

看看程式語言學在幹嘛

柯向上 Josh Ko

中央研究院資訊科學研究所

FLOLAC 識別設計溝通會議，2020 年 3 月 10 日

程式語言學

以**符號**（語言）精準簡練地表達**計算規則**（程式）

厲害：嚴格證明各種「好」性質

易用：作為好工具，協助人思考與解決問題

內涵：常與數學、邏輯學有密切對應

厲害

規則、符號、好性質

拍賣網站購物

「先詢問有無存貨，賣家回覆有貨才可下標。下標後將收件資訊告知賣家，完成匯款後賣家出貨。」

買家動作：

```
    ! [請問還有沒有貨]
    ▶ { 缺貨 : stop
        有貨 : if 老婆說可以買
                then ◁ 下標
                    ! [收件資訊]
                    ! [錢]
                    ? (貨)
                    stop
                else ◁ 算了
                    stop }

```

! 送
? 收
▶ 對方選擇
◁ 自己選擇

拍賣網站購物

「先詢問有無存貨，賣家回覆有貨才可下標。下標後將收件資訊告知賣家，完成匯款後賣家出貨。」

賣家動作：

```
?(請問還有沒有貨)
if 查了一下有貨
then ◁ 有貨
    ▷{ 算了 : stop
        下標 : ?(收件資訊)
            ?(錢)
            ![貨]
            stop    }
else ◁ 缺貨
    stop
```

! 送
? 收
▷ 對方選擇
◁ 自己選擇

程式的合法長相 (語言)

P	$::=$	$\text{request } a(k) \text{ in } P$	session request
		$\text{accept } a(k) \text{ in } P$	session acceptance
		$k![\tilde{e}]; P$	data sending
		$k?(\tilde{x}) \text{ in } P$	data reception
		$k \triangleleft l; P$	label selection
		$k \triangleright \{l_1 : P_1 \parallel \cdots \parallel l_n : P_n\}$	label branching
		$\text{throw } k[k']; P$	channel sending
		$\text{catch } k(k') \text{ in } P$	channel reception
		$\text{if } e \text{ then } P \text{ else } Q$	conditional branch
		$P \mid Q$	parallel composition
		inact	inaction
		$(\nu u)P$	name/channel hiding
		$\text{def } D \text{ in } P$	recursion
		$X[\tilde{e}\tilde{k}]$	process variables
D	$::=$	$X_1(\tilde{x}_1\tilde{k}_1) = P_1 \text{ and } \cdots \text{ and } X_n(\tilde{x}_n\tilde{k}_n) = P_n$	declaration for recursion

實際進行購物

![請問還有沒有貨]

▷ { 缺貨 : stop

有貨 : if 老婆說可以買

then ◁ 下標

![收件資訊]

![錢]

?(貨)

stop

else ◁ 算了

stop }

買家

?(請問還有沒有貨)

if 查了一下有貨

then ◁ 有貨

▷ { 算了 : stop

下標 : ?(收件資訊)

?(錢)

![貨]

stop }

else ◁ 缺貨

stop

賣家

實際進行購物

![請問還有沒有貨]

```
▷ { 缺貨 : stop
    有貨 : if 老婆說可以買
            then ◁ 下標
                ![收件資訊]
                ![錢]
                ?(貨)
                stop
            else ◁ 算了
                stop }
```

買家

?(請問還有沒有貨)

```
if 查了一下有貨
then ◁ 有貨
    ▷ { 算了 : stop
        下標 : ?(收件資訊)
                ?(錢)
                ![貨]
                stop }
    else ◁ 缺貨
        stop
```

賣家

實際進行購物

![請問還有沒有貨]

```
▷ { 缺貨 : stop
    有貨 : if 老婆說可以買
            then ◁ 下標
                ![收件資訊]
                ![錢]
                ?(貨)
                stop
            else ◁ 算了
                stop }
```

買家

?(請問還有沒有貨)

```
if 查了一下有貨
then ◁ 有貨
    ▷ { 算了 : stop
        下標 : ?(收件資訊)
              ?(錢)
              ![貨]
              stop }
    else ◁ 缺貨
        stop
```

