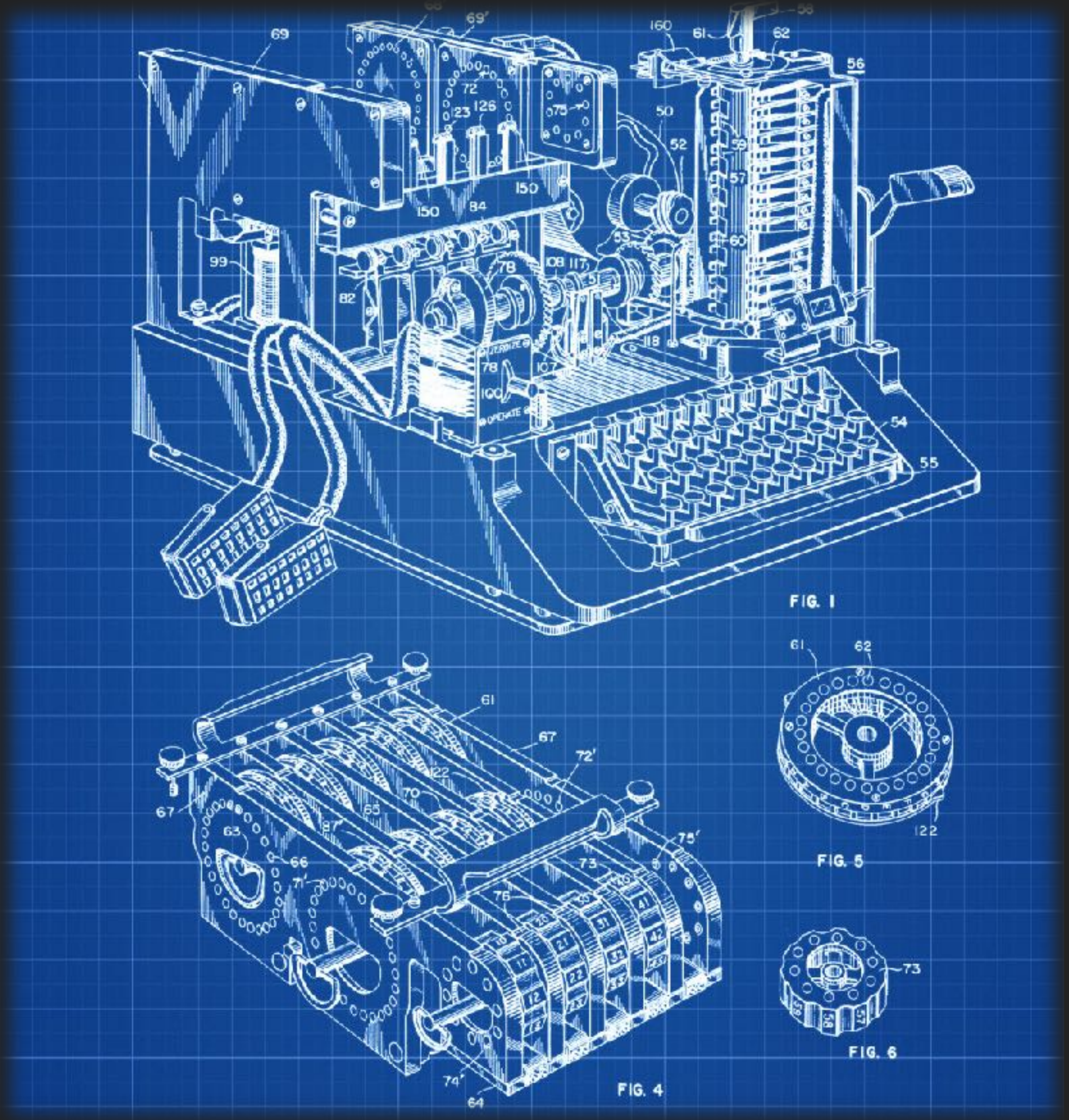
COMPILADORES

MÁQUINAS ABSTRACTAS

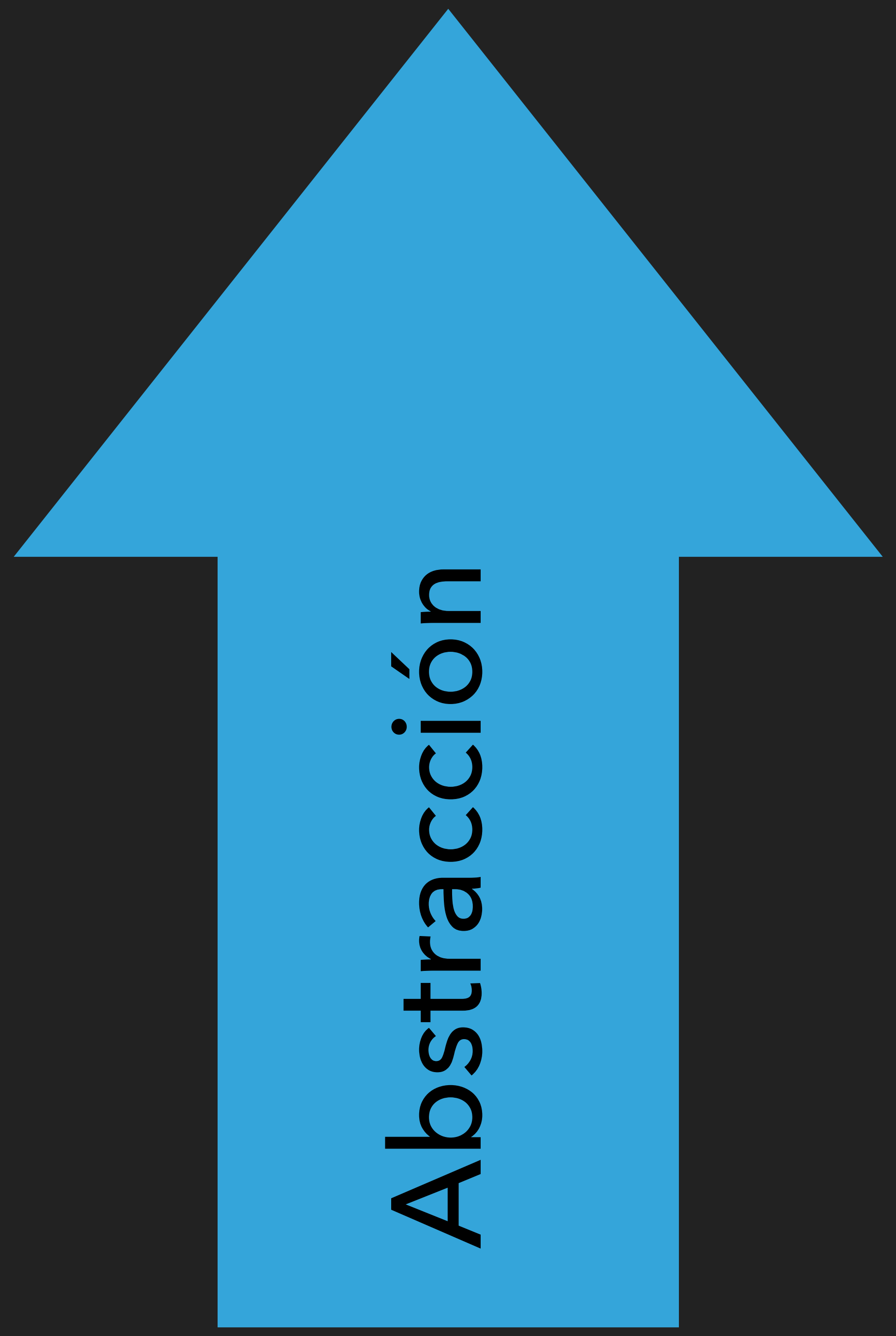
¿QUÉ ES UNA MÁQUINA ABSTRACTA?

- Son **máquinas**
  - Ejecutan programas paso a paso
- Son **abstractas**
  - Omiten detalles de las máquinas reales
- En general se definen por:
  - Un conjunto de **estados** válidos
  - Un conjunto de estados **iniciales** y estados **finales**



NIVELES DE ABSTRACCIÓN

Semántica operacional estructural de paso chico



Diseño para la construcción de una máquina real

¿POR QUÉ USAR MÁQUINAS ABSTRACTAS?

