 to 3.

A process name is any combination of letters starting with an upper-case letter.

Process definitions may contain an arbitrary number of variables in square brackets, which must be given when a process is called.