賣家

實際進行購物

![請問還有沒有貨]

▷ { 缺貨 : stop

有貨 : if 老婆說可以買

then ◁ 下標

![收件資訊]

![錢]

?(貨)

stop

else ◁ 算了

stop }

買家

?(請問還有沒有貨)

if 查了一下有貨

then ◁ 有貨

▷ { 算了 : stop

下標 : ?(收件資訊)

?(錢)

![貨]

stop }

else ◁ 缺貨

stop

賣家

實際進行購物

![請問還有沒有貨]

▷ { 缺貨 : stop

有貨 : if 老婆說可以買

then ◁ 下標

![收件資訊]

![錢]

?(貨)

stop

else ◁ 算了

stop }

買家

?(請問還有沒有貨)

if 查了一下有貨

then ◁ 有貨

▷ { 算了 : stop

下標 : ?(收件資訊)

?(錢)

![貨]

stop }

else ◁ 缺貨

stop

賣家

實際進行購物

![請問還有沒有貨]

▷ { 缺貨 : stop

有貨 : if 老婆說可以買

then ◁ 下標

![收件資訊]

![錢]

?(貨)

stop

else ◁ 算了
stop }

買家

?(請問還有沒有貨)

if 查了一下有貨

then ◁ 有貨

▷ { 算了 : stop

下標 : ?(收件資訊)

?(錢)

![貨]

stop }

else ◁ 缺貨
stop

賣家

實際進行購物

![請問還有沒有貨]

▷ { 缺貨 : stop

有貨 : if 老婆說可以買

then ◁ 下標

![收件資訊]

![錢]

?(貨)

stop

else ◁ 算了
stop }

買家

?(請問還有沒有貨)

if 查了一下有貨

then ◁ 有貨

▷ { 算了 : stop

下標 : ?(收件資訊)

?(錢)

![貨]

stop }

else ◁ 缺貨
stop

賣家

實際進行購物

![請問還有沒有貨]

▷ { 缺貨 : stop

有貨 : if 老婆說可以買

then ◁ 下標

![收件資訊]

![錢]

?(貨)

stop

else ◁ 算了
stop }

買家

?(請問還有沒有貨)

if 查了一下有貨

then ◁ 有貨

▷ { 算了 : stop

下標 : ?(收件資訊)

?(錢)

![貨]

stop }

else ◁ 缺貨
stop

賣家

實際進行購物

![請問還有沒有貨]

▷ { 缺貨 : stop

有貨 : if 老婆說可以買

then ◁ 下標

![收件資訊]

![錢]

?(貨)

stop

else ◁ 算了

stop }

買家

?(請問還有沒有貨)

if 查了一下有貨

then ◁ 有貨

▷ { 算了 : stop

下標 : ?(收件資訊)

?(錢)

![貨]

stop }

else ◁ 缺貨

stop

賣家

實際進行購物

![請問還有沒有貨]

▷ { 缺貨 : stop

有貨 : if 老婆說可以買

then ◁ 下標

![收件資訊]

![錢]

?(貨)

stop

else ◁ 算了
stop }

買家

?(請問還有沒有貨)

if 查了一下有貨

then ◁ 有貨

▷ { 算了 : stop

下標 : ?(收件資訊)

?(錢)

![貨]