- Especificación de la semántica de programas
  - ¿No alcanza con SOS?
  - Importa la eficiencia de ejecución
- Especificación de la máquina
  - Difícil con una máquina real
- Aparecen nociones de la compilación a máquinas reales

Podemos razonar formalmente acerca de la eficiencia

HACIA UNA MÁQUINA ABSTRACTA

- Consideramos el λ-cálculo

  ```
  t ::= x | λ x. t | t t
  ```

  ```
  v ::= λ x. t
  ```

- Empezamos con una semántica operacional estructural de paso chico
- Refinamos hasta obtener una máquina abstracta con operaciones de bajo nivel
  - Operaciones que se pueden realizar en tiempo constante

SOS PASO CHICO

- Definimos una evaluación CBV
- Reglas de congruencia

$$\frac{t \rightsquigarrow t'}{t\ u \rightsquigarrow t'\ u} \qquad \frac{u \rightsquigarrow u'}{v\ u \rightsquigarrow v\ u'}$$

- Regla de computación

$$(\lambda x\,.\,t)\ v \rightsquigarrow [v/x]t$$

- Un único **redex**: β-reducción

¿CUÁN CHICO ES UN PASO?

- Consideremos el término $id \equiv \lambda x\,.\,x$
- ¿Qué sucede en un paso $id\ ((id\ x)\ y) \rightharpoondown id\ (x\ y)$?

$$\dfrac{\dfrac{(\lambda x\,.\,x)\ x \rightharpoondown x}{(id\ x)\ y \rightharpoondown x\ y}}{id\ ((id\ x)\ y) \rightharpoondown id\ (x\ y)}$$

- Las reglas de congruencia hacen una búsqueda del próximo redex a reducir.
- ¡En un paso pasan demasiadas cosas!

CONTEXTOS DE EVALUACIÓN [FELLEISEN & HEIB 1992]

- Cada término se puede factorizar en el próximo redex, y su contexto de evaluación
  - Un contexto es un término con un agujero
- Contextos de evaluación:

  $E ::= [\,] \mid E\ t \mid v\ E$

- El término $id\ ((id\ x)\ y)$ se puede separar en:
  - Redex: $t = id\ x$
  - Contexto de evaluación $C[\,] = id\ ([\,]\ y)$
- Se tiene que $id\ ((id\ x)\ y) = C[t]$

CONTEXTOS DE EVALUACIÓN [FELLEISEN & HEIB 1992]

- La semántica operacional se puede especificar dando:
  - Contextos de evaluación
  - Reglas de computación
- Para cada término r, existe una **única** factorización $C[t]$.
- Si $r = C[t]$ y $r' = C[t']$ entonces $t \rightsquigarrow t' \quad \Leftrightarrow \quad r \rightsquigarrow r'$

CONTEXTOS COMO LISTAS DE MARCOS

- Un contexto de evaluación es una **lista** que termina en un agujero

$$E ::= [\,] \mid E\ t \mid v\ E$$

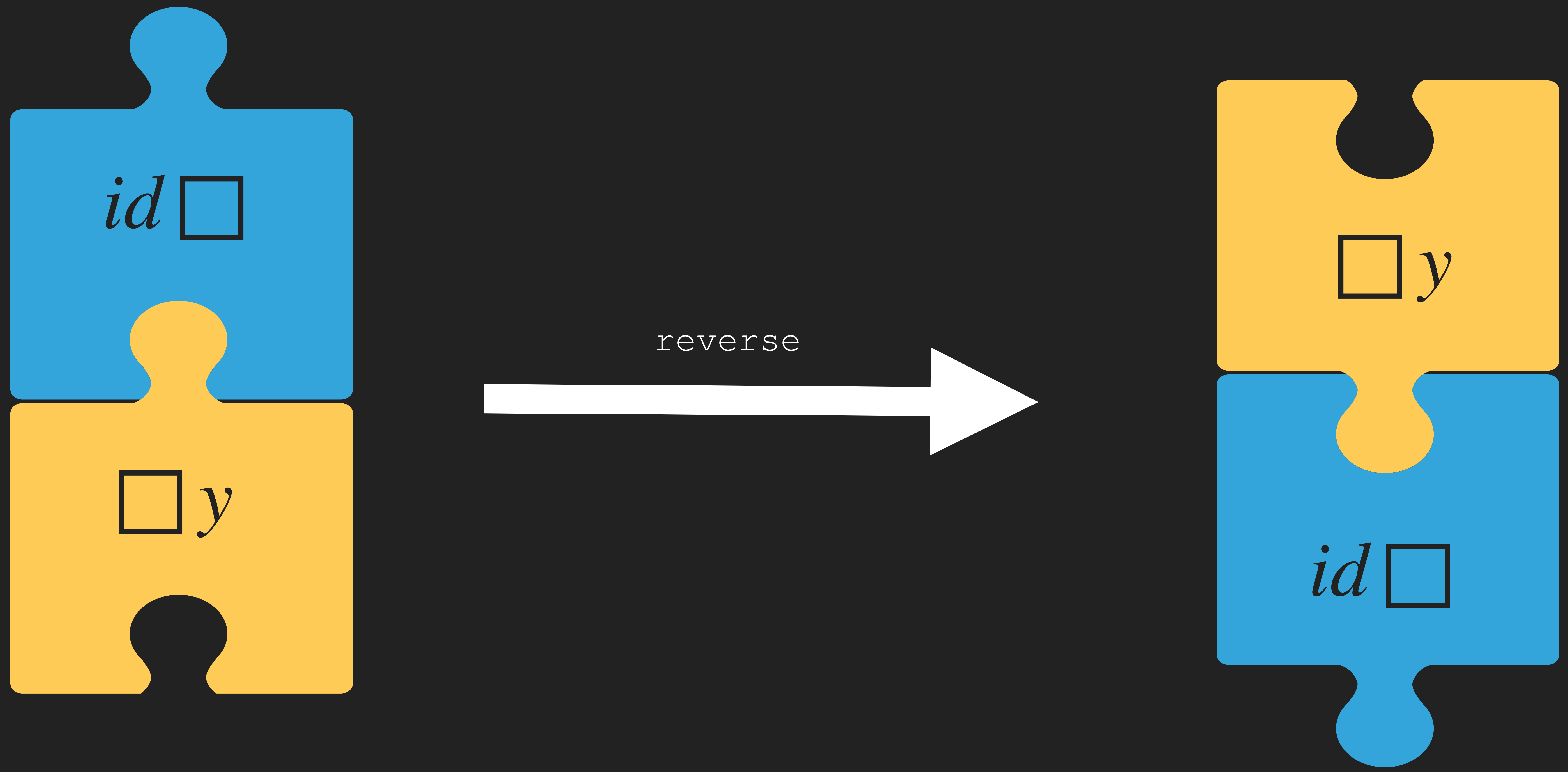
- Por ejemplo, el contexto $id\ ([\,]\ y)$ es la lista

$$id\ \square : \square\ y : [\,]$$

- Cada elemento de la lista construye una parte del contexto y se llama **marco** (frame)

$$fr ::= v\ \square \quad \mid \quad \square\ t$$

CONTEXTOS COMO LISTAS DE MARCOS



MÁQUINA CK

- La máquina **CK** tiene estados que se componen de
  - Control **C**: un término o un valor
  - Continuación **K**: Una pila de marcos (*frame stack*)
- Dos tipos de estado: $\langle t, k \rangle$, y $\langle\!\langle v, k \rangle\!\rangle$
- Estados iniciales: $\langle t, \epsilon \rangle$
- Estados finales: $\langle\!\langle v, \epsilon \rangle\!\rangle$

ϵ es la pila vacía

REGLAS DE TRANSICIÓN

$$
\begin{aligned}
\langle t\ u, k\rangle &\quad\rightarrow\quad \langle t, \square\, u \succ k\rangle \\
\langle v, k\rangle &\quad\rightarrow\quad \langle\!\langle v, k\rangle\!\rangle \\
\\
\langle\!\langle v, \square\, u \succ k\rangle\!\rangle &\quad\rightarrow\quad \langle u, v\, \square \succ k\rangle \\
\langle\!\langle v, (\lambda x\,.\,t)\, \square \succ k\rangle\!\rangle &\quad\rightarrow\quad \langle [v/x]t, k\rangle
\end{aligned}
$$

- Evaluación CBV
- Dos modos de ejecución: búsqueda y reducción
- Mas bajo nivel que SOS. No repite búsquedas. 👍
- Usa substitución. 👎