stop }

else ◁ 缺貨
stop

賣家

定義程式如何執行

[LINK]	$(\text{accept } a(k) \text{ in } P_1) \mid (\text{request } a(k) \text{ in } P_2) \rightarrow (\nu k)(P_1 \mid P_2)$
[COM]	$(k![\tilde{e}]; P_1) \mid (k?(\tilde{x}) \text{ in } P_2) \rightarrow P_1 \mid P_2[\tilde{c}/\tilde{x}] \quad (\tilde{e} \downarrow \tilde{c})$
[LABEL]	$(k \triangleleft l_i; P) \mid (k \triangleright \{l_1 : P_1 \parallel \cdots \parallel l_n : P_n\}) \rightarrow P \mid P_i \quad (1 \leq i \leq n)$
[PASS]	$(\text{throw } k[k']; P_1) \mid (\text{catch } k(k') \text{ in } P_2) \rightarrow P_1 \mid P_2$
[IF1]	$\text{if } e \text{ then } P_1 \text{ else } P_2 \rightarrow P_1 \quad (e \downarrow \text{true})$
[IF2]	$\text{if } e \text{ then } P_1 \text{ else } P_2 \rightarrow P_2 \quad (e \downarrow \text{false})$
[DEF]	$\text{def } D \text{ in } (X[\tilde{e}\tilde{k}] \mid Q) \rightarrow \text{def } D \text{ in } (P[\tilde{c}/\tilde{x}] \mid Q) \quad (\tilde{e} \downarrow \tilde{c}, X(\tilde{x}\tilde{k}) = P \in D)$
[SCOP]	$P \rightarrow P' \Rightarrow (\nu u)P \rightarrow (\nu u)P'$
[PAR]	$P \rightarrow P' \Rightarrow P \mid Q \rightarrow P' \mid Q$
[STR]	$P \equiv P' \text{ and } P' \rightarrow Q' \text{ and } Q' \equiv Q \Rightarrow P \rightarrow Q$

配對錯誤

```
![請問還有沒有貨]
▷{ 缺貨 : stop
   有貨 : if 老婆說可以買
          then ◁ 下標
              ![收件資訊]
              ?(貨)
              ![錢]
              stop
          else ◁ 算了
              stop }
```

以為可以貨到付款的買家

```
?(請問還有沒有貨)
if 查了一下有貨
then ◁ 有貨
    ▷{ 算了 : stop
        下標 : ?(收件資訊)
        ?(錢)
        ![貨]
        stop }
    else ◁ 缺貨
        stop
```

賣家

通訊協議 / 訊程型別

「先詢問有無存貨，賣家回覆有貨才可下標。下標後將收件資訊告知賣家，完成匯款後賣家出貨。」

買家通訊規則：

!詢問存貨

&{ 缺貨 : end

有貨 : +{ 下標 : !收件資訊 !錢 ?貨 end

算了 : end } }

! 送

? 收

& 對方所有選項

+ 自己所有選項

保證沒有配對錯誤

![請問還有沒有貨]

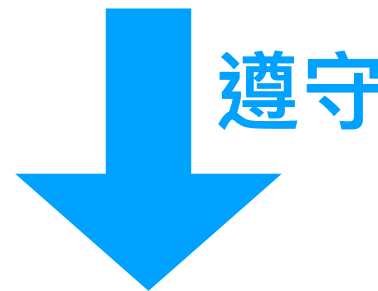
買家動作

▷ { 缺貨 : stop

有貨 : if 老婆說可以買

then ◁ 下標; ![收件資訊] ![錢] ?(貨) stop

else ◁ 算了; stop }



!詢問存貨

買家通訊規則

&{ 缺貨 : end

有貨 : +{ 下標 : !收件資訊 !錢 ?貨 end

算了 : end } }

保證沒有配對錯誤

!詢問存貨

買家通訊規則

&{ 缺貨 : end

有貨 : +{ 下標 : !收件資訊 !錢 ?貨 end

算了 : end } }



?詢問存貨

賣家通訊規則

+{ 缺貨 : end

有貨 : &{ 下標 : ?收件資訊 ?錢 !貨 end

算了 : end } }

保證沒有配對錯誤

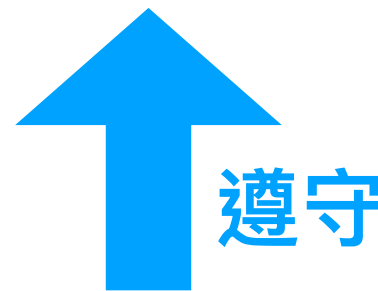
?詢問存貨

賣家通訊規則

+{ 缺貨 : end

有貨 : &{ 下標 : ?收件資訊 ?錢 !貨 end

算了 : end } }



遵守

?(請問還有沒有貨)