EJEMPLO

$$
\begin{array}{rcl}
\langle t\ u, k\rangle & \to & \langle t, \Box\, u \succ k\rangle \\
\langle v, k\rangle & \to & \langle\!\langle v, k\rangle\!\rangle \\
\\
\langle\!\langle v, \Box\, u \succ k\rangle\!\rangle & \to & \langle u, v\,\Box \succ k\rangle \\
\langle\!\langle v, (\lambda x\,.\,t)\,\Box \succ k\rangle\!\rangle & \to & \langle [v/x]t, k\rangle
\end{array}
$$

$$
\begin{aligned}
\mathsf{T} &\equiv \lambda f\, t\,.\,t \\
\mathsf{F} &\equiv \lambda f\, t\,.\,f \\
\mathsf{not} &\equiv \lambda b\,.\,b\,\mathsf{T}\,\mathsf{F}
\end{aligned}
$$

$$
\begin{array}{ll}
\langle \mathsf{not}\ \mathsf{T}, \epsilon\rangle & \to \langle \mathsf{T}\,\mathsf{T}, \Box\,\mathsf{F} \succ \epsilon\rangle \\
\to \langle \mathsf{not}, \Box\,\mathsf{T} \succ \epsilon\rangle & \to \langle \mathsf{T}, \Box\,\mathsf{T} \succ \Box\,\mathsf{F} \succ \epsilon\rangle \\
\to \langle\!\langle \mathsf{not}, \Box\,\mathsf{T} \succ \epsilon\rangle\!\rangle & \to \langle\!\langle \mathsf{T}, \Box\,\mathsf{T} \succ \Box\,\mathsf{F} \succ \epsilon\rangle\!\rangle \\
\to \langle \mathsf{T}, \mathsf{not}\,\Box \succ \epsilon\rangle & \to \langle \mathsf{T}, \mathsf{T}\,\Box \succ \Box\,\mathsf{F} \succ \epsilon\rangle \\
\to \langle\!\langle \mathsf{T}, \mathsf{not}\,\Box \succ \epsilon\rangle\!\rangle & \to \langle\!\langle \mathsf{T}, \mathsf{T}\,\Box \succ \Box\,\mathsf{F} \succ \epsilon\rangle\!\rangle \\
\to \langle\ [\mathsf{T}/b]b\,\mathsf{T}\,\mathsf{F} = \mathsf{T}\,\mathsf{T}\,\mathsf{F}\ , \epsilon\rangle & \to \langle [\mathsf{T}/f]\lambda t\,.\,t = \lambda t\,.\,t, \Box\,\mathsf{F} \succ \epsilon\rangle
\end{array}
$$

EJEMPLO (CONT.)

$$\mathsf{T} \equiv \lambda f\, t\,.\, t$$
$$\mathsf{F} \equiv \lambda f\, t\,.\, f$$
$$\mathsf{not} \equiv \lambda b\,.\, b\,\mathsf{T}\,\mathsf{F}$$

$$
\begin{aligned}
& \langle \lambda t\,.\,t, \Box\,\mathsf{F} \succ \epsilon \rangle \\
\to\ & \langle\!\langle \lambda t\,.\,t, \Box\,\mathsf{F} \succ \epsilon \rangle\!\rangle \\
\to\ & \langle \mathsf{F}, (\lambda t\,.\,t)\,\Box \succ \epsilon \rangle \\
\to\ & \langle\!\langle \mathsf{F}, (\lambda t\,.\,t)\,\Box \succ \epsilon \rangle\!\rangle \\
\to\ & \langle [\mathsf{F}/t]t = \mathsf{F}, \epsilon \rangle \\
\to\ & \langle\!\langle \mathsf{F}, \epsilon \rangle\!\rangle \quad \checkmark
\end{aligned}
$$

$$
\begin{aligned}
\langle t\ u, k \rangle &\to \langle t, \Box\, u \succ k \rangle \\
\langle v, k \rangle &\to \langle\!\langle v, k \rangle\!\rangle \\
\langle\!\langle v, \Box\, u \succ k \rangle\!\rangle &\to \langle u, v\,\Box \succ k \rangle \\
\langle\!\langle v, (\lambda x\,.\,t)\,\Box \succ k \rangle\!\rangle &\to \langle [v/x]t, k \rangle
\end{aligned}
$$

DESHACIÉNDONOS DE LA SUBSTITUCIÓN

- En lugar de substituir, llevamos un entorno ρ

$$(\lambda x\,.\,t)\,v\ ;\ \rho \quad \to \quad t\ ;\ \rho[x \mapsto v]$$

- Al ejecutar nos podemos encontrar con una variable, pero le damos valor buscando en el entorno

$$x \quad \to \quad \rho(x)$$

- Hay dos peligros:
  - ⚠️ Que una variable ligada escape a su alcance
  - ⚠️ Que una variable quede libre sin su entorno

VARIABLES LIGADAS ESCAPANDO A SU ALCANCE

Variables libres

Debería evaluar a v

Evalúa a v' ✗

$$(\lambda x\,.\,\lambda y\,.\,y)\ z\ x\ ;\ \rho \quad \to \quad (\lambda y\,.\,y)\ x\ ;\ \rho[x \mapsto z]$$

$x \mapsto v$

$z \mapsto v'$

$x \mapsto z$

$z \mapsto v'$

- Solución: guardamos el entorno ρ en el marco $\square\ x$ para restaurarlo

VARIABLES LIBRES SIN ENTORNO

$$(\mathsf{fun}\ (x : \mathbb{N})\ (\mathsf{fun}\ (f : \mathbb{N} \to \mathbb{N})\ f\ 20)\ (\mathsf{suma}\ x))\ 10$$

$$\to$$

$$\mathsf{suma}\ 10\ 20$$

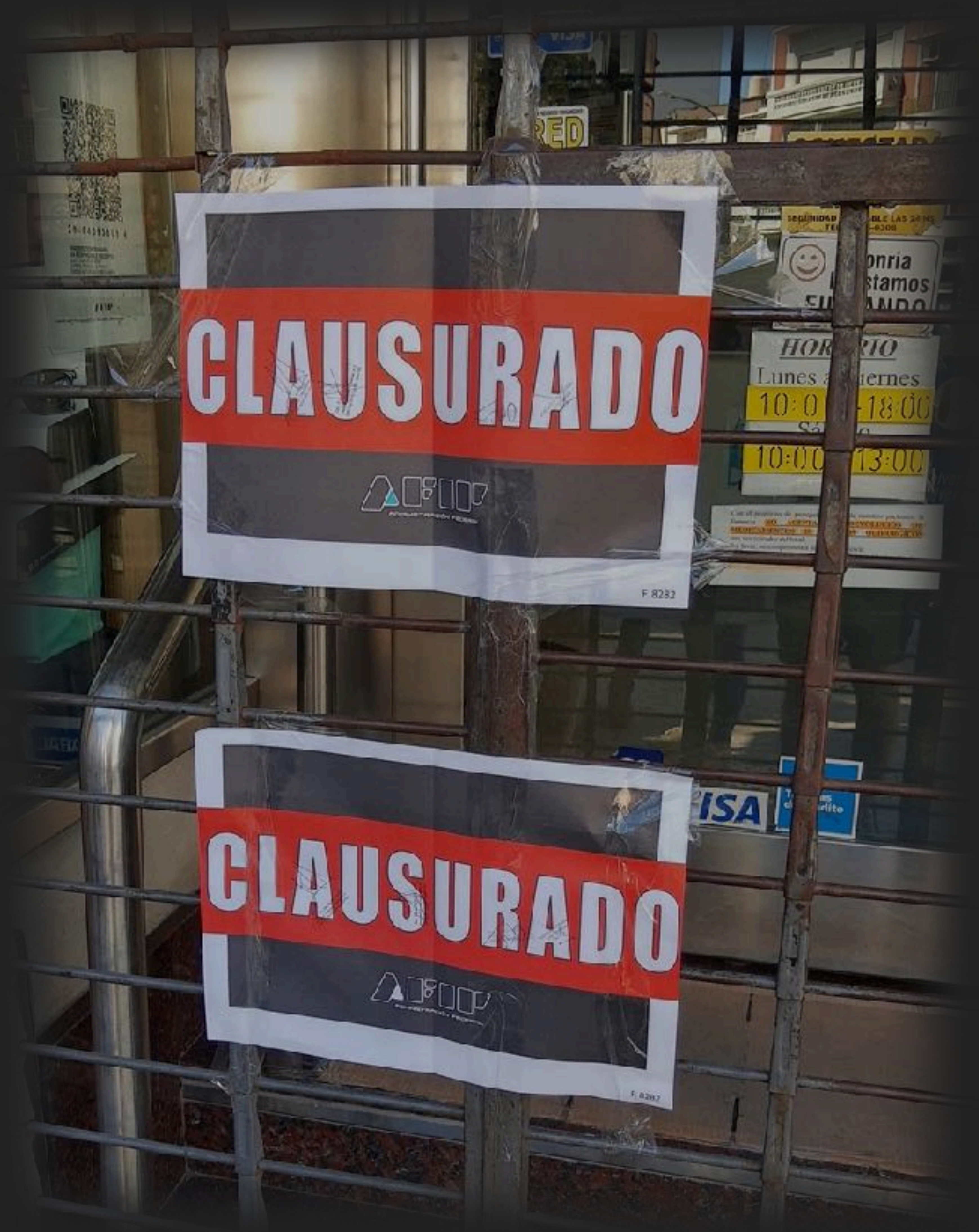
- El valor **suma** x debe llevar el valor asociado a x (10).
- Solución: como valores usamos **clausuras** en lugar de funciones.