賣家動作

if 查了一下有貨

then ◁ 有貨; ▷{ 算了 : stop

下標 : ?(收件資訊) ?(錢) ![貨] stop }

else ◁ 缺貨; stop

型別安全定理

若兩個通訊程式遵守對偶的訊程型別，
則它們一起執行時不可能發生配對錯誤。

$$S ::= \mathbf{nat} \mid \mathbf{bool} \mid \langle \alpha, \bar{\alpha} \rangle \mid s \mid \mu s. S$$

$$\alpha ::= \downarrow[\tilde{S}]; \alpha \mid \downarrow[\alpha]; \beta \mid \&\{l_1: \alpha_1, \dots, l_n: \alpha_n\} \mid \mathbf{1} \mid \perp$$

$$\mid \uparrow[\tilde{S}]; \alpha \mid \uparrow[\alpha]; \beta \mid \oplus\{l_1: \alpha_1, \dots, l_n: \alpha_n\} \mid t \mid \mu t. \alpha$$

$$\overline{\uparrow[\tilde{S}]; \alpha} = \downarrow[\tilde{S}]; \bar{\alpha} \quad \overline{\oplus\{l_i: \alpha_i\}} = \&\{l_i: \bar{\alpha}_i\} \quad \overline{\uparrow[\alpha]; \beta} = \downarrow[\alpha]; \bar{\beta} \quad \overline{\mathbf{1}} = \mathbf{1}$$

$$\overline{\downarrow[\tilde{S}]; \alpha} = \uparrow[\tilde{S}]; \bar{\alpha} \quad \overline{\&\{l_i: \alpha_i\}} = \oplus\{l_i: \bar{\alpha}_i\} \quad \overline{\downarrow[\alpha]; \beta} = \uparrow[\alpha]; \bar{\beta} \quad \overline{t} = t \quad \overline{\mu t. \alpha} = \mu t. \bar{\alpha}$$

We need the following notion: a k -process is a prefixed process with subject k (such as $k![\tilde{e}]; P$ and $\mathbf{catch} k(k') \text{ in } P$). Next, a k -redex is a pair of dual k -processes composed by \mid , i.e. either of forms $(k![\tilde{e}]; P \mid k?(x) \text{ in } Q)$, $(k \triangleleft l; P \mid k \triangleright \{l_1: Q_1 \parallel \dots \parallel l_n: Q_n\})$, or $(\mathbf{throw} k[k']; P \mid \mathbf{catch} k(k') \text{ in } Q)$. Then P is an *error* if $P \equiv \mathbf{def} D \text{ in } (\nu \tilde{u})(P' \mid R)$ where P' is, for some k , the \mid -composition of *either* two k -processes that do not form a k -redex, *or* three or more k -processes.

$$[\text{ACC}] \frac{\Theta; \Gamma \vdash P \triangleright \Delta \cdot k : \alpha}{\Theta; \Gamma, a : \langle \alpha, \bar{\alpha} \rangle \vdash \mathbf{accept} a(k) \text{ in } P \triangleright \Delta} \quad [\text{REQ}] \frac{\Theta; \Gamma \vdash P \triangleright \Delta \cdot k : \bar{\alpha}}{\Theta; \Gamma, a : \langle \alpha, \bar{\alpha} \rangle \vdash \mathbf{request} a(k) \text{ in } P \triangleright \Delta}$$

$$[\text{SEND}] \frac{\Gamma \vdash \tilde{e} \triangleright \tilde{S} \quad \Theta; \Gamma \vdash P \triangleright \Delta \cdot k : \alpha}{\Theta; \Gamma \vdash k![\tilde{e}]; P \triangleright \Delta \cdot k : \uparrow[\tilde{S}]; \alpha} \quad [\text{RCV}] \frac{\Theta; \Gamma \cdot \tilde{x} : \tilde{S} \vdash P \triangleright \Delta \cdot k : \alpha}{\Theta; \Gamma \vdash k?(\tilde{x}) \text{ in } P \triangleright \Delta \cdot k : \downarrow[\tilde{S}]; \alpha}$$

$$[\text{BR}] \frac{\Theta; \Gamma \vdash P_1 \triangleright \Delta \cdot k : \alpha_1 \quad \dots \quad \Theta; \Gamma \vdash P_n \triangleright \Delta \cdot k : \alpha_n}{\Theta; \Gamma \vdash k \triangleright \{l_1: P_1 \parallel \dots \parallel l_n: P_n\} \triangleright \Delta \cdot k : \&\{l_1: \alpha_1, \dots, l_n: \alpha_n\}}$$

$$[\text{SEL}] \frac{\Theta; \Gamma \vdash P \triangleright \Delta \cdot k : \alpha_j}{\Theta; \Gamma \vdash k \triangleleft l_j; P \triangleright \Delta \cdot k : \oplus\{l_1: \alpha_1, \dots, l_n: \alpha_n\}} \quad (1 \leq j \leq n)$$

$$[\text{THR}] \frac{\Theta; \Gamma \vdash P \triangleright \Delta \cdot k : \beta}{\Theta; \Gamma \vdash \mathbf{throw} k[k']; P \triangleright \Delta \cdot k : \uparrow[\alpha]; \beta \cdot k' : \alpha} \quad [\text{CAT}] \frac{\Theta; \Gamma \vdash P \triangleright \Delta \cdot k : \beta \cdot k' : \alpha}{\Theta; \Gamma \vdash \mathbf{catch} k(k') \text{ in } P \triangleright \Delta \cdot k : \downarrow[\alpha]; \beta}$$

$$[\text{CONC}] \frac{\Theta; \Gamma \vdash P \triangleright \Delta \quad \Theta; \Gamma \vdash Q \triangleright \Delta'}{\Theta; \Gamma \vdash P \mid Q \triangleright \Delta \circ \Delta'} (\Delta \asymp \Delta') \quad [\text{IF}] \frac{\Gamma \vdash e \triangleright \mathbf{bool} \quad \Theta; \Gamma \vdash P \triangleright \Delta \quad \Theta; \Gamma \vdash Q \triangleright \Delta}{\Theta; \Gamma \vdash \mathbf{if} e \text{ then } P \text{ else } Q \triangleright \Delta}$$

$$[\text{NRES}] \frac{\Theta; \Gamma \cdot a : S \vdash P \triangleright \Delta}{\Theta; \Gamma \vdash (\nu a) P \triangleright \Delta} \quad [\text{CRES}] \frac{\Theta; \Gamma \vdash P \triangleright \Delta \cdot k : \perp}{\Theta; \Gamma \vdash (\nu k) P \triangleright \Delta} \quad [\text{INACT}] \Theta; \Gamma \vdash \mathbf{inact} \triangleright \Delta$$

$$[\text{VAR}] \frac{\Gamma \vdash \tilde{e} \triangleright \tilde{S}}{\Theta, X : \tilde{S}\tilde{\alpha}; \Gamma \vdash X[\tilde{e}\tilde{k}] \triangleright \Delta \cdot \tilde{k} : \tilde{\alpha}} \quad [\text{DEF}] \frac{\Theta; \Gamma \cdot \tilde{x} : \tilde{S} \vdash P \triangleright \tilde{k} : \tilde{\alpha} \quad \Theta; \Gamma \vdash Q \triangleright \Delta}{\Theta \setminus X; \Gamma \vdash \mathbf{def} X(\tilde{x}\tilde{k}) = P \text{ in } Q \triangleright \Delta} (\Theta(X) = \tilde{S}\tilde{\alpha})$$

易用

解決問題的好工具

解出一個數

問：小明買了 24 隻 9 元的鉛筆和 12 個 15 元的橡皮擦，請問小明總共要付多少錢？

答： $9 \times 24 + 15 \times 12 = 216 + 180 = 396$

$$\begin{aligned} & 15 \times 12 \\ = & (10 + 5) \times (10 + 2) \\ = & 100 + 20 + 50 + 10 \\ = & 180 \end{aligned}$$

$$\begin{array}{r} 15 \\ \times 12 \\ \hline 30 \\ 150 \\ \hline 180 \end{array}$$