CLAUSURAS [LANDIN 1964]

- Una clausura (*closure*) consiste de un término, posiblemente con variables libres, y un entorno que le da significado a esas variables.
- Importante en compilación de diversos lenguajes
- Valores: en lugar de funciones, clausuras.

$$v ::= t^{\rho}$$

- Todo término con variables libres está acompañado de un entorno



MÁQUINA CEK

- Extendemos la máquina CK con un entorno E.
- Los estados son:
  - $\langle t, \rho, k \rangle$: término t, entorno ρ, continuación k.
  - $\langle\langle v, k \rangle\rangle$: valor v, continuación k.
- Los valores son clausuras por lo que no necesitan un entorno.
- Estados iniciales: $\langle t, \varnothing, \epsilon \rangle$, estados finales: $\langle\langle v, \epsilon \rangle\rangle$

MARCOS

- Los marcos se extienden de la siguiente manera:

$$fr ::= v\,\square \quad | \quad \rho \cdot \square\; t$$

- Se guarda el entorno antes de comenzar a evaluar la función.
- Esto permite **restaurar el entorno** a su estado previo luego de evaluada la función.
- Notar que el marco $v\,\square$ también cambió: ahora v es una **clausura**.

TRANSICIONES

$$
\begin{array}{rcl}
\langle t\ u, \rho, k\rangle & \rightarrow & \langle t, \rho, \rho \cdot \square\, u \succ k\rangle \\
\langle x, \rho, k\rangle & \rightarrow & \langle\!\langle \rho(x), k\rangle\!\rangle \\
\langle \lambda x\,.\,t, \rho, k\rangle & \rightarrow & \langle\!\langle (\lambda x\,.\,t)^{\rho}, k\rangle\!\rangle \\
 & & \\
\langle\!\langle v, \rho \cdot \square\, u \succ k\rangle\!\rangle & \rightarrow & \langle u, \rho, v\,\square \succ k\rangle \\
\langle\!\langle v, (\lambda x\,.\,t)^{\rho}\,\square \succ k\rangle\!\rangle & \rightarrow & \langle t, \rho[x \mapsto v], k\rangle
\end{array}
$$

EJEMPLO

$$\begin{array}{rcl}
\langle t\ u, \rho, k\rangle & \to & \langle t, \rho, \rho \cdot \square\, u \succ k\rangle \\
\langle x, \rho, k\rangle & \to & \langle\!\langle \rho(x), k\rangle\!\rangle \\
\langle \lambda x\,.\,t, \rho, k\rangle & \to & \langle\!\langle (\lambda x\,.\,t)^{\rho}, k\rangle\!\rangle \\
\\
\langle\!\langle v, \rho \cdot \square\, u \succ k\rangle\!\rangle & \to & \langle u, \rho, v\,\square \succ k\rangle \\
\langle\!\langle v, (\lambda x\,.\,t)^{\rho}\,\square \succ k\rangle\!\rangle & \to & \langle t, \rho[x \mapsto v], k\rangle
\end{array}$$

$$\boxed{\begin{array}{rl}
\mathsf{T} \equiv & \lambda f\, t\,.\,t \\
\mathsf{F} \equiv & \lambda f\, t\,.\,f \\
\mathsf{not} \equiv & \lambda b\,.\,b\,\mathsf{T}\,\mathsf{F}
\end{array}}$$

$$\begin{array}{rl}
 & \langle \mathsf{not}\ \mathsf{T}, \varnothing, \epsilon\rangle \\
\to & \langle \mathsf{not}, \varnothing, \varnothing \cdot \square\,\mathsf{T} \succ \epsilon\rangle \\
\to & \langle\!\langle \mathsf{not}^{\varnothing}, \varnothing \cdot \square\,\mathsf{T} \succ \epsilon\rangle\!\rangle \\
\to & \langle \mathsf{T}, \varnothing, \mathsf{not}^{\varnothing}\,\square \succ \epsilon\rangle \\
\to & \langle\!\langle \mathsf{T}^{\varnothing}, \mathsf{not}^{\varnothing}\,\square \succ \epsilon\rangle\!\rangle \\
\to & \langle\ b\,\mathsf{T}\,\mathsf{F}, \rho, \epsilon\rangle
\end{array}$$

$$\begin{array}{rl}
\to & \langle b\,\mathsf{T}, \rho, \rho \cdot \square\mathsf{F} \succ \epsilon\rangle \\
\to & \langle b, \rho, \rho \cdot \square\mathsf{T} \succ \rho \cdot \square\mathsf{F} \succ \epsilon\rangle \\
\to & \langle\!\langle \rho(b) = \mathsf{T}^{\varnothing}, \rho \cdot \square\mathsf{T} \succ \rho \cdot \square\mathsf{F} \succ \epsilon\rangle\!\rangle \\
\to & \langle \mathsf{T}, \rho, T^{\varnothing}\,\square \succ \rho \cdot \square\mathsf{F} \succ \epsilon\rangle \\
\to & \langle\!\langle \mathsf{T}^{\rho}, \mathsf{T}^{\varnothing}\,\square \succ \rho \cdot \square\mathsf{F} \succ \epsilon\rangle\!\rangle \\
\to & \langle \lambda t\,.\,t, f \mapsto T^{\rho}, \rho \cdot \square\mathsf{F} \succ \epsilon\rangle
\end{array}$$

donde $\rho = b \mapsto T^{\varnothing}$

EJEMPLO (CONT)

$$\mathsf{T} \equiv \lambda f\, t\,.\, t$$
$$\mathsf{F} \equiv \lambda f\, t\,.\, f$$
$$\mathsf{not} \equiv \lambda b\,.\, b\,\mathsf{T}\,\mathsf{F}$$

$$\langle \lambda t\,.\, t, f \mapsto T^{\rho}, \rho \cdot \square \mathsf{F} \succ \epsilon \rangle$$
$$\to \langle\!\langle (\lambda t\,.\, t)^{(f \mapsto T^{\rho})}, \rho \cdot \square \mathsf{F} \succ \epsilon \rangle\!\rangle$$
$$\to \langle \mathsf{F}, \rho, (\lambda t\,.\, t)^{(f \mapsto T^{\rho})} \square \succ \epsilon \rangle$$
$$\to \langle\!\langle \mathsf{F}^{\rho}, (\lambda t\,.\, t)^{(f \mapsto T^{\rho})} \square \succ \epsilon \rangle\!\rangle$$
$$\to \langle t, (f \mapsto T^{\rho}, t \mapsto \mathsf{F}^{\rho}), \epsilon \rangle$$
$$\to \langle\!\langle \mathsf{F}^{\rho}, \epsilon \rangle\!\rangle \quad \checkmark$$

donde $\rho = b \mapsto T^{\varnothing}$

$$\begin{aligned}
\langle t\ u, \rho, k \rangle &\to \langle t, \rho, \rho \cdot \square u \succ k \rangle \\
\langle x, \rho, k \rangle &\to \langle\!\langle \rho(x), k \rangle\!\rangle \\
\langle \lambda x\,.\, t, \rho, k \rangle &\to \langle\!\langle (\lambda x\,.\, t)^{\rho}, k \rangle\!\rangle \\
\\
\langle\!\langle v, \rho \cdot \square u \succ k \rangle\!\rangle &\to \langle u, \rho, v \square \succ k \rangle \\
\langle\!\langle v, (\lambda x\,.\, t)^{\rho} \square \succ k \rangle\!\rangle &\to \langle t, \rho[x \mapsto v], k \rangle
\end{aligned}$$

USANDO ÍNDICES DE DE BRUIJN

- Podemos usar índices de de Bruijn
- El entorno pasa a ser una lista de valores.
- En lugar de buscar la variable x en el entorno: $\rho(x)$, buscamos el índice i accediendo al índice i de la lista.
- Extender un entorno $\rho[x \mapsto v]$ pasa a ser simplemente agregar v a la cabeza de la lista

OTRAS MÁQUINAS ABSTRACTAS

- Para CBV
  - SECD
  - FAM
  - CAM
  - Zinc
- Para CBN
  - Krivine
  - TIM
- Para Lazy
  - G- Machine
  - STG Machine