解出一個程式

寫下要做的事情

$ordered \cdot perm$

$$\begin{aligned} &\supseteq \{ \text{since } flatten \text{ is a function} \} \\ &\quad ordered \cdot flatten \cdot flatten^\circ \cdot perm \\ &= \{ \text{claim: } ordered \cdot flatten = flatten \cdot inordered \text{ (see below)} \} \\ &\quad flatten \cdot inordered \cdot flatten^\circ \cdot perm \\ &= \{ \text{converses} \} \\ &\quad flatten \cdot (perm \cdot flatten \cdot inordered)^\circ \\ &\supseteq \{ \text{fusion, for an appropriate definition of } split \} \\ &\quad flatten \cdot (nil, split^\circ)^\circ. \end{aligned}$$

符號計算

解出程式

內涵

數學、邏輯學

函數

函數定義

$$f(x) = x^2 + 1$$

函數代入

$$f(5) = 5^2 + 1 = 26$$

無名函數

函數定義

$$\lambda x. x^2 + 1$$

函數代入

$$(\lambda x. x^2 + 1) 5 = 5^2 + 1 = 26$$

λ 算則

$$\Lambda ::= x \mid \lambda x. \Lambda \mid \Lambda \Lambda$$

$$(\lambda x. t) u = t[u/x]$$

$$2 + 3 = 5$$

+

2

3

$$(\lambda m. \lambda n. \lambda f. \lambda x. m \ f \ (n \ f \ x)) \ (\lambda f. \lambda x. f \ (f \ x)) \ (\lambda f. \lambda x. f \ (f \ (f \ x)))$$

$$= (\lambda n. \lambda f. \lambda x. (\lambda f. \lambda x. f \ (f \ x)) \ f \ (n \ f \ x)) \ (\lambda f. \lambda x. f \ (f \ (f \ x)))$$

$$= \lambda f. \lambda x. (\lambda f. \lambda x. f \ (f \ x)) \ f \ ((\lambda f. \lambda x. f \ (f \ (f \ x))) \ f \ x)$$

$$= \lambda f. \lambda x. (\lambda f. \lambda x. f \ (f \ x)) \ f \ ((\lambda x. f \ (f \ (f \ x))) \ x)$$

$$= \lambda f. \lambda x. (\lambda f. \lambda x. f \ (f \ x)) \ f \ (f \ (f \ (f \ x)))$$

$$= \lambda f. \lambda x. (\lambda x. f \ (f \ x)) \ (f \ (f \ (f \ x)))$$

$$= \lambda f. \lambda x. f \ (f \ (f \ (f \ (f \ x))))$$

5

型別

$$\frac{\frac{}{f : \iota \rightarrow \iota, x : \iota \vdash f : \iota \rightarrow \iota} \quad \frac{\frac{}{f : \iota \rightarrow \iota, x : \iota \vdash f : \iota \rightarrow \iota} \quad \frac{}{f : \iota \rightarrow \iota, x : \iota \vdash x : \iota}}{f : \iota \rightarrow \iota, x : \iota \vdash f \ x : \iota}}{f : \iota \rightarrow \iota, x : \iota \vdash f \ (f \ x) : \iota} \quad \frac{}{f : \iota \rightarrow \iota \vdash \lambda x. f \ (f \ x) : \iota \rightarrow \iota} \quad \frac{}{\vdash \lambda f. \lambda x. f \ (f \ x) : (\iota \rightarrow \iota) \rightarrow \iota \rightarrow \iota}$$

型別對應於邏輯

$$\frac{\Gamma \vdash f : A \rightarrow B \quad \Gamma \vdash x : A}{\Gamma \vdash f \ x : B}$$

程式語言學

以**符號**（語言）精準簡練地表達**計算規則**（程式）

厲害：嚴格證明各種「好」性質

易用：作為好工具，協助人思考與解決問題

內涵：常與數學、邏輯學有密切對應